МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
имени М.В. Ломоносова
ФАКУЛЬТЕТ ВЫЧИСЛИТЕЛЬНОЙ МАТЕМАТИКИ И КИБЕРНЕТИКИ

ISSN 2411-1473

Современные информационные технологии и ИТ-образование

Научный журнал

Том 2 (№ 11)

Москва
2015

УДК [004:377/378](063)
ББК 74.5(0)я431+74.6(0)я431+32.81(0)я431
С 56

Современные информационные технологии и ИТ-образование. Т. 2 (№ 11), 2015. - 614 с. (ISSN 2411-1473)

В данном выпуске журнала представлены доклады X Юбилейной международной научно-практической конференции «Современные информационные технологии и ИТ-образование», прошедшей в Московском государственном университете имени М.В. Ломоносова 20-22 ноября 2015 года.

Журнал «Современные информационные технологии и ИТ-образование» включен в наукометрическую базу «Российский индекс научного цитирования» с размещением полнотекстовых версий в научной электронной библиотеке eLIBRARY.RU. URL: http://elibrary.ru/title_about.asp?id=52785

Издание осуществлено при финансовой поддержке Российского фонда фундаментальных исследований (Грант РФФИ № 15-07-20760_г)

Учредитель:
Фонд содействия развитию интернет-медиа, ИТ-образования, человеческого потенциала «Лига интернет-медиа»

Издатель:
Фонд содействия развитию интернет-медиа, ИТ-образования, человеческого потенциала «Лига интернет-медиа»

Адрес редакции:
119991, г. Москва, ГСП-1, Ленинские горы, д. 1, стр. 52, факультет ВМК МГУ имени М.В. Ломоносова, каб. 375. E-mail: sukhomlin@mail.ru, тел./факс (495) 939-46-26.

Журнал зарегистрирован Федеральной службой по надзору в сфере связи, информационных технологий и массовых коммуникаций (Роскомнадзор).
Свидетельство о регистрации средства массовой информации ПИ № ФС77-61433 от 10 апреля 2015 г.
Издается с 2005 года. Выходит 1 раз в год.

Редакционная коллегия журнала:

Главный редактор:
Сухомлин В.А. - доктор технических наук, профессор, заведующий лабораторией ОИТ факультета ВМК МГУ имени М.В. Ломоносова, Президент Фонда «Лига интернет-медиа»;

Члены редакционной коллегии:
Веремей Е.И. - доктор физ.-мат. наук, профессор, СПбГУ;
Гергель В.П. - доктор физ.-мат. наук, профессор, ННГУ им. Н.И. Лобачевского;
Самуйлов К.Е. - доктор физ.-мат. наук, профессор, РУДН;
Калиниченко Л.А. - доктор физ.-мат. наук, профессор, вед. н.с. ИПИ РАН ФИЦ ИУ РАН;
Лугачев М.И. - доктор экономических наук, профессор, МГУ имени М.В. Ломоносова;
Любецкий В.А.- доктор физ.-мат. наук, профессор, ИППИ РАН им. А.А. Харкевича;
Нечаев В. В. - доктор технических наук, профессор, МИРЭА;
Посыпкин М.А.- доктор физ.-мат. наук, вед. н. с. ИППИ РАН им. А.А. Харкевича;
Язенин А.В. - доктор физ.-мат. наук, декан факультета ПМиК, профессор, ТвГУ;
Намиот Д.Е. - кандидат физ.-мат. наук, с.н.с. факультета ВМК МГУ имени М.В. Ломоносова;
Зубарева Е.В. - кандидат пед. наук, доцент, н.с. факультета ВМК МГУ имени М.В. Ломоносова;
Сотникова М.В - кандидат физ.-мат. наук, доцент СПбГУ.

Статьи, поступающие в редакцию, рецензируются. За достоверность сведений, изложенных в статьях, ответственность несут авторы публикаций. Мнение редакции может не совпадать с мнением авторов материалов. При перепечатке ссылка на журнал обязательна.
Материалы публикуются в авторской редакции. При перепечатке и цитировании материалов ссылка на журнал «Современные информационные технологии и ИТ-образование» обязательна.

© Современные информационные технологии и ИТ-образование, 2015

Содержание

ИССЛЕДОВАНИЯ И РАЗРАБОТКИ В ОБЛАСТИ НОВЫХ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ И ИХ ПРИЛОЖЕНИЙ....8

Manfred Sneps-Sneppe
Telecommunications and sensors in robotic war....9

Ilyushin E., Namiot D.
Javascript memory management....20

Намиот Д.Е.
Интерфейсы прикладного уровня в SDN....26

Вахов А.Н., Зотова Е.А., Коломоец И.В., Рыжов А.П., Шварц А.Ю.
Платформа Uchi.ru: опыт разработки и перспективы развития....31

Денисов В.С.
Управление настройками Java-приложений с использованием механизма аннотаций....37

Денисов В.С.
Функциональные требования к библиотеке управления настройками Java-приложений....41

Леонов М.В., Киселева Е.А.
Опыт программирования электронных справочников в условиях «планшетной революции» 45

Игнатенков А.В., Ольшанский А.М.
Применение искусственной нейронной сети для построения расписаний процессов на примере графика движения поездов....50

Ромасевич Е.П.
Проблемы миграции современных сетей на протокол IPv6....56

Ромасевич П.В.
Оценка общего объема памяти ввода-вывода телекоммуникационной системы в условиях самоподобного трафика и потери пакетов....61

Емельченков Е.П., Макаров А.И., Мунерман В.И.
Объектно-ориентированный подход к представлению баз данных в не первой нормальной форме....66

Копытов В.В., Шульгин А.О., Федоров С.А.
Разработка архитектуры интеграционной среды кроссплатформенных мобильных приложений с корпоративной информационной системой....71

Кейно П.П.
Предпосылки формирования новой методологии разработки веб-узлов BlockSet и декларативного языка BML....78

Гарфутдинова А.Р., Макаровских Т.А.
Автоматизированная система для организации и проведения конференции....85

Мунерман В.И., Симакова А.А.
Алгебраический подход к формализации семантики задач....92

Орлова Н.В.
Об интеграции технологий дополненной реальности, 3D-туров и 3D-моделирования для обучения IT-специалистов....98

Туракулова А.И.
Возможности нового объектно-ориентированного языка программирования ObjectScript 105

Актаева А.У., Баймуратов О.А.,Галиева Н.Г., ИлипбаеваЛ.Б.,Искоджаева И.
Квантовая информатика: безопасность информации....109

Ярмухаметов Ф.Ф., Кейно П.П.
Практика использования веб-шаблонизаторов в программном комплексе интерпретатора на примере CTPP2....117

Пивнева С.В., Купцов Н.А.
Математическое моделирование процесса обучения и самообучения на основе мультиэвристического подхода....121

Макашов П.Л., Романенко Н.А.
Сервис-ориентированный подход к управлению ИТ проектами на примере использования программного продукта «Jira»....127

Ошурков В.А., Макашова В.Н.
Оперативное планирование производства в MES-системах с использованием методов и алгоритмов искусственного интеллекта....133
Сапегина В.С., Куренев А.В.
Создание эффективной системы лояльности клиентов посредством внедрения платформы TradeLine....140
Соколова А.А.
Разработка системы управления процессом создания программного продукта....146
ТЕОРЕТИЧЕСКАЯ ИНФОРМАТИКА И КОМПЬЮТЕРНЫЕ НАУКИ 150
Головешкин В.А., Жукова Г.Н., Ульянов М.В., Фомичев М.И.
Сравнение ресурсных характеристик традиционного и модифицированного метода ветвей и границ для TSP....151
Абаев П.О., Бесчастный В.А., Гайдамака Ю.В.
О применении пространственных точечных процессов в решении оптимизационных задач для беспроводных сетей с установлением прямых соединений....160
Абаев П.О., Хачко А.В., Бесчастный В.А.
О задаче оптимизации работы сервера установления соединения с механизмом локального контроля перегрузки....166
Самуйлов К.Е., Гайдамака Ю.В., Сопин Э.С., Горбунова А.В.
Алгоритм вычисления преобразования Лапласа-Стилтьеса для времени отклика системы облачных вычислений с гистерезисным управлением....172
Самуйлов К.Е., Гудкова И.А., Острикова Д.Ю.
Анализ вероятности прерывания обслуживания в модели сети LTE с совместным использованием ресурсов....178
Вихрова О.Г., Сопин Э.С.
Анализ показателей качества сети LTE с помощью систем массового обслуживания с ограниченным ресурсом и случайными требованиями....185
Медведева Е.Г., Гайдамака Ю.В.
К анализу параметров качества передачи мультиканального потокового трафика в одноранговой сети....192
Мокров Е.В., Гудкова И.А.
Модель для расчета уровня снижения мощности в сети 3GPP LTE с совместным использованием частот телеметрии аэропорта....199
Белякова Ю. В., Михалкович С. С.
Концепт-параметры как механизм развития средств обобщённого программирования в языке C#....205
Исаченко А.Н., Исаченко Я.А.
О некоторых характеризациях матроидов и свойствах гамильтоновых матроидов....214
Козлов С.В.
Использование соответствия Галуа как инварианта отбора контента при проектировании информационных систем....220
Пеленицын А.М.
Концепты C++17 в их отношении к концептам C++0x....226
Рамиль Альварес Х., Владимирова Ю.С.
Об извлечении квадратного корня в троичной симметричной системе....233
Рамиль Альварес Х.
Алгоритм извлечения квадратного корня из больших целых чисел....239
Фёдоров И.Г.
Синтаксис и семантика исполняемых моделей бизнес-процессов....242
Афанасьевский Л.Б., Горин А.Н., Фадин А.Г.
Реализация аналитической и имитационной моделей системы массового обслуживания..253
НАУЧНОЕ ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ В ОБРАЗОВАНИИ И НАУКЕ 260
Лазовская Т.В., Тархов Д.А.
Использование асимптотических решений при построении нейросетевой модели в жесткой задаче....261
Васильев А.Н., Тархов Д.А., Шемякина Т.А.

Многоуровневые модели окружающей среды в мегаполисах....267
Васильев А.Н., Тархов Д.А., Шемякина Т.А.
Модель неизотермического химического реактора на основе параметрических нейронных сетей. Гибридный метод....271
Васильев А.Н., Тархов Д.А., Шемякина Т.А.
Мезо-уровневая нейросетевая модель загрязнения атмосферного воздуха Санкт-Петербурга по данным мониторинга....279
Зыкина А.В., Запорожец Д.Н.
Пакет прикладных программ для моделирования и решения процессов с использованием аппарата вариационных неравенств....284
Бабичева Т.С.
Транспортные потоки: математическое и имитационное моделирование 290
Идрисова Д.И., Каверзнева Т.Т., Тархов Д.А., Лазовская Т.В.
Моделирование распределения опасного вещества в тупиковом тоннеле с использованием нейросетевого подхода....297
Пишкинас А.О., Оносов И.А., Корчагин С.А., Романчук С.П., Терин Д.В.
Разработка программных средств моделирования композитных наноматериалов....301
Семененко М.Г.
Модель Марковица: математические аспекты и компьютерная реализация....306
Сенчилов В.В.
О решении одной видоизмененной краевой задачи типа Неймана в классах метааналитических в круге функций с применением системы компьютерной математики Maple....310
Скрипачев В.О., Пирхавка А.П., Полушковский Ю.А., Суровцева И.В., Жуков А.О., Яковлев О.В.
Аспекты создания информационной системы для обработки ионосферных данных....316
Юмагулов М.Г., Беликова О.Н., Исанбаева Н.Р.
Моделирование областей устойчивости точек либрации ограниченной задачи трех тел с помощью систем компьютерной математики....321
Лесников С.В., Булыгина Д.С., Лесников А.В., Лесников Г.С.
Конструирование гипертекстового информационно-поискового тезауруса метаязыка лингвистики....326
Понятский В.М., Егоров Д.Б.
Программный комплекс моделирования последовательности методов видеообработки для задач управления....334
Тархов Д.А., Симакина А.А., Суднева А.И.
Обработка данных методом треугольных приближений....341
Тарасенко Ф.Д., Тархов Д.А.
Сравнительный анализ применения различных базисных функций в алгоритмах последовательного сглаживания данных....350
Тархов Д.А., Шаньшин И.К., Шаханов Д.О.
Сравнение нейросетевого и классического подхода к задаче идентификации миграционных процессов....355
Васильев А.Н., Кузнецов Е.Б., Леонов С.С.
Нейросетевой метод идентификации и анализа модели деформирования металлических конструкций в условиях ползучести....360
Бурцев А.А., Сидоров С.А.
Программный комплекс ДССП-ТВМ для структурированного программирования троичной [виртуальной] машины....371
ПАРАЛЛЕЛЬНОЕ И РАСПРЕДЕЛЕННОЕ ПРОГРАММИРОВАНИЕ, ГРИД-ТЕХНОЛОГИИ, ПРОГРАММИРОВАНИЕ НА ГРАФИЧЕСКИХ ПРОЦЕССОРАХ....380
Курочкин И.И., Гуменный Д.Г.
Безопасность сетей SDN. Классификация атак....381
Захаров В.Н., Мунерман В.И.
Параллельный алгоритм умножения многомерных матриц....384
Мунерман В.И., Мунерман Д.В.

Алгебраический подход к построению программно-аппаратных комплексов для повышения эффективности массовой обработки данных.....391
Орлов Ю.В., Посыпкин М.А.
Среда комплексного анализа производительности алгоритмов балансировки в параллельном методе ветвей и границ.....397
Тархов Д. А., Радченко Д.С.
Распределённые алгоритмы оптимизации.....404
Артамонов Ю.С., Востокин С.В.
Применение облачного сервиса Templet Web при проведении лабораторных практикумов на суперкомпьютере «Сергей Королев».....409
Ефимов А.И.
Виртуалтрединг: новая архитектура для вычислительных систем с прямым тонко гранулированным аппаратным мультипрограммированием.....415
Голубева Я.В.
Разработка и исследование алгоритмов балансировки нагрузки в параллельной реализации метода ветвей и границ.....423
ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ В СИСТЕМАХ УПРАВЛЕНИЯ 430
Веремей Е.И.
Математические модели и методы в цифровых технологиях управления движением морских судов.....431
Веремей Е.И., Сотникова М.В.
Алгоритмы оптимизации движения по заданной траектории с учетом прогноза погодных условий.....440
Жгун Т.В., Липатов А.В., Лемешова Д.Д.
Применение метода главных компонент для построения объективных показателей изменения качества системы.....446
Понятский В.М., Грубенко А.Д., Кислинский И.В., Федорищева В.Г.
Использование ИТ на основе ПК Matlab для моделирования канала обмена CAN для систем управления.....456
Понятский В.М., Федорищева В.Г. , Богданова Л.А., Игумнова Т.
ИТ на основе Polyspace Matlab для отладки программного обеспечения встроенных микропроцессоров систем управления 462
Понятский В.М., Яковлев А.Е., Корзунов О.В., Лужинский А.И., Кислинский И.В.
Использование ИТ при командной организации разработки модели системы управления 474
Смирнова М.А., Смирнов М.Н.
Астатическая коррекция цифровых законов управления.....481
Смирнов М.Н., Смирнова М.А.
Вопросы синтеза цифровых законов управления морскими судами с учетом неопределенностей.....487
Жук А.П., Бурмистров В.А., Гавришев А.А.
Система передачи информации с использованием стохастических ортогональных ансамблей дискретных многоуровневых сигналов.....493
Фараонов А.В.
Принятие решений в условиях неопределенности как способ подготовки специалистов транспортной логистики.....499
Джамбеков А.М.
Разработка системы управления процессом каталитического риформинга на базе нечеткой логики.....506
Коваленченко А.С.
К вопросу об оценке эффективности функционирования беспроводных сенсорных сетей. 513
Пусная О.П., Зайцева Т.В., Лихонина А.В.
Автоматизированная система управления спортивными достижениями.....517
Губарев А.В.
Модель угроз системы передачи данных между управляющим программным обеспечением и аппаратным средством.....524
Марценюк М.А., Селетков И.П.

Использование нечёткого клеточного автомата для моделирования динамических процессов в средах с памятью....532
Селезнева О.В.
Сравнение методов прогнозирования траектории морских судов....541
Плужник Е.В., Никульчев Е.В., Лукьянчиков О.В., Ковшов Е.Е.
Программное обеспечение для управления распределенными приложениями в облачных вычислительных инфраструктурах....547
Беляков Д.В.
Задача об автоколебаниях пластинки в потоке среды....552
Коваленко С.Ю., Кондаков С.А.
Комплексы для проведения лабораторных работ....556
Павлов А.В.
Фильтрация и прогноз смещенных нестационарных последовательностей....560
БОЛЬШИЕ ДАННЫЕ И ИХ ПРИЛОЖЕНИЯ....564
Королев С.А., Селиверстов А.В., Зверков О.А., Любецкий В.А.
Классическая аттенюаторная регуляция, зависимая от концентрации триптофана, у актинобактерий....565
Любецкий В.А., Селиверстов А.В.
О решении одной NP-полной задачи....569
Зверков О.А., Селиверстов А.В., Любецкий В.А.
О транскрипционных факторах, кодируемых в пластидах родофитной ветви....571
Гагарин А.П.
Оценка структурной сложности некоторых типовых объектных моделей данных в молекулярной генетики....576
Рубанов Л.И., Селиверстов А.В., Зверков О.А., Любецкий В.А.
Ультраконсервативные элементы у простейших из надтипа Alveolata....581
Рубанов Л.И., Селиверстов А.В., Любецкий В.А.
Широкомасштабный поиск ультраконсервативных элементов в полных геномах....586
Истомина С.Н.
Сравнительный анализ связанных рядов: длин ортологичных белков и их приращений....594
Горбунов К.Ю., Любецкий В.А.
Реконструкция предковых хромосомных структур....600
Королев С.А., Селиверстов А.В., Любецкий В.А.
О трансляции рибосомного белка L16 в пластидах цветковых растений....606
Фролов А.С., Семенов А.С.
Использование Charm++ для решения графовых задач в масштабах экзафлопсных вычислений....608

ИССЛЕДОВАНИЯ И РАЗРАБОТКИ В ОБЛАСТИ НОВЫХ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ И ИХ ПРИЛОЖЕНИЙ

Manfred Sneps-Sneppe[1]

[1] Venspils University College, Ventspils, professor, manfreds.sneps@gmail.com

TELECOMMUNICATIONS AND SENSORS IN ROBOTIC WAR

КЛЮЧЕВЫЕ СЛОВА

Global Information Grid; sensor network; unmanned systems; TCDL; thinking robots; UVDS; Software-Defined Networking.

АННОТАЦИЯ

In this paper, we discuss a "Global Information Grid" as a key concept of military communications and the enabling foundation for network-centric warfare, information superiority, decision superiority, and ultimately full spectrum dominance. We consider sensor environment for robotic war, unmanned systems roadmap and robotic war exercise "Music", as well as telecommunication data link TCDL. Future work list includes military radio communications, thinking robots, unified video dissemination service and software-defined networks.

Introduction

Drums, horns, flags, carrier pigeons, and riders on horseback were some of the early methods the military used to send messages over distances. In nowadays, the key concept is network-centric warfare (NCW) oriented to modern telecommunications and computer networks, oriented to more effective use of existing military forces and more effective interaction between different military powers.

Our goal here is to deliver an insight into telecommunication and sensor aspects in the future robotic war. As one find in Wikipedia, there are six categories of military communications:

- the alert measurement systems,
- cryptography,
- military radio systems,
- nuclear command control,
- the signal corps, and
- network-centric warfare.

Sensors are relates to the alert measurement systems [1]. Sensors are everywhere in today's technology-driven world. There are sensors in traffic lights, vehicles and smartphones. Sensors in military applications gather data that researchers hope will give soldiers the decisive edge. The Army has multiple legacy sensor systems in theater that communicate only with specific systems. This limitation makes sharing actionable information beyond set parameters and costly point-to-point integration task. The arrangement also poses a growing deployment challenge as emerging sensor technologies become more available.

Network-centric warfare (NCW) regards a "Global Information Grid" – a key concept of military communications. GIG was born out of concerns regarding interoperability and end-to-end integration of automated information systems. The GIG provides the enabling foundation for network-centric warfare, information superiority, decision superiority, and what is the most important - ultimately full spectrum dominance (Figure 1). According to Pentagon's plan, the U.S. must have information superiority all over the world: the capability to collect, process, and disseminate an uninterrupted flow of information while exploiting or denying an adversary's ability to do the same.

The information advantage gained through the use of NCW allows a war-fighting force to achieve dramatically improved information positions, in the form of common operational pictures that provide the basis for shared situational awareness and knowledge, and a resulting increase in combat power. The ability to achieve shared situational awareness and knowledge among all elements of a joint force is increasingly viewed as a cornerstone of transformation to achieve future war-fighting capabilities. The success of the GIG will depend in large part on how well it helps achieve fully interoperable forces by connecting today's islands of interoperability to allow force-wide information sharing.

Figure 1. GIG as an Enabling Foundation for Full Spectrum Dominance [2]

Figure 2 shows the future robotic war model. GIG facilities connect three comand and control (C2) points: ground, airborne and maritime. In own order, they collect data from sensor networks and information sources and deliver commands to weapens systems. Department of Defense (DoD) and commercial gateways provide access to military and nonmilitary satellites and to the Defense Information Systems Network (DISN) transport and Internet Protocol (IP) netcentric services, which in turn provide global distribution of mission data and enable long-range C2 of unmanned systems.

The rest of the paper is organized as follows. In Section II, we discuss sensor environment for robotic war, in Sections III and IV -- unmanned systems integrated roadmap 2013-2038 and the robotic war exercise "Music". In Section V, we consider robotic war telecommunication Data Link TCDL, and some Future Works, in Section VI.

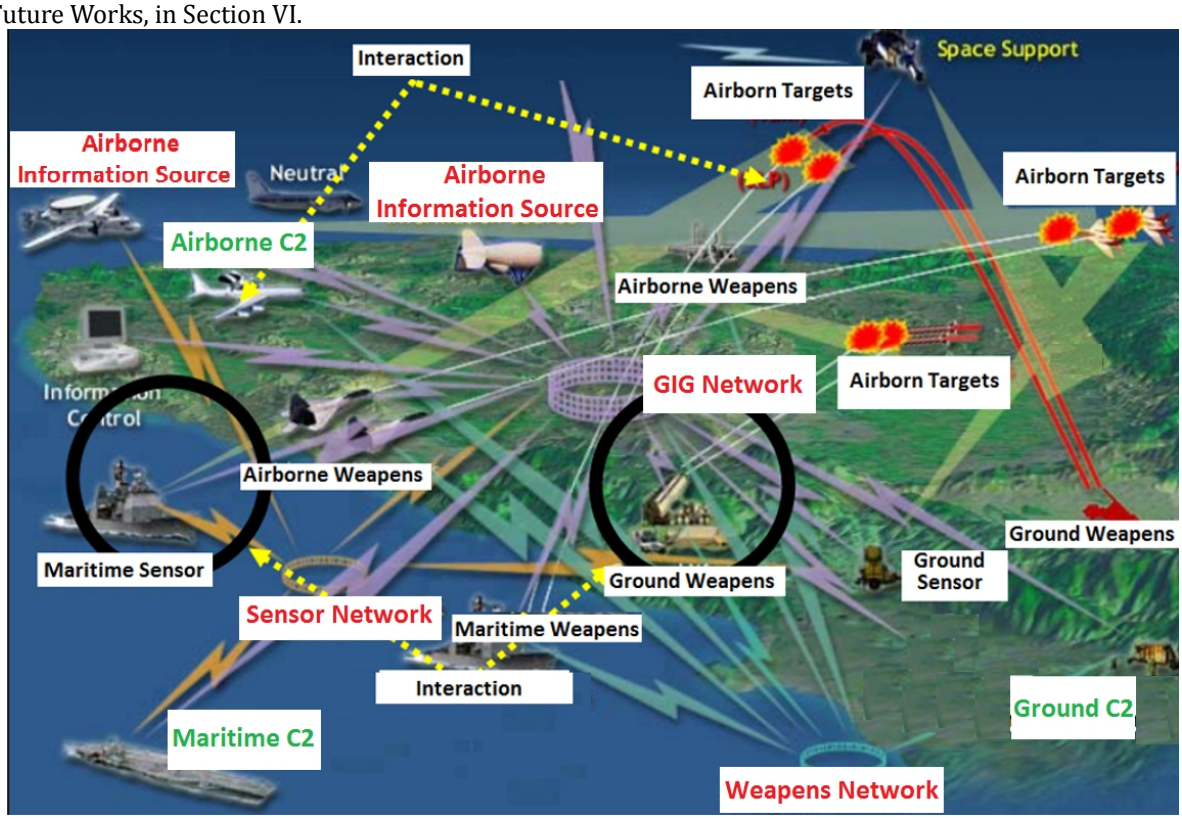


Fig 2 Future Robotic War Model [3]

Sensor Environment for Robotic War

The Integrated Sensor Architecture (ISA) is an interoperability solution that allows for the sharing of information between sensors and systems in a dynamic tactical environment and interaction. The ISA created a Service Oriented Architecture (SOA) that identifies common standards and protocols which support a net-centric system of systems integration. CERDEC NVESD (Communications-Electronics Research, Development and Engineering Center's Night Vision and Electronic Sensors Directorate)

developed ISA under a deployable force protection program, which seeks to provide the critical capabilities needed for a forward operating base to defend itself [4].

Utilizing a common language, these systems are able to connect, publish their needs and capabilities, and interact with other systems even on disadvantaged networks. As Army researchers and engineers develop ISA, they hope to put together fundamental interoperability so that when future sensors come online to a network, they can register and communicate their capabilities; assets and sensors on that network can then subscribe to the types of information they need.

CERDEC has a formal agreement with the Program Executive Office Intelligence, Electronic Warfare and Sensors (PEO IEW&S) under its Sensor Computing Environment program (Sensor CE). Sensor CE is a component of the PEO IEW&S mission, which has a portfolio that covers a broad range of capabilities across the reconnaissance, surveillance and target acquisition spectrum.

The Common Operating Environment (Figure 3) is an approved set of computing technologies and standards that enable secure and interoperable applications to be rapidly developed and executed across a variety of Computing Environments (i.e., Server(s), Client, Mobile, Sensors, and Platform).

Figure 3. Common Operating Environment Architecture [5]

To build up the Common Operating Environment Architecture, many problems should be solved. Point out only few. Seamless integration must exist across the Army Enterprise Network and between computing environments. Control points facilitate the integration of mission environments (Figure 4), and serve as intermediaries between mission environments and the corresponding computing environments. Control points also isolate (or firewall) local standards from the rest of the enterprise.

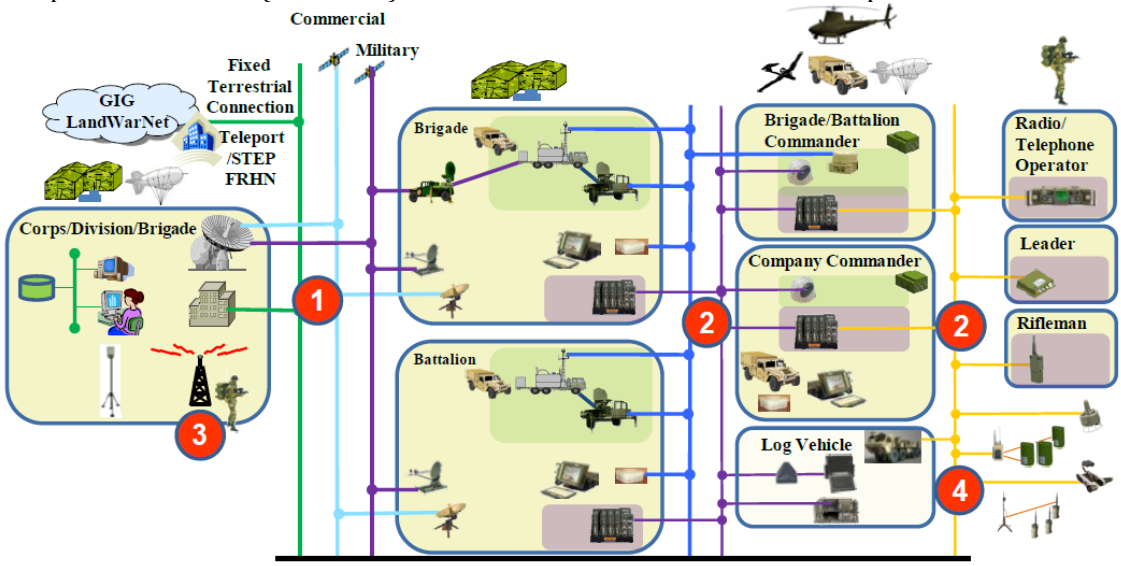


Figure 4 - Tactical army network and its four control points [6]

Control points are placed throughout the Common Operating Environment Architecture [6] to

enable and to enforce:

Interoperability (Logical boundary of mission environments)

Structured Data (e.g., databases, geospatial data, spreadsheets)

Unstructured Data (e.g., documents, presentations, pictures, audio, video)

Security and Gateways (Act as an intermediary for requests from one computing device to Another).

There are four control points shown in Figure 4. Two of them are related to sensors.

Control Point 2 – Enterprise/Command Post to Platform/Soldier/Sensor. Information is flowing between a fixed, somewhat stable network at the command post to individual platforms, Soldiers or sensors. This provides a control point for interface to/from the enterprise standard to a local protocol that is more optimized for disadvantaged networks:

Interoperability – Authentication via PKI, LDAP or Active Directory.

The prescribed interface for data exchange is Mission Command Messaging – Variable Message Format (VMF).

Geospatial data standard is VMF/MIL-STD 2525C.

Security standards will include: network access control – local, varies; external access control – network gateway;

encryption– NSA/NIST-certified solutions; key management - EKMS/KMI-compliant solutions; end-point protection – Host-Based Security System (HBSS); enterprise service management – Remedy/ITSM, IP Management/SPECTRUM); and patch management – manual.

Gateways – The enterprise/command post server is responsible for translation of XML/SOAP to/from VMF.

Control Point 4 – Platform/Soldier to Sensor. Requirements CP4 are identical except gateways – No translation is required.

The more sophisticated tasks relating to ISA should be solved for Unmanned Aircraft Systems. Many UAS interoperability gaps could be pointed out, including between others:

• Provide selectable intelligence, surveillance and reconnaissance (ISR) data in joint approved network formats and waveforms.

• Provide accurate position reporting sufficient for joint common operational picture and joint common air picture applications.

• Provide UAS sensor point and area of interest location information to authorized subscribers in the specified format.

• Provide communication gateway and aerial network or network node services compatible with appropriate joint networks.

• Transmit, relay, or retransmit required voice transmissions or sensor data in accordance with joint standards to authorized DoD and non-DoD subscribers.

• Provide fire support functions that are compatible with joint targeting control systems and procedures.

• Provide meteorological and oceanographic data in common, discoverable, retrievable format to authorized subscribers.

• Provide chemical, biological, radiological, nuclear, and high-yield explosive data in prescribed format to authorized subscribers.

Unmanned Systems as a Basis for Robotic War

The recent document “Unmanned Systems Integrated Roadmap FY2013-2038” [7] continues the path outlined in the 2011 edition of the Roadmap and addresses three unmanned operating domains: air, ground, and maritime. Wherever possible, unmanned systems programs should leverage the DISN core as their baseline terrestrial networking infrastructure for global connectivity. Connection points to the DISN core are already available at DoD gateway sites. The IP networking component of enterprise gateways provides routing and encryption/decryption to multiple security enclaves for access to DISN. Encrypted unmanned systems traffic would be routed through the DoD gateway net-centric convergence router, which provides connectivity between IP modem hubs and DISN.

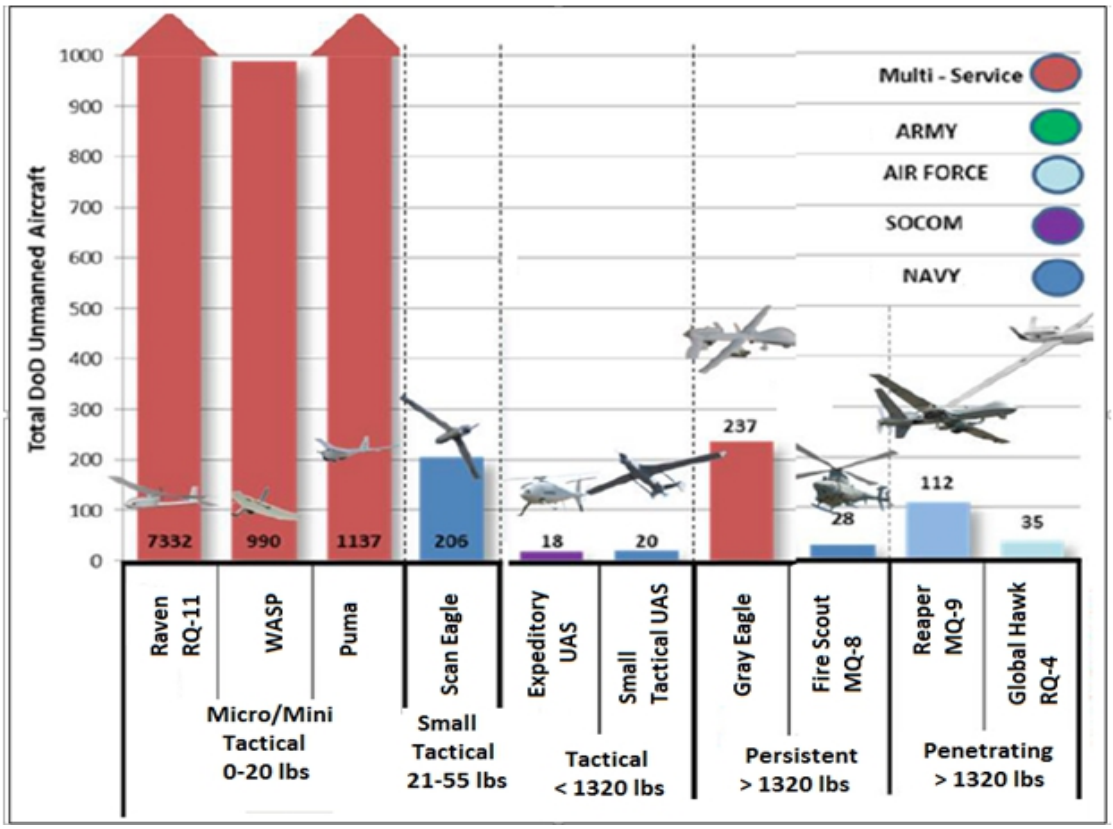


Figure 5. Unmanned aircraft systems (UAS) as of July 1, 2013 [7]

Particular attention is drawn to unmanned aircraft systems (UAS). As of July 1, 2013 (Figure 5), the vast number is covered by micro/mini tactical UAS: more than 90% from the total number around 10 thousand. The trend should be kept in power for the coming 25 years (Figure 6).

Figure 6. Micro/Mini Tactical UAS FY2013-2038 [7]

It is worth to pay attention to micro aerial vehicles [8]. "Micro aerial vehicles" - no larger than a common house fly - are currently being developed by the US military and could enter mass production later this decade. These machines could be used in spying missions, recording and transmitting audio-visual information. An individual robot would serve as a literal "fly on the wall" - equipped with miniature cameras, microphones, modem and GPS. Many terrorist cells could be infiltrated thanks to this radical new technology.

More sophisticated versions might be developed for assassin roles (Figure 7). These would have capsules in the abdomen of the insect, filled with cyanide or another lethal toxin. This would be delivered to the target via a small needle capable of piercing human skin.

However, concerns may be raised as to how this technology affects the safety and security of citizens.

Unmanned Ground Systems (UGS) are a powered physical system with (optionally) no human operator aboard the principal platform, which can act remotely to accomplish assigned tasks (Figure 8). UGS may be mobile or stationary, can be smart learning and self-adaptive, and include all associated supporting components such as operator control units.

Figure 7. Insect-sized assassin drones [8]

Music – the Robotic War Exercise 2011

Interoperability has been a top objective of the Army and the UAS project office for years. Interoperability greatly increases efficiency in Army systems through common interfaces and shared assets In an effort to streamline and coordinate interoperability initiatives across products, the UAS project office hosted its first Manned-Unmanned Systems Integration Capability (MUSIC) exercise in September 2011. The exercise was the largest demonstration of manned-unmanned interoperability ever attempted. The exercise has been in the works for over one and one half years.

Figure 8. Unmanned Ground Systems according to Roadmap 2013-2038 [7]

The MUSIC exercises showcase to the soldier and Army community the progress being made in unmanned interoperability and emerging technologies through common interfaces [9]. There were many objectives to the MUSIC exercise including:

- demonstrating advancements made in manned-to-unmanned teaming (MUM-T);
- demonstrating interoperability among unmanned systems through the Universal Ground Control Station (UGCS), Mini-UGCS (M-UGCS), and the One System Remote Video Terminal (OSRVT); and highlighting open architectural approach that allows multiple control nodes and information access points via the Tactical Common Data Link (TCDL).

Manned-Unmanned Teaming (MUM-T). The concept of MUM-T is to combine the inherent strengths of manned platforms with the strengths of UAS, with product synergy not seen in single platforms. MUM-T combines robotics, sensors, manned/unmanned vehicles, and dismounted soldiers to achieve enhanced situational awareness, greater lethality, improved survivability, and sustainment. Properly designed, MUM-T extends sensor coverage in time and space and provides additional capability

to acquire and engage targets.

The pilot can use the sensor on the UAS, just as a sensor would be used aboard an aircraft, except that the position of the UAS sensor can be up to 80 km ahead from the aircraft. The MUM-T capability provides an unprecedented standoff range from threat weapons and acquisition systems.

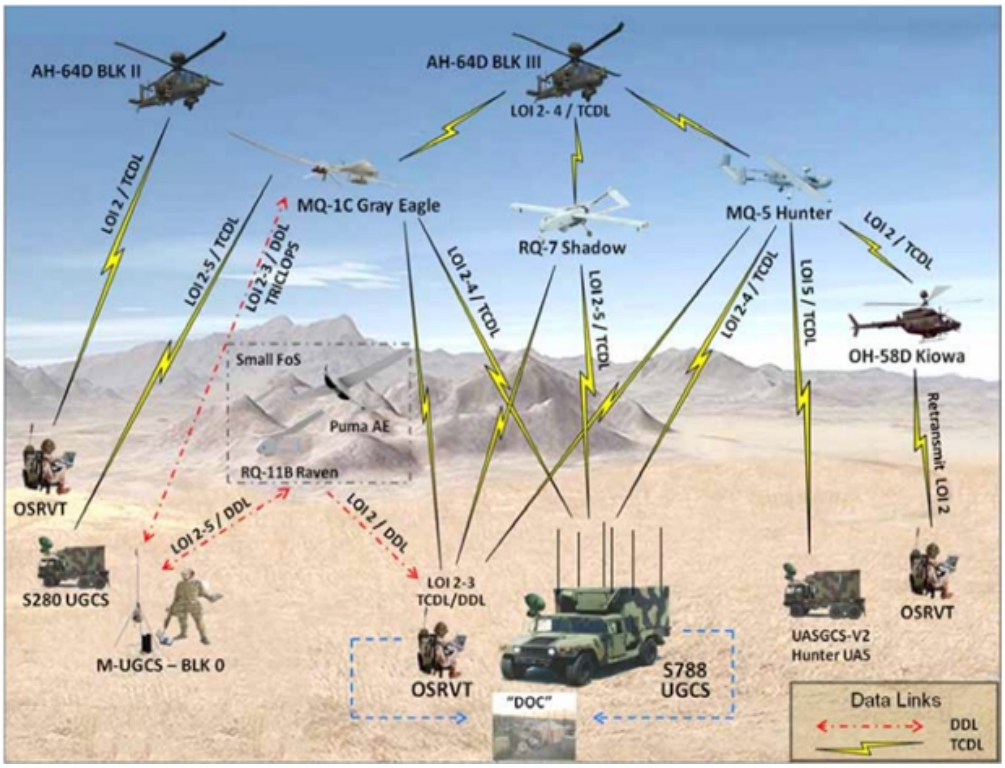


Figure 9. MUSIC Exercise 2011 [9]

UGCS. One major capability that will be demonstrated at the MUSIC exercise will involve the unmanned aircraft Gray Eagle, Shadow, and Hunter platforms, operating off a Universal Ground Control Station (UGCS). The UGCS supports interoperability by providing common hardware and software functionality across the UAS platforms.

OSRVT. The One System RemoteVideo Terminal (OSRVT) will be demonstrating its new bidirectional capability with the larger platforms Gray Eagle, Shadow, and Hunter AVs. The OSRVT will also be demonstrating its ability to receive video from the small aircraft, Raven and Puma, along with the video from the manned platforms Apache and Kiowa.

M-UGCS. Mini-Universal Ground Control Stations (M-UGCS) demonstrate a move toward a common controller for UAS's fleet of small aircraft to include the Raven, Puma, and eventually the Wasp. Current plans are to also demonstrate the M-UGCS control of the electro-optical/infrared (EO/IR) sensors on the Gray Eagle through DDL. This capability known as TRICLOPS, allows for three sensors in the air on a single AV that can be controlled through separate data links increasing range and SA to the soldier.

Data Link TCDL – Robotic War Telecommunication Standard

Two type data links are in use for unmanned aircraft systems: Tactical Common Data Link (DCDL) and Digital Data Link (DDL).

The Tactical Common Data Link (TCDL) is a secure U.S. military communications protocol to send secure data and streaming video links from airborne platforms to ground stations. The TCDL can accept data from many different sources, then encrypt, multiplex, encode, transmit, demultiplex, and route this data at high speeds. It uses a Ku narrowband uplink that is used for both payload and vehicle control, and a wideband downlink for data transfer. The Ku band is the 12–18 GHz portion of the electromagnetic spectrum in the microwave range of frequencies.

The TCDL was designed for UAVs, specifically the MQ-8B Fire Scout, as well as manned non-fighter environments. The TCDL transmits radar, imagery, video, and other sensor information at rates from 1.544 Mbit/s to 10.7 Mbit/s over ranges of 200 km. TCDL will soon support the required higher CDL rates of 45, 137 and 274 Mbps.

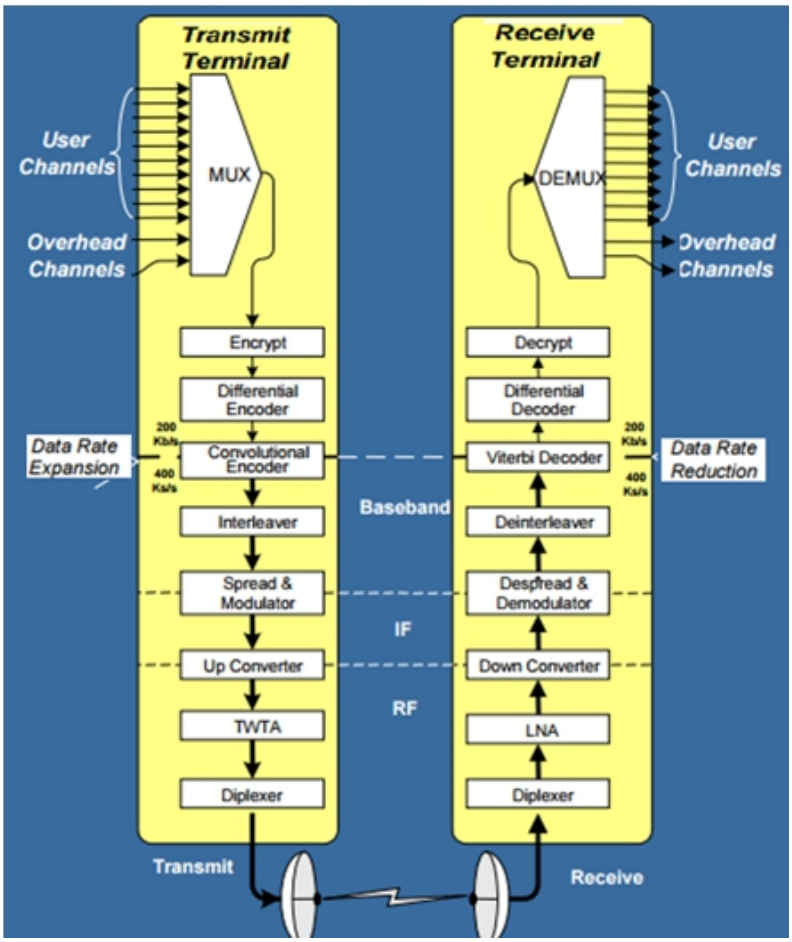


Figure 10. Common Data Link Functions

TCDL Transmitter Functions are the following (Figure 10).

Multiplexer simultaneously provides multiple dedicated channels to users and mixes the data bits from each channel to form one aggregate bit stream. The Multiplexer function is used in the Forward Link (FL) and in all modes of the Return Link (RL).

Encryption is used to protect the data in the event that the data is intercepted, and is performed on the aggregate bit stream by Communications Security (COMSEC) devices. COMSEC devices change (encrypt) the bit stream in such a way that is difficult to reconstruct the original bit stream without a complimentary decryption device. The disadvantage of encryption is that the decryption process adds multiple additional errors for any error encountered. This characteristic is known as error extension.

There are several types of encoders that are used within the CDL system, although differential and convolutional encoders are the two most common. Some types of encoders add redundant bits to the bit stream, which enable the correction of errors that may occur in transmission. The addition of redundant bits expands the data rate.

Differential encoding transforms the digital bit stream by converting space or zero pulses into the transitions that occur between a one and a zero. Since there is no ambiguity of the transmissions coming from the demodulator, the differential decoder can restore the bit stream. This process, however, doubles the bit error rate, because one bit error will affect two transitions.

Convolutional encoding is applied to the data link signal in order to correct bit errors that might occur during transmission and results in coding gain for the system. Decoding is the process that actually corrects transmission errors.

Interleaving is used to inter-mix the bits of the code words generated through convolutional encoding. The motivation for leaving interleaving is to compensate for burst or sequential errors, which can otherwise exceed the capability of the decoder to correct errors.

Direct Sequence Spreading is performed to provide jam resistance against narrow band jammers, which concentrate all power on the signal such as tone and narrow band noise. In addition, it effectively hides the signal spectrum, resulting in low probability of the detection/low probability of intercept signal

qualities.

Modulator converts the aggregate digital bit stream into Radio Frequency (RF) analog signal.

Upconverter/Power Amplifier/Filter. The upconverter translates the 1700 MHZ RF signal to its final X or Ku band frequency where it is amplified to provide the required power to the antenna. Filtering assures that the spectral purity requirements for the allocated frequency are met. The size of the power amplifier is determined through Link Budget analysis and mission requirements. Upconversion, power amplification, and filtering exist in both the FL and all modes of the RL.

The diplexer contains filters which isolate the transmit frequency from the receiver frequency. This makes it possible for the transmitter and receiver to share a common antenna.

Receiver Functions are symmetric one to Transmitter Functions, except Viterbi Decoder. Viterbi decoding is applied to the data link signal to correct errors that may have been encountered through RF transmission. The encoding/decoding process adds what is referred to as coding gain, which may be necessary for the successful data link transmission. Decoding corrects errors before bypassing the aggregate bit stream on to the decryption device. Correction of the errors occurs because the convolutional encoder (or transmit side of the data link) creates code words, which contain data bits with added redundant bits. The redundant bits allow the decoder to detect and correct errors that may exist in each code word. The process of decoding is much more complicated than the encoding process, and limits the speed of the bit stream that can be decoded.

PDDL. The Portable Digital Data Link system (PDDL) was also tesred during MUSIC exercises. PDDL developed by Tadiran Spectralink is a complete solution for digital communication between the ground control station and the unmanned aircraft vehicle. The system consists of a Ground Data Terminal and Aircraft Data Terminal. The PDDL has up to 12 Mbps link rate, serial communication port and Ethernet port. Specifications: Parameter Value Frequency 2.320-2.70 GHz or 2.405-2.407 GHz, Link Rate up to 12 Mbps, Encryption 128 bit AES/ 256 bit AES, Distance up to 100 km range.

Future Work and Discussion

Accurate intelligence about the enemy is always on the military's wish list, and success in future conflicts will require technologies that can perform persistent surveillance to help identify enemies and friendly forces. Robots that can operate autonomously also will be essential tools of war, not necessarily to fire weapons, but to conduct mundane tasks such as delivering cargo.

Anytime Anywhere Communications [10]. High-speed mobile broadband is a military holy grail. Soldiers want the ability to communicate, as well as exploit the capabilities of the latest smartphones. FM radios don't cut it anymore. Troops want the same technology that powers high-speed commercial cellular networks so they can send photos, video and keep track of their unit's location. But there are still some hurdles in achieving that vision. One is the acute shortage of network bandwidth for deployed troops. The Army has been testing deployable 3G broadband networks that troops can set up quickly in a temporary shelter or aboard a military vehicle. The combination of advanced cellular base stations and mesh networking also could give soldiers tens of megabits per second of data throughput in moving vehicles. The Army sooner, rather than later, must bridge the bandwidth deficiency gap that exists today in combat zones.

Experts predict that as 4G networks roll out globally during the next decade, the military will not only benefit from more reliable networks, but it also will be able to use smart radios that can frequency hop and take advantage of unused spectrum. The Army has predicted for some time that smartphones will play an important role in combat, but it is yet unclear just how iPhones and Androids might eventually supplement or replace current combat net radio voice architecture.

Robots That Think for Themselves [10]. Remotely piloted aircraft have supplied troops with overhead surveillance as well as the ability to launch precision-guided bombs. One thing air, land and seaborne robotic systems currently have in common is that there is a human controlling them from a distance. In the next war, military leaders want drones that can operate themselves. For the technology to progress, the machines will have to move away from the current tele-operated model and be able to make decisions on their own.

Autonomy is a key need across every domain where robotic systems operate — land, sea and air. The human brain can function in dynamic environments, reconstruct paths, predict where it needs to go and live adaptively every day. Robots need to be able to do the same and become untethered from their human masters.

Unified Video Dissemination Service [7]. To reduce overhead costs, optimize manpower requirements, and improve data sharing among various services, organizations, and allied partners, unmanned systems data should be consolidated into cloud-enabled enterprise data centers with a

standard infrastructure established to distribute the data to all authorized consumers. This goal includes the Intelligence Community's "big data" cloud computing efforts and DISA's Unified Video Dissemination Service (UVDS) established to support real-time distribution of VMF data to consumers around the globe (Figure 11).

DISA's UVDS provides DoD-wide VMF consolidation, and a robust routing capability for worldwide dissemination of VMF. Installed in DISA's Defense Enterprise Computing Centers, UVDS supports various VMF sources and user communities by providing for the dissemination of black (encrypted) and red (unencrypted) VMF streams via multicast streaming and near-real-time web-based streaming. UVDS implements DoD and industry standards, protocols, and profiles (e.g., SD, HD, H.264, MPEG-2, FLASH) to ensure the greatest level of interoperability among existing systems while taking advantage of existing computing infrastructures associated with DoD's Global Information Grid (GIG) terrestrial connectivity.

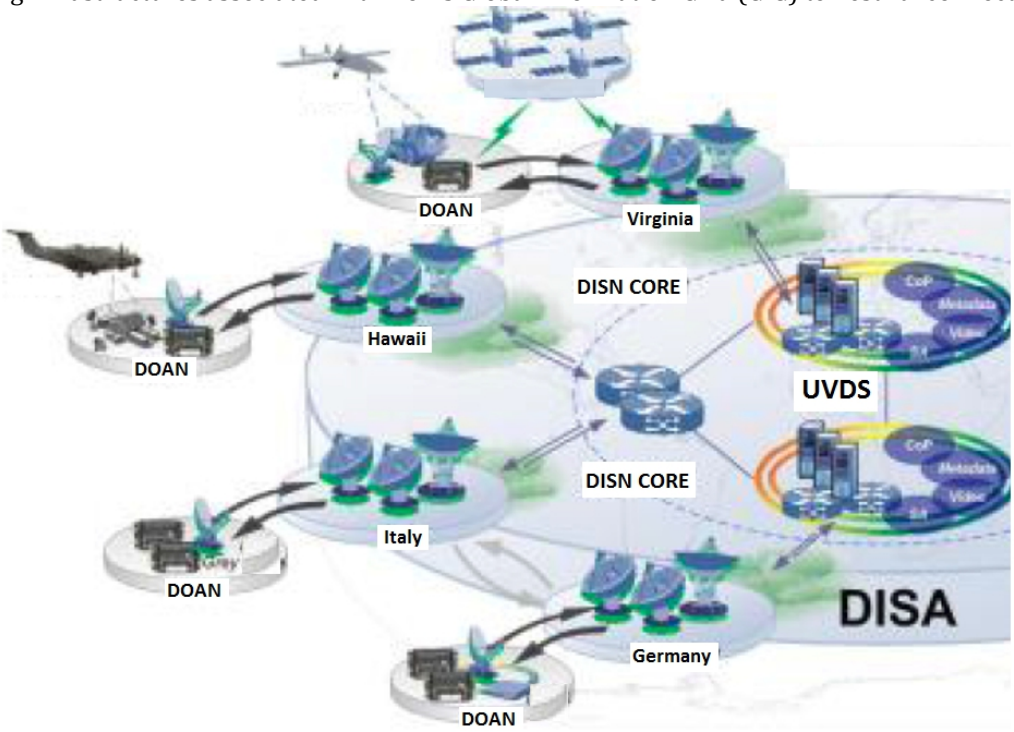


Figure 11. DISA Proposed UVDS Functional Architecture (fragment). DDAN = DISN Deployed Access Node

Software-defined networks [11]. The future of networking will be defined by software. According to the opinion of CERDEC's Space & Terrestrial Communications Directorate, one of the most promising developments in intelligence, surveillance and reconnaissance (ISR) communication is software-defined networks (SDNs), a new approach that unbundles the traditional device-bound, vertically integrated network stack to provide greater network automation, architectural flexibility and programmability for policy-driven control and self-service innovation.

SDN technology works by separating a network's control and forwarding planes, making each easier to optimize. In an SDN, a controller supplies an abstract, centralized view of the overall network, allowing network administrators to quickly and easily make and enact decisions on how underlying systems, such as routers and switches, on the forwarding plane will manage network traffic. With a centralized, programmable network that can automatically and dynamically handle changing requirements, an SDN can deliver increased agility and flexibility. Although SDNs were initially designed for data center environments, they have taken the networking industry by storm due to their cost savings, the ability for customization and operational flexibility they provide, including virtualization, orchestration and automation.

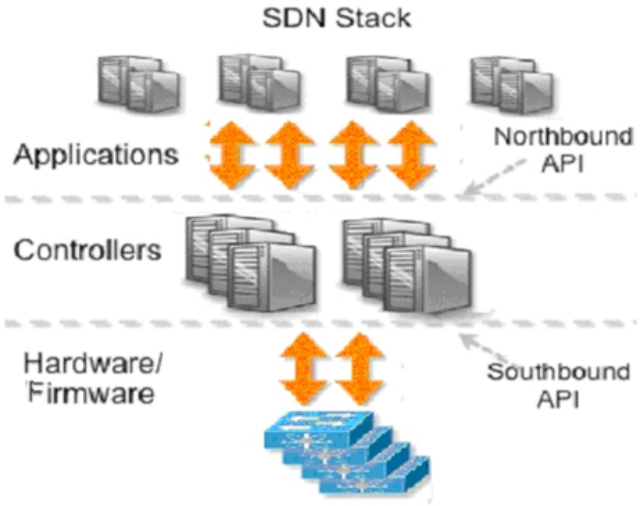


Fig. 12. The classical model for SDN [12]

SDN concept, by the definition, is based on the various programming interface (Figure 12). SDN (and OpenFlow as a key communication protocol) technology is clearly impacting the data center network, including data centers in large carriers [13]. SDN has also the potential to transform the telecom industry by improving the ability of carriers (both wired and wireless) to flexibly deliver bandwidth "on demand." But in the telecom infrastructure market, the way to define SDN is less clear.

The telecom infrastructure as well as GIG consists of a number of unique market segments, including optical networks, carrier-grade routers, wireless edge, wireless core, CPE, etc. It relates to SDN Southbound API and seems rather useful for SDN. Telecom services delivering area is much less clear for SDN success. To deliver services and make money, the transport network must be tied to a complex set of operational support and billing (OSS/BSS) tools to manage the network and bill the customer for services, as well to create the very services. All these features relate to SDN northbound API. By now seems a few papers are on the topic [14,15,16,17].

References

1. Army Technology http://www.defenseinnovationmarketplace.mil/resources/ArmyTechnologyMagazineJan-Feb2015.pdf
2. Global Information Grid. Architectural Vision for a Net-Centric, Service-Oriented DoD Enterprise. Department of Defense. Version 1.0 June 2007
3. SysML http://www.omgsysml.org/INCOSE-OMGSysML-Tutorial-Final-090901.pdf
4. Sensors in Army http://www.army.mil/article/140584/Sensors_move_Army_closer_to_common_environment/
5. (Jan. 7, 2015)
6. SoSECIE http://www.acq.osd.mil/se/webinars/2011-07-12-SoSECIE-Army-COE-Farah-Stapleton-brief.pdf
7. Common Operating Environment Implementation 12 JUL 2011 Appendix C to Guidance for 'End State' ArmyEnterprise Network Architecture U.S. Army CIO/G-6 1 October 2010, DoD's Unmanned Systems Roadmap 2013-2038, December 2013
8. Future timeline http://www.futuretimeline.net/resources/military-war.htm#.Vf_fDt_tlBc
9. Army Mil http://www.army.mil/mobile/article/?p=67838
10. Eric Beidel, Sandra I. Erwin and Stew Magnuson "10 Technologies the U.S. Military Will Need For the Next War " November 2011 http://www.nationaldefensemagazine.org/archive/2011/november/pages/10technologiestheusmilitarywillneedforthenextwar.aspx
11. GGM Whitepaper http://hub.c4isrnet.com/Global/FileLib/GGM_Whitepapers_Editorial/C4-12-11-2014-RedefineISR.pdf
12. The Northbound API- A Big Little Problem http://networkstatic.net/the-northbound-api-2/ Retrieved: Aug, 2015
13. SDN for Telecom http://www.networkworld.com/article/2162090/lan-wan/what-does-sdn-mean-for-telecom-infrastructure-.html
14. Caio Ferreira et al „Towards a Carrier Grade SDN Controller: Integrating OpenFlow With Telecom Services" AICT2014 : The Tenth Advanced International Conference on Telecommunications, 70-75
15. Hampel, G., Steiner, M., & Bu, T. (2013, April). Applying software-defined networking to the telecom domain. In Computer Communications Workshops (INFOCOM WKSHPS), 2013 IEEE Conference on (pp. 133-138).
16. Wang, J. Q., Fu, H., & Cao, C. (2013, August). Software defined networking for telecom operators: Architecture and applications. In 2013 8th Int'l Conf Comm and Net in China (CHINACOM) (pp. 828-833).
17. Sneps-Sneppe M., Namiot D. Metadata in SDN API for WSN //New Technologies, Mobility and Security (NTMS), 2015 7th International Conference on. – IEEE, 2015. – С. 1-5.
18. Sneps-Sneppe M., Namiot D. Micro-service Architecture for Emerging Telecom Applications //International Journal of Open Information Technologies. – 2014. – Т. 2. – №. 11. – С. 34-38.

Ilyushin E.[1], Namiot D.[2]

[1] Lomonosov Moscow State University Ломоносова, Moscow, RKT program, john.ilyushin@gmail.com

[2] Lomonosov Moscow State University, Moscow, senior scientist, dnamiot@gmail.com

JAVASCRIPT MEMORY MANAGEMENT

KEYWORDS

Javascript, memory management, memory leaks.

ABSTRACT

In this paper, we describe the memory management issues in JavaScript applications. Nowadays, JavaScript has become a mainstream programming environment. Modern applications in JavaScript are complex software systems. We can mention here web portals, online games, Internet of Things (Web of Things) applications, and even data mining code. Of course, JavaScript memory management becomes a critical aspect of the development (and deployment) process. In this paper, we discuss the memory leaks patterns in JavaScript code as well as the basic issues behind garbage collection in JavaScript engines.

Introduction

Originally, JavaScript uses garbage collection for automatic memory management. But in the same time, for example, the old conception of web page life cycle (full page refresh) is no more in use. So, in the web programming, we have to deal with long-lived components and the increased complexity. So, JavaScript memory management becomes an even more critical aspect of the development (and deployment) process. The developers will need to understand and deliberately manage the individual lifecycles and memory footprints of the components in their applications. There are different components affecting JavaScript memory distribution. We can mention here DOM Elements (one of the main sources for memory management difficulties), JavaScript Objects as well client-side cache.

Usually, there are two main sources for memory-related issues in JavaScript [2]:

- Orphaned objects;
- Circular references.

Both ways are easily achievable due to “simplicity” of the language. So, it is very important for the developers to understand the background of possible memory-related issues, the associated measurements as well as the solutions which help to avoid them.

Usually, in JavaScript applications, developers do not carry about memory management. Objects could be easily created and reused, where JavaScript engine (its garbage collector) takes care about low-level details. The central concept of JavaScript memory management is a concept of reachability [3]. A distinguished set of objects are assumed to be reachable: these are known as the roots. Typically, these include all the objects referenced from anywhere in the call stack (that is, all local variables and parameters in the functions currently being invoked), and any global variables. Objects are kept in memory while they are accessible from roots through a reference or a chain of references [4]. And there is a Garbage Collector (GC) in the JavaScript engine (in the browser), which cleans memory occupied by unreachable objects [5].

Let us see the following classical example with JavaScript closures [6]. The closure makes all variables of outer functions persist while the inner function is alive. So, suppose our application creates a function and one of its variables contains a large string [4]. While the function inner stays in memory, then the variable data will hang in memory until the inner function is alive. JavaScript engine could have no idea which variables may be required by the inner function, so it keeps everything.

The next classical example is saving JavaScript data in Document Object Model (DOM) [7].

Memory leaks in JavaScript

In this section, we would like to discuss memory leaks patterns in JavaScript. Let us see the details for the above mentioned closure example.

```
function f() {
 var data="Some Large Piece of data . . . ";
 /* do something using data */
function inner() {
```

```
  // ...
 }
 return inner;
}
```

Here the life time for function inner is unknown, so, we have to keep in memory the variable data too.

The Circular references present another classical example of memory leaks.

```
var obj;
function circular_references() {
 obj=document.getElementById("bigdata");
 document.getElementById("bigdata").expandoProperty = obj;
 obj.bigString=new Array(1000).join(new Array(2000).join("XXXXX"));
}
```

In this example, the global variable obj refers to the DOM element bigdata. At the same time, bigdata element refers to the global object through its expandoProperty.

We can see the combination of closures and circular references:

```
function  closureFunction()
{
        var leak =   document.getElementById("element");
        leak.onclick=function innerFunction(){
            alert("Hi! I will leak");};
  leak.bigString = new Array(1000).join(new Array(2000).join("XXXXX")); };
```

Here a JavaScript object leak contains a reference to a DOM object (referenced by the ID "element"). The DOM element, in turn, has a reference to the JavaScript object leak. The resulting circular reference between the JavaScript object and the DOM object causes a memory leak.

One of the most common places associated with memory leaks are setTimeout ()/setInterval () functions.

```
var obj = {
 callMeMaybe: function () {
   var myRef = this;
   var val = setTimeout(function () {
console.log('Time is running out!');
   myRef.callMeMaybe();}, 1000);
   }
};
obj.callMeMaybe();
obj = null;
```

After this section of code, timer still continues to work. An object *obj* isn't cleared, because the closure was transferred to *setTimeout* function and must be maintained for the future performance. In turn, it holds a reference to the life safety as it contains *myRef*. This would be the same if we handed the closure of any other function while retaining the link.

A rule of thumb for all JavaScript applications is obvious. Developers should avoid holding references to DOM elements they no longer need to use, unbind unneeded event listeners and analyze all use cases when storing large chunks of data they are not going to use.

Memory usage measurements

Of course, we need some metrics for memory management. In this section, we would like to discuss memory leaks detection and profiling. There are two main instruments: Google's Chrome Developer [8] and Mozilla Developer [9].

In Chrome Developer Tools, Timeline memory view and Chrome task manager can help developers identify if they are using too much memory. Memory view can track the number of live DOM nodes, documents, and JS event listeners in the inspected render process. The Object allocation tracker can help narrow down leaks by looking at JS object allocation in real-time. Developers can also use the heap profiler to take JS heap snapshots, analyze memory graphs and compare snapshots to discover what objects are not being cleaned up by garbage collection. It is illustrated in Figure 1.

And Figure 2 presents an example of the report. Red nodes (which have a darker background) do not have direct references from JavaScript to them, but are alive because they're part of a detached DOM tree. There may be a node in the tree referenced from JavaScript (maybe as a closure or variable) but is coincidentally preventing the entire DOM tree from being garbage collected.

Yellow nodes (with a yellow background) however do have direct references from JavaScript.

Firefox tool for Garbage Collection measurements looks similar (Figure 3).

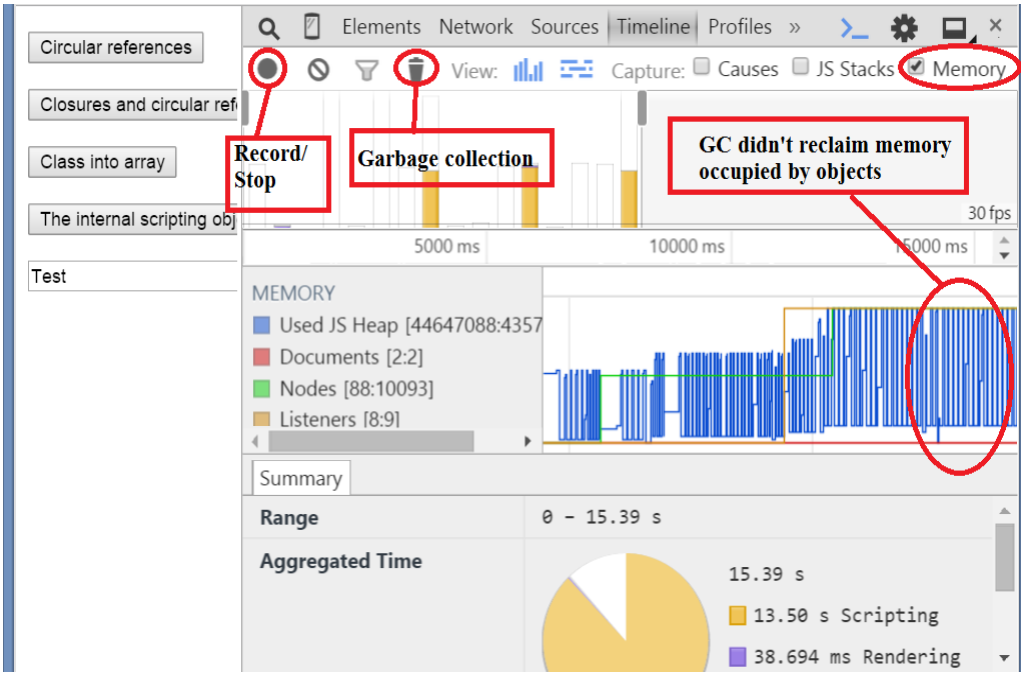


Fig. 1 Chrome Developer Tools [10]

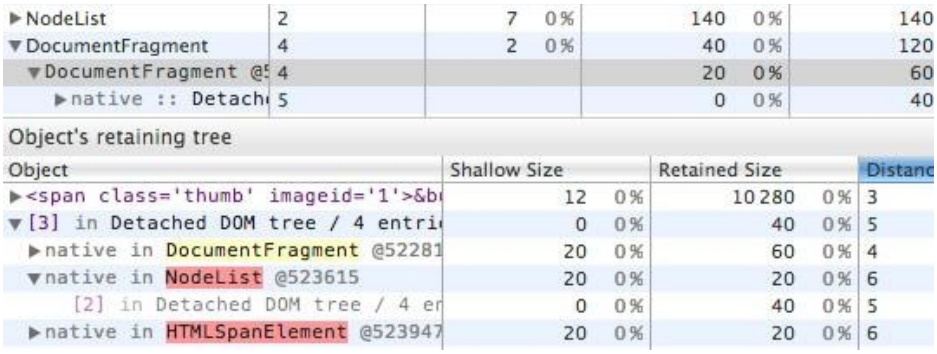


Fig. 2. Google Chrome Developer Report [10]

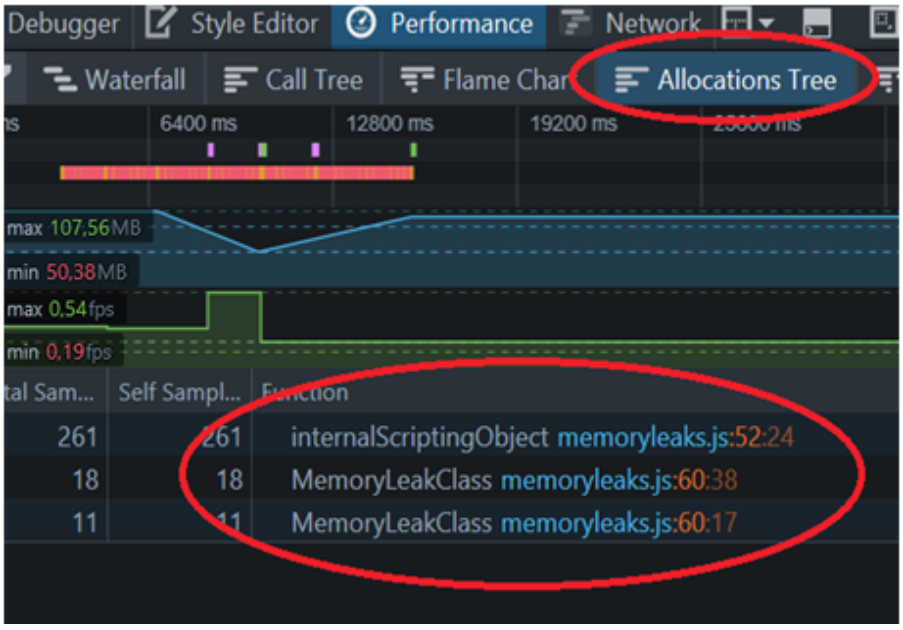


Fig. 3. Firefox memory leaks measurements [2]

Actually, Mozilla provides a list of tools for memory-related measurements [11]. Fox example, Firefox's about:memory page presents fine-grained measurements of memory usage. It is illustrated in Figure 4:

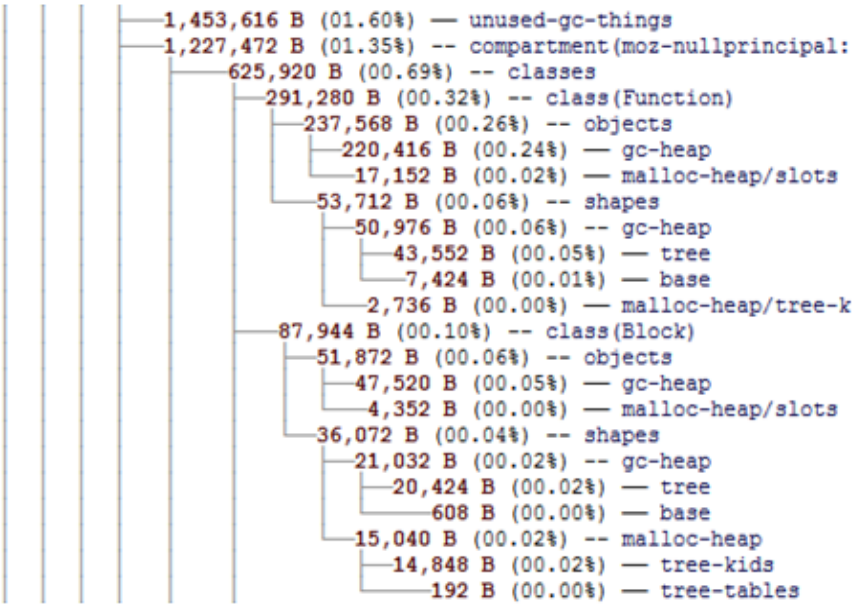


Fig 4. Memory usage in Firefox

Actually, there is a wide list of memory-related measurement tools for Firefox. On practice, most of them, probably, are unknown for developers. The above-mentioned about:memory is the easiest-to-use tool for measuring memory usage. It also lets developers do other memory-related operations like trigger GC and CC, dump GC & CC logs. This tool has got also a special "explanation" reporting – DMD. MD is a tool that identifies shortcomings in about:memory's measurements. The full list of tools (Bloatview, Refcount, GC logs, etc.) is provided in [11]. In the same time, as our experience confirms, many of the tools mentioned on page [11] are either obsolete or not supported anymore.

Garbage Collection in JavaScript

The basic algorithms for Garbage Collectors are well known and widely presented in academic papers [12-13]. Let is review some of the popular choices.

The reference counting algorithm is one of most transparent. An object is considered to be garbage when no references to that object exist. A simple expedient is to keep track in each object of the total number of references to that object. So, the implementation should add a special field to each object called a reference count. Also, every time one reference is assigned to another, the reference counts must be adjusted as above. This increases significantly the time taken by assignment statements.

With reference counting, the garbage (unused data) is easily identified. When it becomes necessary to reclaim the storage from unused objects, the garbage collector needs only to examine the reference count fields of all the objects that have been created by the program. If the reference count is zero, the object is garbage. But in the same time, depends on memory allocation scheme, we will still face fragmentation problems.

As far as we understand, at this moment JavaScript engines do not use reference counting in garbage collection. It is used in PHP, for example [14]. Note, that academic papers describe some high-speed reference counting garbage collectors [15-16].

The classical problems for reference counting are so-called circular references. Nevertheless, the reference counting is an extremely useful technique for dealing with simple objects that don't refer to other objects, such as Strings. So, by our opinion, it is an open question: can we use reference counting as a part of the whole garbage collecting process? E.g., we can use it for String only (there are no circular references). Actually, for JavaScript, String objects are most used in practical applications.

The mark-and-sweep algorithm was the first garbage collection algorithm to be developed for processing cyclic data structures. Variations of the mark-and-sweep algorithm continue to be among the most commonly used garbage collection techniques. At this moment, it is utility (commodity) stuff, deployed as a part of many garbage collectors. The mark-and-sweep algorithm is called a tracing garbage collector because is traces out the entire collection of objects that are directly or indirectly accessible by

the program. The objects that a program can access directly are those objects, which are referenced by local variables on the processor stack as well as by any static variables that refer to objects. In the context of garbage collection, these variables are called the roots. An object is indirectly accessible if it is referenced by a field in some other (directly or indirectly) accessible object. Any accessible object is considered to be live. Any other objects are garbage.

The mark-and-sweep algorithm consists of two phases. In the first phase (mark), it finds and marks all accessible objects. During the second phase (sweep), the algorithm walks through the list of objects and reclaims all the dead objects. The mark-and-sweep algorithm can correctly proceed cycled references. Also, we do not need to introduce additional fields (like reference counter) for our data. The main disadvantage of the mark-and-sweep approach is the fact that that normal program execution is suspended while the garbage collection algorithm runs. It is so-called stop-the-world garbage collector.

The copying garbage collector (sometimes is called stop-and-copy and semi-space collector) starts from a set of roots and traverse all of the reachable memory-allocated objects, copying them from one-half of memory into the other half. The area of memory that we copy from is called old space and the area of memory that we copy to is called new space. When we copy the reachable data, we compact it so that it is in a contiguous chunk. This procedure lets us avoid memory fragmentation.

The mark-compact GC is some combination of copying and mark-and-sweep [17]. In the first phase (mark), it finds and marks all live objects. In the second phase (compact), the garbage collection algorithm compresses the heap by moving all the live objects into contiguous memory locations.

The generational garbage collector is based on the idea of partitioning of live objects. This partitioning procedure is based on time of memory allocation. We assume that most objects are discarded shortly after being used. So, we can deploy different GC policies to different partitions. The policy depends on objects' age. Many generational GC's are not comprehensive - they don't successfully remove all the garbage (long-lived garbage, in particular, may never get collected) [18].

All the garbage collectors in JavaScript engines we know are generational.

Interesting that all the above-mentioned implementations are stop-the-world. A sub-collector (an algorithm for garbage collecting within the generation) is still mark-and-sweep. Of course, there are many improvements for the classical mark-and-sweep. We can mention, for example, a lazy sweeping [19]. Mark-region improves the mark-sweep by dividing the heap in several regions and compacts objects to one end of the regions, and can thus reduce memory fragmentation [20]. Garbage-First (G1) works in per-region manner, marks objects and then evacuates them from current regions to new ones so that current regions can be reclaimed as a whole [21]. It is a garbage collector in Oracle JDK [22].

Mark-split removes the sweep phase from mark-sweep, and thus achieves a time complexity proportional to the size of the live data set. However, this comes with an overhead cost of maintaining a set of free memory intervals. The number of free intervals is much smaller than the number of live objects because some live objects reside adjacent to each other. It seems beneficial, in certain situations, to avoid the sweep phase at the cost of this overhead, which depends on the distribution of live objects and also highly on the data structure selected to store the free intervals. The data structure should preferably provide search for an interval at sub-linear cost, e.g., binary search trees, splay trees, or skip-lists [23].

By our opinion, there are at least two most interesting questions. At the first hand, it is not clear at this moment, why JavaScript engines do not use non-stop-the-world garbage collectors. There are concurrent and parallel implementations for mark-and-sweep, for example. They have been tested with Java, for example [24]. But we have not seen yet such implementations in connection with JavaScript. As seems to us, such a movement would be in line with the common trend to add concurrency into JavaScript [25].

The second interesting moment is the policy for running the garbage collector. Actually, it should be more complex than a simply timer-based event or percentage of free memory. It is especially true for the mobile web with relatively limited resources on mobile phones. Just think about the stop-the-world action in the middle of filling some form on the screen. In the same time, a quick stop action could be almost "invisible" during the AJAX request, when a user is waiting for the response anyway. In other words, the policy for running garbage collector should be based on the behavior and depends on the application.

Литература

1. Wagner, G., Gal, A., Wimmer, C., Eich, B., & Franz, M. (2011, June). Compartmental memory management in a modern web browser. In ACM SIGPLAN Notices (Vol. 46, No. 11, pp. 119-128). ACM.
2. Ilyushin E., Namiot D. On JavaScript Memory Leaks //International Journal of Open Information Technologies. – 2015. – Т. 3. – №. 7.
3. Pienaar, J. A., & Hundt, R. (2013, February). JSWhiz: Static analysis for JavaScript memory leaks. In Code Generation and

Optimization (CGO), 2013 IEEE/ACM International Symposium on (pp. 1-11). IEEE.
4. JavaScript memory leaks http://javascript.info/tutorial/memory-leaks Retrieved: Jun, 2015.
5. Maffeis, Sergio, John C. Mitchell, and Ankur Taly. "An operational semantics for JavaScript." Programming languages and systems. Springer Berlin Heidelberg, 2008. 307-325.
6. CHEN, Y., & ZHOU, X. (2011). JavaScript Closures Research & Typical Applications. Computer Programming Skills & Maintenance, 10, 012.
7. Heilmann, C. (2006). Beginning JavaScript with DOM scripting and Ajax: from novice to professional. Apress.
8. Chrome Developer tools https://developer.chrome.com/devtools Retrieved: Jun, 2015
9. Mozilla Developer https://developer.mozilla.org/en-US/docs/Tools/Performance/Waterfall#Garbage_collection Retrieved: Jun, 2015
10. Chrome Developer Memory Profiling https://developer.chrome.com/devtools/docs/javascript-memory-profiling Retrieved: Jun, 2015
11. Mozilla Performance https://developer.mozilla.org/en-US/docs/Mozilla/Performance
12. R. Jones and R. Lins. Garbage Collection: Algorithms for Automatic Storage Management. John Wiley & Sons Ltd, 1996.
13. Bruno R. Preiss. (2000). Data Structures and Algorithms with Object-Oriented Design Patterns in Java. John Wiley & Sons Incorporated.
14. Suzumura, T., Trent, S., Tatsubori, M., Tozawa, A., & Onodera, T. (2008, September). Performance comparison of web service engines in php, java and c. In Web Services, 2008. ICWS'08. IEEE International Conference on (pp. 385-392). IEEE.
15. Levanoni, Y., & Petrank, E. (2006). An on-the-fly reference-counting garbage collector for java. ACM Transactions on Programming Languages and Systems (TOPLAS), 28(1), 1-69.
16. Blackburn, S. M., & McKinley, K. S. (2003, October). Ulterior reference counting: Fast garbage collection without a long wait. In ACM SIGPLAN Notices (Vol. 38, No. 11, pp. 344-358). ACM.
17. Jones, R., Hosking, A., & Moss, E. (2011). The garbage collection handbook: the art of automatic memory management. Chapman & Hall/CRC.
18. Tauro, C. J., Prabhu, M. V., & Saldanha, V. J. (2012). CMS and G1 Collector in Java 7 Hotspot: Overview, Comparisons and Performance Metrics. Memory, 43(11).
19. Hughes, R.J.M.: A semi-incremental garbage collection algorithm. Software: Practice and Experience 12(11), 1081–1082 (1982)
20. Blackburn,S.M., McKinley,K.S.:Immix:Amark-region garbage collector with space efficiency, fast collection, and mutator performance. SIGPLAN Not. 43(6), 22–32 (2008)
21. Detlefs, D., Flood, C., Heller, S., Printezis, T.: Garbage-first garbage collection. In: Proceedings of the 4th ISMM, pp. 37–48. ACM (2004)
22. Garbage First in Oracle http://docs.oracle.com/javase/7/docs/technotes/guides/vm/G1.html
23. Sagonas, K., Wilhelmsson, J.: Mark and split. In: Proceedings of the 5th International Symposium on Memory Management, ISMM 2006, pp. 29–39. ACM (2006)
24. Nguyen, Nhan, Philippas Tsigas, and Håkan Sundell. "ParMarkSplit: A Parallel Mark-Split Garbage Collector Based on a Lock-Free Skip-List." Principles of Distributed Systems. Springer International Publishing, 2014. 372-387.
25. Namiot D., Sukhomlin V. JavaScript Concurrency Models //International Journal of Open Information Technologies. – 2015. – T. 3. – №. 6. – C. 21-24.

Намиот Д.Е.[1]

[1] Московский Государственный Университет имени М.В. Ломоносова, г. Москва, к.ф.-м.н., старший научный сотрудник факультета ВМК, dnamiot@gmail.com

ИНТЕРФЕЙСЫ ПРИКЛАДНОГО УРОВНЯ В SDN

КЛЮЧЕВЫЕ СЛОВА

Software Defined Networks, API, REST, Parlay.

АННОТАЦИЯ

В этой статье мы обсуждаем интерфейсы прикладного уровня для Software Defined Networks. В то время как программные интерфейсы для взаимодействия с сетевой аппаратурой широко обсуждаются в литературе, интерфейсы для прикладных программ получают меньше внимания. Вместе с тем, вполне очевидно, что именно интерфейсы прикладного уровня очень важны. В настоящей работе мы хотим остановиться именно на этом аспекте Software Defined Networks и сравнить предлагаемые подходы с решениями для интерфейсов прикладного уровня в телекоммуникационных приложениях.

Введение

Концепция Software Defined Networks (SDN), по определению, основана именно на разнообразных программных интерфейсах. В целом, SDN контроллер есть именно набор программных интерфейсов. На рисунке 1 представлена классическая модель SDN.

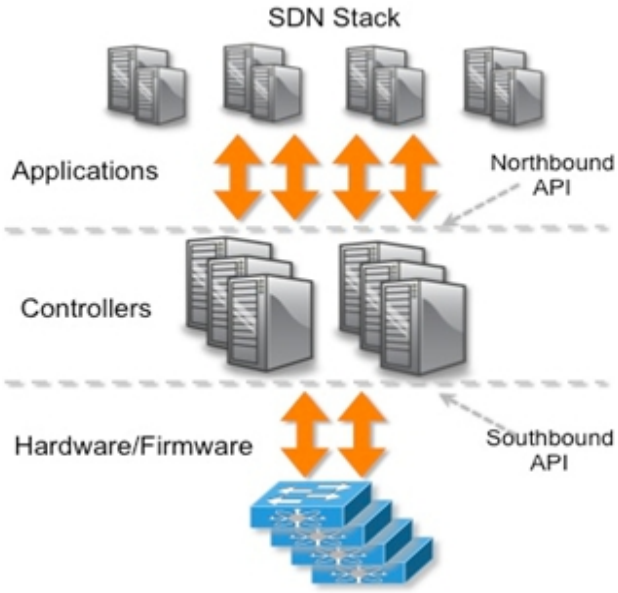


Рис. 1. SDN APIs [1]

В этой работе мы хотим остановиться на элементе, который называется северный API. Northbound API (Application Program Interfaces) предназначен для создания экосистемы приложений. Подобная экосистема и есть основная (по крайней мере, одна из основных) цель SDN. Пользователями данного API являются все разработчики сетевых приложений.

Естественно, имя в виду API, мы говорим именно о едином интерфейсе, который используется всеми разработчиками. В действительности же, насчитывается много различных предложений для интерфейсов прикладного уровня. В работе [2] упоминается более 20 проектов различных SDN контроллеров, каждый из которых вводит свой набор интерфейсов для прикладных программ. Изучением этих интерфейсов занимается консорциум Open Networking Foundation (ONF) [3].

Одной из причин, обуславливающей такие различия, являются разные требования, которые выдвигаются приложениями. Одна из популярных точек зрения состоит в том, что общие API должны быть предложены рынком, а не каким-либо комитетом [4].

API для приложений

Возможно, наиболее подходящим примером для программных интерфейсов прикладных приложений является Parlay. Parlay (Parlay X) представлял собой попытку представить общий API для телекоммуникационных приложений [5]. Основная идея Parlay состояла в объединении сетевого (сете-центрического) подхода в распространении телекоммуникационных услуг (то, за что отвечают интеллектуальные сети) и механизмов представления сервисов на уровне систем управления предприятиями. Parlay API позволяет скрыть сложные механизмы управления в телекоммуникационных сетях, сохраняя в то же самое время уровни безопасности, поддерживаемые телекоммуникационными операторами (рис. 2).

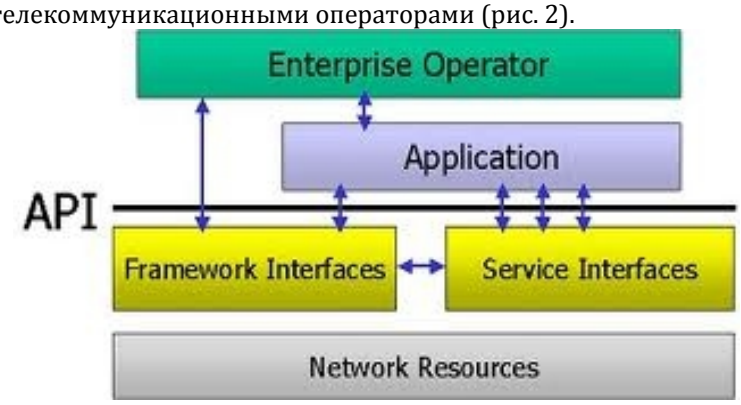


Рис. 2 The Parlay API [6]

Базовые принципы API весьма прозрачны и очевидны:

1) Parlay API посвящен программным интерфейсам, а не сетевым протоколам.
2) Parlay API не зависим от сети.
3) API обеспечивает безопасность на всех уровнях.
4) Простота применения.
5) Расширяемость и возможность пошагового внедрения (использования).

На первый взгляд, это выглядит весьма подходящим для концепции SDN. Но что можно сказать, если мы предметно рассмотрим компоненты Parlay API [7]:

1) Call Control API. Охватывает процесс установки соединения

2) User Interaction API. Отправка SMS, рингтонов и так далее. Чисто телекоммуникационные сервисы.

3) Terminal Capabilities API. Опрос характеристик пользовательских устройств.

4) Connectivity Management API. Это API для Quality of Services (QoS).

5) User Status API. Опрос состояния мобильного терминала.

6) Data Session Control и Account management. Это то, что касается биллинга.

Вывод полностью очевидный. Parlay API описывает, по факту, прикладные телекоммуникационные сервисы. А в случае SDN, мы имеем дело не с сервисами, а с другими приложениями. Название API для приложений нужно понимать здесь именно буквально – API для управления другими приложениями. Для SDN нет такого уровня практической ориентированности, как в случае Parlay API.

Виртуализация функций

ONF занимается также и развитием направления Network Functions Virtualization (NFV). Цель – предложить общие программные интерфейсы для построения виртуальных устройств [8]. Это позволит ускорить процесс создания новых сетевых сервисов. Обычно, операторы используют много типов различного сетевого оборудования. Естественно, что это увеличивает стоимость запуска новых сервисов. NFV призвана создать стандартную технологию виртуализации для консолидации сетевого оборудования в дата-центрах. Иными словами, это концепция сетевой архитектуры, предлагающая использовать технологии виртуализации для виртуализации целых классов функций сетевых узлов в виде составных элементов, которые могут быть соединены вместе или связаны в цепочку для создания телекоммуникационных услуг (сервисов) – рис.3.

Можно сказать, что NFV дополняет SDN, но не заменяет его. В части общих интерфейсов,

SDN Application API, в идеале, должно выступать в качестве программного интерфейса для SDN. Типичные области применения следующие:

- Виртуальные маршрутизаторы и VPN шлюзы;
- Виртуальные сетевые устройства (файрволл, например);
- Виртуальные сетевые сервисы. Например, средства мониторинга;
- Виртуальные приложения. Например, хранилища данных.

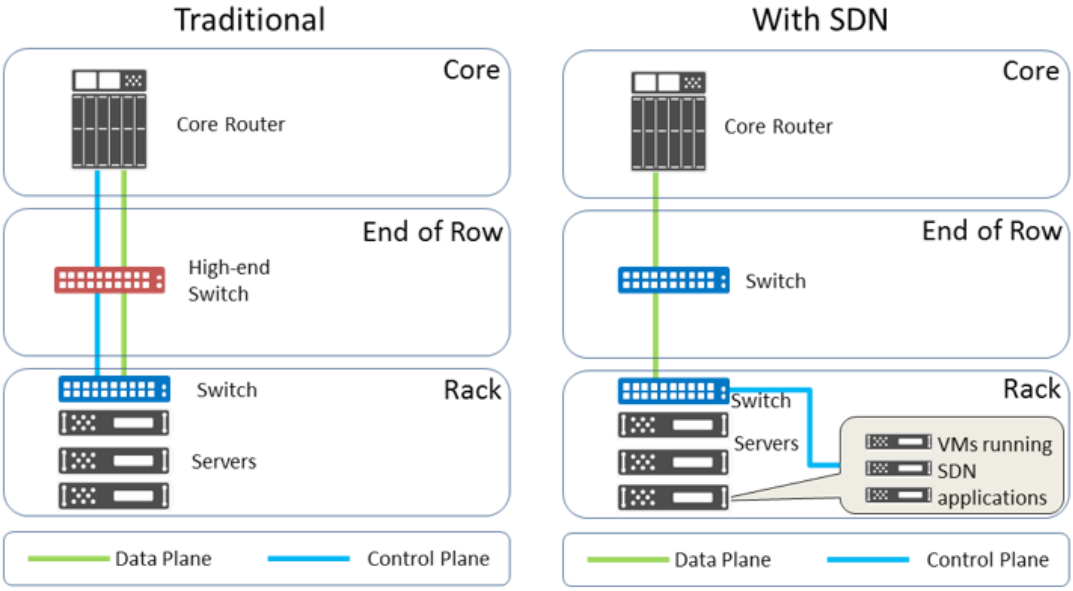


Рис. 3 NFV

Отметим, что понятие “приложение” здесь также весьма отличается от приложений (сервисов) в Parlay.

RESTCONF

Концепция программных интерфейсов в SDN также меняется со временем [11]. Вместо терминов “южный” и “северный” интерфейсы, ONF теперь говорит об интерфейсах данных и приложений [12]. Другая новая модель программных интерфейсов базируется на RSTCONF [13]. В этой связи используется несколько новых терминов: NETCONF, YANG, RESTCONF.

Network Configuration Protocol (NETCONF) [14] предоставляет механизм для управления конфигурацией сетевых устройств. Данные представляются в формате XML. Сами операции загрузки (модификации) конфигураций реализованы как удаленные вызовы (RPCs).

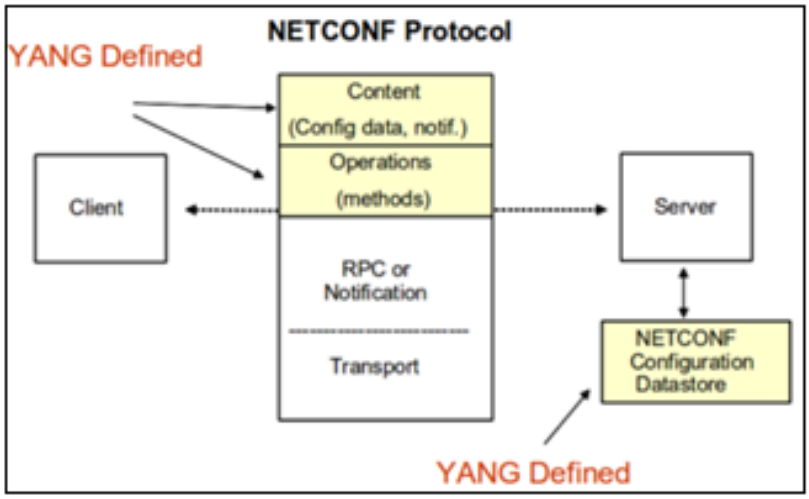


Рис. 4 Уровни NETCONF [15]

YANG представляет собой язык моделирования данных для NETCONF [16]. Модули YANG определяют иерархии данных для NETCONF операций. Модели в YANG представляются в виде

дерева. Например:

```
list interface {
  key "interface-name";
  leaf interface-name {
    type string;
  }
  leaf speed {
    type string;
  }
  leaf duplex {
    type string;
  }
}
```

А типичный файл NETCONF выглядит так:

```
<interface>
  <interface-name>TenGigabitEthernet 1/0/1</login-name>
  <speed>10Gbps</speed>
  <duplex>full</duplex>
</user>
<interface>
  <interface-name>TenGigabitEthernet 1/0/2</login-name>
  <speed>10Gbps</speed>
  <duplex>full</duplex>
</user>
```

На рисунке 5 показана связь NETCONF и YANG:

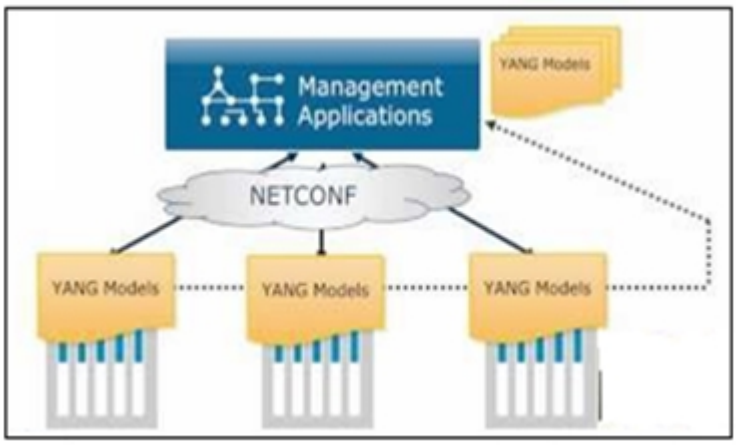


Рис. 5 NETCONF и YANG

RESTCONF описывает основанный на модели REST протокол, который обеспечивает доступ поверх HTTP к данным в YANG, используя хранилище данных в NETCONF [17].

Для SDN это означает появление новой программной модели (модели API), основанной на RESTCONF, NETCONF и YANG.

Пример такой системы приведен на рисунке 6.

В целом, REST-модели от OpenDayLight представляются весь перспективными в плане создания общего API. Схема их построения соответствует общей модели развития программных интерфейсов в межмашинных взаимодействия (M2M) [19] и Internet of Things (Iot) [20].

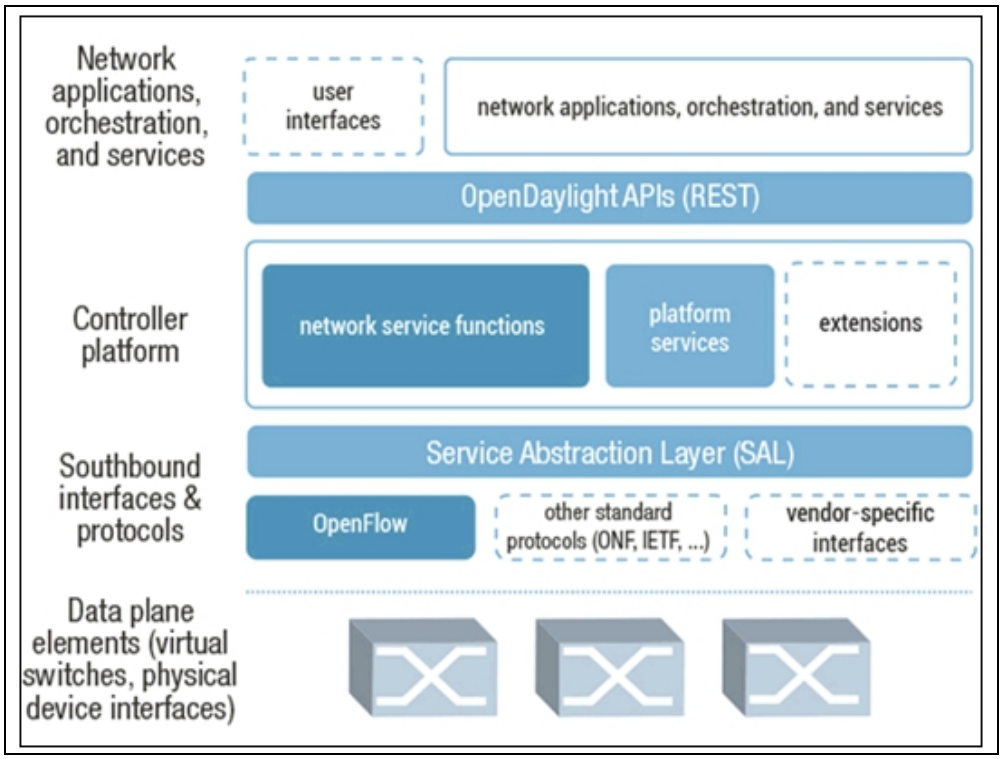


Рис. 6 OpenDayLight модель [18]

Литература

1. The Northbound API- A Big Little Problem http://networkstatic.net/the-northbound-api-2/ Retrieved: Jun, 2015.
2. Sezer, Sakir, et al. "Are we ready for SDN? Implementation challenges for software-defined networks." Communications Magazine, IEEE 51.7 (2013): 36-43.
3. Ortiz, Sixto. "Software-defined networking: On the verge of a breakthrough?." IEEE Computer 46.7 (2013): 10-12.
4. Do SDN northbound APIs need standards? http://searchnetworking.techtarget.com/feature/Do-SDN-northbound-APIs-need-standards Retrieved: Jun, 2015
5. Yates, M. J., and I. Boyd. "The Parlay network API specification." BT Technology Journal 25.3-4 (2007): 205-211.
6. Uve Herzog "OSA & Parlay. Enabling an open services market" http://archive.eurescom.eu/message/messageJun2002/tutorial.asp Retrieved: Jun, 2015
7. Sneps-Sneppe, M., & Namiot, D. (2012, April). About M2M standards and their possible extensions. In Future Internet Communications (BCFIC), 2012 2nd Baltic Congress on (pp. 187-193). IEEE.
8. Schneider, Fabian, et al. "Standardizations of SDN and ITs practical implementation." NEC Technical Journal 8.2 (2014): 16-20.
9. NFV White Paper http://portal.etsi.org/nfv/nfv_white_paper.pdf Retrieved: Jun, 2015
10. Nunes, Bruno, et al. "A survey of software-defined networking: Past, present, and future of programmable networks." Communications Surveys & Tutorials, IEEE 16.3 (2014): 1617-1634.
11. Medved, J., Varga, R., Tkacik, A., & Gray, K. (2014, June). Opendaylight: Towards a model-driven sdn controller architecture. In 2014 IEEE 15th International Symposium on (pp. 1-6). IEEE.
12. Namiot, D., & Sneps-Sneppe, M. (2014, June). On software standards for smart cities: API or DPI. In ITU Kaleidoscope Academic Conference: Living in a converged world-Impossible without standards?, Proceedings of the 2014 (pp. 169-174). IEEE.
13. Bierman, A., Bjorklund, M., Watsen, K., & Fernando, R. (2014). RESTCONF protocol. IETF draft, work in progress.
14. Enns, R., Bjorklund, M., Schoenwaelder, J., & Bierman, A. (2011). Network configuration protocol (NETCONF) (No. RFC 6241).
15. NETCONF layers http://itential.com/tech-briefs/ Retrieved: Jun, 2015
16. YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF) https://tools.ietf.org/html/rfc6020
17. RESTCONF Protocol http://tools.ietf.org/html/draft-bierman-netconf-restconf-04
18. Baucke, S., Mestery, K., Shaikh, A., & Wright, C. (2013). OpenDaylight: An Open Source SDN for your OpenStack Cloud. An Open-Stack Summit, Hong Kong.
19. Namiot D., Sneps-Sneppe M. On M2M Software //International Journal of Open Information Technologies. – 2014. – Т. 2. – №. 6. – С. 29-36.
20. Namiot D., Sneps-Sneppe M. On IoT Programming //International Journal of Open Information Technologies. – 2014. – Т. 2. – №. 10. – С. 25-28.

Вахов А.Н.[1], Зотова Е.А.[2], Коломоец И.В.[3], Рыжов А.П.[4], Шварц А.Ю.[5]

[1] ООО “Uchi.ru”, г.Москва, технический директор, vakhov@uchi.ru

[2] ООО “Uchi.ru”, г.Москва, операционный директор, zotova@uchi.ru

[3] ООО “Uchi.ru”, г.Москва, генеральный директор, kolomoetz@uchi.ru

[4] Российская академия народного хозяйства и государственной службы при Президенте РФ, г.Москва, д.т.н., профессор, заведующий кафедрой “Системы управления бизнес-процессами” ryjov@mail.ru

[5] Московский государственный университете им. М.В. Ломоносова, г.Москва, к.п.н., с.н.с. лаборатории «Психология профессий и конфликта» , shvarts.anna@gmail.com

ПЛАТФОРМА UCHI.RU: ОПЫТ РАЗРАБОТКИ И ПЕРСПЕКТИВЫ РАЗВИТИЯ

КЛЮЧЕВЫЕ СЛОВА

Обучающие системы, обучение математике, персонификация, адаптация, математическое образование.

АННОТАЦИЯ

В статье обсуждаются тренды развития обучающих систем. Рассматривается структура и основные принципы их адаптации и персонификации процесса обучения. Основные положения иллюстрируются

Понятие компьютерных обучающих систем возникло практически одновременно с понятием искусственного интеллекта и претерпело значительную эволюцию одновременно с развитием компьютерных технологий. В данной статье мы не имеем возможности провести такой исторический экскурс; заинтересованному читателю можно порекомендовать работу [1], где эти вопросы достаточно детально обсуждаются. Однако, нельзя не отметить глобальных изменений, происшедших в последние время и происходящих сейчас. Наиболее ярко они представлены в работах McKinsey Global Institute (MGI) [2, 3]. Авторы одного из наиболее авторитетных аналитических институтов, изучая влияние более 100 технологий на развитие экономики и общества, пришли к следующим заключениям:

1. Наиболее важными являются технологии, представленные на рис. 1 [2]. Там же представлены оценки экономического вклада технологии в 2025 году; с методикой расчета эффективности можно ознакомиться в [3].

2. Технологии автоматизации интеллектуальной деятельности (годовой вклад в экономику 2025 года от 5,2 до 6,7 триллионов долларов США в текущих ценах) в первую очередь обеспечат новые способы обучения на базе интеллектуальных систем и big data (рис. 2).

Основным аргументом в пользу такого влияния обучающих систем на экономику авторы видят всеми нами осознаваемое противоречие: исчезновение массовых специальностей (машинистки, операторы ЭВМ и пр.) и появление новых (потребность – сотни тысяч/ миллионы рабочих мест) с одной стороны и практическое отсутствие развития технологий обучения – с другой. Имеется ввиду не замена мела и деревянной доски на компьютер и интерактивную доску, а принципиальные новшества, соразмерные вызовами современной экономики. Более подробно с этим можно ознакомиться в лекции доктора биологических наук, доктора филологических наук, профессора СПбГУ Черниговской Т.В. [4].

Сложившийся к настоящему времени ландшафт направления компьютерных обучающих систем (Ed Tech в англоязычной терминологии) наиболее полно представлен в [5].

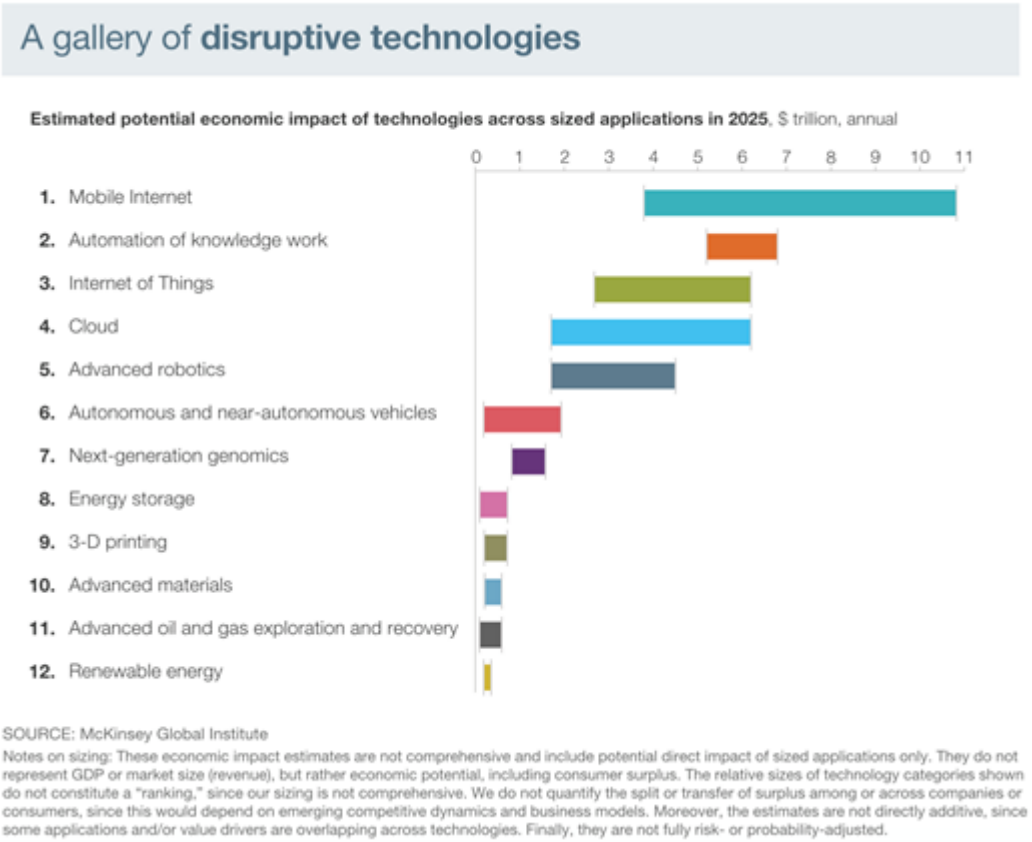


Рис.1. Список прорывных технологий по версии MGI

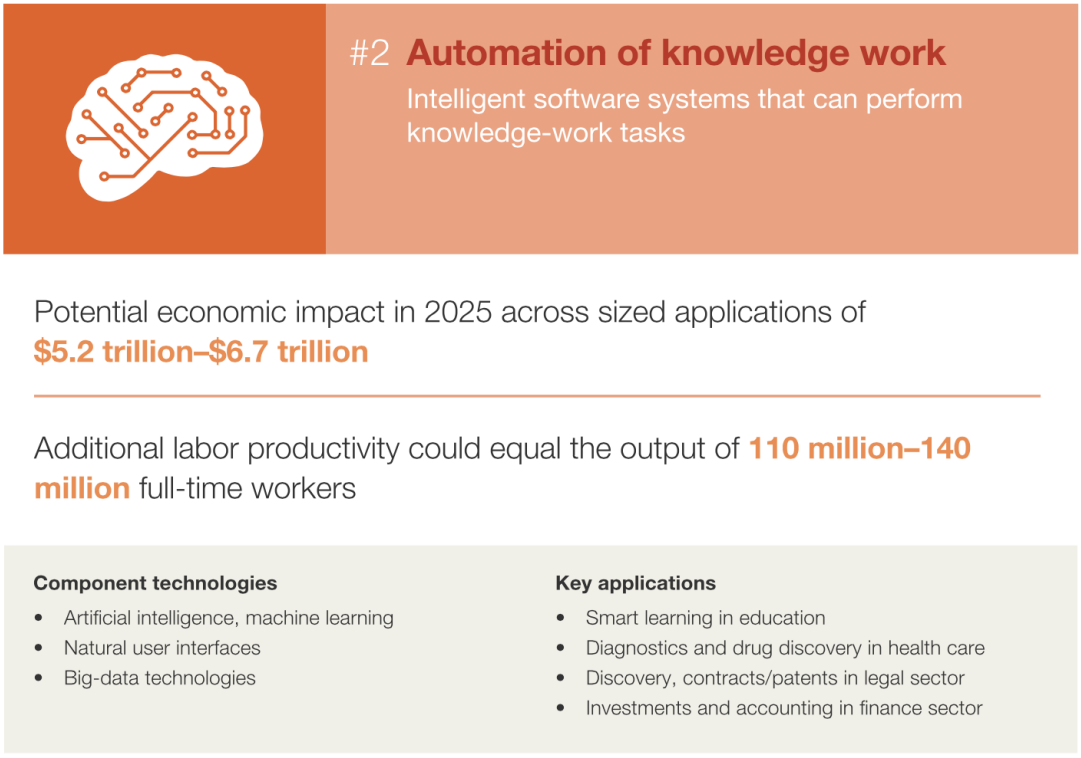


Рис.2. Основные параметры технологии автоматизации интеллектуальной деятельности

На верхнем уровне выделяются инструменты системы для:

- Студентов;
- Преподавателей;
- Учебных заведений и предприятий.

Дальше предлагается следующая классификация:

Для студентов:

- Агрегаторы - компании, предоставляющие доступ к контенту. Выделяют:
  - Discovery – помогают найти контент;
  - Accredited Programs - компании с признаваемой степенью;
- Провайдеры – компании, которые создают контент. Наибольшее распространение:
  - Иностранные языки;
  - Хобби;
  - Навыки 21 века;
- Comprehensions – компании - разработчики инструментов для помощи учить и учиться. Наибольшее распространение имеют:
  - Tutoring – помогают студентам учиться ;
  - Игры как форма обучения.
- Поставщики контента – компании, доставляющие контент до студента. Часто относят поставщиков:
  - Учебников (для традиционного контента);
  - Устройств (для цифрового контента);
- Финансовые сервисы (текущий объем – более 1 триллиона долларов США).

Для преподавателей:

- Classroom Management
  - In-class administration – стандартный набор рутинных операций по управлению процессом обучения;
  - Student Analytics/ Reporting – набор стандартных аналитических инструментов, позволяющих понимать и визуализировать прогресс в обучении студента/ класса/ школы и т.п.
- Curriculum
  - Организация учебного материала в виде курса;
  - Поиск и создание учебного материала/ контента;
- Professional Development – инструменты обучения учителей

Для учебных заведений и предприятий:

- Back office. Обычно включают в себя:
  - Staffing and Professional Assessment – кадровое управление и оценка преподавательского состава;
  - Финансовый сервис (в основном, привлечение денег);
  - Student Information Systems - управление данными по студентам;
  - Learning Management Systems – управление процессом обучения;
- Front office. Обычно включает:
  - Recruiting and Enrollment - нахождение и удержание студентов;
  - Online Distribution – системы, позволяющие синхронизировать on-line и off-line обучение;
  - Learning Content – репозиторий учебных материалов;
  - Credentialing - технологии учета и валидации достижений в обучении.

Поэтому когда мы говорим о компьютерных технологиях в образовании, необходимо уточнение – все перечисленное может быть отнесено к таким технологиям. Ниже мы будем говорить только об обучающих системах – системах для студентов, помогающих эффективно усваивать необходимый учебный материал. И последнее необходимое уточнение – аудитория. Выделяют школьное образование (иногда с включением дошкольного), высшее образование и профессиональное/ корпоративное дополнительное образование; иногда добавляют раздел «навыки/ увлечения/ хобби» - это что-то не связанное с профессиональной деятельностью – охота, фотография и пр. Ясно, что у каждой аудитории много специфики, и соответствующие системы отличаются существенно. Ниже мы будем говорить только о школьном образовании – основном бизнесе и главной компетенции компании Uchi.ru.

Uchi.ru — это онлайн платформа, где ученики начальной школы изучают математику в интерактивной форме. Весь курс разложен на более 700 взаимосвязанных интерактивных заданий по всем темам математики начальной школы, разработанных профессиональными методистами в соответствии с учебной программой. Подход к обучению базируется на данных современных исследований российского [15] и зарубежного [21] математического образования и включает в себя принцип множественных репрезентаций [16], принцип активности ученика, особое внимание уделяется развитию теоретического восприятия [19].

В настоящее время (по состоянию на середину октября 2015 г.) услугами компании пользуются более 180 000 школьников (около 3% всех школьников РФ соответствующих классов), более 20 000 учителей, работающих в 8 000 школах РФ. Решено более 2 000 000 заданий, причем половина – с сентября 2015 года. Таким образом, Uchi.ru является лидером своего сегмента в России.

Особенностью программного обеспечения системы Uchi.ru является возможность сбора и накапливания всей возможной информации о поведении ученика в процессе решения задач: время решения, количество правильных ответов, характеристики не правильных ответов и др. Мы понимаем, происходит поиск правильного решения целенаправленно или случайным образом, насколько ученик внимателен, степень утомления («играется» ли он мышкой, например). Накопленные данные позволяют применять широкий набор инструментов анализа данных для оптимизации процесса обучения и его настройки для конкретного ученика (персонификации процесса обучения). Наработанные шаблоны и решения составят основу Smart learning in Education (рис. 2) или, более широко, «вычислительной педагогики», способной решать упомянутые выше вызовы современной экономики.

Минимальная верхнеуровневая архитектура, позволяющая решать подобные задачи, представлена на рис. 3.

Рис.3. Минимальная верхнеуровневая архитектура

На основе первичных измерений и экспресс-тестов система оценивает принадлежность конкретного ученика к классам по параметрам Внимательность, Скорость обработки информации, Выносливость, Способ обработки информации (аудио-, видео-, смешанный). Первые два параметра являются более подвижными и зависимыми от ряда факторов (от настроения ученика до атмосферного давления), вторые два – более стабильными. Поэтому нужна не только система распознавания или оценки принадлежности к классам, но и система мониторинга (динамическая составляющая распознавания). Заметим, что технология оценки и мониторинга сложных процессов применялись авторами для разработки соответствующих систем для различных областей от соблюдения международных договоров [6, 7] (заказчик – Международное Агентство по Атомной Энергии) до процессов разработки в микроэлектронике [8, 9] (заказчик – Cadence Design Systems, Inc. – лидер рынка САПР микроэлектроники). Математические и технологические основы таких систем, а также особенности разработки приложений достаточно подробно описаны, например, в [10,11].

Аналогичным образом осуществляется оценка и мониторинг процесса обучения конкретного ученика в данное время (рис. 3). На основе вычисленных принадлежностей к заданным классам и первичных измерений осуществляется управление процессом обучения. Управляемыми параметрами являются количество предлагаемых заданий одного типа; время, отведенное на выполнение задание, наличие объясняющих материалов; темы разрешенные для прохождения и ряд других, реализованных в различной степени в текущей версии системы.

Рис.4. Расширенная верхнеуровневая архитектура

В новых версиях мы будем расширять как состав собираемой информации, так и набор методов ее анализа. Состав собираемой информации может быть расширен за счет применения методов отслеживания направления взгляда (eye-tracking) [17, 18], health trackers и различных гаджетов, появляющихся систем оценки различных аспектов психофизического состояния человека (например, [12]), становящейся доступной специальной аппаратуры. Мы также апробируем компьютерные версии нейропсихологических тестов, позволяющие оценивать состоянии соответствующих функций, которые в дальнейшем сделают обучение еще более специализированным в зависимости от нейропсихологического профиля [20].

Расширенная верхнеуровневая архитектура представлена на рис. 4.

Отметим, что в рамках описанных подходов возможна разработка оптимальных систем персонификации процесса обучения. Подходы к такого рода оптимизации описаны в [13, 14], однако их полноценное использование требует проведения дополнительных работ.

Авторы надеются, что изложенные факты и представленные результаты будут интересны и полезны для специалистов по компьютерным обучающим системам, а также привлекут внимание специалистов в области интеллектуальных систем и big data к этой интенсивно развивающейся, перспективной и востребованной области.

Литература

1. Алисейчик П.А., Вашик К., Кнап Ж., Кудрявцев В.Б., Строгалов А.С., Шеховцов С.Г. Компьютерные обучающие системы. Интеллектуальные системы, Т.8, 2004 г. с. 5-44.
2. A gallery of disruptive technologies - http://www.mckinsey.com/assets/dotcom/mgi/slideshows/disruptive_tech/index.html#
3. James Manyika, Michael Chui, Jacques Bughin, Richard Dobbs, Peter Bisson, Alex Marrs. Disruptive technologies: Advances that will transform life, business, and the global economy. McKinsey Global Institute (MGI), May 2013, 176 p. - http://www.mckinsey.com/insights/business_technology/disruptive_technologies
4. Черниговская Т.В. Как научить мозг учиться? http://www.youtube.com/watch?v=nEGmdlJEr8M
5. Ed Tech Market Map by Flybridge Capital Partners on 25 September 2015. https://prezi.com/xguky7u7aur6/ed-tech-market-map/
6. Ryjov A., Belenki A., Fattah A., Hooper R., Pouchkarev V., Zadeh L.A. Development of an Intelligent System for Monitoring and Evaluation of Peaceful Nuclear Activities (DISNA) Volume 1: Conceptual Model. IAEA, STR-310, Vienna, 1998, 122 p.
7. Fattah A., Pouchkarev V., Belenki A., Ryjov A., Zadeh L.A. Application of Granularity Computing to Confirm Compliance with Non-Proliferation Treaty. In: Data Mining, Rough Sets and Granular Computing. Tsau Young Lin, Yiyu Y. Yao, L.A. Zadeh (Eds.). Physica-Verlag Heidelberg, 2002, p. 308-338.
8. Lebedev A.A., Ryjov A.P. Design team capability and project progress evaluation based on information monitoring technology. 5th International Conference on Soft Computing, Computing with Words and Perceptions in System Analysis, Decision and Control. 2-4 September 2009, Famagusta, North Cyprus, pp. 48-52.
9. Лебедев А.А., Рыжов А.П. Оценка и мониторинг проектов разработки высокотехнологичных изделий на примере микроэлектроники. Интеллектуальные системы, Том 11, вып. 1-4, 2008, с. 55-82.
10. Ryjov A. Basic principles and foundations of information monitoring systems. In: Monitoring, Security, and Rescue Techniques in Multi-agent Systems. Barbara Dunin-Keplicz, Andrzej Jankowski, etc. (Eds.). Springer-Verlag, 2005, ISBN 3-540-23245-1, ISSN 16-15-3871, pp. 147-160.

11. Рыжов А.П. Информационный мониторинг сложных процессов: технологические и математические основы. Интеллектуальные системы, Том 11, вып. 1-4, 2008, с. 101-136.
12. Смартфон определит депрессию точнее психолога. https://hi-tech.mail.ru/news/smartphones-and-depression/
13. Alexander Ryjov. Towards an optimal task-driven information granulation. In: Information Granularity, Big Data, and Computational Intelligence. Witold Pedrycz and Shyi-Ming Chen (Eds.). Springer International Publishing Switzerland 2015, pp. 191-208.
14. Рыжов А.П. Некоторые задачи оптимизации и персонификации социальных сетей. – Saarbrücken, LAP, 2015, 88 с.
15. Shvarts. A. Russian mathematics education: programs and practices (series on mathematics education, volume 5). Research in Mathematics Education, 3(15):318–323, 2013. [DOI]
16. Шварц А.Ю. Роль чувственных представлений в математическом познании и понимании математики. Психологические исследования: электронный научный журнал, 17(3), 2011.
17. Barmby P., Andrà C., Arzarello F., Gomez, D. Obersteiner A., Shvarts A. (2014) The use of eye-tracking technology in mathematics education research. In Nicol, C., Liljedahl, P., Oesterle, S. & Allan, D. (Eds.). Proceedings of the Joint Meeting of PME 38 and PME-NA 36 (Vol. 1). Vancouver, Canada: PME. P. 253
18. Chumachenko D., Shvarts A., Budanov A. The development of the visual perception of the cartesian coordinate system: an eye tracking study (2014) In Nicol, C., Liljedahl, P., Oesterle, S. & Allan, D. (Eds.) Proceedings of the Joint Meeting of PME 38 and PME-NA 36 (Vol. 2). Vancouver, Canada: PME.
19. Krichevets, A.N.,Shvarts, A.Yu., & Chumachenko, D.V. (2014). Perceptual action of novices and experts in operating visual representations of a mathematical concept. Psychology. Journal of Higher School of Economics, 11(3).
20. Ахутина Т, Корнеев , А., Матвеева Е., Агрис А. Возрастные изменения высших психических функций у детей 7-9 лет с разными типами дефицита регуляции активности Психология. Журнал Высшей Школы экономии. — 2015. — Т. 12, № 3. — С. 131–152.
21. A. Gutiérrez & P. Boero (Eds.), Handbook of research on the psychology of mathematics education. Rotterdam: Sense, 2006.

Денисов В.С.

Московский государственный университет им. М.В.Ломоносова, ф-т ВМК, науч. сотр.

УПРАВЛЕНИЕ НАСТРОЙКАМИ JAVA-ПРИЛОЖЕНИЙ С ИСПОЛЬЗОВАНИЕМ МЕХАНИЗМА АННОТАЦИЙ

КЛЮЧЕВЫЕ СЛОВА

Программирование, Java, библиотека, фреймворк, аннотации, настройки.

АННОТАЦИЯ

В статье рассматривается библиотека options-util, реализующая управление настройками времени выполнения для Java-приложений*. Конфигурация свойств приложения и их отображение на источники конфигурационной информации настраивается при помощи аннотаций над специальными интерфейсами настроек, что упрощает внедрение библиотеки, гарантирует строгую типизацию и обеспечивает самодокументированность кода.*

Получение настроек времени выполнения и их использование для модификации работы приложения является неотъемлемой частью практически любого приложения. Однако по различным причинам библиотеки-фреймворки, реализующие общие шаблоны управления настройками,

Цели разработки библиотеки

Библиотека *options-util* разрабатывается для упрощения управления настройками времени выполнения в проектах различного уровня и назначения на платформе Java, исполняющихся как на классических серверных и настольных системах, так и в облачных вычислительных сервисах. Библиотека должна использовать все современные языковые, технические и функциональные возможности платформы Java 8, а также применять передовые подходы и эффективные шаблоны, сложившиеся в области Java-программирования. После окончания разработки библиотека должна соответствовать следующим функциональным требованиям[1]:

Общие требования:

- независимость от целевой платформы;
- использование юнит- и интеграционных тестов;
- поддержка внедрения зависимостей.
- Специальные требования:
- минимальное количество зависимостей;
- описание конфигурационной информации с помощью аннотаций;
- самодокументация объектов конфигурации;
- поддержка облачных сервисов;
- безопасность типов (type safety);
- поддержка различных источников информации о настройках/постоянных хранилищ;
- расширяемость;
- поддержка сложных структур данных;
- поддержка валидации и конверсии значений;
- поддержка извещений об изменении настроек;
- поддержка структурированной информации.

На сегодня библиотека *options-util* находится в состоянии альфа-версии, исходный текст библиотеки доступен на платформе GitHub [3], а первый релиз, версия 0.1, может быть получен из Maven-репозитория Maven Central [4].

Примеры использования библиотеки

Основным понятием библиотеки *options-util* является интерфейс настроек (options interface), определяющий все настройки (свойства), используемые приложением, с помощью набора геттеров и сеттеров, которые, в свою очередь, декорируются аннотациями, содержащими

1 Более подробно существующие библиотеки управления настройками рассматриваются в [1], а требования излагаются и анализируются в [2]

дополнительную семантическую информацию, обеспечивающую функционал библиотеки. В самом простом случае интерфейс настроек может выглядеть так, как на Рис. 1:

```
@Persistence(PersistenceType.TRANSIENT)
public interface TestOptions extends Options {
    @Option(key = "int")
    int getInt();
    void setInt(int value);

    @Option(defaultValue = "5")
    int getIntWithDefault();
    void setIntWithDefault(int value);
}
```

Рис. 1. Простейшее использование интерфейса настроек

Даже такой простой пример показывает ряд интересных возможностей библиотеки:

- интерфейс может быть декорирован аннотацией @PersistenceProvider, определяющей тип постоянного хранилища, используемого для хранения настроек;
- все интерфейсы настроек наследуют общий интерфейс org.plukh.options.Options, который определяет некоторые общие полезные методы для работы с настройками (в частности, загрузку и сохранение настроек).
- аннотация @Option передаёт библиотеке дополнительную информацию о конфигурационном свойстве, в данном случае — значение ключа свойства в постоянном хранилище для первого свойства и значение по умолчанию для второго.

Для получения доступа к настройкам приложение должно породить экземпляр объекта, реализующего интерфейс настроек, за эту операцию отвечает фабрика OptionsFactory, см. Рис. 2:

```
TestOptions options = (TestOptions) OptionsFactory.getOptionsInstance(TestOptions.class);
```

Рис. 2. Порождение объекта настроек с помощью фабрики OptionsFactory

Фабрика порождает динамический прокси-объект, реализующий доступ к настройкам при вызове методов интерфейса. Всем классам предоставляется один и тот же экземпляр прокси-объекта, таким образом, все компоненты приложения получают в своё распоряжение одно и то же множество настроек. Прокси-объекты потоко-безопасны и могут быть свободно использоваться одновременно несколькими потоками в многопоточном приложении.

Реализация функциональных требований

На сегодня библиотека реализует, в том или ином виде, большую часть функциональных требований, однако поддержка некоторых требований пока достаточно рудиментарна или носит прототипный характер.

Независимость от целевой платформы. Библиотека не делает никаких предположений относительно платформы, на которой она выполняется, в том числе:

- не предполагается наличие на платформе файловой системы, кроме случаев, когда используется один из провайдеров файлового постоянного хранилища;
- не используется манипуляция байт-кодом;
- не использует иных способов журналирования, кроме вывода на консоль, и то только в том случае, когда информация не может быть передана никаким иным образом;
- не порождает потоков и не использует связанные с ними функции;
- использует небольшое количество памяти и ресурсов процессора (менее 2 Мб динамической памяти для интерфейса из нескольких сот свойств);

Использование юнит- и интеграционных тестов. Библиотека покрыта набором тестов, обеспечивающим покрытие 74% строк; непокрытые строки, в основном, принадлежат тривиальным методам — геттерам, сеттерам, конструкторам и т. п. По мере обнаружения и исправления ошибок в коде библиотеки, пишутся дополнительные регрессионные тесты.

Поддержка внедрения зависимостей. На текущий момент, поддержка технологий внедрения зависимостей не реализована. В будущих версиях планируется поддержка фреймворков Spring [5] и Guice [6].

Минимальное количество зависимостей. Текущая версия не имеет внешних зависимостей

времени выполнения вообще, и планируется, что для базовой функциональности библиотеки зависимости не будут требоваться никогда.

Описание конфигурационной информации с помощью аннотаций. Как можно видеть из примера на Рис. 1, библиотека *options-util* предоставляет аннотации, с помощью которых приложение может передать библиотеке различную дополнительную информацию о свойствах настроек.

Самодокументация объектов конфигурации. Рассмотрим интерфейс настроек на Рис. 3:

```
@Persistence(PersistenceType.PROPERTIES_FILE)
public interface TweetCaptureOptions extends Options {
    @Option(key = "jdbc.url")
    String getDatabaseURL();

    @Option(key = "workerthreads", defaultValue = "5")
    int getWorkerThreads();

    @CollectionOption(backingClass = LinkedBlockingQueue.class, transientOption = true)
    Queue getWorkerQueue();
}
```

Рис. 3. Интерфейс настроек TweetCaptureOptions

Аналогично примеру на Рис. 1, даже не зная специфики библиотеки *options-util*, можно сделать ряд предположений о настройках, определяемым данным интерфейсом:

- интерфейс определяет ровно три настройки (свойства);
- конфигурация приложения доступна в файле свойств (properties);
- первое свойство возвращает строковое значение - URL для доступа к базе данных; значение доступно только на чтение (так как сеттер отсутствует); наконец, значение ключа в постоянном хранилище — "jdbc.url";
- второе свойство возвращает целое число — количество рабочих потоков в данном приложении; значение ключа и значение по умолчанию также определяются аннотацией;
- наконец, третье свойство возвращает очередь, реализуемую классом LinkedBlockingQueue из Java Collections Framework; для него установлен признак "transient" ("временный"), и значение этого свойства не будет читаться или записываться в постоянное хранилище.

Легко видеть, что использование аннотаций вместе со стандартными для Java принципами именования методов позволяет создавать достаточно экспрессивный код, не требующий большого количества дополнительной внешней документации.

Поддержка облачных сервисов. Прототипная реализация облачного модуля реализует поддержку конфигурационных файлов в облачном хранилище AWS Simple Storage Service [7]. Конфигурационный файл может определяться идентификатором виртуальной машины EC2 или упорядоченным множеством тэгов, передаваемых библиотеке в аннотации @AWSTags.

Безопасность типов (type safety). Библиотека *options-util* использует возможности строгой типизации Java для обеспечения безопасности типов — проверка типа гарантирована во время компиляции приложения.

Поддержка различных источников информации о настройках/постоянных хранилищ. Текущая версия библиотеки поддерживает чтение конфигурации из .properties и XML-файлов, а также из источников данных JDBC. Поддерживаются механизмы расширения, позволяющие добавлять реализации произвольных постоянных хранилищ. Дополнительно, в будущих версиях запланирована поддержка следующих источников конфигурационных данных в базовом модуле:

- файлы JSON;
- файлы .properties, XML и JSON через протокол HTTP;
- файлы .properties, XML и JSON из classpath;
- значения переменных окружения Java и переменных окружения ОС;
- поддержка JMX;

Запланированы облачные модули для платформ Amazon Web Services, Google Cloud Computing и Microsoft Azure, со специфическими для каждой платформы источниками

информации. Также запланировано сцепление нескольких источников информации в один мета-источник.

Расширяемость. Библиотека поддерживает несколько точек расширения: провайдеры постоянных хранилищ, провайдер цепочки постоянных хранилищ, обработчик типа свойства и др. По мере развития библиотеки список доступных точек будет естественным образом увеличиваться.

Поддержка сложных структур данных. Библиотека *options-util* поддерживает произвольные типы данных Java в качестве объектов, хранящих настройки приложения. Единственным требованием является предоставление разработчиком конвертера, обеспечивающего сериализацию/десериализацию значений объектов в строковые значения, доступные для редактирования пользователями[2]. Библиотека содержит следующие встроенные конверторы:

- все примитивные типа Java и их объектные аналоги;
- java.lang.String;
- java.util.Date;
- все классы, реализующие интерфейс java.util.Collection.

На сегодня существенным ограничением библиотеки является то, что один и тот же тип свойства должен использовать один и тот же конвертер (внутри библиотеки известный как обработчик типа свойства) для всех свойств этого типа в интерфейсе настроек. В следующих версиях это ограничение будет преодолено.

Поддержка валидации и конверсии значений. Наличие ошибок в файлах, редактируемых пользователями, абсолютно неизбежно. В задачи обработчиков типа свойства (как встроенных, так и внешних) входит максимально терпимое отношение к незначительным ошибкам в форматировании данных[3], а также корректная обработка существенных ошибок.

При получении настроек из постоянного хранилища приложение может выбрать одну из двух стратегий обработки входных данных (представляющих из себя строковые значения, предназначенные для десериализации в соответствующие Java-объекты с помощью обработчиков типа) — гибкую и строгую. При выборе строгой стратегии библиотека выбросит исключение при первой же фатальной ошибке конверсии, и никакие значения настроек не будут изменены — эта стратегия хорошо подходит для не интерактивных приложений, которые, как правило, не могут продолжить выполнение без получения конфигурации. Гибкая политика позволяет считать и применить все корректные значения настроек, а после этого — получить от библиотеки список ошибок конверсии — эта стратегия подходит для интерактивных приложений, которые, например, могут предложить пользователю ввести корректные значения для свойств, которые были считаны с ошибками.

Поддержка извещений об изменении настроек и п*оддержка структурированной информации* на сегодня в библиотеке не реализованы. Их реализация планируется в одной из следующих версий библиотеки.

Литература

1. V. Denisov. Overview of Java application configuration frameworks. International Journal of Open Information Technologies ISSN: 2307-8162, 1(6), 2013. URL: http://injoit.org/index.php/j1/article/view/33
2. V. Denisov. Functional requirements for a modern application configuration framework. International Journal of Open Information Technologies ISSN: 2307-8162 3(10), 2015. URL: http://injoit.org/index.php/j1/article/view/236
3. Options-util: annotation-based Java configuration helper library. URL: https://github.com/options-util/options-util
4. URL: http://search.maven.org/#artifactdetails%7Corg.plukh%7Coptions-util%7C0.1%7Cjar
5. Spring Framework. URL: http://projects.spring.io/spring-framework/
6. Google Guice. URL: https://github.com/google/guice
7. V. Denisov. Annotations-driven configuration framework for Java applications. International Journal of Open Information Technologies ISSN: 2307-8162 3(10), 2015. URL: http://injoit.org/index.php/j1/article/view/237.

2 В обязанности разработчика конвертера входит задача определения такой схемы сериализации/десериализации, которая, с одной стороны, была бы понятна и доступна для редактирования без применения специальных инструментов, а с другой — обеспечивала однозначность сериализации/десериализации

3 Так, встроенный обработчик типа java.util.Date пробует применить к текстовому значению даты/времени несколько различных длинных, средних и коротких форматов, прежде чем выдать ошибку конверсии

Денисов В.С.

Московский государственный университет им. М.В.Ломоносова, ф-т ВМК, науч. сотр.

ФУНКЦИОНАЛЬНЫЕ ТРЕБОВАНИЯ К БИБЛИОТЕКЕ УПРАВЛЕНИЯ НАСТРОЙКАМИ JAVA-ПРИЛОЖЕНИЙ

КЛЮЧЕВЫЕ СЛОВА

Программирование, Java, библиотека, фреймворк, аннотации, настройки.

АННОТАЦИЯ

В статье рассматриваются общие и специальные требования к современной библиотеке управления настройками Java-приложений. Приводится краткое обоснование каждого требования.

Поддержка тех или иных способов настройки времени выполнения является неотъемлемой частью функционала подавляющего большинства приложений. Сформировавшийся за многие годы базовый подход к получению настроек достаточно прост — настройки считываются из некоторого постоянного хранилища (например, из файла на файловой системе, из таблицы в РСУБД, системного хранилища конфигурационной информации и т. п.) при старте приложения (а также, возможно, и после его старта — например, при изменении содержимого файла) и сохраняются во внутренних структурах (объектах) приложения, откуда в дальнейшем используются прочими программными компонентами. Если это необходимо, в ходе выполнения настройки могут также сохраняться в постоянном хранилище.

Формирование типового подхода к решению задачи обычно влечёт за собой разработку фреймворков, реализующих данный шаблон для различных программных платформ. Платформа Java не является исключением - в [1] описаны некоторые популярные библиотеки для работы с настройками, а также проблемы, возникающие при их применении.

К сожалению, библиотеки первого поколения, созданные, в основном, в 2000-2005 гг, в последние годы практически прекратили своё развитие, серьёзно отстав на концептуальном и техническом уровне от новейших возможностей платформы Java, таких как внедрение зависимостей, объектно-реляционные отображения, лямбда-выражения, аннотации, поддержка платформ облачных вычислений и т. п. Возникла потребность в формировании требований, а после этого и в разработке нового поколения библиотек управления настройками времени выполнения.

Всю совокупность требований к библиотеке управления настройками можно разделить на два больших класса: общие требования — требования, не относящиеся к специфике предметной области — и специальные, эту специфику учитывающие.

К общим требованиям относятся:

- независимость от целевой платформы;
- использование юнит- и интеграционных тестов;
- поддержка внедрения зависимостей.

К специальным требованиям относятся:

- минимальное количество зависимостей;
- описание конфигурационной информации с помощью аннотаций;
- самодокументация объектов конфигурации;
- поддержка облачных сервисов;
- безопасность типов (type safety);
- поддержка различных источников информации о настройках/постоянных хранилищ;
- расширяемость;
- поддержка сложных структур данных;
- поддержка валидации и конверсии значений;
- поддержка извещений об изменении настроек;

- поддержка структурированной информации.

Независимость от целевой платформы. Виртуальная машина Java может быть запущена на самых разных программных и аппаратных платформах — с различной архитектурой, операционными системами, системами хранения и т. п. Библиотека управления настройками должна делать как можно меньше предположений о специфике целевой платформы — например, о наличии у неё поддержки файловой системы, доступа к сети и т. п.

Использование юнит- и интеграционных тестов. Библиотека управления настройками предоставляет приложению критически важные сервисы — без получения информации о конфигурации времени выполнения полноценная работа приложения, как правило, не является возможной. Кроме того, в силу специфики функционала библиотеки она неизбежно будет содержать множество различных, слабо пересекающихся потоков вычислений (рассчитанных на различные окружения, источники данных, форматы конфигурационной информации и иные возможности библиотеки), подавляющее большинство которых не будет задействовано в любом конкретном операционном окружении. Учитывая эти особенности, работа библиотеки должна быть верифицирована набором автоматически исполняемых тестов, обеспечивающих адекватное покрытие исходного кода библиотеки (например, в [2] предлагается уровень покрытия в 85% строк).

Поддержка внедрения зависимостей. Внедрение зависимостей может применяться по отношению к библиотеке управления настройками двояко: с одной стороны, библиотека может получать внешние зависимости (например, зависящие от среды выполнения) от фреймворка внедрения зависимостей (или, в более общем случае, от контейнера инверсии контроля — IoC-контейнера), а с другой — предоставлять свои сервисы через подобный фреймворк остальным компонентам приложения. Это особенно важно для организации юнит-тестирования, когда сервисы, использование которых при тестировании неэффективно или сопряжено с финансовыми затратами, заменяются своими фиктивными копиями, которые внедряются в компоненты библиотеки.

Минимальное количество зависимостей. Библиотека управления настройками должна обеспечить простоту использования в проектах любой сложности, в том числе и в насчитывающих всего несколько сот строк кода. "Большие" проекты, как правило, строятся вокруг специализированной системы управления зависимостями, такой как Maven или Gradle (см. в [3] описание нескольких распространённых систем управления зависимостями для платформы Java), однако для "малых" проектов, зачастую, применение таких систем оказывается неоправданно сложным. Соответственно, базовая функциональность библиотеки — в идеале — вообще не должна иметь внешних зависимостей, а функциональность, использование которой невозможно без привлечения зависимостей (например, SDK для облачных сервисов) должна быть вынесена в набор дополнительных необязательных модулей.

Описание конфигурационной информации с помощью аннотаций. Аннотации в Java позволяют разработчику связать дополнительную семантическую информацию с определёнными синтаксическими элементами кода, такими как классы, методы, переменные-члены класса и т. д. Аннотации располагаются непосредственно в исходных текстах приложения, а при компиляции могут становиться частью байт-кода, сохраняя доступность (через механизм отражения) во время выполнения приложения. Всё это делает аннотации идеальным способом передачи подобной информации как между компонентами приложения, так и между компонентами приложения и внешними зависимостями. Для управления настройками, аннотации позволяют передать информацию, специфическую для данной области (например, информацию о том, является ли определённое свойство доступным только для чтения или для чтения и записи, определить способ отражения свойства при сохранении в постоянном хранилище и т. п.) Кроме того, аннотации — за счёт встраивания непосредственно в исходный текст программы — помогают создавать самодокументированный код (см. ниже).

Самодокументация объектов конфигурации. Когда новый разработчик начинает знакомство с настройками, доступными в приложении, у него возникает достаточно типовой набор вопросов:

- какие настройки/свойства доступны для приложения?
- какие свойства доступны на чтение? на чтение и запись?
- какие свойства сохраняются в постоянном хранилище, а какие имеют смысл только во время выполнения приложения?

- какие значения (по типу, по диапазону) может принимать свойство?
- в каком постоянном хранилище сохраняются настройки, и доступно ли это хранилище только на чтение или на чтение и запись?

Удобно, когда ответы на эти вопросы могут быть просто определены непосредственно в исходном тексте приложения — самодокументация ускоряет разработку и уменьшает количество ошибок и разночтений при использовании объектов настроек.

Поддержка облачных сервисов. Эффективное использование современных облачных сервисов (а облачные сервисы используются более 90% компаний, см. [4]) требует серьёзных изменений в архитектуре приложений, затрагивая, в том числе, и работу с настройками — от отсутствия локальной файловой системы до отсутствия концепции "вычислительного узла" в некоторых видах облачных вычислительных сервисов. Приложения, конфигурируемые "в облаке", часто должны поддерживать нестандартные источники информации — в том числе специализированные облачные постоянные хранилища.

Безопасность типов. При хранении настроек внутри приложения в виде строковых объектов (а это до сих пор самый распространённый подход, благодаря классу java.util.Properties) часто возникает вопрос: "Какому типу Java на самом деле соответствуют значения определённого свойства?" Целый это тип или тип с плавающей либо фиксированной запятой? Или, возможно, в качестве значения также может использоваться собственно строка символов? Или это некий сложный объект, сериализованный в строку? Существующие библиотеки, к сожалению, практически не содержат способов определить тип свойства в момент разработки приложения. Соответственно, современная библиотека должна содержать средства, гарантирующие безопасность типов, как при чтении значений свойств, так и при их модификации. Для этих целей естественным образом подходят интерфейсные классы — если для чтения и модификации значений свойств используются только методы этого класса, то безопасность типов гарантируется компилятором Java.

Поддержка различных источников информации о настройках/постоянных хранилищ. Современные Java-приложения работают в самых различных операционных окружениях, на классических серверах и рабочих станциях, в облачных и встраиваемых платформах, на мобильных устройствах. В зависимости от окружения, настройки могут быть получены из различных источников (список, конечно, далеко не исчерпывающий):

- традиционные файлы настроек различного формата;
- переменные окружения (виртуальной машины Java и ОС);
- реляционные и нереляционные СУБД;
- облачные системы хранения документов;
- веб-сервисы.

Кроме того, в некоторых сценариях данные из различных источников могут комбинироваться друг с другом. Так, в облачном окружении AWS EC2 может использоваться следующая цепочка получения настроек:

- значения по умолчанию считываются из ресурса, расположенного внутри classpath;
- общие для всех виртуальных машин настройки считываются из документа в хранилище S3;
- специфические для виртуальной машины настройки считываются с помощью HTTP-запроса из метаданных виртуальной машины.

Библиотека управления настройками должна поддерживать чтение (и запись, в том случае, когда это технически возможно) различных форматов конфигурационной информации из различных источников — с динамическим выбором источника или цепочки источников в зависимости от операционного окружения, прозрачно для приложения.

Расширяемость. Поскольку получение настроек приложения неразрывно связано с операционным окружением приложения, сделать библиотеку управления настройками полностью самодостаточной невозможно — всегда найдётся особенность окружения (или использующего библиотеку приложения), для которой существующей стандартной функциональности библиотеки окажется недостаточно. Можно выделить по крайней мере две основные точки расширения функционала: доступ к постоянному хранилищу и формат конфигурационной информации. Библиотека должна определять стандартные механизмы расширения, позволяющие разработчикам добавлять необходимый функционал без модификации исходного кода библиотеки.

Поддержка сложных структур данных. Большинство существующих библиотек управления настройками поддерживают в качестве значений свойств только очень простые типы данных: числа, строки и, возможно, массивы/списки строк и чисел. Однако при конфигурации сложных приложений возникает потребность в отображении настроек на более сложные структуры данных. Библиотека должна предоставлять возможность отображать настройки на наиболее часто используемые структуры данных (включая Java Beans и коллекции из Java Collections Framework), а также предоставить приложению возможность расширения функциональности для реализации поддержки произвольных, сколь угодно сложных, типов данных.

Поддержка валидации и конверсии значений. Рано или поздно, библиотека обязательно столкнётся с ошибкой в конфигурационных данных — это может быть как ошибка формата (например, неверно сформированный XML-документ), так и недопустимое значение одного из свойств. Реакция библиотеки на подобные ошибки должна быть чёткой и предсказуемой: библиотека должна, в зависимости от требований приложения, или быстро и однозначно сигнализировать о наличии ошибки (например, путём бросания проверяемого исключения), или заменить значения "пострадавших" свойств некоторыми разумными значениями по умолчанию. Кроме того, должна поддерживаться возможность расширения поддерживаемых форматов данных путём добавления конвертеров, сериализующих/десериализующих отдельные свойства или настройки целиком.

Поддержка извещений об изменении настроек. Настройки приложения обычно используются одновременно в нескольких его модулях. Изменение настроек (как внешнее — например, изменение файла на файловой системе — так и внутреннее изменение свойства одним из компонентов) может быть интересно зависящим от этих настроек компонентов. Библиотека должна поддерживать извещение об изменениях настроек — или в виде традиционного шаблона провайдер/слушатель, или более эффективные реализации на базе шины событий.

Поддержка структурированной информации. Для крупных приложений, число параметров конфигурации может быть достаточно велико — несколько сотен или даже тысяч настроек. Однако каждый индивидуальный модуль приложения, скорее всего, работает с небольшим количеством настроек. Разработчикам модуля было бы удобно, если бы библиотека управления настройками могла бы предоставить данному модулю только необходимые ему свойства, скрыв нерелевантные. Кроме того, для постоянных хранилищ, поддерживающих структурированные данные (а большинство распространённых видов хранилищ поддерживают такие данные), было бы удобно аналогично читать и сохранять настройки с полным или частичным сохранением структуры.

Более подробно требования к библиотеке управления настройками изложены в [5]. Библиотека с открытым исходным кодом *options-util,* разработанная на основании этих требований, описана в [6].

Литература

1. V. Denisov. Overview of Java application configuration frameworks. International Journal of Open Information Technologies ISSN: 2307-8162, 1(6), 2013. URL: http://injoit.org/index.php/j1/article/view/33
2. L. Koskela. Test Driven. Practical TDD and Acceptance TDD for Java Developers. Manning, Greenwich, CT, 2008
3. M. Rasmussen. (2013, Nov. 21). Java Build Tools: How Dependency Management Works with Maven, Gradle and Ant + Ivy. URL: http://zeroturnaround.com/rebellabs/java-build-tools-how-dependency-management-works-with-maven-gradle-and-ant-ivy/
4. RightScale (2015). Cloud Computing Trends: 2015 State of the Cloud Survey. URL: http://www.rightscale.com/blog/cloud-industry-insights/cloud-computing-trends-2015-state-cloud-survey
5. V. Denisov. Functional requirements for a modern application configuration framework. International Journal of Open Information Technologies ISSN: 2307-8162 3(10), 2015. URL: http://injoit.org/index.php/j1/article/view/236
6. V. Denisov. Annotations-driven configuration framework for Java applications. International Journal of Open Information Technologies ISSN: 2307-8162 3(10), 2015. URL: http://injoit.org/index.php/j1/article/view/237.

Леонов М.В.[1], Киселева Е.А.[2]

[1] Московский государственный университет имени М.В. Ломоносова, г. Москва, к.б.н., ведущий научный сотрудник факультета вычислительной математики и кибернетики, Leonow_M_W@cs.msu.ru

[2] Московский государственный университет имени М.В. Ломоносова, г. Москва, инженер факультета вычислительной математики и кибернетики, lenakisseleva@gmail.com

ОПЫТ ПРОГРАММИРОВАНИЯ ЭЛЕКТРОННЫХ СПРАВОЧНИКОВ В УСЛОВИЯХ «ПЛАНШЕТНОЙ РЕВОЛЮЦИИ»

КЛЮЧЕВЫЕ СЛОВА

Настольные информационные системы, планшеты, HTML5, Android, Windows, веб-приложение, электронные справочники, Web SQL.

АННОТАЦИЯ

В статье обсуждается опыт программирования электронных справочников в форме веб-приложений для браузера Google Chrome с использованием баз данных формата WebSQL, которые можно использовать как на ноутбуках под управлением операционной системы Windows, так и на планшетах под управлением операционной системы Android. Эти справочники и картотеки, разработанные как инструменты для автоматизации историко-архивных исследований, можно использовать и для накопления других тематических данных.

В последние годы вместо ноутбука – удобного и, казалось, незаменимого технического инструмента научного работника в библиотеках, архивах, командировках все чаще используется планшет с операционной системой (ОС) Android. Планшеты обладают существенным преимуществом – они, как правило, гораздо легче, чем ноутбуки. И хотя уже есть планшеты с ОС Windows, наиболее популярными считаются пока планшеты ОС Android , а перенос важных баз данных из одной среды в другую, как правило, не так прост, как хотелось бы. Миграция баз данных, в отличие от изображений и текстов, с компьютеров на планшет и обратно, вообще говоря, не является тривиальной задачей, особенно для пользователей – специалистов гуманитарного профиля.

Некоторые специалисты предметных областей серьезно опасаются этого нового этапа развития информационных технологий. По словам профессора М. Таллера из Кельнского университета, «надвигающаяся «мобильная революция» может вновь привести к повторению разрушительного процесса, происходившего в ходе ПК- и интернет-революций»[4]. Речь идет о том, что специалист, создавший в своей предметной области содержательную базу данных (БД), при переходе на новую программно-аппаратную платформу почти неизбежно сталкивается с задачей, получившей красивое название «проблемы унаследованных баз данных».

В лаборатории вычислительного практикума и информационных систем факультета ВМК в течение ряда лет ведутся разработки программ для автоматизации работы историков-архивистов. Ядром большинства этих проектов является та или иная БД, причем историки, как и другие пользователи, часто используют как планшеты с Android, так и настольные компьютеры с ОС Windows. Поэтому мы решили найти способ, который сделал бы процесс переноса БД между этими платформами максимально легким.

Решение пришло из области HTML5-технологий, одной из новинок которой была возможность создавать БД на стороне клиента – браузера с использованием технологии Web SQL[5]. Наиболее последовательно и симметрично по отношению к ОС Windows в Android поддерживали эту возможность браузеры Google Chrome, Safari и Opera. Мы выбрали в качестве базового инструмента для наших информационных систем браузер Google Chrome. Фактически наша информационная система – это Web приложение, написанное на языке JavaScript с использованием Web SQL Database APi и библиотек jquery и jqueryUI.

Один из первых результатов – электронная картотека членов Московского Общества испытателей природы (МОИП) до 1955 года, более подробно описанная в работе [2].

Эта информационная система имеет два режима: административный и читательский. В административном режиме можно вводить данные по персонам как в интерактивном режиме, так и в пакетном, а также редактировать их.

В читательском режиме в правом верхнем углу находится форма поиска по любой комбинации Фамилии, Имени, Отчества и году рождения. Можно также просматривать список фамилий, указав первую букву фамилии. В левой части окна программы расположены четыре вкладки: «Основная информация», «Дополнительная информация», «Карточки», «Источники». Первая вкладка, например, отражает значения следующих полей: фамилия, имя, отчество, дата рождения, дата смерти, основные занятия члена МОИП, год принятия в общество, и признак почетного члена. Предусмотрены также поля для фамилии и имени на иностранном языке.

Второй результат с использованием этой же технологии – это программа для управления коллекцией гравюр, хранящихся в Архиве МОИП. Основных требований к этому приложению было два: удобство просмотра изображений, и возможность вставлять в БД описания соответствующих гравюр, то есть информацию об изображенных персонах, о граверах и художниках. Второе требование было обусловлено необходимостью использовать программу для атрибуции гравюр, которую в за все время существовании коллекции еще никто не сделал. Для каждой электронной фотокопии гравюры в БД хранится фамилия и имя изображенного на гравюре человека, причем как оригинальном написании, так и на кириллице.

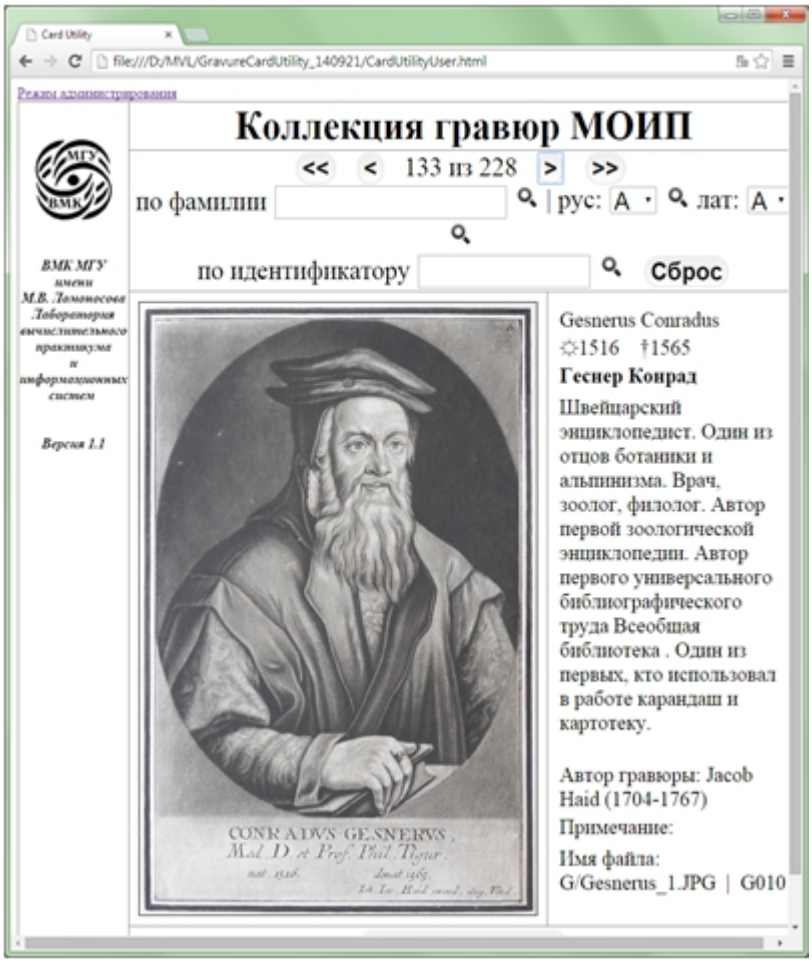


Рис.1. Страница просмотра коллекции гравюр программой на планшете

Создание пофамильных справочников – одна из весьма распространенных задач исследователя, работающего в архиве и библиотеках. В качестве примера можно упомянуть работу одного из авторов данной статьи по сбору данных для книги по истории елецкой мужской гимназии [3]. Трудно обойтись без рабочего электронного справочника, когда количество персонажей превышает несколько сотен. Довольно очевидно, что при разработке каждого следующей специализированного пофамильного справочника приходится повторно программировать очень сходные функции. Поэтому мы решили сделать утилиту с настраиваемым

интерфейсом для пофамильного справочника. При этом названия полей, а также визуализацию некоторых из них можно менять с помощью конфигурационного файла под названием SETTINGS.js. Можно менять количество, тип, порядок расположения на экране значений полей, а также название справочника на заглавной странице. Естественно необходимо наличие двух режимов: административного и пользовательского, а также пакетного ввода. Пакетный ввод облегчает первоначальный ввод, что особенно важно на этапе отладки, и кроме того, подготовленный пакет служит резервной копией данных в случае переноса данных на новый компьютер.

Из нашего опыта участия в историко-архивных проектах[1] мы знаем, что какой бы продуманной ни была структура БД, заполняемая в процессе архивных поисков, все равно в какой-то момент выясняется, что найденный материал не укладывается в «прокрустово ложе» структуры запланированной БД. С другой стороны, ясно, что этот материал может пригодиться в дальнейших работах, и хорошо бы его где-то хранить. Поэтому мы решили ввести возможность «прикрепления» файлов разного типа к записи, описывающей основной объект справочника – «персону». Эти прикрепляемые файлы, как правило, сканы листов из книг, или архивных документов и фотографий.

Далее кратко представим пофамильный справочник для проекта по истории елецкой гимназии (1871-1918 гг), созданный с помощью вышеупомянутой утилиты.

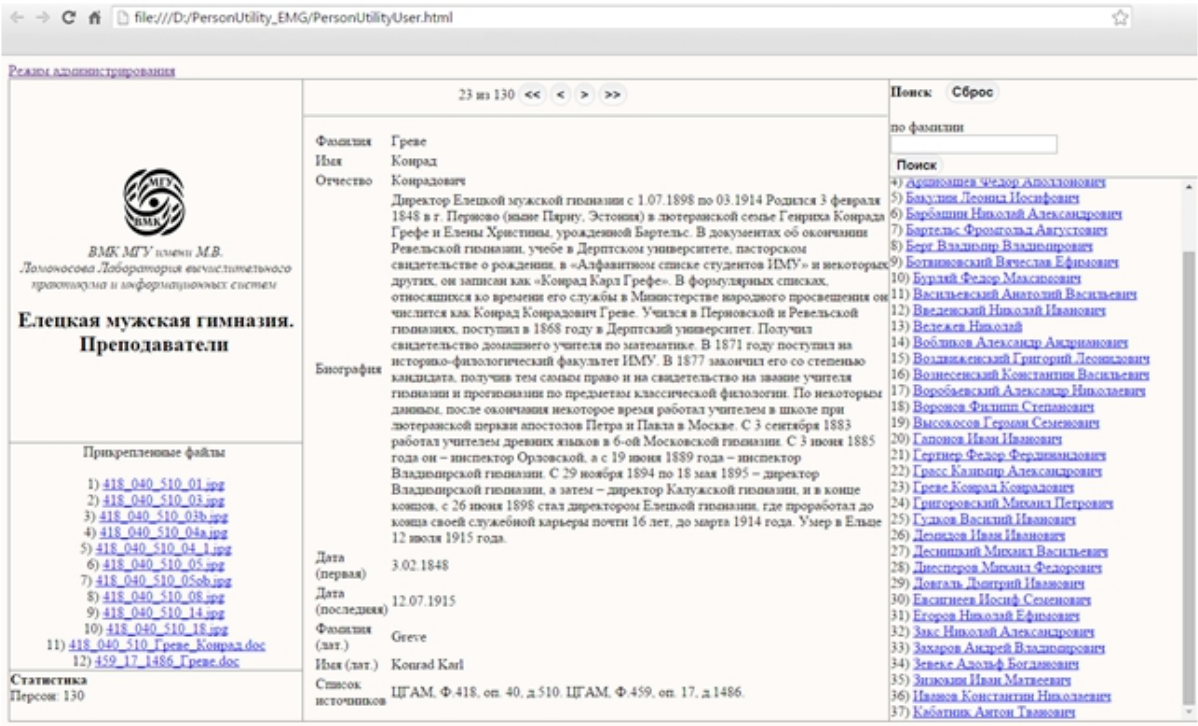


Рис.2. Пользовательский режим справочника «Преподаватели елецкой гимназии»

Пользовательский режим (см. Рис.2) предназначен для просмотра статей справочника (биографических и других данных по преподавателям) и поиска по фамилии. Экран разбит на 8 неравных ячеек. В верхней ячейке находится гиперссылка перехода в режим администратора. Левый столбец состоит из ячеек с логотипом и названием справочника, со списком прикрепленных файлов, и текущем состоянии статистики справочника. Щелчок по имени прикрепленного файла приводит к его открытию либо браузером, либо внешней программой, в зависимости от типа файла и настройки браузера.

В среднем столбце – вверху кнопки-стрелки листания справочника, а ниже основные данные по персоне, в данном случае преподавателю или работнику гимназии: фамилия, имя, отчество, краткая биография, дата первого упоминания, дата последнего упоминания, латинское написание фамилии и имени, список источников.

В правом столбце расположены кнопки управления поиском и поле ввода для фамилии, а ниже – список фамилий, причем щелчок по фамилии приводит к переходу к данным по соответствующему человеку.

В «режиме администрирования» администратор, кроме возможностей пользователя,

может также редактировать и удалять данные, а также редактировать интерфейс (см. Рис.3) Для этого предназначены кнопки «редактировать» (объект редактирования зависит от контекста), добавлять прикрепляемый файл (см. левый столбец экрана на рис. 3). Кнопка «Очистить БД» используется при переходе к новой структуре БД.

Добавление новых «персон» осуществляется как в интерактивном, так и в пакетном режиме. Нижняя ячейка экрана содержит кнопку для указания файла для пакетного ввода и его запуска. Файл пакетного ввода представляет собой текстовый файл в кодировке utf-8, содержащий значения полей БД по персонам через точку с запятой.

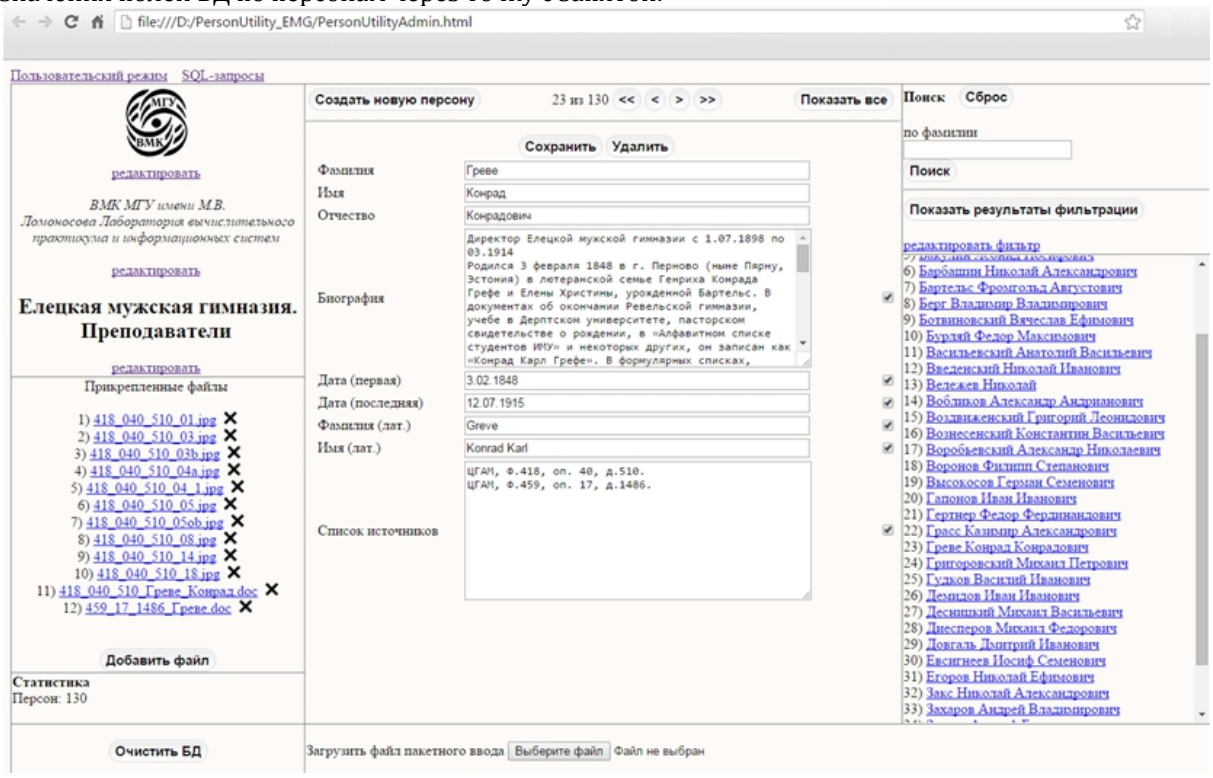


Рис.3. Режим администрирования справочника «Преподаватели елецкой гимназии»

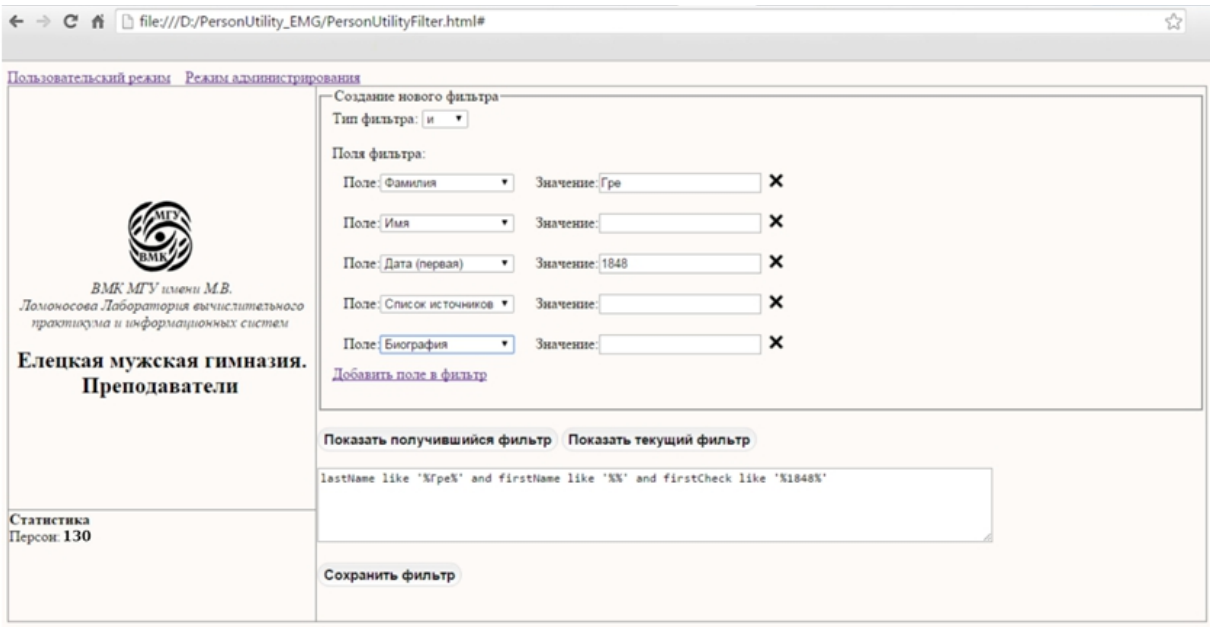


Рис.4. Экран настройки фильтра для просмотра в пользовательском режиме.

Справа от каждого поля данных записи можно установить или снять флажок (check box), отвечающий за отображение соответствующего поля в режиме просмотра. Если убрать

определенную галочку и нажать кнопку «сохранить», то в пользовательском режиме это поле и его значение не будут отображаться.

Так как предполагалось, что утилитой-справочником будут пользоваться не только «простые пользователи», но и «продвинутые» в компьютерном отношении, то введен еще один режим справочника, «режим SQL-запросов» (см. Рис.5).

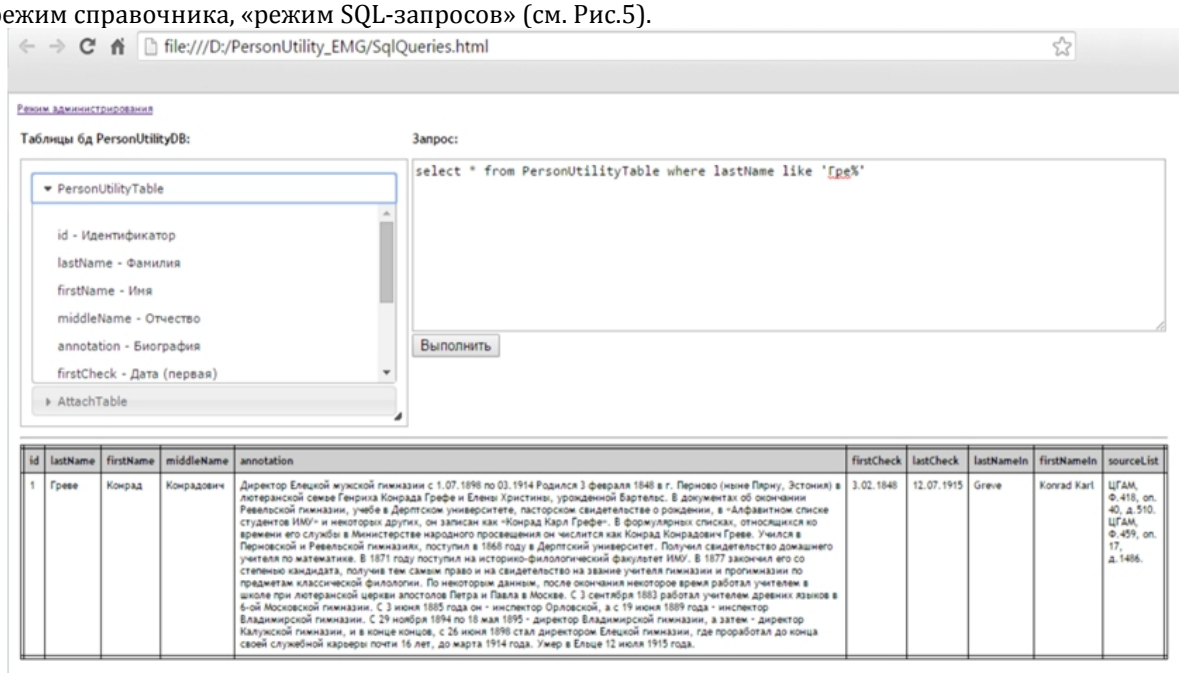


Рис.5. Экран режима SQL-запросов

Гиперссылка «Редактировать фильтр», расположенная в правом столбце экрана администрирования, ведет в режим настройки фильтра для просмотра в пользовательском режиме (см. Рис.4). Это дает возможность в пользовательском режиме работать не со всеми элементами справочника, а только теми, которые отвечают определенному условию. При этом получившееся условие можно посмотреть в виде SQL-запроса.

В этот режим можно переходить только из режима администратора. В правом центральном поле экрана можно задавать команду SELECT языка SQL , при этом в левом центральном в прокручивающемся окошке можно выбрать таблицу БД, а также увидеть названия полей, что удобно при формулировании запросов. Режим SQL запросов особенно важен для тех применений утилиты пофамильных списков, в которых, когда нужно формировать различные сводки по объектам справочника.

Заметим, что благодаря использованию технологии пакетного ввода, мы имеем возможность при необходимости изменять структуру базы данных, и потом без особых трудностей вводить пакет данных заново.

Описанные выше программные средства успешно используются в проектах по истории Московского университета, и постоянно совершенствуются в процессе эксплуатации. В качестве вывода можно сказать, что технология, основанная на применении браузера Google Chrome для реализации настольных информационные систем с базами данных формата SQLite[6] и возможностью несложного переноса между ноутбуками с ОС Windows и планшетами с ОС Android вполне себя оправдала.

Литература

1. Леонов М.В., Гутнов Д.А., Киселева Е.А. Электронная картотека елецких студентов Московского университета до 1917 г. //Информационный бюллетень Ассоциации «История и компьютер», № 42, 2014, С. 35-37.
2. Леонов М.В., Киселева Е.А., Орлов А.С. Опыт конверсии традиционных картотек в электронную форму на примере картотеки Московского общества испытателей природы // Историческая информатика. 2013. № 4. – С.22 – 30.
3. Саввина О.А., Леонов М.В. История Елецкой мужской гимназии. Люди и факты. Часть I: Преподаватели. – Елец: Елецкий государственный университет им. И.А. Бунина, 2015, 178 с.
4. Таллер М. Дискуссии вокруг Digital Humanities // Историческая информатика. 2012. № 1. – С.5 – 13. .
5. Web SQL Database URL: http://www.w3.org/TR/webdatabase/
6. Android - SQLite Database Tutorial URL: http://www.tutorialspoint.com/android/android_sqlite_database.htm

Игнатенков А.В.[1], Ольшанский А.М.[2]

[1]ФГБОУ ВПО «Самарский государственный университет путей сообщения», аспирант кафедры «Автоматика, телемеханика и связь на железнодорожном транспорте»; ООО «Научно-технологический центр по эксплуатации железных дорог», Москва, заместитель генерального директора по информационным технологиям, a.ignatenkov@gmail.com

[2]ФГБОУ ВПО «Самарский государственный университет путей сообщения», докторант кафедры «Автоматика, телемеханика и связь на железнодорожном транспорте»; ООО «Научно-технологический центр по эксплуатации железных дорог», Москва, кандидат технических наук, заместитель генерального директора по науке, lexolshans@gmail.com

ПРИМЕНЕНИЕ ИСКУССТВЕННОЙ НЕЙРОННОЙ СЕТИ ДЛЯ ПОСТРОЕНИЯ РАСПИСАНИЙ ПРОЦЕССОВ НА ПРИМЕРЕ ГРАФИКА ДВИЖЕНИЯ ПОЕЗДОВ

КЛЮЧЕВЫЕ СЛОВА

Нейронные сети, график движения поездов.

АННОТАЦИЯ

Рассмотрены вопросы построения искусственной нейронной сети для формирования расписания процессов. Описаны архитектура сети, методика расчёта выхода сети и обучение сети.

В настоящее время теория расписаний как часть более общей отрасли математики под названием «исследование операций» очень популярна как с практической, так и с теоретической точки зрения. Подобные задачи в общем виде являются NP-сложными, на практике они решаются с помощью одного из многочисленных методов либо их комбинаций [5]. Каждая из постановок задач характеризуется набором ограничений, а процесс составления расписаний сводится к поиску наиболее удовлетворяющего им решения. Часто этот поиск минимизирует некоторую целевую функцию, составленную из комбинаций различных условий. Таким образом, каждое решение задачи теории расписания подходит к конкретному классу требований.

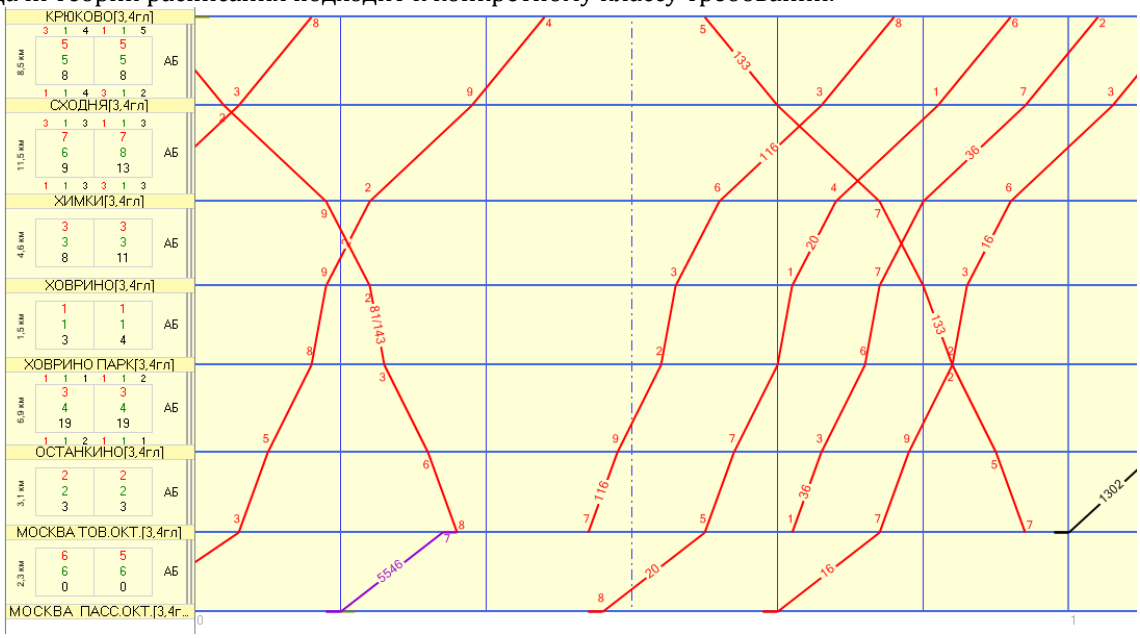


Рис. 1. Пример графика движения поездов

Целью данного доклада является реализация нейросетевого подхода к одному из классов задачи теории расписаний. Данный подход обусловлен такими свойствами нейронных сетей, как способность к самообучению и возможностью работы с малоформализованными данными [4]. Выбор такой области построения расписания, как график движения поездов, обусловлен

значимостью данной сущности для эффективного функционирования железных дорог [2].

График движения поездов является отображением зависимости *t*=f(s), где *t* – время движения поезда, s – координаты поезда на сети железных дорог. Пример графика приведён на рисунке 1.

Объектом исследования будет нейронная сеть, построенная следующим образом [1]. Каждый слой нейронной сети соответствует выбранной железнодорожной станции. В слое представлены нейроны, каждый из которых имеет номер от 0 до 1439. Из каждого нейрона i-го слоя идут связи к каждому нейрону соседнего слоя (всего 1440 связей). Кроме того, каждый нейрон связан с несколькими нейронами слева (т.е. с нейронами с меньшим номером) и справа (с нейронами с большим номером).

Фрагмент сети приведён на рисунке 2.

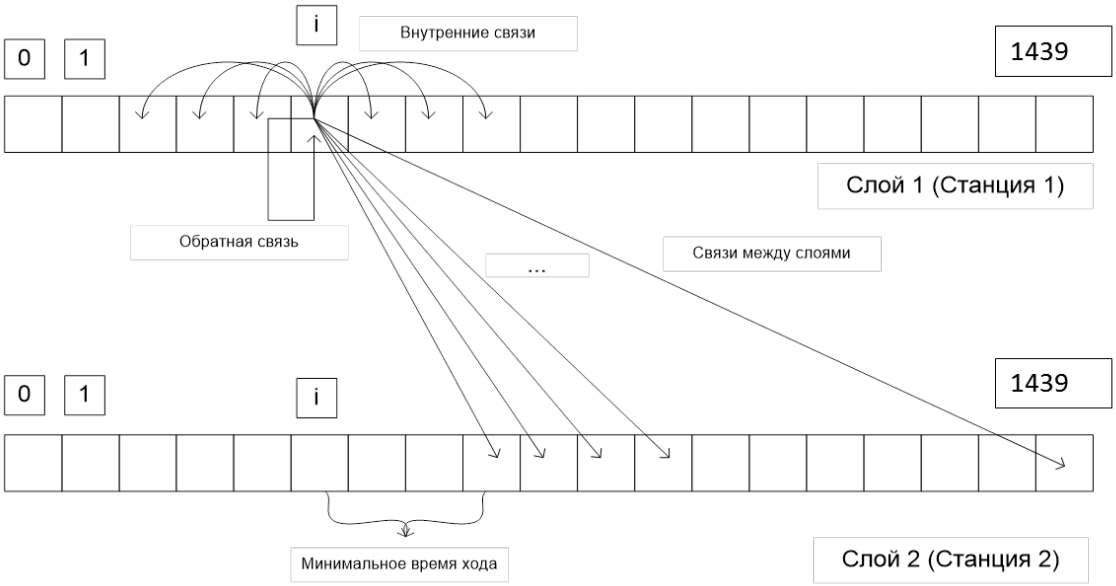


Рис. 2. Фрагмент искусственной нейронной сети

Каждая матрица весов W между двумя слоями с номерами *i*, *i*-1 представляет собой квадратную матрицу с количеством строк и столбцов, равным 1440.

$$W_{i,i-1}=\begin{pmatrix} w_{0,1} & \cdots & w_{0,1339} \\ \vdots & \ddots & \vdots \\ w_{1339,0} & \cdots & w_{1339,1339} \end{pmatrix}$$

где w_{ij} – значение веса на связи, соединяющей нейрон с номером *i* одного слоя с нейроном с номером *j* соседнего слоя.

Каждая строка представляет собой веса нейрона с фиксированным индексом. Номер столбца – номер нейрона предыдущего слоя, который связан с нейроном данного слоя.

Рассмотрим задание весов связей для прокладки нечётных поездов. Нейрон с номером *j* связан со всеми нейронами соседнего слоя. Учитывая, что при движении поезда между двумя станциями должно соблюдаться некоторое минимальное время хода *t*, от нейрона с номером 0 до нейрона с номером *j*+*t* все веса являются нулевыми. Все прочие (веса связей нейрона с номером *i* с нейронами последующего слоя с номерами от *j*+*t* до 1439) являются равномерно распределёнными случайными числами от 0 до 0,1.

$$\begin{pmatrix} 0 & w_{0,2} & w_{0,3} & \cdots & w_{0,1438} & w_{0,1439} \\ & \vdots & & \ddots & \vdots & \\ 0 & 0 & 0 & \cdots & 0 & w_{1439,1439} \end{pmatrix}$$

Сигнал, характеризующий конкретный поезд, выходит со слоя нейронной сети (из данной станции) из некоторого нейрона (в некоторую минуту) и попадает на предыдущий слой.

Передача сигнала по слоям.

Прямой проход работы сети выполняется следующим образом.

- На последний (первый) слой подаётся вектор $X=\{x_0, x_1, \ldots, x_{1439}\}$, который представляет собой последовательность нулей и единиц. Единица означает, что соответствующая минута связана с прохождением поезда. Эти же числа переходят с последнего (первого) слоя как выходные значения.
- Под X^l будем понимать вектор значений нейронов со слоя с номером l.
- Для всех значений, выходящих со слоёв с номерами $n, n-1, \ldots 1$ выполняется:

$\forall k \in (0,1,\ldots 1439)$ *если* $X_k^{(l)} > 0$, *то* $\exists j \in (0,1,\ldots 1439) : w_{k,j} = max\{w_{km}\}, X_j^{l-1} = 0$, *тогда*

устанавливаем $X_j^{l-1} := f(X_k^{(l)}, w_{k,j})$.

где

w_{km} – вес на связи между нейронами слоев *k* и *m*,

k – номер нейрона текущего слоя,

j – номер слоя, смежного с рассматриваемым,

Смысл описанного условия заключается в том, что для каждого нейрона с номером *k* текущего слоя с положительным значением выхода в следующем слое мы ищем такой нейрон, что значение веса их связи является максимальным из всех связей нейрона с номером *k* со следующим слоем.

Функция активации $f(x,w) = \begin{cases} 0, \text{если } x = 0, \\ \dfrac{1}{1+e^{-x*w}}, \text{если } x > 0 \end{cases}$, (1)

где

$f(x,w)$ – значение функции активации,

w – нейрон с максимальным весом, победивший в конкуренции нейронов,

x – вход, подаваемый на нейрон-победитель.

- Выходом сети *Y* примем вектор значений нулевого слоя $X_{\square}^{(0)}$.

Существует желаемый выход сети $Y_ц$, который представляет собой последовательность нулей и единиц, смысл которой идентичен смыслу входного вектора X.

Введём понятие «номер поезда», который обозначим как r. Мы говорим, что поезд проходит по станции с номером l_1 *в минуту* k_1 и по станции с номером l_2 *в минуту* k_2, если выполняется равенство: $r(X_{k_1}^{l_1}) = r(X_{k_2}^{l_2}) \quad \forall r \forall l \exists\, !k : r(X_k^l) = r$, т.е. все поезда проходят по каждой станции ровно в одной точке-минуте. По номерам поездов производится отслеживание прохождения поездов по всем станциям.

Обучение сети производится согласно следующему алгоритму.

Вычисление ошибки: ошибка сети вычисляется по формуле:

$$E = \sum_b (k_ц - k_Y)^2 + C, \quad (2)$$

где

b – число поездов, которые требуется проложить в графике движения.

$r(Y_{k_ц}) = r(Y_{k_Y})$ - номер поезда, для всех поездов, дошедших до нулевого слоя,

C – функция штрафа для недошедших поездов.

В зависимости от полученной функции ошибки, которую необходимо минимизировать, производится разнонаправленное изменение величины весов каждого задействованного нейрона.

Физический смысл обучения заключается в следующем: когда происходит уменьшение весов связей «слева» от связи с максимальным весом, тем самым уменьшается шанс пройти по уменьшенным связям сигналу нейрона при следующей попытке. «Справа» от этой связи, наоборот, производится увеличение значений весов.

Изменение весов производится пропорционально разнице между максимальным весом и весом текущей связи. После каждой эпохи (после нового вычисления значений сети) скорость обучения η увеличивается на значение, равное $\dfrac{\eta}{e}$, т.е.

$$\eta(e)=\eta_{e-1}+\frac{\eta_e}{e} \quad (3)$$

где e – номер эпохи.

Особенности построенной сети

Выше были рассмотрены принципы построения, функционирования и организации сети. Однако существует ряд дополнительных задач и свойств, вытекающих из особенностей сети и требующих дальнейшего исследования.

Ярко выраженной особенностью строения сети является наличие двойного набора весов, служащих для прокладки четных и нечетных поездов.

Спецификой предложенной конструкции является использование функции ошибки, которая в явном виде не связана со значением весов, так как вычисление наиболее вероятного пути прохождения поезда осуществляется по принципу мягкой конкуренции нейронов (выигрывает наиболее «тяжелая» связь из всех ненулевых весов выбранного нейрона, но веса остальных связей при этом не становятся нулевыми). В то же время, величина ошибки влияет на ширину пучка связей, которые будут подвергнуты обучению. Использование связей в качестве носителей «знаний» о состоянии сети является распространенной практикой, однако в предлагаемой сети происходит разнонаправленное обучение связей (по разные стороны от наиболее тяжелой связи).

Проверка сети

В этих условиях был проведён полный факторный эксперимент для данной нейронной сети, осуществлена серия опытов, и были получены табулированные значения функции ошибки сети в зависимости от влияющих факторов (числа эпох, скорости обучения и желаемой точности графика).

Границы исследования: скорость обучения менялась от 0,0005 до 0,0999 с шагом 0,0001, желаемая точность менялась от 5 до 100 с шагом 5. После 5000 эпох обучение прекращалось.

После всех проведённых экспериментов были выбраны те результаты, в ходе получения которых обучение прекратилось до 5000 эпох более чем в половине случаев для заданной скорости обучения и заданной точности.

Описанная авторами искусственная нейронная сеть реализует прокладку поездов для полигона, состоящего из 6 железнодорожных станций. Один из примеров построенного графика дан на рис. 3.

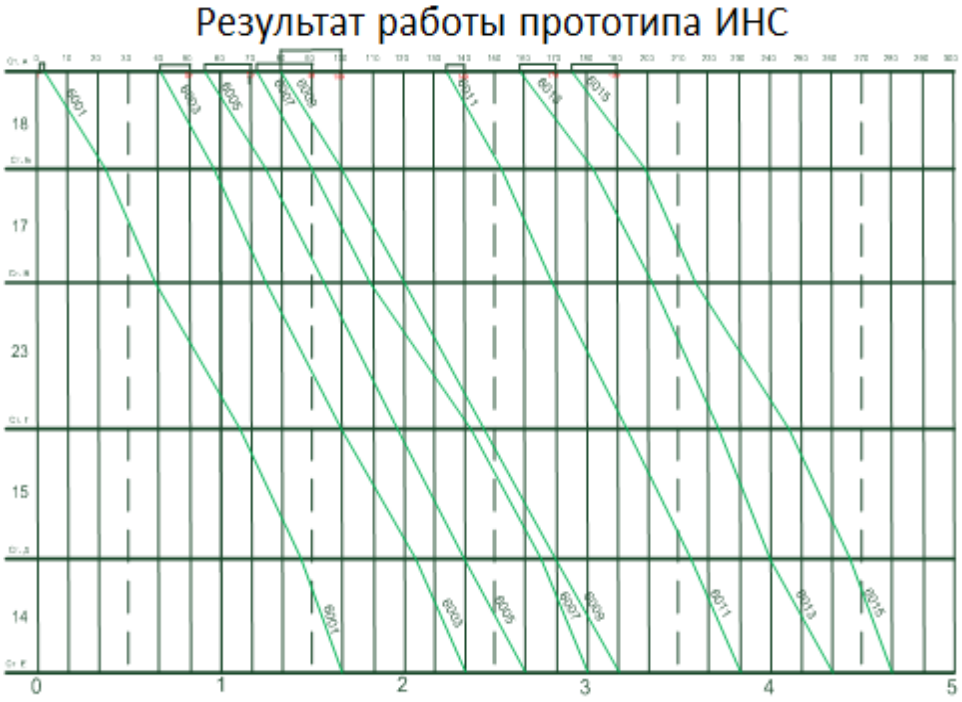


Рис. 3. Пример построения ГДП

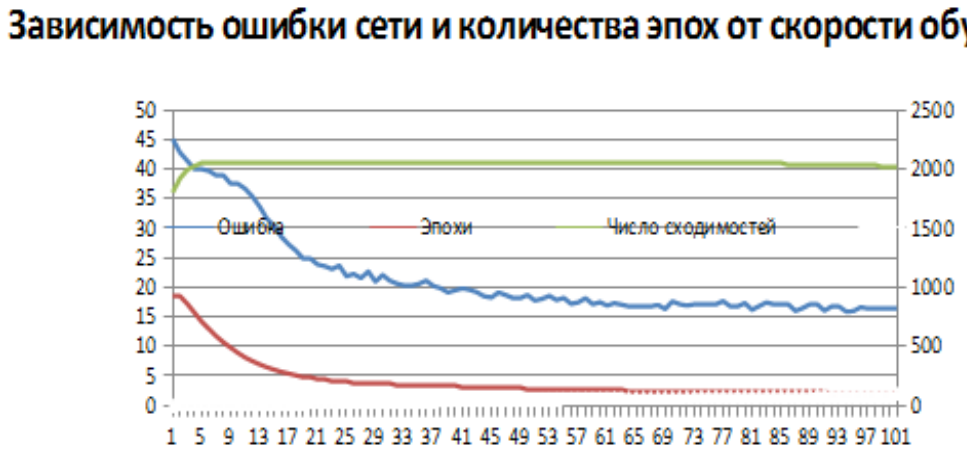


Рис. 4. Зависимость функции ошибки сети от различных факторов при использовании различных выборок

Также авторами был проведен расширенный эксперимент, в ходе которого обучение производилось на 21 выборке, каждая из которых моделировала различные требования к временным режимам движения поездов на учебном пятичасовом полигоне.

Было проведено 9898 прогонов сети. По результатам полученных результатов динамика сети характеризуется следующими данными (рис.4).

Общий вид функции ошибки сети представлен на рис.5.

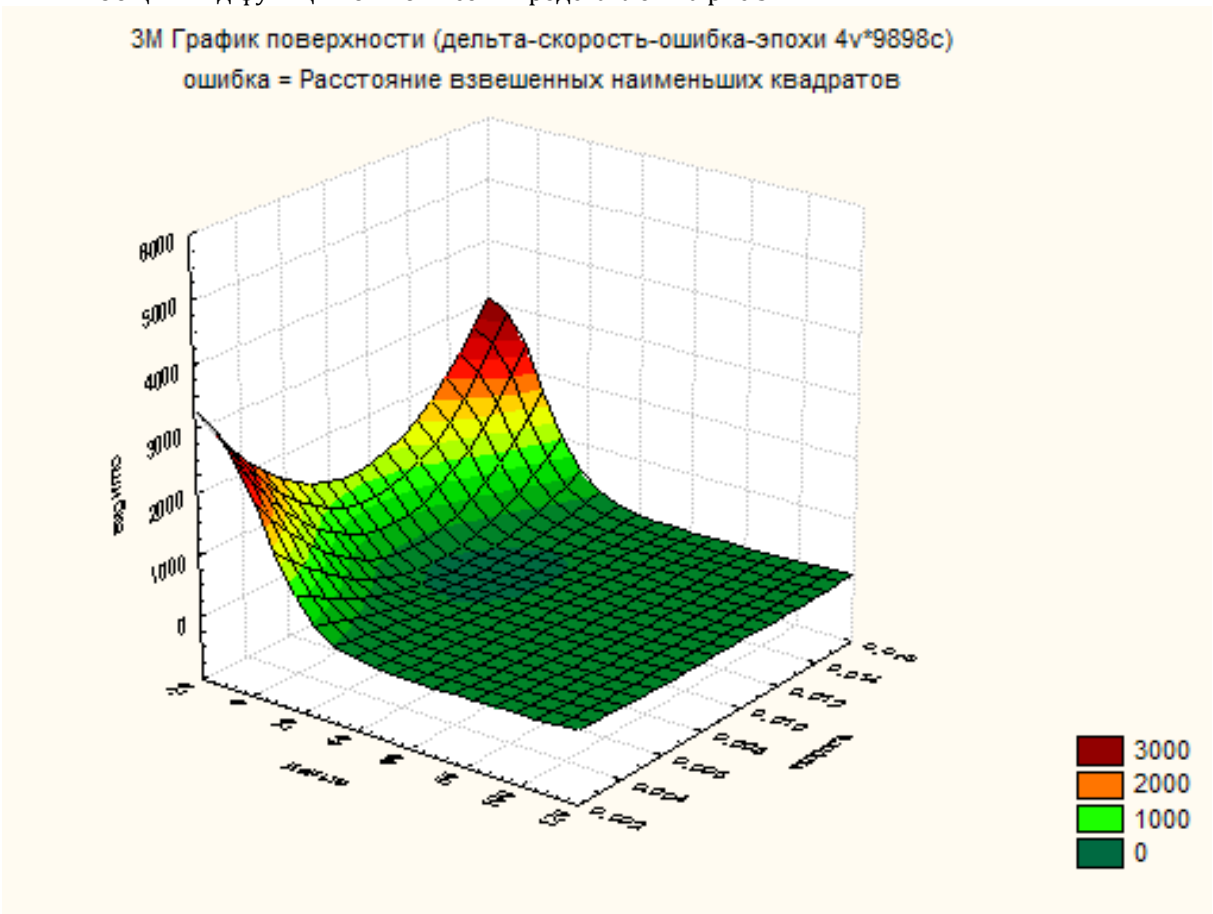


Рис. 5. Трехмерный вид функции ошибок

Таким образом, в представленном докладе авторами предлагается специализированная нейронная сеть, которая совмещает в себе признаки сети Хопфилда (преимущественно в части топологии), признаки многослойных персептронов (алгоритм обратного распространения

ошибки), а также специфические черты поведения.

Данная сеть является прототипом, иллюстрирующим применимость нейросетевого подхода к решению задач построения графика движения поездов. Развитием данной сети будет дальнейший учет особенностей эксплуатации железных дорог при разработке графика движения поездов.

Говоря шире, следует рассматривать не только задачу прокладки графика движения поездов, сколько задачу упорядочения некоторых процессов, понимаемых как длительности работ (потоков) с заданными моментами начала и окончания [3]. Таким образом, предлагаемая проблематика может затрагивать не только сегмент железнодорожного транспорта, но и смешанные транспортные системы.

Литература

1. Игнатенков А.В., Ольшанский А.М. О построении квазихопфилдовской искусственной нейронной сети для решения задач упорядочения процессов// В кн.: Перспективные информационные технологии (ПИТ-2015), том 2: труды Международной научно-технической конференции / под ред.С.А.Прохорова. - Самара, Издательство Самарского научного центра РАН, 2015. - 388 с., с.74-78 - ISBN 978-5-93424-735-7.
2. Бородин А.Ф., Шаров В.А. Интегрированная технология управления движением поездов по расписанию.//Железнодорожный транспорт. - №8. – 2011. – с.5-11.
3. Кормен Т., Лейзерсон Ч., Ривест Р., Штайн К. Алгоритмы. Построение и анализ. Издание второе. – Пер. с англ. – М., издательский дом «Вильямс», 2013. – 1296 с., ил. - ISBN 978-5-8459-0857-5 (rus).
4. Лазарев А.А., Гафаров Е.Р., Теория расписаний. Задачи и алгоритмы. Учебное пособие, Москва, 2012, с. 16.
5. Назаров А.В, Лоскутов А.И. Нейросетевые алгоритмы прогнозирования и оптимизации систем – Спб.: Наука и Техника, 2003. С. 228. ISBN 5-94387-076-8.

Ромасевич Е.П.

Волгоградский государственный университет, г.Волгоград, аспирант кафедры «Телекоммуникационных систем», eromasevich2@mail.ru

ПРОБЛЕМЫ МИГРАЦИИ СОВРЕМЕННЫХ СЕТЕЙ НА ПРОТОКОЛ IPV6

КЛЮЧЕВЫЕ СЛОВА

Протокол IPv6, протокол IPv4, миграция сетей, телекоммуникационные сети, QoS, Ipsec.

АННОТАЦИЯ

В работе рассматривается необходимость миграции современных сетей с протокола IPv4 на протокол IPv6, а также проблемы и темпы внедрения нового протокола. Приводятся существующие технологии перехода для взаимодействия сетей IPv4 и IPv6.

Введение

В основе работы современного Интернета лежит протокол сетевого уровня IPv4. Данный протокол был разработан более 30 лет назад и его спецификация описана в RFC791, в далеком 1981 году. Стек протоколов TCP/IP пришел на смену еще более старому протоколу NCP (Network Control Program), описанным в RFC33. Все соединения и способы связи маршрутизаторов, тогда еще, сети ARPANET – IMP (Interface Message Processor) описывались индивидуально [1]. Переход сети ARPANET с протокола NCP на стек протоколов TCP/IP произошел 1 января 1983 года. Именно тогда термин «Интернет» закрепился за сетью ARPANET [2].

Реализация IPv4 оказалась довольно успешной. Сеть ARPANET стала расти еще быстрее. В 1984году, наряду с сетью ARPANET, появилась университетская сеть NFSNet, которая по темпам роста обогнала ARPANET и полностью вытеснила последнюю в 1990 г. Версия 4 на самом деле была первой рабочей версией протокола IP (Internet Protocol) [3]. Во время разработки протокола IPv4 среди прочих, стоял вопрос о выборе размерности адресации. В IP адресе заложена информация как о номере сети, так и о номере конечного хоста. Сеть ARPANET была научно-исследовательской. Для целей научного эксперимента размерIPv4 адреса был выбран 32-битам. При полной утилизации всего адресного пространства IPv4 количество уникальных адресов составляет: 2^32 или 4 294 967 296. Для нужд эксперимента этого было вполне достаточно, чтобы адресовать каждый хост, подключенный к сети с большим, но обозримым резервом [2].

3 февраля 2011 года агентство IANA распределило последние 5 блоков/8 региональным интернет-регистраторам. Стала очевидной необходимость перехода на новый протокол IPv6, имеющий большее адресное пространство.

IPv6 имеет множество новых механизмов по сравнению со своим предшественником, включая улучшенное адресное пространство, встроенную расширяемость, и фактор наличия более сильного стандарта безопасности. По сути, он способствует формированию новых и более качественных сетевых наборов продуктов и услуг. Этот список не является исчерпывающим, но он даёт понимание, где IPv6 предстоит стать новым строительным блоком Интернет [4].

Необходимость миграции на протокол IPv6

То, что IPv4 имеет ограниченное адресное пространство, давно стало известно. В ближайшем будущем потребуется больше адресного пространства для новых портативных устройств в домах и офисах, чем то количество, которым мы сейчас располагаем. Количество пользователей растёт изо дня в день и уже сейчас миллиарды устройств подключены к Интернету. Процесс перехода на IPv6 находится в стадии разработки с 1990 года. Несмотря на то, что большое количество организаций получили всё необходимое для перехода, полное внедрение IPv6 идёт медленнее, чем ожидалось, хотя данный протокол поддерживает большее адресное пространство и лучшее качество услуг [5].

Количество конечных узлов в Интернете стремительно растёт с каждым годом, что приводит к значительному сокращению IPv4 и использованию IPv6. Протокол IPv6 иногда описывают как надстройка или модификация Ipv4. Но это принципиально другой протокол, который разработан для замены существующего протокола IPv4 в стеке TCP/IP.

Существуют некоторые сходства у двух протоколов, но различий больше. Присутствуют некоторые основные поля, такие как адреса источника и назначения и другие, но эти поля отличаются в размерах и формате. Адрес IPv6 имеет 128 бит, представленных в виде шестнадцатеричных блоков, тогда как адреса IPv4 имеют 32-битовые поля, представленные 4 октетами в десятичном формате [6]. Интернет протокол версии 6 поддерживает много возможностей и лучшую производительность.

Есть несколько важных причин для смены IPv4 на IPv6:

1. Нехватка адресного пространства IPv4;
2. Обработка мультимедийного трафика;
3. Мобильность;
4. Обработка пакетов [5].

Адресное пространство имеет 2^32 адресов. Но на деле огромное количество адресов невозможно использовать по различным причинам. Множество адресных блоков зарезервировано, и не предполагают глобальную маршрутизацию. Такие адреса описаны в RFC6890. Здесь присутствуют адреса для частных сетей, канальные адреса для автоматического назначения адреса по DHCP, для тестирования, примеров, многоадресной рассылки. Одни «петли» занимают почти 17 миллионов адресов. А сеть 240.0.0.0/4, имеющая более 268 миллионов адресов, не маршрутизируется большинством оборудования. Более того, провайдеры повсеместно модернизируют свои сети для предоставления новых сервисов, таких как VoIP, VoD, IPTV. Новые технологии появляются каждый год, и требуют большого числа IP адресов. IPv6 с огромным адресным пространством в 2^128 адресов способен выделить уникальные адреса всем существующим узлам, а также будущим намного лет вперёд [5].

Сегодняшним потребителям требуется больше мобильности с их мобильными устройствами и приложениями. Мобильность в IPv6 использует новые механизмы, благодаря чему переключение между станциями (handover) происходит быстрее, чем у IPv4, а качество связи лучше. Мобильная IP маршрутизация может иметь некоторые ограничения, но является сложной в исполнении и управлении [5].

IPv6 имеет возможностью динамического исследования MTU (максимальный блок передачи) пути следования трафика. Так фреймы создаются размером, определяемым маршрутизатором-отправителем, согласно минимальному MTU в каналах следования. Этим устраняется необходимость исчерпывающей фрагментации центрального процессора и процесса сборки в промежуточных маршрутизаторах, делая обработку пакетов быстрой и плавной [7].

Проблемы внедрения IPv6

При переходе с IPv4 на IPv6 придётся столкнуться не только с техническими проблемами. Существует множество других проблем [5].

Миграция на IPv6 началась медленно, потому что изначальная подключаемость была ограничена, магистральные сервисы с IPv6 не были распространены, была нехватка точек обмена IPv6.

До 2010 года только малая часть подключений приходилась на IPv6. Некоторые организации модернизировали свои сети до IPv6, но не отказывались от IPv4 полностью [5]. Результаты исследований последних лет показывают, что IPv6, хоть и незначительно, но уступает в производительности протоколу IPv4, тем самым оказывая негативный эффект на введение протокола IPv6 в эксплуатацию [9].

Обычные пользователи не задумываются о переходе на IPv6. Им просто необходим доступ в Интернет, независимо от того, какой интернет протокол используется. Все публичные интернет-сервисы доставляются потребителям через интернет протокол версии 4, поэтому пользователи не понимают, что есть необходимость миграции на шестую версию. Все устройства, которые они обычно используют у себя дома, такие как камеры, игровые консоли и телевизоры, поддерживают IPv4. В такой ситуации переход сервис-провайдеров на IPv6 был бы не приемлем для пользователей, потому что они не хотели бы лишаться привычных им сервисов [5].

Также не существует конечного срока, определяющего, когда переход должен быть завершён. Нет решения или закона о каком-либо конечном сроке перехода [5].

Влияние IPv6 на развитие перспективных сегментов рынка телекоммуникаций

Одной из тенденций телекоммуникационной отрасли является быстрый рост мобильных услуг, благодаря взаимосвязанному оборудованию сотовых сетей. Поэтому есть возрастающая необходимость в постоянной инфраструктуре и протоколе, который будет поддерживать мобильность и может связать огромное количество устройств. Это позволяет связи от узла к узлу с

адресацией IPv6 не нуждаться в каком-либо специальном устройстве, и неважно как эти узлы подключены [4].

VoIP создавался с успехом с точки зрения признания технологии. Движение от H.323 к SIP позволило более энергично использовать VoIP. Более того, вид передачи в VoIP сервисах, происходящий по сети, намного разнообразнее, включая данные, голос и видео. Возможность доступа к контенту на любом типе платформ очень привлекает потребителей, главным образом тех, кто использует мобильные устройства. IPv6, с увеличенным адресным пространством, возможностью огромного многоадресного пространства, и привлекательностью для SIP, содействует росту этих возможностей [4].

Способ взаимодействия с семьёй и друзьями изменился с написания писем на телефонные звонки, электронную почту, смс и мгновенные сообщения. Стремление передавать фотографии, вести обсуждения в изолированных P2P сетях, показывать частную информацию в Интернете, находить подходящие сообщества, или играть вместе с другими требует Интернет, который будет гибким и защищённым. IPv6, с его способностями к самоконфигурации и предоставлением протокола безопасности IPSec на уровне стека IP, будет адекватным инструментом для разрешения вышеперечисленных требований [4].

Сенсорные сети – это новый концепт на рынке. Они могут быть применимы в инженерной аппаратуре, крупных машинах, системах безопасности, а также в системах отопления, вентиляции, и воздушного кондиционирования [5]. Новая идея основывается на объединении всей этой техники на единой платформе. Использование систем для обнаружения загрязнения или радиации в воде, систем фильтрации воздуха, а также различных систем подобного типа становится все чаще. Для запуска и управления таким большим числом систем необходимы безопасные и защищённые системы. IPv6 предлагает устоявшуюся приспосабливающуюся платформу, которая поддерживает гибкость и одноранговые сети [8].

Технологии перехода

Существует ряд технологий перехода для взаимодействия сетей IPv4 и IPv6. Протокол ISATAP (Intra-Site Automatic Tunnel Addressing Protocol) используется в частных сетях. ISATAP предполагает использование только в частных сетях. Он не может использоваться в Интернете. Чтобы использовать обе версии протокола, необходимо установить маршрутизатор ISATAP между сетями. Принцип работы данного механизма заключается в назначении узлам IPv6 адреса, сформированного определённым образом. Такой адрес включает в себя уникальную часть для каждого узла и фрагмент, который содержит информацию о том, что работает ISATAP, и IPv4 адрес маршрутизатора ISATAP.

Если планируется доступ в Интернет, возможно использовать такую систему, как 6to4. 6to4 является протоколом перехода, разработанным, чтобы помочь в переходе на IPv6. Тогда как Teredo применяется, когда компания использует устройство NAT. Протокол использует внешние сервера, которые многие интернет-провайдеры разместили в Интернете для помощи с переходом.6to4 разработан, как переходный протокол для помощи по внедрению IPv6. Он не проектировался, как длительное решение. 6to4 позволяет существовать сетям IPv4 в виде туннелей для трафика IPv6. Данные переходные технологии необходимы из-за несовместимости заголовков IPv4 и IPv6. 6to4 использует подход, близкий к ISATAP. Часть адреса IPv6 содержит адрес IPv4. Teredo – протокол, предлагающий инкапсуляцию протоколу IPv6 для последующей передачи по IPv4 сетям [5].

Можно выделить несколько способов перехода [9]:

1. Dual-stack;
2. Туннели IPv6 over IPv4;
3. Протоколы перехода;
4. Специальные каналы данных;
5. Магистрали MPLS.

Технология двойного стека является наиболее распространённым механизмом на сегодняшний день. Данный механизм поддерживает IPv4 и IPv6 в маршрутизаторах и хостах [9] (рис. 1 слева).

Туннелирование осуществляет инкапсуляцию пакетов IPv6 в пакеты IPv4 (рис.1 справа).

Динамика внедрения IPv6

Два эксперимента, направленных на скорость внедрения протокола IPv6 и пропускную способность TCP, было проведено [9]. В ходе данного эксперимента исследователи проводили мониторинг пингов и трассировки маршрутов более чем семи тысяч университетских веб-серверов на протяжении более месяца. Серверы находились по всему миру. Во втором

эксперименте проводились измерения пропускной способности TCP пяти университетских веб-серверов, поддерживающих механизм двойного стека, в течение двух недель.[9]

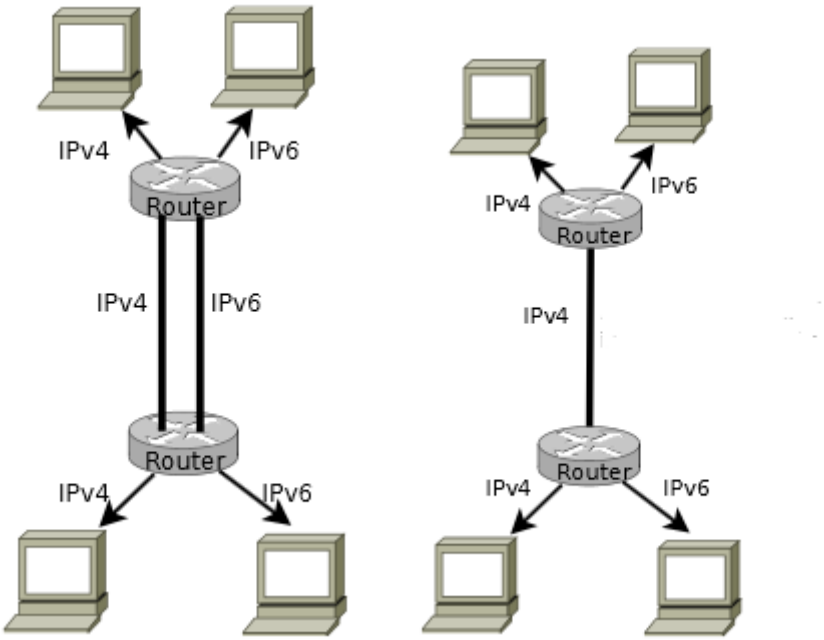


Рис. 1 – Схема реализации механизма двойного стека (слева). Схема реализации туннелирования (справа)

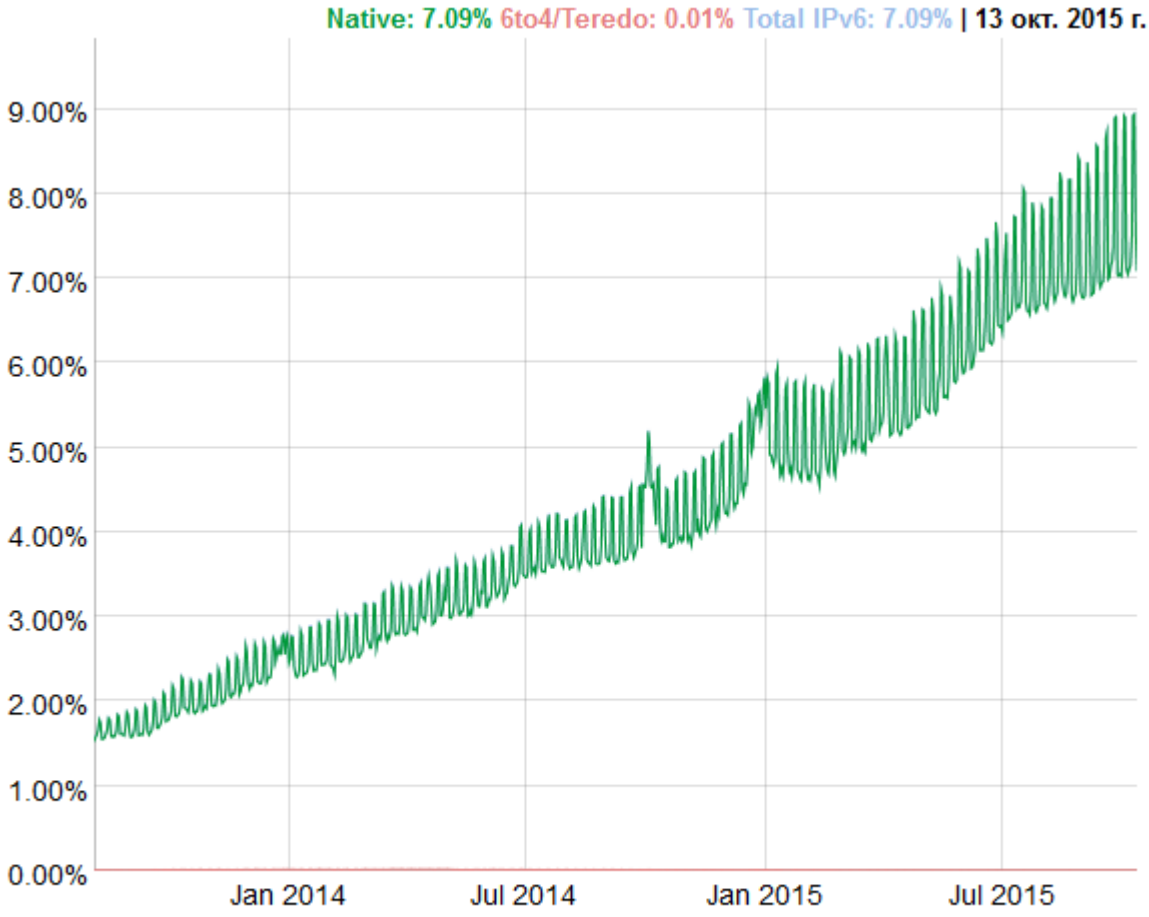


Рис. 2 – Данные по внедрению протокола IPv6

Результаты данного исследования показали, что внедрение IPv6 имеет незначительный

рост, а пропускная способность варьируется незначительно в пределах от 1,63% до 6,65% по данных с пяти серверов [9].

Более явную картину показывает исследование Google по данным на октябрь 2015, как показано на рис. 2 [11].

Из рисунка видно, что с июля 2013 года по октябрь 2015 года количество запросов к серверам Google по протоколу IPv6 выросло с 1,5% до 9% от общего числа запросов. Это может свидетельствовать о хорошей динамике внедрения IPv6. К сожалению, если смотреть по странам, то значительный положительный сдвиг (от 5% до 20%) присутствует лишь в США и некоторых странах Европы и Южной Америки.

Заключение

Современной тенденцией развития телекоммуникационных сетей станет переход Интернет на протокол IP версии 6, очевидным преимуществом которого является увеличенное адресное пространство по сравнению с IPv4. Если у протокола IPv4 – 2^32 адресов, то у IPv6 – 2128 адресов. IPv6 содержит немало и функциональных улучшений, прежде всего в области маршрутизации. Адресация теперь имеет иерархическую структуру, что облегчает передачу пакетов по сети. Также на уровне IP больше нет подсчёта контрольных сумм, что позволяет маршрутизаторам не разбивать пакеты, экономя время обработки. Появились также новые возможности QoS и многоадресное вещание, а IPSec стал обязательным. Максимальный размер пакетов у шестой версии протокола может достигать 4 Гб, что несомненно приведёт к изменениям принципов передачи данных в будущем [10].

Целью данной работы было рассмотрение проблемы миграции сетей и Интернета в целом с протокола IPv4 на протокол IPv6, способы передачи данных в условиях перехода с одного протокола на другой. Также были описаны некоторые причины, благодаря которым внедрение IPv6 является долгим. Приведены результаты работ нескольких исследователей данной области.

Материал данной статьи прямо связан с научными интересами автора, некоторые результаты которых по данной тематике были опубликованы в сборниках Международной научно-практической контференции «Современные информационные технологии и ИТ-образование» в 2013 [12] и 2014 годах [13].

Литература

1. Interface Message Processor. Report No 1822. URL: http://bitsavers.trailing-edge.com/pdf/bbn/imp/BBN1822_Jan1976.pdf
2. Необходимость внедрения протокола IPv6 сервис-провайдерами. Михлевич Д.С., Подсумкова Л.А. Актуальные вопросы технических наук в современных условиях / Сборник научных трудов по итогам международной научно-практической конференции. № 2. Санкт-Петербург, 2015. 231 с.
3. Внедрение IPv6 является сегодня первоочередной задачей для всего Интернета. Андрей Робачевский. URL: http://spbit.ru/interview/i86938/ 18.10.2015
4. TechTarget, 2010. IPv4 or IPv6 - Myths and Realities. URL: http://media.techtarget.com/searchNetworking/downloads/IPv4_or_IPv6.pdf 18.10.2015
5. Migration from Internet Protocol Version 4 To Internet Protocol Version 6. Amjad Mithani. Linneæus University. 2014-11-24
6. Wendell Odom; “CCNP ROUTE 642 - 902” Official Certification Guide; 2010; San Francisco, Cisco Press
7. Siddharta Saha. URL: http://siddhartasaha.weebly.com/uploads/3/7/7/9/3779388/ipv6_writeup.pdf 18.10.2015
8. Monteiro, Ole Hanseth and Eric Monteiro, 1998. Changing infrastructures: The case of IPv6. URL: http://heim.ifi.uio.no/~oleha/Publications/bok.a.html 18.10.2015
9. Measuring IPv6 adoption rate and performance in the Internet. Ali Emre Yildirim. Oslo University 24.05.2011
10. Дериев И. IPv6, не дожидаясь провайдера. 28.07.2011. URL: http://www.ixbt.com/soft/ipv6.shtml 18.10.2015
11. URL: http://www.google.com/intl/en/ipv6/statistics.html
12. Исследование процесса потери пакетов при переходе сети MetroEthernet смешанной архитектуры на протокол IPv6. Ромасевич Е.П., Ромасевич П.В. Современные информационные технологии и ИТ-образование / Сборник избранных трудов VII I Международной научно-практической конференции. Под ред. проф. В.А. Сухомлина. - М.: ИНТУИТ.РУ, 2013. – 829 с.
13. Исследование влияния передачи трафика IPv6 на работоспособность сети MetroEthernet на основе имитационной модели. Ромасевич Е.П. Современные информационные технологии и ИТ-образование [Текст] / Сборник избранных трудов IX Международной научно-практической конференции. Под ред. проф. В.А. Сухомлина. - М.: ИНТУИТ.РУ, 2014. – 957 с.

Ромасевич П.В.

Волгоградский государственный университет, г.Волгоград, к.т.н., доцент кафедры «Телекоммуникационных систем», promasevich@dlink.ru

ОЦЕНКА ОБЩЕГО ОБЪЕМА ПАМЯТИ ВВОДА-ВЫВОДА ТЕЛЕКОММУНИКАЦИОННОЙ СИСТЕМЫ В УСЛОВИЯХ САМОПОДОБНОГО ТРАФИКА И ПОТЕРИ ПАКЕТОВ

КЛЮЧЕВЫЕ СЛОВА

Телекоммуникационная сеть, телекоммуникационная инфраструктура, буферная память, память ввода-вывода, коммутатор, маршрутизатор, самоподобный трафик, коэффициент использования, вариация трафика, параметр Херста.

АННОТАЦИЯ

В статье получена формула оценки общего размера памяти ввода-вывода многопортового телекоммуникационного устройства с учётом максимальных скоростей и степени загрузки интерфейсов, потери пакетов, а также самоподобия и вариации трафика как функции пиковой скорости и объема всплеска трафика показана нелинейная зависимость от объема буферной памяти интерфейсов.

Введение

В связи с динамичным ростом количества устройств в публичных сетях, рынка телекоммуникационных услуг и «облачных» сервисов, технология предоставления которых предполагает удаленное и зачастую динамически меняющееся местоположение контента, важнейшее значение имеет непрерывное их обеспечение при разумных финансовых затратах и в связи с этим автоматически встает задача не только кардинальной модернизации телекоммуникационной инфраструктуры, но и предварительной оценки её ключевых параметров, ответственных за качественное предоставление сервисов.

Одним из таких параметров наряду с задержкой является буферная память телекоммуникационной системы, размер которой прямо влияет на процесс потери пакетов и, как следствие, на качество обслуживания трафика и предоставление услуг.

Зачастую, при построении моделей телекоммуникационных систем принимается допущение, что пакеты в системе не теряются [6]. Это предполагает бесконечность буферной памяти интерфейсов ввода/вывода, что на практике невыполнимо.

Многочисленные зарубежные и отечественные исследования последнего десятилетия также показали, что трафик в сетях передачи данных проявляет свойства самоподобия [4]. Эффект самоподобия трафика оказывает негативное влияние на производительность сетей передачи данных ввиду значительно большей потребности в буферной памяти телекоммуникационных систем, что является одним из основных факторов, влияющих на процесс потери пакетов, и, как следствие на качество обслуживания.

Как показано в [1], [2] и [3], предположение о независимости интервалов между поступающими пакетами и их длинами действительно имеет место в телекоммуникационных сетях с большим количеством интерфейсов и позволяет произвести декомпозицию телекоммуникационных устройств на элементы «канал+интерфейс», рассматривая их отдельно, найденные параметры которых являются слагаемыми для получения общего результата [6]. Это также справедливо с точки зрения аппаратной архитектуры большого числа сетеобразующих устройств. Со своей стороны, примем допущение, что данные результаты распространяются на сети с самоподобным трафиком.

Одной из актуальных проблем эксплуатации сетей пакетной коммутации является неустойчивая работа телекоммуникационных устройств в условиях интенсивного трафика. Это особенно характерно для коммутаторов L3 и маршрутизаторов уровней агрегации и ядра крупных телекоммуникационных сетей, через которые проходят большие объемы транзитного трафика.

Интересное исследование приводит университет Мичигана по причинам отказов современных IP-сетях (Рис.1):

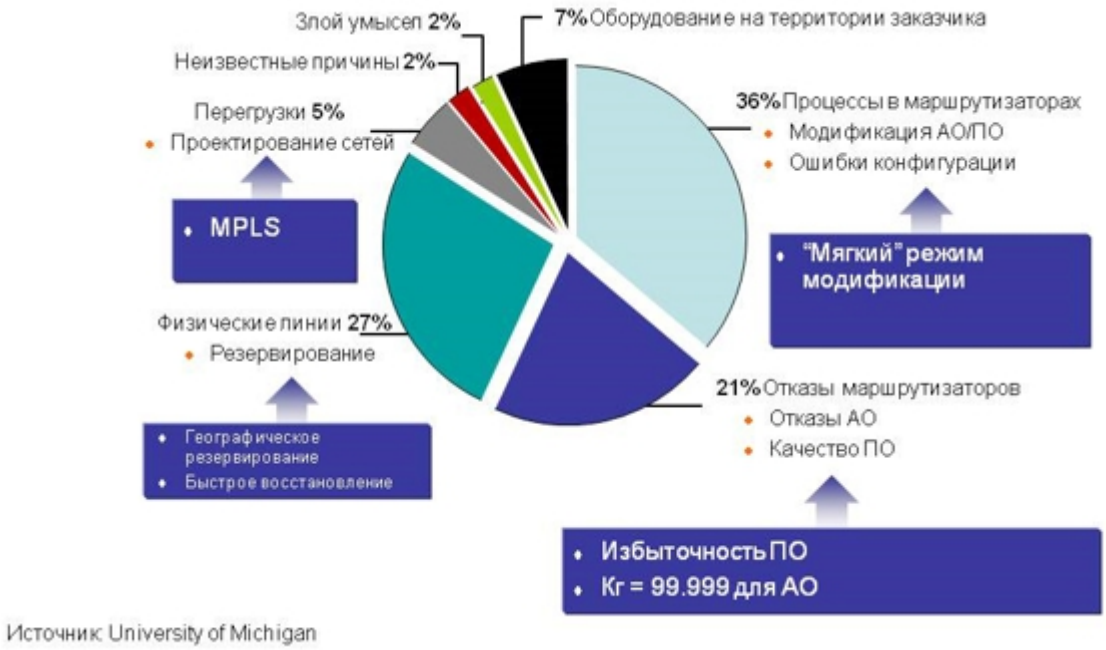


Рис.1 Причины отказов в современных IP-сетях

И одним из наиболее существенных факторов отказов здесь упоминается не только модификация аппаратного и программного обеспечения, которые достаточно быстро исправляет производитель, но и человеческий фактор - ошибки конфигурации, в том числе выделения необходимого объема буферной памяти в конкретных условиях эксплуатации – интенсивности потоков трафика, его статистических свойств, необходимого уровня качества обслуживания (QoS) и т.д.

С этой оценкой достаточно хорошо согласуется информация известного аналитического агентства Gartner Group, которая также отдает в указании причин системной ненадежности значительную долю «ошибкам эксплуатации», что в основном, по значительному производственному опыту автора статьи, происходит от неадекватного конфигурирования телекоммуникационного оборудования (Рис.2).

Поэтому оценка величины необходимой буферной памяти телекоммуникационных систем в условиях самоподобного трафика с учётом различного количества интерфейсов и места в сетевой иерархии является важной прикладной задачей.

При этом с точки зрения эксплуатации телекоммуникационных сетей, которые, как правило, построены на разном оборудовании с различными параметрами, на практике важна именно оценка границы значений общего объема буферной памяти всех интерфейсов узла сети, позволяющая инженеру службы эксплуатации сделать соответствующие действия по конфигурации сетевого оборудования.

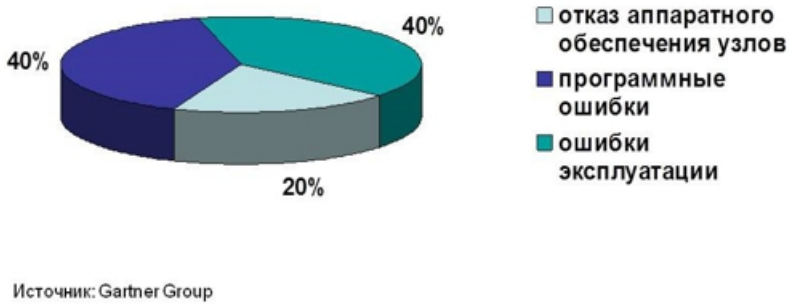


Рис.2. Причины системной ненадежности телекоммуникационных сетей

В этой связи целью данной работы является получение выражений для оценки общего объема буферной памяти телекоммуникационного устройства в условиях самоподобия трафика с учетом потери пакетов и количества сетевых интерфейсов.

Оценка памяти ввода-вывода многопортового телекоммуникационного устройства

Как было отмечено выше, одной из причин неустойчивой работы коммутаторов L3 и маршрутизаторов часто является их настройка, не соответствующая реальной сетевой ситуации, т.е. объем оперативной памяти и ее деление на основную область и память ввода/вывода, не соответствуют уровню интенсивности проходящего через устройство трафика. И наиболее часто это проявляется на магистральных узлах, через которые пользователи получают доступ к «внешнему миру» и трафик через которые формируется большим количеством источников.

В своей многолетней производственной деятельности автор неоднократно сталкивался со случаями и экспериментально исследовал случаи неуправляемости маршрутизаторов по сети, непредсказуемые резкие колебания загрузки их центральных процессоров и нехватки ресурсов оперативной и буферной памяти, что вело к «зависанию» маршрутизаторов, default-маршруты которых во «внешний мир» указывали на интерфейсы с множественным доступом, которыми в современных сетях является Ethernet [10].

Когда стандартный (default) маршрут указывает на интерфейс с множественным доступом (т.е. имеющим адрес канального уровня модели OSI), то даже при подключении «точка-точка», данным интерфейсом формируется ARP-кэш больших размеров, ввиду того, что через данный интерфейс осуществляется доступ к огромному количеству сетей «внешнего мира». Рост размеров ARP-кэша приводит к увеличению времени разрешения IP-адресов, дополнительной памяти на сам кэш и загрузке процессора. Поскольку маршрутизирующее устройство периодически освобождает место в ARP-кэше для размещения информации о результатах новых ARP-запросов, это дополнительно приводит к росту загрузки CPU, т.к. первая коммутация пакета с новым адресом назначения осуществляется чисто программно. И чем больше трафик через данный интерфейс, тем больше пакетов приходится коммутировать программно ввиду отсутствия записей об их адресатах в ARP-кэше. Как следствие, часто возникает ситуация, приводящая к полной очистке ARP-кэша ввиду отсутствия места для всех необходимых записей. Это также приводит к чрезмерному расходу ресурсов CPU и может повлечь неуправляемость и неработоспособность устройства, что в итоге делает участки сети неработоспособными.

Необходимо отметить, что эффективный функционал противодействия данной проблеме имеется в коммутаторах L2 и L3 компании D-Link и называется SafeGuard Engine [5]. Однако подобные средства есть не у всех производителей оборудования, поэтому правильная «ручная» конфигурация памяти является актуальной.

В [10] автором была решена данная практическая задача для определенного сетевого оборудования, а в [11] предложена общая формула расчета буферной памяти телекоммуникационного устройства с множеством интерфейсов с учетом самоподобия трафика и потери пакетов:

$$M = \frac{1}{1024} \sum_i MTU \sum_{j=1,2} \left(\frac{a^{\frac{1}{2(1-H)}} f^{-1}(\varepsilon)^{\frac{H}{1-H}}}{C^{\frac{2H-1}{2(1-H)}}} \frac{\rho^{\frac{1}{2(1-H)}}}{(1-\rho)^{\frac{H}{1-H}}} - \rho \right)_{i,j} \quad (1)$$

где i – количество интерфейсов, а параметры a, C, ε, f, p, H и MTU специфичны для каждого интерфейса. Индекс суммирования j определяет работу интерфейса в режиме полного дуплекса.

В [8],[9] представлено описание вероятностного закона превышения длины очереди над размером буферной памяти одноканальной телекоммуникационной системы для самоподобного трафика, что фактически и является вероятностной функцией потери пакетов, которая может быть выражена распределением Вейбулла:

$$f^{-1}(\varepsilon) \approx -\ln\left(\frac{(C-m)^{2H} \cdot x^{2-2H}}{2 \cdot H^{2H} \cdot (1-H)^{2-2H} \cdot a \cdot m}\right) \quad (2)$$

где

C – максимальная скорость передачи данных интерфейса;

H – параметр Херста;

a – коэффициент вариации трафика;

m – средняя скорость трафика;

x – объем памяти ввода/вывода одноканальной телекоммуникационной системы.

Это также подтверждено в [12], и показано, что модели трафика с долговременной зависимостью (в частности, модели, основанные на фрактальном броуновском движении) приводят к асимптотике распределения вероятностей хвостов вейбулловского типа, т.е. $P(Q > B)$ -

вероятность того, что длина очереди *Q* больше размера выделенной буферной памяти *B:*

$$P(Q > B) \sim \exp(-(\ \gamma B\ )^{2-2H}) \tag{3}$$

Коэффициент вариации трафика *a* можно легко вычислить через его пиковую скорость и объем всплеска следующим образом [7]:

$$a = \frac{h^{2H-1} \cdot (\frac{2-2H}{3-2H} \cdot b)^{2-2H}}{(3-2H) \cdot (2H-1) \cdot H}\ , \tag{4}$$

где:

h – скорость всплеска трафика;

b – средний объем всплеска трафика.

Необходимо отметить, что данный вид распределения, по мнению автора, адекватен практической ситуации, т.к. всегда имеет конечную дисперсию, чего нельзя сказать про наиболее популярное для моделирования самоподобных процессов распределение Парето.

Подставляя (2) в (1), где $\rho = \frac{m}{C}$, а также одновременно упрощая и усиливая выражение, избавляясь от вычитания ρ внутри скобок в выражении (1), получим неравенство для оценки общего объема памяти многопортового телекоммуникационного устройства «с запасом»:

$$M \geq \frac{1}{1024} \sum_i\ MTU_i \sum_{j=1,2} \left[\frac{a\rho}{C^{2H-1}} \cdot \left(\frac{-\ln(\frac{C^{2H-1}}{2\rho a} \left(\frac{1-\rho}{H}\right)^{2H} \left(\frac{x}{1-H}\right)^{2(1-H)}\ )}{1-\rho} \right)^{2H} \right]^{\frac{1}{2(1-H)}}_{i,j} \tag{5}$$

В сетях MetroEthernet в основном используются устройства с одинаковым типом портов, поэтому на практике параметр *MTU* одинаков для всех интерфейсов. Поэтому, упрощая выражение с учетом работы интерфейсов в режиме полного дуплекса с одинаковой природой трафика в обоих направлениях, получаем:

$$M \geq \frac{MTU}{512} \sum_i \left[\frac{a\rho}{C^{2H-1}} \cdot \left(\frac{-\ln(\frac{C^{2H-1}}{2\rho a} \left(\frac{1-\rho}{H}\right)^{2H} \left(\frac{x}{1-H}\right)^{2(1-H)}\ )}{1-\rho} \right)^{2H} \right]^{\frac{1}{2(1-H)}}$$

Анализируя полученные результаты, видно, что общий пул памяти ввода-вывода не есть простая сумма буферных памятей интерфейсов, увеличение максимальной скорости интерфейсов позволяет снизить требования к величине общего пула памяти ввода-вывода, тогда как загрузка интерфейсов и показатель изменчивости трафика напротив диктуют её увеличение.

В случае возможного отсутствия самоподобия при *H=0,5* , формула преобразуется в следующий упрощенный вид и явно не зависит от *C* :

$$M \geq \frac{MTU}{512} \sum_i \left[a\rho \cdot \left(\frac{-\ln(\frac{1}{\rho a} \left(\frac{1-\rho}{H}\right) \left(\frac{x}{1-H}\right)\ )}{1-\rho} \right) \right] \tag{6}$$

Одна из задач дальнейшей работы состоит в проверке данного соотношения методом имитационного моделирования и постановкой серии экспериментов на реальном оборудовании.

Выводы

В данной работе получены формулы для оценки общего объема памяти ввода-вывода многопортового телекоммуникационного устройства с учетом максимальных скоростей и степени загрузки интерфейсов, потери пакетов, а также самоподобия и вариации трафика как функции пиковой скорости и объема всплеска трафика и показана нелинейная зависимость от объема буферной памяти интерфейсов.

Результаты работы могут быть использованы для предварительной оценки параметров телекоммуникационной телекоммуникационные инфраструктуры различного назначения для качественного предоставления услуг через неё.

Литература

1. Л.Клейнрок, Коммуникационные сети, М., Наука, 1970, 255 с.
2. Л.Клейнрок, Вычислительные системы с очередями, М., Мир,1979, 598 с.
3. Л.Клейнрок, Теория массового обслуживания, М., Машиностроение, 1979, 432 с.
4. О.И.Шелухин, А.М.Тенякшев, А.В.Осин. Фрактальные процессы в телекоммуникациях. – М., Радиотехника, 2003. – 479 с.
5. Технологии коммутации и маршрутизации в компьютерных сетях: учеб.пособие / [Е.В.Смирнова, А.В.Пролетарский и др.]; под общ.ред.А.В.Пролетарского. – М.:Изд-во МГТУ им.Н.Э.Баумана, 2013. – 389 с.
6. В.Столлингс, Современные компьютерные сети – М., Питер, 2003. - 782 с.
7. Ромасевич П.В., Адаптивная телекоммуникационная система как средство реализации качества обслуживания в сетях с интенсивным трафиком // Инфокоммуникационные технологии, 2006, - №3. С. 21-26.
8. I.Norros, A Storage Model with Self-Similar Input, Queuing Systems, Vol.16, 1994
9. Norros I. On the Use of Fractanial Brownian Motion in the Theory of Connectionless Networks// IEEE Journal on Selected Areas in Communications. – Aug. 1995. – Vol.13, №6. – Pp.953-962.
10. П.В.Ромасевич. Оценка памяти ввода/вывода маршрутизаторов Cisco с интерфейсами множественного доступа в телекоммуникационных сетях с интенсивным трафиком//Инфокоммуникационные технологии. – 2004. – Т.1, №4 – С.36-40.
11. П.В.Ромасевич, «Адаптивные телекоммуникационные системы для сетей образования и науки. Исследование и создание», Монография, LAP Lambert AcademicPublishing, 2012, 200 с., ISBN 978-3-8473-0443-2
12. Ю.И.Лосев, К.М.Руккас. Анализ моделей вероятности потери пакетов в буфере маршрутизатора с учетом фрактальности трафика // Вісник Харківського національного університету, серія «Математичне моделювання. Інформаційні технології. Автоматизовані системи управління» – 2008, №833, с.163-169.

Емельченков Е.П.[1], Макаров А.И.[2], Мунерман В.И.[3]

[1] Смоленский государственный университет, г.Смоленск, к.ф-м.н., заведующий кафедры информатики, ypy1101@gmail.com

[2] Смоленский государственный университет, г.Смоленск, студент физико-математического факультета, al.makarov8@gmail.com

[3] Смоленский государственный университет, г.Смоленск, к.т.н., доцент кафедры информатики vimoon@gmail.com

ОБЪЕКТНО-ОРИЕНТИРОВАННЫЙ ПОДХОД К ПРЕДСТАВЛЕНИЮ БАЗ ДАННЫХ В НЕ ПЕРВОЙ НОРМАЛЬНОЙ ФОРМЕ

КЛЮЧЕВЫЕ СЛОВА

Базы данных не первой нормальной формы, объектно-реляционные базы данных, абстрактный тип данных.

АННОТАЦИЯ

В статье рассматривается один из подходов к представлению баз данных в не первой нормальной форме. Приводится формализация понятия объектная база данных на основе строгого определения абстрактного типа данных. Приводятся примеры, показывающие возможность реализации запросов, сформулированных в терминах реляционной алгебры, которые обрабатывают отношениям в не первой нормальной форме.

Экспериментальная часть работы выполнена при помощи средств Windows Azure в рамках гранта, предоставленного Смоленскому Государственному Университету корпорацией Microsoft.

В теории реляционных баз данных не нормализованным отношением или отношением в не первой нормальной форме считается отношение, которое содержит хотя бы один не атомарный атрибут. В литературе по базам данных предполагается, «что среди значений домена не могут содержаться значения с видимой структурой, в том числе множества значений, отношения». Вместе с тем понятие «атомарности значения трактуется в том смысле, что с этим значением можно работать только с помощью операций соответствующего типа данных» [1].

Далее рассматривается подход к формализации описаний данных и процедур их обработки в рамках объектно-реляционной модели, который позволит, не выходя за рамки этой модели, конструировать запросы к базам данных независимо от уровня нормализации входящих в нее отношений.

К. Дейт предостерегает [2] «Прежде чем перейти к подробному освещению данной темы [объектные базы данных], необходимо предупредить читателя, что впредь не следует ожидать скрупулезной точности изложения, присущей реляционной теории. Действительно, многие объектные концепции (или их опубликованные определения) выражены не очень точно, относительно формулировки их определений единства мнений не достигнуто, и разногласия возникают даже по фундаментальным вопросам».

Под ОРСУБД следует понимать системы, которые следуют духу *Манифеста систем баз данных третьего поколения* и букве стандартов SQL:1999 и SQL:2003 [3].

Для того, чтобы использовать объектный подход нужно отказаться от неформального взгляда на объектные базы данных, сделать попытку дать формальное определение объекта и объектно-реляционной базы данных. Такое определение не обязательно будет общим, учитывающим все возможные случаи, но оно должно быть достаточно полным для того, чтобы решать практически значимое множество задач. Большинство современных подходов к объектным базам данных основываются на неформальном определении абстрактного типа данных (АТД) впервые сформулированным в [4]. «Абстрактный тип данных определяет класс объектов, который полностью характеризуется операциями возможными над этими объектами. Это означает, что абстрактный тип данных может быть определен описанием операций для этого типа». Однако, попытки формализовать понятие АТД были сделаны задолго до появления этой статьи в работах академика Л.В.Канторовича [5]. Развитие этих взглядов привело к появлению

строгого определения АТД.

Абстрактный тип данных – это многоосновная универсальная алгебраическая система [6].

Под многоосновной алгебраической системой понимается система U_M=<***M***; Ω, Π ;>, состоящая из семейства основных множеств ***M***={ A_α } ($\alpha=1,2,\ldots$) и сигнатуры Ω операций, определенных на семействе ***M*** так, что каждая *n*-арная операция из Ω является отображением декартова произведения *n* множеств из семейства ***M*** в множество из того же семейства $A_{\alpha_1} \times \ldots \times A_{\alpha_n} \to A_{\alpha_r}$, Π – сигнатура *n*-местных предикатов $\pi : A_{\alpha_1} \times \ldots \times A_{\alpha_n} \to \{0,1\}$.

Тогда любая модель данных в том числе объектно-реляционные может рассматриваться как базовый абстрактный тип данных, потому что представляет собой универсальную многоосновную алгебраическую систему, в современно терминологии этому соответствует понятие суперкласс.

На основе модели данных строятся различные базы данных. Каждая такая база данных содержит логическое описание данных (схемы отношений) и новые операции – алгебраические выражения запросов. Исходя из неформального определения АТД каждая такая база данных может рассматриваться как АТД-наследник базового АТД, в современной терминологии как класс-наследник. После того, как такая база данных заполняется конкретными данными получается реализация или экземпляр АТД (класса), который в современной терминологии принято называть объектом. Таким образом мы получаем иерархию классов, которую можно представить в виде древовидного графа, корнем которого будет модель данных, ветви будут проходить через АТД-наследники (конкретные базы данных), а листьями будут объекты – экземпляры баз данных наполненные конкретным содержанием. Пример такого дерева приведен на рисунке 1.

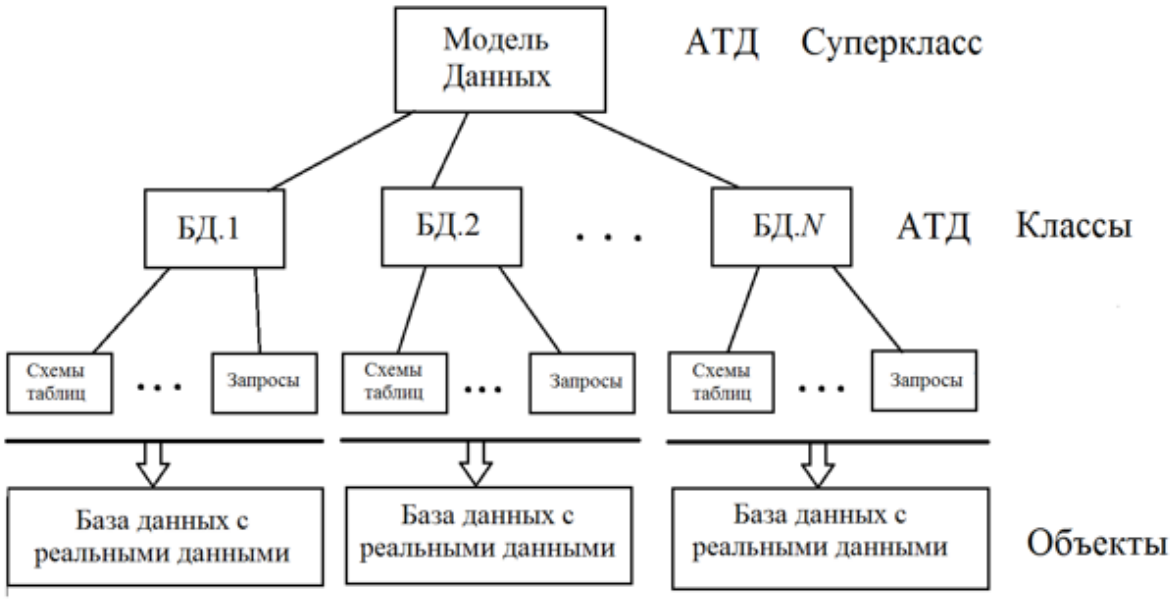


Рис.1. Дерево наследования в объектно-реляционной модели данных

Далее показано, что множество отношений можно рассматривать, как атомарный домен, и показано, что это укладывается в определение ОРСУБД. Пусть **R –** множество всех возможных отношений. В реляционной алгебре над отношениями из множества **R** определены унарные и бинарные алгебраические операции. Отношение в не первой нормальной форме определяется как отношение со схемой $R^0(A_1,\ldots,A_k,R_{k+1},\ldots,R_n)$, где R^0 – отношение в не первой нормальной форме, а $R_{k+1},\ldots,R_n$ – атрибуты, домены которых содержат отношения с одинаковыми схемами, находящиеся в первой или более высоких нормальных формах. В статье рассматривается этот частный случай, который, в принципе, может быть обобщен на случай, когда домены атрибутов $R_{k+1},\ldots,R_n$ могут содержать отношения в не первой нормальной форме.

Современные языки манипулирования данными (ЯМД), такие как Transact-SQL можно рассматривать как языки, поддерживающие технологию объектно-ориентированного

проектирования и программирования. Действительно, поскольку конкретная база данных есть АТД, то её можно рассматривать, как совокупность переменных (свойств), в роли которых выступают отношения (таблицы), и операции над ними (методы), представленные в виде запросов и хранимых процедур, а также функций, триггеров и прочих процедурных элементов базы данных. Далее, следуя терминологии ЯМД, вместо термина «отношение» будет использоваться термин «таблица».

При таком подходе таблицы конкретной базы данных можно разделить на две группы: первая группа содержит таблицы в не первой нормальной форме, а вторая – таблицы, которые могут входить в множества значений атрибутов $R_{k+1},\ldots,R_n$ таблиц первой группы. В конкретных случаях вторая группа таблиц разделена на подгруппы, каждая из которых и есть домен атрибута *R*, входящего в состав схемы одной из таблиц R^0. Над таблицами, входящими в домены атрибутов $R_{k+1},\ldots,R_n$ определены операции реляционной алгебры. Кроме того, над ними могут быть определены предикаты. Например, если таблица R_{ij} принадлежит домену атрибута R_i (*j* – фиксирует таблицу в домене), то можно определить предикат «число строк в таблице R_{ij} больше *L*». Такой предикат позволит выбирать таблицы в зависимости от количества строк.

Над таблицами в не первой нормальной форме могут выполняться любые операции реляционной алгебры, а именно: унарные операции SELECT и PROJECT, а также бинарные теоретико-множественные операции и операция JOIN.

Далее рассматриваются примеры реализация операций над отношениями в не первой нормальной форме средствами ЯМД Transact-SQL на основе объектно-ориентированного подхода. Возможность применения этого подхода к рассматриваемому типу баз данных обусловливается наличием типа данных «таблица». Переменные этого типа данных задаются следующим образом: DECLARE @<Имя переменной-таблицы> Table (<Список полей с указание типов>).

Пример 1. Обработка таблиц в не первой нормальной форме оператором SELECT. Пусть дана таблица *R*0(*A* <простой тип данных>, *R* nchar(255)). В базе данных хранится множество таблиц *R*1 = {*R*11, *R*12, …, *R*1*N*}. Все таблицы *R*1*i* имеют одну и ту же схему и находятся в первой или более высокой нормальной форме. Значениями атрибута *R* таблицы *R*0 будут имена таблиц, которые играют роль ссылок на таблицы из множества *R*1. В этом случае можно считать, что таблица *R*0 находится в не первой нормальной форме. Можно также считать, что множество таблиц *R*1 будет доменом атрибута *R* таблицы *R*0, поскольку этот атрибут содержит ссылки на таблицы, которые будут обрабатываться в запросе. В рассматриваемом примере все таблицы хранятся в одной базе данных, но это упрощение не является существенным, так как вместо простого имени таблицы можно использовать URL. Далее рассматриваются запросы к таблице *R*0 реализованные в виде специальных операций с помощью хранимых процедур. Запрос выполняет выборку таблиц, число строк в которых превышает заданную величину *L*. В запросе участвует таблица *R*0, описание схемы которой дано в начале примера, таблицы из множества *R*1 имеют одинаковую схему, состоящую из двух атрибутов простых типов. Если бы такой запрос можно было написать средствами обычного языка SQL то он бы имел следующий вид:

SELECT <список полей> FROM *R*0 WHERE <число строк в таблице *R*1*i*>*L* >.

Поскольку в команде WHERE возможны только алгебраические выражения в которых участвуют переменные простых типов данных реализация такого запроса возможна по средствам специального метода, организованного как хранимая процедура (листинг 1). Процедура принимает целочисленный параметр @*L*, нижняя граница числа строк для таблиц *R*1*i*. Результат этого запроса помещается в таблицу *R*0*Res* схема которой совпадает со схемой таблицы *R*0.

Листинг 1. Хранимая процедура для реализации запроса SELECT с предикатом, определенным на множестве таблиц

```
1. CREATE PROCEDURE [Select_NF_Where]
2.  @L int
3. DECLARE @R1i TABLE (B1 int, B2 int)
4. DECLARE @A int, @R nchar(255), @row_countR0 int
5. DECLARE @sql_cmd nchar(255)
6. DECLARE @i int
7. DECLARE cursorR0 CURSOR FOR SELECT * FROM R0
8. DECLARE cursorR1i CURSOR FOR SELECT * FROM @R1i
```

```
9.  DELETE FROM R0Res
10. OPEN cursorR0
11. SET @row_countR0=@@CURSOR_ROWS
12. SET @i=0
13. WHILE @i<@row_countR0
14. BEGIN
15.    FETCH NEXT FROM cursorR0 INTO @A, @R
16.    SET @sql_cmd='INSERT INTO @R1i SELECT * FROM'+@R
17.    EXEC (@sql_cmd)
18.    OPEN cursorR1i
19.    IF @@CURSOR_ROWS>@L
20.       INSERT INTO R0Res(A, R) VALUES(@A, @R)
21.    CLOSE cursorR1i
22. END
23. CLOSE cursorR0
```

В строке 3 определяется переменная-таблица, в которую будут размещаться таблицы *R*1*i*. В строке 4 определяются переменные для хранения полей таблицы *R*0 и переменные для организации цикла просмотра всей этой таблицы. В строке 5 определяется переменная для построения SQL-команды

INSERT INTO @*R*1*i* SELECT * FROM @*R*1*i*, (*)

которая перемещает очередную таблицу *R*1*i* в переменную-таблицу. Переменная @*i* используется для организации цикла считывания строк таблицы *R*0. В строках 7 и 8 определяются курсоры для чтения таблиц *R*0 и *R*1*i*, после чего в строках 10-13 формируется и запускается цикл построчного считывания таблицы *R*0. В теле цикла считывается очередная строка таблицы *R*0 и формируется SQL-команда (*), ссылка на таблицу *R*1*i* выбирается из прочитанной строки таблицы *R*0 (строки 15-16). В строке 17 выполняется сформированная SQL-команда, после чего переменная-таблица @*R*1*i* содержит все строки таблицы *R*1*i*. В строках 18-20 определяется число строк в таблице *R*1*i* и проверяется условие, если условие выполнено, строка таблицы *R*0, добавляется к таблице *R*0*Res*, и, поскольку, больше нет потребности в таблице *R*1*i* её курсор закрывается.

Средства языка Transact-SQL позволяют реализовать построенный таким образом метод, реализующий оператор SELECT <список полей> FROM *R*0 WHERE <число строк в таблице *R*1*i*>*L* >, как функцию возвращающую значение типа таблица.

Пример 2. Обработка таблиц в не первой нормальной форме оператором SELECT с групповой операцией UNION. Представление исходной таблицы в не первой нормальной форме и связанных с ней таблиц такое же, как и в примере 1. Текст хранимой процедуры, реализующей этот запрос приведен в листинге 2.

Листинг 2. Хранимая процедура для реализации запроса SELECT с групповой операцией UNION, определенной на множестве таблиц

```
1.  CREATE PROCEDURE [Select_NF_Groop]
2.  DECLARE @R1i TABLE (B1 int, B2 int), @unionR1i TABLE (B1 int, B2 int)
3.  DECLARE @A int, @R nchar(255), @row_countR0 int, @currentA int
4.  DECLARE @sql_cmd nchar(255)
5.  DECLARE @i, @n int
6.  DECLARE cursorR0 CURSOR FOR SELECT * FROM R0
7.  DELETE FROM R0Res
8.  OPEN cursorR0
9.  SET @row_countR0=@@CURSOR_ROWS
10. FETCH NEXT FROM cursorR0 INTO @A, @R
11. SET @currentA =@A
12. DELETE FROM @R1i
13. DELETE FROM @unionR1i
14. SET @sql_cmd='INSERT INTO @R1i SELECT * FROM'+@R
15. EXEC (@sql_cmd)
16. SET @i=1
17. WHILE @i<@row_countR0
18. BEGIN
19.    FETCH NEXT FROM cursorR0 INTO @A, @R
```

```
20.   IF @A = @currentA
21.   BEGIN
22.     SET @sql_cmd= 'INSERT INTO @unionR1i  SELECT * FROM @R1i
                       UNION SELECT * FROM' + @R
        EXEC (@sql_cmd)
23.     DELETE FROM  @R1i
24.     INSERT INTO @R1i  SELECT * FROM @unionR1i
25.   END
26.   ELSE
27.   BEGIN
28.     SET @n = @n + 1
29.              SET  @sql_cmd=  'CREATE  TABLE  R'  +  CAST  (@n  AS  nchar(4))+
                   ' (B1 int,  B2 int)'
30.     EXEC (@sql_cmd)
31.              SET  @sql_cmd=  'INSERT  INTO  R'  +  CAST  (@n  AS  nchar(4))+
                   'SELECT * FROM @unionR1i'
32.     EXEC (@sql_cmd)
33.                       INSERT  INTO  R0Res  (A,  R)  VALUES  (@A,  'R'  +
                            CAST (@n AS nchar(4)))
34.     DELETE FROM @R1i
35.     DELETE FROM @unionR1i
36.     SET @sql_cmd='INSERT INTO @R1i SELECT * FROM'+@R
37.     EXEC (@sql_cmd)
38.   END
39.   SET @i = @i + 1
40. END
41. CLOSE cursorR0
```

В строке 2 определена таблица @*unionR*1*i*, в которой будет накапливаться объединение таблиц *R*1*i*, при фиксированном значении атрибута *A*. В строках 8-15 формируется таблица @*unionR*1*i* для первой строки таблицы *R*0. Затем выполняется цикл чтения таблицы *R*0. В теле цикла выполняются следующие действия. Если значение атрибута *A* очередной прочитанной строки совпадает с текущим значением этого атрибута в переменной @*currentA*, то в строках 22-24 формируется и выполняется команда объединения очередной таблицы *R*1*i* с таблицей @*unionR*1*i*. В противном случае в строках 28-37 создается новая таблица *R*1*i*, где *i*=*n*+1, которая заполняется данными из таблицы @*unionR*1*i*. Затем формируется очередная строка таблицы результата *R*0*Res*.

Таким образом, можно сделать вывод о том, что применение строгого формального подхода к определению АТД позволяет рассматривать классическую реляционную модель как объектную модель данных, а использование средств, реализованных в современных СУБД, и, в частности, в их языках манипулирования данными, позволяет реализовать обработку баз данных, содержащих отношения в не первой нормальной форме.

Литература

1. Кузнецов С.Д. Основы баз данных: учебное пособие – 2-е изд., испр. – М.: Интернет-Университет Информационных Технологий; БИНОМ. Лаборатория знаний, 2007. - 125с.
2. Дейт, К. Дж. Введение в системы баз данных, 8-е издание.: Пер. с англ. — М.: Издательский дом "Вильяме", 2005. - 1022с.
3. Кузнецов С.Д. Объектно-реляционные базы данных: прошедший этап или недооцененные возможности? – Институт системного программирования РАН, 2007. - 117 стр.
4. Liskov B., Zilles S. Programming with abstract data types. – ACM SIGPLAN Notices Volume 9 Issue 4, April 1974. – P. 50 – 59.
5. Канторович Л. В., Петрова А. Т. О математической символике, удобной при вычислениях на машинах. Труды 3-го Всесоюзного математического съезда. Секция вычислительной математики. 1956, т. II, с. 151.
6. Глушков В.М., Цейтлин Г.Е., Ющенко Е.Л. Алгебра. Языки программирование. – Киев: Наукова думка, 1989. – 376с. – ISBN 5-12-000499-7.

Копытов В.В.,[1] Шульгин А.О.,[2] Федоров С.А.[3]

[1] Институт информационных технологий и телекоммуникаций Северо-Кавказского федерального университета, г. Ставрополь, ООО «Инфоком-С», г. Ставрополь, д.т.н., профессор, профессор кафедры организации и технологии защиты информации, v.kopytov@infocom-s.ru

[2] ООО «Инфоком-С», г. Ставрополь, к.т.н., доцент, директор по развитию, a.shulgin@infocom-s.ru

[3] Институт информационных технологий и телекоммуникаций Северо-Кавказского федерального университета, г. Ставрополь, ООО «Инфоком-С», г. Ставрополь, аспирант, sergeiaf@mail.ru

РАЗРАБОТКА АРХИТЕКТУРЫ ИНТЕГРАЦИОННОЙ СРЕДЫ КРОССПЛАТФОРМЕННЫХ МОБИЛЬНЫХ ПРИЛОЖЕНИЙ С КОРПОРАТИВНОЙ ИНФОРМАЦИОННОЙ СИСТЕМОЙ

КЛЮЧЕВЫЕ СЛОВА

Кроссплатформенность, мобильное приложение, информационная система, облачная инфраструктура.

АННОТАЦИЯ

Существующие на данный момент способы и технологии интеграции программных продуктов подходят для высокоуровневой организации взаимодействия различных гетерогенных информационных систем. Однако, при их использовании необходимо учитывать некоторые особенности особого класса приложений, такого как мобильные приложения. К таким особенностям следует отнести слабую аппаратную составляющую оборудования, на котором используются мобильные приложения, и низкую пропускную способность каналов связи, вследствие чего на клиентском приложении зачастую невозможно выполнить ресурсоемкие вычисления или осуществить хранение, прием и передачу результатов промежуточной обработки данных. Также к числу таких особенностей следует отнести предметную область использования мобильного приложения с вариацией способа доставки данных и массовость конечного продукта.

Для эффективной интеграции мобильных приложений с корпоративными информационными системами и для достижения высокого уровня кроссплатформенности должна быть предложена новая архитектура интеграционных механизмов, не имеющая технологических ограничений интеграционных шин, построенных на основе классических технологий.

Целью работы является описание архитектуры и принципиальной структуры технологии построения мобильных приложений, интегрированных с корпоративными информационными системами.

Актуальность исследования обосновывается наличием существенных технологических ограничений при существующих подходах к созданию мобильных приложений, интегрированных с корпоративными информационными системами.

В ходе работ предложена усовершенствованная архитектура интеграционной среды на основе использования облачных сервисов взаимодействия и набора моделей для реализации автоматизированных сценариев подготовки программного кода кроссплатформенных мобильных приложений, встраиваемых в специализированные приложения-контейнеры. Полученные теоретические и прикладные результаты могут быть использованы в программной инженерии при проектировании и разработке интеграционных шин и создании мобильных приложений.

Исследование выполнено в рамках реализации научного проекта по теме «Разработка кроссплатформенной технологии построения мобильных приложений с заданными контурами интеграции для повышения функциональной и ресурсной эффективности корпоративных

информационных систем» в рамках ФЦПИР 2014-2020 (уникальный идентификатор прикладных научных исследований RFMEFI57614X0066 при финансовой поддержке Министерства образования и науки Российской Федерации.

Рынок мобильных приложений демонстрирует уникальный рост за последние 3-5 лет, формируя постоянный растущий спрос среди конечных потребителей. Крупнейшие компании ИТ-отрасли активно занимаются разработкой и продвижением мобильных приложений и мобильных сервисов, что требует значительного внимания к использованию эффективных технологий разработки. Среди всего множества мобильных приложений можно выделить интенсивно развивающийся сегмент приложений, используемых сотрудниками предприятий и организаций для решения своих производственных задач (корпоративные мобильные приложения). Отличительной особенностью данных мобильных приложений является их включение в ИТ-инфраструктуру предприятия и тесная интеграция с корпоративными информационными системами.

В настоящее время широкое распространение получили мобильные устройства, такие как коммуникаторы, планшетные компьютеры, программируемые считывали штрих-кодов, считыватели данных о RFID-метках и прочие устройства. Динамика разработки мобильных приложений обуславливается потребностью использования общеизвестных программ не только на стационарных компьютерах, но и на мобильных устройствах. Особый интерес вызывает возможность мобильных устройств и мобильных приложений по организации сетевого взаимодействия посредством корпоративной сети или сети интернет. Для мобильных устройств известны примеры реализации офисных пакетов, мультимедийных и коммуникативных приложений, перспективными направлениями в сфере разработки корпоративных приложений является разработка, например, складских и бухгалтерских программ.

В исследовании, проведенном в 2012 году группой Marketvisio и посвященном анализу трендов использования мобильных приложений в корпоративной среде, отмечается достаточно низкий уровень проникновения мобильных приложений среди российских компаний. Были исследованы более 400 организаций, использующих мобильные приложения. Наибольшую долю в общем объеме выборки составили респонденты из финансового сектора и торговли. Среди опрошенных организаций использовали приложения для доступа к корпоративной почте или календарю 37% предприятий, использовали приложения для доступа к рабочему столу 18% предприятий, использовали приложения для мгновенного обмена сообщениями между сотрудниками 11% предприятий. Использование мобильных приложений для работы с корпоративными системами отметили 11% предприятий, а 3 % предприятий планировали осуществить внедрение и начать использование таких приложений в течении 1 года.

Одним из основных факторов, сдерживающим использование мобильных приложений называется потеря контроля за ит-системами предприятия и данными (ослабление защиты данных).

В исследовании [2] отмечается, что финансовые ресурсы при внедрении мобильных приложений распределены в неравных долях между внешними и внутренними ресурсами, по этой причине интеграторы и производители мобильных приложений могут более активно участвовать во внедрении мобильных приложений.

Представители российской ИТ-индустрии также уделяют значительное внимание расширению возможностей платформ для построения корпоративных информационных систем, реализуя средства проектирования и разработки мобильных приложений и мобильных сервисов. В направлении создания интегрированных с корпоративными информационными системами мобильных приложений работают лидеры российского рынка делового ПО – Epam Systems, 1С, АйТи, Форс и другие. Поэтому учитывая перспективы роста сегмента корпоративных мобильных приложений в России и мире необходимо создавать и развивать отечественные технологии для разработки и интеграции мобильных приложений в составе корпоративных информационных системам, построенных на базе отечественных программных продуктов.

Одним из перспективных способов организации взаимодействия и обмена данными между приложениями (в том числе и между мобильными приложениями) является создание общего механизма интеграции, не зависящего от конкретной платформы или языка программирования (интеграционная шина). Разработка такого механизма для мобильных приложений корпоративного сегмента требует особых подходов, что обусловлено характеристиками мобильных устройств, условиями их функционирования и взаимодействия, используемыми операционными системами и средами разработки программного обеспечения:

- для мобильных устройств недоступны большие вычислительные ресурсы для отображения сложного функционала клиентской части;

- доступ к данным корпоративной базы данных может потребоваться одновременно нескольким пользователям, как со стационарных компьютеров, так и с удаленных мобильных устройств;

- нередко возникает необходимость продолжения работы удаленного пользователя даже при временном отсутствии доступа к корпоративной сети или сети интернет;

- мобильное приложение должно обеспечивать надежные механизмы однозначной идентификации пользователей и защиты передаваемых данных.

Существующие зарубежные технологии, обеспечивающие задачи создания мобильных приложений, интегрированных с корпоративными информационными системами, как правило, являются закрытыми и платформенно-зависимыми и не могут быть перенесены для отечественных разработок в области программной инженерии. Теоретические исследования зарубежных исследовательских центров и RnD-компаний направлены на потребности конкретных вендоров и не позволяют сформировать общую теоретико-методологическую базу для разработки соответствующих отечественных технологий. Однако, решение указанной задачи является одним из приоритетов российского ИТ-индустрии, а разрабатываемое направление создания мобильных приложения - важной задачей при формировании информационного общества.

Актуальность исследования связана с широким применением мобильных приложений в составе корпоративных информационных систем на транспорте, в сфере государственного управления, в торговле и иных отраслях и необходимостью создания технологических инструментов для интеграции создаваемых мобильных приложений с уже действующими информационными системами. Прикладная значимость разработок также связана с тем, что на данный момент стоит задача интеграции мобильных технологий в те сферы, где они ранее не находили применение: бизнес-структуры, правительство, военная сфера, промышленность, здравоохранение. Учитывая, что перечисленные сферы имеют разный уровень автоматизации, различные платформы и подходы к организации информационных систем, то необходимо выработать единые технические и технологические стандарты для внедрения в них новых технических решений, таких как мобильные устройства.

Объектом исследования является кроссплатформенные мобильные приложения, интегрированные с корпоративными информационными системами. Предметом исследования являются существующие методы и технологии построения кроссплатформенных мобильных приложений и методы их интеграции.

Использование зарубежного опыта построения кроссплатформенных мобильных приложений на основе программных платформ SAP, IBM, Microsoft, обеспечит создание технологического задела при построении «бесшовной» интеграции, обеспечить формирование оптимальных интеграционных решений для отдельных задач на основе интеграционных шин, разработать принципы описания архитектуры мобильного приложения на основе описанных в информационной системе бизнес-процессов.

Добавление в состав корпоративной информационной системы (КИС) мобильных приложений, существенно расширяющих пространство функционирования за счет смартфонов, фаблетов, планшетов и других мобильных устройств, приводит к расширению технологического состава и функций системы. Для терминологической унификации в рамках данной статьи под интегрированной корпоративной информационной системой (интегрированной КИС) понимается набор программно-технологических средств, обеспечивающих интегрированное функционирование комплекса программных средств корпоративной информационной системы и мобильных приложений для различных платформ или кроссплатформенных [3].

Интегрированная КИС уже может обеспечивать работу персонала за пределами офиса или здания за счет использования современных технологий обеспечения доступа к внутрикорпоративным системам и базам данных при использовании личных или рабочих мобильных устройств. При этом необходимо учесть разнообразие платформ рынка мобильных устройств. Одним из вариантов оптимизации времени разработки мобильных приложений является выбор одной мобильной платформы, для которой и будет разрабатываться мобильное приложение в рамках интегрированной КИС. Но это, в свою очередь, добавляет дополнительные ограничения и ставит под вопрос целесообразность данного частного решения. При разработке комплекса мобильных приложений, охватывающих большинство платформ, требуются большие временные затраты на этап разработки и тестирования каждого мобильного приложения в

отдельности [4].

Другим решением оптимизации времени разработки мобильных приложений под разные платформы является использование альтернативного подхода, позволяющего разрабатывать кроссплатформенные приложения для различных мобильных платформ.

Существующие на данный момент способы и технологии организации взаимодействия программных продуктов подходят для высокоуровневой организации взаимодействия различных гетерогенных информационных систем. Однако при их использовании необходимо учитывать некоторые особенности особого класса приложений, такого как мобильные приложения. К таким особенностям следует отнести, слабую аппаратную составляющую оборудования, на котором используются мобильные приложения и низкую пропускную способность каналов связи, вследствие чего на клиентском приложении невозможность выполнить ресурсоемкие вычисления или осуществить хранение, прием и передачу результатов промежуточной обработки данных. Также к числу таких особенностей следует отнести предметную область использования мобильного приложения с вариацией способа доставки данных и массовость конечного продукта.

Одной из возможностей решения быстрой разработки кроссплатформенных мобильных приложений для взаимодействия с КИС является выделение однотипных для различных КИС технологий, принципов и процессов в единую программно-технологическую платформу (фреймворк). Уменьшение временных затрат при разработке мобильных приложений с использованием кроссплатформенного фреймверка заключается в отделении всех процессов, не связанных с особенностями мобильной платформы, в отдельный поток и реализация этого потока только один раз. Данный подход позволяет разово разработать общую платформу, на основании которой и будут в дальнейшем строиться различные интегрированные КИС.

Рассмотрим концептуальную схему платформы быстрого построения кроссплатформенных мобильных приложений для построения интегрированных КИС, представленную на рисунке 1.

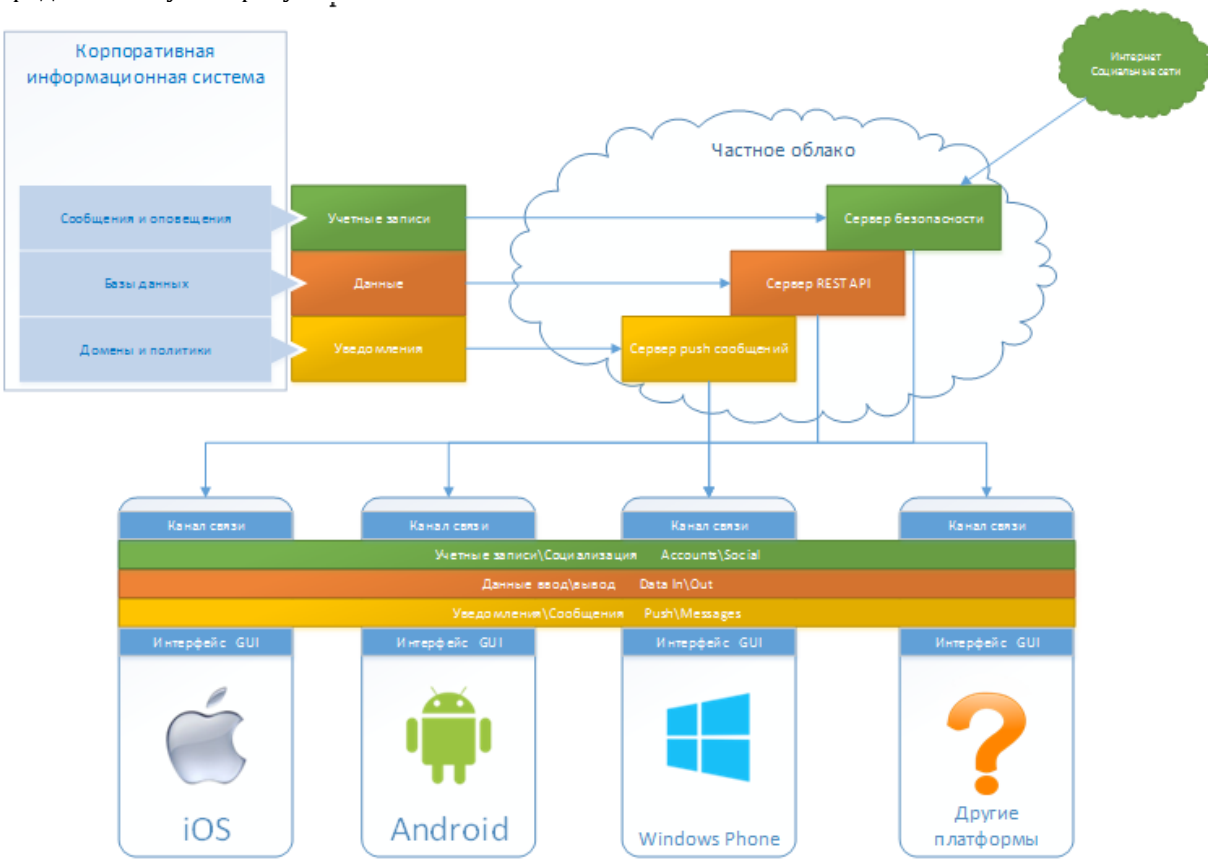


Рисунок 1 – Концептуальная архитектура интегрированной КИС

Архитектура интегрированной КИС включает в себя следующие составные блоки:

- корпоративная информационная система;
- подсистема расширений КИС;

- частное облако вычислительных ресурсов;
- мобильные приложения.

Подсистема расширений КИС представляет собой пакет программных средств преобразования данных и процессов, находящихся в составе КИС, в соответствующие идеологии и требованиям кроссплатформенной технологии данные и процессы. Подсистема расширений, в зависимости от типа и архитектуры КИС, может собой представлять, как монолитное программное обеспечение с использованием того же стека технологий, так и распределенный слабосвязанный комплекс служб и сервисов, обеспечивающий механизмы преобразования данных.

Частное облако – это инфраструктура вычислительных мощностей компании, предназначенная только для использования в рамках интегрированной КИС. Потребителями ресурсов частного облака является КИС, подсистема расширений КИС и сами мобильные приложения. Используется модель обслуживания «Инфраструктура как услуга» (IaaS), обеспечивающая самостоятельное управление ресурсами обработки, хранения и доступа к информации. Основными требованиями к частному облаку является её высокая надежность и доступность как со стороны корпоративной информационной системы, так и со стороны мобильных приложений. При этом частное облако является частью общей вычислительной сети компании, использующей технологии виртуальных частных сетей поверх сети Интернет (virtual private networks). Отдельные компоненты частного облака должны предоставлять сервисы для подключения различных ресурсов сети Интернет (социальные сети, базы и банки данных, почтовые сервисы и т.д.)[10-14].

Мобильные приложения являются исполняемыми программными модулями, разработанными с использованием кроссплатформенной технологии для различных мобильных платформ. К популярным платформам (экосистемам) можно отнести Android (Google), iOS (Apple), WindowsPhone (Microsoft), менее популярные платформы это BlackberryOS (RIM), Bada (Samsung), SymbianOS (Nokia). Между различными платформами существенно различается реализация транспортного слоя доступа к данным частного облака, а также пользовательский интерфейс и способ взаимодействия с человеком. Все различия продиктованы теми или иными особенностями операционных систем, используемыми средами функционирования, способами доступа к ресурсам и требованиями разработчиков платформ.

Среди всего объема данных, оперируемых в интегрированной КИС, можно выделить три основных потока: учетные записи, данные и уведомления. Данные потоки имеет смысл разделять между собой и описывать отдельно по причине больших различий между собой. Прежде всего, различия кроются в модели доступа к ним со стороны мобильных приложений и способах обработки данных. Менее существенные различия заключаются в направлении передачи, потребности в хранении и т.д. Кроме этого, блоки данных по-разному представляются пользователю для визуализации.

Учетные записи являются компонентом подсистемы безопасности КИС, обеспечивающим аутентификацию и идентификацию пользователей системы. В интегрированных КИС должны использоваться те же механизмы проверки пользователей и разграничения доступа, что и в настольных приложениях. Различается только модель аутентификации за счет наличия его в единственном экземпляре в КИС и менее доверенная среда исполнения – операционная система мобильного приложения. В связи с этим используются специальные средства обработки и хранения учетных данных, такие как: ключи сессии, временные токены авторизации и сертификаты подписи [5,6].

К потоку **данных** в интегрированных КИС относится процесс обмена основной обрабатываемой информацией, такой как: данные учетных систем, статистическая, качественная и количественная информация, различная оперативная информация, финансовая и производственная документация, информация о ресурсах и т.д. Основной моделью взаимодействия КИС и мобильных приложений является модель «запрос-ответ». При этом чаще всего пассивной стороной (сервером) в такого рода взаимодействии выступает КИС. Немаловажная возможность мобильных приложений является функция ввода новой информации в банки данных КИС, при этом модель взаимодействия остается такой же за исключением большего объема данных в запросе.

Уведомления и сообщения, хоть и частично можно отнести к потоку данных, тем не менее являются отдельным потоком по причине совершенно другой модели функционирования и решаемой основной задачи. Поток уведомления предназначен для передачи информации со стороны КИС в мобильное приложение, когда КИС является инициатором сообщений. В этом

случае мобильное приложение является пассивным компонентом в понимании клиент-серверной архитектуры. Для такого рода взаимодействия используются так называемые push-технологии в соответствии со следующими шаблонами (паттернами) PUSH-PULL и PUB-SUB. PUSH-PULL паттерн основан на процедурах помещения (push)и получения (pull) сообщения из очереди, и наличии постоянно действующих компонентах (worker)для чтения очередей. PUB-SUB паттерн подразумевает наличие процедуры подписки (subscribe) конкретного приложения на получение сообщений из очереди, а при необходимости отправки публикации (publish) в эту очередь отправителем.

Данный поток используется для доставки сообщений пользователям в рамках оперативной деятельности, срочного информирования и оповещений, осуществления коммуникаций, передачи извещений и событий. Характеристики и особенности канала доставки сообщений могут существенно варьироваться в зависимости от производственной необходимости, например, по скорости: мгновенно, в рамках времени жизни, при появлении связи, без ограничения; по охвату: unicast, multicast, broadcast, geocast, anycast; по предоставлению: текст, изображение, аудио, видео, комплексное и т.д. В зависимости от типа мобильной платформы способ уведомления пользователей о поступлении сообщений в приложение и принцип обмена данными с частным облаком может существенно отличаться.

Основным достоинством описанной технологии является дополнение функционала КИС до работы с мобильными приложениями без изменения ее внутренней архитектуры. Благодаря этому расширение КИС возможно не только на этапе разработки КИС, но и при успешном функционировании на протяжении длительного времени, что позволяет осуществить процесс расширения до интегрированной КИС, без нарушения работоспособности существующей КИС.

Облачная инфраструктура является ядом технологической платформы и обеспечивает функционирование интегрированной КИС. Облачная инфраструктура связывает блок взаимодействия с КИС и блок мобильной платформы в единое целое. Данный блок содержит большую часть бизнес-логики взаимодействия КИС и мобильного приложения. Исходя из этого нагрузка на облачную инфраструктуру прямо пропорциональна от количества мобильных приложений, а также от количества используемых адаптеров сопряжения для взаимодействия с КИС. Данные факторы накладывают дополнительные условия в области частного облака, а именно горизонтальная масштабируемость [21].

Выводы. Предложенная технология построения интеграционной среды для кроссплатформенных мобильных приложений учитывает современные принципы коммуникации и технологической интеграции, обеспечивая улучшение потребительских свойств и снижение временных затрат на создание и эксплуатацию.

Ключевыми преимуществами предложенных решений являются:

- гетерогенность, обеспечиваемая совместимостью технологических решений облачных сервисов;

- кроссплатфоменность на стороне мобильных устройств, заключающаяся в минимизации функционального кода и использовании приложений-контейнеров;

- снижение стоимости разработки и поддержки набора приложений для популярных мобильных операционных систем;

- снижение временных затрат на разработку функциональной части мобильных приложений за счет использования унифицированного описания бизнес-логики в составе описанных моделей в облачной инфраструктуре.

В тоже время использование данного подхода оправдано только при необходимости разработки мобильных приложений сразу для нескольких мобильных платформ.

Основными зависимостями при использовании данного подхода являются:

- использование дополнительного звена в виде облачной инфраструктуры

- дополнительное изучение конструктивных особенностей и способов взаимодействия с программно-техническим комплексом (фреймворком).

Литература

1. Абдикеев Н.М., Данько Т.П., Идельменов С.В., Киселёв А.Д. Реинжиниринг бизнес-процессов. – М.: Изд. «Эксмо», 2005 – 592 с.
2. Mikhail Y. Evaluating the potential of developing cross-platform mobile applications // Master of Science Thesis №4 (2013), Gothenburg, Sweden, 2013 – pp. 30-61
3. Брауде Э. Технология разработки программного обеспечения. — СПб, ПИТЕР, 2009. – 655 с.
4. Фаулер М. Архитектура корпоративных программных приложений.: Пер. с англ. — М.: Издательский дом

«Вильямс», 2006. — 544 с.
5. Levy A.Y. Logic-Based Techniques in Data Integration. Logic-based Techniques in Data Integration. In: Logic Based Artificial Intelligence. Edited by J. Minker. – Kluwer Publishers, 2000 – 344 p.
6. Питер Пин-Шен Чен. Модель «сущность-связь» — шаг к единому представлению о данных. Пер. М.Р. Когаловского.
7. Пруцков А.В, Цыбулько Д.М. Проблемно-ориентированное программирование // Вестник РГРТУ. 2013. №45. С. 57-62.
8. Репин В.В., Елиферов В.Г. Процессный подход к управлению. Моделирование бизнес-процессов / Владимир Репин, Виталий Елиферов. – М. : Манн, Иванов и Фербер, 2013. – 544 с.
9. Таненбаум Э., Ван Стеен М. Распределенные системы. Принципы и парадигмы – СПб.: Питер, 2003. – 877 с.
10. Шафер Д.Ф. Управление программными проектами: достижение оптимального качества при минимуме затрат / Д.Ф. Шафер, Р.Т. Фатрелл, Л.И. Шафер. – М.: Издательский дом «Вильямс», 2003. – 1136 с.
11. Гольфанд И.Я., Хлебутин П.С. Оценка трудозатрат разработки программной компоненты // Труды ИСА РАН. – Т. 15 – 2005 – с. 125-136;
12. Иготти М.К. Подход к созданию автоматизированной информационной системы для приборостроительного предприятия / М.К. Иготти, В.Н. Юрьев. – С.80-95.
13. Бородакий Ю. В., Лободинский Ю. Г. Эволюция информационных систем (современное состояние и перспективы). — М.: Горячая линия - Телеком, 2011. — 368 с. — 1000 экз. — ISBN 978-5-9912-0199-5.
14. Наумова О.А., Ульянов М.В. Классификация методов построения алгоритмических систем // Вычислительные технологии. – 2011. – Т. 16. – № 1. – С. 105-118.
15. SoCC '10: Proceedings of the 1st ACM symposium on Cloud computing / Hellerstein, Joseph M. — N. Y.: ACM, 2010. — ISBN 978-1-4503-0036-0.
16. Yvette E. Gelogo, Haeng-Kon Kim Mobile integrated enterprise resource planning system architecture // International Journal of Control and Automation Vol. 7, No. 3 (2014), pp. 379-388.
17. Charlie Collins, Michael D. Galpin, Matthias Kappler Android in practice, Manning Publications Co., 2012, 647 p.
18. Gillam L. Cloud Computing: Principles, Systems and Applications / Nick Antonopoulos, Lee Gillam. — L.: Springer, 2010. — 379 p. — (Computer Communications and Networks). — ISBN 9781849962407.
19. Harrison-Broninski, K. Human Interactions: The Heart and Soul of Business Process Management, Meghan-Kiffer Press, 2005. – 304 p.
20. Jin Han, Qiang Yan, Debin Gao, Jianying Zhou, Robert Deng Comparing mobile privacy protection through cross-platform applications. [Электронный ресурс]. – Режим доступа. – URL: http://flyer.sis.smu.edu.sg/ndss13-han.pdf (дата обращения: 27.06.2015).
21. Gritsenko A.V., Demurchev N.G., Kopytov V.V. & Shulgin A.O. Decomposition Analysis and Machine Learning in a Workflow-Forecast Approach to the Task Scheduling Problem for High-Loaded Distributed Systems // Modern Applied Science Vol. 9, No. 5 (2015), pp. 38-49.

Кейно П.П.

ФГБОУ ВПО «Московский авиационный институт (национальный исследовательский университет)»,
г. Москва, преподаватель кафедры «Системное моделирование и инженерная графика»,
science@blockset.ru

ПРЕДПОСЫЛКИ ФОРМИРОВАНИЯ НОВОЙ МЕТОДОЛОГИИ РАЗРАБОТКИ ВЕБ-УЗЛОВ BLOCKSET И ДЕКЛАРАТИВНОГО ЯЗЫКА BML

КЛЮЧЕВЫЕ СЛОВА

Веб-разработка, веб-приложение, программирование, серверное программирование, серверы, интерпретатор, декларативное программирование, визуальный редактор, BML, BlockSet, CGI, FastCGI, XML.

АННОТАЦИЯ

Представленная работа посвящена поиску эффективного метода решения задач, связанных с веб-разработкой. Затрагиваются аспекты разработки как клиентской, так и серверной стороны. Предоставляется теоретическая реализация разработанной методологии в виде модели, а также практической — в виде программного модуля интерпретатора и визуального редактора.

Относительно молодая сфера веб-разработки за свою недолгую историю обросла огромным количеством различных технологий и инструментариев. Как и в любых других отраслях, связанных с информационными технологиями в целом и программированием в частности, в веб-разработке важнейшим фактором является эффективность. Существует множество характеристик, определяющих данный фактор [1]. В представленной работе речь идёт о разработке эффективной методологии, позволяющей упростить решение ряда тривиальных задач, не затрагивая, при этом, гибкость.

Теоретическая реализация методологии включает в себя задачи перераспределения уровней абстракции, а также выявление основных сущностей, с которыми происходит работа. Практическая реализация представляет собой два самостоятельных проекта, связанных между собой понятиями методологии. Первый проект является интерпретатором декларативного языка **BML** (*англ. BlockSet Modeling Language*) [2], являющегося основной составляющей. Цель второго проекта заключается в наглядном представлении специфичных сущностей языка BML и интеграцией с классическими клиентскими технологиями [3].

Эффективный инструментарий строится по принципу оптимального распределения уровней **абстракции**: подчёркивания основных высокоуровневых сущностей и вынесении рутинных задач в базовые обобщённые алгоритмы. С повышением уровня абстракции упрощается процесс разработки, однако снижается гибкость. Зачастую, инструментарий с высоким уровнем абстракции не может предоставить достаточный функционал для реализации специфических свойств разрабатываемого объекта. И наоборот, низкий уровень абстракции заставляет разработчика тратить время на освоение инструментария и постоянному возвращению к рутинным задачам вместо сосредоточения сил на поиске основного решения [4].

Представленная работа посвящена серверной стороне веб-разработки. Говоря о серверной стороне, можно привести в пример различные технологии, начиная от фундаментальных интерфейсов связи внешней программы и веб-сервера (**CGI**, **FastCGI**) [5], заканчивая интерпретируемыми языками программирования, наилучшим образом решающих задачи динамической веб-разработки. К таким языкам относятся: Perl, PHP, Ruby, Python и др. Каждый из них оброс большим количеством дополнительных библиотек (фреймворков), облегчающих выполнение рутинных задач. На каждом из языков разработаны специализированные системы управления контентом (**CMS**, *англ. Content Management System*). Все упомянутые технологии работают на разных уровнях абстракции, имеют широкую целевую группу потребителей, начиная от новичков и заканчивая профессионалами индустрии. Каждый инструментарий имеет свои особенности:

Языки программирования:

- низкий уровень абстракции, повышенная гибкость разработки, появление большого

количества рутинных задач;

- требуются профессиональные знания в области составления алгоритмов и написания программного кода;
- Фреймворки:
- средний уровень абстракции, оптимальное отношение эффективность/гибкость, рутина снижена до минимума;
- помимо профессиональных знаний, требуются дополнительные временные ресурсы на изучение специфики каждой из библиотек;
- Системы управления контентом:
- высокий уровень абстракции, знания в области программирования требуются либо на базовом уровне, либо не требуются вовсе;
- абсолютно не гибкая разработка, решаются только самые тривиальные задачи.

На клиентской же части технологии заточены на решение специфичных задач, которые не требуют реализации сложных алгоритмических структур, а вся немногочисленная логика, нерешаемая этими технологиями, решается одним единственным языком программирования клиентской стороны – JavaScript, интерпретируемым браузером.

Совсем по-другому обстоят дела на серверной стороне, где исполняется основная логика работы приложения, происходит обработка больших объёмов данных, их выборка из внешних хранилищ, вспомогательные вычисления и многое другое. Этим и обуславливается технологическое многообразие серверной стороны. Условно, все классические серверные технологии разработки можно разделить на четыре группы, расположив их в определённой иерархической последовательности (рис. 1). Похожая модель была предложена Эндрю Таненбаумом для наглядного представления зависимости между скоростью и объёмом по отношению к различным типам компьютерной памяти [6].

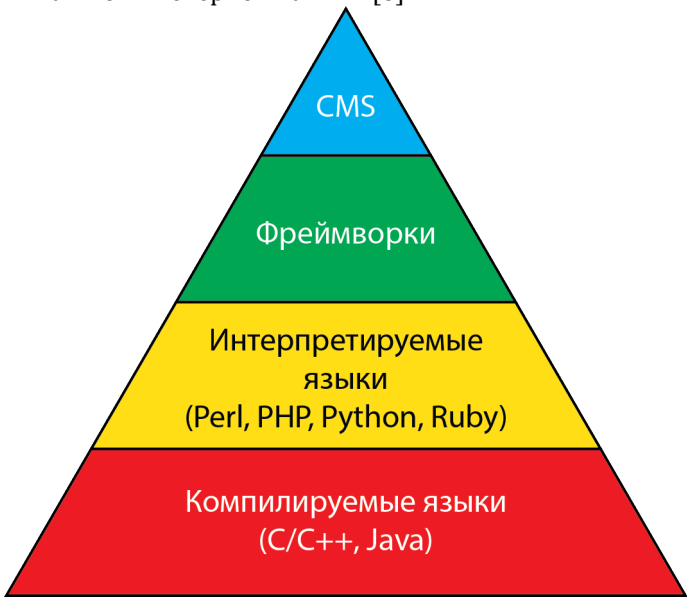


Рис.1. Иерархическая модель инструментариев веб-разработки

По мере продвижения с вершины иерархической пирамиды к её основанию, изменяются две основные характеристики. Во-первых, повышается **гибкость** разработки, что означает возможность реализации функционала любой сложности, в том числе и специфичного, разработка которого на более высоких уровнях затруднительна, либо невозможна в принципе.

Во-вторых, увеличивается **время**, затрачиваемое на разработку. Это следствие повышенной гибкости: на низких уровнях работа происходит с абстрактными сущностями, составляющих в комплексе сущности более высокого уровня с конкретно заданной спецификой. Таким образом, на нижних уровнях высокоуровневый функционал приходится составлять из элементарных частей.

Задачи, решаемые на серверной части, требующие детального анализа и поиска эффективного решения, сводятся к следующим группам:

- манипуляция данными (выборка, создание, обновление, удаление) [7];

- валидация данных [8];
- авторизация и разграничение прав доступа пользователей [9];
- сопоставление запрошенного адреса URI и динамической страницы.

Становится понятно, что разрабатываемая методология должна обладать высоким уровнем гибкости, низким порогом вхождения, высокой скоростью разработки. Разумеется, заявленные приоритеты могут противоречить друг другу, поэтому основной задачей ставится поиск необходимого оптимума между ними.

Целью работы является разработка эффективного решения в сфере серверных веб-технологий и определение его места в представленной выше иерархии. Разрабатываемая методология построена по принципу постепенной доработки и модификации базовых сущностей, придании им дополнительных свойств, атрибутов. Основными сущностями методологии являются: *модель*, *локация*, *набор*, *блок*. По названиям наиболее значимых сущностей, методология получила название **BlockSet** (блок и набор) [10]. В BlockSet модель описывает основную структуру проекта. Локация представляет собой динамическую страницу и описывает её поведение. Набор является непосредственной составляющей модели и локации, содержит коллекцию данных различного типа. Блок – часть набора, ячейка данных определённого типа. У всех сущностей (кроме модели) имеется большое число модификаторов – атрибутов, которые задают определённую логику работы.

Важным свойством методологии является её самодостаточность: оперируя лишь описанными выше сущностями, становится возможной реализация проекта начального уровня. Однако, грамотное оперирование атрибутами-модификаторами даёт возможность реализовать и узко-специализированный функционал. Так наглядно демонстрируется принцип «от простого к сложному».

Язык **BML** (*англ. BlockSet Modeling Language*), упомянутый выше, синтаксически идентичен языку **XML** и основан на парадигме **декларативного программирования**. Декларативное программирование является одной из самых простых практик ведения разработки. Оно описывает не методику создания объекта, а его состояние в зависимости от внешних условий. Декларативный стиль программирования по уровню абстракции находится выше классического императивного, следовательно, освоение языка не будет представлять значительных трудностей для начинающего разработчика. Например, язык HTML, по своей природе являющийся декларативным, описывает общие свойства документа, а не то, как этот документ должен отображаться на экране. В работе предлагается применить данный подход на серверной стороне, описывая основную логику работы сущности в виде её свойств, а не абстрактных алгоритмов.

Опишем более детально основные сущности методологии. С помощью **модели** (элемент *Model*) определяется структура данных, и устанавливаются отношения объектов проекта между собой. Структурой данных проекта BlockSet является набор, содержащий в себе блоки и другие, вложенные наборы. Все наборы, используемые в проекте, необходимо описать в модели. Единственным дочерним элементом модели является набор, в то время как блок является единственным дочерним элементом набора. Именно в модели для всех блоков устанавливается тип данных.

Локация или расположение (элемент *Location*) представляет собой динамическую страницу, описывающую индивидуальную логику работы. У локации, как и у модели, прямыми потомками могут быть только наборы. Большинство атрибутов у наборов и блоков, присутствующих в модели, может быть переопределено в локации, либо определено вновь. Существуют также атрибуты, характерные только для локации. То есть локация задаёт уточняющие свойства обработки тех или иных данных. Исходя из этого, можно утверждать, что локации в парадигме MVC соответствуют *контроллерам*.

Локация идентифицируется шаблоном адреса. Шаблон адреса представляет собой часть URI, идентифицирующую путь, и содержащую динамические вставки, на место которых подставляются данные определённого типа, фигурирующие в генерации динамической страницы. Шаблон адреса динамической страницы задаётся атрибутом *base* (базовый адрес).

Набор (элемент *Set*) является коллекцией данных различного типа. Набор идентифицируется уникальным именем, с помощью которого набор может быть определён в модели и локации. Имя задаётся с помощью атрибута *name*. В контексте модели набор определяет структуру данных отдельно взятой коллекции и её отношение с другими наборами. Результатом выборки из набора является один или множество **экземпляров набора**. Выборка из набора осуществляется при обработке соответствующей локации, в которой задействован этот набор.

Блок (элемент Block) является непосредственной частью набора и представляет собой определённый тип данных. Тип данных может быть как элементарным, так и представлять собой сложную структуру, в том числе осуществляющей функции файловой обработки. В методологии BlockSet блок при любых обстоятельствах является элементарным типом данных, то есть обладает свойством *атомарности* [11]. Предварительное объявление блока у соответствующего набора в модели также является обязательным, как и объявление самого набора.

Условно, набор представляет собой таблицу в базе данных, а блок – поле в этой таблице. Отличие в том, что блок может обрабатывать сложные данные, оставаясь при этом элементарной сущностью в контексте методологии BlockSet. В некоторых случаях блок может брать на себя обработку сложных типов и структур, которые не ограничиваются лишь одним полем, а также работать с файлами. Кроме того, блок выполняет функцию *валидации* данных. Именно внутри блока происходит проверка на соответствие входных данных своему типу. Блоку каждого типа может соответствовать разный набор атрибутов. Это особенность методологии.

Оперируя понятиями методологии, разберём пример гипотетического проекта, логическая структура которого описывается на языке BML (листинг 1). Отметим, что для наглядности в листинге отсутствует упоминание блоков, задающих права доступа (о них упоминалось в [12]).

Листинг 1. Модель гипотетического проекта «диссертационные советы»

```
<model>
	<set name="specialty">
		<block name="cipher" type="string" />
		<block name="name" type="string" />
		<set name="council" relation="multi">
			<block name="org" type="string" />
			<block name="cipher" type="string" />
			<set name="thesis" relation="parent">
				<block name="subject" type="string" />
				<block name="author" type="string" />
				<block name="supervisor" type="string" />
				<block name="pubdate" type="datetime" />
				<set name="related_docs" relation="multi">
					<block name="pubdate" type="datetime" />
					<block name="name" type="string" />
					<block   name="attachment"   type="file"
ext="pdf doc rtf odt" />
				</set>
			</set>
		</set>
	</set>
</model>
```

Гипотетический проект представляет собой базу диссертационных советов, проводящих защиту диссертаций по различным специальностям. Как можно видеть на структуре модели, проект содержит четыре набора: *specialty* (научная специальность), *council* (диссертационный совет), *thesis* (диссертация), *related_docs* (сопутствующие документы). Диссертационных советов может быть несколько, как и специальностей, по которым принимается защита. Поэтому у дочернего к *“specialty”* набора *“council”* установлен атрибут *“relation”* со значением *“multi”*, что означает связь «многие ко многим» между наборами, если оперировать понятиями баз данных. Аналогичный атрибут установлен у набора “*thesis*” и набора “*relation_docs*”, т.к. диссертационный совет может принимать защиту множества диссертаций, а каждая диссертация может содержать множество сопутствующих документов. У каждого блока есть свой тип, а у каждого типа может быть свой набор атрибутов, задающих, в том числе, и правила валидации данных. Например, блок “*attachment*”, являющийся частью набора сопутствующих документов и представляющий собой вложенный файл, принимает только документы в формате PDF, DOC, RTF и ODT. Данные, сохраняемые в блоках других типов, также могут валидироваться, согласно правилам, установленным в атрибутах этих блоков.

Листинг 2. Пример локации

```
	<location base="/diss/[council:cipher]/"
template="list.tpl">
```

```
        <set name="specialty">
              <block name="cipher" />
              <block name="name" />
              <set name="council">
                    <block name="org" />
                    <block name="cipher" />
                    <set name="thesis" limit="20">
                          <block name="subject" />
                          <block name="author" />
                          <block name="supervisor" />
                          <block name="pubdate"
condition=">01.01.2014"/>
                          <set name="related_docs">
                                <block name="pubdate" />
                                <block name="name" />
                                <block name="attachment"/>
                          </set>
                    </set>
              </set>
        </set>
</location>
```

Локация, как уже было сказано выше, представляет собой одну из динамических страниц проекта. В локации необходимо продублировать все наборы и блоки, участвующие в выборке данных. В примере (листинг 2) представлена динамическая страница, отображающая список диссертаций определённого совета. Значение в атрибуте "*base*", заключённое в квадратные скобки, ([council:cipher]) обозначает, что на этом месте в строке адреса ожидается строка, идентифицирующая шифр диссертационного совета, и эта строка будет задействована при выборке данных. Все выбранные данные будут переданы в шаблонизатор, а в качестве шаблона будет задействован файл list.tpl (атрибут "*template*"). Кроме того, диссертации не должны быть старше даты 1 января 2014 года (атрибут "*condition*" у блока "*pubdate*"). Также выборка из набора "*council*" ограничена количеством 20 экземпляров (атрибут "*limit*").

Проведём аналогию использования парадигм программирования со строительством загородного дома. Непосредственное участие субъекта в постройке дома: укладка кирпича, установка окон и дверей, монтирование коммуникаций – всё это похоже на императивное программирование, т.к. дом строится на изначально пустом месте, а использование готовых компонентов (стройматериалов, окон и дверей, электрооборудования, сантехники) очень похоже на использование заранее подготовленных библиотек, коих существует огромное количество.

Стоит также добавить, что низкоуровневое программирование напоминает аналогичный процесс с оговоркой на то, что все готовые компоненты приходиться изготавливать самостоятельно из сырья: выжигать кирпич, вытачивать брус, выплавлять металлические изделия. Несмотря на свою сложность, достоинством этого процесса является то, что на выходе получаются именно те компоненты, какими они задумывались архитекторами и декораторами дома, в то время как заранее заготовленные компоненты не всегда соответствуют изначальной задумке проекта, а изменить их вид уже невозможно.

Декларативное программирование же напоминает работу архитектора, когда тот не задумывается подробно о том, как именно будет строиться та или иная часть дома, однако он чётко знает, какой вид должно принять определённое помещение, где и сколько должно быть установлено окон и дверей, как должен выглядеть фасад дома и многое другое. По макету, разработанному архитектором, производится непосредственное строительство. Этот процесс аналогичен **интерпретации** программы. Разработка эффективного и производительного интерпретатора является одной из задач работы.

На рисунке 2 представлена расширенная иерархическая модель в объёмном виде, на которой показано место методологии BlockSet и языка BML в сфере веб-разработки. Нижние два уровня остаются такими же, как в упомянутой ранее модели. Верхние три уровня левой стороны пирамиды являются непосредственными частями BlockSet и занимают смежные позиции с существующими технологиями. Так, на представленной схеме язык BML находится на уровне фреймворков, однако фреймворком не является, т.к. фреймворк подразумевает набор библиотек уже существующего языка программирования. Тем не менее, по характеристикам, которые

описывает предыдущая модель – скорость разработки и гибкость – предметно-ориентированный язык BML закономерно располагается именно на этом уровне.

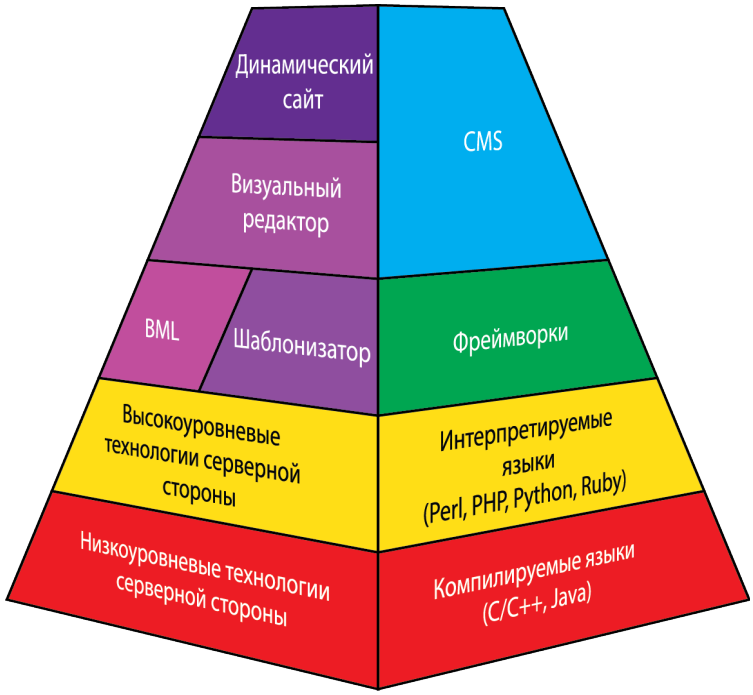


Рис. 2. Место методологии BlockSet и языка BML в сфере веб-разработки

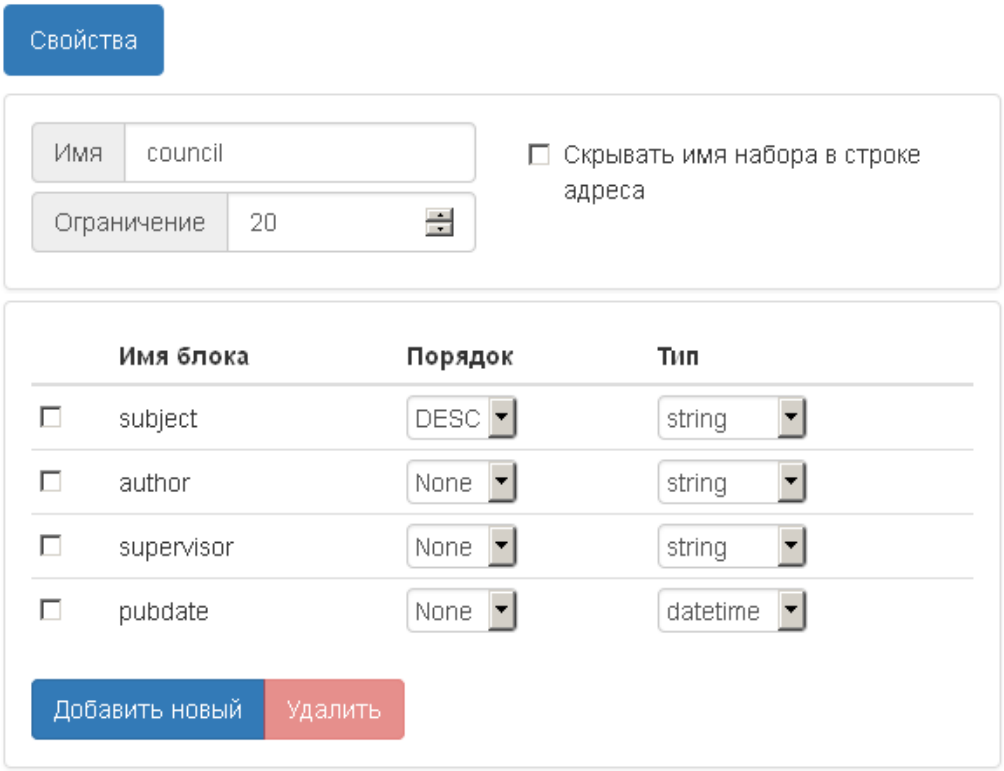


Рис. 3. Окно создания нового набора

Методология BlockSet следует парадигме **MVC** или **Model-View-Controller** (*модель-представление-контроллер*) [13]. Уровни модели и контроллера строятся в языке BML, а роль представления берёт на себя шаблонизатор, получающий сгенерированные данные и накладывающий их на определённый шаблон. Шаблон представляет собой страницу на языке HTML с простейшими алгоритмическими структурами и специальными вставками, на место

которых будут подставлены данные в процессе генерации документа. Работа интерпретатора делится на два этапа: непосредственная интерпретация структур BML и передача выходных данных шаблонизатору. Именно по этой причине шаблоны и язык BML расположены на одном уровне.

Уровнем выше, наряду с системами управления контентом, расположен специализированный визуальный редактор, позволяющий наглядным образом, с помощью компонентов пользовательского интерфейса, строить конструкции языка BML (рис. 3), а также корректные шаблоны HTML, задавать стили CSS, составлять интерактивные алгоритмы JavaScript. По сути, визуальный редактор генерирует шаблоны для серверной стороны, а также целиком отвечает за разработку клиентской части. Часть функций по разработке клиентской части выполняют и системы управления контентом [14], как правило, имеющие в своём арсенале простейший визуальный редактор HTML/CSS.

Стоит оговориться, что визуальный редактор BlockSet не предназначен для управления содержимым. Он лишь предоставляет эргономичный интерфейс, в котором сочетается необходимый функционал для генерации динамических структур BML и работой с клиентской частью. Правильно построенный проект сам берёт на себя роль системы управления контентом и в целом представляет собой динамический веб-ресурс с возможностью разграничения прав доступа к своему содержимому. Именно поэтому законченный проект находится на самой вершине и также делит место на этом уровне наравне с системами управления контентом.

Описанная методология представляет собой новое средство веб-разработки, ориентированное как на начинающих веб-разработчиков, так и профессионалов индустрии. Главной особенностью является декларативная разработка, когда разработчик оперирует ограниченным набором сущностей, но может максимально изменить поведение, добавляя новые свойства каждой сущности. Взятый за основу язык XML предполагает минимизацию синтаксических ошибок, в силу своей распространённости, а визуальный редактор помогает ускорить процесс освоения новой методологии благодаря наглядности.

Литература

1. Ahmed A. et al. Agile software development: Impact on productivity and quality // Management of Innovation and Technology (ICMIT), 2010 IEEE International Conference on. – IEEE, 2010. – С. 287-291.
2. Кейно П.П., Силуянов А.В. Разработка и внедрение интерпретатора декларативного языка моделирования Web-интерфейсов на высоконагруженных системах // Прикладная информатика. 2015. № 1. С. 5–20.
3. Кейно П. П. Визуальный редактор BlockSet. Свидетельство о государственной регистрации программы для ЭВМ 2014662832, Dec 10, 2014.
4. Brodie M. L., Mylopoulos J., Schmidt J. W. (ed.). On conceptual modelling: Perspectives from artificial intelligence, databases, and programming languages. – Springer Science & Business Media, 2012.
5. Apte V., Hansen T., Reeser P. Performance comparison of dynamic web platforms // Computer Communications. – 2003. – Т. 26. – №. 8. – С. 888-898.
6. Tanenbaum A. S. Structured computer organization. – Pearson, 2006.
7. Battle R., Benson E. Bridging the semantic Web and Web 2.0 with representational state transfer (REST) //Web Semantics: Science, Services and Agents on the World Wide Web. – 2008. – Т. 6. – №. 1. – С. 61-69.
8. Groenewegen D. M., Visser E. Integration of data validation and user interface concerns in a DSL for web applications //Software & Systems Modeling. – 2013. – Т. 12. – №. 1. – С. 35-52.
9. Sun H. M., Chen Y. H., Lin Y. H. oPass: A user authentication protocol resistant to password stealing and password reuse attacks // Information Forensics and Security, IEEE Transactions on. – 2012. – Т. 7. – №. 2. – С. 651-663.
10. Кейно П.П., Силуянов А.В. Автоматизированная разработка динамических Web-узлов средствами декларативного языка программирования // Прикладная информатика. 2015. № 1. С. 55–70.
11. Bhiri S., Perrin O., Godart C. Ensuring required failure atomicity of composite web services // Proceedings of the 14th international conference on World Wide Web. – ACM, 2005. – С. 138-147.
12. Кейно П.П. Унификация типов прав доступа к информационным ресурсам облачных Web-сервисов // Материалы XXII всероссийской научно-практической конференции «Проблемы информационной безопасности в системе высшей школы», Безопасность информационных технологий, НИЯУ-«МИФИ». 2015
13. Deacon J. Model-view-controller (mvc) architecture // [Online][Citado em: 10 de março de 2006.] http://www.jdl.co.uk/briefings/MVC.pdf. – 2009.
14. Báscones P., Carreras C. Managing memory institutions portals: from HTML to CMS and towards applications in XML for multi-platforms //International Journal of Digital Culture and Electronic Tourism. – 2008. – Т. 1. – №. 1. – С. 18-36.

Гарфутдинова А.Р. [1], Макаровских Т.А. [2]

[1] Южно-Уральский государственный университет, г.Челябинск, магистрант кафедры экономико-математических методов и статистики, alina_garfutdinova@mail.ru

[2] Южно-Уральский государственный университет, г.Челябинск, к.ф.-м.н., доцент кафедры экономико-математических методов и статистики, kwark@mail.ru

АВТОМАТИЗИРОВАННАЯ СИСТЕМА ДЛЯ ОРГАНИЗАЦИИ И ПРОВЕДЕНИЯ КОНФЕРЕНЦИИ

КЛЮЧЕВЫЕ СЛОВА

СУБД, автоматизация, 1С:Предприятие.

АННОТАЦИЯ

В статье рассматривается вопрос автоматизации деятельности Оргкомитета при проведении научных мероприятий. На сегодняшний день по ряду причин отсутствует единое программное решение по поддержке проведения научных конференций. В качестве такого решения предложена информационная система, разработанная на основе платформы 1С:Предприятие 8.3.

В настоящее время неотъемлемой частью деятельности высших учебных заведений является регулярное проведение научных конференций, семинаров, совещаний и др. Основная цель организации подобных научных мероприятий в учебных заведениях – это развитие и поддержка научных исследований как основы фундаментализации образования, как базы подготовки работников квалифицированного труда в соответствии с потребностями государства и общества. Масштабы научных мероприятий могут достигать как межвузовских и региональных, так и международных уровней. Такие собрания требуют интенсивной работы, полной отдачи со стороны всех ее участников, главным образом организаторов. Так для проведения научного мероприятия на высококачественном уровне необходима основательная подготовка. Важно оправдать ожидания участников конференций, так как высокий уровень организации является основанием для укрепления статуса конференции, увеличения количества участников, привлечения интересных докладчиков. Эффективность подготовки и проведения мероприятия в значительной степени повышается за счет использования Оргкомитетом вспомогательных программных средств.

В настоящее время известно немало программного обеспечения для помощи Оргкомитету в сборе и систематизации информации. К ним можно отнести, например, бесплатный портал Агора [1], его иностранный аналог EasyChair Conference System [2] и 1С:Битрикс «Сайт конференции» [3]. Все перечисленные ресурсы удобны и позволяют быстро и качественно собрать информацию об участниках конференции, загрузить и провести рецензирование их докладов.

Однако для организации любой конференции не достаточно тех сервисов, которые предлагают вышеперечисленные ресурсы. Например, перед оргкомитетом могут стоять следующие задачи: сбор организационного взноса, оплата мероприятий конференции (аренда залов, организация кофе-брейков, работа секретарей, проведение культурной программы и пр.), составление расписания работы секций, организация встреч и проводов участников, расселение участников и т.д.. Ни один из вышеперечисленных интернет-сервисов не позволяет решить всю совокупность вопросов [4]. Существующий программный комплекс AE Software [5] выгодно отличает от других только то, что он позволяет автоматизировать регистрацию оргвзносов.

Проанализируем некоторые известные системы поддержки конференций в соответствии со следующими критериями (табл. 1):

- доступ – платный/бесплатный/нет доступа (п/б/–);
- возможность создания и поддержки сайта конференции;
- возможность регистрации заявок;
- рецензирование – назначение рецензента на каждую заявку;
- публикация данных конференции – возможность публикации списка участников и списка докладов на основе поданных заявок;

- организация рассылки новостей;
- формирование отчетов по многокритериальной выборке;
- составление программы конференции – распределение по секциям принятых докладов;
- учет оплаты регистрационных взносов;
- расселение иногородних участников.

Таблица 1 – Сравнительная таблица информационных систем

	Условия доступа	Сайт	Регистрация участников	Рецензирование **Докладов**	Публикация данных конференции	**Рассылка e-mail** сообщений	Отчеты	Программа конференции	Оргвзносы	Расселение участников
Агора [1]	Б	+	+	+	–	+	+	–	–	–
1С-Битрикс: **Сайт конференции [3]**	П	+	+	–	+	+	–	–	–	–
AE Software [5]	Б	+	+	–	–	+	–	–	–	–
Система [6]	–	+	+	–	+	+	–	+	–	+
Система [7]	–	+	+	+	+	–	+	+	+	+
Система [8]	–	+	+	–	+	–	+	–	–	–
ИС СО РАН [12]	Б	+	+	+	+	+	+	+	–	–
My Review [9]	Б	+	+	+	+	+	+	–	+	–
Open Conference System [10]	Б	+	+	+	+	+	–	–	+	–
Indico [11]	Б	+	+	+	+	+	+	+	+	–

Таким образом, для структурных подразделений высших учебных заведений имеет смысл разработать свое программное обеспечение для поддержки проведения конференций. Поскольку каждое научное мероприятие имеет свою специфику, в каждом случае бизнес-процессы и требования к информационному сопровождению могут отличаться. В силу этого, важным требованием к системам поддержки является возможность быстрой адаптации под задачи конкретной конференции. Такой гибкостью конфигурирования примечательна платформа 1С:Предприятие. Ввиду этого, автор предлагает разработку конфигурации «1С:Конференция» с программным решением следующих задач:

- регистрация участников конференции;
- составление расписания работы секций;
- распределение иногородних участников по номерам в гостинице;
- организация транспортного обслуживания;
- формирование отчетов для конференции.
- работа с потоками денежных средств (фондами конференции).

В настоящее время реализовано решение вопросов (1), (3) и (4).

Схема базы данных конфигурации В ходе реализации информационной системы была построена ER-диаграмма предметной области (рис 1).

Выделены сущности схемы метаданных: «Участник», «Гостиница», «Тип номера», «Организатор», «Адрес» и др. Сущности схемы метаданных в среде разработки 1С:Предприятие очень удобно описать с помощью объекта Справочники.

Решение поставленных задач реализовано с помощью предлагаемых платформой объектов Документы, Регистры накопления, Регистры сведений, Отчеты.

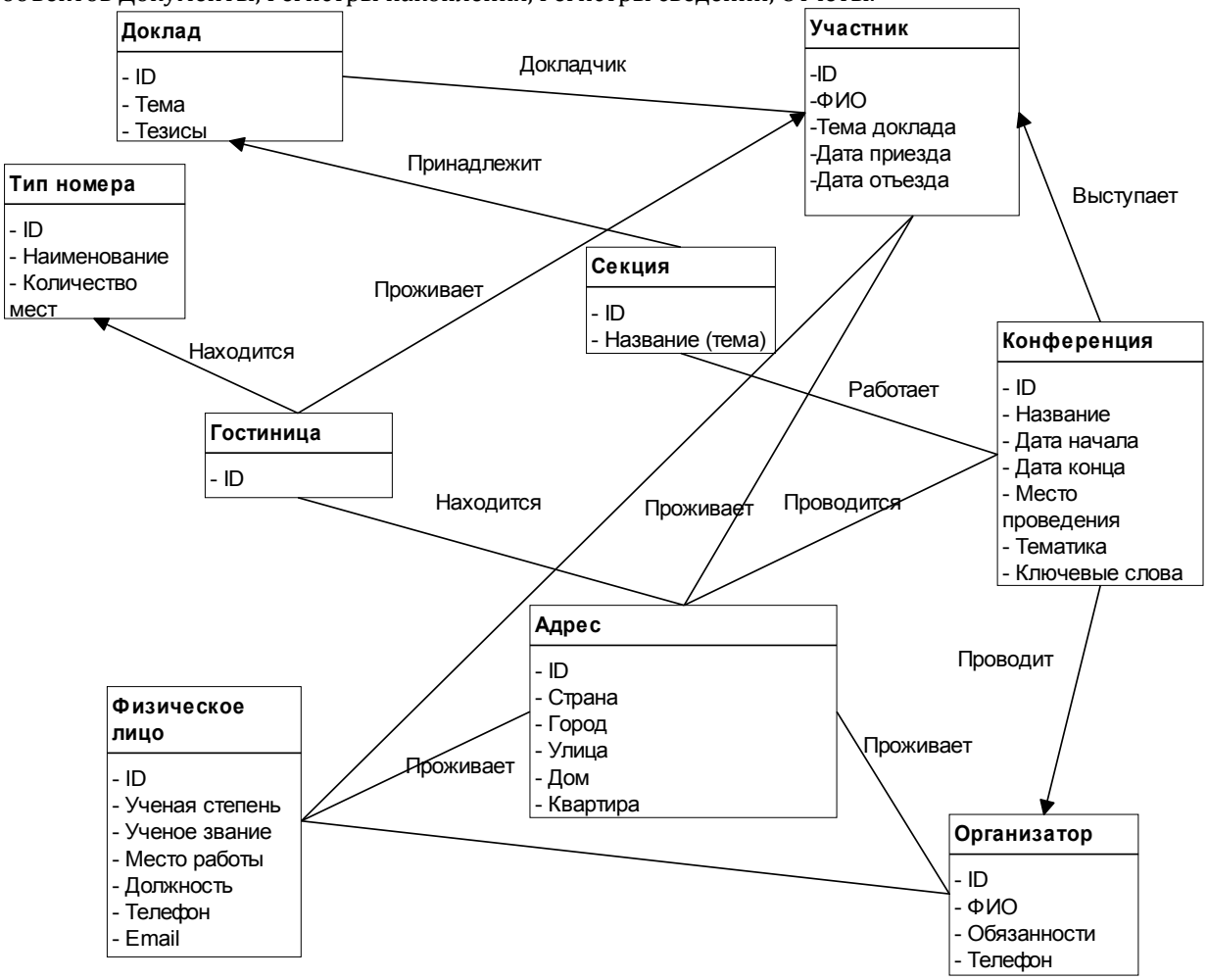


Рис. 1. ER-диаграмма системы «1С: Конференция»

Бронирование мест в гостиницах для последующего расселения участников осуществляется документом «ЗаказГостиницы». Для подбора и бронирования гостиничного номера для участника в соответствии его пожеланиям реализован документ «БронированиеНомера». Для оперативного получения информации о свободных гостиничных номерах в процессе расселения участников используется регистр накопления «ОстаткиНомеров».

Для планирования транспортного обслуживания иногородних участников реализованы документы «ЗаказАвтомобиля» и «ЗаявкаНаТрансфер» и регистр накопления «ТранспортныеУслуги».

Результаты решения задачи расселения иногородних участников и организации для них трансфера предоставляются соответствующими отчетами «ПоГостиницам» и «ТранспортныеУслуги».

Рассмотрим более подробно задачи бронирования гостиниц и организации сопровождения участников.

Задача бронирования номеров гостиницы для иногородних участников возникает при подготовке мероприятия любого масштаба (от 50 до 1000 человек). Для размещения иногородних участников могут быть предложены гостиницы, общежития, пансионаты. При размещении в гостинице следует учитывать:

- уровень мероприятия;
- количество участников;
- социальный статус участника;
- финансовые условия;
- территориальную удаленность от места проведения мероприятия.

В 1С:Конференция процесс расселения участников по забронированным гостиницам осуществляется в два этапа.

Этап 1. *Поиск и бронирование гостиниц, расположенных наиболее удобно по месту проведения конференции.* Для заполнения информации о забронированных гостиницах в 1С:Конференция создан справочник «Гостиница».

С помощью документа «Заказ гостиницы» Оргкомитет резервирует номера гостиниц, заполняет категорию и стоимость номера. При проведении документа «ЗаказГостиницы» производится запись в регистр сведений «ЦеныНаНомера». Данный регистр сведений хранит информацию по стоимости номеров гостиниц в сутки. Также документ «ЗаказГостиницы» осуществляет движение прихода по регистру накопления «ОстаткиНомеров». Этот регистр является основным в реализации задачи расселения, так как содержит необходимую информацию о наличии номеров определенной категории.

Этап 2. *Расселение участников по забронированным номерам гостиниц.* С учетом пожеланий участника, указанным в заявке гостиницы и категории гостиничного номера, производится подбор подходящего номера. Данная операция в 1С:Конференция реализована в документе «БронированиеНомера». Для удобства предусмотрена возможность вызова формы документа при заполнении регистрационных данных участника, кнопкой Забронировать номер формы справочника «Участники». При открытии формы документа «Бронирование номера» командой «Забронировать номер», поля документа Гость, Дата приезда и Дата отъезда заполняются автоматически копированием соответствующих реквизитов справочника «Участники». Для исключения повторного бронирования номера на одного и того же человека при заполнении поля Участник вызывается функция проверки на наличие брони на текущего участника.

Поиск гостиничного номера для участника, в соответствии его пожеланиям, реализован в обработке команды «Найти» документа «Бронирование номеров». Подбор свободных гостиничных номеров по полям Гостиница, ТипНомера, ДатаПриезда, ДатаОтъезда документа выполняется запросом к виртуальной таблице регистра накопления «ОстаткиНомеровОстаткиИОбороты».

Результаты расселения участников по гостиничным номерам наглядно отражает отчет «РасселениеПоВсемГостиницам» (рис. 2). Для просмотра информации о расселении по каждой гостинице в отдельности реализован отчет «РасселениеПоОпределеннойГостинице» (рис. 3). Отчеты разработаны с помощью механизма Система компоновки данных, предлагаемого платформой 1С:Предприятие.

Гостиница	Тип номера	Номер	Дата заезда	Дата выезда	Гость
Турист					
	Двухместный				
		202			
			09.07.2015 0:00:00	15.07.2015 0:00:00	Ухоботов
	Одноместный				
		101			
			09.07.2015 0:00:00	13.07.2015 0:00:00	Михайлова
		102			
			07.07.2015 0:00:00	20.07.2015 0:00:00	Тигрова
Уют					
	Двухместный				
		201			
			27.06.2015 0:00:00	13.07.2015 0:00:00	Финееева
		202			
			09.07.2015 0:00:00	11.07.2015 0:00:00	Смирнов
	Одноместный				
		101			

Рис. 2. Отчет «РасселениеПоВсемГостиницам»

Рис. 3. Отчет «РасселениеПоОпределеннойГостинице»

Задача планирования оказания транспортного обслуживания иногородних участников возникает в зависимости от масштаба мероприятия и условий его проведения. Как правило, для иногородних участников Оргкомитет организовывает трансфер от места прибытия (аэропорта или ж/д вокзала) до гостиницы и отъезд из гостиницы до места оправления в последний день пребывания. График встреч (проводов) составляется на основе даты и времени приезда (отъезда) участников. При этом почетному участнику независимо от количества прибывающих вместе с ним коллег предоставляется отдельный транспорт.

Так как большую часть организаторов необходимо задействовать в конференц-зале, для предоставления транспортных услуг должно быть назначено как можно меньше человек, при этом качество транспортного обслуживания гостей должно остаться на высоком уровне. Таким образом, график работы организаторов, ответственных за предоставление трансфера, необходимо оптимизировать. В 1С:Конференция реализовано решение данного вопроса. Процесс распределения организаторов по заявкам участников на транспортное обслуживание состоит из двух этапов.

Этап 1. *Подбор резерва водителей.* Для подбора резерва водителей в системе нужно выполнить следующие действия:

- в справочнике «РезервВодителей» заполнить данные об организаторах, имеющих в наличии автомобиль;

- по мере необходимости проведением документа «ЗаказАвтомобилей» добавить организатора из резерва водителей в список ответственных за транспортные услуги. При проведении документ записывает движение прихода в регистр накопления «ТранспУслуга». Сначала в списке ответственных за транспортные услуги находится только один представитель Оргкомитета.

Этап 2. *Назначение водителя на заявку на трансфер.* Этот этап реализован документом «Заявка на трансфер». Поля документа заполняются данными из отправленных участниками заявок на транспорт. После заполнения документа нажатием кнопки «Назначить водителя» производится поиск свободного автомобиля на время приезда (отъезда) участника (рис. 4). Для исключения повторного бронирования услуги на транспортное обслуживание участника в модуле формы документа «Заявка на трансфер» реализована функция, которая проверяет, имеется ли бронь транспорта на данного участника.

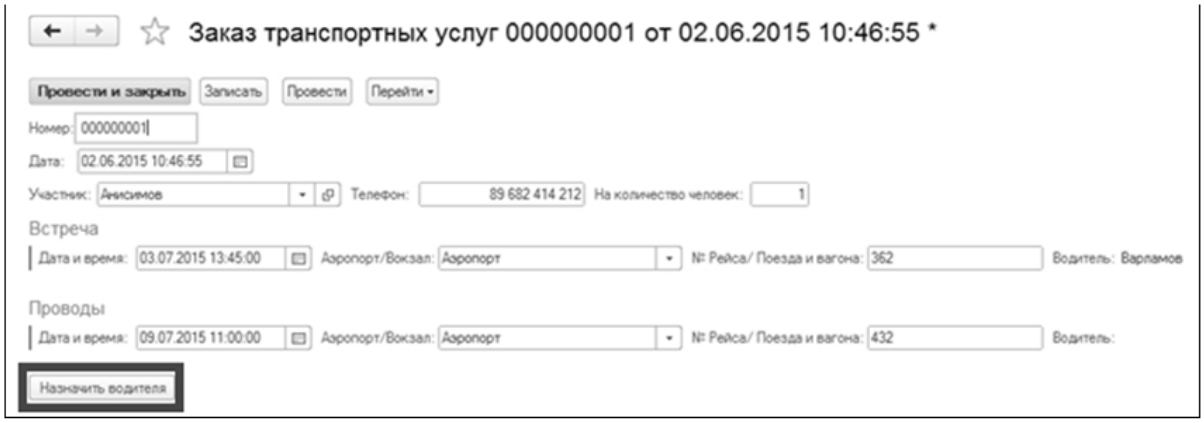


Рис.4. Создание документа «ЗаявкаНаТрансфер»

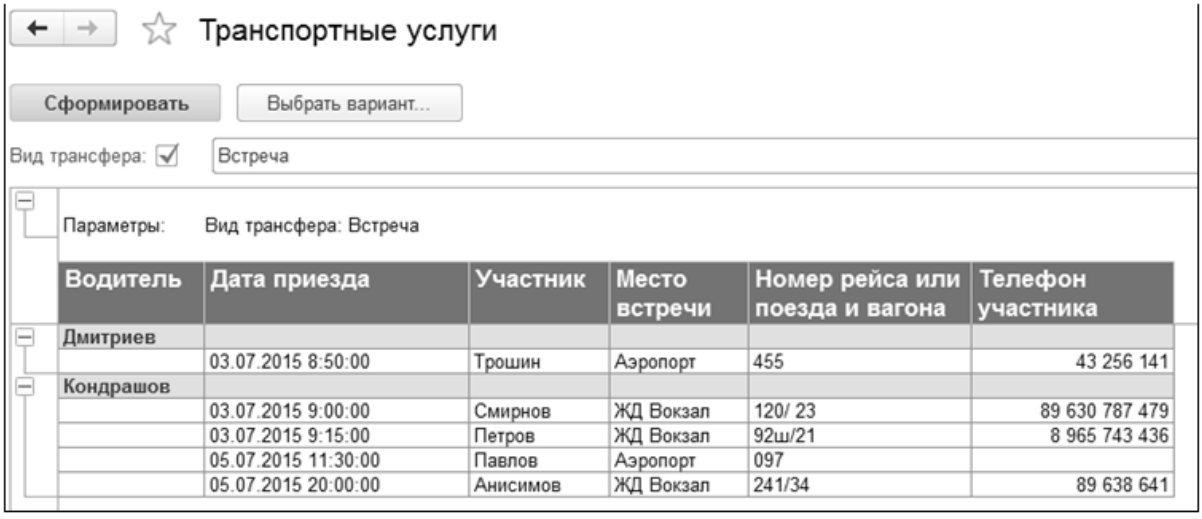
При реализации функции подбора свободного автомобиля на заявку участника предполагаются следующие ограничения:

- один автомобиль – три человека;
- один автомобиль назначается для участников разница во времени между прибытием (отправлением) которых не более полчаса и место встречи (проводов) совпадают;
- один рейс занимает по времени два с половиной часа.

Нужно заметить, что данные ограничения несложно изменить в соответствии с особенностями проводимого мероприятия.

Таким образом, если в свободном остатке есть подходящие автомобили, производится назначение водителя на заявку и становится возможным проведение документа «ЗаявкаНаТрансфер». При проведении документа на дату приезда (отъезда) записывается расходное движение в регистр накопления «ТранспУслуга», а период для движения прихода рассчитывается как период расхода плюс два с половиной часа. Тем самым обеспечивается резервирование водителя на данный промежуток времени. В случае отсутствия свободных автомобилей на нужный промежуток времени, система оповещает пользователя. Следовательно, необходимо с помощью документа «ЗаказАтомобиля» назначить на оказание транспортного обслуживания еще одного организатора.

В результате обработки всех заявок участников на транспортное обслуживание, будет сформирован план оказания данной услуги. График встреч (проводов) участников отражает отчет Отчет «ТранспортныеУслуги» (рис. 5). Указав поле «ВидТрансфера», можно просмотреть график встреч и проводов по отдельности. Отчет «ТранспУслугиПоВодителю» отражает график работы конкретного организатора, ответственного за трансфер (рис. 6).



Водитель	Дата приезда	Участник	Место встречи	Номер рейса или поезда и вагона	Телефон участника
Дмитриев					
	03.07.2015 8:50:00	Трошин	Аэропорт	455	43 256 141
Кондрашов					
	03.07.2015 9:00:00	Смирнов	ЖД Вокзал	120/ 23	89 630 787 479
	03.07.2015 9:15:00	Петров	ЖД Вокзал	92ш/21	8 965 743 436
	05.07.2015 11:30:00	Павлов	Аэропорт	097	
	05.07.2015 20:00:00	Анисимов	ЖД Вокзал	241/34	89 638 641

Рис. 5. Отчет «ТранспортныеУслуги»

Рис.6. Отчет «ТранспУслугиПоВодителю»

Заключение

Разработана конфигурация 1С:Конференция. Конфигурация обладает следующими функциональными возможностями:

- регистрация участников конференции;
- расселение иногородних участников по гостиничным номерам;
- планирование транспортного обслуживания иногородних участников.

В дальнейшем планируется расширение функционала информационной системы.

Разрабатываемое программное обеспечение значительно упростит работу Оргкомитета и повысит уровень организации мероприятия.

Литература

1. Агора: Служба автоматизации создания , размещения и поддержки интернет-страниц конференций [Офиц. сайт]. URL: http://agora.guru.ru (дата обращения: 15.06.2015).
2. EasyChair: conference management system [Интернет-портал]. URL: http://www.easychair.org/ (дата обращения: 15.06.2015).
3. 1С:Битрикс. Сайт конференции [Офиц. сайт]. URL: http://www.1c-bitrix.ru/solutions/conf/ (дата обращения: 15.06.2015).
4. Панюкова Т.А. Информационная база для организации документооборота научной конференции // Новые информационные технологии в образовании: сб. науч. тр. – Москва: ООО «1С-Паблишинг», 2013. – С. 193-194.
5. Организация конференций на базе программного комплекса AEvents Software [Интернет-портал]. URL: http://onlinereg.ru/site.php?go=42&page=15=RUS/ (дата обращения: 15.06.2015).
6. Глот И.О. Разработка информационной системы по проведению конференций/ Глот И.О., Масич Г.Ф., Перетятько С.С., Цаплина Г.С.// Труды Всероссийской научной конференции . – 2002 г. – № 9, с.162–164.
7. Алексеев А.Н. Подсистема проведения конференции/ Алексеев А.Н., Бездушный А.Н., Масич Г.Ф., Созыкин А.В.// Российский научный электронный журнал Электронные библиотеки . – 2005 г.– № 3, с. 56–58.
8. Квиринг А.В. Программный комплекс для конструирования информационной системы проведения конференций/ Сборник института сплошных сред УрО РАН/ /Квиринг А.В., Кузякин Ю.И., Масич Г.Ф., Перетятько С.С., Рогачев С.Н. – 2002 г.
9. The MyReview system [Интернет-портал]. URL: http://myreview.lri.fr/ (дата обращения: 15.06.2015).
10. Public knowledge project: Open Conference System [Интернет-портал]. URL: https://pkp.sfu.ca/ocs (дата обращения: 15.06.2015).
11. Indico: The effortless open source tool for event organization, archival and collaboration [Интернет-портал]. URL: http://indico-software.org/ (дата обращения: 15.06.2015).
12. Конференции Уральского отделения РАН [Интернет-портал]. URL: http://confer.uran.ru/ (дата обращения: 15.06.2015).
13. Гуськов А.Е. Средства поддержки проведения научных конференций: обзор и сравнение/ Гуськов А.Е., Васильков А. В. – Институт вычислительных технологий СО РАН г. Новосибирск. – 2003 г.
14. Гуськов А.Е. О задачах создания систем для поддержки проведения научных конференций/ А.Е. Гуськов. – Институт вычислительных технологий СО РАН г. Новосибирск. – 2004г.
15. Учебник по 1С:Предприятие [Интернет-портал]. URL: http://www.mista.ru/1c/ (дата обращения: 15.06.2015).

Мунерман В.И.[1], Симакова А.А.[2]

[1]Смоленский государственный университет, г. Смоленск, доцент кафедры информатики, vimoon@gmail.com

[2]Смоленский государственный университет, г. Смоленск, студентка физико-математического факультета, bessima13@yandex.ru

АЛГЕБРАИЧЕСКИЙ ПОДХОД К ФОРМАЛИЗАЦИИ СЕМАНТИКИ ЗАДАЧ

КЛЮЧЕВЫЕ СЛОВА

Семантика задачи, алгебра, семантическая сеть, система интегрированного вида, нейронная сеть, искусственный интеллект, нечёткие множества.

АННОТАЦИЯ

В статье рассматривается алгебраический подход, который позволит решать проблему формализации задачи предметной области с помощью алгебраических систем, формализующих объекты предметной области и связи между ними. Для решения задачи предложена система интегрированного видана основе методов искусственного интеллекта. Показано, как решение поставленной задачи может быть осуществлено с использованием систем Mashing Learning.

Процесс решения задачи, как правило, включает два этапа: представление задачи и поиск (перебор). Успех решения задачи в значительной мере определяется формой ее представления. Формы представления задачи могут быть различными и зависят как от природы самой задачи, так и от ее решателя.

Например, выбирая кратчайший маршрут движения, человек может воспользоваться схемным представлением возможных путей перемещения и оценками расстояний между промежуточными пунктами.

Характер человеческого мышления таков, что форма представления задачи, которую он использует, не всегда им осознаются. Так, шахматист не может четко объяснить форму представления шахматной ситуации, позволяющую ему не перебирать все возможные варианты продолжения при поиске очередного хода. Поэтому этап представления задачи часто выпадает из поля зрения человека. Тем не менее, важность этого этапа осознается сразу же при попытке построить программно - реализуемый алгоритм решения задачи.

В настоящее время при решении задач в математических пакетах нам приходится заранее изучить их документацию, ознакомится с их синтаксисом, и только потом приступать к поиску решения. Все это происходит потому, что нет единой формы записи условия задачи, человеку приходится изначально обдумать её решение, а зачастую даже сначала решить самому, и только потом вводить входные данные в вычислительную систему для обработки. Очевидно, что форма представления задачи, то есть её формальная модель, играет важную роль при нахождении решения с применением вычислительных средств.

Одной из форм представления задачи является построение её алгебраической системы. Было замечено, что человек, прочтя впервые некоторое условие задачи, осознает, с какой алгеброй ему предстоит в дальнейшем работать. Не прикладывая особых усилий, он понимает, какие операции ему могут понадобиться для её решения, а так же какова структура задачи. Так постепенно человек делает незначительные шаги, продвигаясь к основной цели – решению поставленной задачи. Именно подобные шаги способствуют продвижению человека непосредственно к правильному решению. Если вычислительная система сможет проделать подобные операции, то станет возможным автоматизировать процесс конкретизации пространства решений, в котором осуществляется поиск решения задачи. Такой подход позволит сформировать единый формальный язык представления задач.

Далее рассматривается один способ формализации условия задачи, основанный на подходе, который может обеспечить автоматизированный подбор алгебры для формализации решения задачи. Подбор осуществляется на основе семантики задачи.

В дальнейшем под алгеброй понимается конечное множество, вида:

$$\langle S, M, \Omega_s, \Omega_M \rangle . \quad (1)$$

Здесь:

- S – структура задачи, например, матрица, граф и тому подобные структуры;
- M – множество, которому принадлежат элементы структуры, например, множества натуральных, действительных чисел, множество {0, 1};
- Ω_s – множество операций над S;
- Ω_M – множество операций над M.

То есть, и структура, и элементы должны представлять собой универсальные алгебры или универсальные алгебраические системы. Вместе они образуют двухосновную универсальную алгебру (алгебраическую систему).

Например, в задачах на графах, в частности в задаче поиска кратчайшего пути от одной вершины до остальных (на самом деле это могут быть задачи различных классов, но обладающих некими свойствами). В данном случае имеем некую структуру, именуемую графом и фиксированную операцию преобразования этой структуры (например, алгоритм Флоида-Уоршелла), а также известно конкретное арифметическое множество (N, R и т. п.), к которому будут принадлежать элементы данной структуры. Итак, в подобных задачах алгеброй будет являться следующее множество объектов <G, X, A, Ω x>, где G – граф, X – некоторое множество, A - алгоритм Флоида-Уоршелла, Ω x - операции над A.

Реализацию подобной системы было решено разбить на два этапа:

- этап лингвистической обработки данных (этап формализации и преобразования полученных данных);
- этап машинного обучения (этап накопления, обработки и принятия решения).

Оба находятся в тесной взаимосвязи, рассмотрим их подробнее.

Этап лингвистической обработки подразумевает работу с неструктурированными данными, представленными в обычной форме. На этом этапе мы строим модель знаний, то есть переходим от знаний, представленных в текстовой форме, к их аналогам, пригодных для ввода в память интеллектуальной системы, т.е. формализуем данную задачу. Формализация задачи происходит за счет получения и обработки лингвистических переменных. Под лингвистической переменной понимается единица входных данных (слово), полученная из формулировки задачи. Множество лингвистических переменных обозначим через $L = \{ l_1, l_2, \ldots, l_s \}$, где s>0.

Этот этап построения системы основан на обработке множества L методами кластерного анализа. А именно, данное множество подвергнем кластеризации, в ходе которого определим множество дескрипторов, по которым будет оцениваться объекты в выборке. При этом количество объектов, полученных в результате преобразования, не должно превышать количества объектов во множестве L. Обозначим это множество через $K = \{ k_1, k_2, \ldots, k_m \}$, где 0 < m ≤ s. Таким образом, получаем сюръективное отображение α множества L во множество K:

$$\alpha: L \to K . \quad (2)$$

Полученное множество K будем разделять на кластеры. В качестве определяющих групп будем использовать номера построенных заранее множества объектов вида (1), которые будут храниться в базе. Обозначим это множество через N = {1, 2, ..., c}, где c>0. (более подробно кластеризация будет описана на втором этапе реализации системы). Таким образом, получим следующее инъективное отображение β:

$$\beta: K \to N \quad (3)$$

Очевидно, что некоторый объект $k_i \in K$ может отобразится как в a_i, так и в a_j, при этом i ≠ j. Поэтому мы получаем некоторое множество векторов, размерности m. А именно:

$$\{ (k_1, n_i), (k_2, n_j), \ldots (k_m, n_z) \mid 0 < m \le s, k_i \in K, a_j \in A ; i, j, z \in [1, c], c > 0 \} . \quad (4)$$

Итак, на первом этапе наша система имеет следующее представление (рис. 1)

Рассмотрим второй этап построения нашей системы – этап машинного обучения. На этом этапе происходит накопление и обработка данных, а также принятие решения относительно алгебраической системы. Основной метод преобразований данных осуществляется с помощью нейронной сети, которая непосредственно занимается кластеризацией наших данных.

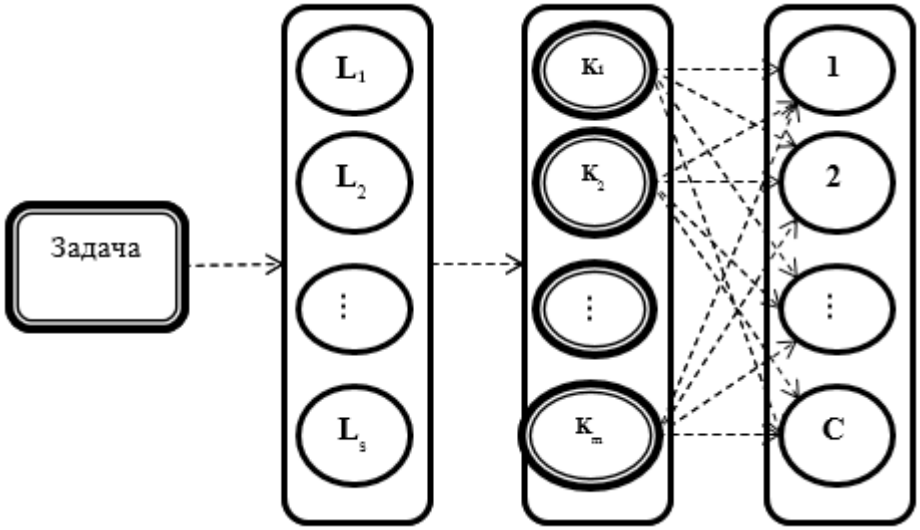


Рис. 1 Лингвистический этап системы

Кластерный анализ является частью обучающейся многослойной нейронной сети, которая способна, на основе входных данных, подаваемых с первого этапа, принимать правильное решение относительно её алгебры. А именно она рассматривает принадлежность элементов множества К к различным определяющим группам. Эти группы являются составляющими элементами двухосновной универсальной алгебры, вида (1). Каждому представителю такой алгебры принадлежат свои группы, по которым и происходит кластеризация данных. При этом элементы множества К могут принадлежать сразу нескольким кластерам в определённом представителе алгебры. На основе полученных вхождений элементов множества К в определённый кластер, формируется некоторое действительное число (вес синапса), которое показывает степень принадлежность данной задачи к определённой универсальной системе.

Итак, учитывая, что нам необходимо на выходе получить конкретную алгебраическую систему вида (1), то наша сеть будет выглядеть следующим образом (рис. 2).

Данные, полученные с первого этапа, отправляются сразу на 2 слоя: множество структур S и множеств различных мощностей M. На каждом из них данные обрабатываются с помощью алгоритма кластеризации, рассмотрим более подробно кластеризацию на втором слое.

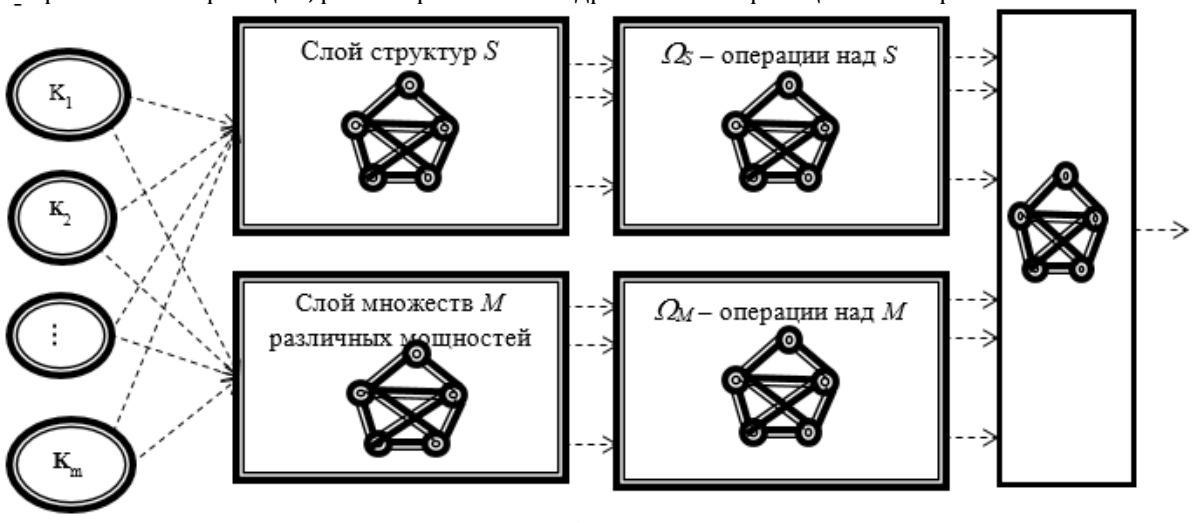


Рис. 2 Нейронная сеть

При поступлении данных на слой, они обрабатываются с помощью нейронов, которые сначала объединяют эти данные в нечёткие кластеры, а затем находят степень принадлежности входных элементов к определённому элементу множества M, например степень принадлежности данного кластера к множеству действительных чисел.

В результате формируется множество упорядоченных пар или кортежей, вида:

$<x, \mu_M(x)>$, где x – это кластер, полученный на этом этапе, а $\mu_M(x)$ - функция принадлежности, которая ставит в соответствие каждому элементов k соответствующий элемент из X, при этом $\mu_M(x)$ выражается действительным число из интервала [0,1], которое определяет степень соответствия элемента множества К конкретному элементу из X. Данная функция определяется отображением $\mu_M(x):(K \to X)$.

В качестве элементов множества X используем множества различных мощностей. Само же значение $\mu_M(x)$ будем определять с помощью частоты вхождения элементов в данную группу. Если для $x \in A$, $\mu_A(x)=1$, то это означает, что элемент x определённо принадлежит нечёткому множеству A, а значение $\mu_A(x)=0$ означает, что элемент x определённо не принадлежит ему.

Полученные результаты передаются на следующие соответствующие слои: слой операций над структурами и слой операций над множествами.

Рассмотрим слой операций над множествами. На нем обрабатываем данные с предыдущего слоя тем же способом, но только кластеры другие и поэтому множество, полученное в результате обработки, будет иметь другие ещё один весовой коэффициент, и полученный кортеж с предыдущего слоя перейдёт уже в четверку элементов, вида $<x, \mu_M(x), O, \mu_X(O)>$, где *x* – это кластер, полученный на предыдущем этапе, $\mu_M(x)$ - функция принадлежности для *X*, O – множество операций над X, $\mu_X(O)$ – степень принадлежности операции к данному кортежу.

Полученные результаты, со своими весовыми коэффициентами поступают на ещё один слой, который формулирует окончательный результат – это слой Кохонена. На этом слое выходные сигналы с предшествующих слоёв обрабатываются сначала вместе, образуя конечные кластеры со своими новыми коэффициентами принадлежности, а затем эти группы обрабатываются по правилу «победитель забирает всё», т.е. наибольший сигнал превращается в единичный, а остальные обращаются в ноль. Схематически это можно представить в виде следующей структуры (рис. 3).

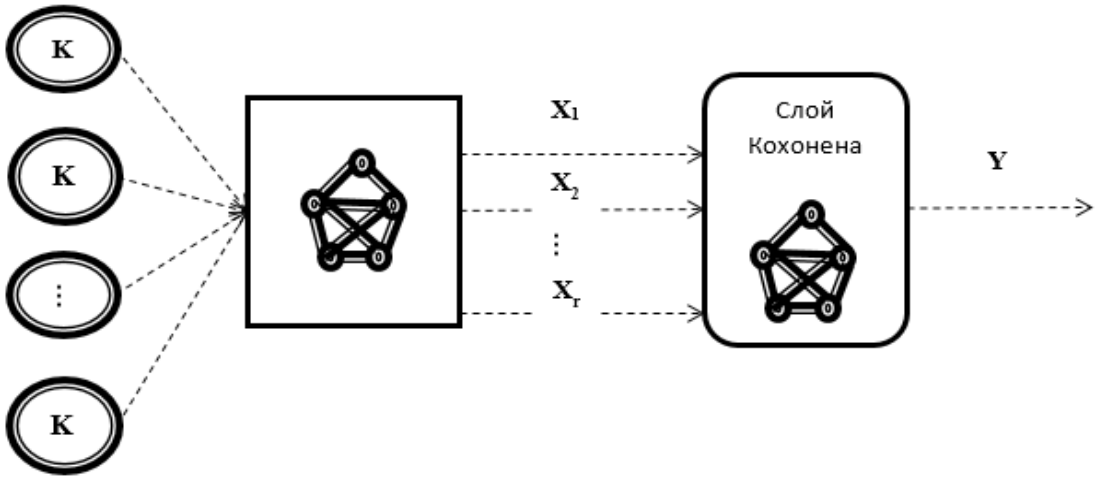


Рис. 3 Нейронная сеть со слоем Кохонена

Таким образом происходит обработка поступающих данных, которая позволяет в конечном итоге получить алгебраическую систему вида (1). Рассмотрим коротко состояние самой нейронной сети да того, как она будет обучена.

В первоначальном состоянии наша нейронная сеть представляет систему соединённых и взаимодействующих между собой простых процессоров. На её вход поступает множество К, в базе данных хранится несколько записей вида (1) – это в простейшем случае, либо в более сложном представлении она будет комбинировать выходной сигнал вида (1) из нескольких баз данных (в-первой базе находится множество структур, во-второй – множества различных мощностей, в-третьей - операции над структурами, в-четвёртой – операции над множествами).

Для накопления знаний нашей системы воспользуемся рядом обучающихся примеров, то есть строим такое отображение $X \to Y$, чтобы в ответ на каждый входной сигнал X формировался правильный выходной сигнал Y, т.е. формируем более полную базу знаний. При этом база данных, предположительно, пополняется за счёт поступления новых задач, которые не

похожи на предыдущие задачи, ранее использовавшиеся в качестве обучающих примеров.

Обучить нейросеть – значит сообщить ей, чего мы от неё добиваемся. Этот процесс очень похож на обучение ребёнка алфавиту. В нашем случае будем строить процесс «обучения с учителем» (рис. 4).

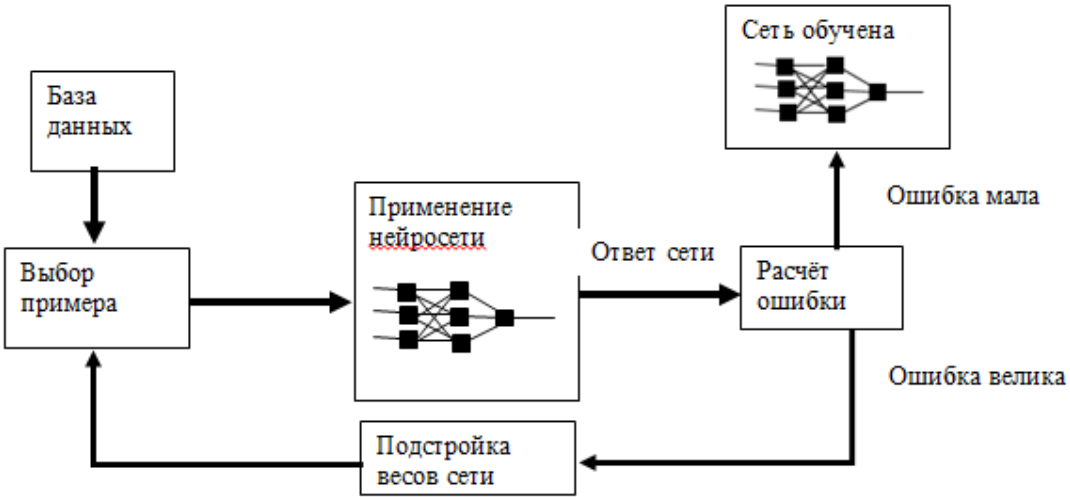


Рис. 4 Обучение нейронной сети

У нас имеется некоторая база данных, содержащая примеры. Предъявляя формулировку множества A на вход сети, мы получаем от неё некоторый ответ, не обязательно верный. Но нам известен и верный (желаемый) ответ – в данном случае нам хотелось бы, чтобы он вывел конкретную алгебру, которая бы удовлетворяла данным условиям, при этом уровень сигнала должен быть максимальным. Обычно в качестве желаемого выхода в задаче классификации берут набор (1,0,0,….), где 1 – стоит на метке с правильным ответом, а на всех остальных – ноль. Вычисляя разность между желаемым ответом и реальным ответом сети, мы получаем вектор ошибки. Учитывая, что для обучения необходимо закрепить нужные знания, один и тот же пример мы можем предъявлять сети несколько раз. В этом смысле обучение скорее напоминает повторение упражнений в спорте – тренировку.

После многократного предъявления примеров веса сети стабилизируются, причем сеть дает правильный ответы на все (или почти все) примеры из базы данных. В таком случае говорят, что «сеть обучена» или «сеть натренирована». В программной реализации можно видеть, что в процессе обучения функция ошибки постепенно уменьшается. Когда функция ошибки достигает нуля или приемлемого малого уровня, тренировку останавливают, а полученную сеть считают натренированной и готовой к применению на новых данных.

Вся информация, которую сеть имеет о задаче, содержится в наборе примеров. Поэтому качество сети напрямую зависит от количества примеров в обучающейся выборке, а также от того, насколько полно эти примеры описывают данную задачу. Считается, что для полноценной тренировки требуется хотя бы несколько десятков примеров.

Математически процесс обучения можно описать следующим образом.

В процессе функционирования нейронная сеть формирует выходной сигнал Y в соответствии с входным сигналом X, реализуя некоторую функцию $Y=G(X)$ Если архитектура сети задана, то вид функции G определяется значениям, синоптических весов и смещений сети.

Пусть решением некоторой задачи, является функция $Y=F(X)$, заданная парами входных-выходных данных $(X^1,Y^1),(X^2,Y^2),\ldots,(X^n,Y^n)$, для которых $Y^k=F(X^k)$ (k=1,2,…,n).

Обучение состоит в поиске (синтезе) функции G, близкой к F в смысле некоторой функции ошибки.

Если выбрать множество обучающих примеров – пар (X^k,Y^k), где (k=1,2,…,n) и способ вычисления функции ошибки, то обучение нейронной сети превращается в задачу многомерной оптимизации, имеющую очень большую размерность, при этом, поскольку функция ошибки может иметь произвольный вид, обучение в общем смысле – многоэкстремальная невыпуклая задача оптимизации.

Необученная сеть подобна ребенку — ее можно научить чему угодно. В используемых на практике нейросетях количество весов может составлять несколько десятков тысяч, поэтому обучение — действительно сложный процесс. Для многих архитектур разработаны специальные алгоритмы обучения, которые позволяют настроить веса сети определенным образом.

После того как сеть обучена, мы можем применять её для решения различных задач.

Важнейшая особенность человеческого мозга состоит в том, что, однажды обучившись определенному процессу, он может верно действовать и в тех ситуациях, в которых он не бывал в процессе обучения. Например, мы можем читать почти любой почерк, даже если видим его первый раз в жизни. Так же и нейросеть, грамотным образом обученная, может с большой вероятностью правильно реагировать на новые, не предъявленные ей ранее данные. Веса обученной сети хранят достаточно много информации о сходстве и различиях множества А, поэтому можно рассчитывать на правильное опознание и нового варианта изображения.

Предполагается, что такая система будет построена с использованием методов искусственного интеллекта, которые доступны в системе Microsoft Mashing Learning Azure. А именно, Mashing Learning дает возможность осуществить кластеризацию данных с использованием различных алгоритмов кластерного анализа, что необходимо при обработке семантики задачи.

Итак, для построения системы, способной формализовать семантику задачи в качестве алгебры, используется в основном два метода: метод кластерной обработки данных и метод построения многослойной нейронной сети, которая способна самостоятельно формировать свою бузу знаний на основе обучающих примеров, а также принимать решение о том, к какой алгебраической системе можно отнести данную задачу.

Литература

1. Мунерман В.И., Симакова А. А. Преобразование семантики задачи в алгебру –2015.
2. Люгер Дж.Ф. Искусственный интеллект: стратегии и методы решения сложных проблем, 4-е издание. : Пер. с англ. — М ,: Издательский дом “Вильямс”, 2003.
3. Дьяконов В. П., Круглов В. В. Mathlab 6.5 SP1/7/7 SP1/7 SP2 +Simulink 5/6. Инструменты искусственного интеллекта и биоинформатики. – М. СОЛОН-ПРЕСС,2006.
4. Павлов С.Н. Системы искусственного интеллекта – Томск: Эль Контент, 2011.—Ч. 1.
5. Емельченков Е.П., Левин Н.А., Мунерман В.И. Математические модели для проектирования информационных систем.
6. Мальцев А. И. Алгебраические системы – М., «Наука» , 1970.

Орлова Н.В.

Ростовский государственный экономический университет (РИНХ), г. Ростов-на-Дону, к.э.н., доцент кафедры Информационных систем и прикладной информатики, orlova65@mail.ru

ОБ ИНТЕГРАЦИИ ТЕХНОЛОГИЙ ДОПОЛНЕННОЙ РЕАЛЬНОСТИ, 3D-ТУРОВ И 3D-МОДЕЛИРОВАНИЯ ДЛЯ ОБУЧЕНИЯ IT-СПЕЦИАЛИСТОВ

КЛЮЧЕВЫЕ СЛОВА

Дополненная реальность, 3D модель, 3D тур, 3D визуализация, IT-образование, 3D проект.

АННОТАЦИЯ

В статье автор делится опытом обучения IT-специалистов технологиям интеграции в приложения дополненной реальности виртуальных путешествий и 3D моделей. Приведены реальные примеры участия обучающихся в проекте 3D-университет. Описана технология реализации 3D-туров различными по сложности освоения и трудоёмкости методами.

В Ростовском государственном экономическом университете РГЭУ (РИНХ) уже не первый год идут исследования в области новейших технологий обработки изображений, распознавания образов, 3D-моделирования и др. Группой студентов факультета Компьютерных технологий и информационной безопасности под руководством автора было разработано приложение дополненной реальности, включившее в себя технологии разработки 3D-туров и презентации логотипов факультетов и университета в виде 3D-моделей.

Дополненная реальность (AR) — термин, относящийся ко всем проектам, направленным на дополнение реальности любыми виртуальными элементами. Самые распространённые примеры дополненной реальности — параллельная лицевой цветная линия, показывающая нахождение ближайшего полевого игрока к воротам при телетрансляции футбольных матчей, стрелки с указанием расстояния от места штрафного удара до ворот, нарисованная траектория шайбы во время хоккея и т. п. Сам термин был предложен работавшим на корпорацию Boeing исследователем Томом Коделом в 1990 году.

В 1994 году Пол Милгром (англ. Paul Milgram) и Фумио Кисино (англ. Fumio Kishino) описали континуум «виртуальность — реальность» (англ. Milgram's Reality-Virtuality Continuum) — пространство между реальностью и виртуальностью, между которыми расположены дополненная реальность (ближе к реальности) и дополненная виртуальность (ближе к виртуальности). Дополненная реальность — результат добавления к воспринимаемым как элементы реального мира мнимых объектов (обычно в качестве вспомогательной информации).

Существует множество программных решений для мобильных телефонов, которые позволяют при помощи дополненной реальности получить необходимые сведения об окружении. Это браузеры дополненной реальности и специализированные программы для отдельных сервисов, компаний или даже единственных моделей. Само распространение дополненной реальности и нарастающая известность технологии среди потребителей связано с тем, что вычислительная мощность и набор датчиков в аппаратных платформах для смартфонов и планшетов позволяют производить наложение любых цифровых данных на получаемое в реальном времени со встроенных в устройства камер изображение. Часть решений в этой области воплощается в виде нательных компьютеров (в том числе в качестве элементов умной одежды) для постоянного контакта со средой дополненной реальности.

Приложения дополненной реальности постепенно выходят за пределы исключительно развлекательной сферы, становясь действительно полезными. Благодаря им теперь можно чинить автомобили, переводить тексты в реальном времени, искать ближайшие рестораны и делать покупки в виртуальных супермаркетах. Редкие разработчики пишут софт для таких приложений с нуля, большинство использует готовые фреймворки или библиотеки с открытым API. Компании, создающие браузеры дополненной реальности, позволяют другим разработчикам создавать собственные AR-проекты на базе своей платформы и, по сути, берут на себя функцию распространителей их контента.

Это такие компании как:

1. Layar - компания из Нидерландов, создавшая мобильный AR-браузер в 2009 году.
2. Metaio - немецкая компания-мастодонт, работающая в области дополненной реальности с 2003 года.
3. Aurasma - молодая английская компания, использующая разработанную в Кэмбридже технологию. Услугами Aurasma в основном пользуются крупные бренды;
4. Wikitude - австрийская компания, победитель номинации «Лучший браузер дополненной реальности» на протяжении последних четырёх лет.
5. Blippar - молодая английская компания, которая использует технологию распознавания изображения для небольших рекламных кампаний и уже успела поработать со многими известными брендами, включая Unilever, Nestle, Heinz, Xbox, Samsung, Cadbury, Domino.

В самом простом варианте, для того чтобы создать приложение с дополненной реальностью нужно просто взять и наложить виртуальное изображение на видео с камеры телефона.

Полноценная дополненная реальность включает в себя не только отрисовку виртуальных объектов поверх изображения с камеры, а привязывание их к окружающей обстановке. Для этого используются либо метки расположенные в реальном мире, и к которым привязывается виртуальный объект, либо якоря (GPS-координаты). Ведь куда приятнее видеть не только одну сторону виртуального предмета расположенного у вас на столе, но и иметь возможность обойти его со всех сторон, подойти поближе и рассмотреть его в деталях. Помимо меток к которым привязывается объект, также необходимо учитывать положение устройства в пространстве считывая и обрабатывая показания компаса, акселерометра и гироскопа, для того чтобы правильно отображать виртуальный объект.

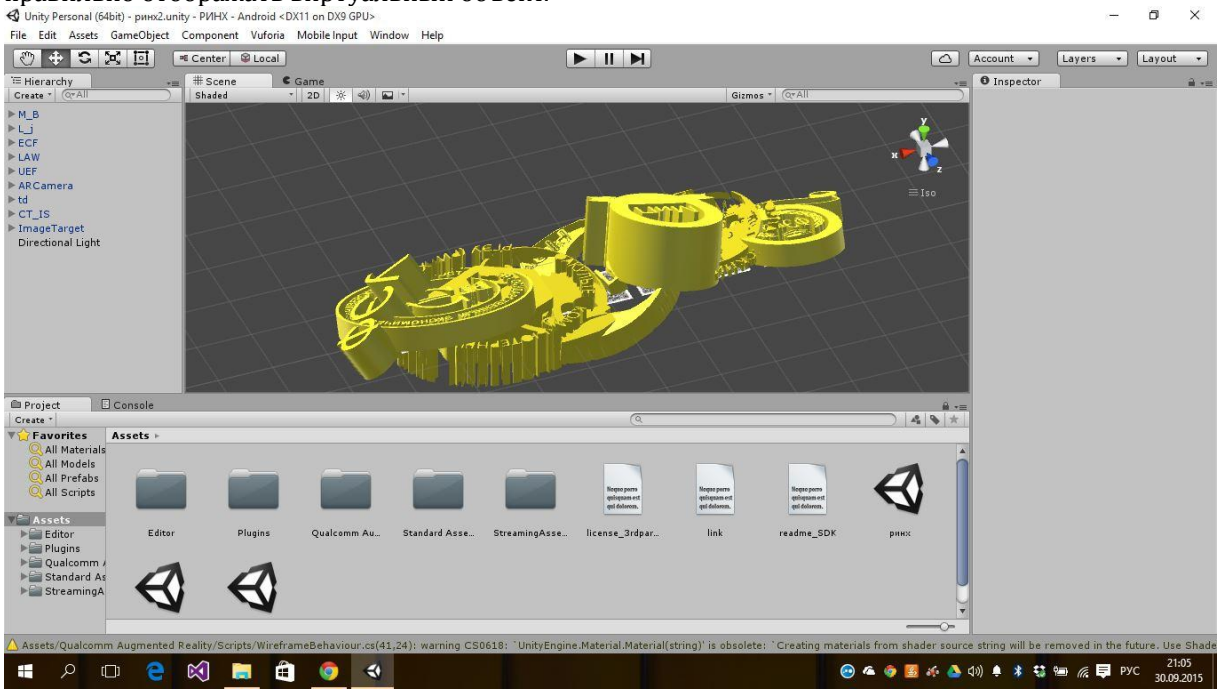


Рисунок 1 – Связывание 3D-моделей и QR-метки в Unity

Проект предполагает использование 3 основных составляющих: QR-меток с зашифрованными данными, смартфона или планшета на основе операционной системы Android (Версия 4.4 Kit Kat или выше) и специального программного обеспечения, разработанного для дешифровки и трансляции закодированной информации в окружающей среде. Использование технологий таких компаний как Unity Technologies (Unity Engine 5/Unity Editor), Qualcomm (Vuforia Software Development Kit) и Google (Android Software Development Kit) позволило существенно сократить время разработки и тестирования продукта, не влияя на качество, производительность и функционал программного обеспечения. Также стоит отметить, что приложение не требовательно к аппаратной части устройства. Все необходимые компоненты, такие как гироскоп,

акселерометр, камера и процессор, построенный с применением архитектуры ARMv6 (или выше) с возможностью проведения вычислений с плавающей запятой, являются одними из базовых элементов любого современного смартфона или планшета.

3D графика или трёхмерная графика – это один из разделов компьютерной графики, комплекс приемов и инструментов, которые позволяют создать объемные объекты при помощи форма и цвета. От двухмерных изображений она отличается тем, что подразумевает построение геометрической проекции трехмерной модели сцены (виртуального пространства) на плоскость, делается это при помощи специализированных программ. Полученная модель может соответствовать объектам реального мира.

Для получения трёхмерного изображения на плоскости требуются следующие шаги:

- моделирование — создание трёхмерной математической модели сцены и объектов в ней;
- текстурирование — назначение поверхностям моделей растровых или процедурных текстур (подразумевает также настройку свойств материалов — прозрачность, отражения, шероховатость и пр.);
- освещение — установка и настройка источников света;
- анимация (в некоторых случаях) — придание движения объектам;
- динамическая симуляция (в некоторых случаях) — автоматический расчёт взаимодействия частиц, твёрдых/мягких тел и пр. с моделируемыми силами гравитации, ветра, выталкивания и др., а также друг с другом;
- рендеринг (визуализация) — построение проекции в соответствии с выбранной физической моделью;
- композитинг (компоновка) — доработка изображения;
- вывод полученного изображения на устройство вывода — дисплей или принтер.

При создании 3D-моделей были использованы программы CorelDRAW и 3ds Max.

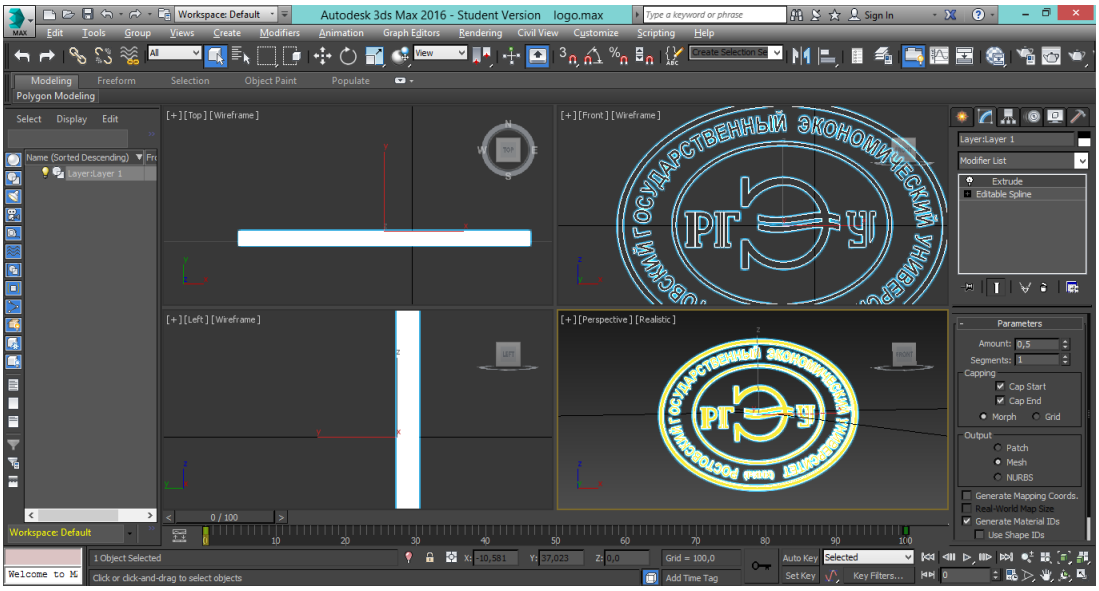


Рисунок 2 – Создание 3D-модели

Импортировав нужное изображение в среду CorelDRAW и обработав ее, было получено векторное изображение. Затем оно было импортировано в среду 3Ds Max. Настроив математическую модель сцены и обработав импотированное изображение (настройка света, настройка объема проекта, настройка анимации и т.д.), было получено готовое 3D-изображение.

Наше приложение включает разработку 3d тура для этого проекта. Для начала надо рассмотреть, что же такое 3d тур, а начнем мы с 3d панорамы.

Рисунок 3 – Результат работы приложения

И так что же такое 3d панорама- это сравнительно молодая современная технология панорамной сферической фотографии, которая открывает возможности стать действующим лицом как будто вы находитесь на месте съемки и свободны выбирать, куда повернуться, что рассмотреть к чему подойти по ближе. На сегодняшний день 3d панорама — это фотореалистичный способ презентовать любые существующие объекты посредством внедрения в трехмерное пространство. И так что же приставляет собой 3d панорама — это снимок окружающего пространства на 360 градусов по горизонтали и на 180 по вертикали.

Такая технология позволяет добиться эффекта присутствия в месте съемки, так как кроме изменения направления взгляда также можем приближать или удалять от любых мест 3d панорамы, так же есть возможности объединения нескольких панорам в виртуальный тур.

3D тур объединяет несколько 3D-панорам соединённых между собой точками перехода.

Так же они могут включать в себя Flash анимацию, звуковые эффекты и шумовое сопровождение.

В настоящее время тема 3D туров оживает, по интернету все чаще можно встретить 3D тур по клубу, сауне и загородному дому.

И так теперь можно перейти к самому проекту.

Для создания университетского 3D тура нам потребовалось панорамные фотографии университетских помещений, а также программа, в которой можно работать с данными панорамами. Одна из таких программ Kolor Panotour Pro – это для создания интерактивных виртуальных туров из цифровых фотографий, которая освободит Вас от задач программирования. Все действия осуществляются самой программой.

В проект можно добавлять изображения, разрешение которых не превышает 3600x1800 пикселей. Поддерживаются все основные графические форматы (JPG, PSD, PNG, BMP, RAW). Интерфейс в Panotour возможность настраивать так, как удобно пользователю. Отдельные инструменты можно переместить, убрать или даже выделить для них отдельное окно. Связи между изображениями создавать очень просто.

Основные возможности программы Panotour:

- Создание виртуальных туров занимает гораздо меньше времени.
- Процесс создания тура происходит визуально, никакого программирования.
- Создание связей в пару нажатий.
- Поддержка форматов: JPG, PNG, PSD, TIFF и RAW файлов с большинства камер.
- Настраиваемый интерфейс.
- Простое взаимодействие с использованием горячих точек.
- Поддержка 3D-эффектов.
- Интуитивно-понятный интерфейс.

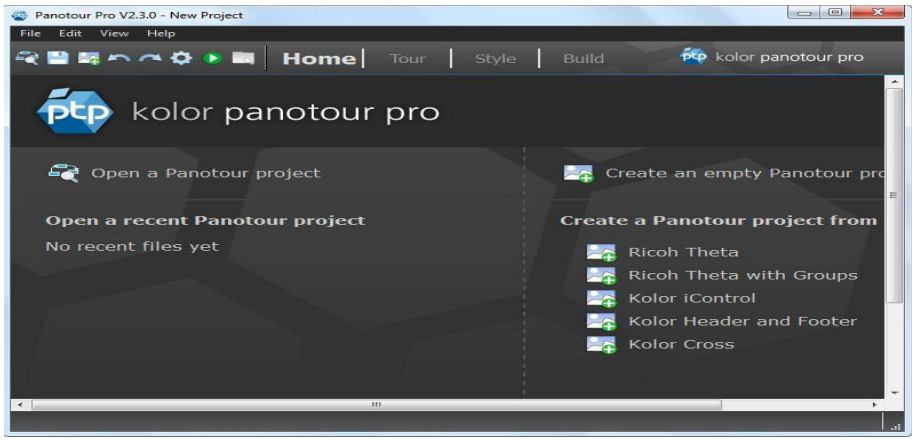


Рисунок 4 - Меню программы

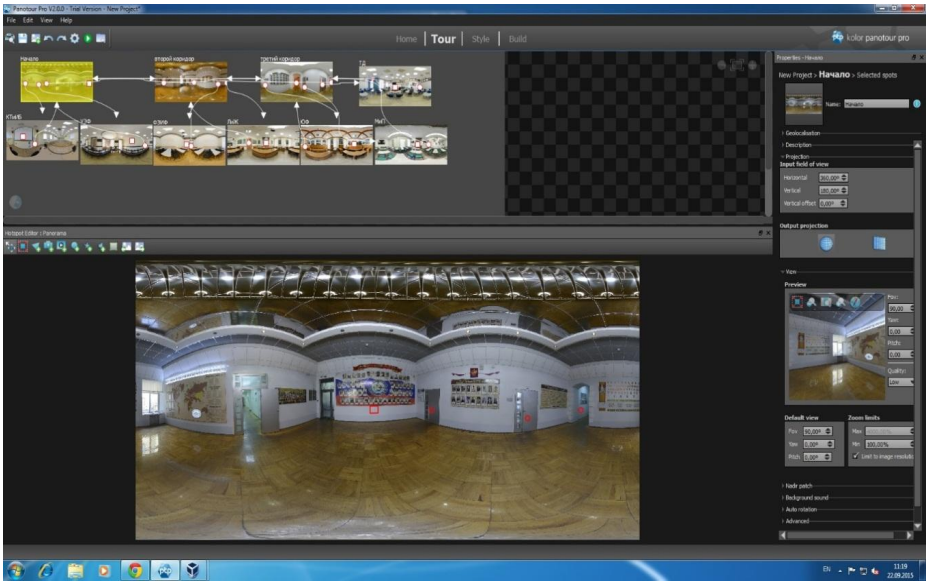


Рисунок 5 – Проект университет

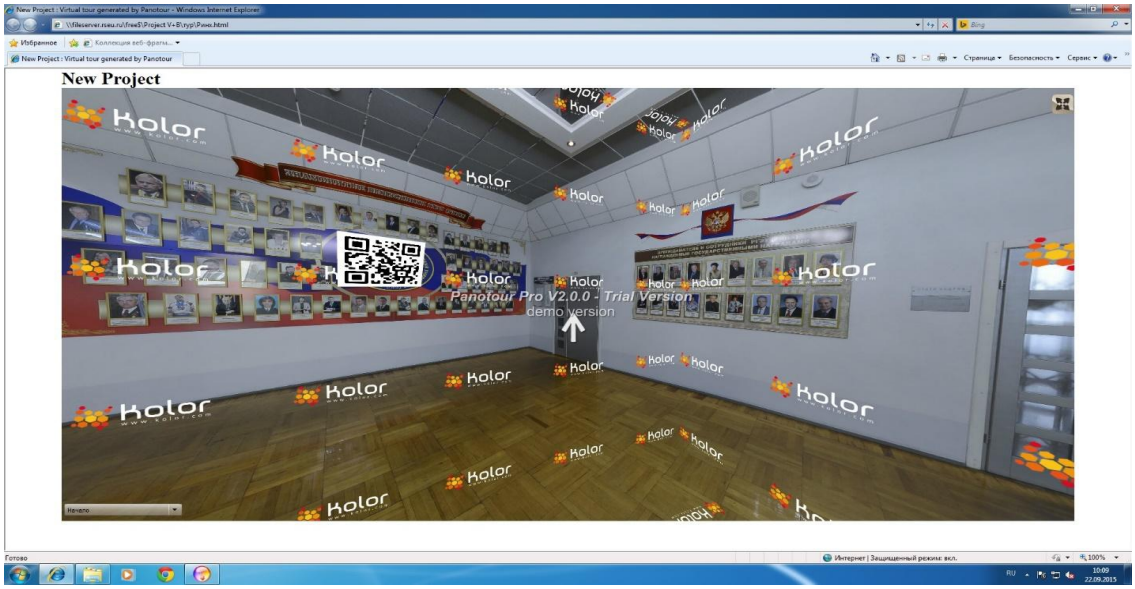


Рисунок 6 – Полностью готовый 3D тур

На рисунке 6 показан готовый проект по 3D туру и так в этом проекте находятся 10 панорамных изображений, соединенных между собой связями перехода и выстроены по

очередности. Так же на данном рисунке можно увидеть стрелки, которые обозначаю каком панорамном изображении можно перейти в другое.

Итак, данное изображение показывает нам как выглядит готовый 3D тур, в котором можно увидеть белую стрелку на двери это обозначает переход в другу зону, то есть на новое панорамное изображение. На этом изображении расположена QR метка на которой нанесена информация об университете, а также 3D модель эмблемы РГЭУ(РИНХ).

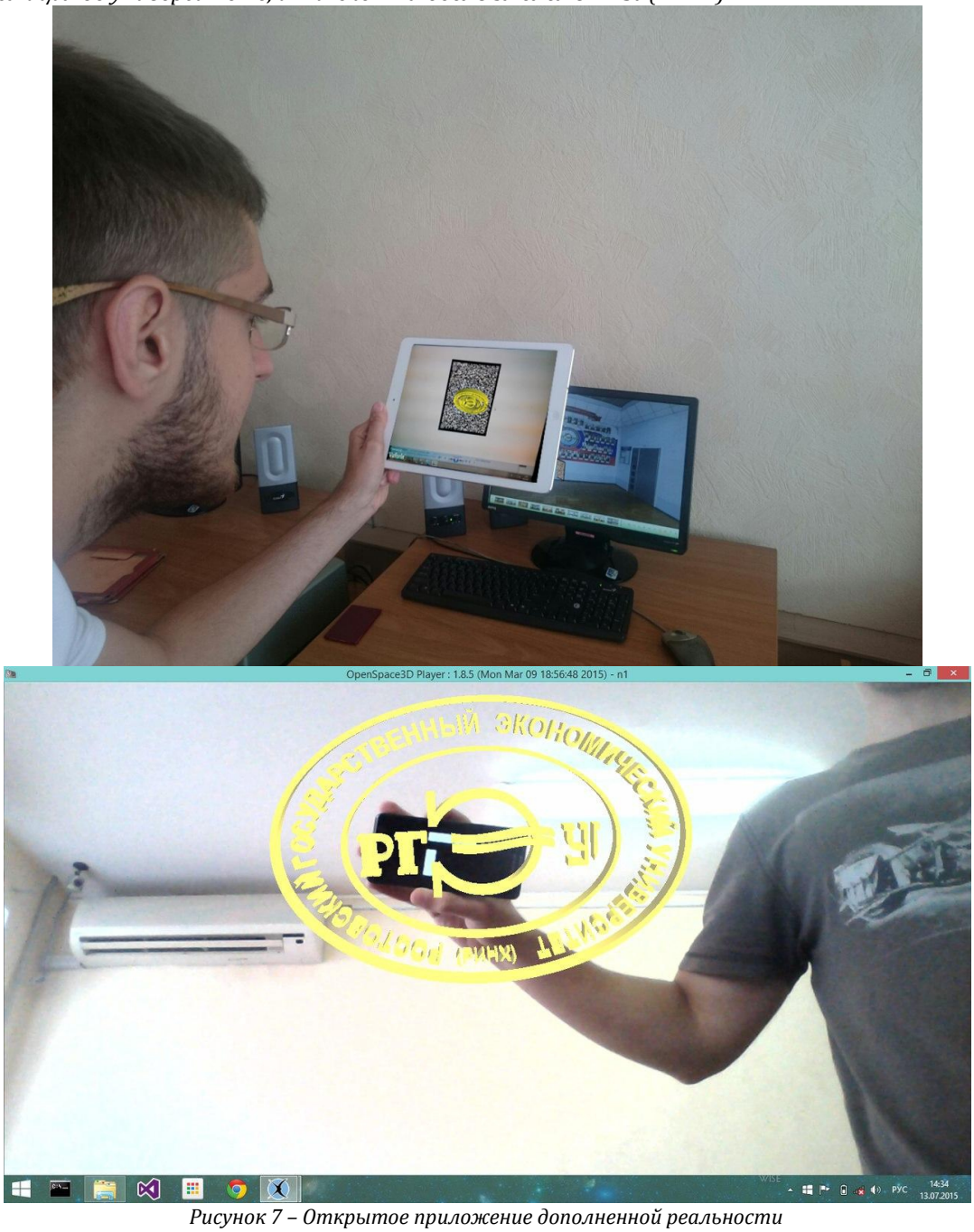


Рисунок 7 – Открытое приложение дополненной реальности

Участие студентов факультета Компьютерных технологий и информационной безопасности в подобных проектах формирует важные компетенции и даёт неоценимый опыт разработки уникальных приложений. Следует отметить и богатый научный потенциал этих разработок. Интеграция таких сложных технологий, как 3D-моделирование, 3D-туры и дополненная реальность в одной разработке на сегодняшний день является весьма актуальной и мало заполненной нишей рынка презентационных продуктов.

Литература

1. Орлова Н.В. Применение методов 3d-моделирования в экономических информационных системах: Статистика в современном мире: методы, модели, инструменты: материалы Всероссийской научно-практической конференции. (26-28 сентября 2012 г.) / РГЭУ (РИНХ). – Ростов-на-Дону, 2012. –0,2 п.л.
2. Орлова Н.В. Применение методов 3D-моделирования в IT-образовании// Современные информационные технологии и ИТ-образование [Электронный ресурс] / Сборник научных трудов VIII Международной научно-практической конференции / под ред. В.А. Сухомлина. – Москва: МГУ, 2013. – Т.1. – 478с. – с.45-=454 - 1 электрон. опт. диск (CD-ROM). – ISBN 978-5-9556-0156-4.
3. Н.В. Орлова Интеграция технологий дополненной реальности, 3D-туров и 3D-моделирования для разработки приложений // Информационные системы, экономика, управление трудом и производством: Ученые записки. Выпуск 16 / Ростовский государственный экономический университет «РИНХ».- Ростов-на-Дону, 2015. - с. 99-101 – 0,1 п.л. (в соавт.)

Туракулова А.И.

Ташкентский государственный педагогический университет им. Низами, Республика Узбекистан, старший преподаватель кафедры «Методика преподавания информатики», e-mail: k.informatiki@yandex.ru

ВОЗМОЖНОСТИ НОВОГО ОБЪЕКТНО-ОРИЕНТИРОВАННОГО ЯЗЫКА ПРОГРАММИРОВАНИЯ OBJECTSCRIPT

КЛЮЧЕВЫЕ СЛОВА

Программирование, объектно-ориентированные языки программирования, ObjectScript, создание скриптов, веб-приложения, OS-FCGI.

АННОТАЦИЯ

В статье проанализированы преимущества объектно-ориентированных языков программирования, раскрыты возможности нового объектно-ориентированного языка программирования ObjectScript. Сделан обзор основных задач, для решения которых спроектирован этот язык.

Развитие новых технологий — важное звено в установлении гармонических отношений между обществом и производством на основе предельно экономного использования всех видов природных и социальных ресурсов. Важнейшей из таких технологий, оказывающей определяющее влияние на все другие, является информационная технология, включающая в себя проектирование и создание современных компьютеров, разработку всех видов программ и специальных языков программирования, обеспечивающих решение самых сложных задач, связанных с хранением, переработкой, поиском и распространением информации. В силу этого информационная технология становится катализатором нового технологического этапа научно-технического прогресса.

По мере развития технологии программирования все адекватнее отражалась структура решаемой прикладной задачи и осуществлялось объединение данных и процедур для их обработки. Это и привело к появлению в программировании объектного подхода. Сейчас уже совершенно ясно, что на смену традиционному программированию, пришло объектно-ориентированное программирование(ООП).

Объектно-ориентированные языки программирования пользуются в последнее время большой популярностью среди программистов. Первый объектно-ориентированный язык программирования был разработан в конце 60-х годов в Норвегии. Авторы этого языка очень точно угадали перспективы развития программирования: их язык намного опередил свое время. В 70-е годы было разработано большое число экспериментальных объектно-ориентированных языков программирования. Эти языки так и остались экспериментальными, но в результате их исследования были разработаны современные объектно-ориентированные языки программирования. Разработка новых объектно-ориентированных языков программирования продолжается.

Успех ООП объясняется предоставляемой им возможностью естественно имитировать объекты нашего мира. Решая сложные проблемы, выдвигаемые жизнью, человек постепенно развил в себе замечательные способности к обобщению, классификации и созданию абстракций. В окружающих нас объектах мы находим общие признаки, свойства, способы поведения.

Любая программа, написанная на языке ООП, отражает в своих данных состояние физических предметов либо абстрактных понятий – объектов программирования, для работы, с которыми она предназначена.

Все данные об объекте программирования и его связях с другими объектами можно объединить в одну структурированную переменную. В первом приближении ее можно назвать объектом.

Объекты, описанные одним и тем же набором параметров и способные выполнять один и тот же набор действий представляют собой класс однотипных объектов.

С точки зрения языка программирования класс объектов можно рассматривать как тип данного, а отдельный объект - как данное этого типа. Определение программистом собственных

классов объектов для конкретного набора задач должно позволить описывать отдельные задачи в терминах самого класса задач (при соответствующем выборе имен типов и имен объектов, их параметров и выполняемых действий) [2].

В современных ОО языках используются механизмы [1]:

- наследование: создание нового класса объектов путём добавления новых элементов (методов). Некоторые ОО языки позволяют выполнять множественное наследование, то есть объединять в одном классе возможности нескольких других классов;
- инкапсуляция: сокрытие деталей реализации, которое позволяет вносить изменения в части программы безболезненно для других её частей, что существенно упрощает сопровождение и модификацию ПО;
- полиморфизм: при полиморфизме некоторые части (методы) родительского класса заменяются новыми, реализующими специфические для данного потомка действия. Таким образом, интерфейс классов остаётся прежним, а реализация методов с одинаковым названием и набором параметров различается.

Одним из новых объектно-ориентированных языков программирования является ObjectScript — новый встраиваемый и очень легкий объектно-ориентированный язык программирования с открытым исходным кодом. ObjectScript расширяет возможности таких языков, как JavaScript, Lua, Ruby и PHP.

ObjectScript — это универсальный язык программирования, ObjectScript предназначен для встраивания в программы на C++, разработки сайтов, написания скриптов и др.. ObjectScript - кроссплатформенный, бесплатный язык программирования, который спроектирован для решения следующих основных задач [3]:

- веб-программирование, создание сайтов и скриптов для выполнения на стороне сервера через веб-браузер. Для того чтобы можно было просматривать результаты выполнения OS-скриптов в браузере, нужен работающий веб-сервер и установленный ObjectScript для веба (например, os-fcgi). Просмотреть вывод OS-программы можно в браузере, получив OS-страницу, сгенерированную сервером. В случае, если вы просто экспериментируете, то вполне можете использовать свой домашний компьютер вместо сервера;
- создание скриптов для выполнения из командной строки. Вы можете создать OS-скрипт, способный запускаться без сервера или браузера. Все, что вам потребуется – консольный OS для запуска скриптов из командной строки. Такой способ использования идеально подходит для скриптов, которые должны выполняться регулярно, например, с помощью cron (на платформах *nix или Linux) или с помощью планировщика задач (Task Scheduler) на платформах Windows. Эти скрипты также могут быть использованы в задачах обработки текстов и мн. др.;
- встраивание ObjectScript в программы на C++. ObjectScript изначально спроектирован, как встраиваемый язык, поэтому сделать это довольно просто, например:

```
#include <objectscript.h>
using namespace ObjectScript;
int main(int argc, char* argv[])
{
OS * os = OS::create(); // craete ObjectScript instance
os->require("main.os"); // run ObjectScript program
os->release();        // release the ObjectScript instance
return 0;
}
```

Программа os-fcgi (с помощью которой работает этот сайт) является одним из примеров использования ObjectScript на C++ и распространяется абсолютно бесплатно в открытом исходном коде.

Программа os-fcgi является одним из примеров использования ObjectScript на C++ и распространяется абсолютно бесплатно в открытом исходном коде.

Вы можете быстро начать программировать на ObjectScript под веб, используя программу os-fcgi. Она работает через клиент-серверный интерфейс FastCGI, который поддерживается большинством современных веб-серверов, например, apache, nginx и мн. др. При этом ObjectScript является кроссплатформенным решением, поэтому вести разрабатку можно на Linux, Windows и др. платформах.

Таким образом, выбирая OS, вы получаете свободу выбора операционной системы и веб-сервера. Более того, у вас появляется выбор между использованием процедурного или объектно-ориентированного программирования (ООП) или же их сочетания.

Кроме этого в базовый состав ObjectScript входят множество расширений, например:

- Regexp (работа с регулярными выражениями на базе библиотека PCRE);
- DateTime (работа с датой и временем);
- Curl (запрос данных по http, ftp и др. протоколам, в том числе генерация запросов методом POST);
- ODBO (работа с базами данных);
- iconv (ковертация данных из одних кодировок в другие);
- zlib (архивирование и разархивирование данных);
- json (работа с данными в json формате);
- math (математические функции) и др.

ObjectScript сконструирован специально для ведения веб-разработок, поэтому его код может внедряться непосредственно в HTML, например:

```
<!DOCTYPE html>
<html>
 <head>
  <title>Тестируем ObjectScript</title>
 </head>
 <body>
 <% echo '<p>Привет мир!</p>' %>
 </body>
</html>
```

Вместо рутинного вывода HTML-кода командами языка (как это происходит, например, в Perl или C), скрипт ObjectScript содержит HTML с вставками кода на OS (в нашем случае, это вывод текста" Привет мир! !"). Код ObjectScript отделяется специальными начальным и конечным тегами<% и %>, которые позволяют переключаться в OS-режим и выходить из него.

OS-скрипты выполняются на сервере и генерируют HTML, который посылается клиенту в браузер. Если бы у вас на сервере был размещен скрипт, подобный вышеприведенному, клиент получил бы только результат его выполнения, но не смог бы выяснить, какой именно код его произвел. Вы даже можете настроить свой сервер таким образом, чтобы обычные HTML-файлы обрабатывались OS, так что клиенты даже не смогут узнать, получают ли они обычный HTML-файл или результат выполнения скрипта.

ObjectScript крайне прост для освоения, но вместе с тем способен удовлетворить запросы профессиональных программистов. Вы можете быстро начать и уже в течение нескольких часов сможете создавать простые OS-скрипты.

Область применения ObjectScript не ограничивается только веб-разработками. ObjectScript разработан таким образом, чтобы иметь полную поддержку в разработке сайтов и скриптов в кодировке UTF-8. Вы можете писать код скриптов и веб-страниц в вашем любимом редакторе прямо в кодировке UTF-8.

Многие редакторы при этом добавляют в начало файла BOM (Byte Order Mark). ObjectScript автоматически обрабатывает BOM, делая это прозрачно для разработчика.

В отличии от OS, во многих др. языках программирования BOM создает много проблем в веб-разработке, т.к. байты (составляющие BOM) отправляются в выходной поток данных генерируемых сервером. Зачастую программист даже не знает об этом, т.к. BOM не показывается в большинстве редакторов.

Это приводит к тому, что становится невозможным отправка заголовков HTTP. Например, если программист захочет в коде установить заголовок HTTP Content-type: image/png, то не сможет этого сделать после исполнения файла, содержащего BOM.

В ObjectScript такой проблемы нет, ObjectScript автоматически обрабатывает BOM таким образом, чтобы он не мешал в работе.

Кроме этого, ObjectScript имеет все необходимые функции для работы со строками в кодировке UTF-8 (вычисление длины строки, поиск подстроки и др.), а также имеет поддержку UTF-8 в регулярных выражениях.

Для целей веб разработки в парсер языка были добавлены следующие возможности:

1. Автоматическое определение UTF-8 BOM — это три невидимых служебных байта (EF BB BF) в начале документа с кодировкой UTF-8. Если BOM присутствует, то он не будет отправлен в выходной поток (проблема с отправкой BOM встречается, например, в скриптах на PHP, которая блокирует отправку заголовков HTTP);

2. Добавлены теги <% ... %> и <%= ... %>;

3. OS работает с двумя своими расширениями файлов: os и osh.

При использовании расширения os, парсер в начале файла активирует режим распознавания скрипта ObjectScript, т.е. это обычный скрипт OS.

При использовании osh (спроектировано специально для веб-разработки), файл начинается с выходного содержимого (т.е. то, что отправляется в output). Чтобы переключиться в режим скриптования, нужно использовать тег <% или <%=. Для переключения обратно в режим выходного содержимого, нужно использовать тег %>. Тег <%= выводит в output следующее за ним значение, например, <%=value%> или<%=sprintf("%.2f", num)%>.

Также можно использовать расширения html и htm, которые обрабатываются аналогично osh. Это может быть полезно при использовании редакторов html с подсветкой синтаксиса. Для подсветки синтаксиса OS, лучше всего подходит правила подсветки синтаксиса языка JavaScript.

Характеристики языка ObjectScript:

- ООП — есть;
- функции как значения первого класса — есть;
- замыкания — есть;
- компиляция в байткод — есть;
- загрузка откомпилированной программы — есть;
- модульность — есть;
- подгружаемые в рантайме модули — есть;
- кроссплатформенность — есть;
- интеграция с C/C++ — есть;
- исходники — open source;
- лицензия — MIT (можно использовать в любых продуктах бесплатно);
- идеи для языка позаимствованы из JavaScript, Lua и PHP;
- цели и задачи языка: скриптование игровой и программной логики, кросс переносимость, веб и серверная разработка;
- автор — Евгений Головин (разработчик oxsar.ru и др.) срок разработки ObjectScript — 2 месяца.

Существует много способов подключения скриптовых языков к вебу, например: свой вебсервер, модуль апача, phusion passenger, fastcgi и др. Каждый способ имеет свои плюсы и минусы.

Для сайтов с высокой нагрузкой, использующих, например, PHP, хорошо известна связка nginx + php-fpm (fastcgi). Nginx сам по себе хороший продукт и отлично себя зарекомендовал, он отдает статику (картинки, css и т.п.), а выполнение скриптов перенаправляет на php-fpm по протоколу fastcgi. Такое решение обеспечивает высокую стабильность и быструю обработку запросов. Кроме этого, fastcgi, можно подключить к апачу и др. веб-серверам.

В качестве реализации протокола fastcgi на C++ была выбрана кроссплатформенная библиотека fastcgi, с использованием которой был разработан OS-FCGI. OS-FCGI — это сервис, работающий по протоколу FastCGI, для разработки сайтов на ObjectScript.

В заключении надо заметить, новый объектно-ориентированный язык программирования ObjectScript крайне прост для освоения, но вместе с тем способен удовлетворить запросы профессиональных программистов.

Литература

1. Иан Грэхем, Объектно-ориентированны методы. Принципы и практика, Москва, «Вильямс», 2004 г.
2. Патрикеев Ю.Н. «Объектно-ориентированный анализ программирование», Москва, Московский государственный университет экономики, статистики и информации, 2002 г.
3. http://objectscript.org

Актаева А.У.[1], Баймуратов О.А.[2],Галиева Н.Г.[3], ИлипбаеваЛ.Б.[4],Искоджаева И.[5]

[1]Казахская академия транспорта и коммуникации им. М.Тынышбаева, г.Алматы, Республика Казахстан, доцент кафедры «Вычислительная техника и информационные системы»,dr.PhD, aaktaewa@list.ru

[2]Казахская академия транспорта и коммуникации им. М.Тынышбаева,г.Алматы, Республика Казахстан, доцент кафедры «Вычислительная техника и информационные системы», dr.PhD, alimzhan1@mail.ru

[3]Казахская академия транспорта и коммуникации им. М.Тынышбаева,г.Алматы, Республика Казахстан, исследователь кафедры «Вычислительная техника и информационные системы», MSc, nggaliyeva@gmail.com

[4]Казахский национальный технический университет им. К.Сатпаева,г.Алматы, Республика Казахстан, доцент , к.т.н.,l.ilipbayewa@gmail.com

[5]Казахский национальный технический университет им. К.Сатпаева,г.Алматы, Республика Казахстан, исследователь

КВАНТОВАЯ ИНФОРМАТИКА: БЕЗОПАСНОСТЬ ИНФОРМАЦИИ

«Квантовая механика - вычислительная черная магия»
(Альберт Эйнштейн)

КЛЮЧЕВЫЕ СЛОВА

Квантовая информатика, квантовые каналы связи, квантовая криптография, квантовый компьютер, фотон, кубит.

АННОТАЦИЯ

В статье обсуждается значимость в современном информационном обществе информационных ресурсов, требующих надежных методов защиты от несанкционированного доступа. Рассматривается структура и основные принципы технология квантовой криптографии на основе свойств квантовых систем. Предложен метод повышения уровня информационной безопасности и защиты конфиденциальной информации путем квантовой телепортации на каналах инфракрасной лазерной связи. Теоретическая разработка была экспериментально реализована авторами на установке для лазерной связи.

Введение

В современном информационном обществе существует большое и постоянно нарастающее количество информационных ресурсов, требующих надежных методов защиты от несанкционированного доступа. В последние годы весьма актуальной и востребованной стала проблема применения квантовых технологий в области обеспечения системы информационной безопасности и защиты конфиденциальной информации, передаваемой по открытым каналам связи. Причиной этому стали научные открытия и технологические достижения, сделавшие принципиально возможным решение целых классов сложнейших вычислительных технологий, имеющих стратегическое значение к критически важным технологиям, таким как инновационные технологии: квантовые, лазерные и оптические.[20].

В настоящее время квантовая информатика представляет собой новую, быстро развивающуюся отрасль науки, связанную с использованием квантовых технологий для реализации принципиально новых методов инфо-телекоммуникации и вычислений: квантовая информация, квантовая информатика, квантовые каналы связи, квантовая криптография, квантовый компьютер [15-21].

Квантовая информация — это физическая величина, характеризующая изменения, происходящие в системе при взаимодействии информационного потока с внешним окружением.

Квантовая информация — это новый вид информации, который можно передавать, но нельзя размножать. Квантовый бит или кубит (qubit) описывается единичным вектором в двумерном комплексном векторном пространстве и представляет собой двухуровневую квантовую систему. В качестве кубитов могут выступать ионы, атомы, электроны, фотоны, спины атомных ядер, структуры из сверхпроводников и многие другие физические системы [7].

Квантовые состояния могут использоваться для записи значений классического бита информации. Базис векторного пространства задается всего двумя единичными ортогональными векторами, обозначаемыми соответственно |0> и |1>. В отличие от классического бита, квантовый бит может быть представлен произвольной суперпозицией базисных векторов состояния фотона | ψ >=a|H>+b|V>, где $\boldsymbol{a}$ и $\boldsymbol{b}$ — произвольные комплексные числа, удовлетворяющие условию $|\boldsymbol{a}|+|\boldsymbol{b}|=1$, может быть представлено, как и в случае спина, на блоховской сфере (рис.2) и однокубитовые операции представляют собой вращение вектора Блоха[1-5].

Распространяющийся со скоростью света фотон имеет два состояния вектора поляризации (H) и (V), ортогональных друг другу и ортогональных направлению распространения фотона. Горизонтально поляризованный фотон (H) представляет базисное состояние кубита |0>, а вертикально поляризованный (V)-базисное состояние |1>:|0>=|H>,|1>=|V>. Если над кубитом производится измерение в базисе, то кубит может быть реализован в различных физических системах[1-5].

С точки зрения классической теории информации, кубиты характеризуют прямые ресурсы передаваемого сигнала, которые могут быть использованы для передачи информации по квантовому каналу связи. Для обеспечения помехоустойчивости квантовых вычислений возможен и другой подход, при котором создаются такие операции на логических кубитах, когда распространение ошибок среди физических кубитов было бы ограничено настолько, чтобы можно было использовать соответствующие корректирующие коды. Этого можно добиться путем построения специальных перекрестных (transversal) вентилей, которые осуществляли бы взаимодействие кубитов одного кодируемого кластера только с соответствующими кубитами в другом кластере [11].

Если имеется источник, производящий чистые состояния $|\psi_1\rangle, \ldots, |\psi_a\rangle$ с вероятностями $p_1, \ldots, p_a$ (аналог классического алфавита), тогда могут посылаться длинные последовательности букв слова, т.е. каждое слово задается последовательностью $w = (x_1, \ldots, x_n),\ x_j \in \{1, \ldots, a\}$.

Экспериментально, такие операции выполняются с помощью двулучепреломляющей волновой пластины, которая задерживает фазу одной поляризации на определенную долю длины волны по отношению к ортогональной к ней поляризации, вызывая вращение вектора Блоха на блоховской сфере (рис.1).

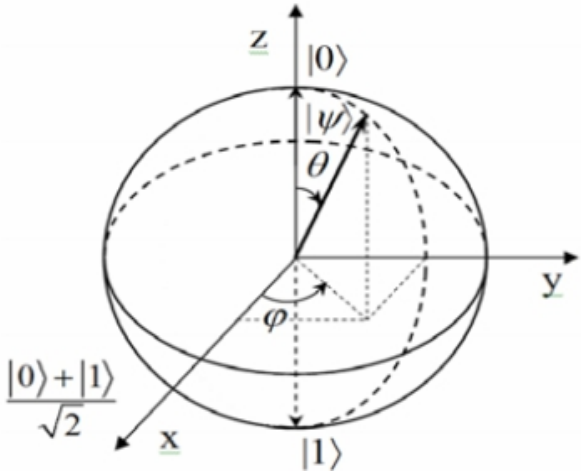


Рис. 1. Кубит на сфере Блоха

Операции над кубитами носят квантовый, вероятностный характер, что обусловливает некоторые особенности таких операций. В общем случае, выделяют три класса квантовых алгоритмов:

- алгоритмы, основанные на квантовых версиях преобразования Фурье;
- алгоритмы квантового поиска;
- алгоритмы моделирования квантовых систем [1,2].

Во всех случаях квантовый алгоритм решает задачу эффективней классического [3].

Попыткой поиска ответов на квантовые вызовы в области обеспечения системы информационной безопасности и защиты информации является квантовая криптография. Основные усилия в этой области сосредоточены на задачах синтеза стойких к возможностям квантовых компьютеров криптографических алгоритмов и протоколов (рис.2). К настоящему времени предложено несколько десятков различных по назначению протоколов квантовой безопасной связи (BB84, ЭПР, B92(4+2), SARG04, CSS, ЛО-ЧУ, Гольденберга-Вайдмана, Коаши-Имото, Пинг-Понг и др.) [1-5, 7].

Рис. 2. Основные направления исследований в СИБ [7]

Многими экспертами квантовая криптография рассматривается как единственный метод, способный обеспечить реальную защиту системам коммуникаций, как на данный момент, так и в обозримом будущем, основанный на передаче информации квантовыми состояниями фотонов (рис. 3) [1-5, 7].

Рис. 3 Схема реализации квантовой криптографии

В отличие от традиционной криптографии, которая использует математические методы, чтобы обеспечить секретность информации, квантовая криптография работает с физикой передачи информации [27].

Технология квантовой криптографии опирается на свойства квантовых систем:

- невозможность произвести измерение квантовой системы, не нарушив ее;
- невозможность определить одновременно координату и состояние частицы со сколь угодно высокой точностью;
- невозможность одновременно проверить поляризацию фотона в вертикально-горизонтальном и в диагональных направлениях;
- невозможность дублировать квантовое состояние, пока оно не измерено.

Процесс отправки и приема информации выполняется физическими средствами, при помощи фотонов в линиях волоконно-оптической связи, естественной среде или вакууме. Когерентные оптические состояния могут нести в себе большое число фотонов. Это, с одной

стороны, способствует увеличению дальности передачи секретных сообщений, а с другой стороны, создает дополнительные проблемы с защитой информации.

Особенность квантовой информации, заключающееся в том, что нельзя произвести измерение не изменив состояние системы, лежит в основе создания квантовых каналов связи, передающих классическую информацию без риска ее неконтролируемого перехвата. Квантовая «запутанность» является фундаментальным свойством квантовой механики. Она позволяет распространять квантовую информацию на десятки тысяч километров и ограничивается только лишь тем, насколько быстро и насколько далеко составляющие «запутанной» пары смогут распространиться в пространстве.

Понятно, что из-за ограниченности возможностей по измерению использовать квантовые способы передачи данных в целом невыгодно, однако, используя квантовые явления, можно спроектировать и создать такую систему связи, которая всегда может обнаруживать подслушивание. Это обеспечивается тем, что попытка измерения взаимосвязанных параметров в квантовой системе вносит в нее нарушения, разрушая исходные сигналы, а значит, по уровню шума в канале пользователи могут распознать степень активности перехватчика.

При создании криптосистем, основанных на квантовом распространении ключа, приходится сталкиваться со следующими проблемами:

- низкая скорость передачи – скорость передачи по каналам большой длины ~ Кбит, по коротким каналам ~ 10–100 Кбит;
- небольшие расстояния- до 100 км со скоростями порядка ~ Кбит;
- интенсивность импульсов квантов – обычно фотоны излучаются пучком, что позволяет злоумышленнику отделить часть фотонов и проанализировать их состояние;
- излучение одиночных фотонов заданной поляризации возможно только с некоторой вероятностью[11,18].

В настоящее время для реализации квантового канала в схеме квантовой криптографии наиболее подходящей средой является оптическое волокно, свойство которого позволяет предавать на расстояние до 120 км. Использование волокна накладывает ограничение на возможность работы с поляризационной кодировкой, поскольку оптоволокно обладает ощутимыми флуктуационными двулучепреломления. В силу этого для квантовой криптографии используется фазовая модуляция с интерферометрическими детектированием. Основополагающими принципами защиты данных в квантовых линиях связи являются невозможность копирования заранее неизвестного состояния отдельного квантового объекта и невозможность получения любой информации о квантовых состояниях этого объекта без их возмущения. Таким образом, гарантией защиты передаваемой информации выступают фундаментальные законы квантовой механики [11,18,26].

Квантовая телепортация - передача неизвестного квантового состояния на расстояние при помощи разделенной в пространстве и поделенной между двумя корреспондентами ЭПР-пары и классического канала связи. Квантовая телепортация, в отличие от плотного кодирования, происходит при отсутствии квантового канала связи, т.е. без передачи кубитов (рис. 4).

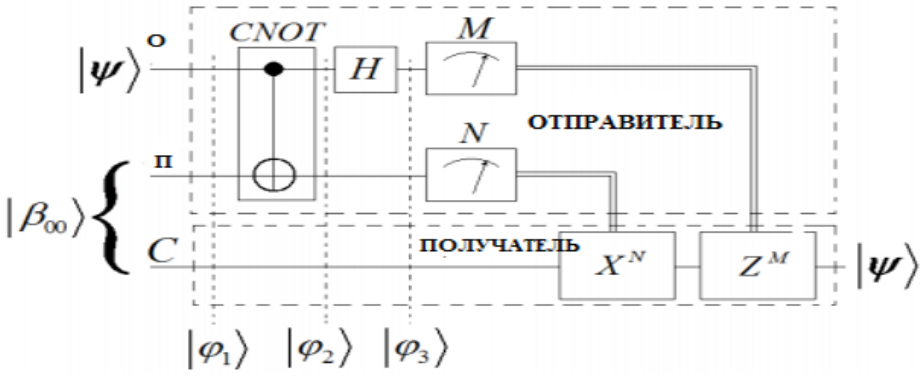


Рис. 4.Схема квантовой телепортации: одинарные линии- квантовые каналы связи ; двойные – классические[17]

Телепортация представляет собой идеальный способ передачи секретной информации, а также:

- процедура телепортации не нарушает теорему о неклонируемости;

- перенос квантовой информации от фотона к фотону может осуществляться на произвольные расстояние (более 144 км по открытой пространстве, 102 км -по оптическому волокну);
- телепортация не предполагает передачу информации о факте ее осуществления;
- классический канал (о факте передачи информации);
- если не проводить измерение состояний Белла и ограничиться проецированием на фермионное состояние, то телепортация будет успешно осуществлена в среднем один раз из четырех попыток [11,18,26].

Квантовая телепортация не дает возможности передавать информацию быстрее скорости света, как может показаться на первый взгляд, поскольку неотъемлемой частью протокола телепортации является передача информации по классическому каналу связи, а классический канал ограничен скоростью света (рис. 5).

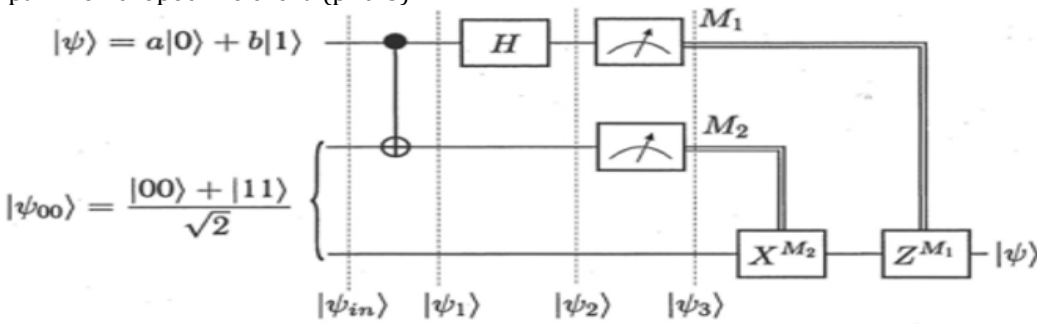


Рис.5. Квантовая схема телепортации неизвестного состояния $|\Psi\rangle$ *кубита [20]*

Квантовая телепортация используемая в качестве некоторой базисной составляющей в квантовой схеме, открывает заманчивые возможности для решения этой и ряда других экспериментальных проблем, возникающих при реализации квантовых компьютеров, она позволяет осуществлять целый ряд квантовых логических операций, невозможных при использовании прямых унитарных операций. Формирование помехоустойчивых квантовых логических вентилей сводится в этом случае к приготовлению соответствующего вспомогательного запутанного состояния в схеме однокубитовои телепортации. [11]

Новые возможности инфракрасных лазеров для реализации квантовой телепортации состояний открывают преимущество в решении проблемы передачи легко разрушаемых суперпозиционных состояний на большие расстояния без потери ими когерентности.

Лазер генерирует слабо расходящийся в воздухе пучок света (диаметр порядка 1 мм). Это позволяет осуществлять передачу открытым лучом на относительно большое расстояние (до 10 км). Следует учитывать, что солнце создает поток излучения в инфракрасной области не меньше, чем в видимой области. Оптические каналы предполагают использование двух параллельных лучей, по одному для каждого направления передачи. Диаметр чувствительной поверхности детектора обычно не превышает 1 мм (см. рис.) [25].

А чтобы исключить влияние конвективных воздушных потоков обычно используют дефокусировку пучка, чтобы даже при отклонении оси пучка пятно засветки не покидало чувствительную область детектора. Этот метод предполагает, что имеется избыток световой мощности передающего лазера. Открытый инфракрасный луч предоставляет достаточно высокий уровень безопасности, а это является следствием самой природы передачи сигнала, а не обеспечивается какими-либо специальными методами. Важнейшее свойство беспроводной оптической связи - высокая степень защищенности канала от несанкционированного доступа. Осуществить перехват канала технически весьма трудно - в силу острой направленности луча и применения уникального для каждой модели метода кодирования информации импульсами излучения. Тем не менее, для обнаружения попыток несанкционированного доступа разработан ряд мер, основанных на разнообразных принципах - обращения волнового фронта, анализа изменения принимаемого сигнала и др., что еще больше повышает защищенность канала связи [24]. Проектируя такие каналы связи надо учитывать ослабление сигнала α^0 [дБ], связанное с геометрией пучка:

$$\alpha^0 = 20 \log(\alpha^0 R * d^0) \quad \{Дб\},$$

где α^0 - угол расхождения в радианах;

R - расстояние передачи в метрах;

d^0- диаметр входного окна в метрах.

Необходимо также принимать во внимание ослабление, связанное с поглощением и рассеянием:

$$\alpha^1 = \frac{17}{S} * \left(\frac{0.55^{0.195*S}}{\lambda} \right) \{Дб/км\},$$

где α^1 - ослабление в децибелах на километр;

λ - длина волны излучения в микронах;

S - дальность видимости [км] [25].

Сигналы входного интерфейса системы используются для модуляции сигнала в открытом оптическом канале. Сама технология передачи основывается на передаче данных модулированным излучением в инфракрасной части спектра через атмосферу. Передатчиком служит – полупроводниковый излучающий диод. В качестве приемника используется высокочувствительный фотодиод. Излучение воздействует на фотодиод, вследствие чего регенерируется исходный модулированный сигнал. Далее, сигнал демодулируется и преобразуется в сигналы выходного интерфейса. С обеих сторон используется система линз, на передающей стороне - для получения коллимированного луча, а на приемной стороне, для фокусирования принятого излучения на фотодиод. Для дуплексной передачи организуется точно такой же обратный канал. Самым непредсказуемым элементом в системе является среда передачи - непрогнозируемость атмосферы с ее погодными явлениями [24]. Дальность и надежность передачи информации при инфракрасной лазерной связи зависит от погодных условий (рис. 6).

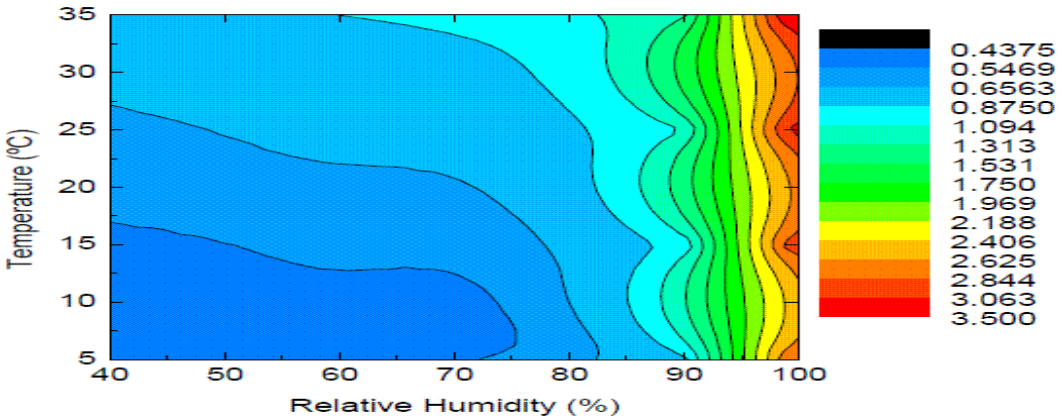


Рис. 6. Зависимость поглощения сигнала на базе в 1 км от температуры и влажности [24]

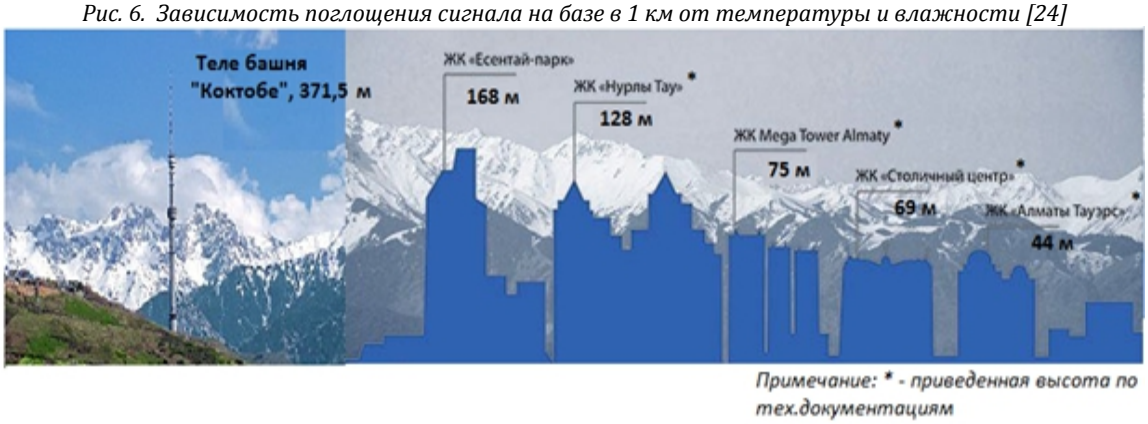


Рис. 7. Сравнение высотных зданий в г.Алматы

Обычно информация передается мощными лазерными импульсами. В их квантовом состоянии (поляризации, фазе, времени) кодируется случайная последовательность битов. В один фотон кодируется один бит и передается получателю. Главная проблема в том, что одиночные фотоны очень сильно теряются. Пройдя несколько десятков километров 99% одиночных фотонов рассеиваются. Поэтому необходимо использовать квантовые повторители, принцип которых основан на технологии квантовой телепортации.

При проведении исследования преимущества применения технологии квантовой телепортации с помощью лазерной связи были изучены возможности проектирования трасс связи и проведены сравнения высотных зданий в Алматы (см. рис.7).

Одновременно при прокладке трасс анализируются логика маршрутизации канала лазерной связи с учетом месторасположении объектов- заказчика и предложен вариант проектирования канала инфракрасной лазерной связи (см. рис. 8).

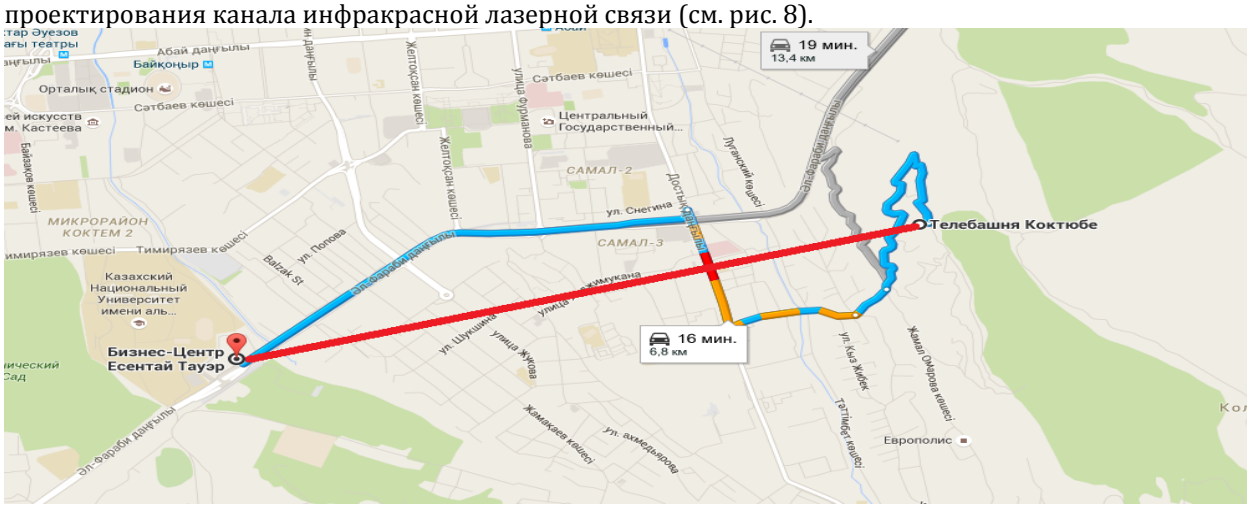


Рис.8. Предполагаемый канал лазерной связи

При проектировании линии связи традиционно ставим сервер, тянем оптоволокно к каждому потребителю информационных услуг, то вместо этого мы предлагаем два направленных друг на друга –ЛПП (лазерный приемник –передатчик), через которые передаются данные по открытому пространству с помощью лазерного света (рис.9).

Проблема телепортации затрагивает ряд вопросов принципиального характера, в частности, обмена квантовой информации в сложных пространственно разнесенных молекулярных структурах, в том числе биологических. Ни одна беспроводная технология передачи не может предложить такую конфиденциальность связи как лазерная. Перехватить сигнал можно, только установив сканеры-приемники непосредственно в узкий луч от передатчиков.

Рис . 9.Схема передачи информационных потоков

Реальная сложность выполнения этого требования делает перехват практически невозможным. Наличие лазерных лучей нельзя определить с помощью различных сканеров. Также используются разнообразные собственные протоколы передачи данных, что обеспечивают дополнительную конфиденциальность. Лазерные системы уже применяются для разнообразных приложений, где требуется высокая конфиденциальность передачи данных, включая финансовые, медицинские и военные организации. Одновременно они анализируются логикой маршрутизации.

Заключение. В связи с интенсивным развитием инновационных технологий особое значение приобретают исследования в электронике, создании интеллектуальных программных и аппаратных продуктов прикладной информатики и квантовых технологий. Квантовая

информация и технологии, основанные на ее необычных свойствах, в будущем повлияют на основы и дальнейшее развитие информационного пространства, а сама теория квантовой информации кардинально изменит современные взгляды научного сообщества на основу системы информационной безопасности.

Проведение экспериментов и исследований по обеспечению информационной безопасности представляет большой научный интерес по поиску решения основных задач и проблем, стоящих перед квантовыми криптографическими системами: задача детектирования единичных фотонов с высокой вероятностью в заданном квантовом состоянии при низком уровне ложных срабатываний, отсутствие управляемых источников одиночных фотонов, проблема увеличения дальности передачи и малая скорость генерации квантового ключа. Применение квантовых технологий в области обеспечения системы информационной безопасности - одно из наиболее парадоксальных проявлений квантовой технологии, вызывающее в последние годы огромный интерес специалистов, в первую очередь, при передаче зашифрованных сообщений по двум каналам связи - квантовому и традиционному.

Квантовая телепортация информации является одним из самых стремительно развивающихся прикладных направлений квантовой физики, и обеспечивает информирование о попытке перехвата передаваемой информации из-за необратимости коллапса волновой функции. Исследования в области квантовой телепортации информации могут привести не только к положительным последствиям, но и отрицательным. Квантовая криптография, основанная на применении квантовой телепортации, в будущем заменит все используемые криптографические системы, и будет применяться наравне с обычными средствами инфотелекоммуникации. Актуальность и масштабность проблем, связанных с обеспечением информационной безопасности, с каждым днем будут возрастать, а развитие квантовой информации в ближайшем будущем принесет свои результаты и, возможно, приведет к существенному изменению научной картины мира в области ИТ.

Литература

1. Benioff P. The computer as a physical system: A microscopic quantum mechanical Hamiltonian model of computers as represented by Turing machines // J. Stat. Phys. – 1980,V. 22, p. 563–591
2. Deutsch D. Quantum theory, the Church-Turing principle and the universal quantum computer // Proc. Roy. Soc.- London,1985, V. A400,p. 96–117
3. Cleve R., Ekert A., Macchiavello C., Mosca M. Quantum algorithms revisited // Phil. Trans. Royal Soc. - London,1998, V. A454,p. 339–354
4. Turing A. On computable numbers with an application to the Entscheidungsproblem // Proc. LondonMath. Society. - 1937. - V. 42.–p. 230–265
5. ХаляпинД.Б.Защита информации. Вас подслушивают? Защищайтесь. -М.: БАЯРД, 2004
6. Бауместер Д., Экерт А., Цайлингер А. Физика квантовой информации. -М.: Постмаркет, 2002
7. Актаева А.У., Илипбаева Л.И. Инновационные технологии в системе информационной безопасности: квантовые технологии // Современные инновационные технологии и ИТ- образование.-2014, том 1, № 1(9), 320-326 стр.
8. Белокуров В.В., Тимофеевская О.Д., Хрусталев О.А. Квантовая телепортация – обыкновенное чудо. - Ижевск: РХД, 2000
9. Бройль Л. Революция в физике. -М: Атомиздат, 1965
10. Валиев К.А. Квантовая информатика: компьютеры, связь и криптография.-М.: Вестник РАН, 2000
11. Валиев К.А., Кокин А.А. Квантовые компьютеры: надежда и реальность. -Ижевск: Регулярная и хаотическая динамика, 2001
12. Гейзенберг В. Физика и философия.- М.:Наука,1989, перевод – Акчурин И.А., Андреев Э.П.
13. Кадомцев Б.Б. Динамика и информация.- М.:Успехи физических наук, 1999
14. Клышко Д.Н. Физические основы квантовой электроники.-М.: Наука, 1986
15. Мандель Л., Вольф Э. Оптическая когерентность и квантовая оптика. -М.: Физматлит, 2000
16. Прескилл Дж. Квантовая информация и квантовые вычисления.-М.: РХД, 2008
17. Холево А.С. Введение в квантовую теорию информации. – М.: МЦНМО, 2002
18. Эйнштейн А., Подольский Б., Розен Н.// «Можно ли считать, что квантово-механическое описание физической реальности является полным?» //УФН, 1934, Том XVI, выпуск 4. - перевод – Любина А.Г., под редакцией Фока А. В.
19. Долгов В.А. и др. Криптографические методы защиты информации.-Хабаровск,2008
20. Емельянов В.И. Квантовая физика: Биты и Кубиты.-М.:Изд.МГУ,2012
21. http://www.gartner.com/newsroom/id/2819918?fnl=search&srcId=1-3478922254
22. http://www.itsec.ru
23. http://sci-article.ru
24. http://works.tarefer.ru/71/100019/index.html
25. http://book.itep.ru/3/optic_32.htm
26. Килин С.Я Квантовая информация // Успехи физической науки, 1999, Том 169, №5
27. http://tcode.tinro.ru/cryptography/src/38.pdf

Ярмухаметов Ф.Ф.[1], Кейно П.П.[2]

[1] ФГБОУ ВПО «Московский авиационный институт (национальный исследовательский университет)», г. Москва, студент кафедры «Системное моделирование и инженерная графика», science@blockset.ru

[2] ФГБОУ ВПО «Московский авиационный институт (национальный исследовательский университет)», г. Москва, преподаватель кафедры «Системное моделирование и инженерная графика»

ПРАКТИКА ИСПОЛЬЗОВАНИЯ ВЕБ-ШАБЛОНИЗАТОРОВ В ПРОГРАММНОМ КОМПЛЕКСЕ ИНТЕРПРЕТАТОРА НА ПРИМЕРЕ CTPP2

КЛЮЧЕВЫЕ СЛОВА

Веб-разработка, веб-приложение, шаблонизатор, программирование, MVC, CTPP2, C++, JSON, BlockSet, BML.

АННОТАЦИЯ

В статье обсуждаются принципы построения шаблонизаторов на основе концепции MVC. Рассматриваются особенности реализации шаблонизатора CTPP2 на языке C++. Сопоставляются показатели производительности с другими шаблонизаторами.

Разработка и внедрение современных веб-проектов является комплексной трудоёмкой задачей, решаемой целым коллективом веб-разработчиков: программистами, верстальщиками, дизайнерами. Для того, чтобы предоставить каждому участнику проекта комфортную среду для выполнения своей подзадачи, необходимо разделить структуру веб-страницы и генерируемые на ней данные. Такую возможность предоставляют шаблонизаторы.

Шаблонизатор – это программное обеспечение, оперирующее некоторым каркасом (шаблоном) сайта с использованием переменных, простых логических конструкций и встроенных функций шаблонизатора. Все перечисленные конструкции работают с данными, которые соединяются с шаблоном. В конечном итоге на выходе генерируется готовая страница в формате HTML или любом другом. Благодаря этой особенности, работа верстальщиков, дизайнеров и программистов над одним проектом может быть разграничена.

Большинство современных шаблонизаторов основаны на принципе MVC (Model-view-controller) (рис. 1) [1,2,3]. Этот принцип требует жесткого разделения структуры шаблонизатора на «Модель», «Контроллер» и «Представление». «Модель» – это абстрактный набор данных, который будет использоваться в шаблоне. При помощи компоненты «Контроллер» указывается список используемых моделей, а «Представление» используется только для вставки в шаблон конкретных данных, подготовленных «Контроллером» и «Моделью». Подобное жесткое разделение приводит к необходимости постоянной коммуникации между частями шаблонизатора, что в свою очередь приводит к замедлению работы сервера.

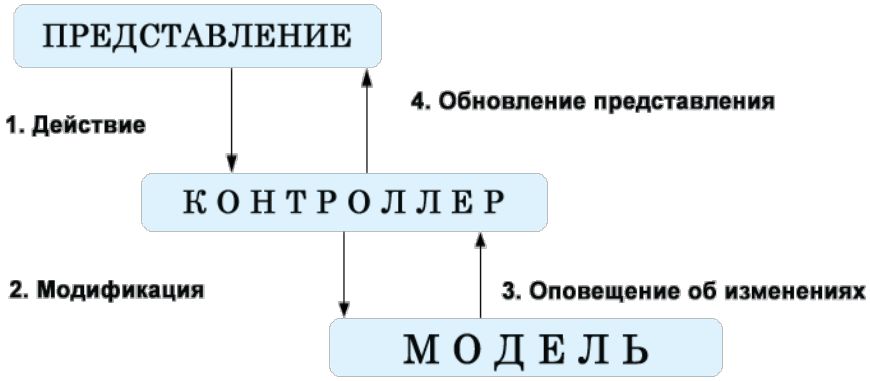


Рис. 1. Схема парадигмы MVC

Практическая реализация концепции MVC сложнее, чем её теоретическое описание и

может иметь множество особенностей реализации и применения связанных с требованиями языка программирования. Общий вид MVC может быть изменён с целью повышения скорости обработки данных. Одним из возможных вариантов является объединение компоненты «Представление» с «Контроллером». Таким образом, будет уменьшена зависимость от необходимости постоянной коммуникации между частями шаблонизатора, но остается преимущество разделения верстки страницы и обрабатываемых данных. Идея реализации шаблонизатора бесспорно удобна в языках программирования без жесткого указания типов данных.

Подобный метод реализует шаблонизатор CTPP2. Архитектура шаблонизатора состоит из четырех подсистем: компилятора шаблонов, виртуальной машины, системы кэширования шаблонов и библиотеки функций.

Компилятор шаблонов преобразует код шаблона в код виртуальной машины и сразу, «на лету» выполняет простые операции, такие как вставка других шаблонов и подстановка значений переменных. Виртуальная машина генерирует конечную страницу, используя байт-код. Если в байт-коде нет обращений к функциям шаблонизатора, то генерация происходит сразу. В противном случае, сначала вызываются необходимые функции из библиотеки.

Система кэширования шаблонов позволяет ускорить работу как виртуальной машины в частности, так и всего шаблонизатора в целом. Библиотека функций используется виртуальной машиной для обработки сложных шаблонов. Она также была спроектирована таким образом, что позволяет без особых затруднений добавлять новые функции в шаблонизатор.

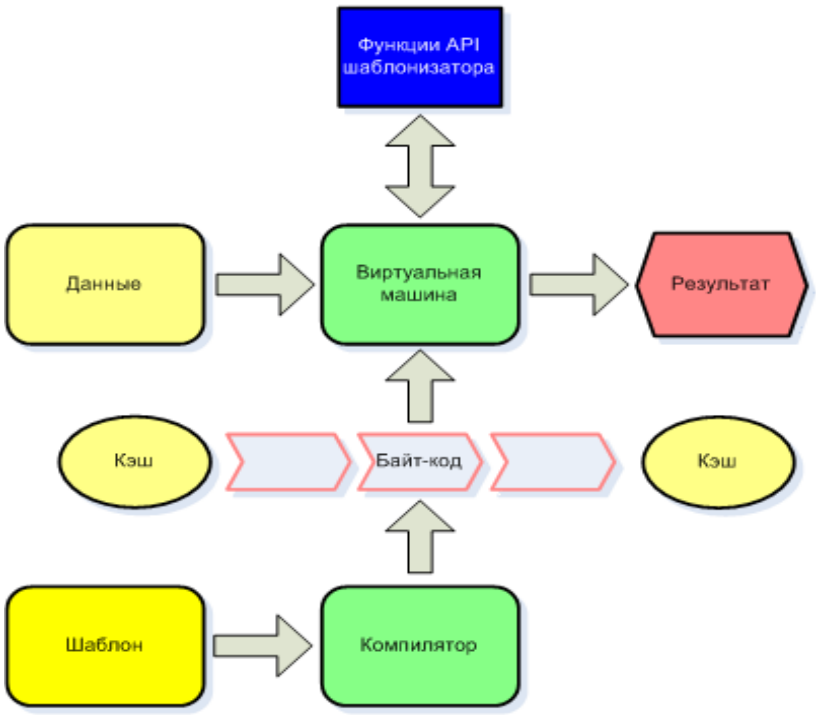


Рис. 2. Архитектура шаблонизатора CTPP2

В реализации CTPP2 в роли «Модели» выступает компилятор, в роли «Контроллера» – виртуальная машина, а роль «Представления» разделена между ними. Компилятор обрабатывает примитивные данные, а виртуальная машина изменяет их, используя различные функции.

Существуют реализации CTPP2 на PHP, PERL и C++. Различия в них заключаются исключительно в требованиях используемого языка программирования. Самое заметное различие заключается в представлении данных. Для передачи данных в шаблонизатор используется универсальный тип, называемый CDT (Common Data Type), к которому, теоретически, возможно привести данные любого типа. В PHP и PERL, благодаря отсутствию жесткой типизации данных, использование CDT не вызывает никаких проблем, но не в C++. Из-за строгого контроля типов данных в C++, в реализации CTPP2 данные могут быть представлены как значение без имени (string), переменная (HASH_VAL) и массив (ARRAY_VAL). В результате перегрузка оператора присваивания была реализована только для встроенных типов и элементов типа CDT.

Например, класс CDT хранит только строковое (или любой другой встроенный тип, а так же типа CDT) значение без имени переменной:

```
CTPP::CDT data = "Данные";
CTPP::CDT value = data;
```

В другом примере, класс CDT хранит переменную “*name*” в виде HASH_VAL со значением строкового типа (здесь: строка «Шаблонизатор CTPP2»). Кроме того, значение может быть и любого другого встроенного в язык C++ типа, а также экземпляром класса CDT. Так, во второй строке происходит переопределение новым значением из переменной “*data*” неопределённого типа, что является вполне допустимым, если “*data*” является экземпляром класса CDT.

```
CTPP::CDT var["name"] = "Шаблонизатор CTPP2";
var["name"] = data;
```

Так же есть реализация функции *Push_Back()* для заполнения массивов, причем элементом массива может быть переменная типа CDT:

```
CTPP::CDT mas;
for(int i = 0; i<10; i++){
        mas.Push_Back(i);
}
mas.Push_Bach(data);
```

Переменная *mas* заполняется элементами массива, но в представлении шаблонизатора не имеет имени, по которому к ней можно обратиться. Для того чтобы иметь возможность обратиться к массиву по имени необходимо присвоить его переменной типа CDT:

```
CTPP::CDT name_mas;
name_mas["array"] = mas;
```

В результате передача сложных структур данных в шаблонизатор становится нетривиальной задачей. Необходимо определять является ли данная переменная массивом, и, если да, то возникает необходимость создавать дополнительный элемент типа CDT с последующим его заполнением, а если те, в свою очередь, являются элементами массива, то возникает необходимость в еще одной переменной типа CDT и т.д.

Пример [4]:

```
{
        "Articles" [
                {
"Name" : "Scientific_Article",
"Author"        : "Dr. Smith",
"Comments" [
                {
"Author"        : "Dr. Johns",
"Comment"       : "Very good article"
},
{
"Author"        : "Adams, Ph.D.",
"Comment"       : "Boo…"
}
]
                },
{
"Name" : "New_Scientific_Article",
"Author"        : "Vikulin, ma",
"Comments" [
                {
"Author"        : "Ivanov, B.F.A.",
"Comment"       : "It’s very useful…"
}
]
                } ] }
```

Простым решением данной проблемы является генерация JSON-строки, подобной представленной выше, или файла с необходимыми данными, а затем передача его в функцию

шаблонизатора, которая, в свою очередь, произведет синтаксический анализ JSON-а и, при его корректности, CDT будет заполнен. Данный вариант достаточно прост, но несет в себе дополнительные накладные расходы, ведь возникает необходимость в лишнем шаге на составление JSON-а, да и анализ его достаточно дорогая операция в плане ресурсов.

Другой вариант заключается в построении структуры данных, позволяющей легко определить глубину вложенности данных, а затем, используя рекурсивно вызываемую функцию, заполнить CDT путём применения стандартных функций шаблонизатора. При условии выбора оптимальной структуры данных второй способ заметно эффективней и по памяти, и по использованию процессорного времени во время работы, и при заполнении структуры данных нет возможности допустить синтаксическую ошибку, в отличие от возможных проблем при составлении JSON-строки.

В ходе работы с шаблонизатором CTPP2 в реализации на C++ для решения данной проблемы с передачей данных было принято решение использовать второй метод, а именно согласно поступившим данным составлять вектор структуры, в которой хранится глубина текущего элемента, имя элемента и его значение [5].

```
struct data {
        int deep;
        std::string name;
        std::string value;
}
```

Затем была написана рекурсивно вызываемая функция, в которую передается вектор структуры, представленной выше. Данная функция поэлементно просматривает вектор и заполняет CDT. Если значение (value) текущего элемента вектора является пустой строкой, а значение глубины (deep) текущего элемента вектора меньше значения глубины следующего элемента вектора, то данный элемент является массивом. В этом случае функция рекурсивно вызывает себя для его заполнения, иначе элемент является простой переменной и сразу добавляется в CDT. После добавления в CDT элемент вектора удаляется, освобождая память, чего не происходит при работе с JSON-ом.

Шаблонизатор CTPP2 является одним из самых эффективных средств управления шаблонами, несмотря на сложности с добавлением данных в реализации на C++. Доказательством тому служат данные тестов производительности CTPP2 вместе с аналогами. Этот тест опубликован на сайте библиотеки [6]:

```
Template Toolkit    = 2.878 (25.2)
HTML::Template      = 2.682 (23.1)
HTML::Template::Pro = 0.596 (5.1)
HTML::CTPP2         = 0.116 (1)
```

Шаблонизаторы играют ключевую роль при реализации парадигмы MVC на современных веб-проектах, в особенности на больших высоконагруженных системах интернет-сервисов, где данная парадигма является краеугольным камнем эффективной разработки. Эти системы особенно требовательны к скорости выполнения математических операций, поэтому любое внедрение дополнительных компонентов может негативно сказаться на производительности. Статистика, приведённая выше, показывает, что шаблонизатор CTPP2 является эффективным решением, рекомендуемым для применения не только в легковесных проектах, но и на высоконагруженных системах.

Литература

1. Джесс Чедвик и др. ASP.NET MVC 4: разработка реальных веб-приложений с помощью ASP.NET MVC = Programming ASP.NET MVC 4: Developing Real-World Web Applications with ASP.NET MVC. — М.: «Вильямс», 2013. — 432 с.
2. Pitt. C. Experts Voice - Pro PHP MVC. – Apress,inc, 2012. – 500.
3. Симонова О.Н., Лясин Д.Н. Шаблон проектирования MVC как эффективное средство построения архитектуры программной системы // Современные наукоёмкие технологии. 2014. №5-2. С. 96-97.
4. Smith B. Beginning JSON – Apress,inc, 2015. – 324.
5. Кейно П.П., Силуянов А.В. Автоматизированная разработка динамических Веб-узлов средствами декларативного языка программирования. // Прикладная информатика.-2014.-6(54).-70-78.
6. Шетухин А. Документация // Шаблонизатор CTPP. URL: http://ctpp.havoc.ru/doc.

Пивнева С.В.[1], Купцов Н.А.[2]

[1] Тольяттинский государственный университет, г. Тольятти, к.п.н., доцент кафедры высшей математики и математического моделирования, tlt.swetlana@rambler.ru

[2] МБУ СОШ №20, г. Тольятти, ученик 11 класса

МАТЕМАТИЧЕСКОЕ МОДЕЛИРОВАНИЕ ПРОЦЕССА ОБУЧЕНИЯ И САМООБУЧЕНИЯ НА ОСНОВЕ МУЛЬТИЭВРИСТИЧЕСКОГО ПОДХОДА

КЛЮЧЕВЫЕ СЛОВА

Мультиэвристический подход, моделирование процесса обучения, модель мозга, детерминированные траектории, конечный автомат.

АННОТАЦИЯ

В статье рассматривается моделирование процесса обучения человека на основе мультиэвристического подхода. Модель описывает взаимодействие центральной нервной системы мозга с окружающей средой. В данной модели механизмы изменения проводимости связей и их параметры обеспечивают быстрое обучение только в случае хорошо детерминированных событий.

Моделируя процесс обучения целесообразно рассматривать развитие человека как детерминированную динамическую систему в вероятностной среде с соответствующими параметрами. При описании работы мозга мы опираемся на исследования профессора, доктора биологических наук Евгения Алексеевича Умрюхина. Модель описывает взаимодействие центральной нервной системы мозга с окружающей средой [14].

Обобщая его работы и работы физиологов П.К.Анохина и К.В. Судакова, опишем последовательность информационных процессов в мозге, которая с точки зрения психобиологического подхода является основой построения модели его деятельности [1, 13, 14]. В процессе афферентного синтеза[4] доминирующее мотивационное возбуждение с учетом поступающей извне обстановочной и пусковой афферентации[5], проигрывает в структурах мозга возможные варианты удовлетворения существующей потребности.

Модель описывает взаимодействие центральной нервной системы мозга с окружающей средой. Устройство мозга, а, следовательно, и модели обеспечивает выживание организма в среде и выполнение генетически заложенной программы продолжения вида.

Принимается решение о выборе будущей программы действий и результатов действия — параметров будущих результатов, необходимых для удовлетворения мотивации. Осуществляемое поведение в случае рассогласования параметров промежуточных или конечного результата с намеченными параметрами результатов корректируется. Коррекция обеспечивает получение намеченных результатов и удовлетворение существующей потребности. Обратная связь от совпадений или рассогласований служит основой формирования памяти, т.е. обучения.

Мозг взаимодействует со средой, осуществляя генетическую программу организма. Отметим, что предлагаемая схема представляет функциональную систему в самом общем виде. Среда составляет ее часть и определяется взаимодействием между мозгом и средой [15]. Граница между мозгом и средой является условной и зависит от конкретной функциональной системы. В модели, в аспекте ее применения для математического моделирования обучения, будем считать средой не только внешнюю среду организма, но и его внутреннюю среду. Таким образом, все, что мозг воспринимает и то, чем мозг управляет, включая разные органы организма и органы движения можно считать средой [14, 15].

Взаимодействие мозга и среды рассматривается в виде траекторий поведения, представляющих собой последовательность сигналов X и действий У. При этом, если при сигнале $X_а$, действие У вызывает переход к сигналу $X_в$, то такой переход Умрюхин Е.А. называет ситуацией

4 **Афферентный синтез** (от лат. afferens (afferentis) — приносящий) — в теории функциональной системы (П. К. Анохин) синтез материала, запечатленного в памяти, мотивации, информации о среде и пускового стимула с целью принятия решения [12].

5 **Афферентация** (в психофизиологии) (от лат. **affero** – приношу, доставляю) – термин, обозначающий передачу нервного возбуждения от периферических чувствительных нейронов к центральным [12].

$X_а \rightarrow У \rightarrow X_в$. Результаты $X_в$ при разных действиях могут быть разными, в том числе совпадать с исходным сигналом X . В общем случае результат $X_в$ зависит от исходного сигнала $X_а$ и от совершаемого действия. Одни и те же ситуации в среде могут повторяться. Если каждый раз одно и то же действие У при одном и том же исходном сигнале $X_а$ дает один и тот же результат $X_в$, то такую ситуацию называют детерминированной [15].

Если ситуация не детерминирована, но один и тот же переход от сигнала $X_а$ к сигналу $X_в$ при определенном действии совершается очень часто, почти всегда (хотя и с редкими исключениями), то такую ситуацию называют высоко воспроизводимой, а сигналы $X_а$ и $X_в$ воспроизводимыми сигналами [15].

Если при многократном появлении сигнала $X_а$ и совершении действия У сигнал $X_в$ появляется лишь в части случаев, а в другой значительной части этих случаев получаются другие сигналы, то такая ситуация может быть вероятностной. Для того, чтобы ситуация могла считаться вероятностной, необходимо, чтобы в разных реализациях среды воспроизводилась, т.е. была практически одинаковой, частота переходов между сигналами.

Три описанные ситуации — детерминированная, высоко воспроизводимая и вероятностная (как обобщение двух предыдущих) не исчерпывают различных возможных вариантов среды. Наиболее общим случаем является ситуация многовариативности, т.е. когда нельзя говорить о воспроизводимости даже частоты определенных переходов. В этом случае траектории сигналов в среде могут быть непредсказуемыми. Эта ситуация близка к хаосу [15]. Многовариативная ситуация неизбежно возникает даже для строго детерминированной среды, если информация о ней, передаваемая сигналами X, является неполной, и выбор действий ограничен. Фактически, реальная среда нашего существования всегда была и останется многовариативной [2, 3, 4]. Но при этом в ней иногда реализуются ситуации вероятностные, воспроизводимые и даже детерминированные.

Неоднозначность траекторий, т.е. переходов между сигналами X при совершении действий Y, является следствием неполноты получаемой мозгом в сигналах X информации, даже если за этими сигналами скрывается строго детерминированный мир. Реальный мир и взаимодействующий с ним мозг устроены так, что при соответствующем выборе сигналов X и действий Y становится возможным выделение и запоминание повторяющихся траекторий поведения. П.К. Анохин определил это свойство окружающего мира как наиважнейшее для формирования в мозге способности предвидения важных для выживания организма будущих событий [15].

При соответствующем выборе характеристик воспринимаемых сигналов и действий среда становится вероятностной. Это значит, что определенные траектории воспроизводятся, повторяются с определенной частотой, которая может в определенных случаях приближаться к единице. При этом траектории становятся высоко воспроизводимыми или даже детерминированные [14]. Оптимальное обучение в определенной среде и должно обеспечить выбор высоко достоверных, практически детерминированных траекторий.

Для математического моделирования обучения приведем обоснование применения предлагаемого мультиэвристического подхода [6, 10, 11].

Определим модель обучения как конечный автомат *A* с выходным языком как кортеж[6]

$$A = (Q, X, Y, \delta, \pi, I, F),$$

где

- Q – конечное множество состояний; в нашей модели они выражают принятие решения в процессе обучения;
- X – входной алфавит *{H, C} (сигналы)*;
- Y – выходной алфавит *{H, C} (действия)*;
- δ – функция переходов состояний (transition function) или траектории поведения под действием символов входного алфавита X, т. е. $\delta : Q \times X \rightarrow 2^Q$;
- π – функция выходов под действием символов входного алфавита X, т. е. $\pi : Q \times X \rightarrow Y$;
- I – множество начальных состояний;
- F – множество заключительных состояний.

Рассмотрим две возможные ситуации:

1) правильно принятое решение;
2) неправильно принятое решение.

6 Понятие согласуется с терминологией, введенной в [6].

С помощью конечного автомата моделируется т. н. процесс принятия решений. Человек принимает решения по потоку информации в каждый момент времени, оценивая ее. Он может правильно оценивать ситуацию, состояние (С), а может и неправильно – состояние (А) (рис. 1).

С конечным автоматом A естественно связывается конечная марковская цепь с вероятностями переходов $p_{i,j}$ из состояния i в состояние j (рис. 2).

Для вероятностей перехода $p_{i,j}$ выполняются естественные соотношения $p_{i,j}>0$ и $\sum_j p_{i,j}=1$, т. е. если мы находимся в состоянии i, то в какое-нибудь состояние мы перейти должны [5]. Скрытые марковские модели предполагают, что мы не знаем, сколько состояний и какие связи между ними, т. е. структура модели неизвестна, мы можем выдвигать гипотезы и определять параметры модели.

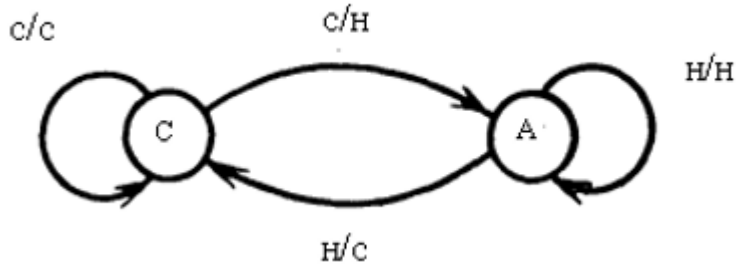


Рис. 1. Пример автомата A, моделирующего обучение

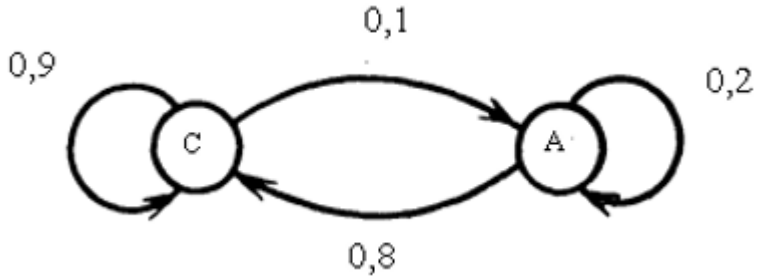


Рис. 2. Пример автомата A с вероятностями переходов

Рассмотрим теперь оптимальное обучение, т. е. детерминированные траектории поведения. Результат преобразования автомата A_1 представлен на рис. 3 в виде графа состояний автомата A_2.

Алгоритм сворачивания ребер графа состояний эквивалентен выделению системы образующих подгруппы в конечно порожденной свободной группе и может использоваться в качестве вспомогательного, эвристического алгоритма для поиска недетерминированного состояния, эквивалентного заданному состоянию обучения [5].

Итак, соотнесем представленный математический аппарат с моделью, воспроизводящей структуру и механизм работы мозга.

Рассмотрим механизм формирования и изменения проводимостей связей при обучении модели [7, 8, 9]. В необученной модели связи между элементами блока (мозга) имеют малые проводимости, а также множество траекторий a, b. На рис. 4 определим:

- X – входные сигналы;
- Y – выходные действия;
- 1 - мотивационный элемент;
- L – условно выделенный слой мозга;
- 2, 3, 4, 5, 6, 7 – элементы структуры мозга;
- a, b – траектории.

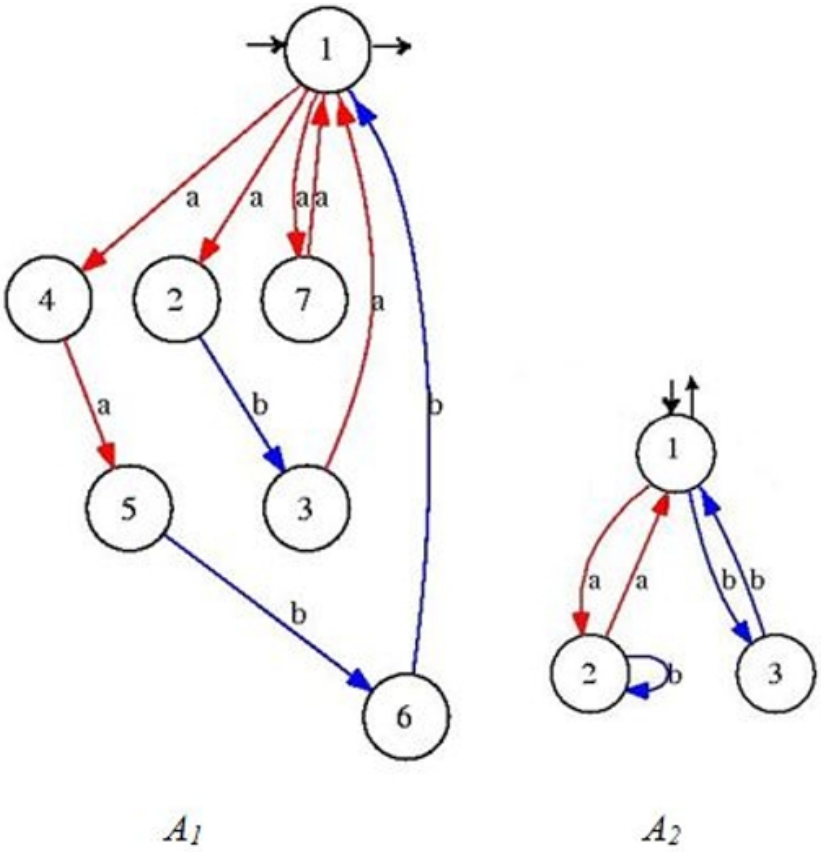


Рис. 3. Графы состояний исходного недетерминированного автомата A_1 и свернутого детерминированного A_2

При этом учитывается, что различие траекторий a и b относится к разным сигналам, имеющим разный уровень возбуждения соответствующий значимости информации. Возникающая в организме потребность выражается в возбуждении, распространяющемся из элемента 1 по связям или траекториям a и b в слой мозга. На вход поступают сигналы X, соответствующие входной афферентации.

Условно выделенные элементы мозга одного уровня соединены связями a, а элементы разного уровня связями b. Принятый в модели механизм изменения связей a и b при обучении одинаков для разных уровней и отличается только временными параметрами. Память формируется с помощью обратной связи от совпадения или рассогласования в акцепторе результатов действия намечавшихся результатов с получаемыми результатами.

Пусть в мозг поступают входные данные X, тогда получение конечного результата Y, закодированного в мотивационном элементе 1, служит сигналом положительного подкрепления для элементов мозга. Допустим, положительной подкрепление осуществляется через возбуждение элемента 4, выполняющего функцию положительной эмоции. Возбуждение из элемента 4 распространяется в слой мозга, а траектория a остается как «след» в памяти. На рис. 4 представлена цепочка элементов «слоя» мозга (1→4→5→6→1), а также траектории для запоминания в памяти, которая составляет акцептор результатов действия, намеченный для получения результата Y. При успешном осуществлении действия, каждый элемент данной последовательности мозга возбуждается, элементы выделенной цепочки сохраняют «след» этого возбуждения и, следовательно, вероятность наступления подобного действия в будущем возрастает

В случае рассогласования в акцепторе результата действия, т.е. несовпадения получаемого результата с намеченным результатом, переходим в модели к элементу 2 или 7, формирующему отрицательное подкрепление и отрицательную эмоцию. При этом существенно уменьшаются проводимости связей: (1→2→3→1) и (1→7→1).

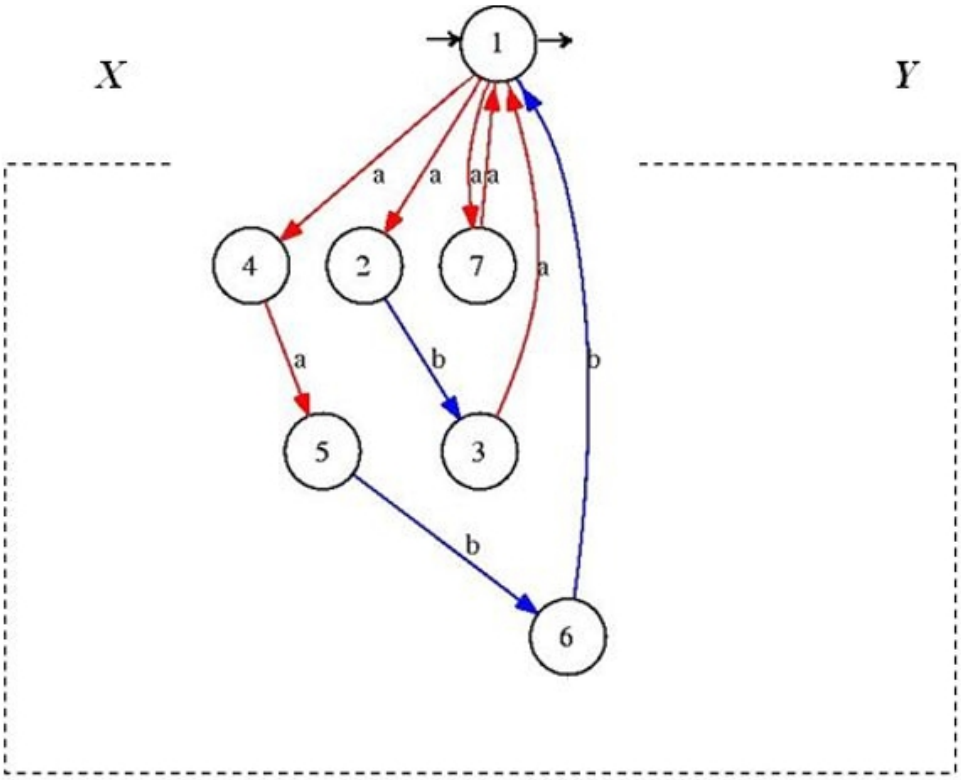


Рис. 4. Представление необученной модели мозга, как графа состояний исходного недетерминированного автомата

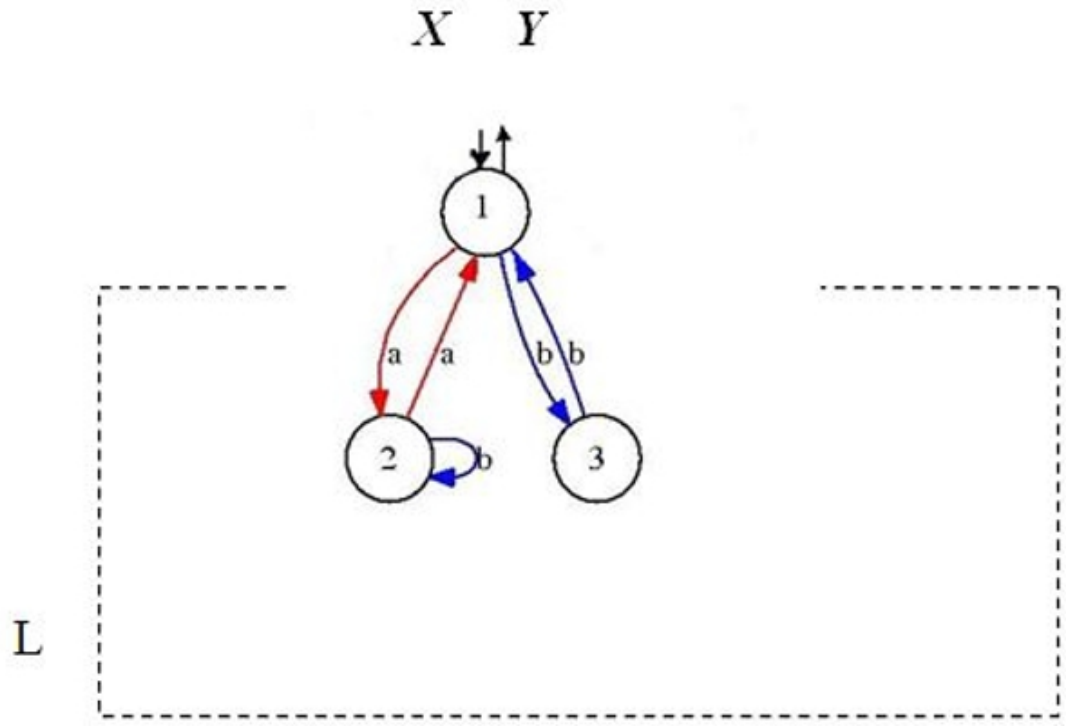


Рис. 5. Представление обученной модели мозга, как графа состояний свернутого детерминированного автомата

С элементами 4, 2, 7 связано представление в модели положительных и отрицательных эмоций. В слое мозга каждого блока, проходя по цепочкам элементов памяти, мотивационное возбуждение «проигрывает» возможные варианты будущих траекторий поведения.

На рис. 5, в качестве свернутого детерминированного, выступает «обученный» мозг, с

установившимися детерминированными траекториями.

Поясним представленную модель. В этой модели связи или траектории a и b, объединяющие механизм доступа к информации и ее представление в виде сложных перцептивных элементов, реализуют принцип сочетания смысла информации и ее адреса. Это обеспечивает модели быстрое формирование будущей траектории поведения с учетом мотивации, входной афферентации и памяти без излишнего просмотра и перебора неадекватных вариантов такой траектории. В данной модели механизмы изменения проводимости связей и их параметры обеспечивают быстрое обучение только в случае хорошо детерминированных событий.

Применяемая эвристика отвечает на следующий вопрос: насколько чаще (реже) нужно обрабатывать недетерминированные вершины по сравнению с детерминированными? Ответ на вопрос мог быть найден с помощью генетических алгоритмов, но существует простое решение, основанное на рассмотрении нескольких возможных вариантов. На основе практического программирования делаем вывод, что для вершин обоих видов необходим примерно одинаковый объём работы [10, 11]. То есть алгоритмом, близким к оптимальному, является обработка одной детерминированной вершины после обработки одной недетерминированной и т.д.

Хочется отметить, что, предлагая такой вариант математического моделирования процесса обучения, авторы не претендуют на воспроизведение уникальности человеческой природы, его мозга, уникального сознания, поведения, настроения, чувств и т.д. Модель основана на принципе повторяемости, воспроизводимости в своих свойствах и явлениях человеческого мозга. Еще П.К. Анохин отмечал, что «эта повторяемость явлений в окружающем мире обусловила развитие и наличие способности мозга к предвидению будущих результатов» [1].

В настоящее время авторами проводится работа по созданию на основе предложенного математического аппарата компьютерной автоматизированной системы. Такая система должна позволить смоделировать принятие «правильного» управленческого решения в сложных условиях многовариативности.

Литература

1. Анохин П.К. Особенности афферентного аппарата условного рефлекса и их значение для психологии // Вопросы психологии. 1955. № 6. С. 16-38.
2. Витковская Н.Г. Использование технологии проблемного обучения при подготовке журналистов // Вестник Волжского университета им. В.Н. Татищева. 2013. Т. 2. № 4[14]. С. 77-84.
3. Витковская Н.Г. К вопросу о формировании информационной компетентности будущих журналистов в процессе их обучения в вузе // Вестник Волжского университета им. В.Н. Татищева. 2013. № 3[13]. С. 79-88.
4. Витковская Н.Г. Информационная компетентность студентов как одно из условий индивидуального обучения // Вестник Волжского университета им. В.Н. Татищева. 2015. № 2[18]. С. 70-78.
5. Крайнюков Н.И. Обратимые конечные автоматы и условия вложения полугруппы в группу // Вектор науки Тольяттинского государственного университета. 2010. № 3. С. 34-36.
6. Мельников Б. Ф. Применение мультиэвристического подхода в задаче вершинной минимизации недетерминированных конечных автоматов / Б. Ф. Мельников, С. В. Пивнева, А. В. Цыганов // Нейроинформатика, ее приложения и анализ данных : материалы XVII всерос. семинара (ИВМ СО РАН, 2-4 октября 2009 г.). – Красноярск, 2009. – С. 73-78.
7. Мельников Б. Ф. Эвристические алгоритмы принятия решений в гуманитарных областях / Б. Ф. Мельников, С. В. Пивнева // Известия Самарского научного центра Российской академии наук. Вып. 8. – Самара : Изд-во Самарского научного центра РАН, 2009. – С. 137-143.
8. Пивнева С. В. Математическое моделирование процесса обучения // Проблемы университетского образования: содержание и технологии : труды II всерос. науч.-метод. конф. – Тольятти : Тольят. гос. ун-т, 2005. – С. 67-71.
9. Пивнева С. В. Принятие решений в гуманитарных областях – подходы к репрезентативности входных данных и сравнению эффективности алгоритмов // Информационные технологии моделирования и управления. – № 8 (51) – Воронеж : Изд-во «Научная книга», 2008. – С. 888-896.
10. Пивнева С. В., Мельников Б. Ф. Мультиэвристический подход применительно к задачам дискретной оптимизации // Методы и средства обработки информации : труды Третьей всерос. науч. конф. (ВМК МГУ 6-8 октября 2009 г.). – М., 2009. – С. 268-280.
11. Пивнева С.В. Минимизация недетерминированных конечных автоматов по различным критериям // Вектор науки Тольяттинского государственного университета. 2011. № 1. С. 19-21.
12. Словарь практического психолога / Сост. С. Ю. Головин. – Минск: Харвест, 1998. – 800с.
13. Судаков К.В. Системное квантование жизнедеятельности // В кн. Системокванты физиологических процессов. М. 1997. С. 9-53.
14. Умрюхин Е. А. Целенаправленное поведение и самообучение живых организмов // Изв. РАН. ТиСУ. – 2003. – № 3. – С. 114-124.
15. Умрюхин Е.А., Судаков К.В. Теория хаоса: преобразующая роль функциональных систем // Российский физиологический журнал им. И.М. Сеченова. Т.83, № 5-6, 1997. С.190-216.

Макашов П.Л.[1], Романенко Н.А.[2]

[1] ЗАО «КонсОМ СКС», город Магнитогорск, менеджер проектов, makashov.p@konsom.ru

[2] ЗАО «КонсОМ СКС», город Магнитогорск, менеджер проектов, romanenko.n@konsom.ru

СЕРВИС-ОРИЕНТИРОВАННЫЙ ПОДХОД К УПРАВЛЕНИЮ ИТ ПРОЕКТАМИ НА ПРИМЕРЕ ИСПОЛЬЗОВАНИЯ ПРОГРАММНОГО ПРОДУКТА «JIRA»

КЛЮЧЕВЫЕ СЛОВА

Сервис-ориентированный подход, программные проекты, управление проектами, методология, облачные технологии.

АННОТАЦИЯ

Внедрение информационных систем в компании любого размера, неизбежно связано с организационными и техническими трудностями. Как правило, подобные проекты долговременные, и зачастую поставленным целям могут не соответствовать. В данной статье рассматривается внедрение корпоративного сервиса управления проектами, анализируются проблемы, возникающие при внедрении корпоративного сервиса, рассматриваются преимущества внедрения подобных сервисов. На примере будет рассмотрено внедрение корпоративного сервиса на основе программного продукта «Jira» в существующую систему управления ИТ проектами. Внедрение данного программного продукта позволяет улучшить контроль над выполнением ИТ проектов.

Современные компании, занимающиеся автоматизацией крупных промышленных предприятий, в первую очередь проектные организации, т.к. проектный подход позволяет эффективно управлять разработкой и внедрением большого количества решений по автоматизации. При этом возникает задача по эффективному управлению пулом проектов. Для организаций, осуществляющих свою деятельность в пределах одного города или одного предприятия, достаточно просто придерживаться одной из методологий управления проектами (например, PMBoK). Однако для организаций имеющих широкую географию одновременно реализуемых проектов этого недостаточно. Необходим инструмент управления проектами, который позволяет консолидировано отслеживать и контролировать ход выполнения проектов, занятость ресурсов, который позволяет оперативно обмениваться документами, выдавать задания и распоряжения. При этом есть необходимость получать безопасный доступ к этому инструменту из любой точки мира, посредством сети интернет. В настоящее время подобным критериям соответствуют сервис ориентированные приложения.

Опишем опыт применения сервис-ориентированного подхода, одной из ведущих проектных организаций в области промышленной автоматизации. В настоящий момент программные решения ЗАО «КонсОМ СКС» внедрены более чем в 150 автоматизированных системах, данный факт свидетельствует о надежности разработок и выгодно сказывается на имидже компании. ИТ-проекты, реализуемые компанией ЗАО «КонсОМ СКС», охватывают все ИТ-пространство современных промышленных предприятий. Это позволяет: повысить ритмичность производства; сократить простои; оптимизировать технологические маршруты; снизить потери от брака и отходов производства и т.д. ЗАО «КонсОМ СКС» занимается реализацией крупномасштабных, затратных, долгосрочных и сложных проектов. Управлять такими проектами достаточно трудно, так как даже небольшие отклонения в ходе реализации могут привести к их срыву и большим финансовым потерям.

Внедрение новых для компании технологий нацелено в первую очередь на достижение основных целей бизнеса и фокусируется на оптимизации процесса управления проектами.

Для управления проектами ЗАО «КонсОМ СКС» использует общепринятый стандарт PMBOK, согласно которому управление проектами разбивается на 5 групп процессов (рис.1): инициация; планирование; исполнение; мониторинг и управление; завершение.

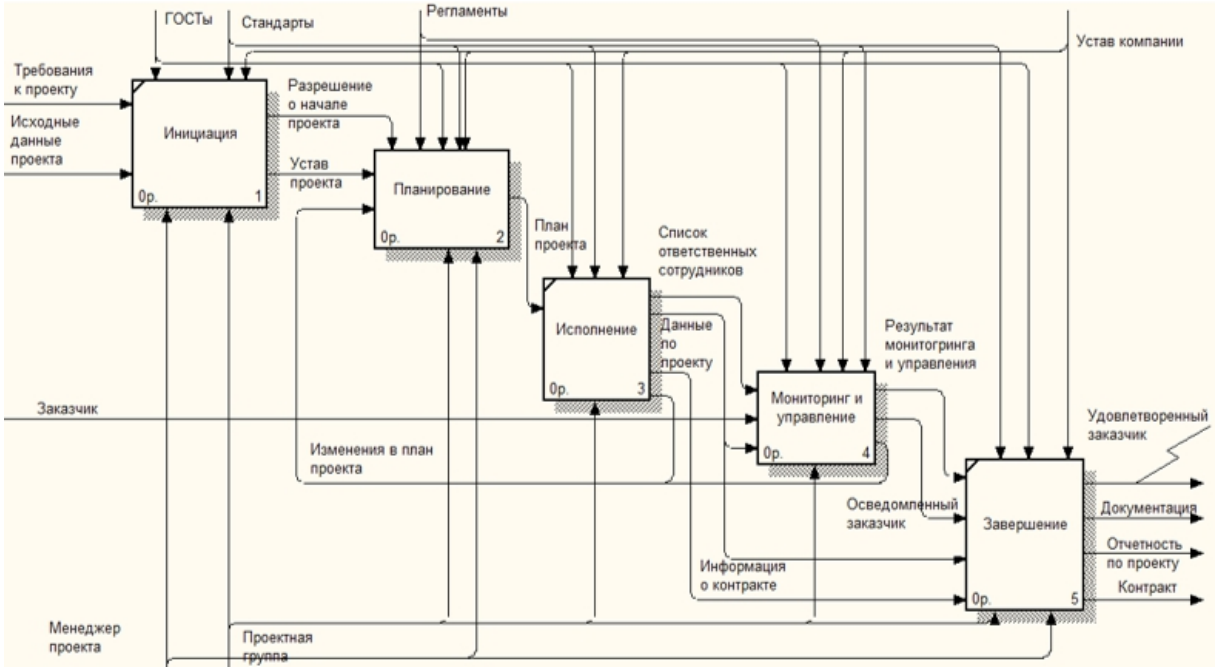


Рис.1. Декомпозиция управления проектами ЗАО «КонсОМ СКС»

С целью повышения эффективности управления ИТ-проектами компании, был произведен опрос ведущих менеджеров компании, проанализирован архив проектов, что показало необходимость:

- сбалансирования программы, то есть достижение равновесия между краткосрочными и долгосрочными проектами, между рисками проектов и возможными доходами от их реализации, разработка новых и улучшение старых и так далее;
- мониторинга процессов планирования и выполнения выбранных проектов. В частности, принятие решений относительно выделения ограниченных ресурсов, обеспечение всех проектов необходимыми ресурсами в адекватном количестве при одновременном обеспечении выгодного и эффективного использования ресурсов;
- сравнения возможностей новых проектов между собой и по отношению к проектам, уже включенным в портфель, а также оценка их взаимовлияния;
- обеспечения стабильного и эффективного механизма управления проектами.

В современных условиях команда проекта может находиться в удаленных местах друг от друга, и поэтому для успешной совместной работы над проектом проще всего использовать именно сервис-ориентированный подход. Также нашей компанией, для более эффективного управления программами проектами встала необходимость внедрения сервиса, удовлетворяющего следующим критерия. Критерии были определены экспертными оценками ведущих менеджеров.

Таблица 1 – Требования к системе

№п/п	**Критерий**
1.	Возможность формирования нескольких проектов.
2.	Многопользовательское управление.
3.	Многоязычность.
4.	Формирование отчетов.
5.	Отслеживание проекта.
6.	Документооборот.
7.	Управление содержанием.
8.	Коммуникации.
9.	Кроссбраузерность
10.	Интеграция с мобильными устройствами.

11.	Авторизация с помощью клиентских SSL сертификатов
12.	Импорт файлов MS Project
13.	Пул ресурсов.

Проанализировав вышеизложенные требования в качестве сервис-ориентированного инструмента было выбрано приложение JIRA, разработанное компанией Atlassian Software Systems. Имеет Web-интерфейс, поддерживает защищенные типы интернет соединений (HTTPS). Таким образом, было принято решение внедрять JIRA как корпоративный сервис управления проектами.

В процессе внедрения JIRA нами были определены роли в системе. Участники системы, их компетентность и роли представлены в таблице 2.

Таблица 2 – Участники, компетентность и роли в ИСУП

№	Должность	Компетентность	Роль в системе
1	Руководитель отдела	Отвечает за организацию работы в системе; руководитель ОПУ несет ответственность перед генеральным директором за результаты деятельности; руководитель принимает оперативные решения.	Администратор
2	Аналитики	Основной задачей аналитиков является мониторинг программы проектов, на основании данных которого принимаются предварительные управленческие решения, касающихся приоритета проекта, распределения ресурсов и судьбы проекта.	Пользователь, занимающийся мониторингом ресурсов и финансов
3	Финансист-контроллер	Это специалист, который должен следить, чтобы при постановке целей, планировании и принятии решений менеджером прибыль была достаточной. Включая: процент оставшегося резерва; прогнозирование инфляции; совет по перераспределению денежных средств; изменения со стороны законодательства.	Пользователь, занимающийся мониторингом финансовой стороны программы
4	Риск-менеджер	Риск-менеджер должен обнаруживать в ежедневной работе программы всевозможные риски, оценивать и снижать их негативные последствия.	Пользователь, занимающийся мониторингом рисков программы
5	Менеджер проекта	Несет ответственность перед руководителем отдела проектного управления за результат исполнения проекта	Пользователь, с определенными правами, который отвечает за реализацию своего единичного проекта
6	Проектная группа	Временная структура, которая формируется на этапе инициации проекта и является координирующим центром управления отдельным проектом.	Пользователи-ресурсы
7	Стейкхолдеры	Заинтересованные лица в реализации программы или отдельно взятого проекта	Гости, которым виден лишь ход отслеживания

Внедрение сервис-ориентированной системы по управлению проектами позволило организации:

- улучшить контроль по выполнению проектов;
- повысить удовлетворенность клиентов, т.к. они могут самостоятельно отслеживать ход выполнения проекта;
- повысить эффективность использования ресурсов;

- улучшить коммуникации, т.к. JIRA идеально подходит для территориально-распределенных организаций. Выше было сказано о том, что ЗАО «КонсОМ СКС» работает с проектами не только в городе Магнитогорске, а также по всей России и за рубежом;
- добиться мобильности, т.е. возможность сотрудникам работать повсюду, где есть доступ в Интернет;
- значительно экономить время.

Отметим, что ежегодно продуктивность процесса управление проектами будет расти, не только за счет системы, но и за счет компетентности персонала.

Необходимо также отметить, что существенную долю в проектах автоматизации крупных промышленных предприятий составляет разработка специализированного программного обеспечения. Для этого в ЗАО «КонсОМ СКС» существует отдел разработки и внедрения ПО, где работают более 40 сотрудников. Для повышения эффективности работы команды разработки программных продуктов, в компании следуют методологии «Agile». «Agile» это гибкая методология разработки программных продуктов. В рамках данной методологии выполняется командная разработка программного обеспечения. Целью методологии «Agile» является минимизация рисков, путем сведения разработки к серии коротких циклов, называемых итерациями, которые обычно длятся одну-две недели. Каждая итерация сама по себе выглядит как программный проект в миниатюре, и включает все задачи, необходимые для выдачи мини-прироста по функциональности: планирование, анализ требований, проектирование, кодирование, тестирование и документирование. В «Agile» делается упор на непосредственное общение лицом к лицу [1].

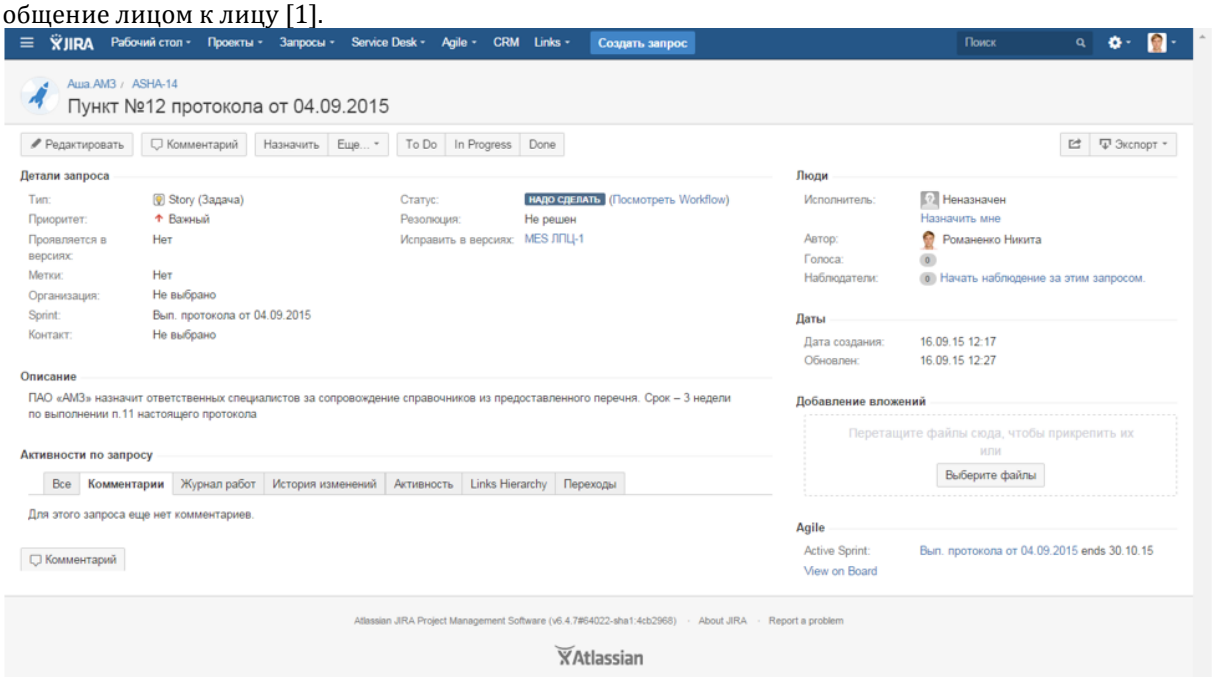


Рис. 2. Пример визуализации информации в системе

Для автоматизации бизнес-процессов, имеющихся в отделе разработки программного обеспечения, также подходит программный продукт «JIRA». Таким образом и разработчики программного обеспечения получили эффективный сервис управления. Преимущества использования «Jira» [5]:

- пользователь может использовать для работы ПК практически любой конфигурации;
- облачные технологии позволяют пользователю не быть привязанным к своему месту работы, пользователь может работать в любом месте, и может использовать любой ПК, имеющий выход в интернет;
- пользователи могут вести совместную работу, и могут делиться результатами своей работы с другими участниками команды проекта;
- пользователь застрахован от потери данных, т.к. они хранятся в облаке, и поломка ПК пользователя не приведет к потере данных;
- формирование опыта в дальнейшем планировании подобных работ.

Для компаний неоспоримым преимуществом выноса части работы в «облако» является снижение затрат на обслуживание, поддержку, модернизацию и администрирование аппаратного и программного обеспечения на месте [5].

Для руководителей проектов JIRA предоставляет механизмы отслеживания ресурсов, для этого в сервисе «Jira» выделены следующие категории необходимой информации (рис. 2): информация по текущей работе ресурса: название работы; время решения работы (в часах, днях); общее количество текущих работ ресурса; интересующие работы ресурса; выполненные задачи; предстоящие работы.

Это позволяет увидеть многозадачность ресурса, отслеживать его работу, повысить заинтересованность в работе над проектом и увеличить продуктивность.

Заключение и обсуждение

С помощью программного продукта «Jira» нами были описаны и настроены бизнес-процессы позволяющие:

- создавать проекты. Каждый проект уникален, и имеет свое название, описание, команду и приоритет;
- создавать запросы (работы). Запросы создаются менеджером проектов или начальниками отделов, которые впоследствии распределяются между непосредственными исполнителями. У каждого запроса есть описание (задание), ориентировочная продолжительность, комментарии, исполнители и приоритеты. К запросам можно прикреплять документы, и ограничивать доступ к ним;
- подбирать трудовые ресурсы. Система позволяет подбирать трудовые ресурсы в соответствии с определенными критериями – загрузка, опыт, отношение к отделу;
- формировать графики. Формирование графиков подразумевает визуальное соотношение созданных и решенных запросов (работ) по проекту;
- отслеживать ход выполнения работ (запросов). При назначении работы (запроса) проставляется время начала, после окончания – время завершения. Это позволяет формировать опыт в дальнейшем планировании подобных работ;
- автоматизировать бизнес-процессы отдела разработки программного обеспечения.

Сервис позволяет управлять несколькими проектами, многопользовательское управление, многоязычность, импорт файлов MS Project документооборот, возможность коммуникаций, кроссбраузерность и доступ через мобильные устройства, что позволяет вести мониторинг в удобное время.

При внедрении сервиса были определённые трудности.

- перенос всех данных по проектам в новую систему и обучение пользователей без отрыва от основного производственного процесса;
- довольно длительный переходный период, в процессе которого пользователи привыкали работать в системе.

Таким образом, опыт внедрения сервиса управления проектами можно признать успешным. Применение сервис-ориентированного подхода для решения задач по правлению проектами в нашей организации себя оправдало, именно поэтому наша компания приняла решение дальше развивать сервис ориентированные подходы, направленные на повышение эффективности бизнеса.

Литература

1. Atlassian JIRA 6.2: Быть лучше, чем вчера [интернет портал]. URL: http://habrahabr.ru/post/213867.html (дата обращения 26.09.2015).
2. Asif, M., Sawar, J. and Abdullah, U, “Design of Decision Support System in Electronic Medical Record Using Structured Query Language”, DOI:10.7763/IPEDR.V63.3 2013.
3. What is Agile? What is Scrum? [интернет портал]. URL: https://www.cprime.com/resources/what-is-agile-what-is-scrum/ (дата обращения: 29.09.2015).
4. Scrum Methodology and Project Management [интернет портал]. URL: http://www.mountaingoatsoftware.com/agile/scrum (дата обращения: 25.03.2015).
5. Макашова В.Н., Старков А.Н., Чусавитина Г.Н. Информационные системы и технологии [Текст]: практикум. - Магнитогорск, 2011.- 188 с.
6. Макашова В.Н., Миронова А.А. Применение информационных технологий как инструмента минимизации рисков инвестиционных проектов в сфере автоматизации промышленных предприятий//Инновационный Вестник Регион. 2013. № 4.2. С. 55-60.
7. Ошурков В.А. Управление ресурсными рисками в проектах по разработке программного обеспечения [Текст] //В.А. Ошурков, В.Н. Макашова. – Москва: Экономика и социум, 2014. № 3(12). С. 437-443.

8. Ошурков В.А. Обзор стандартов в области управления рисками [Текст] //В.А. Ошурков, В.Н. Макашова. – Современные тенденции развития науки и технологий. 2015. № 1-5. С. 79-81.
9. Ошурков В.А., Макашова В.Н. Механизмы оптимизации управления программой ИТ-проектов // Сборник научных трудов SWORLD. – N 1. – С.66-72.
10. Почему программные проекты проваливаются, и как сделать их успешными [интернет портал]. URL: http://www.pmtoday.ru/project-management/rescue-and-recovery/why-software-projects-fail.html (дата обращения: 25.09.2015).
11. Управление развитием крупномасштабных систем | Россия, Москва | 1—3 октября 2012 г. – [Электронный ресурс]. – 2012. – Режим доступа: http://mlsd2012.ipu.ru/content/содержание-материало-конференции-т2 (дата обращения: 20.09.2015).
12. Чусавитина Г.Н., Макашова В.Н. Использование информационных технологий в управлении проектами [Текст]: учеб. пособие - Магнитогорск, 2011. – 216 с.

Ошурков В.А.[1], Макашова В.Н.[2]

[1] ЗАО «КонсОМ СКС», аналитик, г. Магнитогорск, oshurkov92@mail.ru

[2] ФГБОУ ВПО «Магнитогорский государственный технический университет им. Г.И.Носова», г.Магнитогорск, кандидат пед. наук, доцент кафедры бизнес-информатики и информационных технологий, makashova.vera@mail.ru

ОПЕРАТИВНОЕ ПЛАНИРОВАНИЕ ПРОИЗВОДСТВА В MES-СИСТЕМАХ С ИСПОЛЬЗОВАНИЕМ МЕТОДОВ И АЛГОРИТМОВ ИСКУССТВЕННОГО ИНТЕЛЛЕКТА

КЛЮЧЕВЫЕ СЛОВА

Открытые искусственный интеллект, планирование производства, система управления производством, MES-система.

АННОТАЦИЯ

В данной статье приведено описание системы управления производством (далее MES-система), рассмотрены основные функции и определено, что организация любого производства начинается с его планирования (оперативного, календарного). В основе функций планирования производства лежат различного рода методы и алгоритмы искусственного интеллекта. Таким образом, нами были рассмотрены наиболее распространенные методы и алгоритмы искусственного интеллекта для осуществления планирования производства: метод табуированного поиска, муравьиный алгоритм, генетические алгоритмы, нейронные сети.

Введение

Стремительный рост конкуренции на рынке производственных предприятий, обусловленный экономической ситуацией и высокими требованиями заказчиков, все чаще заставляет менеджмент пересматривать прежние подходы к управлению производством [16]. Для того чтобы удерживать свои позиции на рынке, производить продукцию высокого качества и при этом стабильно быть в прибыли, любое современное промышленное предприятие должно уделять внимание оптимизации и автоматизации технологического процесса. Это касается как модернизации оборудования, обучения персонала, так и внедрения новых эффективных технологий управления производством [15]. Главный критерий эффективной работы любого предприятия – достижение плановых показателей при минимальных затратах. В данном случае только эффективного производственного процесса явно недостаточно, так как производственная часть тесно связана системой сбыта, снабжения, хранения, конструкторско-технологическими данными. В качестве примера может быть приведено производство, на котором функционируют автоматизированные системы, отвечающие за управление: технологическим процессом (SCADA), оперативное управление производством (MES), а также системы управления деятельностью предприятия (ERP) и автоматизацию процесса проектирования (CAD) [16].

В последнее время под эффективными технологиями управления производством имеют в виду системы класса MES, связывающие воедино технологический и бизнес уровни управления предприятием в единый информационный комплекс, решая при этом множество важнейших для промышленного предприятия задач. Однако максимальный эффект от использования всех автоматизированных систем, присутствующих на предприятии, можно получить только при создании единого информационного пространства (ЕИП), с помощью которого все перечисленные системы могут оперативно и своевременно обмениваться информацией. Сложно представить эффективную автоматизацию производственного процесса без учета планирования обеспечения материалами и комплектующими, работы склада и многих других бизнес-процессов, которые неразрывно связаны друг с другом [16].

MES – автоматизированная система управления и оптимизации производственной деятельности, которая в режиме реального времени: инициирует; отслеживает; оптимизирует; документирует все производственные процессы от начала выполнения заказа до выпуска готовой продукции (определение по стандарту ISA-95) [1].

Согласно международному стандарту ISA-95 MES-система состоит из следующих функций [2], приведенных ниже. Схематичное представление функций MES-системы приведено на рисунке 1.

Контроль состояния и распределение ресурсов (RAS (Resource Allocation and Status)). Управление ресурсами технологического оборудования, сырья, персонала, а также другими объектами, необходимыми для ведения производственной деятельности. Обеспечивает детальную историю ресурсов и гарантирует готовность оборудования к работе. Контролирует состояние ресурсов в режиме реального времени. Управление ресурсами включает резервирование и диспетчеризацию, с целью достижения целей оперативного планирования.

Оперативное/Детальное планирование (ODS (Operations/Detail Scheduling)). Обеспечивает упорядочение производственных заданий, основанное на очередности, атрибутах, характеристиках, связанных с маршрутными картами продукции и технологией производства. Формирует производственное расписание с минимальными перенастройками оборудования и параллельной работой производственных мощностей для уменьшения времени получения готового продукта и времени простоя.

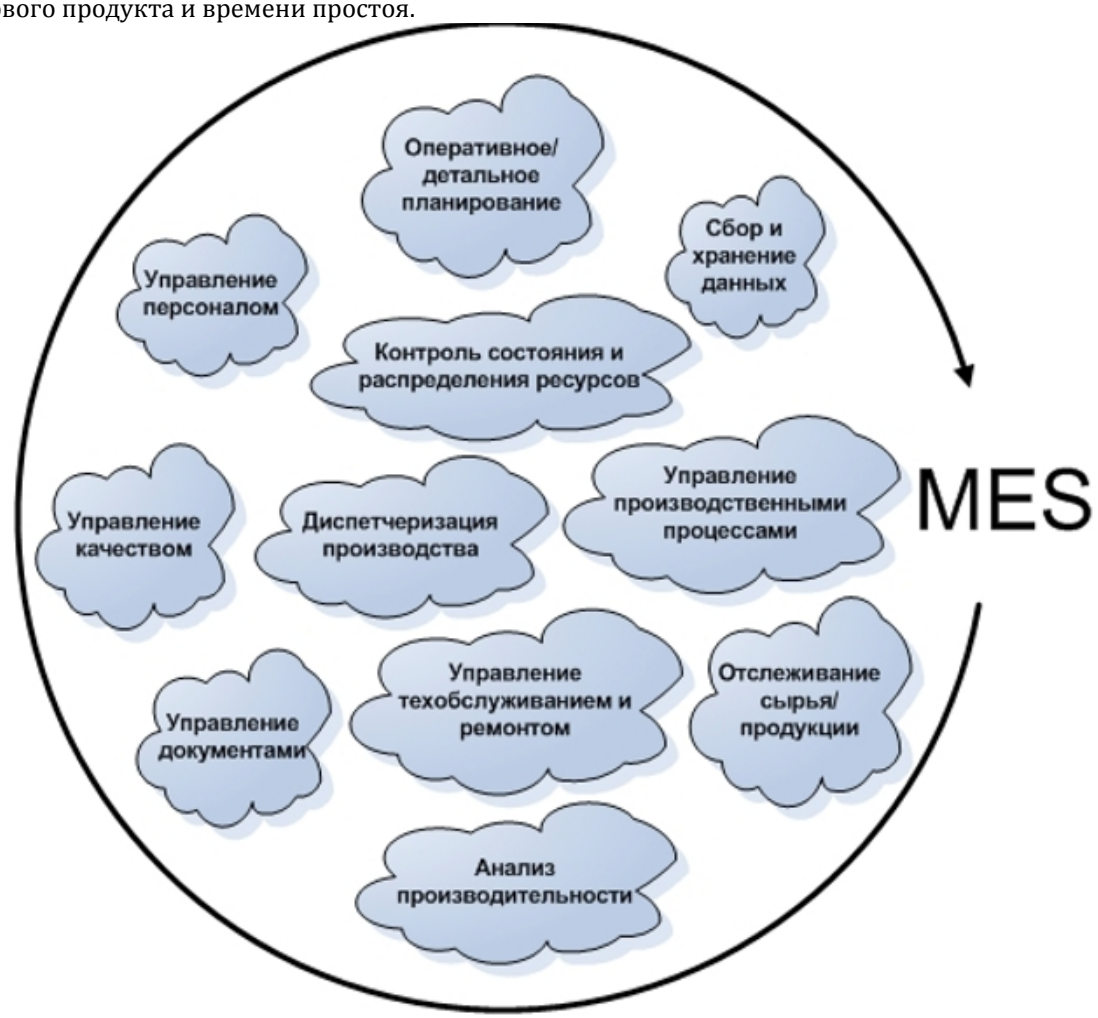


Рис. 1. Подсистемы MES-системы согласно стандарту ISA-95

Управление документами (DOC (Document Control)). Контролирует содержание и прохождение документов, сопровождающих выпускаемую продукцию, включая инструкции и нормативы работ, способы выполнения, рецептуры, процедуры стандартных операций, программы обработки сырья, записи партий продукции, сообщения о технических изменениях, передачу информации от смены к смене, а также обеспечивает возможность вести плановую и отчётную производственную документацию. Также включает инструкции по безопасности, контроль защиты окружающей среды, государственные и необходимые международные стандарты. Хранит историю прохождения и изменения документов.

Диспетчеризация производства (DPU (Dispatching Production Units)). Управляет потоком

сырья в виде заданий, партий. Диспетчерская информация представляется в той последовательности, в которой работа должна быть выполнена, и изменяется в реальном времени по мере возникновения событий на производственной линии. Дает возможность изменения заданного календарного плана на уровне производственных линий. Включает функции устранение брака и переработки отходов.

Сбор и хранение данных (DCA (Data Collection/Acquisition)). Взаимодействие информационных подсистем ERP, MES, WMS-уровней в целях получения, накопления и передачи технологических и управляющих данных, циркулирующих в производственной среде предприятия. Функция обеспечивает интерфейс для получения данных и параметров технологических операций, которые используются в формах и документах, прикрепляемых к единице продукции.

Управление персоналом (LM (Labor Management)). Обеспечивает получение информации о состоянии персонала и управление им в требуемом масштабе времени. Включает отчетность по присутствию и рабочему времени, отслеживание сертификации, возможность отслеживания непроизводственной деятельности, такой, как подготовка материалов или инструментальные работы, в качестве основы для учета затрат по видам деятельности. Возможно взаимодействие с функцией распределения ресурсов, для формирования оптимальных заданий.

Управление качеством (QM (Quality Management)). Обеспечивает анализ в реальном времени измеряемых показателей, полученных от производства, для гарантированно правильного управления качеством продукции и определения проблем, требующих вмешательства обслуживающего персонала. Формирует рекомендации по устранению проблем, определяет причины брака путём анализа взаимосвязи симптомов, действий персонала и результатов этих действий. Управляет выполнением лабораторных исследований параметров продукции.

Управление производственными процессами (PM (Process Management)). Отслеживает производственный процесс и либо корректирует автоматически, либо обеспечивает поддержку принятия решений оператором для выполнения корректирующих действий и усовершенствования производственной деятельности. Эта деятельность может быть, как внутриоперационной и направленной исключительно на отслеживаемые и управляемое оборудование, так и межоперационной, отслеживающей ход процесса от одной операции к другой. Включает управление тревогами для обеспечения гарантированного уведомления персонала об изменениях в процессе, выходящих за приемлемые пределы устойчивости.

Управление техобслуживанием и ремонтом (MM (Maintenance Management)). Отслеживает и управляет обслуживанием оборудования и инструментов. Обеспечивает их работоспособность. Обеспечивает планирование периодического и предупредительного ремонтов, ремонта по состоянию. Накапливает и хранит историю произошедших событий (отказы, уменьшение производительности и др.) для использования в диагностировании возникших и предупреждения возможных проблем.

Отслеживание сырья/продукции (PTG (Product Tracking and Genealogy)). Обеспечивает возможность получения информации о состоянии и местоположении партии в каждый момент времени. Информация о состоянии может включать данные о том, кто выполняет задачу, сырье и их поставщиках, текущих условиях производства, а также любые тревоги, данные о повторной обработке и другие события, относящиеся к продукту.

Анализ производительности (PA (Performance Analysis)). Обеспечивает формирование отчетов о фактических результатах производственной деятельности, сравнение их с историческими данными и ожидаемым коммерческим результатом. Результаты производственной деятельности включают такие показатели, как коэффициент использования ресурсов, доступность ресурсов, время цикла для единицы продукции, соответствие плану и соответствие стандартам функционирования. Систематизирует информацию, полученную от разных функций, измеряющих производственные параметры. Эти результаты могут быть подготовлены в форме отчета или представлены в реальном времени в виде текущей оценки эксплуатационных показателей.

Любое производство начинается с планирования. Рассмотрим подробнее функции планирования в MES-системах [14]:

Планирование выпуска продукции в соответствии с планами продаж, проектами, заказами.

Формирование производственных планов цехов по всей технологической цепочке создания продукции.

Планирование потребностей в материалах, сырье, комплектующих, рабочей силе,

оборудовании.

Расчет плановых затрат на производство продукции и ее рентабельности.

Учет фактических затрат на производство продукции и калькуляция ее фактической себестоимости.

План/факт анализ производственной деятельности предприятия.

Для реализации выделенных функций планирования производства, в MES-системах используются различные виды искусственного интеллекта. Рассмотрим их подробнее.

Искусственный интеллект в планировании

При планировании производства искусственный интеллект позволяет прогнозировать на некотором обобщенном уровне, например, на уровне группы номенклатуры и потом разбивать обобщенный прогноз до единичного уровня, используя дополнительные расчеты.

Табуированный поиск представляет собой вариант метода градиентного спуска с памятью. В процессе поиска ведётся список табуированных (запрещённых для перехода) позиций из числа уже рассчитанных. Критическими параметрами алгоритма является диапазон запретов. В процессе поиска осуществляются операции включения в запрещённый список состояний вокруг текущего состояния, что добавляет фактор случайности в процесс поиска [3].

С целью уменьшения времени ожидания обработки для всех деталей или узлов заказа, простоев оборудования, а также сокращения времени всего производственного цикла выполнения заказа решение задачи оптимизации производственного процесса можно выполнять в соответствии со следующими критериями:

1. Минимизация времени ожидания обслуживания, т. е. обеспечение комплектного выпуска деталей, улучшение структуры штучно-калькуляционного времени:

$$K_1 = \min_i \left(\sum_{i \in I_S} W_{ij} \right), \quad (1)$$

где

W_{ij} - интервал времени между окончанием (j-1)-й и началом j-й операции i -ой детали;

I_s - множество деталей заказа s.

2. Минимизация простоев оборудования (максимальная загрузка):

$$K_2 = \min \left(q_{ik} - \sum_{i \in Is} F_{iks} \right), \quad (2)$$

где

q_{ik} - трудоемкость операции i-той детали на k-том станке,

$F_1=(W_{ij}+p_i)$ - длительность прохождения i-той детали в системе, включая время простоя;

$p_i = \sum_{j \in G_i} p_i$ - общая длительность всех операций i-той детали;

p_{ij} - длительность выполнения операции;

Gi - множество операций i-той детали.

3. Минимизация времени выполнения всех работ по комплекту деталей:

$$K_3 = \min_s \left(\sum_{i \in Is} R_{ik} - \sum_{i \in Is} F_i \right), \quad (3)$$

где R_{ik} - объем работ, который необходимо выполнить по i-той детали заказа s.

Муравьиный алгоритм позволяет решить задачу минимизации переналадки и простоев оборудования при большом числе станков [3]. Процесс переналадки занимает важное место в системе планирования, так как он занимает значительную часть общего календарного времени (от нескольких часов до целой смены). Чем чаще требуется переналадка (по условиям производства), тем больше оказываются потери времени. Поэтому одной из основных задач является совершенствование систем переналадки оборудования, а также использование методов, которые позволяют получить оптимальную последовательность обработки деталей на станках с минимальными потерями времени на переналадку. Муравьиный алгоритм позволяет найти оптимальный производственный маршрут при заданный ограничениях. В проведенном исследовании [10] процесса нахождения оптимального маршрута муравьиным алгоритмом были получены данные, по которым из 100 итераций «муравьи» нашли 3 раза оптимальные маршруты. Проведенные компьютерные эксперименты показывают, что муравьиные алгоритмы находят хорошие маршруты значительно быстрее, чем точные методы комбинаторной оптимизации. Эффективность муравьиных алгоритмов увеличивается с ростом размерности задачи оптимизации [10]. На рисунке 2 представлена логика муравьиных алгоритмов [6].

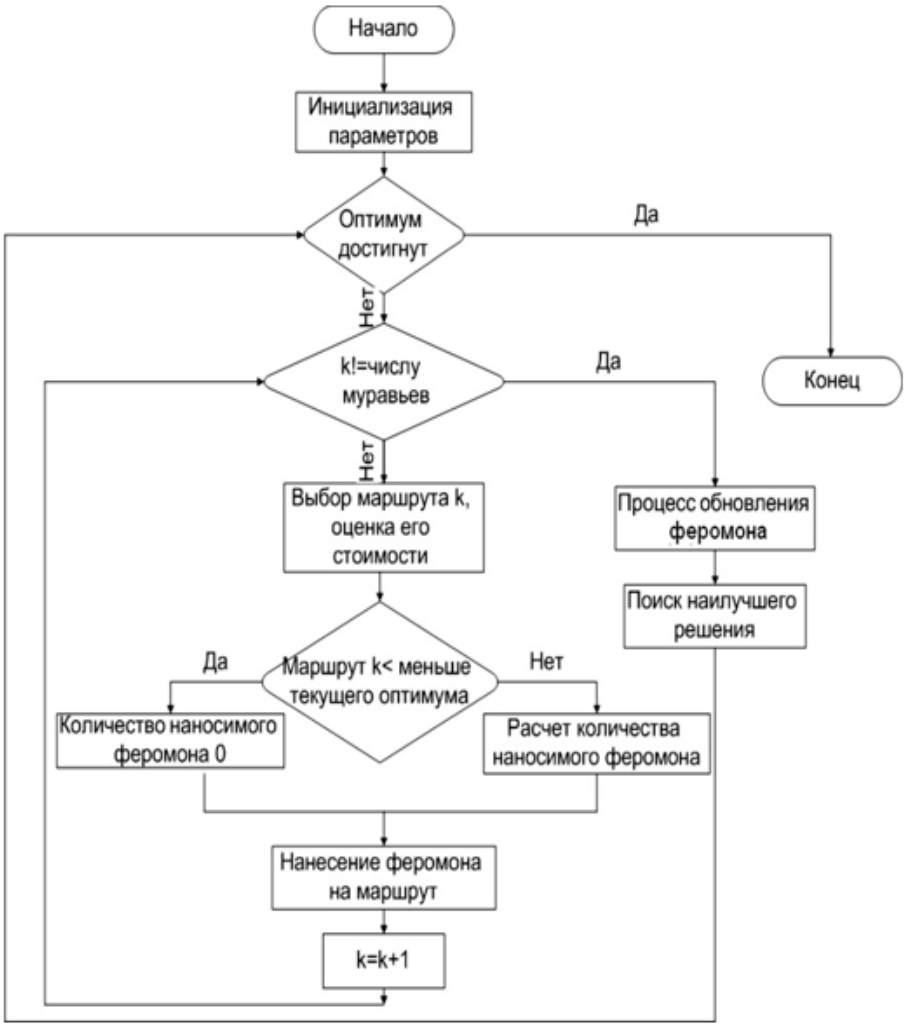


Рис. 2. Логика муравьиных алгоритмов

Генетические алгоритмы

Если взять за основу модифицированный генетический алгоритм, то эффективность применения генетических алгоритмов зависит от того, насколько целесообразно осуществляется кодирование решения. В качестве генетических операторов используются: селекции; репродукции; кроссинговера; мутации; миграции [7]. Традиционно специфические ограничения в поисковой задаче оптимизации закладывали в виде штрафных функций, которые учитывали при расчете значения целевой функции (функции пригодности) соответствующей особи.

Наложение некоторого «штрафа» на особь означает понижение значения ее целевой функции в случае, если решение, которое представляет собой конкретная особь, выходит за границы области допустимых решений. Таким образом, вся специфика решаемой оптимизационной задачи учитывалась именно в блоке расчета функции пригодности особи, и, как следствие, в операторе репродукции. Качество разработки штрафных функций, а именно чувствительность алгоритма к изменениям, напрямую влияет на скорость сходимости и получения решения, удовлетворяющего ограничениям [7].

Помимо использования штрафных функций предлагается модифицировать логику базовых операторов генетического поиска. Идея модификации сводится к применению правил, которыми руководствуются эксперты в предметной области при построении расписания. Суть этих правил состоит в направленной корректировке отдельных параметров производственных заданий с целью разрешения коллизий, возникающих при нарушении ограничений, обусловленных спецификой задачи. Модификация предполагает изменение базовых операторов мутации и кроссинговера.

В частности, для оператора кроссинговера применяется специализированное правило

выбора аллелей при реализации скрещивания хромосом. Для визуализации модифицированной логики представлен упрощенный пример, в рамках которого рассматриваются 5 работ, 3 альтернативные производственные линии. Выбраны 2 хромосомы, представляющие собой различные решения, а именно, варианты назначения и последовательность выполнения работ производственными линиями. Каждый прямоугольник помечен индексом соответствующей работы. Прямоугольники расположены по горизонтали вдоль линий, определяющих ту или иную производственную линию. Таким образом, получаем вариацию диаграммы Ганта, зачастую, используемую для визуализации расписаний.

Для каждой производственной линии образуются пары работ, по одной из каждой особи. Для каждой пары случайным образом определяем будущую принадлежность соответствующему потомку. При этом выполняется проверка на дублирование работ в одном и том же решении.

Итог скрещивания – получены новые решения, заведомо исключающие нарушения некоторых ограничений. Таким образом, в логику базового генетического оператора добавляется фактор предопределенности, исключающий потенциальные решения, то есть потомки, которые заведомо не удовлетворяют ограничениям.

Нейронные сети

Нейронная сеть представляют собой систему соединённых и взаимодействующих между собой простых процессоров (нейронов). Структурная схема нейронной сети представлена на рисунке 3, где зеленый цвет – входной слой нейронов, синий – скрытый(промежуточный) слой нейронов, желтый – выходной слой нейронов.

Нейрон – базовый элемент нейронной сети, единичный простой вычислительный процессор способный воспринимать, преобразовывать и распространять сигналы, в свою очередь объединение большого количества нейронов в одну сеть позволяет решать достаточно сложные задачи.

Нейронные сети - это очень мощный и гибкий механизм планирования и прогнозирования. При определении того, что нужно планировать, необходимо указывать переменные, которые анализируются и предсказываются [13]. Здесь очень важен требуемый уровень детализации. На используемый уровень детализации влияет множество факторов: доступность и точность данных, стоимость анализа и предпочтения пользователей результатов планирования. В ситуациях, когда наилучший набор переменных неясен, можно попробовать разные альтернативы и выбрать один из вариантов, дающий наилучшие результаты. Обычно так осуществляется выбор при разработке систем планирования, основанных на анализе исторических данных.

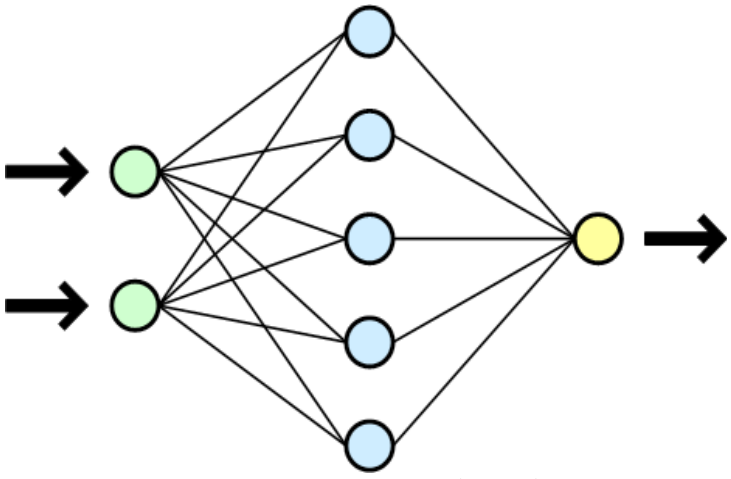


Рис. 3. Структурная схема нейронной сети

Второй важный этап при построении нейросетевой системы планирования – это определение следующих трех параметров: периода планирования, горизонта планирования и интервала планирования.

Период планирования - это основная единица времени, на которую делается прогноз. Горизонт планирования - это число периодов в будущем, которые покрывает прогноз. То есть, может понадобиться прогноз на 10 дней вперед, с данными на каждый день. В этом случае период - сутки, а горизонт - 10 суток. Наконец, интервал планирования – частота, с которой делается новый прогноз. Часто интервал планирования совпадает с периодом планирования. Выбор

периода и горизонта планирования обычно диктуется условиями принятия решений в области, для которой производится прогноз. Выбор этих двух параметров – едва не самое трудное в нейросетевом планировании. Для того чтобы планирование имело смысл, горизонт планирования должен быть не меньше, чем время, необходимое для реализации решения, принятого на основе прогноза. Таким образом, планирование очень сильно зависит от природы принимаемого решения. В некоторых случаях, время, требуемое на реализацию решения, не определено, например, как в случае поставки запасных частей для пополнения запасов ремонтных предприятий. Существуют методы работы в условиях подобной неопределенности, но они повышают вариацию ошибки планирования. Поскольку с увеличением горизонта планирования точность прогноза, обычно, снижается, часто можно улучшить процесс принятия решения, уменьшив время, необходимое на реализацию решения и, следовательно, уменьшив горизонт и ошибку планирования.

В некоторых случаях не так важно предсказание конкретных значений планируемой переменной, как предсказание значительных изменений в ее поведении. Такая задача возникает, например, при предсказании момента, когда текущее направление движения рынка (тренд) изменит свое направление на противоположное.

Выводы

Применение методов и алгоритмов искусственного интеллекта расширяют возможности систем планирования в контексте формализации экспертных правил, позволяющих учитывать специфику технологических и производственных процессов. Искусственный интеллект позволит существенно улучшить систему оперативного планирования, тем самым сократив время получения оптимальных или приемлемых производственных расписаний. При появлении случайных событий, влияющих на процесс производства, позволит быстро реагировать на изменение и внесение корректив в исходные данные. Возможно также комбинирование методов и алгоритмов искусственного планирования при реализации функций планирования в системах управления производством.

Литература

1. ANSI/ISA-95.00.01-2000, Enterprise-Control System Integration.
2. MES (Управление производством). 2014. - URL: http://www.omega-soft.ru/mes (дата обращения 01.10.2015).
3. Анализ современных методов искусственного интеллекта применительно к задачам календарного планирования единичного производства. 2014. – URL. http://masters.donntu.org/2008/kita/khaustova/library/5.htm (дата обращения 01.10.2015).
4. Бурмистров К.В., Цуприк Л.С., Бурмистрова И.С., Ошурков В.А. Особенности проектирования mes и erp-систем на горнодобывающих предприятиях//Научные труды SWorld. 2014. Т. 9. № 4. С. 94-99.
5. Доронин В.Ю., Волщуков Ю.Н., Макашов П.Л., Романенко А.В., Ишметьев Е.Н., Леднов А.В., Макашова В.Н. Построение системы диспетчеризации и контроля технологических процессов как элемента управления промышленным предприятием//Управление большими системами. Москва, 2011. С. 116-119.
6. Еременко Ю.И., Цуканов М.А., Соловьев А.Ю. О применении мультиагентных алгоритмов муравьиных колоний для решения задачи структурной оптимизации в энергетических системах // Фундаментальные исследования. – 2013. – № 10–15. – С. 3316-3320; URL: www.rae.ru/fs/?section=content&op=show_article&article_id=10002326 (дата обращения: 08.10.2015).
7. Интегрированные интеллектуальные системы оперативного планирования производства. – 2011. URL: http://izv-tn.tti.sfedu.ru/wp-content/uploads/2011/7/8.pdf (дата обращения 01.10.2015).
8. Ишметьев Е.Н., Романенко А.В., Волщуков Ю.Н., Марков П.Е. Интеграция как катализатор развития систем класса MES//Автоматизация в промышленности. 2013. № 8. С. 47-50.
9. Майорова Е.С., Ошурков В.А., Бубер М.Г. Концепция технического и коммерческого энергоучета промышленных предприятий//Актуальные проблемы современной науки, техники и образования. 2015. Т. 2. № 1. С. 135-138.
10. Муравьиные алгоритмы. 2014. – URL. http://www.serhiy-shtovba.narod.ru/doc/Shtovba_Ant_Algorithms_ExponentaPro_2003_3.pdf (дата обращения 01.10.2015).
11. Ошурков В.А., Цуприк Л.С., Майорова Е.С., Бурмистров К.В., Бурмистрова И.С. Разработка системы управления технологическим процессом многономенклатурного машиностроительного производства в условиях применения станков с ЧПУ//Машиностроение: сетевой электронный научный журнал. 2015. Т. 3. № 3. С. 53-55.
12. Ошурков В.А., Чернова Е.В. Метод выделения сегмента на рынке автоматизации производства//Научные труды SWorld. 2013. Т. 30. № 4. С. 84-90.
13. Прогнозирование с помощью нейронных сетей. – 2015. URL: http://apsheronsk.bozo.ru/Neural/Lec9.htm (дата обращения: 08.10.2015).
14. Управление производством. 2014. - URL:http://www.parus.com/products/system/production/ (дата обращения 01.10.2015).
15. Чусавитина Г.Н., Макашова В.Н. и др. Повышение конкурентоспособности выпускников ИТ-специальностей вуза в условиях монопромышленного города. Монография. Под ред. Г.Н. Чусавитиной. -Магнитогорск: МаГУ, 2010. -158 с
16. Эффективные системы управления производством. 2015. - URL: http://www.cfin.ru/itm/kis/choose/Manufacturing.shtml (дата обращения 01.10.2015).

Сапегина В.С.[1], Куренев А.В.[2]

[1]Магнитогорский государственный технический университет, г.Магнитогорск, студентка кафедры прикладной информатики Института энергетики и автоматизированных систем ФГБОУ ВПО «МГТУ им. Г.И.Носова», v.sapegina@its74.ru

[2] г.Магнитогорск, директор ООО «ККМ02», kurenev@gmail.com

СОЗДАНИЕ ЭФФЕКТИВНОЙ СИСТЕМЫ ЛОЯЛЬНОСТИ КЛИЕНТОВ ПОСРЕДСТВОМ ВНЕДРЕНИЯ ПЛАТФОРМЫ TRADELINE

КЛЮЧЕВЫЕ СЛОВА

Система лояльности клиентов, внедрение, бонусная программа, бонусы, программная платформа.

АННОТАЦИЯ

Рассмотрено построение системы лояльности клиентов как совокупности различных способов повышения привлекательности торгового предприятия. Приведены требования, предъявляемые к системе лояльности для сети магазинов, реализация которых повышает эффективность системы как на этапе внедрения, так и при последующей эксплуатации. Перечислены автоматизированные решения для создания эффективных систем лояльностей клиентов. Рассмотрена программная платформа TradeLine с представлением успешного опыта внедрения в сеть магазинов.

Введение

Как привлечь потенциальных клиентов? Как удержать уже существующих? Эти вопросы очень актуальны для любой компании на современном рынке. И самый простой способ – организация диалога с клиентами. Другими словами, необходимо повысить интерес клиентов к компании. Для достижения этой цели многие участники рынка успешно используют различные маркетинговые мероприятия, реализация подобные мероприятия возможно с помощью системы лояльности, например, ввести бонусные программы или акции на товары. Но как осуществить подобные мероприятия с помощью автоматизации? В данной статье речь пойдет именно об этом и будет предложено решение данной проблемы.

Система лояльности как средство реализации маркетинговых мероприятий

Система лояльности клиентов представляет собой совокупность различных способов повышения привлекательности торгового предприятия, когда клиент сам заинтересован посетить магазин и сделать покупки. Чем значительнее его лояльность, тем больше желание прийти, купить, порекомендовать и т.д.

Основная задача системы лояльности для подобных предприятий – это увеличение количества новых клиентов, уменьшение количества пассивных клиентов, а также снижение вероятности, что клиенты уйдут к конкурентам. Для этого необходим индивидуальный подход, а также эффективная система поощрений.

Еще одна задача системы лояльности – это получение информации о предпочтениях клиента. Для этого можно использовать электронные анкеты, которые клиент сможет заполнить, к примеру, на сайте компании. Проведение опросов в электронном виде не занимает у клиента много времени. При этом, экономится время самих сотрудников, снимается необходимость печати анкет.

Система лояльности клиентов строится на основе разнообразных программ. Программа лояльности – это комплекс маркетинговых мероприятий для развития повторных продаж существующим клиентам в будущем, для продажи им дополнительных товаров и услуг, продвижения корпоративных идей и ценностей, других видов потенциально прибыльного поведения [1].

Существуют два основных вида маркетинговых мероприятий для привлечения клиентов: направленные на внешнего потребителя и внутренние программы для сотрудников.

Маркетинговые мероприятия, направленные на внешнего потребителя, заключаются в том, чтобы удержать интересы клиента, создать дополнительную коммуникацию, обеспечить

вторичные продажи.

Возможности данных мероприятий для потребителей:

1. Получение информации о клиентах. Перечислим возможности, которые предоставляет полная клиентская база, что впоследствии приводит к увеличению прибыли:

- предоставление разных скидок/цен;
- понимание ценностей целевой аудитории для дальнейшего планирования развития предприятия;
- оповещение о деятельности и мероприятиях предприятия.

2. Другие возможности. Существуют маркетинговые мероприятия – акции, основанные на предоставлении потребителям бонусов в разнообразных формах. Задача такого шага – сделать привлекательным для клиента конкретный продукт или собственное имя с целью увеличения чека или повторной продажи [2].

К сожалению, не все маркетинговые мероприятия одинаково эффективны, к задаче нужно подойти системно, учитывая факторы, которые могут повлиять на конечный результат. В результате, мы получим грамотно построенный комплекс мероприятий, благодаря которому клиент сделает выбор в пользу нашей компании.

Изучение проблемы создания эффективной системы лояльности клиентов позволило выделить ряд факторов, определяющих успешность такого проекта: комплекс маркетинговых мероприятий; программная платформа для создания систем лояльности; развитие мотивации для привлечения клиентов; разработка рекламных материалов и др [14-24].

Графически данные факторы представлены в виде модели Исикавы на рис.3.

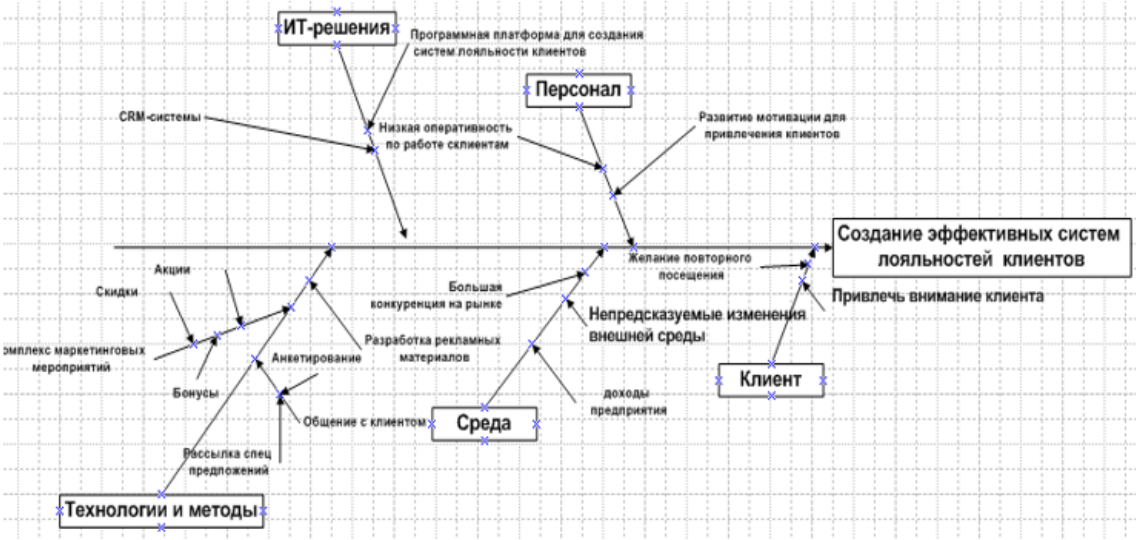


Рис.1. Факторы, влияющие на успешность проекта

Что же подразумевается под грамотно построенным комплексом мероприятий? Это маркетинговые методы, которые при разумных затратах со стороны компании позволят привлечь и удержать клиентов, организовать с ними диалог и тем самым быть впереди конкурентов. Достигнуть такого результата возможно только в случае единого подхода к задаче. С точки зрения маркетинга это можно достичь с помощью систем поощрения (скидки, накопительные баллы и т. д.) и способа общения с клиентом (рассылка специальных предложений, анкетирование и т. д.).

С точки зрения автоматизации нужно сделать выбор программной платформы для создания систем лояльностей клиентов, при помощи, которой будет осуществлен маркетинговый план [3].

В подобных случаях, большинство участников рынка используют на своих предприятиях готовые ИТ-решения в виде программных платформ для создания систем лояльности клиентов.

Программная платформа представляет собой программный продукт, имеющий возможности гибкой настройки и модульность.

Для осуществления оптимального выбора необходимо определить масштабы предприятия. Это может быть небольшой магазин, сеть супермаркетов или предприятие и сеть смешанного типа. Рассмотрим сеть супермаркетов и рассмотрим, какие требования предъявляются к системам лояльности в подобных предприятиях.

Сеть супермаркетов – это предприятия с распределенной структурой. Большое количество посетителей и территориально распределенная структура – это основные факторы, которые необходимо учитывать при проектировании системы лояльности для таких предприятий. Важно чтобы клиент, получив карту лояльности в одном магазине сети, мог также воспользоваться ею и в других магазинах. Для этого необходим доступ к единой базе клиентов для всех магазинов сети, а также централизованное управление системой лояльности. Постоянно растущее количество клиентов требует оперативной обработки данных, необходимых для ведения диалога с покупателем.

Для сетей магазинов наиболее актуальна комбинированная система лояльности (бонусы и акции на товары). Бонусная система предполагает начисление бонусов на счет клиента. Бонусы, по сути, представляют собой абсолютную скидку, которой клиент сможет воспользоваться при последующих посещениях. Каждый раз, когда клиент приходит, он получает на свой счет некоторую сумму, воспользоваться которой он сможет в зависимости от условий начисления бонусов. Варианты начисления бонусов могут быть различными. Например, можно начислять бонусы только в случае, если сумма чека превышает порог, заданный в настройках системы лояльности или, если в чеке есть определенный товар. Стоимость бонуса устанавливается в настройках системы лояльности относительно курса к денежной единице (например, 1 бонус = 1 рубль). Бонусы – это очень гибкий механизм, позволяющий выстроить систему поощрений, соответствующую самым разнообразным требованиям [3].

Помимо бонусов, в сетях супермаркетов целесообразно использовать акции по товарам. Однако недостаточно просто проводить акции, необходимо также информировать клиентов об их проведении, при этом информировать не всех покупателей, а именно тех, кого предстоящие акции могут заинтересовать, то есть, необходимо сохранять индивидуальный подход в работе с клиентом. Для этой цели необходимо иметь возможность анализировать данные о покупках клиентов, а также возможность формировать целевую рассылку информационных сообщений по результатам проведенного анализа [3].

Для предприятий с распределенной структурой актуальны такие механизмы системы лояльности клиентов, как: бонусная система поощрений; акции по товарам; анкетирование клиентов, с целью исследования потребностей и пожеланий клиентов; рассылка целевых электронных сообщений; печать на чеке информации важной для покупателя; мотивация клиентов к приглашению новых покупателей; при расчете на кассе, дополнительно печатать информацию о новых товарах, в продвижении которых заинтересован супермаркет, основываясь на индивидуальных интересах покупателей.

Вышеперечисленные механизмы позволят расширять клиентскую базу и поддержать заинтересованность у существующих покупателей [3].

Перечислим требования, предъявляемые к системе лояльности для сети магазинов:

- карта клиента должна быть доступна для использования в любом магазине сети;
- открытый доступ к базе данных клиентов для сотрудников и руководителей сети;
- возможность централизованного управления системой лояльности;
- нечувствительность системы лояльности к внешним техническим условиям / проблемам;
- возможность увеличивать размеры сети;
- учитывать особенности работы регионально распределенных сетей:
- надежность и безотказность в работе системы;
- высокая производительность системы при больших объемах данных [3].

Если система лояльности соответствует вышеперечисленным критериям, то вероятность возникновения ошибок невелика как в момент внедрения, так и при последующей эксплуатации системы.

Программные платформы для создания систем лояльностей клиентов

Рассмотрим наиболее распространенные программные платформы, которые являются автоматизированными решениями, разработанными для создания эффективных систем лояльностей клиентов:

1. Платформа Plas-Tek® Loyalty™ выступает единым процессинговым центром, куда стекается информация о транзакциях. Служит общим хранилищем полной информации о клиенте, предоставляет необходимые инструменты для построения эффективных взаимоотношений с клиентом и формирования его лояльности. [5];
2. Платформа RapidSoft предназначена для автоматизации систем лояльности с использованием персональных клиентских пластиковых платежных и неплатежных

карт [6];

3. Платформа SailPlay используется для запуска системы лояльности. В основе работы платформы находится механизм начисления бонусных баллов [7];
4. Платформа Loyax предназначена для управления системами лояльности, поддерживает множество готовых функций и инструментов [8];
5. Платформа Starcard рассчитана на создание программ лояльности [9];
6. Платформа Scan&Play позволяет сохранить текущих клиентов и добавлять новых [10];
7. Платформа RSB+ позиционируется как инновационный и гибкий инструмент для запуска маркетинговых кампаний при помощи эквайрингового оборудования Банка Русский Стандарт [11];
8. Платформа лояльности iReward24 Loyalty Platform является комплексным, модульным решением, дающим возможность построить лояльность клиентов и управлять ею [12];
9. Платформа U-Solutions. Ключевым отличием и преимуществом U-Solutions является инструментарий электронных денег [13];
10. Платформа TradeLine направлена на построение эффективных систем лояльности покупателей [4].

В данной статье пойдет речь о создании эффективных систем лояльности клиентов посредством внедрения программной платформы TradeLine.

Характеристика программной платформы TradeLine

Разработчиком программной платформы TradeLine является компания ООО «ККМ02». Программная платформа TradeLine представляет собой комплексное решение для создания системы повышения лояльности покупателей и обеспечивает решение таких вопросов, как административное обслуживание, информационная безопасность, масштабирование. Основной сферой применения платформы являются торговые предприятия.

TradeLine позволяет выполнять комплекс функций управления системой взаимоотношений с клиентами, таких как: управление дисконтными ресурсами и схемами; управление коммуникациями с покупателями; объединение и аналитика информационных ресурсов.

Виды информационных ресурсов, управляемых платформой: данные о покупателях; товаро-учетная информация; информация о кассовых транзакциях; данные о маркетинговых мероприятиях; организационно-административная информация.

Платформа TradeLine обладает таким важным свойством, как масштабируемость: по спектру решаемых задач, по числу пользователей, по объемам хранящейся информации, по интеграционным связям с внешними информационными системами, что делает данное автоматизированное решение универсальным.

Кроме того, к преимуществам TradeLine можно отнести: использование доступных защищенных протоколов обмена данными, простой и понятный api-интерфейс, интеграцию с популярным кассовым программным обеспечением и решениями на платформе "1С: Предприятие", работу в виде «облачного» сервиса, а также простое управление сервисом.

Архитектура спроектирована так, чтобы максимально соответствовать аспектам функционирования, безопасности, современным требованиям по разработке и сопровождению систем торговых предприятий любого масштаба. В ее основе лежат управляющие сервисы, обеспечивающие взаимодействие с несколькими базами данных и структурами других целевых решений. Взаимодействие клиентской и серверной частями происходит через сервер приложений, на котором реализована бизнес-логика. Архитектура платформы включает в себя уровень ввода и вывода информации, уровень приложений и уровень хранения данных.

Архитектура обеспечивает:

- защиту данных от прямого доступа за счет наличия службы приложений;
- рост производительности и большую скорость доступа и обработки информации за счет использования трехзвенной архитектуры системы;
- возможность оперативного переконфигурирования системы, за счет изолированности уровней друг от друга;
- репликацию данных в распределенной системе;
- доступ по протоколу HTTP (толстый и тонкий клиенты);
- интеграцию данных в информационные системы различного уровня и функционала;
- масштабируемость по числу баз данных и сервисов.

Информационная безопасность, которая обеспечивается данной платформой, осуществляется с помощью средств логирования операций, разграничения прав доступа, мониторинга работы модулей системы и управления данными.

Платформа обладает следующими технологическими параметрами:

- интеграцией с системами и приложениями с возможностью обмена данными по протоколу HTTP;
- интеграцией с API внешних сервисов – sms, e-mail провайдер;
- поддержкой ОС – MS Windows, Linux, MacOS;
- поддержкой СУБД – Microsoft SQL Server, MySQL, Postgresql.

Компания ООО «ККМ02» имеет опыт успешных внедрений программной платформы TradeLine.

Внедрение программной платформы TradeLine в сеть магазинов «Альбатрос»

Существует большое количество методологий по внедрению программных решений (Oracle Unified Method, On Target, MBS, Microsoft Solutions Framework, Microsoft Dynamics Sure Step и др).

Под методологией внедрения программной платформы компания ООО «ККМ02» понимает совокупность наиболее важных принципов, обеспечивающих максимально высокое качество проектных работ при заданных заказчиком ограничениях на временные и финансовые ресурсы.

Эти принципы в компании ООО «ККМ02» формулируются следующим образом:

1. В результате внедрения системы лояльности должны быть полностью или частично удовлетворены основные интересы заказчика;
2. Проект должен быть реализован в установленные заказчиком сроки;
3. Проект должен быть реализован с минимальными для заказчика затратами;
4. Успешность проекта обеспечивается соблюдением определенных условий, как со стороны исполнителя, так и со стороны заказчика.

Используя универсальную методологию Oracle Unified Method, компания ООО «ККМ02» успешно выполнила внедрение программной платформы TradeLine в сеть магазинов «Альбатрос».

Сеть магазинов «Альбатрос» – это активно развивающаяся торговая сеть, магазины которой ежедневно посещают тысячи клиентов. Торговое предприятие «Альбатрос» обладает большими масштабами, в сети находится более двадцати магазинов.

В методологию Oracle Unified Method входят следующие фазы внедрения: начальная фаза (Inception); проектирование системы (Elaboration); разработка системы (Construction); переход к эксплуатации системы (Transition); эксплуатация системы (Production).

Данная методология включает в себя ряд задач: управление проектом; бизнес-требования; анализ требований; анализ; проектирование; разработка; тестирование; контроль производительности; техническая архитектура; конвертация и перенос данных; документация; управление изменениями; обучение; переход к эксплуатации; эксплуатация и поддержка.

Руководствуясь, кроме того, ГОСТ 34.601.90 (ИТ. Автоматизированные системы. Стадии создания), компания ООО «ККМ02» подготовила сеть магазинов «Альбатрос» к вводу программной платформы TradeLine в действие. Сотрудники компании обучили персонал с помощью руководств по эксплуатации. Были проведены строительно-монтажные работы: сооружение кабельных каналов; выполнение работ по монтажу технических средств и линий связи; испытание смонтированных технических средств; сдача технических средств для проведения пусконаладочных работ. Сотрудники ООО «ККМ02» провели пусконаладочные работы, в том числе, автономную наладку технических и программных средств, загрузку информации в базу данных и проверку системы её ведения; комплексную наладку всех средств системы. После успешных предварительных испытаний, т.е. проверки на работоспособность и соответствие техническому заданию, был оформлен акт о приёмке программной платформы TradeLine в опытную эксплуатацию.

После внедрения программной платформы TradeLine в рассматриваемой сети была реализована система лояльности клиентов в виде бонусной системы. Внедрив в сеть магазинов «Альбатрос» данную программную платформу для создания систем лояльностей клиентов, торговое предприятие получило экономию на скидке. Существенно возросло количество клиентов, которым нравится личная экономия. С помощью бонусных карт посетитель может расплатиться бонусами, находящимися на карте, при этом, не потратив наличные или безналичные средства.

Выводы

Программная платформа TradeLine обеспечивает создание и поддержку систем лояльности клиентов различных торговых схем, идентификацию и ведение счетов клиентов, формируя широкую область применения (торговые предприятия, индустрия развлечений, продажа билетов, сфера услуг, медицина). Опыт успешных внедрений в сеть магазинов является основанием для дальнейшего расширения ее функционала и продвижения на рынке.

Литература

1. Программа лояльности [Электронный ресурс]. – 2010 – Режим доступа: https://ru.wikipedia.org/
2. Возможности систем лояльности клиентов и персонала [Электронный ресурс]. – 2012 – Режим доступа: http://moneymakerfactory.ru/articles/sistem-lojalnosti-klientov-i-personala/
3. Система лояльности – путь к сердцу клиента [Электронный ресурс]. – 2010 – Режим доступа: http://s-m.com.ua/txt/item125.html
4. TradeLine [Электронный ресурс]. – 2015 – Режим доступа: http://tlsys.ru/
5. Платформа Plas-Tek® Loyalty™. [Электронный ресурс]. – 2012 –Режим доступа: http://www.plas-tek.ru/ru/plas-tek-loyalty/0/loyalty-platform-plas-tek
6. RapidSoft: Система лояльности. [Электронный ресурс]. – 2012 –Режим доступа: http://www.tadviser.ru/index
7. SailPlay – платформа для запуска программы лояльности [Электронный ресурс]. – 2010 – Режим доступа: http://sailplay.ru/loyalty/
8. Loyax – платформа управления лояльности. [Электронный ресурс]. – 2011 – Режим доступа: http://loyax.com.ru/features.html
9. Starcard – платформа для создания программ лояльности. [Электронный ресурс]. – 2015 – Режим доступа: http://promo.scard.ru/#
10. Платформа лояльности Scan&Play [Электронный ресурс]. – 2010 –Режим доступа: http://scanandplay.ru
11. Платформа RSB+ [Электронный ресурс]. – 2013 – Режим доступа: http://acquiring.ru/rsb-plus/
12. iReward24 – программа лояльности. [Электронный ресурс]. – 2012 – Режим доступа: http://ireward24.ru/predlozhenie/instrumenty-it/platforma-loyalnoci
13. Платформа U-Solutions. [Электронный ресурс]. – 2010 – Режим доступа: http://u-solutions.com.ua/
14. Гаврилова И.В., Новикова Т.Б., Назарова О.Б., Петеляк В.Е., Агдавлетова А.М. Совершенствование методов построения причинно-следственных диаграмм для решения задач управления в организационных системах / И.В. Гаврилова, Т.Б. Новикова, О.Б. Назарова, В.Е. Петеляк, А.М. Агдавлетова. - Фундаментальные исследования, №8 (часть 2), 2015. - с. 247-251
15. Давлеткиреева Л.З., Новикова Т.Б., Курзаева Л.В., Лактионова Ю.С., Подкользина Л.В. Управление технологической архитектурой организации на основе усовершенствованного метода Enterprise Architecture Planning / Л.З. Давлеткиреева, Т.Б. Новикова, Л.В. Курзаева, Ю.С. Лактионова, Л.В. Подкользина. - Фундаментальные исследования, №8 (часть 2), 2015. - с. 252-256
16. Курзаева Л.В., Новикова Т.Б., Давлеткиреева Л.З., Назарова О.Б., Белоусова И.Д.. Совершенствование методики построения моделей событийно-управляемого процесса для постановки задач управления в социальных и экономических системах / Л.В. Курзаева, Т.Б. Новикова, Л.З. Давлеткиреева, О.Б. Назарова, И.Д. Белоусова. - Фундаментальные исследования, №8 (часть 2), 2015. - с. 297-302
17. Масленникова, О.Е. Что даёт методология для эффективности процесса внедрения корпоративной информационной системы? /О.Е. Масленникова//Теплотехника и информатика в образовании, науке и производстве: сборник докладов IV Всероссийской научно-практической конференции студентов, аспирантов и молодых учёных (ТИМ'2015) с международным участием, посвящённой 95-летию основания кафедры и университета (Екатеринбург, 26–27 марта 2015 г.). – Екатеринбург: УрФУ, 2015. – 379 с. – С.247-251. – Библиограф. с. 250.
18. Назарова, О.Б. Сопровождение корпоративных информационных систем [Текст]: учебник/О.Б. Назарова, Л.З. Давлеткиреева, О.Е. Масленникова, Н.О. Пролозова. -Магнитогорск: МаГУ, 2013. -220 с.
19. Назарова, О.Б., Масленникова, О.Е. Разработка реляционных баз данных с использованием CASE-средства ALL FusionDataModeler: учеб.пособие/О.Б. Назарова, О.Е. Масленникова. -Москва: Изд-во «ФЛИНТА», 2013. -74 с.. -Библиогр.: с. 52. -500 экз.. -ISBN 978-5-9765-1601-4
20. Назарова, О.Б., Масленникова, О.Е., Давлеткиреева, Л.З. Формирование компетенций специалиста в области информационных систем с привлечением вендоров /О.Б. Назарова, О.Е. Масленникова, Л.З. Давлеткиреева//Прикладная информатика. -2013. -№ 2(44). -С. 49-56. -Библиогр.: с.56, ISSN 1993-8314
21. Новикова Т.Б., Курзаева Л.В., Петеляк В.Е., Масленникова О.Е., Белоусова И.Д. Описание управления бизнес-процессами предприятия на основе методологии IDEF0: трудности разработки, рекомендации по совершенствованию построения диаграмм / Т.Б. Новикова, Л.В. Курзаева, В.Е. Петеляк, О.Е. Масленникова, И.Д. Белоусова. - Фундаментальные исследования, №8 (часть 2), 2015. - с. 318-322
22. Петеляк В.Е., Новикова Т.Б., Масленникова О.Е., Махмутова М.В., Агдавлетова А.М. Data Flow Diagramming: особенности построения моделей описания управления потоками данных в организационных системах / В.Е. Петеляк, Т.Б. Новикова, О.Е. Масленникова, М.В. Махмутова, А.М. Агдавлетова. – Фундаментальные исследования, №8 (часть 2), 2015. - с. 323-327
23. Пролозова, Н.О., Назарова, О.Б., Давлеткиреева, Л.З. Анализ стандартов в области сопровождения автоматизированных информационных систем /Н.О. Пролозова, О.Б. Назарова, Л.З. Давлеткиреева//Современные научные исследования и инновации, 2012. -№ 11 (19). -С. 7. -Режим доступа: http://web.snauka.ru/issues/2012/11/18571
24. Требования к выпускной квалификационной работе студентов специальности 080801 «Прикладная информатика (в экономике)» (методические рекомендации) /Назарова О.Б., Масленникова О.Е., Махмутова М.В., Белоусова И.Д., Давлеткиреева Л.З., Попова И.В., Новикова Т.Б., Удотов А.С.//Международный журнал экспериментального образования, 2010. -№ 3. -С. 13-14.

Соколова А.А.

ФГБОУ ВПО «Магнитогорский государственный технический университет им. Г.И. Носова», г. Магнитогорск, студентка 4 курса Института ЭиАС, angsok1@yandex.ru

РАЗРАБОТКА СИСТЕМЫ УПРАВЛЕНИЯ ПРОЦЕССОМ СОЗДАНИЯ ПРОГРАММНОГО ПРОДУКТА

КЛЮЧЕВЫЕ СЛОВА

Система, управление, процесс, программный продукт, Asana, Basecamp, JIRA, Redmine, Trello, Мегаплан, Scram, методология.

АННОТАЦИЯ

В данной статье раскрыто определение системы управления процессом разработки программного продукта и основные варианты такой системы. Рассмотрен вариант системы для компании «ККМ02». Раскрыты основные положения новой системы и описана методология SCRAM, на основе которой разрабатывалась данная система.

Любой трудовой процесс имеет технологию, под которой обычно понимают методы и последовательность выполнения работ для достижения целей деятельности. Разработка, принятие и реализация решений как процесс управленческого труда также имеет определенную технологию. В каждой организации практика осуществления этого процесса имеет свои особенности, определяемые характером и спецификой ее деятельности, организационной структурой, действующей системой коммуникаций, внутренней культурой [1]. Кроме того, для каждого вида управленческих решений есть свои особенности технологии процесса разработки и реализации. Однако для всех видов решений, где бы они ни осуществлялись, существуют и общие работы, методы и последовательность их выполнения [2].

Существуют компании, работники которой не могут физически размещаться в одном помещении и совместно использовать инструментарий для управления требованиями и процессом разработки. Для таких компаний существуют эффективные системы управления процессом разработки [3].

Система реализована, чаще всего, в виде приложения с удобным инструментарием, направленным преимущественно для:

- общения заказчика (пользователей) и разработчиков;
- групповой работы разработчиков над проектом;
- подготовки требований;
- управления проектом (планирования сроков и состава релизов, отслеживания доработок и ошибок);
- подготовки справочной документации;
- обмена артефактами (моделями, кодом и другими документами) проекта;
- трассировки между требованиями, задачами и справочной документацией.

На сегодняшний день существует не малое количество уже готовых систем, которые можно отнести к системам управления процессом разработки программного продукта, с имеющимся различным инструментарием [4]. Рассмотрим некоторые основные примеры таких систем:

Asana – веб-сервис и мобильное приложение для командной работы без использования традиционной электронной почты. Его создали ДастинМосковитц и ДжастинРозенштайн, ранее работавшие в Facebook. Предлагает, как бесплатное использование (в командах до 14 сотрудников), так и платный пакет услуг для больших команд.

Basecamp – инструмент проектного менеджмента на основе веб-интерфейса. Особенность данного инструмента управления проектами заключается в том, что здесь нет ограничения по числу пользователей даже в минимальном тарифном плане.

JIRA – проприетарный продукт для трекинга задач у разработчиков, созданный компанией Atlassian. Выполняет функции баг-трекера и площадки для управления проектами. Название появилось как результат сокращенного японского произношения слова «Годзилла».

Redmine – бесплатный open-source инструмент для управления проектами и баг-трекинга с

поддержкой диаграмм Ганта, визуализацией дедлайнов и прогресса по нескольким проектам одновременно. Написан на RubyonRails, работает как кроссплатформенное решение. У проекта создан онлайн-центр языковой локализации продукта, поддерживаемый волонтерами. Некоторые из основных особенностей Redmine являются: поддержка нескольких проектов, контроль доступа Гибкая роль, гибкая система отслеживания вопрос, диаграмма Ганта и календарь, новости, документы и файлы управления, ленты и уведомления по электронной почте [5].

«Битрикс24» - социальный интранет, включающий в себя, наряду с CRM, систему управления задачами. Создан и поддерживается российской компанией «1С-Битрикс». Работает как облачный сервис, но возможна и поставка коробочной версии. По оценкам компании-разработчика, сервисом «Битрикс24» пользуются свыше 188 тыс. компаний в странах СНГ и за рубежом. В системе можно работать с сервисом в облаке или установить коробочную версию на ваш сервер, чтобы индивидуально настроить бизнес-логику, интерфейс, интегрировать с «1С:ЗУП». Коробочная версия сервиса - это продукт «1С-Битрикс24».

Trello — веб-приложение для управления задачами, разработанноеFogCreekSoftware. В основе его работы лежит японская философия менеджмента «канбан». Все проекты здесь представлены в виде «досок» со списком задач. Интерфейс системы в результате похож на Pinterest, и «карточки» с проектами и задачами прикрепляются к каждому пользователю в команде [6, 13].

«Мегаплан» - облачная система управления бизнесом, включает в себя CRM, ERP и инструменты для постановки и делегирования задач. Предлагает несколько масштабируемых тарифных пакетов, работает как SaaS. Особенность платформы - наличие тарифного конструктора: четких тарифов здесь нет, а итоговая цена зависит от числа сотрудников, подключения инструментов финансового учета и документооборота и желаемой скидки при оплате на год вперед.

К сожалению, не одна из существующих систем не смогла полностью удовлетворить запросам компании ООО «ККМ02». В последствие было принято решение разработать новую систему управления процессом разработки программного продукта, которая должна быть реализована на основе методологии Scram.

Что же такое методология Scram? Scram (Скрам) – это фреймворк для разработки и поддержки функционально сложных продуктов. Данное руководство содержит определение Скрама. Определение состоит из описания ролей, мероприятий, артефактов Скрама, а также правил, связывающих их.

Фреймворк Скрама состоит из Скрам Команд и связанных с ними ролей, мероприятий, артефактов и правил. Каждый элемент фреймворка служит определенной цели и является ключевым для успеха и использования Скрама. Правила Скрама объединяют мероприятия, роли и артефакты, регулируя отношения и взаимодействия между ними. Правила Скрама описаны в данном документе. Существуют различные стратегии использования фреймворкаСкрама, их описание выходит за пределы данного документа. Теория СкрамаСкрам основывается на теории управления эмпирическими процессами или эмпиризме. Эмпиризм утверждает, что знание приходит с опытом, решения принимаются на основании того, что является известным. Скрам использует итеративно-инкрементальный подход для оптимизации прогнозируемости и управления рисками [7, 12].

Скрам предписывает четыре формальные возможности для инспекции и адаптации:

- планирование Спринта;
- ежедневный Скрам;
- обзор Спринта;
- ретроспектива Спринта.

Скрам Команда состоит из Владельца Продукта, Команды Разработки и Скрам Мастера. Скрам Команды являются самоорганизующимися и кросс-функциональными. Самоорганизующиеся команды сами выбирают, как лучше выполнить работу и не ждут указаний от людей, не входящих в их состав. Кросс-функциональные команды имеют все необходимые навыки, чтобы выполнять работу и не зависеть от тех, кто не является частью Команды [8, 11].

Повторимся, руководство компании ООО «ККМ02» приняло решение разрабатывать новую систему управления процессом создания программного продукта. На данный момент она необходима прежде всего для улучшения самого крупного проекта компании – системы лояльности TrideLine.

Компании ООО «ККМ02» (Кассовые системы) на рынке автоматизации более 10 лет.

Основное направление деятельности - автоматизация предприятий торговли, общественного питания и индустрии развлечений. Компания поставляет надежное оборудование и программное обеспечение для автоматизации бизнес-процессов, оказывает полный спектр услуг - от установки и настройки оборудования, до комплексного внедрения информационных систем управления предприятием. Основным большим проектом компании является проект TrideLine.В основе систем управления розничной торговлей и общепитом лежат высокоэффективные и надежные программные и аппаратные компоненты. В рамках реализации проектов заказчиков осуществляется:

- аудит и консалтинг при выборе решений;
- внедрение систем управления бизнес-процессом: back-office и front-office;
- обучение персонала;
- поставку и инсталляцию кассового оборудования;
- поставку и инсталляцию весового оборудования;
- поставку и инсталляцию оборудования для штрихового кодирования (принтеров этикеток, сканеров штрихового кода, терминалов сбора данных);
- информационно-техническое сопровождение предприятий.

Особенность компании заключается в том, что сотрудники работают удаленно, находясь по всей территории России. [9, 10]

TrideLine – это программная платформа для создания эффективных систем лояльности покупателей. Это своего рода бонусная программа, которая не требует реальных затрат на скидку. Основные возможности программы: регистрация и идентификация в бонусной программе через телефонный номер; создание специальных бонусных карт для сотрудников; возможность SMS-оповещения и рассылки; создание личного кабинета покупателя.

Руководство компании ООО «ККМ02» поставило задачу создать систему, которая позволит скоординировать действия сотрудников, которые расположены на большом расстоянии друг от друга. Особенностью системы будет не только решение и выполнение рабочих процессов создания новых проектных решений, но и возможность подключения клиентов компании к этой системе и дальнейшее взаимодействие с ними посредством данной системы.

Чтобы процесс разработки системы был достаточно упрощен для разработчика, руководство не стало требовать абсолютно нового продукта. Достаточно было использовать существующий инструментарий для каждой из функций, который будет собран в единую систему [14, 15].

Данная система будет предоставлять гораздо большую функциональность, нежели типичная система управления процессом разработки программного продукта. Система позволит вести базу знаний по проекту, планировать состав релизов, загрузку участников, контролировать ход работ по проекту, подготавливать требования, справочную документацию, организовывать хранилище артефактов, трассировать изменение функциональности начиная от пожелания и заканчивая справочной документацией. А главное, система будет максимально соответствовать бизнес-процессам компании. Однако, для дальнейшего представления системы на рынке будет реализована возможность перенастраивания системы для других похожих компаний [16, 17].

Вся работа системы будет построена на основе методологии Scram, это значит, что будет производиться последовательное выполнение всех фаз проекта, с промежуточным и конечным оглашением результатов. В проекте создаются участники и наделяются соответствующими ролями. Роли определяют полномочия на выполнение тех или иных действий в системе, а также на возможность отображения различных разделов системы. В процессе присутствуют следующие фазы: анализ и проектирование, разработка, тестирование, стабилизация, документирование. Менеджер проекта должен тщательно следить за выполнением хода работ каждым из участников и регулярно проводить отчетные мероприятия, пусть и в режиме онлайн.

Естественно, система имеет существенный недостаток: данная система не сможет выполнить за вас проект, как и любая методология, это лишь набор методик и инструментария, которые существенно повышают ваши шансы не провалить проект и получить весомое преимущество над другими командами разработчиков [18].

В итоге можно сказать, что типичные системы управления процессом разработки программных продуктов могут подойти далеко не каждому предприятию. Каждый стремится внедрить максимально эффективные технологии для поддержания работы компании, и, что немало важно, приемлемые по вопросам финансирования.

На сегодняшний день разработано и утверждено техническое задание, в котором подробно и рассмотрены все аспекты данной системы, которая будет полностью реализована в плановые сроки 2016г.

Литература

1. Гаврилова И.В., Новикова Т.Б., Назарова О.Б., Петеляк В.Е., Агдавлетова А.М. Совершенствование методов построения причинно-следственных диаграмм для решения задач управления в организационных системах / И.В. Гаврилова, Т.Б. Новикова, О.Б. Назарова, В.Е. Петеляк, А.М. Агдавлетова. - Фундаментальные исследования, №8 (часть 2), 2015. - с. 247-251
2. Глущенко Т.Б. Модель подготовки будущих учителей к использованию новых информационных технологий в формировании имиджа образовательного учреждения / Т.Б. Глущенко. - Вестник Челябинского государственного педагогического университета. № 3. С. 54-61.
3. Глущенко Т.Б. Повышение профессиональной квалификации преподавателей как педагогическое условие подготовки будущего учителя к использованию новых информационных технологий в формировании имиджа образовательного учреждения / Т.Б. Глущенко. - Научные труды SWorld. Т. 14. № 1. С. 77а-79.
4. Глущенко Т.Б. Структура и содержание готовности будущего учителя к использованию новых информационных технологий в формировании имиджа образовательного учреждения / Т.Б. Глущенко. - Информатика и образование. № 2. С. 111-113.
5. Давлеткиреева Л.З., Новикова Т.Б., Курзаева Л.В., Лактионова Ю.С., Подкользина Л.В. Управление технологической архитектурой организации на основе усовершенствованного метода EnterpriseArchitecturePlanning / Л.З. Давлеткиреева, Т.Б. Новикова, Л.В. Курзаева, Ю.С. Лактионова, Л.В. Подкользина. - Фундаментальные исследования, №8 (часть 2), 2015. - с. 252-256
6. Кручинская В.С., Новикова Т.Б. Необходимость поддержки и модернизации сайта / В.С. Кручинская, Т.Б. Новикова Т.Б. - Актуальные проблемы современной науки в 21 веке: сборник материалов 6-й международной научно-практической конференции. - Махачкала, 2014
7. Курзаева Л.В., НовиковаТ.Б., ДавлеткирееваЛ.З., НазароваО.Б., БелоусоваИ.Д.. Совершенствование методики построения моделей событийно-управляемого процесса для постановки задач управления в социальных и экономических системах / Л.В. Курзаева, Т.Б. Новикова, Л.З. Давлеткиреева, О.Б. Назарова, И.Д. Белоусова. - Фундаментальные исследования, №8 (часть 2), 2015. - с. 297-302
8. Назарова О.Б., Масленникова О.Е., Давлеткиреева Л.З., Новикова Т.Б. Modeling and factor analysis of efficient increase of bachelors and masters' training connected with applied informatics / О.Б. Назарова, О.Е. Масленникова, Л.З. Давлеткиреева, Т.Б. Новикова. - ARPN Journal of Engineering and Applied Sciences: Asian Research Publishing Network (ARPN), Пакистан.
9. Новикова Т.Б. Интернет-представительство в формировании имиджа образовательного учреждения / Т.Б. Новикова. - Вестник компьютерных и информационных технологий. № 9. С. 49-51.
10. Новикова Т.Б. Подготовка будущего учителя к использованию новых информационных технологий в формировании имиджа образовательного учреждения : автореферат диссертации на соискание ученой степени кандидата педагогических наук / Т.Б. Новикова. - Магнитогорский государственный университет, Магнитогорск
11. Новикова Т.Б. Подготовка будущего учителя к использованию новых информационных технологий в формировании имиджа образовательного учреждения: диссертация на соискание ученой степени кандидата педагогических наук / Магнитогорский государственный университет. Магнитогорск, 2009
12. Новикова Т.Б., Курзаева Л.В., Петеляк В.Е., Масленникова О.Е., Белоусова И.Д. Описание управления бизнес-процессами предприятия на основе методологии IDEF0: трудности разработки, рекомендации по совершенствованию построения диаграмм / Т.Б. Новикова, Л.В. Курзаева, В.Е. Петеляк, О.Е. Масленникова, И.Д. Белоусова. - Фундаментальные исследования, №8 (часть 2), 2015. - с. 318-322
13. Новикова Т.Б., Новые информационные технологии в формировании имиджа образовательного учреждения, Москва, 2015
14. Новикова Т.Б., Сумароковская Е.Ф. ИТ-решения для государственных учреждений на примере отделения пропаганды / Т.Б. Новикова, Е.Ф. Сумароковская. - Современные тенденции развития науки и производства, Сборник материалов Международной научно-практической конференции : в 4-х томах. Западно-Сибирский научный центр, Кузбасский государственный технический университет имени Т.Ф. Горбачева, Международный научно-образовательный центр КузГТУ-Arena Multimedia. Кемерово, 2014. С. 105.
15. Петеляк В.Е., Новикова Т.Б., Масленникова О.Е., Махмутова М.В., Агдавлетова А.М. Data Flow Diagramming: особенности построения моделей описания управления потоками данных в организационных системах / В.Е. Петеляк, Т.Б. Новикова, О.Е. Масленникова, М.В. Махмутова, А.М. Агдавлетова. – Фундаментальные исследования, №8 (часть 2), 2015. - с. 323-327
16. Сафрина С.В., Новикова Т.Б., Давлеткиреева Л.З., Развитие технической инфраструктуры в сфере торговли спортивными товарами, Современные научные исследования и инновации, Международный научно-инновационный центр, Москва, 2013
17. Сильвестрова О.В., Новикова Т.Б. Автоматизация бизнес-процессов медицинского учреждения в рамках проекта «Электронная Россия» / О.В. Сильвестрова, Т.Б. Новикова. - Современные научные исследования и инновации. 2012. № 11 (19). С. 4.
18. Соколова А.А., Новикова Т.Б. Моделирование бизнес-процесса «Ипотечная сделка купли-продажи недвижимости» с использованием методологии ARIS / А.А. Соколова, Т.Б. Новикова. - Гуманитарные научные исследования. 015. № 4-3 (44). С. 149-155.

ТЕОРЕТИЧЕСКАЯ ИНФОРМАТИКА И КОМПЬЮТЕРНЫЕ НАУКИ

Головешкин В.А.[1], Жукова Г.Н.[2], Ульянов М.В.[3], Фомичев М.И.[4]

[1] Институт прикладной механики РАН, Московский государственный университет информационных технологий, радиотехники и электроники (МИРЭА), г. Москва, д.т.н., профессор, nikshevolog@yandex.ru

[2] Московский государственный университет печати (МГУП) им. Ивана Федорова, г. Москва,

к. ф.-м. н., доцент, gnzh@mail.ru

[3] Институт проблем управления РАН им. В.А. Трапезникова, г. Москва, д.т.н., профессор, muljanov@mail.ru

[4] ФКН НИУ Высшая школа экономики (ФКН НИУ ВШЭ), г. Москва, студент, michan94@yandex.ru

СРАВНЕНИЕ РЕСУРСНЫХ ХАРАКТЕРИСТИК ТРАДИЦИОННОГО И МОДИФИЦИРОВАННОГО МЕТОДА ВЕТВЕЙ И ГРАНИЦ ДЛЯ TSP[7]

КЛЮЧЕВЫЕ СЛОВА

Задача коммивояжера (TSP), метод ветвей и границ (МВГ), ресурсная эффективность, время работы алгоритма, бета распределение II типа, система функций Пирсона.

АННОТАЦИЯ

Сравниваются ресурсные характеристики модифицированного и классического МВГ для TSP. На основании экспериментальных результатов показано, что по величине затраченного на поиск решения времени модифицированный вариант МВГ эффективнее классического. Исследована стохастическая зависимость между временем работы каждого из двух исследуемых вариантов МВГ при фиксированном порядке матрицы стоимостей. Также описана зависимость затраченной памяти и времени работы алгоритма от порядка матрицы стоимости.

Постановка задачи

Задача коммивояжера относится к NP-трудным задачам комбинаторной оптимизации и заключается в отыскании оптимального (например, самого дешевого) гамильтонова цикла в полном ориентированном графе. Граф задается матрицей стоимостей, внедиагональные элементы которой (неотрицательные числа) интерпретируются как расстояния между пунктами назначения или стоимости перевозок. Диагональные элементы матрицы стоимостей полагаются условно равными бесконечности, чтобы осуществить запрет петель.

Впервые метод ветвей и границ (МВГ) для решения задачи коммивояжера был предложен в [1], хотя предварительные результаты и общие формулировки идей были представлены в более ранних работах [2-6]. Хорошо исследована задача коммивояжёра с симметричной матрицей, изучен асимметричный случай при условии, что выполняется неравенство треугольника (т.е. перевозка напрямую из А в Б выгоднее, чем через один или несколько промежуточных пунктов).

В дальнейшем для решения задачи коммивояжера применялись эволюционные, в основном генетические и муравьиные алгоритмы, а также комбинации метода ветвей и границ с генетическими алгоритмами [7]. Генетические и муравьиные алгоритмы использовались для повышения точности оценки снизу, что позволило более эффективно исключать бесперспективные подзадачи.

Несмотря на то, что задача коммивояжера достаточно хорошо исследована, прогнозирование трудоемкости или временных характеристик для конкретной задачи вызывает существенные трудности. Проводились исследования по прогнозированию времени, требующегося для решения задачи коммивояжера методом ветвей и границ, на основе случайного выбора [9]. В [10] предложена процедура прогнозирования числа вершин дерева решений на основе анализа поддерева решений, полученного на начальном этапе расчетов. Особенностью перечисленных методов прогноза является зависимость прогноза от конкретного экземпляра задачи, а также то, что прогноз может быть получен только спустя какое-то время после начала расчетов.

7 Набота выполнена при поддержке РФФИ, грант №16-07-160

Особенности модифицированной реализации МВГ

Поиск решения задачи коммивояжера методом ветвей и границ (МВГ) заключается в разделении множества допустимых решений на подмножества с целью дальнейшего сокращения перебора. На каждом шаге выбирается дуга ветвления, и множество туров делится на классы, один из которых состоит из туров, содержащих эту дугу, а второй — из оставшихся допустимых туров (ветвление). В каждом классе производится оценка минимальной стоимости тура и находится стоимость какого-нибудь допустимого тура из этого класса. Если стоимость найденного тура меньше стоимости ранее найденного «рекордного» тура, то новый тур объявляется «рекордом». Классы допустимых туров с нижней оценкой, превышающей стоимость «рекорда», исключаются из дальнейшего рассмотрения как бесперспективные. Если стоимость «рекорда» такова, что все нижние оценки других классов больше, то найден оптимальный тур. Каждый класс допустимых туров подвергается процедуре ветвления, пока не будет найдено оптимальное решение.

Метод ветвей и границ состоит в построении и исследовании древовидной структуры пространства решений задачи коммивояжера. Построение поискового дерева решений начинается с корня, который будет соответствовать множеству «всех возможных туров», т. е. корень дерева представляет множество R всех $(n-1)!$ возможных туров в задаче с n городами. Ветви, выходящие из корня, отличаются наличием или отсутствием в турах одной дуги, например, (k,l). Множество R делится на два класса $\{k,l\}$ (туры из R, содержащие дугу (k,l)) и $\{\overline{k,l}\}$ (туры, не содержащие эту дугу). Подробное поэтапное изложение метода можно найти, например, в [2,3].

Пример фрагмента такого дерева приведен на рисунке 1 (источник рисунка [2]).

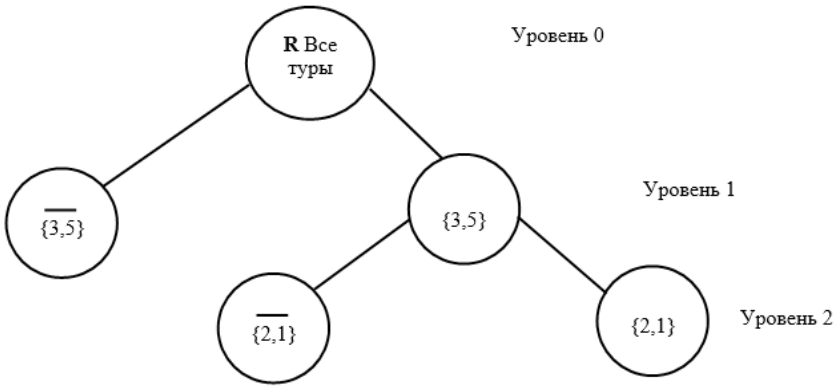


Рис. 1. Фрагмент поискового дерева решений

Вычислительный эксперимент

Генерация матриц стоимости несимметричной задачи коммивояжера проводилась стандартным для языка C++ псевдослучайным равномерным генератором. Элементы матриц (размерности от 11 до 46 с шагом 1) были целыми числами из диапазона от 1 до 2^{15}. Был реализован алгоритм [1] (без предвычисления начального тура жадным алгоритмом) со структурой хранения поискового дерева решений в виде бинарной кучи в классической реализации с перевычислением матрицы стоимостей (т.е. без использования дополнительной памяти) и в модифицированной реализации с хранением локальных (для вершин поискового дерева) матриц стоимостей (далее будем для краткости называть эту модифицированную реализацию МВГ1).

Для каждого фиксированного *n* выполнялось 10 000 псевдослучайных генераций матрицы стоимостей порядка *n,* каждый элемент матрицы – псевдослучайное число с дискретным равномерным распределением в диапазоне от 1 до 2^{15}. Для каждой матрицы была решена задача коммивояжера классическим МВГ (с перевычислением матриц) и МВГ1 (с хранением матриц). Для каждого запуска программы замерялось время, затраченное на поиск оптимального тура, зависящее от числа вершин порожденного поискового дерева решений, а для МВГ1 также и максимальное значение затраченной на хранение матриц дополнительной памяти.

Эксперименты проводились на стационарном компьютере со следующими характеристиками:

— процессор: Intel i7 3770K 3800 ГГц

— оперативная память: Kingston KHX1600C9D3P1 16 ГБ
— материнская плата: GIGABYTE GA-Z77X-D3H
— операционная система: Fedora 21 workstation
Алгоритмы реализованы на языке C++ (компилятор gcc 4.9.2 20150212, Red Hat 4.9.2-6).

Взаимозависимость временных характеристик модифицированного и классического МВГ

Обозначим t_1 время, затраченное классическим МВГ на поиск оптимального тура коммивояжера, t_2 - время работы МВГ1, m_2 - объем памяти, использованный для хранения промежуточных матриц. Изобразим результаты вычислительного эксперимента на диаграмме рассеяния (рис. 2)

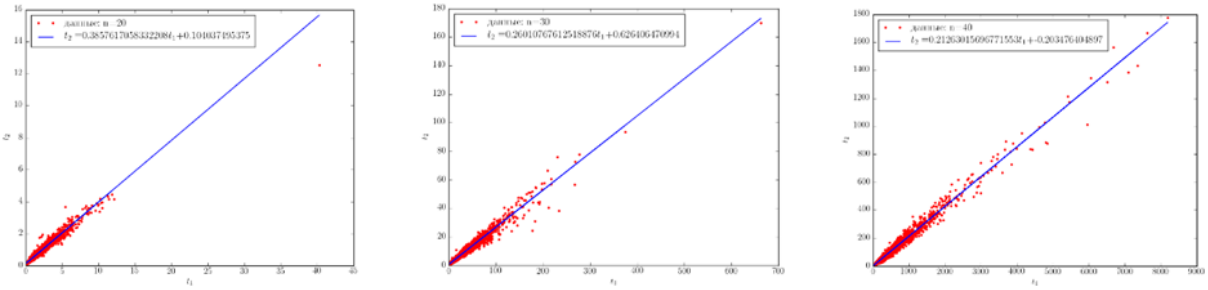


Рис. 2. Диаграмма рассеяния t_1 *и* t_2 , $n=20,30,40$

На рис. 2 изображена синяя линия регрессии, подобранная по методу наименьших квадратов. Отметим, что с ростом размерности матрицы стоимостей уменьшается коэффициент наклона этой прямой, что свидетельствует об увеличении «превосходства» модифицированного метода над классическим в аспекте экономии затраченного времени. Зависимость коэффициента наклона *slope* линии регрессии t_1 на t_2 от размерности *n* матрицы стоимостей (а также коэффициента детерминации r^2) изображена на рис. 3.

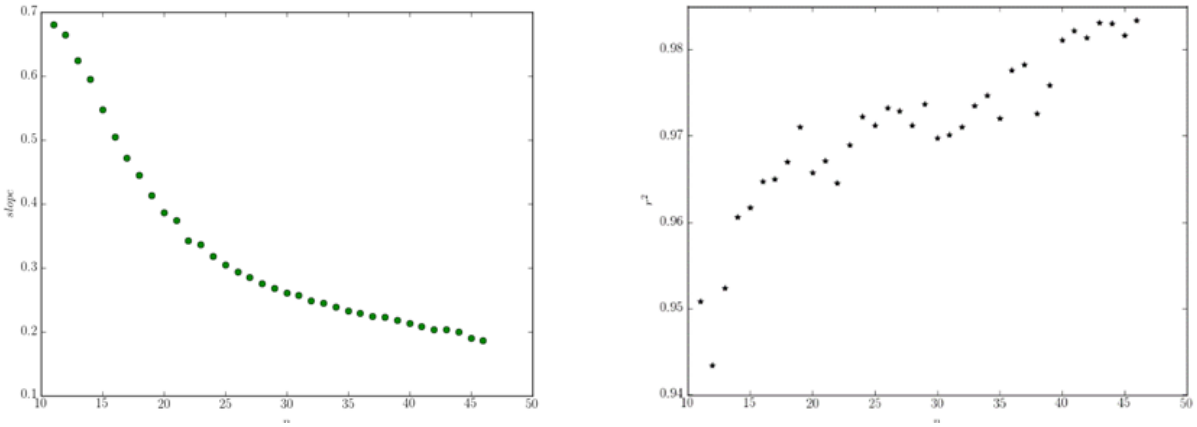


Рис. 3. Зависимость slope и r^2 *от n*

Близкие к единице значения коэффициента детерминации указывают на тесную линейную зависимость t_1 и t_2 , что свидетельствует о большей экономичности (в смысле затрат времени) МВГ1.

Характеристики распределения времени работы МВГ1

Время t , затраченное на поиск оптимального тура коммивояжера, будем рассматривать как случайную величину, определенную на множестве всевозможных матриц размерности n с элементами — натуральными числами, не превосходящими 2^{15} . Это пространство конечное, но содержит огромное число элементов, поэтому для определенной выше случайной величины будем подбирать такое непрерывное распределение, что его функция распределения не слишком сильно отличается от выборочной функции распределения полученных данных.

Данные t_1 представляют собой положительные числа, равные времени, затраченному алгоритмом МВГ на решение задачи коммивояжера размерности n , t_2 — время работы МВГ1

над теми же задачами, m_2 — объем памяти, затраченной МВГ1 на хранение матриц. Задача решалась для n от 11 до 46, в каждом случае было проведено 10 000 вычислительных экспериментов.

Сначала проанализируем гистограммы результатов вычислительного эксперимента. На рис. 4, 5 и 6 приведены гистограммы времени t_1 работы МВГ, t_2 (времени работы МВГ1) и m_2 — объема памяти, затраченной МВГ1 на хранение матриц. На каждом рисунке три гистограммы, соответствующие матрицам стоимости порядка $n=20$, 30, 40.

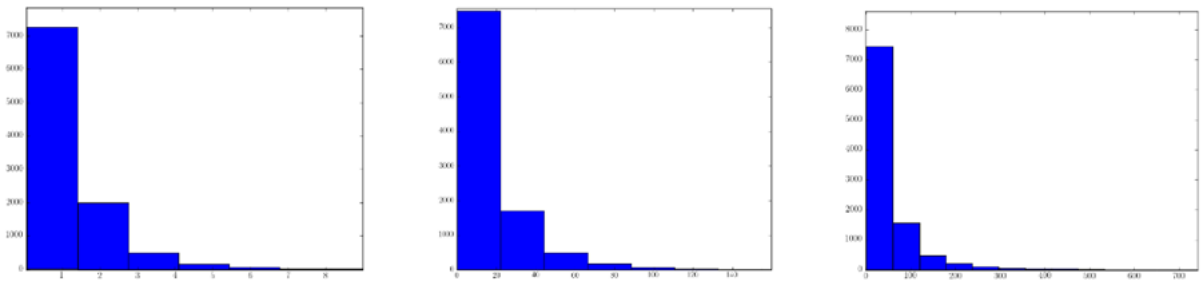


Рис. 4. Гистограммы t_1, $n=20$, 30, 40

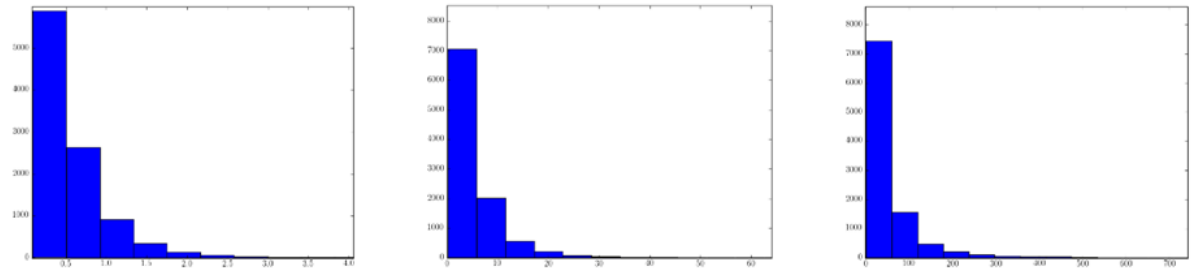


Рис. 5. Гистограммы t_2, $n=20$, 30, 40

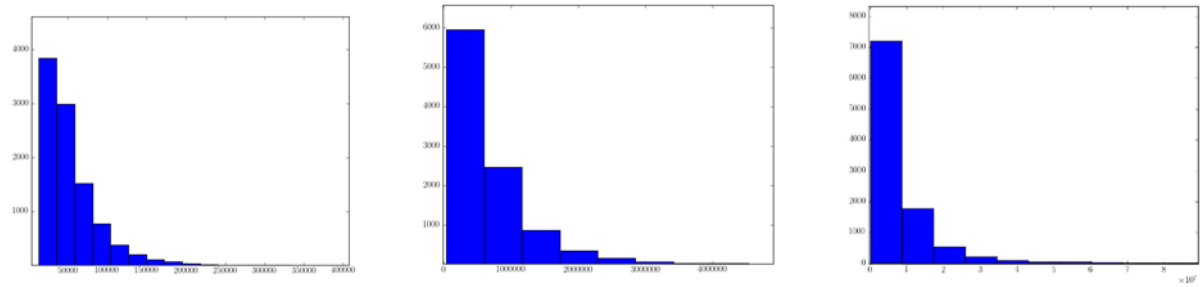


Рис. 6. Гистограммы m_2, $n=20$, 30, 40

Все гистограммы похожи по форме, что и следовало ожидать, учитывая тесную линейную зависимость t_1 и t_2 (аналогичная зависимость имеет место и для m_2 и t_2). Вид гистограмм побуждает подбирать возможное распределение экспериментальных данных в классе непрерывных распределений с плотностью, представляющей собой убывающую на $[a;b]$ функцию, или же имеющую один максимум вблизи a, где $a\geq 0$ $b\leq\infty$.

Будем искать подходящее распределение, имеющее конечный четвертый момент. Для таких распределений определены коэффициент асимметрии γ_1 и коэффициент эксцесса γ_2. Пусть наблюдается случайная величина X, имеющая математическое ожидание EX, дисперсию DX, третий центральный момент $\mu_3(X)=E(X-EX)^3$ и четвертый центральный момент $\mu_4(X)=E(X-EX)^4$, тогда коэффициент асимметрии γ_1 и коэффициент эксцесса γ_2 определяются по формулам [11]

$$\gamma_1=\frac{\mu_3(X)}{(DX)^{3/2}}=\frac{E(X-EX)^3}{\left(E(X-EX)^2\right)^{3/2}},\qquad \gamma_2=\frac{\mu_4(X)}{(DX)^2}=\frac{E(X-EX)^2}{\left(E(X-EX)^2\right)^2}.$$

Покажем, что γ_1 и γ_2 инвариантны относительно линейных преобразований X, для этого преобразованием сдвига и масштаба построим по X случайную величину Y:

$$Y=kX+x_0 . \quad (1)$$

Выразим моменты и коэффициенты асимметрии и эксцесса Y через соответствующие характеристики X.

1.Матожидание $EY=E(kX+x_0)=kEX+x_0$.

2.Дисперсия

$$DY=E(Y-EY)^2=E(kX+x_0-(kEX+x_0))^2=E(kX-kEX)^2=k^2 DX .$$

3.Третий центральный момент $E(Y-EY)^3=E(kX-kEX)^3=k^3E(X-EX)^3$.

4.Четвертый центральный момент $E(Y-EY)^4=k^4E(X-EX)^4$.

5.Коэффициент асимметрии

$$\gamma_1(Y)=\frac{E(Y-EY)^3}{(DY)^{3/2}}=\frac{k^3E(X-EX)^3}{(k^2E(X-EX)^2)^{3/2}}=\frac{k^3E(X-EX)^3}{k^3(E(X-EX)^2)^{3/2}}=\frac{E(X-EX)^3}{(E(X-EX)^2)^{3/2}}=\gamma_{1_1} .$$

6.Коэффициент эксцесса

$$\gamma_2(Y)=\frac{E(Y-EY)^4}{(DY)^2}-3=\frac{k^4E(X-EX)^4}{k^4(DX)^2}-3=\frac{E(X-EX)^4}{(DX)^2}-3=\gamma_2(X)=\gamma_2 .$$

Как видно, коэффициенты асимметрии и эксцесса не изменяются при линейном преобразовании, что позволяет подбирать вид распределения по выборочным коэффициентам асимметрии и эксцесса, а затем за счет подбора коэффициентов k и x_0 добиваться равенства выборочных среднего и дисперсиии Y теоретическим значениям для подобранного распределения.

В соответствии с подобранным распределением вычисляются математическое ожидание EY и дисперсия DY, затем по ним находятся k и x_0 из (1):

$$k=\sqrt{\frac{DY}{DX}} , \qquad x_0=EY-kEX . \quad (2)$$

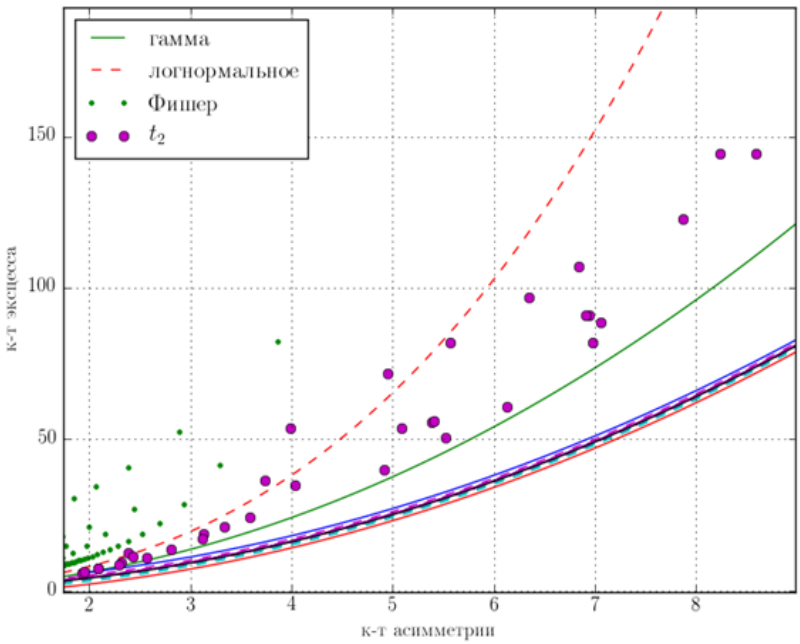


Рис.7. Коэффициенты асимметрии и эксцесса (t_2)

Следует отметить, что по значениям выборочных коэффициентов асимметрии и эксцесса не всегда возможно подобрать хотя бы одно «классическое» распределение.

Для подбора вида распределения воспользуемся методом К. Пирсона [13]. На рис. 7 в системе координат «коэффициент асимметрии – коэффициент эксцесса» изображены некоторые

классические распределения, а также точки, соответствующие значениям коэффициентов асимметрии и эксцесса экспериментальных данных. Данные представляют собой 10000 положительных чисел, равных времени t_2, затраченному модифицированным алгоритмом МВГ на решение задачи коммивояжера размерности n. Задача решалась для n от 11 до 46, каждому из этих случаев соответствует фиолетовый маркер.

Выборочные коэффициенты асимметрии и эксцесса t_1 (время работы классического МВГ), t_2 (время работы модифицированного МВГ), а также m_2 (объем затраченной на хранение матриц памяти) изображаются на рисунке 8 точками примерно из одной и той же области.

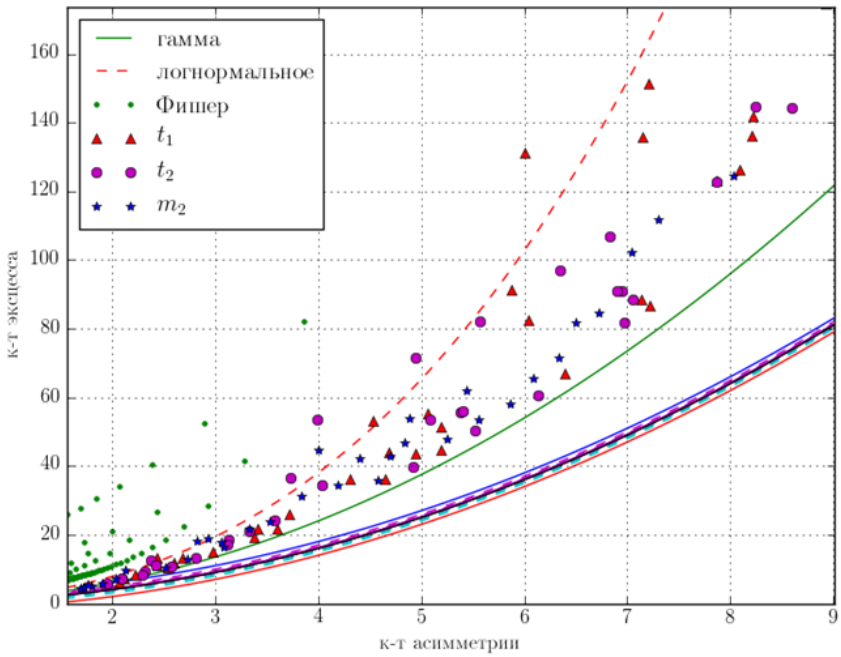


Рис.8. Коэффициенты асимметрии и эксцесса (t_1 , t_2 , m_2)

Если результаты эксперимента возможно описать единообразно некоторым распределением, то это распределение должно иметь коэффициенты асимметрии и эксцесса «между» соответствующими значениями для гамма и логнормального распределения, это область распределений шестого типа системы Пирсона (бета распределение второго типа, Beta prime distribution) [12,13].

Бета распределение второго типа описывается двумя параметрами $\alpha>0$ и $\beta>0$, причем четвертый момент конечен только при $\beta>4$. Плотность распределения имеет вид [12]:

$$f(x)=\frac{x^{\alpha-1}(1+x)^{-\alpha-\beta}}{B(\alpha,\beta)}, \tag{3}$$

где $B(\alpha,\beta)$ — бета функция, коэффициенты асимметрии и эксцесса равны соответственно

$$\gamma_1=\frac{2(2\alpha+\beta-1)}{\beta-3}\sqrt{\frac{\beta-2}{\alpha(\alpha+\beta-1)}} \tag{4}$$

$$\gamma_2=6\frac{\alpha(\alpha+\beta-1)(5\beta-11)+(\beta-1)^2(\beta-2)}{\alpha(\alpha+\beta-1)(\beta-3)(\beta-4)}=$$
$$=\frac{6(5\beta-11)}{(\beta-3)(\beta-4)}+\frac{6(\beta-1)^2(\beta-2)}{\alpha(\alpha+\beta-1)(\beta-3)(\beta-4)}=B+\frac{C}{\alpha(\alpha+\beta-1)}, \tag{5}$$
$$B=\frac{6(5\beta-11)}{(\beta-3)(\beta-4)},C=\frac{6(\beta-1)^2(\beta-2)}{(\beta-3)(\beta-4)}.$$

Параметр α находится из (5) как решение квадратного уравнения относительно α

$$\alpha(\alpha+\beta-1)=\frac{C}{\gamma_2-B}, \qquad \alpha=\frac{1-\beta+\sqrt{D}}{2}, \tag{6}$$

где $D=(\beta-1)^2+\frac{4C}{\gamma_2-B}$. Перед дискриминантом выбран знак плюс, чтобы обеспечить положительное значение α с учетом $\beta>4$.

Подставляя C и D в (6) получим

$$\alpha=\frac{\beta-1}{2}\left(\sqrt{\frac{24(\beta-2)}{\gamma_2(\beta-3)(\beta-4)-6(5\beta-11)}+1}-1\right), \tag{7}$$

Затем возведем обе части (4) в квадрат (обе части положительны, поскольку $\beta>4$)

$$\gamma_1^2=\frac{4(2\alpha+\beta-1)^2(\beta-2)}{\alpha(\alpha+\beta-1)(\beta-3)^2}$$

и придадим полученному уравнению вид

$$\alpha(\alpha+\beta-1)(\beta-3)^2\gamma_1^2-4(2\alpha+\beta-1)^2(\beta-2)=0 . \tag{8}$$

Параметр β найдем, подставив в (8) α из (7) и решив полученное уравнение:

$$\beta=3+\frac{12+2\gamma_2}{2\gamma_2-3\gamma_1^2} .$$

Подставляя найденное значение β в (7), найдем α .

Некоторые значения параметров, полученные по экспериментальным данным приведены в таблице 1 и на рис. 9.

Табл. 1. Точечные оценки параметров бета распределения II типа

	t_1		$s\,t_2$		m_2	
n	α	β	α	β	α	β
21	0.656233	14.78672	0.771754	23.20291	0.851123	19.771012890
22	0.259704	5.144625	0.672774	5.228451	1.196278	5.443517978
23	0.626576	9.684045	0.72864	15.68017	0.842763	13.592175809
24	0.373771	6.602246	0.606926	6.985486	0.68882	7.012765161
25	0.427045	8.047167	0.594325	9.015214	0.620072	8.944139370
26	0.528095	9.268592	0.819729	9.190158	0.871106	9.230012948
27	0.416076	15.87569	0.692569	9.277104	0.755955	8.554410508
28	0.493696	14.66053	0.659561	12.06239	0.642138	13.137767265
29	0.273568	6.170698	0.270182	5.814129	0.443837	6.279142455
30	0.370631	5.464604	0.420186	5.485285	0.524322	6.653685070
31	0.25838	10.17488	0.256808	8.034325	0.353448	7.460506640
32	0.380466	7.469743	0.339419	7.011105	0.459047	8.019133906
33	0.271364	8.240574	0.254145	8.05216	0.350575	8.009574496
34	0.117049	7.045287	0.100416	7.475167	0.165922	6.909838078
35	0.236292	5.401035	0.212038	6.056703	0.299458	6.838542935
36	0.052658	5.186972	0.060503	5.204344	0.086996	5.065607549
37	0.069879	7.094084	0.06939	7.53807	0.113203	7.665997073
38	0.253724	13.82903	0.213711	16.2068	0.272672	12.530710979
39	0.195116	14.98218	0.171445	15.1198	0.213455	11.395406361
40	0.102776	13.48028	0.111546	12.99175	0.154693	9.911065410
41	0.112186	6.277624	0.126788	6.524766	0.174313	7.811211803
42	0.110775	7.706199	0.123706	7.295158	0.160005	6.678007491
43	0.123975	16.2609	0.130429	18.62152	0.161407	12.659731130
44	0.115105	10.82112	0.125164	9.71375	0.153199	8.375463681

45	0.124168	6.660459	0.145357	8.200421	0.165672	10.185727312
46	0.124124	7.273358	0.150179	7.998998	0.188199	11.502324733
MIN	0.052658	5.144625	0.060503	5.204344	0.086996	5.065607549
MAX	0.656233	16.2609	0.819729	23.20291	1.196278	19.77101289

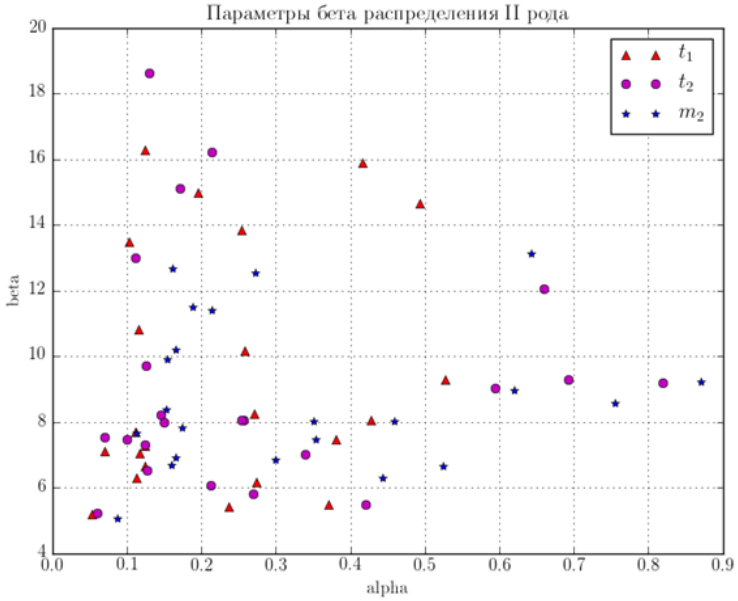


Рис.9. Точечные оценки параметров бета распределения II типа

На рис. 10 приведены гистограмма для t_1 при $n=40$ и плотность бета распределения II типа с параметрами, рассчитанными по выборке.

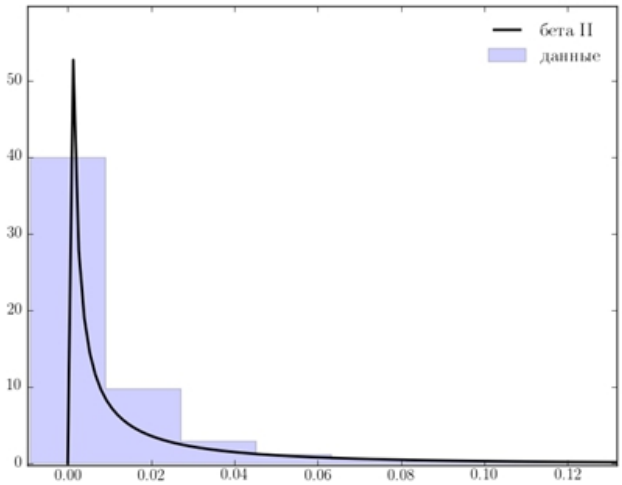


Рис.10. Гистограмма t_1 и плотность бета распределения II типа

Выводы

Модифицированный вариант МВГ по показателю «затраченное на поиск решения время» эффективнее классического варианта, причем выигрыш во времени (для $n=40$ почти в пять раз) становится все более значимым с ростом размерности матрицы стоимостей. Между значениями затраченного времени обоих вариантов МВГ наблюдается тесная линейная зависимость при фиксированном n, теснота связи растет с ростом n, приближаясь к 1.

Выигрыш во времени достигается ценой значительных затрат памяти, которые также растут с ростом n. Данные вычислительных экспериментов достаточно хорошо аппроксимируются бета распределением II типа, выбранным по предложенной методике на основе коэффициентов асимметрии и эксцесса.

Литература

1. Little, J. D. C., K. G. Murty, D.W. Sweeney, C. Karel. 1963. An algorithm for the traveling salesman problem. Operations Research 11, 972–989.
2. Dantzig, G., R. Fulkerson, S. Johnson. 1954. Solution of a large scale traveling salesman problem. Technical Report P-510. RAND Corporation, Santa Monica, California, USA.
3. Dantzig, G. B., D. R. Fulkerson, S. M. Johnson. 1959. On a linearprogramming, combinatorial approach to the traveling-salesman problem. Operations Research 7, 58–66.
4. Eastman, W. L. 1958. Linear Programming with Pattern Constraints. Ph.D. Thesis. Department of Economics, Harvard University, Cambridge, Massachusetts, USA.
5. Land, A. H., Doig A. G. 1960. An automatic method of solving discrete programming problems. Econometrica 28, 497–520.
6. Manne, A. S., H. M. Markowitz. 1956. On the solution of discrete programming problems. Technical Report P-711. RAND Corporation, Santa Monica, California, USA.
7. C. Cotta, J. Aldana, A. Nebro, J. Troya, Hybridizing genetic algorithms with branch and bound techniques for the resolution of the TSP, in: D. Pearson, N. Steele, R. Albrecht (Eds.), Artificial Neural Nets and Genetic Algorithms 2, Springer-Verlag, Wien New York, 1995, pp. 277-280.
8. G. Carpaneto, P. Toth, Some new branching and bounding criteria for the asymmetric traveling salesman problem, Management Science. 1980, v. 26, 736–743.
9. Knuth, D. E. 1975. Estimating the efficiency of backtracking programs. Mathematics of Computing, v. 29, pp. 121–136.
10. Cornuéjols, G.; Karamanov, M.; and Li, Y. 2006. Early estimates of the size of branch-and-bound trees. INFORMS Journal on Computing. 2006, v. 18, No. 1, pp. 86—96.
11. Крамер Г. Математические методы статистики. М.: Мир, 1975. — 648 с.
12. Jonhnson, N.L., Kotz, S., Balakrishnan, N. 1995. Continuous Univariate Distributions, v. 2, Wiley.
13. Pearson, K Contributions to the Mathematical Theory of Evolution. III. Regression, Heredity and Panmixia, Philosophical Transactions of the Royal Society of London, 1896. 187, pp. 253–318.
14. Ульянов М.В., Фомичев М.И. Ресурсные характеристики способов организации дерева решений в методе ветвей и границ для задачи коммивояжера // Бизнес – информатика, 2015, (статья принята к публикации в №4 журнала в 2015 г.)
15. Ульянов М.В. Ресурсно-эффективные компьютерные алгоритмы. Разработка и анализ. — М.: ФИЗМАТЛИТ, 2008. — 304 с.

Абаев П.О.[1], Бесчастный В.А.[2], Гайдамака Ю.В.[3]

[1] Российский университет дружбы народов, г. Москва, к.ф.-м.н., доцент кафедры прикладной информатики и теории вероятностей, pabaev@sci.pfu.edu.ru

[2] Российский университет дружбы народов, г. Москва, магистр кафедры прикладной информатики и теории вероятностей, vbeschastny@sci.pfu.edu.ru

[3] Российский университет дружбы народов, г. Москва, к.ф.-м.н., доцент кафедры прикладной информатики и теории вероятностей, ygaidamaka@mail.ru

О ПРИМЕНЕНИИ ПРОСТРАНСТВЕННЫХ ТОЧЕЧНЫХ ПРОЦЕССОВ В РЕШЕНИИ ОПТИМИЗАЦИОННЫХ ЗАДАЧ ДЛЯ БЕСПРОВОДНЫХ СЕТЕЙ С УСТАНОВЛЕНИЕМ ПРЯМЫХ СОЕДИНЕНИЙ[8]

КЛЮЧЕВЫЕ СЛОВА

Пространственные точечные процессы, SIR, D2D.

АННОТАЦИЯ

Наиболее точные результаты при моделировании беспроводных сетей позволяют достичь процессы твердого ядра Матерна. Тем не менее они достаточны сложны для анализа, и потому часто вместо них применяют пуассоновские пространственные процессы, которые отличаются своей простотой и широким применением в различных направлениях исследований. При этом переходе происходит некоторая потеря точности. В работе проводится оценка целесообразности такого перехода в контексте беспроводных технологий передачи данных, в которых ключевым показателем качества функционирования является отношение сигнала к интерференции SIR между взаимодействующими устройствами.

Установление прямых соединений между близко расположенными мобильными устройствами позволяет эффективно использовать выделенный спектр частот, увеличить общую пропускную способность сети, а также позволяет предоставлять услуги на базе технологии P2P (Peer-to-Peer) и определения текущего местоположения (Location-based services). Еще одним преимуществом такого типа соединений является возможность организации с помощью LTE-устройств (Long Term Evolution), поддерживающих технологию D2D (Device-to-Device), сетей общественной безопасности, которые должны функционировать там, где сотовая связь недоступна по тем или иным причинам, например, в случае возникновения аварийных ситуаций. Внедрение D2D-технологий постепенно ставит вопрос о пересмотре традиционной архитектуры сотовых сетей, центральными элементами которых являются базовые станции (рис. 1) [1]. Также к преимуществам использования D2D-технологии относятся[2]:

- высокая скорость передачи данных и низкая задержка за счет прямого соединения на малом расстоянии;
- более эффективное с точки зрения использования ресурсов устройств прямое взаимодействие, чем через базовую станцию и (возможно) базовую сеть;
- экономичное использование энергоресурсов беспроводных устройств;
- увеличение зоны покрытия за счет механизма ретрансляции.

При организации D2D-соединений возникает множество требующих решения задач. Одной из наиболее принципиальных задач является разделение частотного ресурса между двумя типами соединений: D2D-соединениями и соединениями традиционной сотовой связи.

Совокупность различных аспектов, в том числе интерференция и безопасность, не позволяет с уверенностью дать ответ на вопрос: должны ли D2D-устройства использовать строго выделенный для них спектр частот, или у них может быть доступ к частотам, используемым устройствами сотовой связи. К тому же, задача распределения частот осложняется тем, что устройства в некоторых случаях могут переключаться между обоими типами соединений. Ответ может быть получен с помощью решения задач, целью которых является оптимизация

8 Работа выполнена при поддержке РФФИ в рамках научных проектов № 14-07-00090, 15-07-03051, 15-07-03608.

архитектуры и параметров сети, обеспечивающих наилучшие показатели её производительности, и последующего сравнительного анализа полученных результатов. К наиболее важным показателям можно отнести пиковые скорости передачи данных между устройствами, задержка начала передачи данных, соотношение сигнала к интерференции SIR (Signal-to-Interference Ratio), энергосбережение, эффективность использования частотного спектра и др.

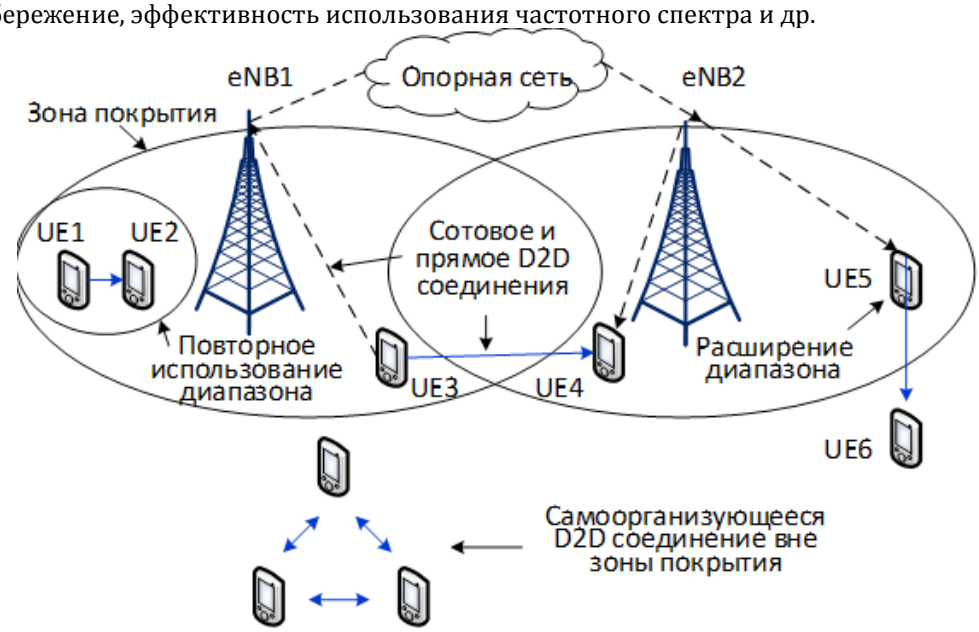


Рис. 1. Беспроводная сеть с установлением D2D-соединений

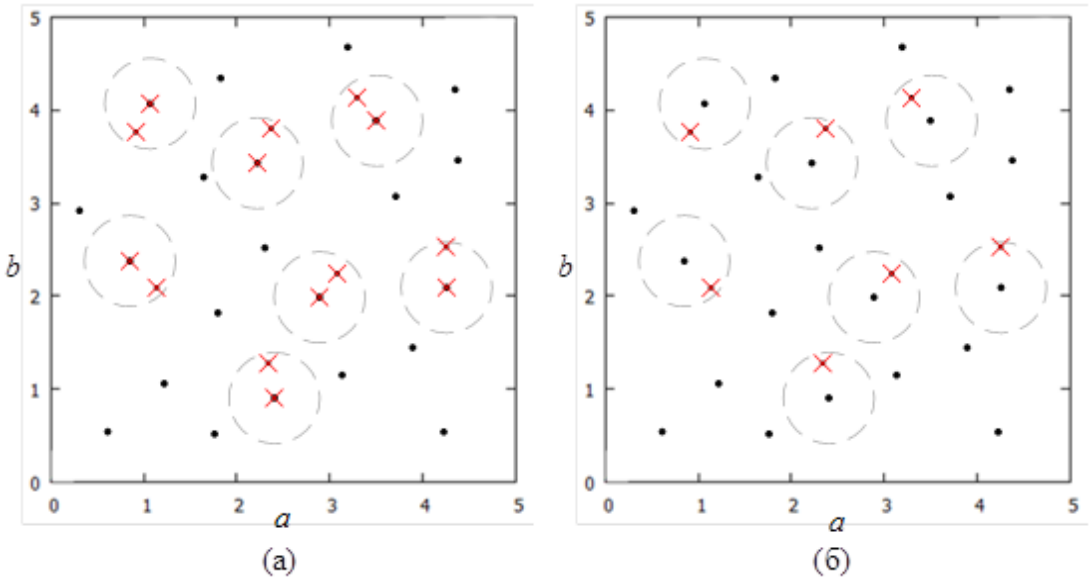


Рис. 2. Процессы твердого ядра MHC I (а) и MHC II (б), полученные из исходного пуассоновского пространственного процесса

Одними из наиболее интересных задач при изучении установления соединений в D2D-сетях являются задача поиска оптимального метода распределения данных в узлах сети для обеспечения высокой вероятности их доступности, а также задача выбора стратегии оптимального хранения файлов в сети. Для их решения в [3] применятся пуассоновские пространственные процессы (PPP, Poisson Point Process). Однако по сравнению с процессами твердого ядра Матерна (MHC, Matern Hardcore Process) PPP не позволяет достаточно точно учитывать факторы, оказывающие влияние на процесс передачи данных между устройствами, такие как расположение устройств относительно друг друга, затухание сигнала, интерференция и др. [4,5]

Для получения процессов Матерна различают два способа зависимого прореживания (т. е.

прореживание зависит от исходного процесса):

1.МНС I – процесс получается путем удаления каждой пары точек, расстояние между которыми меньше, чем параметр *r* (рис. 2 (а));

2.МНС II – процесс получается путем удаления точки с большим весом из каждой пары точек, расстояние между которыми меньше, чем параметр *r* (рис. 2(б), в центре окружностей находятся точки с меньшим весом).

При этом количество оставшихся точек будет распределено также, как и в случае с PPP, в соответствии с пуассоновским распределением с параметром $\lambda_{MHC}=\frac{1-e^{-\lambda_{ppp}\pi r^2}}{\pi r^2}$ [6].

Таким образом, алгоритм прореживания для получения МНС II процесса можно представить в виде:

points_field ← (empty)
for each k from 1 to n do
    point ← ξ
    point.marker ← k
end for
join point to points_field
points_to_delete ← (empty)
for each point in points_field do
    for each neighboring_point in points_field do
        if || point – neighboring_point || < *r* and neighboring_point.marker < point.marker **then**
            if point ∉ points_to_delete **then**
                join point to Points_to_delete
            end if
            (next point)
        end if
    end for
end for
for each point in points_to_delete
    delete point from Points_field
end for

Выбор модели оказывает большое влияние на значение интерференции, в то время как влияние на SINR значительно меньше. В большинстве случаев МНС II-типа дает лучшее приближение, чем МНС I-типа [5]. Однако такие процессы очень сложны для анализа, поскольку получение аналитического решения в этом случае представляется практически невозможным. Часто их заменяют хорошо изученными и относительно простыми пространственными пуассоновскими процессами, с интенсивностью, полученной с помощью модели МНС [6]. Для PPP, в частности, известно распределение расстояний до *n*-ближайшего соседа [7], которое имеет классический вид гамма-распределения (1):

$$f(t,n)=\frac{T^{n-1}}{(n-1)!}e^{-t},\ t>0,\ n=1,2,\ldots \quad (1)$$

где $t=\pi\lambda_{ppp}r^2$, что практически полностью определяет все характеристики взаимодействия устройств в модели.

В работе исследуется возможность и точность аппроксимации процессов твердого ядра Матерна пространственными пуассоновскими процессами. Оценка точности построена на анализе расстояний между устройствами, а также отношения сигнала к интерференции SIR [8].

Рассмотрим фрагмент сети, который представляет собой плоскость с распределенными на ней устройствами. Для простоты будем считать, что на плоскости нет объектов, которые могли вызвать затухания сигнала, мощности сигнала всех устройств положим равными. В рамках данной работы рассматривается только нисходящий сигнал от устройств-источников, которые в процессе функционирования могут вызывать явление интерференции. Таким образом, имеет смысл рассматривать все устройства как потенциальные источники.

Из представленных упрощений можно получить формулу для определения SIR в виде (2).

$$SIR = \frac{g l_0^{-\alpha}}{g \sum_{i=1}^{N} l_i^{-\alpha}} = l_0^{-\alpha} \cdot \left[\sum_{i=1}^{N} l_i^{-\alpha} \right]^{-1}, \quad (2)$$

Где α – показатель затухания сигнала в среде (далее в работе полагается $\alpha = 2$, что соответствует наиболее приближенным к идеальным условиям), l_0 – расстояние между рассматриваемой парой «целевой источник-приемник» (ЦИ-П), l_i – расстояние между парами «целевой источник-интерферирующий *i*-источник» (ЦИ-ИИ), g – мощность сигнала от источников, *N+1* – общее количество источников (рис. 3).

Рис. 3. Пары ЦИ-П и ЦИ-ИИ

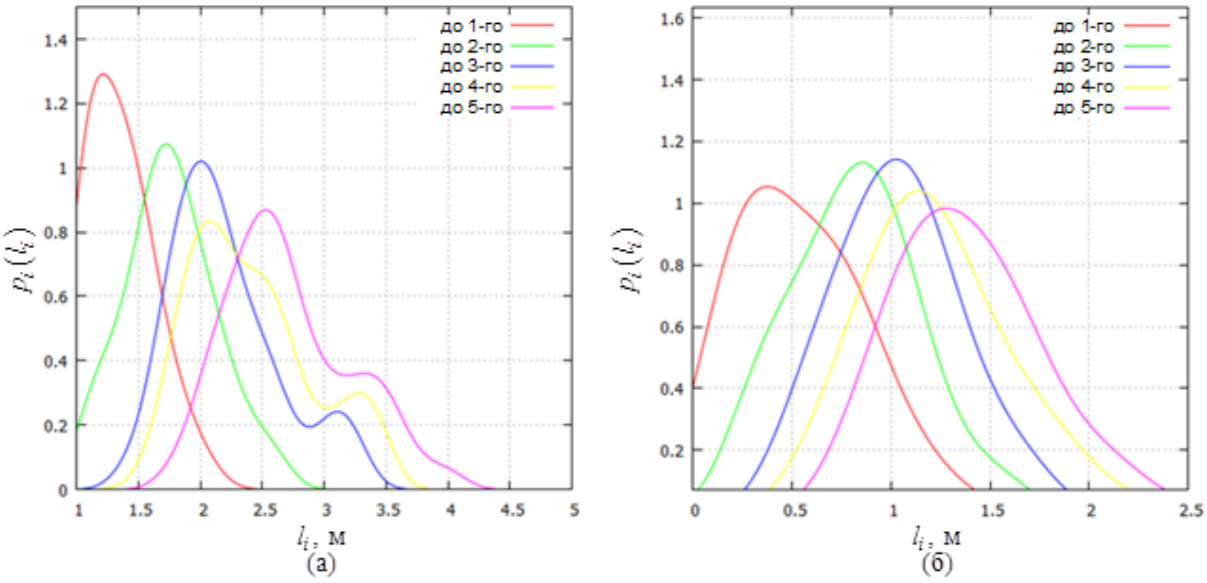


Рис. 4. Эмпирические функции плотности распределения расстояний до ближайших пяти ИИ при (а) МНС II, (б) PPP

Как уже ранее отмечалось, наиболее интересным параметром при исследовании взаимодействия беспроводных устройств является SIR, которое в общем случае является случайной величиной и меняется в зависимости как от принимающего устройства, так и от момента времени. Из формулы (2) видно, что наибольший вклад в уменьшение этого показателя

привносят ближайшие устройства, поэтому целесообразно проводить анализ в контексте именно таких устройств. Для определенности будем рассматривать пять пар ЦИ-ИИ с наименьшими расстояниями. На рисунке 4 представлены эмпирические функции плотности распределения расстояний, из которых видно, что отличия в расстояниях, даже в среднем значении, существенны. Результаты показаны для пространственных процессов с интенсивностями $\lambda_{MHC}=\lambda_{ppp}=1$ ИИ/м^2, параметром МНС-процесса r=1.

На рисунке 5 представлены эмпирические функции распределения и функция плотности SIR для пары ЦИ-П, где источник находится в трех положениях: $l_0 \in \left\{ \frac{1}{4}r; \frac{1}{2}r; \frac{3}{4}r \right\}$. Параметры моделирования аналогичны.

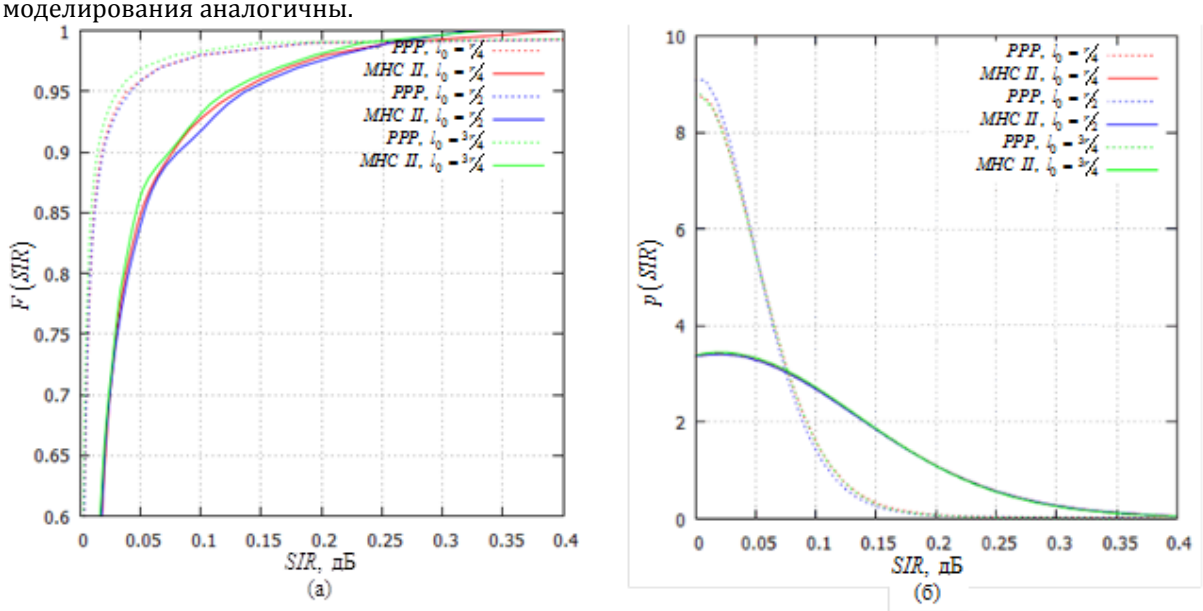


Рис. 5. Эмпирические функция распределения (а) и функция плотности (б) SIR для ЦИ-П

Таким образом, можно сделать вывод о том, что в терминах SIR погрешность при переходе между процессами усугубляется еще больше.

В качестве критерия целесообразности перехода от МНС к PPP выберем критерий Фишера равенства выборочных дисперсий SIR [9]. В случае если критерий Фишера выполнятся при некотором наборе исходных параметров, то переход между процессами не приведет к значительной потер точности. В случае если критерий не выполняется, переход приведет к резкому ухудшению приближенности моделируемых процессов к реальным. На рисунке 6 представлена диаграмма, где обозначена область (зона 1) значений параметров, на которых критерий выполняется полностью.

В зонах 2 и 3 критерий не выполняется только для не более чем двух и четырёх ближайших пар ЦИ-ИИ соответственно.

Стоит отметить, что на рисунке 6 по оси абсцисс отложены значения параметра $y=\frac{r}{s}$, где S – это площадь исследуемого фрагмента сети. Несложно показать, что при выполнении этого соотношения, результаты являются действительными для любых плоскостей выпуклой формы. Исключением являются слишком малые плоскости, площадь которых не позволяет разместить статистически достаточное количество устройств (приблизительно менее 25 м^2).

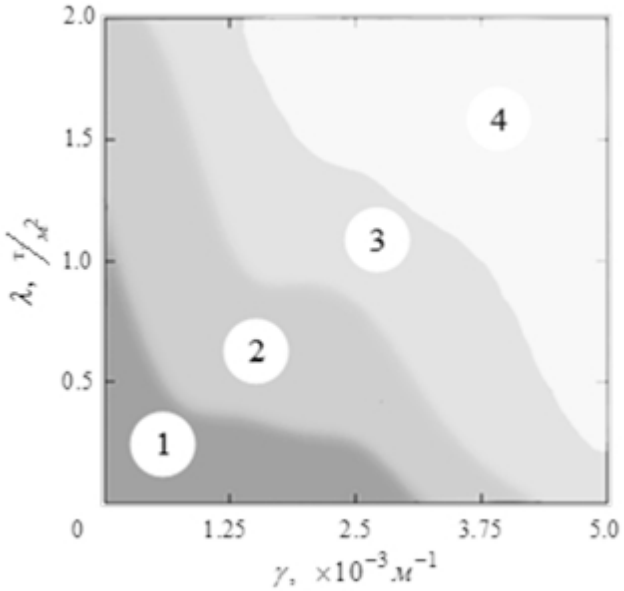


Рис. 6. Критерий Фишера равенства выборочных дисперсий SIR

В работе построена математическая модель взаимодействия беспроводных устройств, применимая для решения задач оптимизации для беспроводных сетей с установлением D2D-соединений. Установлено, что аппроксимация процессов твердого ядра Матерна пространственными пуассоновскими процессами приводит к значительной потери точности при моделировании процессов. Тем не менее, существует ограниченная область значений параметров системы, при которых такой переход не влечет за собой качественного ухудшения приближенности модели к реальным процессам, и при этих значениях пуассоновские пространственные процессы могут применяться с большой долей уверенности.

Литература

1. Hesham ElSawy, Ekram Hossain, Mohamed-Slim Alouini Analytical Modeling of Mode Selection and Power Control for Underlay D2D Communication in Cellular Networks // IEEE Ieee Transactions on Communications, Vol. 62, No. 11. – 2014, pp.4147-4161.
2. Hesham ElSawy, Ekram Hossain, Martin Haenggi Stochastic Geometry for Modeling, Analysis, and Design of Multi-Tier and Cognitive Cellular Wireless Networks: A Survey// IEEE Communications Surveys & Tutorials, Vol. 15, No. 3. – 2013, pp.996-1019.
3. Derya Malak, Mazin Al-Shalash Device-to-Device Content Distribution: Optimal Caching Strategies and Performance Bounds // IEEE ICC 2015 - Workshop on Device-to-Device Communication for Cellular and Wireless Networks. – 2015, pp.664-669.
4. Abdelrahman M. Ibrahim, Tamer ElBatt , Amr El-Keyi Coverage Probability Analysis for Wireless Networks Using Repulsive Point Processes // IEEE 24th International Symposium on Personal, Indoor and Mobile Radio Communications: Fundamentals and PHY Track. – 2013, pp.1002-1007.
5. Qiaoyang Ye, Mazin Al Shalash, Constantine Caramanis, Jeffrey G. Andrews Device-to-Device Modeling and Analysis with a Modified Matern Hardcore BS Location Model // IEEE Communication Theory Symposium. – 2013, pp.1825 1830.
6. F. Baccelli, B. Blaszczyszyn Stochastic geometry and wireless networks, vol. 1. Now Publishers Inc, 2009. – 150 c.
7. D. Moltchanov Distance distributions in random networks // Tampere University of Technology. – 2013, pp.1-32.
8. Begishev V., Kovalchukov R., Samuylov A., Ometov A., Moltchanov D., Gaidamaka Yu., Andreev S.,Koucheryavy E. An analytical approach for SINR estimation in adjacent rectangular premises // Proc. of the 15th IEEE International Conference on Next Generation Wired/Wireless Networking NEW2AN 2015. – 2015, pp.446-458.
9. Левин Б. Р. Теоретические основы статистической радиотехники. – М.: Радио и связь, 1989. – 656 с.

Абаев П.О.[1], Хачко А.В.[2], Бесчастный В.А.[3]

[1] Российский университет дружбы народов, г. Москва, к.ф.-м.н., доцент кафедры прикладной информатики и теории вероятностей, pabaev@sci.pfu.edu.ru

[2] Российский университет дружбы народов, г. Москва, магистр кафедры прикладной информатики и теории вероятностей, axachko@sci.pfu.edu.ru

[3] Российский университет дружбы народов, г. Москва, магистр кафедры прикладной информатики и теории вероятностей, vbeschastny@sci.pfu.edu.ru

О ЗАДАЧЕ ОПТИМИЗАЦИИ РАБОТЫ СЕРВЕРА УСТАНОВЛЕНИЯ СОЕДИНЕНИЯ С МЕХАНИЗМОМ ЛОКАЛЬНОГО КОНТРОЛЯ ПЕРЕГРУЗКИ[9]

КЛЮЧЕВЫЕ СЛОВА

SIP, локальное управление нагрузкой, приоритетное обслуживание, оптимальное управление.

АННОТАЦИЯ

Качество предоставляемых услуг в современных телекоммуникационных сетях связи зависит от своевременной и надежной доставки сигнальных сообщений между узлами сети. Протокол SIP является основным сигнальным протоколом в современных телекоммуникационных сетях. В стандарте RFC 5390 Розенберг, один из авторов протокола SIP, признал несостоятельность протокола в преодолении проблем, связанных с перегрузкой сети, и сформулировал основные требования к механизмам контроля перегрузки. Созданная в 2010 году, рабочая группа SOC (SIP Overload Control) комитета IETF разработала две схемы контроля перегрузки – схему со сбросом сообщений на стороне клиента (Loss-based overload control) и схему с ограничением скорости потока сигнальных сообщений (Rate-based overload control). Обе схемы предполагают наличие обратной связи между серверами и требуют динамического изменения контрольных параметров с учетом изменяющего во времени шаблона трафика, поступающего на сервера. В статье построена модель сервера с алгоритмом локального контроля перегрузки, которая может быть применена совместно с одним из механизмов LBOC или RBOC. Получены формулы для основных вероятностно-временных характеристик и проведен численный анализ.

Переход сетей связи на технологии NGN и IMS, наряду с новыми возможностями, приводит к появлению дополнительных задач по поддержанию надлежащего качества предоставляемых услуг. Протокол SIP является основным сигнальным протоколом NGN, а качество предоставляемых услуг напрямую зависит от своевременной и надежной доставки по сети его сообщений. С развитием сетей NGN и ростом числа услуг наблюдается значительный рост числа пользователей, и, следовательно, возрастает нагрузка на SIP-сервера, которые, как и любой другой элемент сети, могут находиться в состоянии перегрузки. Перегрузки на SIP-серверах возникают из-за отсутствия достаточных ресурсов для обработки поступающих на сервер сообщений.

Стандарт протокола SIP предусматривает механизм борьбы с перегрузками с помощью сообщения 503 Service Unavailable [1]. Когда сервер находится в состоянии перегрузки и не в состоянии обработать запрос, он может уведомить об этом отправителя с помощью сообщения 503, а также может включить в него заголовок Retry-After с указанием временного интервала, по истечении которого отправитель сможет возобновить отправку сообщений. В процессе эксплуатации был выявлен ряд недостатков механизма 503 [2, 3].

В комитете IETF рабочей группой SOC для топологии сети типа «сервер-сервер» предложен межузловой механизм контроля перегрузок со сбросом сообщений на стороне отправителя (LBOC, Loss-based overload control) [4, 5], который должен заменить базовый механизм 503. Анализ документов рабочей группы показывает, что, несмотря на подробное описание принципов работы этого механизма, вопросы выбора параметров управления для механизма LBOC остаются за

9 Работа выполнена при поддержке РФФИ в рамках научных проектов № 15-07-03051, 15-07-03608.

рамками деятельности группы.

Алгоритмы межузлового контроля перегрузки, не могут полностью гарантировать стабильную работу сервера при больших всплесках трафика или при появлении новых источников трафика, поэтому помимо межузлового алгоритма применяют локальные алгоритмы контроля перегрузки [6-11]. Далее рассмотрим алгоритм с резервированием места для INVITE сообщений и порогом. Алгоритм позволяет управлять ресурсами сервера так, чтобы сервер при любой нагрузке мог продолжать принимать новые сессии и обслуживать текущие.

На систему с двумя буферами конечной ёмкости R_1 и R_2 и C приборами поступают два пуассоновских потока сообщений с интенсивностями λ_1 и λ_2 : сообщения типа INVITE с низким приоритетом, и non-INVITE-сообщения с высоким приоритетом обслуживания соответственно. Заявки обслуживаются в соответствии с дисциплиной FCFS по экспоненциальному закону с средним временем обслуживания μ^{-1}. Для обслуживания INVITE-сообщений зарезервированы C_1 приборов, оставшиеся $C_2=C-C_1$ приборов выполняют функцию полнодоступной группы. Если все C_2 приборов заняты, сообщения второго типа помещаются в буфер конечной ёмкости R_2. Также в модели установлено пороговое значение $1\le m<C_2$, с помощью которого обеспечивается эффективное распределение ресурсов системы при обслуживании обоих типов трафика. Так, сообщения первого типа могут занимать свободные приборы из C_2 -группы только в том случае, если R_2 буфер пуст и количество серверов, обслуживающих сообщения второго типа, меньше m .

Построенную систему, изображенную на рис. 1, в обозначениях Башарина-Кендала будем кодировать в виде $M \mid M \mid C \mid R_1, R_2 \mid f_0^1$.
$\lambda_1,\lambda_2 \quad \mu_1,\mu_2$

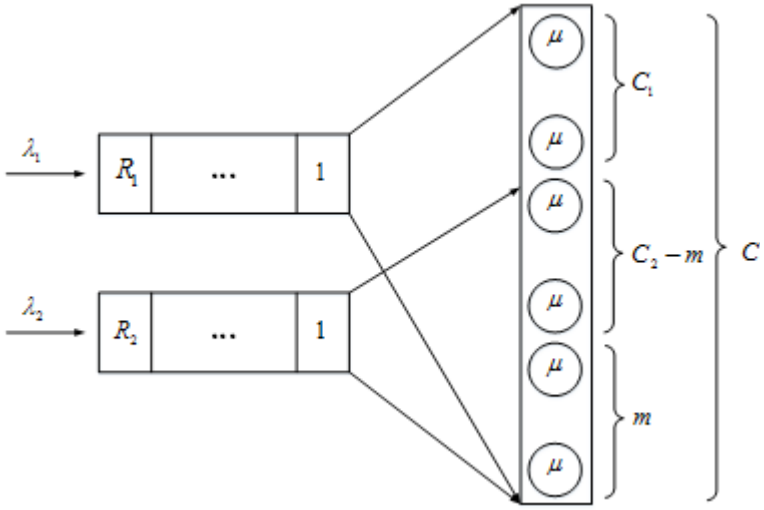


Рис. 1.. Схема модели обслуживания SIP-сообщений с резервированием и пороговым управлением

Составной случайный процесс $\mathbf{N}(t)=\{n_1(t),n_2(t),n_3(t),n_4(t)\}$, где $n_1(t)$ – число заявок первого типа в системе в момент времени t, $0\le n_1(t)\le R_1+C$, $n_2(t)$ – число заявок первого типа на приборах в момент времени t, $0\le n_2(t)\le C$, $n_3(t)$ – число заявок второго типа в системе в момент времени t, $0\le n_3(t)\le R_2+C_2$, $n_4(t)$ – число заявок второго типа на приборах в момент времени t, $0\le n_4(t)\le C_2$, по построению является марковским и описывает поведение исследуемой системы.

Обозначим $p(n_1,n_2,n_3,n_4)$, стационарное распределение вероятностей состояний МП $\mathbf{N}(t)$. Обозначим множество блокировок заявок первого типа

$$\mathcal{B}_1=\{(n_1,n_2,n_3,n_4):n_1-n_2=R_1\}, \quad (1)$$

$$\mathcal{B}_2=\{(n_1,n_2,n_3,n_4):n_3-n_4=R_2\} \quad (2)$$

С учетом введенных обозначений вероятности блокировок заявок первого и второго типа

$$B_i = \sum_{\mathbf{n}\in\mathcal{B}_i} \mathbf{n},\ i=1,2. \qquad (3)$$

Средняя длина очереди для заявок первого и второго типа соответственно равна

$$Q_1 = \sum_{\mathbf{n}\in\mathcal{N}:\, n_1-n_2>0} (n_1-n_2)\, p(n_1,n_2,n_3,n_4), \qquad (4)$$

$$Q_2 = \sum_{\mathbf{n}\in\mathcal{N}:\, n_3-n_4>0} (n_3-n_4)\, p(n_1,n_2,n_3,n_4) \qquad (5)$$

Среднее время пребывания заявки в очереди соответственно равно

$$W_i = \frac{Q_i}{\lambda_i(1-B_i)},\ i=1,2. \qquad (6)$$

Найдем стационарные вероятности численно, решив систему уравнений равновесия. Ненулевые элементы матрицы интенсивностей переходов представлены в таблице 1.

Таб.1. Матрица интенсивностей переходов

Элемент $(\mathbf{n}',\mathbf{n}'')$	$(\mathbf{n}',\mathbf{n}'')$	Условие
λ_1	$\mathbf{n}''=\mathbf{n}'+\mathbf{e_1}$	$(n_2+n_4=C)\wedge(n_1-n_2=R_1)$ $(n_3-n_4=0)\wedge(C_2>n_4\geq m)\wedge(n_2\geq C_1)\wedge(n_1-n_2<R_1)$
	$\mathbf{n}''=\mathbf{n}'+\mathbf{e_1}+\mathbf{e_2}$	$(n_3=0)\wedge(C-n_2>0)\wedge(n_1-n_2=0)$ $(n_3-n_4=0)\wedge(n_4<m)\wedge(C-n_2-n_4>0)\wedge(n_1-n_2=0)$ $(n_3-n_4=0)\wedge(m\leq n_4\leq C_2)\wedge(C_1-n_2>0)\wedge(n_1-n_2=0)$ $(n_3-n_4>0)\wedge(n_4=C_2)\wedge(C_1-n_2>0)\wedge(n_1-n_2=0)$
λ_2	$\mathbf{n}''=\mathbf{n}'+\mathbf{e_1}$	$(n_2+n_4=C)\wedge(n_3-n_4<R_2)$ $(n_4=C_2)\wedge(n_3-n_4<R_2)$
	$\mathbf{n}''=\mathbf{n}'+\mathbf{e_3}+\mathbf{e_4}$	$(n_2+n_4<C)\wedge(n_4<C_2)\wedge(n_3-n_4=0)$
$n_2\mu_1$	$\mathbf{n}''=\mathbf{n}'-\mathbf{e_1}-\mathbf{e_2}$	$(n_1-n_2=0)\wedge(n_3-n_4=0)$ $(n_3-n_4=0)\wedge(n_1-n_2>0)\wedge(n_4\geq m)\wedge(n_2\geq C_1)$ $(n_1-n_2=0)\wedge(n_3-n_4>0)\wedge(n_4=C_2)$
	$\mathbf{n}''=\mathbf{n}'-\mathbf{e_1}$	$(n_3-n_4=0)\wedge(n_1-n_2>0)\wedge(n_2<C_1)$ $(n_3-n_4=0)\wedge(n_1-n_2>0)\wedge(n_2\geq C_1)\wedge(n_4<m)$ $(n_1-n_2>0)\wedge(n_3-n_4>0)\wedge(n_4=C_2)$
	$\mathbf{n}''=\mathbf{n}'-\mathbf{e_1}-\mathbf{e_2}+\mathbf{e_4}$	$(n_3-n_4>0)\wedge(n_4<C_2)$
$n_4\mu_2$	$\mathbf{n}''=\mathbf{n}'-\mathbf{e_3}-\mathbf{e_4}$	$(n_3-n_4=0)\wedge(n_1-n_2=0)$ $(n_3-n_4=0)\wedge(n_1-n_2>0)\wedge(n_4>m)\wedge(n_2\geq C_1)$
	$\mathbf{n}''=\mathbf{n}'-\mathbf{e_3}$	$(n_3-n_4>0)\wedge(n_4\leq C_2)$
	$\mathbf{n}''=\mathbf{n}'-\mathbf{e_3}-\mathbf{e_4}+\mathbf{e_2}$	$(n_3-n_4=0)\wedge(n_1-n_2>0)\wedge(n_4>m)\wedge(n_2\geq C_1)$ $(n_1-n_2>0)\wedge(n_3-n_4=0)\wedge(n_4>m)\wedge(n_2<C_1)$

При проведении численного анализа были использованы следующие исходные данные: *C=20*, $\mu^{-1}=5$ мс. Введём обозначения: $\lambda=\lambda_1+\lambda_2$ – суммарная интенсивность поступления SIP-сообщений, B_1 – вероятность блокировки Non-Invite-сообщений, W_1 – среднее время ожидания обслуживания non-INVITE-сообщений, *UTIL* – средний коэффициент использования обслуживающих приборов, $\rho=\frac{\lambda_1+\lambda_2}{C\mu}$. На рис. 2 представлен график зависимости вероятности блокировки от интенсивности поступления сообщений и порогового значения m.

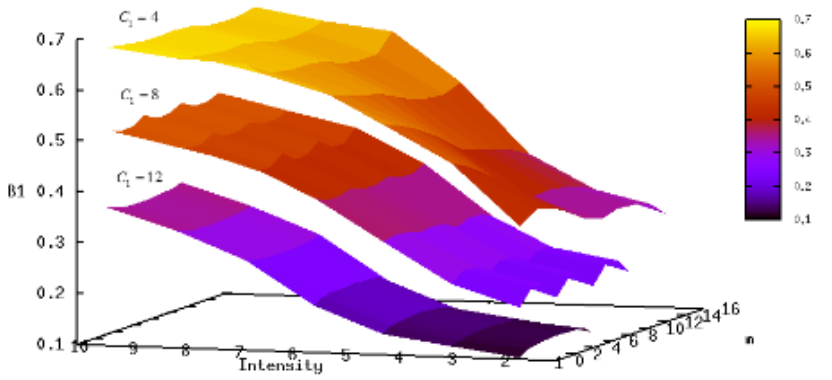


Рис. 2. Вероятность блокировки INVITE-сообщений

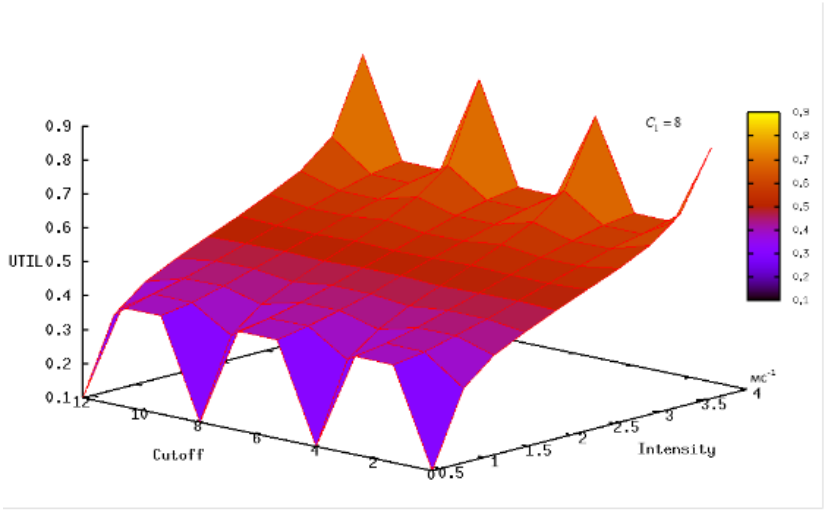


Рис. 3. Средний коэффициент использования обслуживающих приборов в системе при различных пороговых значениях

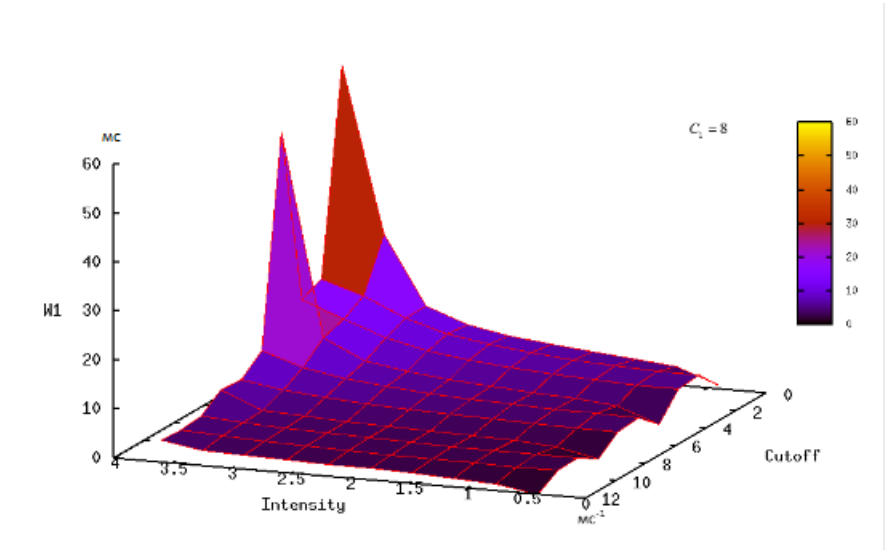


Рис. 4. Среднее время ожидания обслуживания non-INVITE-сообщений

Как видно из рис. 3, при интенсивности $\lambda \geq 4$ ($\rho \geq 1$) серверы используются более чем на 90%.

На рис. 3 и рис. 4 представлены графики коэффициента использования обслуживающих приборов и среднее время ожидания начала обслуживания non-INVITE сообщений.

Для рассмотренной СМО сформулируем следующую оптимизационную задачу, решение которой позволило бы определить оптимальные параметры системы для различной нагрузки.

В качестве ключевого параметра, подлежащего оптимизации, выберем *UTIL*, поскольку эффективное распределение нагрузки на приборы позволит повысить производительность всей системы, а также время ожидания W_1 как один из наиболее существенных параметров, влияющих на подсчет показателя качества обслуживания (QoS). Оптимизационная задача может быть записана следующим образом: $F = F_1 \circ F_2$, где

$$F_1 = \begin{cases} UTIL\left(C_1, m, \lambda\right) \to \min \\ R_1 : 0 \le \lambda \le 4 \\ R_2 : 0 \le B_1 \le 0{,}05 \\ R_3 : 0 \le B_2 \le 0{,}1 \end{cases} \quad \text{и } F_2 = W_1\left(C_1, m, \lambda\right) \to \min .$$

На рис. 5 и рис. 6 представлены основные вероятностные характеристики системы при оптимальных значениях параметров, полученных путем решения вышеизложенной задачи методом событийного имитационного моделирования.

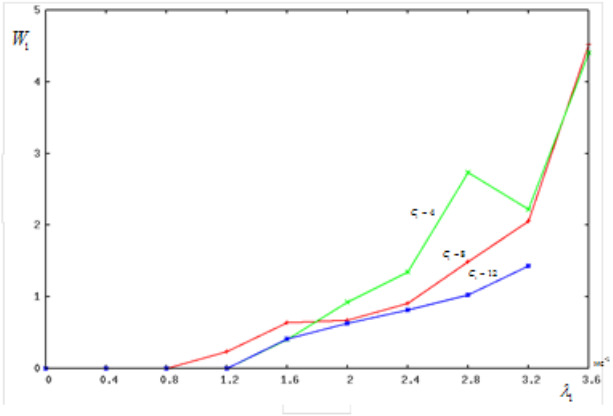


Рис. 5. Оптимальные значения среднего времени ожидания обслуживания non-INVITE-сообщений

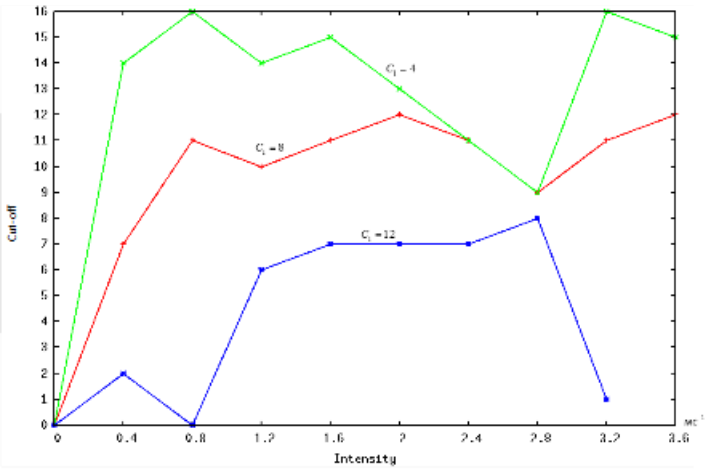


Рис.6. Оптимальные пороговые значения

Литература

1. J. Rosenberg, H. Schulzrinne, G. Camarillo et al. SIP: Session Initiation Protocol // RFC3261. – 2002.
2. Rosenberg J. Requirements for Management of Overload in the Session Initiation Protocol // RFC5390. – 2008.
3. V. Hilt, E. Noel, C. Shen, A. Abdelal Design Considerations for Session Initiation Protocol (SIP) Overload Control // RFC6357. – 2011.
4. Абаев П.О., Гайдамака Ю.В., Самуйлов К.Е. Гистерезисное управление сигнальной нагрузкой в сети SIP-серверов // Вестник Российского университета дружбы народов. «Математика. Информатика. Физика». Россия, Москва. –№ 4. – 2011. – С. 54-71.

5. Pavel O. Abaev, Yuliya V. Gaidamaka, Alexander V. Pechinkin, Rostislav V. Razumchik, Sergey Ya. Shorgin Simulation of overload control in SIP server networks // Proceedings of the 26th European Conference on Modelling and Simulation, ECMS 2012. – Germany, Koblenz. – 2012. – pp. 533-539.
6. Абаев П.О., Хачко А.В. К выбору эффективной стратегии обслуживания сообщений SIP-сервером // Тезисы докладов Всероссийской конференции с международным участием «Информационно-телекоммуникационные технологии и математическое моделирование высокотехнологичных систем». – Россия, Москва. – 2012. – С. 64-66.
7. Хачко А.В. Анализ вероятностных характеристик модели SIP-сервера с гистерезисным управлением и пульсирующим трафиком // Тезисы докладов Всероссийской конференции с международным участием «Информационно-телекоммуникационные технологии и математическое моделирование высокотехнологичных систем». – Москва. – 2013. – С. 108-110.
8. Абаев П.О., Хачко А.В. Разработка модели SIP-сервера в условиях перегрузки с пульсирующим трафиком // T Comm: Телекоммуникации и транспорт. – №11. – 2013. – С. 16-20.
9. Konstantin Samouylov, Pavel Abaev, Anastasiya Khachko Modelling of SIP server with hysteretic overload control and K-state MMPP input flow // XXXII International Seminar on Stability Problems for Stochastic Models. Norway, Trondheim. – 2014. pp. 127-129.
10. Pavel Abaev, Anastasia Khachko, Vitaliy Beschastny Queuing Model for SIP Server Hysteretic Overload Control with K-state MMPP Bursty Traffic // Статья в сборнике трудов Proc. of the 6th International Congress on Ultra Modern Telecommunications and Control Systems ICUMT-2014, Saint-Petersburg, Russia, October 6–8, 2014. – pp. 595-600. – IEEE. – 2014.
11. Abaev P., Razumchik R., Uglov I. Statistical Analysis and Modeling of SIP Traffic for Parameter Estimation of Server Hysteretic Overload Control // Journal of Telecommunications and Information Technology (JTIT). –Vol. 4. – 2013. Pp. 22-31.

Самуйлов К.Е.[1], Гайдамака Ю.В.[2], Сопин Э.С.[3], Горбунова А.В.[4]

[1] Российский университет дружбы народов, г. Москва, д.т.н., профессор, заведующий кафедры прикладной информатики и теории вероятностей, ksam@sci.pfu.edu.ru

[2] Российский университет дружбы народов, г. Москва, к.ф.-м.н., доцент кафедры прикладной информатики и теории вероятностей, ygaidamaka@mail.ru

[3] Российский университет дружбы народов, г. Москва, к.ф.-м.н., доцент кафедры прикладной информатики и теории вероятностей, esopin@sci.pfu.edu.ru

[4] Российский университет дружбы народов, г. Москва, аспирант кафедры прикладной информатики и теории вероятностей, avgorbunova@rambler.ru

АЛГОРИТМ ВЫЧИСЛЕНИЯ ПРЕОБРАЗОВАНИЯ ЛАПЛАСА-СТИЛТЬЕСА ДЛЯ ВРЕМЕНИ ОТКЛИКА СИСТЕМЫ ОБЛАЧНЫХ ВЫЧИСЛЕНИЙ С ГИСТЕРЕЗИСНЫМ УПРАВЛЕНИЕМ[10]

КЛЮЧЕВЫЕ СЛОВА

Облачные вычисления, время отклика, гистерезисное управление, динамическое масштабирование, пороговая система, преобразование Лапласа-Стилтьеса.

АННОТАЦИЯ

С развитием технологий сетей связи последующих поколений системы облачных вычислений превратились в высокопроизводительную вычислительную среду. Они обладают хорошей масштабируемостью и простотой в обслуживании, что приводит к быстрому росту числа облачных приложений и поставщиков облачных услуг. Правильное определение нужного количества ресурсов, необходимого для выполнения задачи с учетом текущей рабочей нагрузки и технических требований в рамках соглашения о качестве услуг является сложной задачей. Это обусловлено тем, что, с одной стороны, выделение большего числа виртуальных машин, чем необходимо, ведет к большим затратам для пользователя, а с другой стороны, выделение меньшего, чем требуется, объема ресурсов, ведет к нарушению соглашения об уровне обслуживания (англ. Service Level Agreement, SLA). Таким образом, поскольку для поставщиков облачных услуг важным показателем производительности системы является энергоэффективность, для уменьшения числа срабатываний системы управления масштабированием применяется в том числе и гистерезисный подход. Для анализа показателей производительности модели системы облачных вычислений с динамическим масштабированием в статье рассмотрена многолинейная система массового обслуживания с гистерезисным управлением количеством включенных приборов на основе порогов длины очереди и немгновенным подключением дополнительных приборов, для которой разработан рекуррентный метод вычисления преобразования Лапласа-Стилтьеса распределения времени пребывания заявки в системе и времени ожидания начала обслуживания.

Введение

Под облачными вычислениями (англ. cloud computing) обычно понимается предоставление пользователю компьютерных ресурсов и мощностей в виде интернет-сервиса [1]. Системы облачных вычислений применяются для хранения и обработки данных, для распределенных вычислений при решении научных и коммерческих задач. Современные облачные системы проектируются, как правило, масштабируемыми, что позволяет системе справляться с высокой нагрузкой и иметь возможность снизить энергопотребление в периоды снижения нагрузки.

Проблема недостаточного использования ресурсов возникает вследствие того, что их распределение обычно происходит в соответствии с пиковой нагрузкой, которая может длиться только в течение короткого периода времени. Затем происходит переход в состояние простоя, но

10 Исследование выполнено при частичной финансовой поддержке РФФИ в рамках научных проектов № 14-07-00090, 15-07-03051, 15-07-0360.

при этом потребляемая мощность может по-прежнему составлять около 50% от пиковой, что вызывает значительные потери энергии.

Вообще говоря, методы повышения энергетической эффективности и энергосбережения можно условно разделить на три категории: изменение рабочей нагрузки, влияние на поведение пользователей, перенастройка ресурсов. Идея первого заключается в использовании дополнительного устройства между сервисом и пользователем, которое изменяет интенсивность входящего трафика посредством буферизации: увеличивает продолжительность «спящего» режима и удлиняет периоды высокой загрузки [13]. Если же говорить о втором подходе, то положительный опыт внедрения мер стимулирования потребителей к энергосбережению имеется, в частности, у энергетических компаний. Применение системы тарификации, предполагающей значительное снижение стоимости электроэнергии в периоды низкой активности, например, ночью, побуждает потребителей к использованию электричества хотя бы отчасти в ночное время, что в свою очередь, приводит к выравниванию рабочей нагрузки. Однако у этих методов имеется существенный недостаток, который заключается в снижении качества оказываемых услуг, что является нарушением SLA.

Одним из способов реализации третьего подхода является динамическая активация виртуальных машин [5,14,15]. Иными словами, регулируется количество ресурсов (серверов, машин) в зависимости от текущей нагрузки [7-10,16]. При анализе таких систем применяются модели с пороговым управлением обслуживанием, в том числе, с гистерезисным управлением количеством включенных приборов [4, 6]. При этом основной проблемой при моделировании облачных систем с помощью методов теории телетрафика является высокая вычислительная сложность получаемых в результате алгоритмов [2,11,12]. В [6] исследована система с гистерезисным управлением обслуживанием для анализа предоставления услуги видео по требованию, стационарные характеристики системы получены с помощью матричных методов, которые не применимы для анализа современных облачных платформ из-за высокой вычислительной сложности. В [14, 15] система с гистерезисным управлением обслуживанием применена для анализа облачных систем с конечным числом приборов, а для случая трех приборов получен эффективный вычислительный алгоритм расчета стационарных характеристик функционирования облачной системы, имеющий линейную сложность. Данная статья является развитием работ [14, 15], а также продолжением работы [3], в которой был разработан эффективный алгоритм вычисления стационарных вероятностей и показателей качества функционирования системы.

Математическая модель

Рассматривается система облачных вычислений с гистерезисным подключением и отключением дополнительных виртуальных машин в виде многолинейной системы массового обслуживания с K приборами, часть которых может быть не активна, и конечной емкостью системы R. На систему поступает пуассоновский поток заявок с параметром λ. Считаем, что приборы являются однородными, время обслуживания распределено по экспоненциальному закону с параметром μ [3].

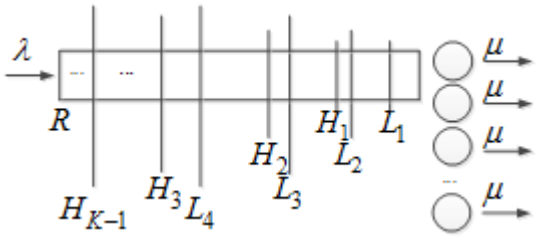


Рис. 1. Многолинейная система массового обслуживания с порогами H и L

В пустой системе активным, т.е. готовым при поступлении заявки мгновенно начать ее обслуживание, является один прибор. При поступлении заявок в систему активация приборов происходит не мгновенно, при этом количество активных приборов определяется числом заявок в очереди, в которой установлены парные пороги, заданные значениями векторов $\mathbf{H}=(H_1, H_2, \ldots, H_{K-1})$, $H_1<H_2<\ldots<H_{K-1}$ и $\mathbf{L}=(L_1, L_2, \ldots, L_{K-1})$, $L_1<L_2<\ldots<L_{K-1}$, где $L_{i+1}<H_i<L_{i+2}$, $i=\overline{1,K-1}$. Очередь имеет дисциплину FCFS (First Come First Served). При поступлении на прибор заявка сохраняет место в очереди. Правила работы системы следующие:

1.Если в системе уже есть H_i заявок, то при поступлении новой заявки активируется

(подключается) один $(i+1)$ –й дополнительный прибор, но не мгновенно, а через случайное время, имеющее экспоненциальное распределение с параметром α ;

2.Если в системе находится L_i заявок и при этом одна заявка обслужилась, то $(i+1)$ –й прибор мгновенно отключается, либо, если он не был активен, останавливается процедура его активации.

Функционирование системы описывается Марковским процессом $X(t)$ с множеством состояний

$$S=\left\{(k,i,n)\left|\begin{array}{l}0\le n\le H_1,\ k=1,\ i=1\\ L_{k-1}\le n\le H_k,\ k=\overline{2,\ K-1},\ \ i=\overline{1,\ K-1}\\ L_{k-1}\le n\le R,\ k=K,\ i=\overline{1,\ K}\end{array}\right.\right\},$$

где k – необходимое количество приборов; i – количество активированных приборов; n – количество заявок в очереди. Для выбранного расположения пороговых значений относительно друг друга диаграмма интенсивностей переходов для системы с $K=4$ приборами представлена на рисунке 2.

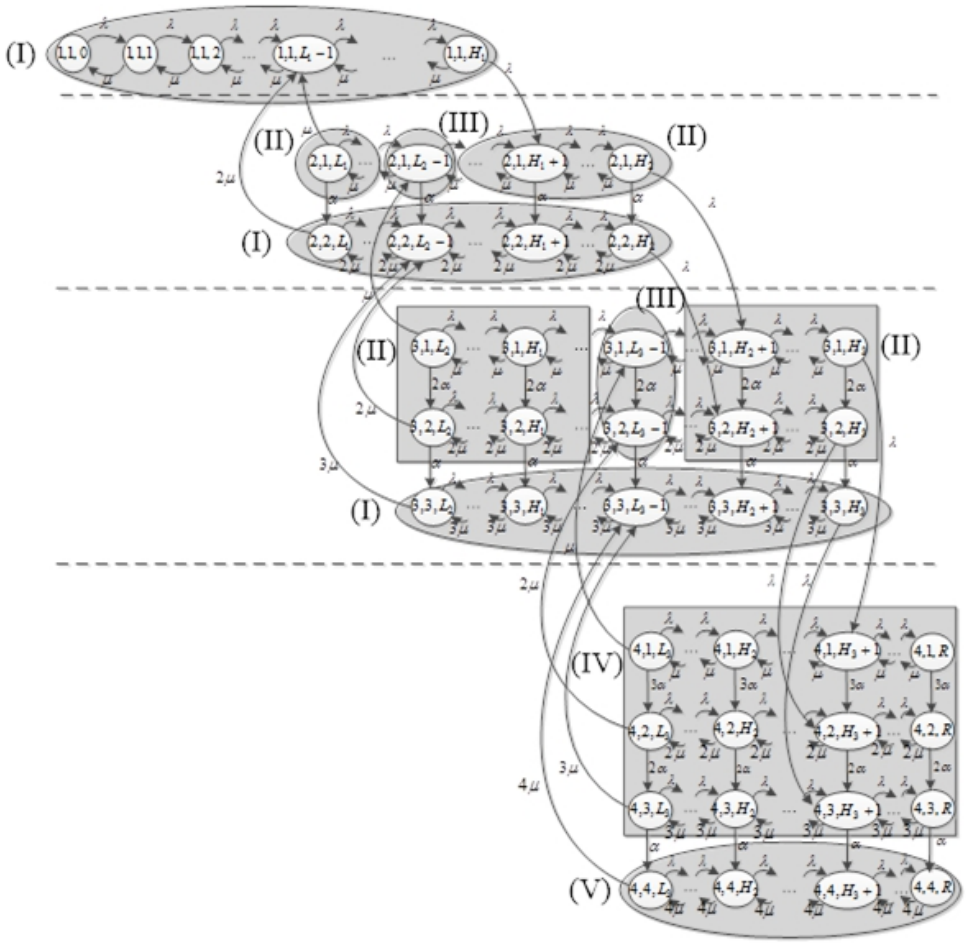


Рис. 2. Диаграмма интенсивностей переходов для системы с K=4 приборами

На диаграмме выделены состояния системы (I)-(V), для которых различаются соответствующие формулы вычисления стационарных вероятностей в случае выбора описанного выше расположения порогов.

Алгоритм вычисления преобразования Лапласа-Стилтьеса времени отклика системы

Для того чтобы составить объективное мнение о поведении времени отклика системы, недостаточно знать только среднее значение этой случайной величины. Для всей полноты картины, а также для решения актуальных практических задач наряду с математическим ожиданием желательно иметь представление о значениях дисперсии и квантили. В этой связи

определение функции распределения времени отклика – в нашем случае в терминах преобразования Лапласа-Стилтьеса (ПЛС) – приобретает особую важность.

Обозначим $V^m_{k,i,n}(s)$ – ПЛС времени ожидания m-й в очереди заявки, если система находится в состоянии (k,i,n). В случае, когда $m \le i$, время ожидания равно нулю, а значит

$$V^m_{k,i,n}(s) = 1, \quad m \le i. \tag{1}$$

В остальных случаях справедливы следующие рекуррентные соотношения:

$$V^m_{k,k,n}(s) = \frac{k\mu}{\lambda + k\mu} \cdot \frac{k\mu}{k\mu + s} V^{m-1}_{k,k,n-1}(s) + \frac{\lambda}{\lambda + k\mu} \cdot \frac{\lambda}{\lambda + s} V^m_{k,k,n+1}(s), \tag{2}$$
$$1 \le k \le K,\ L_{k-1} < n < H_k;$$

$$V^m_{k,k,L_{k-1}}(s) = \frac{k\mu}{\lambda + k\mu} \cdot \frac{k\mu}{k\mu + s} V^{m-1}_{k-1,k-1,L_{k-1}-1}(s) + \tag{3}$$
$$+ \frac{\lambda}{\lambda + k\mu} \cdot \frac{\lambda}{\lambda + s} V^m_{k,k,L_{k-1}+1}(s),\ \ 2 \le k \le K;$$

$$V^m_{k,k,H_k}(s) = \frac{k\mu}{\lambda + k\mu} \cdot \frac{k\mu}{k\mu + s} V^{m-1}_{k,k,H_k-1}(s) + \tag{4}$$
$$+ \frac{\lambda}{\lambda + k\mu} \cdot \frac{\lambda}{\lambda + s} V^m_{k+1,k+1,H_k+1}(s),\ \ 1 \le k \le K-1;$$

$$V^m_{K,K,R}(s) = \frac{K\mu}{K\mu + s} V^{m-1}_{K,K,R-1}(s); \tag{5}$$

$$V^m_{k,i,n}(s) = \frac{i\mu}{\lambda + i\mu + (k-i)\alpha} \cdot \frac{i\mu}{i\mu + s} V^{m-1}_{k,i,n-1}(s) + \tag{6}$$
$$+ \frac{\lambda}{\lambda + i\mu + (k-i)\alpha} \cdot \frac{\lambda}{\lambda + s} V^m_{k,i,n+1}(s) +$$
$$+ \frac{(k-i)\alpha}{\lambda + i\mu + (k-i)\alpha} \cdot \frac{(k-i)\alpha}{(k-i)\alpha + s} V^m_{k,i+1,n}(s),$$
$$1 \le k \le K,\ 1 \le i < k,\ L_{k-1} < n < H_k;$$

$$V^m_{k,i,L_{k-1}}(s) = \frac{i\mu}{\lambda + i\mu + (k-i)\alpha} \cdot \frac{i\mu}{i\mu + s} V^{m-1}_{k-1,i,L_{k-1}-1}(s) + \tag{7}$$
$$+ \frac{\lambda}{\lambda + i\mu + (k-i)\alpha} \cdot \frac{\lambda}{\lambda + s} V^m_{k,i,L_{k-1}+1}(s) +$$
$$+ \frac{(k-i)\alpha}{\lambda + i\mu + (k-i)\alpha} \cdot \frac{(k-i)\alpha}{(k-i)\alpha + s} V^m_{k,i+1,L_{k-1}}(s),$$
$$2 \le k \le K,\ 1 \le i < k;$$

$$V^m_{k,i,H_k}(s) = \frac{i\mu}{\lambda + i\mu + (k-i)\alpha} \cdot \frac{i\mu}{i\mu + s} V^{m-1}_{k,i,H_k-1}(s) + \tag{8}$$
$$+ \frac{\lambda}{\lambda + i\mu + (k-i)\alpha} \cdot \frac{\lambda}{\lambda + s} V^m_{k+1,i,H_k+1}(s) +$$
$$+ \frac{(k-i)\alpha}{\lambda + i\mu + (k-i)\alpha} \cdot \frac{(k-i)\alpha}{(k-i)\alpha + s} V^m_{k,i+1,H_k}(s),$$
$$1 \le k \le K-1,\ 1 \le i < k;$$

$$V^{m}_{K,i,R}(s) = \frac{i\mu}{i\mu + (K-i)\alpha} \cdot \frac{i\mu}{i\mu + s} V^{m-1}_{K,i,R-1}(s) + \\ + \frac{(K-i)\alpha}{i\mu + (K-i)\alpha} \cdot \frac{(K-i)\alpha}{(K-i)\alpha + s} V^{m}_{K,i+1,R}(s),\ 1 \le i < K. \quad (9)$$

На основе рекуррентных соотношений (1)–(9) может быть составлен алгоритм для вычисления ПЛС $V^{m}_{k,i,n}(s)$, с помощью которого можно найти ПЛС $V(s)$ времени ожидания и ПЛС $W(s)$ времени пребывания заявки в системе:

$$V(s) = \sum_{(k,i,n)\in S} \pi_{k,i,n} V^{n}_{k,i,n}(s); \quad (10)$$

$$W(s) = \frac{\mu}{\mu + s} V(s), \quad (11)$$

где $\pi_{k,i,n}$ – стационарные вероятности для соответствующих состояний (k,i,n), алгоритм вычисления которых приведен в [3].

Заключение

Разработанный метод расчета временных характеристик модели системы облачных вычислений в терминах преобразования Лапласа-Стилтьеса в сочетании с полученными ранее результатами позволяет с высокой точностью получить оценки важнейших показателей качества обслуживания, на которые следует ориентироваться при расчете и проектировании облачных систем. Однако, для реальных систем облачных вычислений, где вычислительные облака включают тысячи серверов и обеспечивают ресурсами десятки тысяч приложений, которые одновременно используют миллионы пользователей, размерность пространства состояний модели создает вычислительные трудности, даже несмотря на линейную вычислительную сложность предложенного в статье алгоритма. Поэтому задачей дальнейших исследований является разработка приближенного метода оценки показателей производительности модели системы облачных вычислений.

Авторы благодарят аспиранта кафедры прикладной информатики и теории вероятностей РУДН Таланову Маргариту за помощь в оформлении работы.

Литература

1. ETSI Cloud Standards Coordination. Final Report 2013, ver. 1.0. Available at: http://www.etsi.org/images/files/Events/2013/2013_CSC_Delivery_WS/CSC-Final_report-013-CSC_Final_report_v1_0_PDF_format-.PDF.
2. Basharin G.P., Gaidamaka Yu.V., Samuilov K.E. Mathematical teletraffic theory and its application to the analysis of the next generations multiservice networks // Automatic control and computer sciences, Latvia, Riga: Institute of electronics and computer science of the Latvian university. - 2013. - No. 2. - Pp. 11-21.
3. Гайдамака Ю.В., Сопин Э.С., Таланова М.О. Анализ показателей качества функционирования систем облачных вычислений с гистерезисным управлением // T-Comm - Телекоммуникации и Транспорт. – в печати.
4. Гайдамака Ю.В., Печинкин А.В., Разумчик Р.В., Самуйлов А.К., Самуйлов К.Е., Соколов И.А., Сопин Э.С., Шоргин С.Я. Распределение времени выхода из множества состояний перегрузки в системе M | M | 1 | <L,H> | <H,R> с гистерезисным управлением нагрузкой // Информатика и ее применения. - 2013. - Т. 7, Вып. 4. - С. 20-33.
5. Goswami V., Patra S.S., Mund G..B. Performance Analysis of Cloud with Queue-Dependent Virtual Machines // Proc. of 1st Int'l Conf. on Recent Advances in Information Technology, Dhanbad, India. - 2012. - Pp. 357-362.
6. Golubchik L., Lui J.C.S. Bounding of Performance Measures for Threshold-Based Queuing Systems: Theory and Application to Dynamic Resource Management in Video-on-Demand Servers // IEEE Trans. Computers. – 2002. - Vol. 51, No. 4. - Pp. 353-372.
7. Kaxiras S., Martonosi M. Computer Architecture Techniques for Power-Efficiency // Synthesis Lectures on Computer Architecture. - 2008. - Vol. 3, No. 1. - Pp. 1-207.
8. Lin M., Wierman A., Andrew L.L.H., Thereska E. Dynamic right-sizing for power-proportional data centers // INFOCOM, Proceedings IEEE. - 2011. - Pp. 1098–1106.
9. Meisner D., Gold B.T., Wenisch T.F. Powernap: eliminating server idle power // CM SIGPLAN Notices. – 2009. - Vol. 44. - Pp. 205–216.
10. Miyoshi A., Lefurgy C., Hensbergen E.V., Rajamony R., Rajkumar R., Critical power slope: Understanding the runtime effects of frequency scaling // Proceedings of the 16th Annual ACM International Conference on Supercomputing. - 2002. - Pp 35–44.
11. Мокров Е.В., Чукарин А.В. Анализ показателей эффективности системы облачных вычислений с миграцией серверов // T-Comm – Телекоммуникации и Транспорт. - 2014. - Т. 8, № 8. - С. 64-67.
12. Мокров Е.В., Самуйлов К.Е. Модель системы облачных вычислений в виде системы массового обслуживания с несколькими очередями и с групповым поступлением заявок // T-comm – Телекоммуникации и транспорт. - 2013. - № 11, С. 139-141.

13. Nedevschi S., Popa L., Iannaccone G., Ratnasamy S., Wetherall D. Reducing network energy consumption via sleeping and rate-adaptation // Proceedings of the 5th USENIX Symposium on Networked Systems Design and Implementation, NSDI'08, Berkeley, CA, USA. USENIX Association. - 2008. - Pp. 323–336.
14. Печинкин А.В., Гайдамака Ю.В., Сопин Э.С., Таланова М.О. Анализ показателей эффективности функционирования системы облачных вычислений с динамическим масштабированием // Сборник избранных трудов IX Международной ежегодной научно-практической конференции «Современные информационные технологии и ИТ-образование». - 2014. - С. 395-406.
15. Shorgin S.Y., Pechinkin A.V., Samouylov K.E., Gaidamaka Y.V., Gudkova I.A., Sopin E.S. Threshold-based Queuing System for Performance Analysis of Cloud Computing System with Dynamic Scaling // Proc. of the 12th International Conference of Numerical Analysis and Applied Mathematics ICNAAM-2014, Rhodes, Greece, 2014, USA, AIP Publishing - 2015. - Vol. 1648. - Pp. 1–3.
16. Wu Q., Juang P., Martonosi M., Peh L.-S., Clark D.W. // Formal control techniques for power performance management / IEEE Micro. - 2005. - Vol. 25. - Pp. 52–62.

Самуйлов К.Е.[1], Гудкова И.А.[2], Острикова Д.Ю.[3]

[1] Российский университет дружбы народов, г. Москва,
д.т.н., заведующий кафедрой прикладной информатики и теории вероятностей,
ksam@sci.pfu.edu.ru

[2] Российский университет дружбы народов, г. Москва,
к.ф.-м.н., доцент кафедры прикладной информатики и теории вероятностей,
igudkova@sci.pfu.edu.ru

[3] Российский университет дружбы народов, г. Москва,
ассистент кафедры прикладной информатики и теории вероятностей, dyostrikova@sci.pfu.edu.ru

АНАЛИЗ ВЕРОЯТНОСТИ ПРЕРЫВАНИЯ ОБСЛУЖИВАНИЯ В МОДЕЛИ СЕТИ LTE С СОВМЕСТНЫМ ИСПОЛЬЗОВАНИЕМ РЕСУРСОВ[11]

КЛЮЧЕВЫЕ СЛОВА

LTE, LSA, СМО с ненадежными приборами, вероятность прерывания обслуживания

АННОТАЦИЯ

В современных беспроводных сетях увеличение спроса на высокоскоростные мультимедийные услуги приводит к проблеме нехватки радиоресурсов. Решение проблемы может быть достигнуто посредством совместного использования ресурсов владельцем и арендатором-оператором под контролем третьей стороны. Правила взаимодействия трех сторон определяет регулирующая система совместного использования лицензированных частот (Licensed Shared Access, LSA). Для оператора ресурсы совестного использования вре́менно доступны, что может приводить к прерыванию обслуживания пользователей при возврате ресурсов их владельцу. Авторами построена математическая модель схемы доступа к ресурсам совместного использования в виде системы массового обслуживания (СМО) с одновременными отказами ненадежных приборов, надежными приборами и конечной очередью. Проведен расчет и анализ вероятностных характеристик прерывания обслуживания пользователей.

Введение

В настоящее время в мире продолжается активное распространение телекоммуникационных беспроводных сетей последующих поколений на базе технологии LTE (Long Term Evolution) [1], поддерживающих высокие скорости передачи данных, что позволяет операторам предоставлять широкий перечень мультимедийных услуг. Стремительный рост пользователей беспроводных сетей, а также числа интеллектуальных устройств (от светофоров до медицинского оборудования), способных подключаться к сети без участия человека, увеличение спроса на высокоскоростные услуги, такие как, например, видео с высоким качеством разрешения, приводят к экспоненциальному росту трафика, передаваемого в беспроводных сетях. При этом увеличиваются требования пользователей к качеству предоставляемых услуг. Описанные тенденции и ограниченность частотного спектра неизбежно приводят к проблеме нехватки радиоресурсов в сетях четвертого поколения (4th Generation, 4G). Решение проблемы может быть достигнуто как за счет более эффективного использования имеющихся ресурсов, так и посредством привлечения дополнительных.

Подход, основанный на привлечении дополнительных ресурсов, может быть реализован посредством совместного их использования владельцем и арендатором-оператором под контролем третьей стороны. Правила взаимодействия трех сторон определяет регулирующая система совместного использования лицензированных частот LSA [2,3]. Абсолютный приоритет доступа к ресурсам совместного использования имеет их владелец. Для оператора они вре́менно доступны, что может приводить к прерыванию обслуживания пользователей при возврате

11 Исследование выполнено при финансовой поддержке РФФИ в рамках научных проектов №№ 13-07-00953, 15-07-03608, и при поддержке Фонда содействия развитию малых форм предприятий в научно-технической сфере, грант № 6264ГУ2015.

ресурсов их владельцу. Следовательно, с точки зрения оператора, наибольший интерес представляет анализ именно показателей прерывания обслуживания. Ввиду изложенного, актуальной является задача оптимального распределения нагрузки между ресурсами совместного и индивидуального использования оператора, при котором показатели прерывания обслуживания пользователей были бы минимальны.

Авторами построена математическая модель схемы доступа к ресурсам совместного использования в виде СМО с одновременными отказами ненадежных приборов, надежными приборами и конечной очередью, а также получены формулы для расчета вероятностных характеристик прерывания обслуживания и проведен их численный анализ. При этом в построенной модели рассматривается трафик, толерантный к задержкам [4]. Модель с трафиком, не толерантным к задержкам, исследована в [5]. Необходимо отметить, что модель с одной полосой совместного использования и трафиком, толерантным к задержкам, ранее исследовалась в [6]. В работах [7,8] исследования схем доступа к ресурсам совместного использования проводились методами имитационного моделирования.

Математическая модель

На систему, состоящую из C_1 надежных приборов и C ненадежных приборов, а также очереди размера r, поступает пуассоновский поток заявок с интенсивностью λ. Заявки занимают сначала надежные приборы. В случае отсутствия свободных надежных приборов, заявки занимают ненадежные приборы. Предполагается, что находящиеся на ненадежных приборах заявки сохраняют за собой место в очереди. При этом ненадежные приборы могут одновременно отказывать с интенсивностью α, а затем восстанавливаться через некоторое время с интенсивностью β. Время безотказной работы и время восстановления распределены по экспоненциальному закону. Если в момент поступления заявки в системе есть свободные приборы, то заявка занимает один прибор и обслуживается на нем в течение случайного времени, имеющего экспоненциальное распределение с параметром μ. Если же в момент поступления заявки в системе нет свободных приборов для ее обслуживания, то заявка помещается в очередь. В случае если очередь полна, поступившая заявка теряется. Если ненадежные приборы отказывают, то обслуживающиеся на них заявки занимают свободные надежные приборы и на них дообслуживаются. В случае отсутствия свободных мест на надежных приборах заявки переходят в очередь для ожидания. После восстановления ненадежных приборов, либо появления свободных мест на надежных приборах, заявки продолжают обслуживаться с того места, на котором были прерваны.

Пусть $N_1(t)$ – число заявок на надежных приборах, $N(t)$ – число заявок на ненадежных приборах, $M(t)$ – число заявок в очереди, $S(t)$ – состояние ненадежных приборов в момент t: $S(t)=1$, если все приборы исправны, $S(t)=0$, если все приборы неисправны. Тогда функционирование системы описывает составной марковский случайный процесс (МП) $\{(N_1(t), N(t), M(t), S(t)), t \geq 0\}$ над пространством состояний (1):

$$Z = \{n_1 = 0,\ldots,C_1,\ n = 0,\ldots,C,\ m = 0,\ s = 1;\ n_1 = C_1,\ n = C,\ m = 1,\ldots,r-C,\ s = 1;$$
$$n_1 = 0,\ldots,C_1,\ n = 0,\ m = 0,\ s = 0;\ n_1 = C_1,\ n = 0,\ m = 1,\ldots,r,\ s = 0\}. \quad (1)$$

На рис. 1 представлена структура пространства состояний.

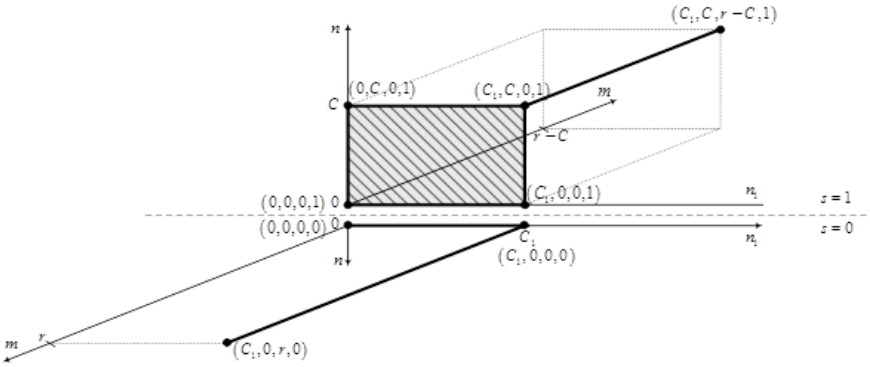


Рис.1. Структура пространства состояний МП $\{(N_1(t), N(t), M(t), S(t)), t \geq 0\}$

Описывающий функционирование системы МП $\{(N_1(t), N(t), M(t), S(t)), t \geq 0\}$ не является обратимым. Стационарное распределение вероятностей $\mathbf{p} = (p(n_1, n, m, s),\ (n_1, n, m, s) \in Z)$ состояний может быть найдено с помощью численного решения СУР $\mathbf{p}^T \cdot \mathbf{A} = \mathbf{0}^T$ с матрицей $\mathbf{A}$ интенсивностей переходов, элементы которой задаются формулой (2):

$$a\big((n_1,n,m,s),(n_1',n',m',s')\big) =$$

$$= \begin{cases} \lambda, & n_1' = n_1 + 1, & n_2' = n_2, & m' = m, & s' = s & \text{или} \\ & n_1' = n_1 = C_1, & n' = n+1, & m' = m, & s' = s = 1 & \text{или} \\ & n_1' = n_1 = C_1, & n' = n, & m' = m+1, & s' = s, & \\ n_1\mu, & n_1' = n_1 - 1, & n' = n, & m' = m = 0, & s' = s, & \\ n\mu, & n_1' = n_1, & n' = n-1, & m' = m = 0, & s' = s = 1, & \\ C_1\mu, & n_1' = n_1 = C_1, & n' = n = 0, & m' = m-1, & s' = s = 0, & \\ (C_1 + C)\mu, & n_1' = n_1 = C_1, & n' = n = 0, & m' = m-1, & s' = s = 1, & \\ \alpha, & n_1' = n_1, & n' = n = 0, & m' = m = 0, & s' = s-1 & \text{или} \\ & n_1' = n_1 + n, & n' = 0, & m' = m = 0, & s' = s-1 & \text{или} \\ & n_1' = C_1, & n' = 0, & m' = m + n - & s' = s-1, & \\ & & & -C_1 + n_1, & & \\ \beta, & n_1' = n_1, & n' = n = 0, & m' = m = 0, & s' = s+1 & \text{или} \\ & n_1' = n_1 = C_1, & n' = n + m, & m' = 0, & s' = s+1 & \text{или} \\ & n_1' = n_1 = C_1, & n' = C, & m' = m - C, & s' = s+1, & \\ *), & n_1' = n_1, & n' = n, & m' = m, & s' = s, & \\ 0, & \text{иначе}, & & & & \end{cases} \quad (2)$$

$$*) - \big(\lambda \cdot 1\{n_1 + n + m < C_1 + r\} + (n_1 + n)\mu + \alpha s + \beta(1-s)\big).$$

Вероятностные характеристики

Основными вероятностными характеристиками рассматриваемой модели являются:

1. Вероятность I_1 прерывания обслуживания хотя бы одной заявки, т.е. вероятность того, что в случае изъятия полосы совместного использования на полосе индивидуального использования не хватит ресурсов для принятия на обслуживание хотя бы одного пользователя

$$I_1 = \sum_{n=1}^{C} \sum_{n_1 = C_1 - n + 1}^{C_1} \frac{\alpha}{\alpha + \lambda + (n + n_1)\mu} p(n_1, n, 0, 1) + \\ + \sum_{m=1}^{r-C-1} \frac{\alpha}{\alpha + \lambda + (C + C_1)\mu} p(C_1, C, m, 1) + \frac{\alpha}{\alpha + (C + C_1)\mu} p(C_1, C, r - C, 1); \quad (3)$$

2. Вероятность I_2 прерывания обслуживания конкретной заявки, т.е. вероятность того, что в случае изъятия полосы совместного использования на полосе индивидуального использования не хватит ресурсов для принятия на обслуживание конкретного пользователя

$$I_2 = \sum_{n=1}^{C} \sum_{n_1 = C_1 - n + 1}^{C_1 - 1} \frac{n - C_1 + n_1}{n} \cdot \frac{\alpha}{\alpha + \lambda + (n + n_1)\mu} p(n_1, n, 0, 1) + \\ + \sum_{n=1}^{C} \frac{\alpha}{\alpha + \lambda + (n + C_1)\mu} p(C_1, n, 0, 1) + \sum_{m=1}^{r-C-1} \frac{\alpha}{\alpha + \lambda + (C + C_1)\mu} p(C_1, C, m, 1) + \quad (4) \\ + \frac{\alpha}{\alpha + (C + C_1)\mu} p(C_1, C, r - C, 1);$$

3. Вероятность E_1 того, что обслуживание ни одной заявки на ненадежных приборах не

будет прервано при их отказе, т.е. вероятность того, что в случае изъятия полосы совместного использования на полосе индивидуального использования хватит ресурсов для принятия всех пользователей, находящихся на обслуживании на полосе совместного использования

$$E_1 = \sum_{n=0}^{C} \sum_{n_1=0}^{C_1-n} \frac{\alpha}{\alpha+\lambda+(n+n_1)\mu} p(n_1, n, 0, 1); \quad (5)$$

4. Вероятность E_2 того, что обслуживание конкретной заявки на ненадежном приборе не будет прервано при его отказе, т.е. вероятность того, что в случае изъятия полосы совместного использования на полосе индивидуального использования хватит ресурсов для принятия конкретного пользователя, находящегося на обслуживании на полосе совместного использования

$$E_2 = \sum_{n=1}^{C} \sum_{n_1=0}^{C_1-n} \frac{\alpha}{\alpha+\lambda+(n+n_1)\mu} p(n_1, n, 0, 1) + \\ + \sum_{n=1}^{C} \sum_{n_1=C_1-n+1}^{C_1-1} \frac{C_1-n_1}{n} \cdot \frac{\alpha}{\alpha+\lambda+(n+n_1)\mu} p(n_1, n, 0, 1). \quad (6)$$

Численный анализ

Для проведения численного анализа за основу исходных данных был взят сценарий совместного использования частот на примере аэропорта (владельца частот) и арендатора-оператора. Аэропорт сдает в аренду оператору часть диапазона частот на всей своей территории покрытия за вычетом территории аэропорта и близлежащей к нему зоны (лесополосы вокруг аэропорта) (рис. 2). Предполагается, что самолеты взлетают через определенные промежутки времени, под определенным углом, с некоторой начальной скоростью и ускорением. Вокруг самолета существует радиотень определенного радиуса, в пределах которой внешние сигналы создают помехи передатчику самолета. Поэтому основным условием, налагаемым владельцем на арендатора, является ограничение интерференции в сотах, через которые проходит радиотень самолета. В соответствии с данным условием, во время прохождения радиотени через территорию соты пользователи не могут использовать полосу частот совместного использования для передачи данных.

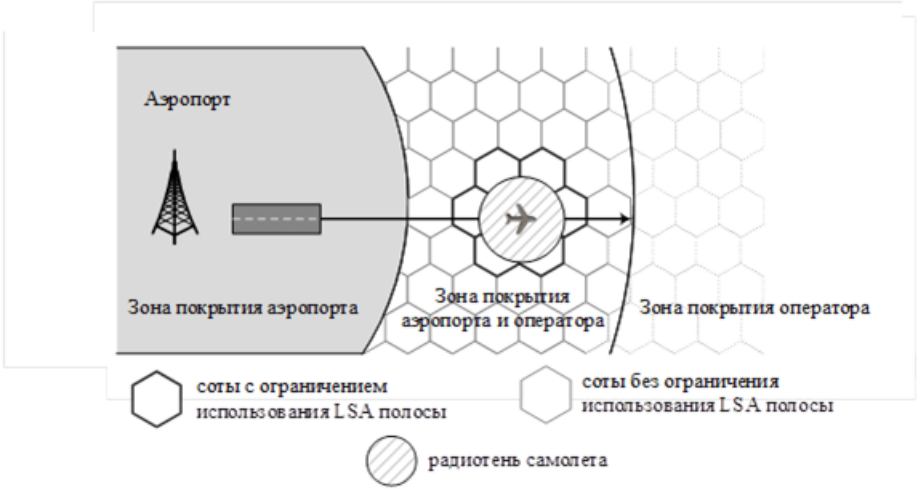


Рис.2. Сценарий совместного использования частот на примере аэропорта и оператора

Рассмотрим соту с полосами частот индивидуального и совместного использования шириной 10 МГц и 5 МГц соответственно, спектральной эффективностью 4 бит/с/Гц. Пользователи снимают фрагменты видео средней длительностью 15 с и загружают их на облачный сервер со скоростью 2 Мбит/с. При этих исходных данных максимальное число пользователей, занимающих ресурсы индивидуальную использования, равно $C_1 = 20$, а ресурсы совместного использования – $C = 10$. Число мест в буфере положим равным $r = 30$. Над сотой каждые $\alpha^{-1} + \beta^{-1} = 3$ мин, 4 мин, 5 мин пролетают самолеты. Во время пролета самолета над сотой (прохождения радиотени через территорию соты) в течение $\beta^{-1} = 1$ мин ресурсы

совместного использования передаются аэропорту, что может вызвать прерывание обслуживания пользователей оператора.

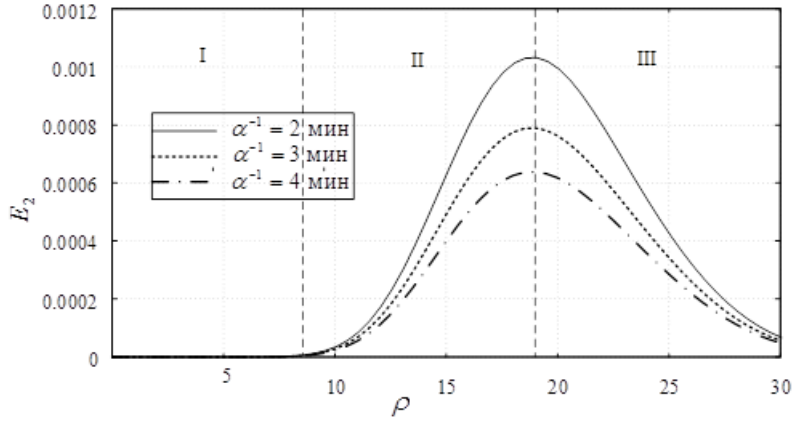


Рис.3. Вероятность E_2 того, что обслуживание пользователя не будет прервано при изъятии полосы совместного использования

На рис. 3 представлена зависимость вероятности E_2 того, что обслуживание конкретного пользователя не будет прервано при изъятии ресурсов совместного использования, от интенсивности предложенной нагрузки $\rho=\lambda/\mu$ при различных значениях среднего времени α^{-1} доступности ресурсов. График данной характеристики имеет немонотонный характер и состоит из двух монотонных участков и одного экстремума – глобального максимума, достигаемого при одних и тех же значениях нагрузки для разных значений α^{-1}. Можно выделить три интервала предложенной нагрузки ρ, на которых данная характеристика ведет себя по-разному.

Для объяснения причин наблюдаемого эффекта рассмотрим другие вероятностные характеристики модели, а именно, среднее число N пользователей, занимающих полосу частот совместного использования, и средний объем N_1 свободных ресурсов (среднее число свободных мест) на полосе индивидуального использования. Данные характеристики рассчитываются по формулам:

$$N=\sum_{n_1=0}^{C_1}\sum_{n=0}^{C} np(n_1,n,0,1)+C\sum_{m=1}^{r-C}p(C_1,C,m,1), \tag{7}$$

$$N_1=\sum_{n_1=0}^{C_1}\sum_{n=0}^{C}(C_1-n_1)p(n_1,n,0,1). \tag{8}$$

Графики зависимостей характеристик N и N_1 от интенсивности предложенной нагрузки ρ представлены на рис. 4.

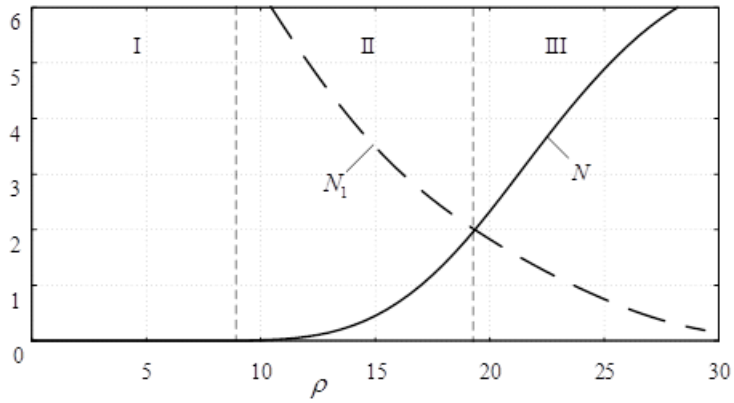


Рис.4. Среднее число пользователей, занимающих полосу совместного использования, и среднее число свободных мест на полосе индивидуального использования

На интервале I вероятность E_2 близка к нулю, что объясняется тем, что при низкой нагрузке на полосе индивидуального использования хватает ресурсов для обслуживания всех пользователей в системе, и следовательно, нет необходимости задействовать полосу совместного использования. Это подтверждает поведение кривых на рис. 4. Действительно, в данном диапазоне нагрузки среднее число пользователей, занимающих полосу совместного использования, близко к нулю, следовательно, вероятность E_2 стремится к нулю. На интервале II вероятность E_2 монотонно растет. Это объясняется тем, что растет среднее число пользователей, занимающих полосу совместного использования, однако это число не превышает средний объем свободных ресурсов на полосе индивидуального использования. Следовательно, в случае изъятия полосы совместного использования вероятность того, что все пользователи с нее перейдут на обслуживание на полосу индивидуального использования, близка к 1. На стыке областей I и II вероятность E_2 достигает своего максимального значения при любых значениях среднего времени α^{-1} доступности ресурсов. При соответствующей нагрузке средний объем свободных ресурсов индивидуального использования такой, что потенциальное число пользователей, которые могут их занять, близко к среднему числу пользователей, занимающих ресурсы совместного использования, т.е. происходит оптимальное распределение нагрузки между обеими полосами частот. Далее, на интервале III среднее число пользователей, занимающих полосу совместного использования, превышает средний объем свободных ресурсов на полосе индивидуального использования, и следовательно, вероятность E_2 монотонно убывает.

На рис. 5 представлены графики зависимостей вероятностей E_1 и I_1 от роста интенсивности предложенной нагрузки ρ при разных значениях среднего времени α^{-1} доступности ресурсов. С ростом нагрузки объем свободных ресурсов на полосе индивидуального использования убывает, следовательно, с меньшей вероятностью на ней хватит ресурсов для принятия на обслуживание всех пользователей с полосы совместного использования, и с большей вероятностью хотя бы один пользователь, занимающий полосу совместного использования, будет прерван. При этом, как видно из графиков, чем меньше значение среднего времени α^{-1} доступности ресурсов, тем больше вероятности E_1 и I_1. Это можно объяснить следующим образом. С ростом интенсивности α изъятия полосы совместного использования, растет вероятность ее изъятия. Поскольку перенаправление пользователей на обслуживание на индивидуальную полосу и прерывание обслуживания пользователей происходят только в случае изъятия полосы совместного использования, то с ростом вероятности ее изъятия увеличиваются вероятности E_1 и I_1.

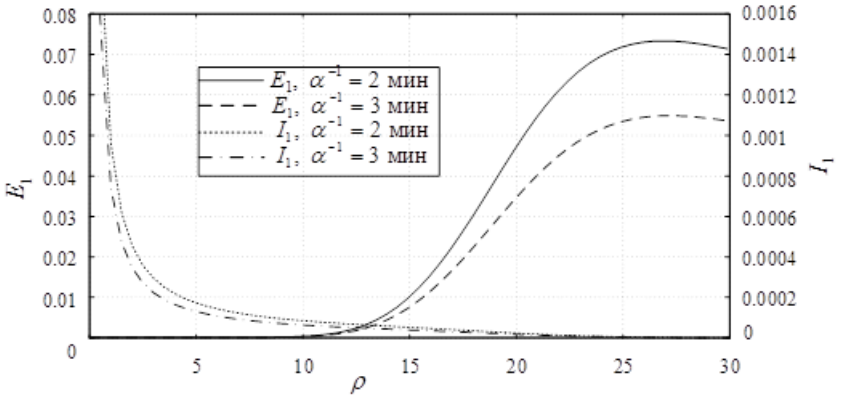


Рис.5. Вероятность того, что обслуживание ни одного пользователя не будет прервано при изъятии полосы совместного использования, и вероятность прерывания обслуживания хотя бы одного пользователя

Заключение

Таким образом, в данной статье авторами построена модель схемы доступа к ресурсам совместного использования в виде СМО с одновременными отказами ненадежных приборов, надежными приборами и конечной очередью, получены формулы для расчета основных вероятностных характеристик прерывания обслуживания и проведен их численный анализ. Целью дальнейших исследований является решение задачи оптимального выбора объема ресурсов совместного использования в условиях ограничений на вероятность прерывания

обслуживания.

В заключение авторы выражают благодарность ассистенту кафедры прикладной информатики и теории вероятностей РУДН Е.В. Мокрову, а также студенту магистратуры РУДН 1-го года обучения А.А. Пономаренко-Тимофееву за ценные советы при разработке сценария совместного использования частот аэропортом и оператором.

Литература

1. Stasiak M., Glabowski M., Wisniewski A., and Zwierzykowski P. Modelling and dimensioning of mobile wireless networks: from GSM to LTE // John Wiley & Sons, 2010. – 340 p.
2. ETSI TS 103 113: Mobile broadband services in the 2300 MHz 2400 MHz band under Licensed Shared Access regime. – ETSI, 2013.
3. Mustonen M., Chen T., Saarnisaari H., Matinmikko M., Yrjola S., and Palola M. Cellular architecture enhancement for supporting the European licensed shared access concept // Wireless Communications. – 2014. – Vol. 21, No. 3. – P. 37–43.
4. Gudkova I., Samouylov K., Ostrikova D., Mokrov E., Ponomarenko-Timofeev A., Andreev S., and Koucheryavy E. Service failure and interruption probability analysis for licensed shared access regulatory framework // Proc. of the 7th Int. Congress on Ultra Modern Telecommunications and Control Systems ICUMT-2015 (October 6–8, 2015, Brno, Czech Republic). – IEEE. – 2015. – P. 123–131.
5. Mokrov E.V., Ponomarenko-Timofeev A.A., Gudkova I.A., Andreev S.D., and Samouylov K.E. Modeling a load balancing scheme between primary licensed and LSA frequency bands in 3GPP LTE networks // Proc. of the IX International Workshop "Applied Problems in Theory of Probabilities and Mathematical Statistics related to modeling of information systems" APTP+MS-2015 (August 10–13, 2015, Tampere, Finland). – Finland, Tampere. – 2015. – P. 54–57.
6. Borodakiy V.Y., Samouylov K.E., Gudkova I.A., Ostrikova D.Y., Ponomarenko A.A., Turlikov A.M., and Andreev S.D. Modeling unreliable LSA operation in 3GPP LTE cellular networks // Proc. of the 6th International Congress on Ultra Modern Telecommunications and Control Systems ICUMT-2014 (October 6–8, 2014, St. Petersburg, Russia). – IEEE. – 2014. – P. 490–496.
7. Perez E., Friederichs K.-J., Viering I., and Diego Naranjo J. Optimization of authorised/licensed shared access resources // Proc. of the 9th International Conference on Cognitive Radio Oriented Wireless Networks and Communications CROWNCOM 2014 (June 2–4, 2014, Oulu, Finland). – 2014. – P. 241–246.
8. Wirth T., Holfeld B., Wieruch D., Halfmann R., and Friederichs K.-J. System level performance of cellular networks utilizing ASA/LSA mechanisms // Proc. of the 1st International Workshop on Cognitive Cellular Systems CSS 2014 (September 2–4, 2014, Germany). – IEEE. – 2014. – P. 1–5.

Вихрова О.Г.[1], Сопин Э.С.[2]

[1] Российкий университет дружбы народов, г. Москва, аспирант кафедры прикладной информатики и теории вероятностей, o.vikhrova@gmail.com

[2] Российский университет дружбы народов, г. Москва, к.ф.-м. н., доцент кафедры прикладной информатики и теории вероятностей, esopin@sci.pfu.edu.ru

АНАЛИЗ ПОКАЗАТЕЛЕЙ КАЧЕСТВА СЕТИ LTE С ПОМОЩЬЮ СИСТЕМ МАССОВОГО ОБСЛУЖИВАНИЯ С ОГРАНИЧЕННЫМ РЕСУРСОМ И СЛУЧАЙНЫМИ ТРЕБОВАНИЯМИ[12]

КЛЮЧЕВЫЕ СЛОВА

LTE, разделение восходящего и нисходящего каналов передачи данных, распределение ресурсов, вероятность блокировки, система массового обслуживания, ограниченные ресурсы, случайная величина выделяемого ресурса

АННОТАЦИЯ

Высокая загрузка каналов передачи данных в сети LTE на сегодняшний день одна из ключевых проблем, для решения которой разрабатываются эффективные алгоритмы распределения радио ресурсов и методы анализа показателей качества сети. В данной работе, предлагается аналитический подход к решению задачи распределения ресурсов в терминах теории массового обслуживания. Рассматривается многолинейная система с несколькими входящими потоками и ограниченным объемом ресурсов, каждой вновь поступившей заявке на обслуживание выделяется случайный объем ресурса. Предположение о случайном объеме выделяемых ресурсов позволяет анализировать поведение системы, приближенной к реальности. Было доказано, что распределение стационарных вероятностей предложенной модели имеет мультипликативный вид. В работе приведен численный анализ полученных результатов.

Введение

С ростом нагрузки в беспроводных сетях следующего поколения LTE, необходимо постоянно увеличивать скорость передачи данных, уменьшать задержки, тем самым улучшая качество восприятия предоставляемых услуг конечными пользователями. Однако ожидания потребителей телекоммуникационных услуг растут быстрее, чем возможности беспроводных сетей. Операторы телекоммуникационных услуг все больше заинтересованы в эффективных инструментах анализа показателей качества сети, которые бы учитывали критические особенности современных условий развития беспроводных сетей.

В сетях LTE, чтобы запустить процесс передачи данных, для каждой новой пользовательской сессии у базовой станции запрашивается некоторое количество ресурса. Выбор режима передачи данных или величины выделяемого ресурса для определенной сессии может происходить автоматически в зависимости от текущего значения отношения сигнала к шуму. Однако в некоторых случаях, например, в случае возникновения эффектов, которые существенно снижают качество передачи данных в беспроводных сетях, применение существующих алгоритмов распределения ресурсов могут привести к неожиданно большому количеству ошибок передачи данных. В следствие чего приходится снова перераспределять ресурсы таким образом, чтобы удовлетворить требования всех сессий. Более того, несмотря на одинаковые требования к скорости передачи данных, каждая пользовательская сессия занимает различный неотрицательный случайный объем доступного ей радио ресурса. Данная особенность современных беспроводных сетей заслуживает особого внимания и должна учитываться, поскольку объем необходимых сессии ресурсов может существенно меняться в зависимости от удаленности пользователя от базовой станции. Случайные величины, которые описывают объем выделяемых ресурсов каждой сессии, могут иметь как дискретное, так и непрерывное

12 Исследование выполнено при частичной финансовой поддержке РФФИ в рамках научных проектов № 14-07-00090, 15-07-03051, 15-07-0360.

распределение. Выбор распределения зависит от многих аспектов, таких как, например, условия распространения сигнала. На рисунке 1 приведен пример распределения ресурсов между пользователями в зависимости от степени их удаленности от базовой станции.

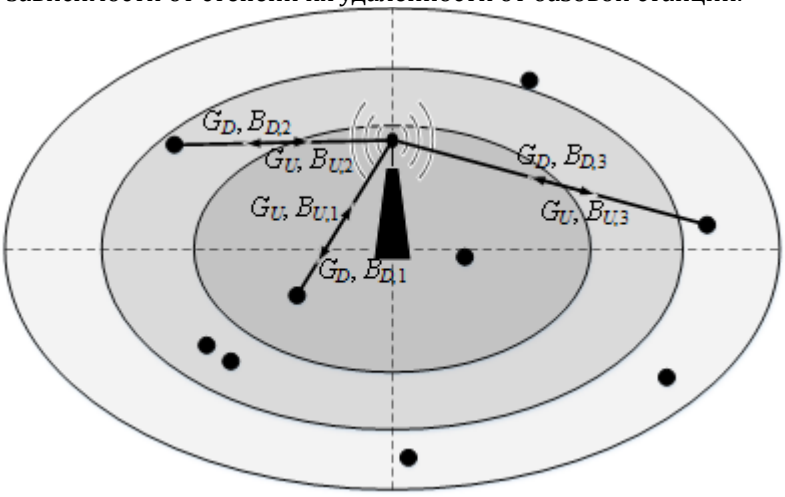


Рис 1. Пример распределения ресурсов со случайными требованиями

На рисунке 1 G_U and G_D - постоянная скорость потока данных в восходящем и нисходящем каналах передачи соответственно, $B_{D,I}$ и $B_{U,i}$ представляют собой количество ресурсных блоков, необходимых *i*-ой сессии, где $B_{D,3} > B_{D,2} > B_{D,1}$ и $B_{U,3} > B_{U,2} > B_{U,1}$.

С бурным развитием сетей 4-го и 5-го поколений, на смену классическим представлениям о соте и обмене данными между пользователями и базовой станцией приходит новый подход, который позволяет выбирать наиболее подходящую базовую станцию соседних сот из числа доступных в гетерогенной среде для скачивания и передачи данных, и таких станций может быть несколько. Поскольку необходимая мощность передающих антенн для нисходящего и восходящего каналов передачи данных существенно отличаются, то пользователь может получать данные от базовой станции макросоты, а передавать их через базовую станцию малой соты, работающей с использованием другого стандарта радиосвязи и других диапазонов частот. Подобный подход получил название разнесение восходящего и нисходящего каналов передачи данных (DUDe, Downlink Uplink Decoupling) [1]. Системы с разнесением каналов исследуются в терминах систем массового обслуживания с несколькими типами ресурсов с применением методов стохастической геометрии.

В пользу обоснованности разнесения восходящего и нисходящего каналов передачи данных говорят результаты исследований в [1]. Разнесение каналов позволяет:

- улучшить отношение сигнал/шум в восходящем канале и снизить мощность передающей антенны;
- снизить уровень воздействия интерференции на восходящий канал;
- увеличить скорость передачи данных в восходящем канале;
- использовать различные алгоритмы балансировки нагрузки в восходящем и нисходящем каналах.

В классических мультисервисных моделях Эрланга или Келли [2] требования к обслуживанию сессий всегда постоянные величины. Эти величины как правило определяются количеством единиц канального ресурса, необходимых для обслуживания сессии. В данной работе предложено проанализировать системы массового обслуживания, в которых требования к обслуживанию новых запросов не являются постоянной величиной, а подчиняются законам распределения случайных величин. Такой подход учитывает особенности современных беспроводных сетей.

Системы массового обслуживания со случайными требованиями были изучены в [3]. Для предложенного случайного процесса, описывающего поведение системы, было необходимо запоминать объем ресурсов, который занимала каждая сессия. В работах [4] и [5] было предложено упрощение модели в случае одного потока требований. Вместо того, чтобы отслеживать объем ресурсов, который занимала каждая сессия, упрощенная модель отслеживает только суммарный объем занятого ресурса. С помощью имитационного моделирования было показано, что стационарные характеристики первоначальной и упрощенной моделей совпадают

для различных входящих потоков и дисциплин обслуживания. В работе [6] были получены стационарные вероятности для пуассоновского потока и экспоненциального времени обслуживания заявок.

В данной работе представлен анализ многолинейной системы массового обслуживания со случайными требованиями и несколькими входящими потоками, предложен вывод распределения стационарных вероятностей и доказан их мультипликативный вид. Также продемонстрировано применение полученных результатов к анализу показателей качества беспроводных систем с учетом особенностей технологии LTE.

Математическая модель

Рассматривается многолинейная система с $N \le \infty$ серверами и $M \le \infty$ типами ресурсов. Заявки *L* типов поступают в систему как *L* независимых пуассоновских потоков с интенсивностями $\lambda_1, \lambda_2, \ldots, \lambda_L$ соответственно. Времена обслуживания заявок не зависят от поступающего потока, независимы между собой и имеют экспоненциальное распределение с интенсивностью μ_l для заявок *l*-го типа. Пусть $\mathbf{R} = (R_1, \ldots, R_M)$ - общий объем ресурсов системы, а объем ресурсов, необходимый заявкам *l*-го типа является случайной величиной (СВ) с функцией распределения $F_l(\mathbf{x})$, независимой от процессов поступления и обслуживания заявок.

Во время обслуживания каждой заявке назначается порядковый номер, который определяется согласно оставшемуся времени обслуживания в порядке убывания. Так заявке, у которой оставшееся время обслуживания наибольшее, будет присвоен номер 1. Эти номера отличаются от очередности поступления заявок в систему. Система функционирует следующим образом:

1. Каждому пользователю для обслуживания требуется некоторый объем ресурсов различных типов;
2. Если для обслуживания вновь поступившей заявки недостаточно ресурсов, то заявка будет отклонена;
3. Как только заявка принята к обслуживанию общий объем занятых ресурсов увеличивается на величину, выделенную поступившей заявке;
4. По завершении обслуживания заявки, общий объем занятых ресурсов уменьшается на величину, необходимую для обслуживания данной заявки.

Состояние системы в некоторый момент времени $t \ge 0$ может быть описано полумарковским процессом $X(t) = (\xi(t), \theta(t), \gamma(t))$, где $\xi(t)$ – число заявок в системе; $\theta(t) = (\theta_1(t), \theta_2(t), \ldots, \theta_{\xi(t)}(t))$ – типы заявок, $\theta_i(t)$ -*i*-ый тип заявки; и $\gamma(t) = (\gamma_1(t), \gamma_2(t), \ldots, \gamma_{\xi(t)}(t))$, $\gamma_i(t)$ – вектор ресурсов, занимаемых *i*-ой заявкой.

Состояние системы может измениться только в моменты времени t_i, когда в систему поступила новая заявки или одна из заявок завершила свое обслуживание. Рассмотрим интервал (t_{i-1}, t_i), на котором система находится в состоянии $X(t) = (k, l_1, l_2, \ldots, l_k, \mathbf{c_1}, \ldots, \mathbf{c_k})$. Длительность этого интервала имеет экспоненциальное распределение с параметром $\lambda + \mu(l_1, \ldots, l_k)$, где $\lambda = \sum_{l=1}^{L} \lambda_l$ и $\mu(l_1, \ldots, l_k) = \sum_{i=1}^{k} \mu_{l_i}$. По окончании данного интервала либо придет новая заявка с вероятностью $\frac{\lambda_l}{\lambda + \mu(l_1, \ldots, l_k)}$ либо систему покинет одна из заявок с вероятностью $\frac{\mu(l_1, \ldots, l_k)}{\lambda + \mu(l_1, \ldots, l_k)}$. Заявка с наименьшим остаточным временем обслуживания покинет систему первой. Учитывая, что номера заявкам присваиваются в соответствии с остаточным временем обслуживания в порядке убывания, если в момент времени t_i систему покинет одна из заявок, то случайный процесс $X(t)$ перейдет из состояния $(k, l_1, l_2, \ldots, l_k, \mathbf{c_1}, \ldots, \mathbf{c_k})$ в состояние $(k-1, l_1, l_2, \ldots, l_{k-1}, \mathbf{c_1}, \ldots, \mathbf{c_{k-1}})$.

Если в момент времени t_i в систему поступит новая заявка, то $X(t)$ может перейти в одно из различных состояний. Пусть $\mathbf{c}$ – это объем ресурсов, необходимый заявке *j*-го типа в момент времени t_i, а $\mathbf{d}$ – общий объем занятый ресурсов, то есть $\mathbf{d} = \mathbf{c_1} + \mathbf{c_2} + \ldots + \mathbf{c_k}$.

Лемма 1. Если независимые случайные величины $\zeta_1, \zeta_2, \ldots, \zeta_k$ распределены экспоненциально с параметрами $\nu_1, \nu_2, \ldots, \nu_k$ соответственно, то

$$P\{\zeta_1 \geq \zeta_2 \geq \ldots \geq \zeta_k\} = \prod_{n=1}^{k} \frac{\nu_n}{\sum_{i=1}^{n} \nu_i}. \quad (1)$$

Доказательство. Лемма 1 доказывается прямым повторным интегрированием выражения (1).

Согласно лемме 1 новая поступившая и принятая к обслуживанию заявка *j*-го типа получит порядковый номер *i* с вероятностью

$$\sigma_{ji}(l_1,\ldots,l_k) = \begin{cases} \prod_{r=1}^{k} \frac{\mu(l_1,\ldots,l_r)}{\mu_j + \mu(l_1,\ldots,l_r)}, & i = 1; \\ \frac{\mu_j}{\mu_j + \mu(l_1,\ldots,l_{i-1})} \prod_{r=i}^{k} \frac{\mu(l_1,\ldots,l_r)}{\mu_j + \mu(l_1,\ldots,l_r)}, & 1 < i \leq k; \\ \frac{\mu_j}{\mu_j + \mu(l_1,\ldots,l_k)}, & i = k+1. \end{cases} \quad (2)$$

Если на момент поступления новой заявки в системе уже находится менее *N* заявок и $\mathbf{c}+\mathbf{d} \leq \mathbf{R}$, то заявка будет принята к обслуживанию, а процесс $X(t)$ перейдет из состояния $(k, l_1, l_2, \ldots, l_k, \mathbf{c_1}, \ldots, \mathbf{c_k})$ в состояние $(k+1, l_1, \ldots, l_{i-1}, j, l_i, \ldots, l_k, \mathbf{c_1}, \ldots, \mathbf{c_{i-1}}, \mathbf{c}, \mathbf{c_i}, \ldots, \mathbf{c_k})$ с вероятностью $\sigma_{ji}(l_i, \ldots, l_k)$.

Рассмотрим распределение стационарных вероятностей процесса $X(t)$:

$$p_0 = \lim_{t\to\infty} P\{\xi(t) = 0\};$$

$$p^k_{l_1,\ldots,l_k}(\mathbf{x}_1,\ldots,\mathbf{x}_k) = \lim_{t\to\infty} P\{\xi(t) = k; \theta_1(t) = l_1, \ldots, \theta_k(t) = l_k; \gamma_1(t) \leq \mathbf{x}_1, \ldots, \gamma_k(t) \leq \mathbf{x}_k\}.$$

Описанный выше переход однозначно определяет переходное ядро процесса $X(t)$ [7] и позволяет записать систему уравнений для вычисления стационарных вероятностей следующий образом:

$$p_0 \sum_{j=1}^{L} \lambda_j F_j(\mathbf{R}) = \sum_{j=1}^{L} \mu_j p^1_j(\mathbf{R}); \quad (3)$$

$$\sum_{j=1}^{L} \lambda_j \int_{\mathbf{0} \leq \mathbf{y}_1 \leq \mathbf{x}_1} F_j(\mathbf{R} - \mathbf{y}_1) p^1_{l_1}(d\mathbf{y}_1) + p^1_{l_1}(\mathbf{x}_1)\mu_{l_1} = \lambda_{l_1} F(\mathbf{x}_1) p_0 +$$
$$+ \sum_{j=1}^{L} (\mu_j + \mu_{l_1}) \int_{\substack{0 \leq \mathbf{y}_1 \leq \mathbf{x}_1 \\ 0 \leq \mathbf{y}_2 \leq \mathbf{R} - \mathbf{y}_1}} p^2_{l_1 j}(d\mathbf{y}_1, d\mathbf{y}_2), \quad 1 \leq l_1 \leq L, \quad \mathbf{0} \leq \mathbf{x}_1 \leq \mathbf{R}; \quad (4)$$

$$\sum_{j=1}^{L}\lambda_j \int\limits_{\substack{0\le \mathbf{y}_i\le \mathbf{x}_i,\\ i=1,2,..,k\\ \mathbf{y}_1+..+\mathbf{y}_k\le \mathbf{R}}} F_j(\mathbf{R}-\mathbf{y}_1-\ldots-\mathbf{y}_k)p^k_{l_1l_2..l_k}(d\mathbf{y}_1,..,d\mathbf{y}_k)+p^k_{l_1l_2..l_k}(\mathbf{x}_1,\mathbf{x}_2,..,\mathbf{x}_k)\sum_{i=1}^{k}\mu_{l_i}=$$

$$=\sum_{i=1}^{k}\lambda_{l_i}\sigma_{l_ji}(l_1..l_{i-1}l_{i+1}..l_k)p^{k-1}_{l_1..l_{i-1}l_{i+1}..l_k}(\mathbf{x}_1,..,\mathbf{x}_{i-1},\mathbf{x}_{i+1},..,\mathbf{x}_k)F_{l_i}(\mathbf{x}_i)+$$

$$+\sum_{j=1}^{L}(\mu_j+\mu(l_1..l_k)) \int\limits_{\substack{0\le \mathbf{y}_i\le \mathbf{x}_i,\ i=1,2,..,k\\ 0\le \mathbf{y}_{k+1}\le \mathbf{R}-\mathbf{y}_1-..-\mathbf{y}_k}} p^{k+1}_{l_1..l_kj}(d\mathbf{y}_1,..,d\mathbf{y}_{k+1}), \tag{5}$$

$$1\le l_1,..,l_k\le L,\quad \mathbf{x}_1,..,\mathbf{x}_k\ge\mathbf{0},\quad \sum_{i=1}^{k}\mathbf{x}_i\le\mathbf{R},\quad 1<k<N;$$

$$p^N_{l_1l_2..l_N}(\mathbf{x}_1,\mathbf{x}_2,..,\mathbf{x}_N)\sum_{i=1}^{N}\mu_{l_i}=$$

$$=\sum_{i=1}^{N}\lambda_{l_i}\sigma_{l_ji}(l_1..l_{i-1}l_{i+1}..l_N)p^{N-1}_{l_1..l_{i-1}l_{i+1}..l_N}(\mathbf{x}_1,..,\mathbf{x}_{i-1},\mathbf{x}_{i+1},..,\mathbf{x}_N)F_{l_i}(\mathbf{x}_i), \tag{6}$$

$$1\le l_1,..,l_N\le L,\quad \mathbf{x}_1,..,\mathbf{x}_N\ge\mathbf{0},\quad \sum_{i=1}^{N}\mathbf{x}_i\le\mathbf{R}$$

Подставив полученные выражения (3)-(6) с условием нормировки (7)

$$p_0+\sum_{k=1}^{N}\sum_{1\le l_1,..,l_k\le L}\int\limits_{\substack{\mathbf{x}_1,..,\mathbf{x}_k\ge\mathbf{0}\\ \mathbf{x}_1+..+\mathbf{x}_k\le\mathbf{R}}} p^k_{l_1,..,l_k}(d\mathbf{x}_1,\ldots,d\mathbf{x}_k)=1, \tag{7}$$

получим уравнение для вычисления стационарных характеристик в мультипликативном виде:

$$p_0=\left(1+\sum_{r=1}^{N}\sum_{k_1+..+k_r=r}F_1^{(k_1)}*\ldots*F_r^{(k_r)}(\mathbf{R})\frac{\rho_1^{k_1}}{k_1!}\ldots\frac{\rho_r^{k_r}}{k_r!}\right)^{-1}, \tag{8}$$

$$p^k_{l_1,..,l_k}(\mathbf{x}_1,\ldots,\mathbf{x}_k)=p_0F_{l_1}(\mathbf{x}_1)\ldots F_{l_k}(\mathbf{x}_k)\prod_{n=1}^{k}\frac{\lambda_{l_n}}{\sum_{i=1}^{n}\mu_{l_i}}, \tag{9}$$

где $\rho_l=\lambda_l/\mu_l$, а "*" – оператор свертки, $F_l^{(k)}(x)$ также является k-кратной сверткой функции распределения $F_l(x)$. Таким образом уравнения (8) и (9) позволяют найти распределение стационарных вероятностей предложенной модели.

Численный анализ

Рассмотрим соту с расположенной в центре базовой стацией LTE, у которой пользователи запрашивают услуги видеоконференции. Для данного примера используются два типа ресурсов – для восходящего и нисходящего каналов передачи данных соответственно. Под ресурсами подразумевается скорость передачи данных. Если по какой-то причине скорость передачи данных ниже, чем необходимо для видео-конференции, пользователь не сможет подключиться к ней. В LTE сети максимальная скорость передачи данных по восходящему каналу - 75 Мбит/с, а по нисходящему составляет 150 Мбит/с. Для численного анализа считаем, что пользователи пользуются Skype для проведения видеоконференции. В таблице 1 представлены требования к различным типам видеовызовов рекомендованные Microsoft для Skype. Предполагается, что каждый из указанных типов видеовызовов может быть инициирован с некоторой вероятностью.

В предложенном примере количество типов ресурсов $M=2$, объем ресурсов первого типа $R_1=150$ (нисходящий канал), а для второго – $R_2=75$ (восходящий канал). Также предполагается, что максимальное число пользователей в соте не может превысить $N=100$. На

основании данных, представленных в таблице 1, была рассчитана вероятность того, что пользователю будет отказано в установлении сессии для проведения видеоконференции из-за нехватки радио ресурсов, рисунок 2. На рисунках 3 и 4 показана зависимость среднего значения и дисперсии скорости передачи данных в соте от нагрузки. Начиная со значения $\rho = 80$ дисперсия уменьшается. Это происходит потому, что система перегружена и не может принимать больше запросы на установление групповых или HD видеовызовов, однако она может продолжать принимать запросы на установление видеовызовов с меньшими требованиями к скорости передачи данных. Любой дальнейший рост нагрузки заставляет систему функционировать на пределе своих возможностей, наблюдается переполнение ресурсов в обоих каналах (на графике дисперсия еще больше уменьшается).

Таблица 1. Требования к видео вызовам

Тип вызова	Рекомендованная скорость скачивания	Рекомендованная скорость загрузки	Вероятность
Видеовызов (высокое качество)	400 кбит/с	400 кбит/с	0,4
Видео вызов (HD)	1,2 Мбит/с	1,2 Мбит/с	0,3
Групповой вызов (3 польз.)	2 Мбит/с	500 кбит/с	0,2
Групповой вызов (5 польз.)	4 Мбит/с	500 кбит/с	0,1

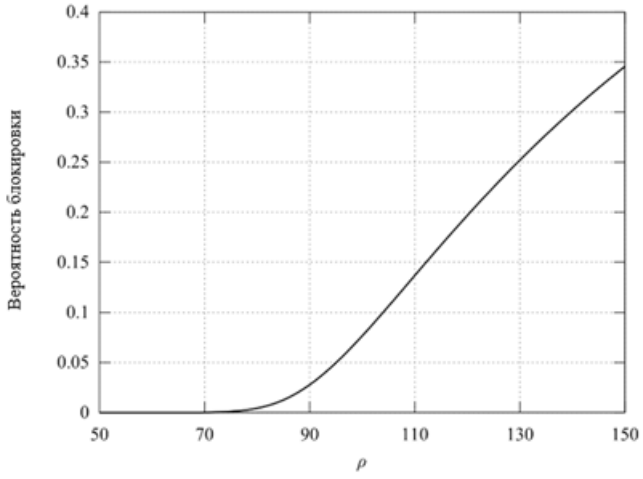


Рис 2. Пример распределения ресурсов со случайным требованием

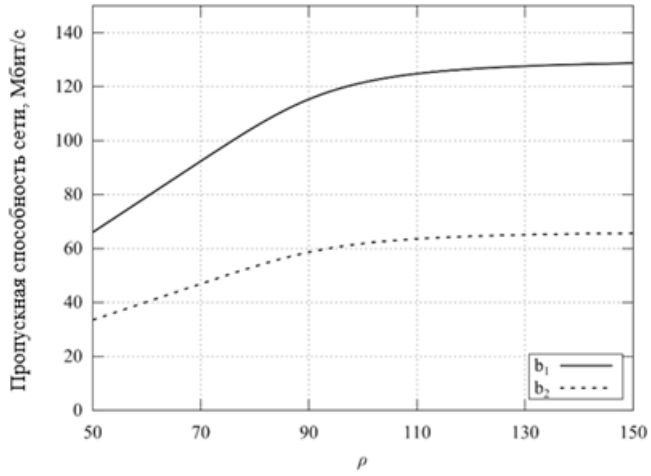


Рис 3. Зависимость средней скорости передачи данных от нагрузки

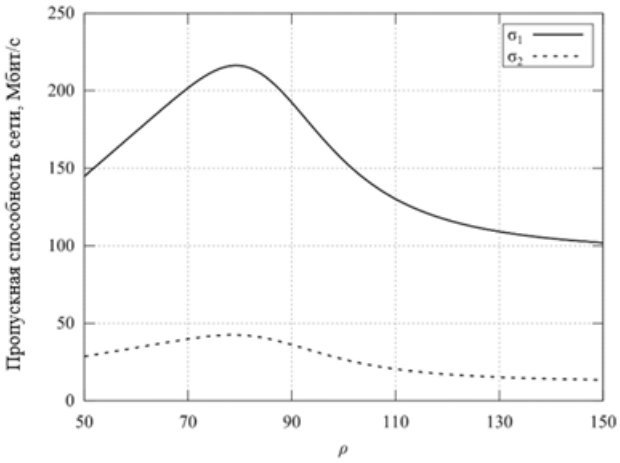


Рис 4. Зависимость дисперсии скорости передачи данных от нагрузки

На основе результатов, представленных на рисунках 2, 3 и 4 можно сделать вывод об оптимальной интенсивности трафика в системе, которая находится в пределах от 70 до 80, что позволяет отклонять запросы с вероятность 0,01. Если предположить, что средняя длительность одного вызова составляет 3 минуты, то базовая станция сможет принимать новые запросы в среднем каждые 3 секунды.

Заключение

В данной работе был рассмотрен алгоритм распределения радио ресурсов между пользоватлеями одной базовой станции сети LTE. Была предложена математическая модель в виде мультисервисной системы массового обслуживания с ограниченным ресурсом нескольких типов и случайными требованиями к выделяемому ресурсу. На примере пуассоновского потока данных и экспоненциального времени облуживания было получено распределение стационарных вероятностей в мультиприкативной форме.

Наши дальнейшие исследования заключаются в поиске эффективных алгоритмов вычисления стационарных вероятностей для дискретного распределения ресурса в представленной модели и ее применение к гетерогенным сетям с разнесением восходящего и нисходящего каналов передачи данных.

Литература

1. H. Elshaer, F. Boccardi, M. Dohler, R. Irmer “Downlink and Uplink Decoupling: A disruptive architectural design for 5G networks”, Global Communications Conference (GLOBECOM), pp. 1798 – 1803, 2014.
2. F.P. Kelly, «Loss Networks. The Annals of Applied Probability», vol.1, no. 3, pp. 319-378, 1991.
3. О. М. Тихоненко, «Определение характеристик систем обслуживания с ограниченной памятью», Автомат. и телемех., 1997. № 6. С. 105–110.
4. Наумов В.А., Самуйлов К.Е. О моделировании систем массового обслуживания с множественными ресурсами // Вестник РУДН. Серия «Математика. Информатика. Физика», 2014. - № 3. С. 60 -64.
5. V. Naumov, K. Samuoylov, E. Sopin, and S. Andreev, “Two approaches to analysis of queuing systems with limited resources,” Ultra Modern Telecommunications and Control Systems and Workshops, pp. 485–488, 2014.
6. Наумов В.А., Самуйлов К.Е., Самуйлов А.К. «О суммарном объёме ресурсов, занимаемых обслуживаемыми заявками» // Автоматика и телемеханика, 2015. – в печати.
7. В.С. Королюк, А.Ф. Турбин, «Процессы марковского восстановления в задачах надежности систем. Киев: Наук, думка, 1982. С. 236.

Медведева Е.Г.[1], Гайдамака Ю.В.[2]

[1]Российский университет дружбы народов, г. Москва, ул. Орджоникидзе, д. 3, аспирант кафедры прикладной информатики и теории вероятностей, egmedvedeva@gmail.com

[2]Российский университет дружбы народов, г. Москва, ул. Орджоникидзе, д. 3, доцент кафедры прикладной информатики и теории вероятностей, к.ф.-м.н., ygaidamaka@mail.ru

К АНАЛИЗУ ПАРАМЕТРОВ КАЧЕСТВА ПЕРЕДАЧИ МУЛЬТИКАНАЛЬНОГО ПОТОКОВОГО ТРАФИКА В ОДНОРАНГОВОЙ СЕТИ[13]

КЛЮЧЕВЫЕ СЛОВА

Peer-to-peer, одноранговая сеть, замкнутая экспоненциальная сеть массового обслуживания, потоковые данные, Isolated-Channel, вероятность всеобщей передачи, View-Upload Decoupling, P2PTV.

АННОТАЦИЯ

В работе построена модель обмена данными между пользователями системы P2P-телевидения в виде замкнутой однородной экспоненциальной сети массового обслуживания. Рассмотрены два подхода к распространению потокового видео — Isolated-Channel (ISO) и View-Upload Decoupling (VUD). Для исследования состояния всеобщей передачи для отдельного телевизионного канала и системы в целом в зависимости от структурных и нагрузочных параметров системы P2P-телевидения и сравнение вероятности всеобщей передачи для указанных подходов в статье сформулирована и решена средствами имитационного моделирования задача максимизации вероятности всеобщей передачи, одного из основных показателей качества восприятия пользователя.

Одним из приложений, отвечающих концепции одноранговой сети [1], является P2P-телевидение (P2PTV), которое обеспечивает мультиканальное телевещание по сетям P2P, перераспределяя видеопотоки в режиме реального времени. Большинство успешных P2PTV–приложений, таких как Tribler [2], PPS.tv [3], PPLive [4], предоставляют своим пользователям несколько сотен телевизионных каналов для просмотра в режиме реального времени. В настоящее время в P2P-видео системах пользователи (пиры) – зрители одного и того же телевизионного канала образуют группу распределения канала, внутри которой происходит обмен фрагментами видео [5,6]. Такая структура телевизионного вещания, наложенная поверх P2P сети, получила название «модель ISO» (Isolated-Channel) [9]. Пример структурного взаимодействия пользователей P2PTV-систем изображен на рис.1. При переключении пользователей №1 и №5 с канала 1 на канал 2, прекращается раздача (download) канала 1 этими пользователями среди своих соседей — зрителей канала 1. В результате этого соседи должны быстро найти новый источник данных, чтобы снова стало возможно поддерживать устойчивую загрузку (upload) видео потока. При этом в группе канала 2 вновь подсоединившейся пользователь должен быстро найти новых соседей с достаточной пропускной способностью и данными для загрузки. В опубликованных ранее работах по моделированию систем P2PTV, например, [8], учитывается переключение пользователя с канала на канал (так называемая «ротация каналов») и отключение пользователя от системы (так называемая «ротация пользователей»). В данной статье полагаем, что число пользователей фиксировано, таким образом, ротация пользователей не учитывается.

Пусть n – число пользователей, подключенных к мультиканальной P2PTV-системе, J – число каналов. Тогда процесс передачи видеоинформации может быть описан с помощью замкнутой однородной экспоненциальной сети. В данном случае сеть представляет собой множество $\mathrm{J} = \{1,\ldots,J\}$ узлов, в качестве которых выступают каналы, доступные для просмотра пользователям. Сеть не имеет источника и стока, в ней циркулирует постоянное число однотипных заявок, в качестве которых выступают пользователи. Очевидно, что распределение

13 Исследование выполнено при частичной финансовой поддержке РФФИ в рамках научных проектов № 14-07-00090, 15-07-03608 .

времени просмотра канала пользователем является произвольным. Например, для модели, в которой пользователи быстро переключают каналы, прежде чем остановится на одном из них на длительный период времени, это время может сначала составлять около 10 секунд, а затем — около одного часа. Тогда время непрерывного просмотра каждым пользователем любого канала *j* является случайной величиной (СВ), не зависящей от состояния сети и от времени просмотра этим пользователем других каналов из множества *J*, и распределена по экспоненциальному закону с параметром $\mu_j, j \in J$.

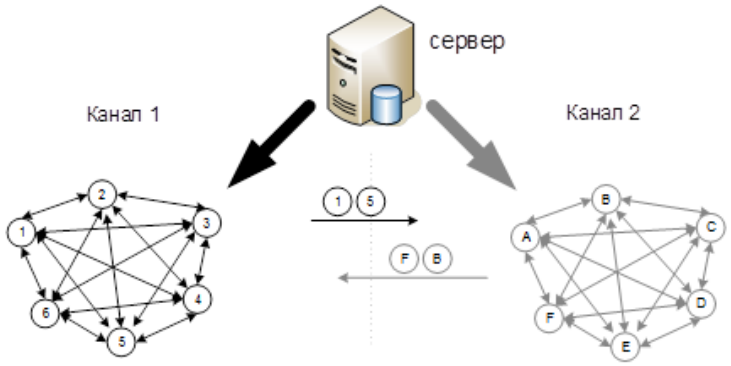


Рис.1 Пример структуры ISO в системе 2-х канального потокового вещания

Таким образом, пользователь попадает на канал *i* на произвольный промежуток времени среднем длительности $1/\mu_i$, а затем переключается на канал *j* с вероятностью p_{ij} .

Рассмотрим процесс переключения каналов некоторым пользователем и предположим, что Y_k – номер канала, на который пользователь попал на *k*-м переключении, *k=0,1,...* . Данный процесс описывает однородная цепь Маркова $\{Y_k\}_{k\geq 0}$, вложенная по моментам переключения пользователей на другой канал с матрицей переходных вероятностей P:

$$P\{Y_{k+1}=j \mid Y_k=i\}=p_{ij}, i, j \in J\ ,\ \mathbf{P}=(p_{ij})_{i,j\in J}, \sum_{j=1}^{J} p_{ij}=1, i=\overline{1,J}\ ,$$

где **P** – стохастическая маршрутная матрица рассматриваемой замкнутой сети.

Пусть ρ_j – относительная популярность канала *j*, $\rho_1+K+\rho_J=1$. Причем непопулярные каналы имеют относительно низкие значения параметра ρ_j . Данная величина может быть задана, например, с помощью закона Ципфа $\rho_j=\dfrac{1}{j^z \sum_{i=1}^{J} \frac{1}{i^z}}$, где *z* – параметр, характеризующий распределение.

Тогда относительная интенсивность λ_j входящего потока на узел *j* сети, т.е. интенсивность, с которой пользователи переключаются на канал *j*, может быть определена из соотношения $\rho_j=\lambda_j/\mu_j, j\in J$.

Положим ξ_j – СВ числа зрителей канала *j*. В модели ISO каждый из *n* подключенных к системе P2PTV пользователей является зрителем некоторого канала. Так как общее число пользователей равно *n*, то $\xi_1+\ldots+\xi_J=n$. Следовательно, $\xi_1,\ldots,\xi_J$ – зависимые СВ.

Лемма [9]. Если для некоторых $m_1,\ldots,m_J$ выполняется условие $m_1+\ldots+m_J=n$, то $P(\xi_1=m_1,\ldots,\xi_J=m_J)=n!\dfrac{\rho_1^{m_1}}{m_1!}\ldots\dfrac{\rho_J^{m_J}}{m_J!}$.

Тогда вероятность появления события $(\xi_j = m_j)$ вычисляется по формуле биноминального распределения:

$$P(\xi_j = m_j) = \frac{n!}{m_j!(n-m_j)!}\rho_j^{m_j}(1-\rho_j)^{(n-m_j)} \qquad (1)$$

Следует отметить, что далее эта модель устойчивого поведения пользователей может быть обобщена, принимая во внимание различные типы их поведения при переключении каналов.

Одним из показателей качества предоставления услуги потокового P2P-вещания является так называемая «всеобщая потоковая передача» [7,8]. Всеобщая потоковая передача происходит в том случае, когда *каждый* пользователь загружает запрашиваемый канал на потоковой скорости этого канала. В современных сетях передачи данных скорости загрузки пользователей достаточно высоки, поэтому можно считать, что скорость загрузки каждого пользователя выше скорости воспроизведения видео потока этим пользователем. Таким образом, в роли ограничения для загрузки воспроизведения видео потока пользователем выступают скорости раздачи других пользователей, от которых он получает порции данных. Поэтому всеобщая передача в системе наблюдается не всегда, а только в случаях, когда каждый пользователь сумел подобрать раздающих пользователей, способных обеспечить для него суммарную скорость загрузки не ниже скорости воспроизведения видео потока. Задачей является исследование вероятности состояния всеобщей передачи для отдельного канала и системы в целом в зависимости от структурных и нагрузочных параметров системы.

Введем следующие обозначения: u_i – скорость раздачи канала пользователем *i, i=1,…,n*; r_j – скорость воспроизведения канала *j*; v_j – скорость раздачи сервером канала *j, j=1,…,J.*

Ресурсный коэффициент канала $\sigma_j(\xi_j)$ (полезный канальный ресурс), зависящий от количества ξ_j зрителей канала, определяем для каждого канала следующим образом:

$$\sigma_j(\xi_j)=\frac{b_j}{d_j(\xi_j)} \tag{2}$$

Здесь b_j – доступная в P2P-системе суммарная пропускная способность для раздачи канала, $d_j(\xi_j)$ – требуемая для просмотра всеми зрителями канала пропускная способность.

Тогда вероятность всеобщей передачи для канала *j* определяется по формуле

$$PU_j=P(\sigma_j(\xi_j)\geq 1), j=1,..,J \tag{3}$$

Рассмотрим два случая: однородная система, в которой все пользователи имеют одинаковые скорости раздачи, и неоднородная система, в которой скорости раздачи пользователей различаются.

Для однородной системы обозначим n_j – число пользователей, смотрящих канал *j*, а N_j – множество таких пользователей. Для модели ISO вероятность всеобщей передачи вычисляется по формуле

$$PU_j=P\left(v_j+\sum_{i\in N_j}u_i\geq n_j r_j\right), \text{где } n_j=\left|N_j\right|. \tag{4}$$

Для неоднородной системы предположим, что существует два типа пользователей: N^l пользователей с низкой скоростью раздачи u_l и N^h пользователей с высокой скоростью раздачи u_h. $N^l+N^h=n$. Тогда ξ_j – случайное число пользователей с низкой скоростью раздачи, смотрящих канал *j*, а η_j – число пользователей с высокой скоростью раздачи, смотрящих канал *j*. Вероятность всеобщей передачи для канала *j* может быть вычислена по формуле

$$PU_j=P\left(v_j+\sum_{i\in N_j}u_i\geq n_j r_j\right)= \tag{5}$$

$$=\sum_{n_j^h=0}^{N^h}\sum_{n_j^l=0}^{N^l}P\left(v_j+u^h n_j^h+u^l n_j^l\geq\left(n_j^h+n_j^l\right)r_j\right)\cdot P\left(\xi_j=n_j^h\right)P\left(\eta_j=n_j^l\right).$$

Анализ характеристик производительности китайского P2PTV приложения PPLive [7], основанного на модели ISO, выявил неустойчивость системы в случае передачи потоков непопулярных каналов (каналов с небольшой аудиторией), а также для систем с возможностью воспроизведения видео потока канала на различных потоковых скоростях. В связи с этим Кейтом Россом был предложен новый подход организации потоковой передачи – так называемая «модель VUD» (View-Upload Decoupling) [9]. Данный подход основан на разделении (Decoupling) загружаемых пользователем потоков данных на два типа: поток для собственного просмотра,

соответствующий выбранному телевизионному каналу (View), и поток (один или несколько) исключительно для раздачи другим пользователям (Upload). Как правило, последние — это потоки телевизионных каналов с низкой популярностью, принудительное распространение которых обеспечивает стабильность мультиканальной системы. Таким образом, в модели VUD для каждого телевизионного канала создается полупостоянная группа, которая сформирована из пользователей, ответственных за его раздачу. Суммарный объем трафика, загружаемый и раздаваемый этой группой, определяет так называемые «непроизводительные потери» (bandwidth overhead) — дополнительные затраты пропускной способности системы, которые являются платой за повышение стабильности системы в модели VUD и отсутствуют в модели ISO.

Для модели VUD формула (2) для ресурсного коэффициента канала $\sigma_j(\xi_j)$, зависящего от количества ξ_j зрителей канала, примет следующий вид:

$$\sigma_j(\xi_j)=\frac{b_j-o_j}{d_j(\xi_j)}, \tag{6}$$

где o_j – непроизводительные потери канала (для модели ISO $o_j=0$).

На рис. 2 показана схема модели VUD для примера сети P2PTV с одним сервером, транслирующим два телевизионных канала, и 12 пользователями от 1 до 6 (зрители канала 1) и от A до F (зрители канала 2). В модели VUD предусмотрено разделение потока телевизионного канала на части – подпотоки, при этом пользователю принудительно назначается загрузка не всего канала, а только одного из его подпотоков. Так, пользователь 2, являющийся зрителем канала 1, загружает не только оба подпотока этого канала (подпоток 1 от пользователя A и подпоток 2 от пользователя D) для собственного просмотра, но и назначенный ему подпоток 1 канала 2. Из рис. 2 видно, что группа раздачи подпотока 1 канала 2 кроме пользователя 2 включает также пользователей 1 и 3. Сервер направляет порции данных этого подпотока только одному пользователю группы раздачи, от которого дальнейшее распространение порций данных этого подпотока внутри группы происходит по принципам P2P. Из рис. 2 также видно, что каждый зритель канала 2 загружает все подпотоки этого канала из обеих групп распределения, например, пользователь E загружает подпоток 1 канала 2 от пользователя 3 и подпоток 2 от пользователя 5.

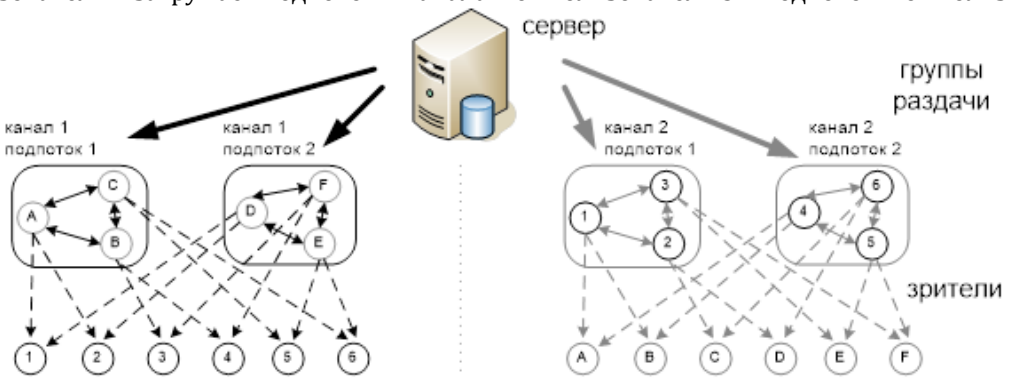


Рис. 2 Пример структуры модели VUD при передачи потокового видео

Подход VUD, основанный на разделение видео потоков для собственного просмотра и для раздачи другим пользователям, требует больше пропускной способности пользователей, поскольку каждый пользователь загружает уже не только видео поток канала для собственного просмотра, но и видео поток другого канала для раздачи. Уменьшение непроизводительных потерь достигается благодаря разделению на подпотоки. Чтобы сформировать S_j подпотоков для канала j , сервер относит каждую *s*-ю порцию данных в соответствующий подпоток, $s=1,\ldots,S_j$. Каждому подпотоку соответствует определяемое системой подмножество пользователей, которые его раздают (см. рис.3).

Предполагается, что совокупная емкость при раздаче подпотока должна соответствовать требованию по скорости воспроизведения данного подпотока. Чем выше среднее требование к скорости канала, тем больше численность соответствующих групп раздачи подпотоков.

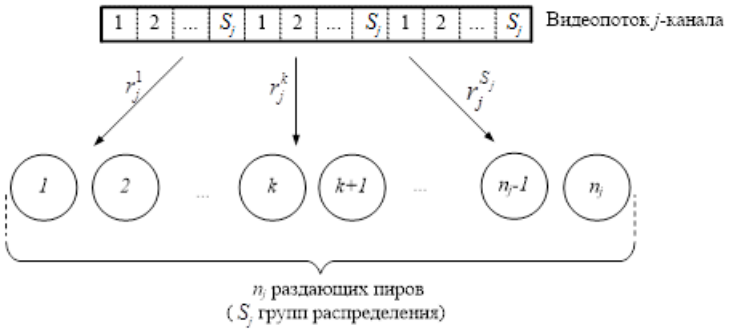


Рис.3. Принцип распределения порций данных в подпотоки

Пусть r_j^s – скорость подпотока *s* канала *j*. Следовательно, $r_j^1+\ldots+r_j^{S_j}=r_j$. Далее, пусть N_j – число пользователей, которым присвоен канал *j*, N_j^s – множество пользователей, составляющих группу раздачи подпотока *s* канала *j*, при этом $n_j^s=|N_j^s|$. Ресурсный коэффициент для подпотока *s* канала *j* определяется по формуле

$$\sigma_j^s(\xi_j)=\frac{b_j^s-o_j^s}{d_j^s(\xi_j)}=\frac{v_j^s+\sum_{i\in\nu_j^s}u_i-n_j^s r_j^s}{\xi_j r_j^s}. \qquad (7)$$

В формуле (7) непроизводительные потери o_j^s для подпотока *s* канала *j* вычисляются следующим образом. Каждый пользователь, раздающий подпоток *s* канала *j*, должен загружать себе этот подпоток на потоковой скорости r_j^s. Тогда общая пропускная способность раздачи подпотока *s* всеми пользователями из множества N_j^s составляет $n_j^s r_j^s$. Заметим, что модель VUD для случая одного телевизионного канала *J=1* исследовалась в [10].

Далее в статье исследуется один из основных показателей качества восприятия пользователя, состояние всеобщей передачи, для отдельного телевизионного канала и системы в целом в зависимости от структурных и нагрузочных параметров системы P2P-телевидения и сравнение вероятности всеобщей передачи для двух подходов – ISO и VUD.

Для однородной системы ISO с конечным числом *n* пользователей все $u_1=u$, где $u>r_j$ для всех *j=1,K,J*. Введем величину K_j, которая имеет смысл доли зрителей канала *j=1,K,J*, от общего числа пользователей системы: $n_j=K_j n$, причем по определению $K_1+K+K_j=1$. Заметим, что в [11] показано, что для однородной системы с бесконечным числом пользователей $\lim_{n\to\infty}\sigma_j(\xi_j)=K_j(u-r_j)/r_j\rho_j$.

Для неоднородной системы с конечным числом *n* пользователей двух типов пользователей с низкой скоростью раздачи u_l и с высокой скоростью раздачи u_h определим долю *f* пользователей с низкой скоростью раздачи. Тогда средняя скорость раздачи пользователями каналов составляет $u^l f+u^h(1-f)$. Аналогично случаю однородной системы введем $N_j^l=K_j^l n$, $N_j^h=K_j^h n$, где $\sum_j K_j^l=f$ и $\sum_j K_j^h=1-f$. Можно показать, что $\sigma_j(\xi_j)\geq 1$, тогда и только тогда, когда $(u^l-r_j)K_j^l+(u^h-r_j)K_j^h\geq r_j\rho_j$.

Задачу оптимизации сформулируем для более общего случая неоднородной системы с пользователями двух типов.

В качестве целевой функции выберем $(u^l-r_j)K_j^l+(u^h-r_j)K_j^h\geq r_j\rho_j$. Тогда задача оптимизации выглядит следующим образом:

$$F(K_j^l,K_j^h)\to\max$$

при условии, что $\sum_{j=1}^{J}K_j^l=f$, $\sum_{j=1}^{J}K_j^h=1-f$, $\rho_1+K+\rho_J=1$.

Одним из решений задачи является $K_j^l=\rho_j f$ и $K_j^h=\rho_j(1-f)$.

Для проведения численного эксперимента был написан программный код в среде Scilab и использовались следующие значения исходных данных [9]: $n=1800$, $J=20$, $r_j=r$, т.е. скорость просмотра всех каналов одинаковая; $f=0{,}5$, $u^l=0.2r$, $u^h=0.2r$, $z=\{0{,}5;1;1{,}5\}$, $v_j=r$.

Результаты расчетов вероятностей всеобщей передачи для модели ISO по формуле (5), а именно графики зависимости вероятности всеобщей передачи PU_j от номера канала j при различных значениях z, изображены на рис. 4.

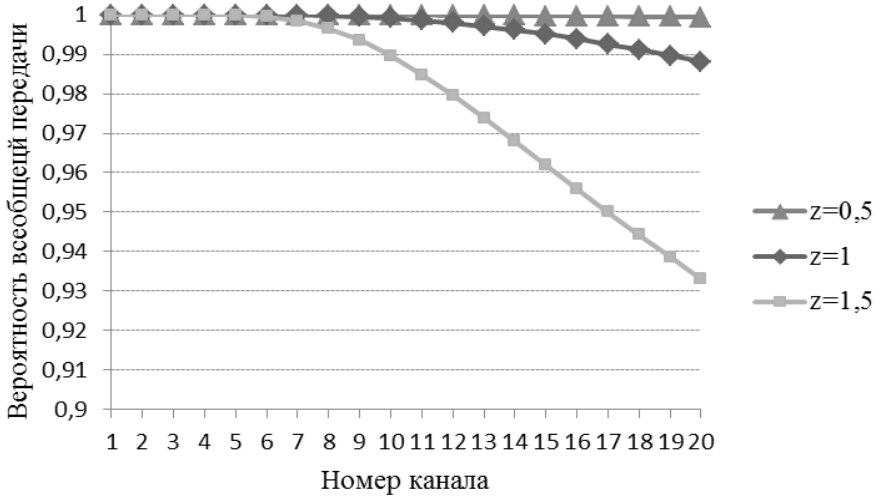


Рис.4. Зависимость PU_j от j при различных значениях z для модели ISO

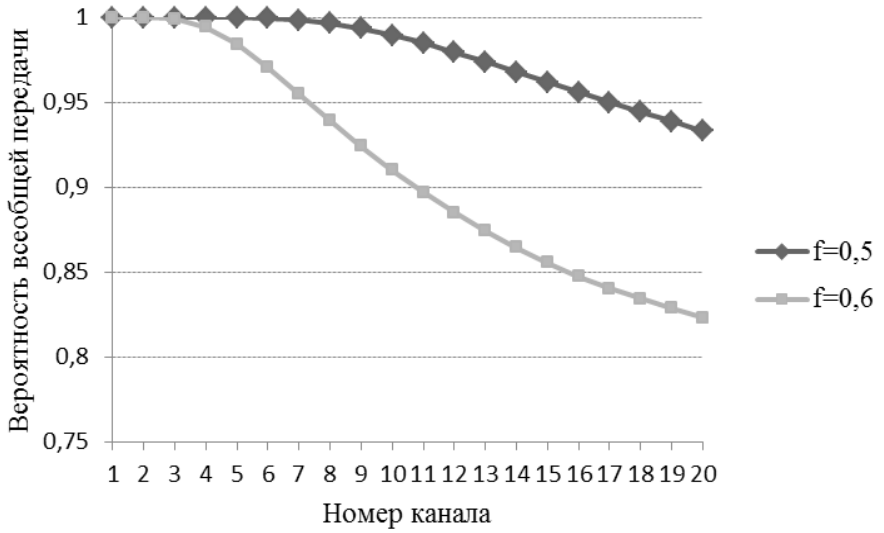


Рис.5. Зависимость PU_j от j при различных значениях f для модели ISO

Влияние значения параметра распределения Ципфа на вероятность всеобщей передачи заметно для каналов с невысокой популярностью, например, для канала № 20. Чем больше значение параметра Ципфа, тем меньше доля пользователей, просматривающих канал № 20. Для трех рассмотренных значений параметра Ципфа $z=\{0{,}5;1;1{,}5\}$ самой малочисленной аудитория канала № 20 будет в случае $z=1{,}5$. Именно поэтому с ростом значения параметра Ципфа вероятность всеобщей передачи для канала с одним и тем же номером падает.

На рис. 5 представлена зависимость вероятности всеобщей передачи в канале j при различных значениях параметра f – доли пользователей с низкой скоростью передачи от общего числа пользователей в модели ISO.

Сравнение графиков на рис. 5 при одинаковых значениях параметра Ципфа ($z=1{,}5$)

показывает увеличение вероятности всеобщей передачи для канала при увеличении доли пользователей с высокой скоростью раздачи на 10%. Это объясняется тем, что с ростом числа пользователей с высокой скоростью раздачи увеличивается суммарный видеопоток (суммарная скорость раздачи), доступный для загрузки зрителям канала.

Для сравнения моделей ISO и VUD выбраны другие значения скоростей раздачи каналов пользователями: $u^l = 0.2\,r$ и $u^h = 6\,r$, коэффициент $f = 0{,}6$, параметр Ципфа $z = 1{,}5$, при этом значения других параметров остаются такими же. Результаты вычислений представлены на рис. 6.

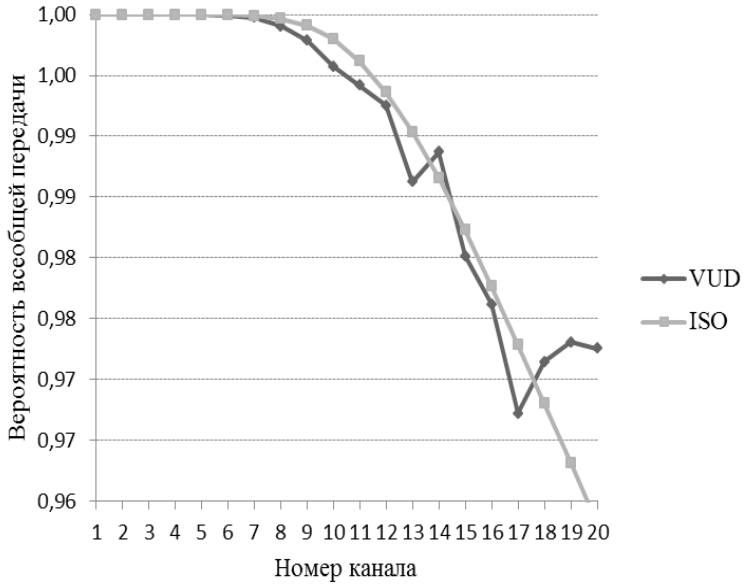


Рис.6. Зависимость PU_j от номера канала j для моделей ISO и VUD

Из графиков рис. 6 видно, что для модели VUD вероятность всеобщей передачи для непопулярных каналов достигает более высоких значений, чем для модели ISO.

В дальнейшем планируется усложнить модель, рассмотрев случаи вещания каналов с различными потоковыми скоростями. Для модели ISO также интересно рассмотреть случай, когда происходит не только ротация каналов, но и ротация пользователей. В этом случае число пользователей не фиксировано, и процесс обмена порциями данных в системе P2P-телевидения может быть описан с помощью открытой однородной экспоненциальной сети.

Исследование выполнено при частичной финансовой поддержке РФФИ в рамках научных проектов № 14-07-00090, 15-07-03608.

Литература

1. Cohen Bram (July 2, 2001). "BitTorrent — a new P2P app". Yahoo eGroups. Archived from the original on 2015-10-01. Retrieved 2007-04-15.
2. Сайт системы P2P TV Tribler [Электронный ресурс] / Режим доступа: https://www.tribler.org/. Дата обращения 01.10.2015.
3. Сайт системы P2P TV Pps.tv [Электронный ресурс] / Режим доступа: www.pps.tv/. Дата обращения 01.10.2015.
4. Сайт системы P2P TV PPLive [Электронный ресурс] / Режим доступа: http://www.pptv.com/. Дата обращения 01.10.2015.
5. Setton E., Girod B. 2007. "Peer-to-Peer Video Streaming". Springer. 150 p.
6. Shen, X., H. Yu, J. Buford, and M. Akon. 2010. "Handbook of Peer-to-Peer Networking". Springer. 1421 p.
7. X. Hei, C. Liang, J. Liang, Y. Liu, K. W. Ross. A measurement study of a large-scale P2P IPTV system, IEEE Trans. Multimedia, vol. 9, no. 8, pp. 1672-1687, Dec. 2007.
8. Yuliya Gaidamaka, Andrey Samuylov, and Konstantin Samouylov. Mathematical Modeling and Performance Analysis of P2P Streaming Networks // Int. Conf. INTHITEN (INternet of THings and ITs ENablers), 3-4 June 2013. - St Petersburg, Russia. - Pp. 69-81.
9. Di Wu, Yong Liu, Keith W. Ross. Modeling and Analysis of MultiChannel P2P Live Video System, IEEE/ACM Transactions on Networking, vol. 18, no. 4, Aug. 2010, pp. 1248–1260.
10. Медведева Е.Г., Гайдамака Ю.В., Агеев К.А. К анализу параметров качества передачи одноканального потокового трафика в одноранговой сети // Информационно-телекоммуникационные технологии и математическое моделирование высокотехнологичных систем: материалы Всероссийской конференции с международным участием ИТТММ-2015. Москва, РУДН, 20–24 апреля 2015 г. — Москва: РУДН, 2015. — С. 99-102.
11. Адаму А., Гайдамака Ю.В. Аппроксимация вероятностных характеристик модели сети P2P // T-Comm - Телекоммуникации и Транспорт. - 2011. - №7. – С.4-7.

Мокров Е.В.[1], Гудкова И.А.[2]

[1] Российский университет дружбы народов, г. Москва, аспирант кафедры прикладной информатики и теории вероятностей, melkor77@yandex.ru

[2] Российский университет дружбы народов, г. Москва, к.ф.-м.н., доцент кафедры прикладной информатики и теории вероятностей, igudkova@sci.pfu.edu.ru

МОДЕЛЬ ДЛЯ РАСЧЕТА УРОВНЯ СНИЖЕНИЯ МОЩНОСТИ В СЕТИ 3GPP LTE С СОВМЕСТНЫМ ИСПОЛЬЗОВАНИЕМ ЧАСТОТ ТЕЛЕМЕТРИИ АЭРОПОРТА[14]

КЛЮЧЕВЫЕ СЛОВА

LTE, совместное использование частот, LSA, интерференция, снижение мощности.

АННОТАЦИЯ

На сегодняшний день существует проблема роста объема трафика, передаваемого в беспроводных сетях, в том числе на базе технологии LTE (Long Term Evolution). Одним из решений этой проблемы является применение регулирующей системы совместного использования лицензированных частот (Licensed Shared Access, LSA). Совместный доступ к ресурсам имеют владелец (абсолютный приоритет) и арендатор-оператор под контролем третьей стороны. Абсолютный приоритет может быть реализован за счет снижения мощности оборудования пользователей на стороне оператора. В работе построена модель для расчета уровня снижения мощности в случае, если владельцем является аэропорт, использующий частоты для телеметрии. Предложены формулы для расчета интерференции сигнала пользователей в восходящем канале с сигналом телеметрии пролетающего над покрытием оператора самолета.

Введение

Согласно данным компании Cisco Systems [1], к 2019 году объем трафика, передаваемого в мобильных сетях, будет составлять около 24.3 экзабайт в месяц. В связи с этим для обслуживания пользователей с требуемым уровнем качества обслуживания у мобильных операторов возникает необходимость задействования дополнительных ресурсов. Одним из возможных решений этой проблемы является система совместного использования лицензированного спектра (LSA) [2].

В связи с тем, что правила совместного использования операторами частот не определены, возможны различные варианты реализации алгоритмов совместного доступа. Они могут отличаться друг от друга в зависимости от числа операторов, приоритета доступа к полосе, механизмов распределения нагрузки между основной полосой оператора и полосой совместного использования [3-4], а также различными политиками реагирования оператора на необходимость доступа к полосе совместного использования владельца.

В работе рассмотрен сценарий совместного использования частот оператором-арендатором, имеющим доступ к совместной с аэропортом-владельцем полосе частот, для анализа интерференции сигналов пользователей оператора и телеметрии аэропорта. Рассматривается политика снижения мощности, т.е. при необходимости, мощность сигналов, передаваемых со стороны оператора в полосе совместного использования радиочастот, снижается, чтобы не создавать интерференции аэропорту. Предложен алгоритм и формулы для расчета уровня снижения мощности передачи пользовательского оборудования для одной соты сети.

Описание модели

В данной работе рассматривается сценарий использования LSA, при котором аэропорт имеет собственную лицензированную полосу частот, которую он использует для телеметрии самолетов при их взлете. Предполагается, что самолеты взлетают достаточно редко, т.е. одновременно через покрытие оператора взлетает только один самолет, и таким образом, полоса частот используется в течение малых интервалов времени на локализованных участках. В зоне

14 Исследование выполнено при финансовой поддержке РФФИ в рамках научных проектов №№ 13-07-00953, 15-07-03608 и 15-07-03051.

действия телеметрии аэропорта расположен мобильный оператор, имеющий возможность регулировать свою интерференцию с сигналами телеметрии аэропорта. Когда частоты телеметрии аэропорта понадобятся для передачи сигнала телеметрии самолету от аэропорта, оператор снижает интерференцию со стороны соты на этих частотах вокруг текущей позиции самолета. Используется политика ограничения мощности, оператор понижает мощность оборудования пользователей, использующих LSA полосу в данном районе. Рассмотрим наихудший случай. В Табл.1 приведены используемые обозначения, которые будут дополнительно пояснены далее в тексте.

Табл. 1. Основные обозначения

Обозначение	Описание
Параметры аэропорта	
f	Несущая частота
p_a^{tx}	Мощность передатчика в аэропорту
v_0	Начальная скорость самолета
a	Ускорение самолета
β	Угол взлета самолета
I_0	Порог интерференции
Свободное пространство (free space)	Модель распространения сигнала в пространстве
γ	Угол поворота взлетно-посадочной полосы относительно оси x
$(x_a, y_a, 0)$	Положение аэропорта
$(x(t), y(t), z(t))$	Положение самолета в момент t
$d_a(t)$	Расстояние от самолета до аэропорта в момент t
$p_a^{rx}(t)$	Мощность сигнала, полученного самолетом от аэропорта в момент t
$r_a(t)$	Радиус радиотени самолета (эффективный радиус интерференции) в момент t
$R_a(t)$	Радиус проекции радиотени самолета на вектор i в момент t
Параметры оператора	
r_c	Радиус соты
p_u^{tx}	Мощность пользовательского оборудования
$(x_c, y_c, 0)$	Положение базовой станции
$p_u^{rx}(t)$	Мощность сигнала, полученного самолетом от пользователя соты в момент t
$d_c(t)$	Расстояние от самолета до базовой станции в момент t
$D_c(t)$	Проекция расстояния от самолета до базовой станции на вектор i в момент t
$d(t)$	Расстояние от самолета до ближайшего пользователя в момент t
$I(t)$	Интерференция по отношению к самолету со стороны ближайшего пользователя соты в момент t
$d_u(t)$	Расстояние от самолета до ближайшего пользователя, расположенного на границе соты (в худшем случае) в момент t
$D_u(t)$	Проекция расстояния от самолета до ближайшего пользователя, расположенного на границе соты на вектор i в момент t
$\overline{p}_u^{tx}(t)$	Сниженная мощность сигнал передачи пользовательского оборудования в момент t

Пусть аэропорт расположен в точке $(x_a, y_a, 0)$ и имеет передатчик, посылающий сигналы телеметрии самолетам во время взлета. Мощность передатчика p_a^{tx}, частота передачи сигнала f.

Самолет взлетает с начальной скоростью v_0 и ускорением a под углом β к горизонту вдоль траектории j (Рис. 1). Поворот взлетно-посадочной полосы i относительно введенной оси координат x составляет γ. В районе аэропорта расположен мобильный оператор, использующий LSA полосу и передающий на той же частоте, что и аэропорт. Передатчик базовой станции имеет направленную антенну, не вызывающую интерференции с сигналами, принимаемыми самолетом от аэропорта. Пользовательское оборудование имеет ненаправленные антенны, передающие с мощностью p_u^{tx}, и может интерферировать с пролетающими самолетами. Рассмотрим наихудший случай, когда интерферирующий с самолетом пользователь расположен в ближайшей к самолету точке соты. Базовая станция рассматриваемой соты имеет координаты $(x_c, y_c, 0)$, радиус соты r_c.

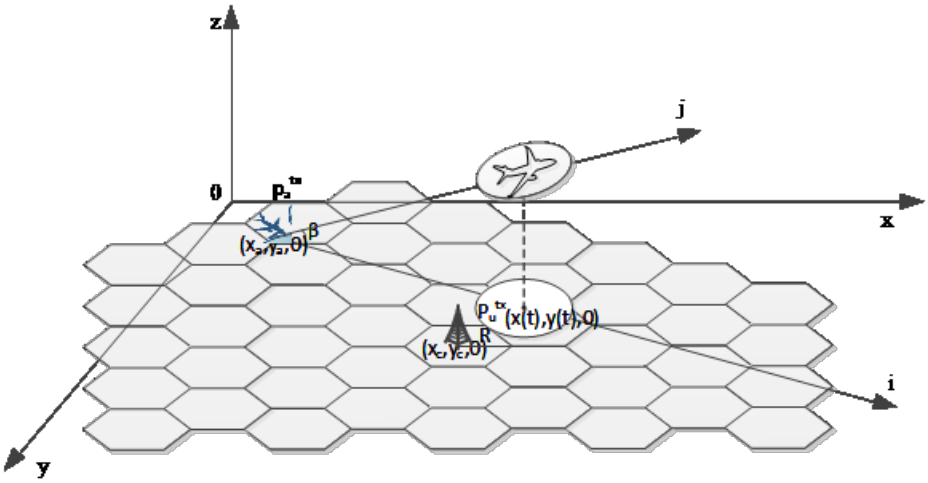


Рис. 1. Сценарий использования LSA

Пользовательское оборудование интерферирует с сигналами телеметрии, при этом порог интерференции составляет I_0. Это значит, что если интерференция $I(t)$ со стороны пользователей соты относительно самолета в момент t достигнет значения I_0, мощность передачи пользовательского оборудования будет снижена таким образом, чтобы интерференция упала до порогового значения. В случае падения интерференции ниже порогового значения, мощность пользовательского оборудования может быть восстановлена. Следовательно, необходимо определить интервал времени, когда интерференция по отношению к самолету со стороны соты может превышать пороговое значение I_0.

Пусть $(x(t), y(t), z(t))$ – координаты расположения самолета в пространстве в момент t. Зная позицию аэропорта и все начальные данные самолета, его положение можно определить как

$$x(t) = \left(v_0 t + \frac{a t^2}{2} \cos\gamma \right) \cos\beta , \quad (1)$$

$$y(t) = \left(v_0 t + \frac{a t^2}{2} \cos\gamma \right) \sin\beta , \quad (2)$$

$$z(t) = \frac{a t^2}{2} \sin\beta . \quad (3)$$

Обозначим расстояние от самолета до базовой станции

$$d_c(t) = \sqrt{\left(x(t) - x_c \right)^2 + \left(y(t) - y_c \right)^2 + z^2(t)} , \quad (4)$$

расстояние от самолета до ближайшего пользователя, расположенного на границе соты,

$$d_u(t) = \sqrt{\left(D_c(t) - r_c \right)^2 + z^2(t)} , \quad (5)$$

а расстояние от самолета до аэропорта

$$d_a(t)=\sqrt{\left(x(t)-x_a\right)^2+\left(y(t)-y_a\right)^2+z^2(t)}\,. \tag{6}$$

Затухание сигнала в свободном пространстве на расстоянии d определяется формулой

$$FSPL(d)=20\,lg\left(\frac{4\pi fd}{c}\right), \tag{7}$$

где c – скорость света.

Алгоритм расчета уровня снижения мощности

С учетом введенных обозначений, алгоритм нахождения уровня снижения мощности пользовательского оборудования может быть записан в следующем виде.

1. Используя формулу (7), получим сигнал, принятый самолетом от аэропорта в момент t:

$$p_a^{rx}(t)=p_a^{tx}-FSPL\left(d_a(t)\right). \tag{8}$$

2. Используя формулы (7) и (8), выразим радиус радиотени самолета следующим образом:

$$r_a(t)=\frac{c}{4\pi f}10^{\frac{p_u^{tx}-p_a^{rx}(t)I_0}{20}}. \tag{9}$$

Получена максимальная область вокруг самолета, находясь в которой, пользователь может вызвать интерференцию, превышающую пороговое значение I_0.

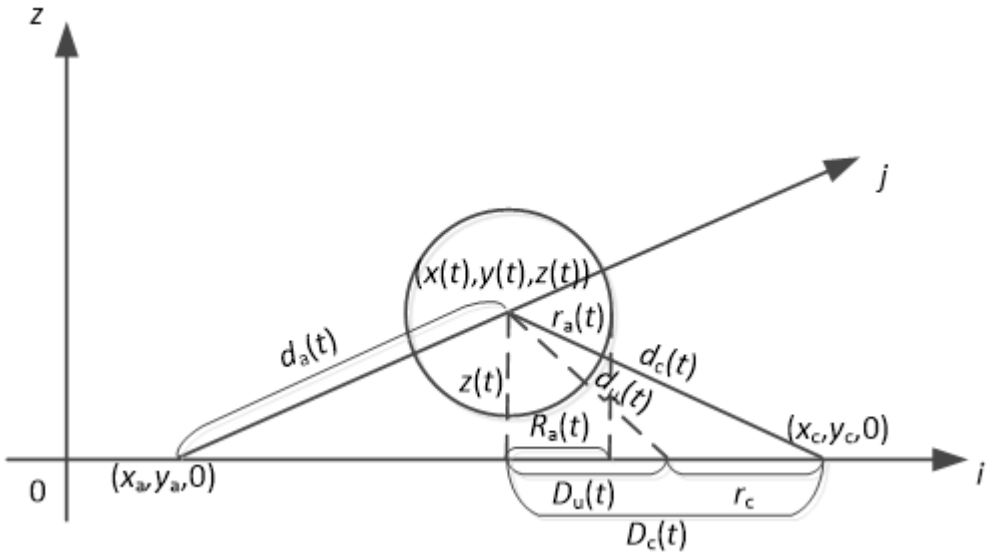


Рис. 2. Расстояние от самолета до пользователя

3. Оценим границы временного интервала $\left[t_{in},t_{out}\right]$, при котором самолет может получать достаточную интерференцию.

Определим проекции (Рис. 2):

$D_c(t)$ – проекция расстояния от самолета до базовой станции на вектор i;

$R_a(t)$ – проекция радиотени самолета на вектор i.

Тогда границы временного интервала $\left[t_{in},t_{out}\right]$ находятся как положительные решения уравнения:

$$D_c(t)=R_a(t)+r_c\,. \tag{10}$$

Стоит заметить, что если уравнение не имеет положительных корней, то это означает, что самолет пролетает достаточно далеко от соты, чтобы сигналы пользовательского оборудования не превышали порогового значения. В случае если уравнение имеет один положительный корень, радиотень самолета касается соты в одной точке. В этом случае интерференция со стороны пользовательского оборудования будет достигать порогового значения, однако не будет превосходить его. Таким образом, в случае когда радиотень самолета не пересекает границ соты, снижения мощности передачи пользовательского оборудования не требуется.

4. Определим расстояние от самолета до ближайшего пользователя в соте. Возможны три случая.

Случай А. Если $D_c(t) \le r_c$, т.е. самолет находится непосредственно над сотой. Тогда, поскольку рассматривается худший случай, ближайший пользователь будет находиться непосредственно под самолетом, таким образом, расстояние от самолета до ближайшего пользователя будет равно $d_u(t) = z(t)$ (Рис. 3А).

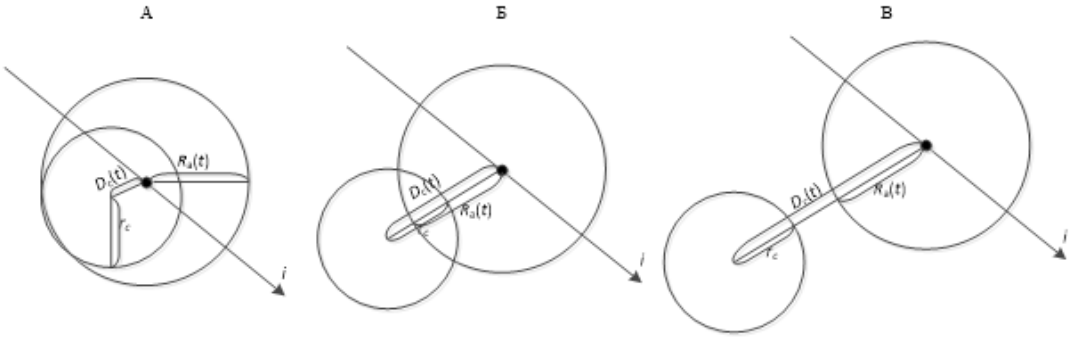


Рис. 3. Случаи расположения самолета относительно соты

Случай Б. Если $r_c < D_c(t) < R_a(t) + r_c$, т.е. самолет располагается достаточно близко к соте, чтобы ощущать интерференцию, однако не находится непосредственно над ней (Рис. 3Б). Тогда расстояние от самолета до пользователя равно расстоянию до границы соты $d(t) = d_u(t)$.

Случай В. Если $R_a(t) + r_c \le D_c(t)$, т.е. самолет пролетает достаточно далеко от соты, радиотень самолета не пересекает границы соты и получаемая самолетом со стороны пользователей интерференция не достигает порогового значения (Рис. 3В).

Можно видеть, что в случаях Б и В расстояние до ближайшего пользователя считается одинаково, однако в случае В интерференция со стороны рассматриваемого пользователя не вызовет необходимости снижения его мощности его оборудования.

Таким образом, расстояние до ближайшего пользователя в сети можно представить следующим образом

$$d(t) = \begin{cases} z(t), & D_c(t) \le r_c, \\ d_u(t), & r_c < D_c(t). \end{cases} \tag{11}$$

5. С учетом найденного временного интервала $[t_{in}, t_{out}]$, вычислим интерференцию для каждого момента времени из него:

$$I(t) = \begin{cases} \dfrac{p_u^{rx}(t)}{p_a^{rx}(t)}, & t \in [t_{in}, t_{out}], \\ 0, & t \notin [t_{in}, t_{out}]. \end{cases} \tag{12}$$

6. Таким образом, подставляя в выражение (12) пороговое значение интерференции I_0 и раскрыв числитель согласно формулам (7) и (8), получаем формулу для расчета уровня снижения мощности пользовательского оборудования:

$$\overline{p}_u^{tx}(t) = \begin{cases} I_0 p_a^{rx}(t) + FSPL(d_u(t)), & t \in [t_{in}, t_{out}], \\ p_u^{tx}, & t \notin [t_{in}, t_{out}]. \end{cases} . \tag{13}$$

Выводы

В работе рассмотрена модель совместного использования полосы частот аэропортом и мобильным оператором. Получено значение интерференции, принимаемой самолетом во время взлета от одной соты, а также необходимый уровень снижения мощности передачи сигнала пользователей в соте. В докладе приведен численный анализ модели совместного использования частот, интерференции принимаемой самолетом со стороны соты и уровень снижения мощности сигнала пользовательского оборудования в соте.

Литература

1. Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update 2014–2019, February 3, 2015.
2. ETSI TR 103 113v1.1.1(2013-07) Electromagnetic compatibility and Radio spectrum Matters (ERM); System Reference document (SRdoc); Mobile broadband services in the 2 300 MHz - 2 400 MHz frequency band under Licensed Shared Access regime.
3. Borodakiy V.Y., Samouylov K.E., Gudkova I.A., Ostrikova D.Y., Ponomarenko A.A., Turlikov A.M., and Andreev S.D. Modeling unreliable LSA operation in 3GPP LTE cellular networks // Proc. of the 6th International Congress on Ultra Modern Telecommunications and Control Systems ICUMT-2014 (October 6–8, 2014, St. Petersburg, Russia). – IEEE. – 2014. – P. 390–396.
4. Gudkova I.A., Samouylov K.E., Ostrikova D.Y., Mokrov E.V., Ponomarenko-Timofeev A.A., Andreev S.D., and Koucheryavy Y.A. Service failure and interruption probability analysis for Licensed Shared Access regulatory framework // Proc. of the 7th International Congress on Ultra Modern Telecommunications and Control Systems ICUMT-2015 (October 6–8, 2015, Brno, Czech Republic). – IEEE. – 2015. – P. 123–131.

Белякова Ю. В.[1], Михалкович С. С.[2]

[1] Южный федеральный университет, г. Ростов-на-Дону, ассистент кафедры информатики и вычислительного эксперимента, julbel@sfedu.ru

[2] Южный федеральный университет, г. Ростов-на-Дону, доцент кафедры алгебры и дискретной математики, miks@sfedu.ru

КОНЦЕПТ-ПАРАМЕТРЫ КАК МЕХАНИЗМ РАЗВИТИЯ СРЕДСТВ ОБОБЩЁННОГО ПРОГРАММИРОВАНИЯ В ЯЗЫКЕ C#

КЛЮЧЕВЫЕ СЛОВА

Обобщённое программирование, объектно-ориентированный язык, концепты, интерфейсы, ограничения, типовые параметры, C#, Java, Scala, F-ограниченный полиморфизм, концепт-паттерн, концепт-параметры.

АННОТАЦИЯ

Современные объектно-ориентированные языки реализуют различные инструменты обобщённого программирования. В статье обсуждаются ключевые критерии, по которым отличаются эти инструменты, сформулированные в результате их сравнительного анализа. Рассматриваются некоторые проблемы средств обобщённого программирования в промышленных языках C#/Java, которые привели к появлению целого ряда расширений, улучшающих поддержку обобщённого программирования в этих языках. На основании выделенных критериев разработан механизм обобщённого программирования на основе концепт-параметров, не уступающий средствам в других языках, но предлагающий принципиально иной стиль обобщённого программирования: мотивация и дизайн механизма для языка C# представлены в статье.

Введение

Обобщённое программирование (ОП) [1] поддерживается на уровне средств языка во многих языках программирования, однако по выразительной силе и удобству использования эти средства существенно отличаются [2]. К наиболее выразительным инструментам ОП относятся классы типов Haskell, в то время как промышленные объектно-ориентированные (ОО) языки программирования C# [3] и Java [4] обладают самыми бедными средствами: некоторые существенные недостатки этих средств рассматриваются в Разд. 2. В течение последних десяти лет было предложено несколько расширений C#/Java, улучшающих возможности обобщённого программирования; возникли новые ОО-языки с лу́чшей поддержкой ОП; был пополнен список возможностей, влияющих на удобство ОП [2,6,12,13]. Исследовав инструменты обобщённого программирования в этих ОО-языках и расширениях, мы выделили несколько критериев, которые характеризуют средства и определяют стиль обобщённого программирования в языке: они обсуждаются в Разд. 3. На основании этих критериев был разработан язык C#^{CP} – расширение C# с механизмом ОП на основе *концепт-параметров*, которые принципиально отличают обобщённое программирование в C#^{CP} от подходов к ОП, принятых в других языках, но не уступают им по выразительной силе. Дизайн концепт-параметров представлен в Разд. 4.

1. Обобщённое программирование

В отличие от обычного кода, который оперирует значениями конкретных типов (int, string, и т. д.), *обобщённые* алгоритмы и структуры данных зависят от *типовых параметров* и оперируют значениями этих типов-параметров. Примером обобщённой структуры данных является обобщённый класс Stack<T> – стек значений типа T, где T – типовый параметр, представляющий произвольный тип. Процесс подстановки конкретных типов вместо типовых параметров называют *инстанцированием* обобщённого кода, а полученный конкретный код – *инстанцией*. Например, инстанция класса Stack<int> представляет стек целых чисел, а Stack<string> – стек строк. Обобщённый код не всегда можно инстанцировать произвольными типами: например, в обобщённом алгоритме сортировки элементов типа T необходимо уметь сравнивать значения типа T друг с другом, поэтому инстанцировать алгоритм сортировки можно только типами, для

которых определена операция сравнения. Подобные требования («тип T поддерживает сравнение») называют *ограничениями* на типовые параметры.

2. Недостатки механизмов обобщённого программирования C#/Java

В C#/Java требования к типовым параметрам обобщённого кода явно выражаются с помощью *ограничений подтипирования* T : C, где T – это типовый параметр обобщённого кода, а C – конкретный класс, и *ограничений-интерфейсов* T : I, где I – интерфейс; в рамках *F-ограниченного полиморфизма* [5] допускаются также *рекурсивные* ограничения вида T : I<T>, где I<> – обобщённый интерфейс. На Рис. 1 представлен обобщённый интерфейс IComparable<T> из стандартной библиотеки .NET и выдержка из обобщённого алгоритма сортировки массива Sort<T>: секция where алгоритма содержит рекурсивное ограничение на типовый параметр T, благодаря которому в теле Sort<T> для значений типа T можно использовать метод сравнения CompareTo.

```
interface IComparable<T> { int CompareTo(T other); }
void Sort<T>(T[] values) where T : IComparable<T>
{ …
  if (values[i].CompareTo(values[j]) < 0) …
}
```

Рис. 1. Примеры обобщённого интерфейса и алгоритма (C#)

Интерфейсы C#/Java не поддерживают обращение к *собственному типу* (то есть типу, реализующему интерфейс), именно поэтому ограничение «в типе T есть метод сравнения со значением типа T» можно выразить только через рекурсивное обобщённое ограничение-интерфейс T : IComparable<T>.

Одним из наиболее существенных недостатков механизма ОП в C#/Java является отсутствие так называемого *ретроактивного моделирования*: если некоторый тип Foo не реализует интерфейс I, установить отношение Foo : I ретроактивно, то есть после определения типа, невозможно, даже если семантически тип удовлетворяет интерфейсу. Поэтому, например, алгоритм сортировки на Рис. 1 нельзя применить к массиву объектов типа Foo, определённому следующим образом:

```
class Foo { ... int CompareDefault (Foo other ); ... }
```

Поскольку тип реализует интерфейс *единственным образом*, отсортировать значения этого типа с помощью алгоритма Sort<> также можно лишь одним способом. Например, целые числа можно отсортировать только по возрастанию, но не по убыванию. Другие серьёзные недостатки средств ОП C#/Java подробно обсуждаются в статьях [8,9], написанных нами, а также в [2,10,11].

```
interface IComparer<T> { int Compare(T a, T b); }
void Sort<T>(T[] values, IComparer<T> cmp)
{ …
  if (cmp.Compare(values[i], values[j]) < 0) …
}
```

Рис. 2. Пример использования концепт-паттерна (C#)

Проблемы ретроактивного моделирования и единственности реализации интерфейса решаются использованием *концепт-паттерна* [6]: вместо явного ограничения на типовый параметр в секции where, в обобщённый код в качестве параметра передаётся некий объект, реализующий требуемую от типа T функциональность; такой параметр называют *концептом*. На Рис. 2 представлена вторая версия алгоритма сортировки из стандартной библиотеки .NET, использующая концепт-паттерн. Теперь за сравнение отвечает не сам тип T, а внешний по отношению к T объект cmp типа IComparer<T> с операцией сравнения двух значений типа T. Чтобы отсортировать массив значений типа Foo, нужно определить новый класс, называемый *моделью*, который реализует интерфейс IComparer<Foo>, и передать объект этого класса в алгоритм сортировки. Используя объект другого класса, реализующего всё тот же интерфейс IComparer<Foo>, можно отсортировать массив другим способом.

Возможность использования различных моделей является одновременно и достоинством, и серьёзной проблемой концепт-паттерна. На Рис. 3 представлен метод объединения двух множеств типа HashSet<T>: класс множества HashSet<T> содержит в качестве поля концепт-

компаратор, который используется для сравнения элементов множества типа T; при объединении множеств для результирующего множества используется компаратор первого аргумента (s1.Comparer). Пусть есть два множества строк, ss1 и ss2 типа HashSet<string>, причём в первом используется регистрозависимое сравнение строк, а во втором – регистронезависимое. Несмотря на то, что оба множества имеют *одинаковый тип*, результат GetUnion(ss1, ss2) будет отличаться от GetUnion(ss2, ss1), так как объекты ss1, ss2 используют разные модели сравнения на равенство. Причём *согласованность этих моделей* нельзя проверить на этапе компиляции кода.

```
static HashSet<T> GetUnion<T>(HashSet<T> s1, HashSet<T> s2)
{
  var us = new HashSet<T>(s1, s1.Comparer);
  us.UnionWith(s2);   return us;
}
```

Рис. 3. Функция объединения множеств типа HashSet<T> (C#)

Заметим, что в примере на Рис. 2 интерфейс IComparer<T> выступает не в роли *ограничения* (как IComparable<T> на Рис. 1) а в роли *типа*, то есть своей изначальной роли в объектно-ориентированном программировании. Один блок кода может использовать интерфейсы и в обеих ролях. Например, алгоритм сортировки произвольной коллекции значений типа T:

void Sort<T>(ICollection<T> values) **where** T : IComparable<T>;

Здесь интерфейс ICollection<T> используется в роли типа, а IComparable<T> – в роли ограничения. Подобная *неоднозначность семантики* интерфейсов усложняет обобщённое программирование и затрудняет восприятие кода. Представляется более правильным использовать интерфейсы исключительно в роли типов, а для *ограничения* типовых параметров обобщённого кода ввести **новую конструкцию языка**. В исследовании [7] на примере почти 14 млн. строк Java-кода было показано, что на практике один и тот же интерфейс никогда не используется в двух ролях, а либо всегда используется в роли типа, либо всегда в роли ограничения. Так, например, интерфейс Comparable<X> – полный аналог интерфейса IComparable<T> в C# – всегда используется в роли ограничения.

3. Поддержка обобщённого программирования в современных объектно-ориентированных языках и их расширениях

В последние годы появился целый ряд объектно-ориентированных языков программирования, в которых на смену «классическим» интерфейсам C# и Java пришли новые OO-конструкции: в Scala [6,14] и Rust [15] это *трейты*, в Swift [16] – *протоколы*. Подобно C#/Java, в этих языках реализован механизм ОП на основе *явных ограничений*, но вместо интерфейсов для ограничения типовых параметров используются указанные конструкции, которые, как и интерфейсы, применяются и в качестве типов. Средства обобщённого программирования в упомянутых языках обладают бóльшей выразительной силой, чем аналогичные инструменты C#/Java. Вместе с тем для языков C# и Java было предложено несколько расширений, улучшающих возможности ОП посредством модификации интерфейсов или внедрения новых конструкций для ограничения типовых параметров: расширенные интерфейсы C# [10]; C#^{cpt} [8] – язык C# концептами, разработанный нами; генерализованные интерфейсы JavaGI [11]; Genus [12] – язык Java с конструкциями-ограничениями. Мы выполнили обзор различных средств обобщённого программирования в современных OO-языках и расширениях C#/Java и провели их сравнительный анализ; результаты этого исследования представлены в работе [13]. Здесь мы приведём некоторые выводы, касающиеся современных подходов к организации средств ОП в OO-языках.

Использование ограничений в роли типов. Возможны два варианта реализации ограничений на типовые параметры:

1. *Конструкции, используемые для ограничения типовых параметров, могут использоваться и как типы*. Этот подход используется во всех рассмотренных нами OO-языках: C# [3], Java [4], Scala [14], Ceylon [17], Rust [15], Swift [16]. Возможность использования конструкции в качестве типа означает, что эта конструкция описывает функциональность *одного объекта* некоторого типа (как интерфейсы C#/Java и трейты Scala) и, возможно, статические методы этого типа (как интерфейсы Ceylon, трейты Rust и протоколы Swift). Из этого следует, что выразить естественным образом *мультипараметрические ограничения*, то есть ограничения на несколько типов, при таком подходе невозможно: для каждого типа из семейства связанных типов нужно определить обобщённую конструкцию,

типовые параметры которой представляют остальные типы из этого семейства, а также привести соответствующие ограничения для каждого типа. На Рис. 4 представлено определение обобщённого паттерна «Наблюдатель» на языке C# (аналогичный пример на Java был приведён ранее в [11]): этот паттерн связывает два типа – субъекта и наблюдателя, причём методы наблюдателя зависят от субъекта, а методы субъекта – от наблюдателя. Ограничения на оба типа приходится указывать трижды: в определениях интерфейсов для каждого из типов семейства, а также в обобщённом коде.

```
interface IObserver<O, S>
  where O : IObserver<O, S>
  where S : ISubject<O, S>                { void Update(S subj); }

interface ISubject<O, S>
  where O : IObserver<O, S>
  where S : ISubject<O, S>                { void Register(O obs); … }

void GenericUpdate<O, S>(S subject, O observer)
  where O : IObserver<O, S>
  where S : ISubject<O, S>                { observer.Update(subject); }
```

Рис. 4. Обобщённый паттерн Наблюдатель (C#)

Следует отметить, что трейты Rust и протоколы Swift поддерживают *собственные типы*, поэтому, в отличие от C#/Java/Scala, в этих языках не нужны рекурсивные ограничения, обсуждавшиеся в Разд. 2. Например, аналогичный интерфейсу IComparable<T> (Рис. 1) трейт Rust определяется следующим образом:

trait Comparable { **fn** CompareTo(**self**, other: **Self**) -> **bool**; }

Тип Self означает тип, реализующий данный трейт, а значение self – объект, для которого вызывается метод CompareTo. Соответственно, обобщённый код вместо рекурсивного ограничения-интерфейса T : IComparable<T> содержит не рекурсивное ограничение-трейт T : Comparable.

2. *Для ограничения типовых параметров обобщённого кода используются отдельные конструкции, которые не могут выступать в роли типов*. Такими конструкциями являются концепты G [18] (не путать с концепт-паттерном), мультипараметрические генерализованные интерфейсы JavaGI [11], ограничения Genus [12] и концепты C#^{cpt} [8]. На Рис. 5 приведён пример C#^{cpt} концепта сравнения CComparable[T] и обобщённого алгоритма сортировки, в котором используется соответствующие *концепт-ограничение* (или концепт-требование). Концепт-ограничение выступает в роли *предиката на типы*. При таком подходе нет принципиальной разницы между ограничениями на один и несколько типов: одна конструкция описывает все требования к семейству связанных типов, она же используется в качестве ограничения в обобщённом коде. Соответствующий пример для паттерна «Наблюдатель» приведён на Рис. 5. Заметим также, что параметром алгоритма сортировки на Рис. 5 является коллекция элементов типа T, представленная обобщённым интерфейсом ICollection<T>, но путаницы между типом и ограничением, как при использовании концепт-паттерна, не возникает.

```
concept CComparable[T] { int Compare(T a, T b); }
void Sort<T>(ICollection<T> values) where CComparable[T] {…}

concept CObserverPattern[S, O] {…}
void GenericUpdate<O, S>(S subj, O obs) where CObserverPattern[O, S] {…}
```

Рис. 5. Примеры обобщённого кода на C#^{cpt}

Ретроактивное моделирование несомненно является одной из важнейших возможностей обобщённого программирования. Подавляющее большинство современных языков и расширений поддерживают ретроактивное моделирование: в Rust и Swift разрешена ретроактивная реализация трейтов и протоколов, а для концептов G/C#^{cpt}, генерализованных интерфейсов JavaGI и ограничений Genus ретроактивно реализуются *модели* (модель задаёт способ, которым тип или набор типов реализуют концепт/генерализованный интерфейс/ограничение).

Множественное определение моделей. В языках C# и Java типы реализуют интерфейс единственным образом. То же касается интерфейсов, трейтов и протоколов в языках Ceylon, Scala, Rust и Swift, а также концептов G и генерализованных интерфейсов JavaGI, где для каждого набора типов в области видимости допускается лишь одна модель. Из-за единственности способа реализации ограничений обобщённый алгоритм сортировки, например, нельзя использовать для сортировки значений одного типа в разном порядке. В языках C#, Java и Scala множественная реализация симулируется использованием концепт-паттерна [6], проблемы которого обсуждались в Разд. 2. Лишь два расширения языков C# и Java – C#^{cpt} [8] и Genus [12] – поддерживают *множественное определение моделей* для одного набора типов. На Рис. 6 представлен заголовок класса HashSet<T>, определённого на C#^{cpt}: вместо поля-компаратора значений типа T используется концепт-ограничение на тип T. В методе объединения множеств это ограничение можно не указывать, так как оно выводится из использования типа HashSet<T> для параметров. При этом гарантируется, что для объектов s1 и s2 используется одна и та же модель компаратора: проверка выполняется на этапе *компиляции* кода. В примере на Рис. 6 объединение множеств ss1 и ss2 не допустимо, так как для множеств ss1 и ss2 используются разные модели концепта сравнения на равенство: для ss1 используется модель по умолчанию (определена с ключевым словом default), реализующая регистрозависимое сравнение строк, а для ss2 – модель StringEqCaseIS регистронезависимого сравнения.

```
class HashSet<T> where CEquatable[T] {…}
static HashSet<T> GetUnion<T>(HashSet<T> s1, HashSet<T> s2)
{
  var us = new HashSet<T>(s1);
  us.UnionWith(s2);   return us;
}

model default StringEqCaseS for CEquatable[string] {…}
model StringEqCaseIS for CEquatable[string] {…}

var ss1 = new HashSet<string>(…);
var ss2 = new HashSet<string>[using StringEqCaseIS](…);
var ss3 = GetUnion(ss1, ss2); // Ошибка компиляции!
```

Рис. 6. Решение проблемы согласованности моделей в C#^{cpt}

В Genus для применения моделей используется иной синтаксис: вызов конструктора множества ss2 выглядит как **new** Set[String **with** StringEqCaseIS](…). Ограничения записываются аналогично, в секции where. Авторы называют типы в Genus *типами, зависящими от моделей*: как и в C#^{cpt}, согласованность моделей проверяется на этапе компиляции.

4. Обобщённое программирование на основе концепт-параметров в языке C#^{CP}

В данном разделе мы представляем мотивацию и дизайн инструментов обобщённого программирования *на основе концепт-параметров* на примере языка C#^{CP} – расширения C# (компилятор C#^{CP} в настоящее время находится на стадии разработки). Подобно G, C#^{cpt}, JavaGI и Genus, для описания *ограничений* в C#^{CP} используется отдельная конструкция языка – *концепт*. Принципиальное отличие концептов C#^{CP} от концептов G [18]/C#^{cpt} [8], генерализованных интерфейсов JavaGI [11] и ограничений Genus [12] заключается в **способе ограничения** типовых параметров обобщённого кода: во всех перечисленных проектах ограничения имеют форму *предикатов на типы*. Заметим, что в G и JavaGI ограничения where C[T^1..T^N] действительно являются предикатами вида «для набора типов T^1..T^N определена модель концепта C[…]», так как в этих языках для данного набора типов можно определить лишь одну модель концепта. В то же время в C#^{cpt} и Genus, в которых поддерживается множественное определение моделей, ограничения в действительности *не* являются предикатами – при инстанцировании обобщённого кода на место каждого концепт-требования подставляется одна модель, выбранная из множества возможных; даже название «тип, зависящий от модели» говорит о том, что наряду с типовыми параметрами модель, фактически, также является *параметром* обобщённого типа. В C#^{CP} вместо концепт-ограничений (или концепт-требований) в форме предикатов обобщённые конструкции *явно зависят* от **концепт-параметров**.

Основываясь на результатах сравнительного анализа средств обобщённого программирования в современных объектно-ориентированных языках [13] и выводах, представленных в предыдущем разделе, мы выделили несколько ключевых вопросов, по которым должны быть приняты решения при проектировании дизайна средств обобщённого программирования:

1. Можно ли использовать в *роли типов* конструкции, применяющиеся для *ограничения типовых параметров* обобщённого кода, или ОО-элементы языка, через которые реализуется полиморфизм, и ограничения на типовые параметры представлены разными конструкциями;
2. Поддерживаются ли *ограничения подтипирования* и *надтипирования* на типовые параметры обобщённого кода;
3. Каким образом обеспечивается возможность *ретроактивного моделирования* ограничений типами;
4. Возможно ли *множественное* определение моделей.

В C#^{CP} концепты используются только для описания *ограничений*, а интерфейсы, в том числе обобщённые, выступают в качестве типов. Можно привести по крайней мере три аргумента в пользу такого решения вопроса 1: конструкции-ограничения, используемые в роли типов, плохо справляются с мультипараметрическими ограничениями (см. Разд. 3); использование одной и той же конструкции в различных ролях усложняет восприятия кода из-за неоднозначной семантики этой конструкции (см. Разд. 2); как показало исследование [7], множество интерфейсов-типов и множество интерфейсов-ограничений на практике действительно не пересекаются, поэтому невозможность использования ограничений в роли типов не вызовет практических проблем. Помимо требований к функциональности, концепты C#^{CP} поддерживают ограничения *подтипирования* T <: B и *надтипирования* T >: D, где T – типовый параметр, а B, D – конкретные классы. Для любого набора типов можно определить именованную модель концепта, причём моделей может быть несколько. Таким образом поддерживается *множественное ретроактивное* определение моделей.

```
concept Equality[T]
{  bool Equal(T x, T y);
   bool NotEqual(T x, T y) { return !Equal(x, y); } }

class HashSet<T ! Equality[T] eq>
{ …
   void UnionWith(HashSet<T!eq> other) {…}
}
static HashSet<T!eq> GetUnion<T ! Equality[T] eq>(HashSet<T!eq> s1,
                                HashSet<T!eq> s2)
{  var us = new HashSet<T!eq>(s1);
   us.UnionWith(s2);   return us;  }
```

Рис. 7. Примеры обобщённого кода в C#^{CP}

Рис. 7 иллюстрирует несколько примеров обобщённого кода с использованием концептов в языке C#^{CP}. Концепт Equality[T] описывает возможность сравнения значений типа T на равенство, причём метод NotEqual имеет *реализацию по умолчанию* (эта возможность поддерживается также в Scala, Ceylon, Rust, Swift, JavaGI, G и C#^{cpt}). Обобщённый класс HashSet<…> множества зависит не только от типа элементов T, но и от концепт-параметра Equality[T] eq; объединение (метод UnionWith) допустимо только со множеством, использующим такую же модель сравнения элементов, то есть со множеством типа HashSet<T!eq>. Заголовок метода GetUnion<…> ясно показывает, что s1 и s2 это множества с одинаковым типом элементов и общей моделью сравнения элементов eq.

На Рис. 8 представлены определения двух моделей концепта Equality[] для строк: eqString для регистрозависимого и eqCInsString для регистронезависимого сравнения. При инстанцировании класса множества нужно указать не только тип элементов, но и модель концепта Equality[] для этого типа. При создании инстанции класса с помощью оператора **new** HashSet<string>(…) концепт-параметр eq инициализируется моделью Equality[string] по умолчанию, если такая есть, в противном случае будет получена ошибка компиляции; модель

можно указать и явно, как при создании ss12 и ss2. Множества ss11 и ss12 используют одну и ту же модель Equality[string], eqString, поэтому к ним можно применить операцию объединения, в то время как ss11 и ss2 – разные, поэтому попытка получить их объединение приведёт к ошибке *компиляции*.

```
model default eqString for Equality[string]
{  bool Equal(string x, string y) { return x == y; }  }

model eqCInsString for Equality[string]
{  bool Equal(string x, string y) { return x.ToLower() == y.ToLower(); }}

var ss11 = new HashSet<string>(…);
var ss12 = new HashSet<string!eqString>(…);
var ss1  = GetUnion(ss11, ss12);  // OK

var ss2 = new HashSet<string!eqCInsString>(…);
var ss3 = GetUnion(ss11, ss2);     // Ошибка компиляции!
```

Рис. 8. Определение моделей и инстанцирование в C#CP

Модели могут быть обобщёнными и зависеть от других моделей. На Рис. 9 представлено определение модели концепта Equality[] для типа обобщённой пары Pair<S, T>: пары одного типа можно сравнивать на равенство, если указаны модели сравнения для их элементов. Заметим, что так называемое условное определение моделей, то есть определение обобщённых моделей с требованиями к типовым параметрам, поддерживается в языках Rust, G, JavaGI и C#cpt, а вот в Genus, где поддерживается множественное определение моделей, реализована полноценная зависимость моделей от моделей.

```
struct Pair<S, T> { public S first; public T second; … }

model eqPair<S, T ! Equality[S] eqS, Equality[T] eqT>
    for Equality[Pair<S, T>]
{
 bool Equal(Pair<S, T> a, Pair<S, T> b){
 { return eqS.Equal(a.first, b.first) && eqT.Equal(a.second, b.second);}
}
```

Рис. 9. Пример обобщённой модели в C#CP

```
Genus:
interface List[E]{ boolean remove(E e) where Eq[E]; … }
List[string] ss = …;
ss.remove[with eqCInsString]("qwerty");

C#CP:
interface IList<T>{ bool Remove<!Equality[T] eq>(T x); … }
IList<string> ss = …;
ss.remove<!eqCInsString>("qwerty");
```

Рис. 10. Ограничения на типовые параметры в методах в Genus и C#CP

Ещё одной интересной возможностью C#CP является возможность накладывать ограничения на типовые параметры в методах класса, которая также реализована в языках Rust и Genus. В статье, посвящённой Genus [12], приводится пример обобщённого интерфейса списка с методом удаления элемента, для которого необходимо сравнение элементов списка на равенство. Соответствующий код приведён на Рис. 10: метод remove накладывает на тип элементов E ограничение в форме предиката, а при вызове метода в квадратных скобках можно указать конкретную модель сравнения элементов. В аналогичном примере на C#CP (Рис. 10) концепт

сравнения является *явным параметром* метода Remove и заменяется на конкретную модель при инстанцировании.

Концепты C#^{CP} образуют пространства имён, в которые включены не только функции, но и имена типовых параметров концепта. В обобщённом коде не обязательно специфицировать все типовые параметры концепт-параметра. Соответствующий пример иллюстрирует Рис. 11: концепт Weighing[T, Weight] описывает возможность взвешивания значений типа T, где вес задаётся типом Weight; FindMaxWeight<…> – алгоритм поиска элемента массива с наибольшим весом, параметризованный типом элементов T и концептом взвешивания Weighing[T, ?] w, где вместо типа веса указан символ «?». Можно было бы сделать тип веса явным типовым параметром функции:

static int FindMaxWeight<T, Weight ! Weighing[T, Weight] w>(T[] values)

Как и на базовом C#, в котором тип веса указывать необходимо:

static int FindMaxWeight<T, Weight> **where** T : IWeighed<Weight> (T[] values)

Но тип веса не имеет значения в заголовке функции FindMaxWeight и лишь делает определение более громоздким. В реализации же функции к типу веса можно обратиться через концепт-параметр: w.Weight. Поскольку при инстанцировании кода вместо w будет подставлена конкретная модель, тип веса можно вывести. Рис. 11 также демонстрирует возможность *уточнения концептов*: концепт взвешивания включает все возможности концепта упорядочения для типа веса. При определении модели нужно указывать модель уточняемого концепта, как это сделано в моделях stringLetCntWeight (вес определяется как количество букв) и stringPnctCntWeight (вес – количество знаков пунктуации). В результате в переменную indMaxLetCnt будет записан индекс строки с наибольшим количеством букв, а в indMinPnctCnt – индекс строки с наименьшим количеством знаков пунктуации. Следует отметить, что возможность *не указывать* типовые параметры концептов в ограничениях является уникальной возможностью языка C#^{CP}. В C#^{cpt} и Genus, где для ограничения типовых параметров используются концепт-ограничения в форме предикатов, такая возможность не предусмотрена.

```
concept Ordering[T]
{  int Compare(T x, T y);
   bool Greater(T a, T b) { return this.Compare(x, y) > 0; } … }
concept Weighing[T, Weight] refining Ordering[Weight]
{  Weight GetWeight(T x);  }

static int FindMaxWeight<T ! Weighing[T, ?] w>(T[] values)
{  … // обработка null и пустого массива
   int indMax = 0;   w.Weight weightMax = w.GetWeight(values[0]);
   for (int i = 1; i < values.Length; ++i)
      if (w.Greater(w.GetWeight(values[i]), weightMax))
      { indMax = i;  weightMax = w.GetWeight(values[i]); }
   return indMax;
}

model default intAscOrd for Ordering[int] {…} // упорядочение int по возрастанию
model intDescOrd for Ordering[int] {…}       // упорядочение int по убыванию
model stringLetCntWeight refining intAscOrd for Weighing[string, int]
{  int GetWeight(string x) { return x.Count(Char.IsLetter); }  }
model stringPnctCntWeight refining intDescOrd for Weighing[string, int]
{  int GetWeight(string x) { return x.Count(Char.IsPunctuation); }  }

string[] ls = { "Oh, no", "Hello, world!", "cat" };
var indMaxLetCnt  = FindMaxWeight<string ! stringLetCntWeight>(ls);  // 1
var indMinPnctCnt = FindMaxWeight<string ! stringPnctCntWeight>(ls); // 2
```

Рис. 11. Ограничения на типовые параметры в методах в Genus и C#^{CP}

Заключение

Новые объектно-ориентированные языки программирования и расширения предлагают более выразительные инструменты обобщённого программирования по сравнению с

классическим F-ограниченным полиморфизмом на основе интерфейсов, реализованным в языках C# и Java. Мы выполнили сравнительный анализ средств ОП в современных языках и расширениях, и выделили несколько ключевых параметров, характеризующих эти инструменты: ретроактивное моделирование, использование конструкций-ограничений в роли типов, множественное определение моделей. Предложенный механизм обобщённого программирования на основе **концепт-параметров** не уступает по выразительной силе средствам ОП в других языках, но предполагает принципиально иной стиль обобщённого программирования: «ограничения как параметры» вместо «ограничения как предикаты на типы». Использование конструкций-ограничений как явных параметров обобщённого кода облегчает написание и восприятие кода, если разрешено множественное определение моделей, что, в свою очередь, является удобным элементом обобщённого программирования. Концепт-параметры также позволяют упростить «интерфейсы» обобщённых компонент, так как типы, которые могут быть выведены из моделей, можно не указывать в качестве параметров: таким образом, заголовки обобщённых компонент содержат только параметры, существенные для пользователей этих компонент.

Литература

1. David R. Musser, Alexander A. Stepanov. Generic Programming // Proceedings of the International Symposium ISSAC '88, p. 13–25. Springer-Verlag, London, UK, UK, 1989.
2. Ronald Garcia, Jaakko Jarvi, Andrew Lumsdaine, Jeremy Siek, Jeremiah Willcock. An Extended Comparative Study of Language Support for Generic Programming // J. Funct. Program., 17(2), March 2007, p. 145–205. Cambridge University Press, New York, NY, USA.
3. Andrew Kennedy, Don Syme. Design and Implementation of Generics for the .NET Common Language Runtime // SIGPLAN Not., 36(5), May 2001, p. 1–12. ACM, New York, NY, USA.
4. Gilad Bracha, Martin Odersky, David Stoutamire, Philip Wadler. Making the Future Safe for the Past: Adding Genericity to the Java Programming Language // Proceedings of the 13th ACM SIGPLAN Conference OOPSLA '98, p. 183–200. ACM, New York, NY, USA, 1998.
5. Peter Canning, William Cook, Walter Hill, Walter Olthoff, John C. Mitchell. F-bounded Polymorphism for Object-oriented Programming // Proceedings of the Fourth International Conference FPCA '89, p. 273–280. ACM, New York, NY, USA, 1989.
6. Bruno Oliveira, Adriaan Moors, Martin Odersky. Type Classes As Objects and Implicits // Proceedings of the ACM International Conference OOPSLA '10, p. 341–360. ACM, New York, NY, USA, 2010.
7. Ben Greenman, Fabian Muehlboeck, Ross Tate. Getting F-bounded Polymorphism into Shape // Proce-edings of the 35th ACM SIGPLAN Conference PLDI '14, p. 89–99. ACM, New York, NY, USA, 2014.
8. Julia Belyakova, Stanislav Mikhalkovich. Pitfalls of C# Generics and Their Solution Using Concepts // Proceedings of the Institute for System Programming, 27(3), June 2015, p. 29–45. Institute for System Programming RAS, Moscow, Russia.
9. Белякова Ю. В., Михалкович С. С. Средства обобщённого программирования в современных объектно-ориентированных языках. Часть 1. Анализ проблем. // Труды научной школы И. Б. Симоненко. Второй выпуск. Ростов н/Д: Изд-во ЮФУ, 2015. С. 60–74.
10. Jaakko Jarvi, Jeremiah Willcock, Andrew Lumsdaine. Associated Types and Constraint Propagation for Mainstream Object-oriented Generics. // Proceedings of the 20th Annual ACM SIGPLAN Conference OOPSLA '05, p. 1–19. ACM, New York, NY, USA, 2005.
11. Stefan Wehr, Peter Thiemann. JavaGI: The Interaction of Type Classes with Interfaces and Inheritance // ACM Trans. Program. Lang. Syst., 33(4), July 2011, p. 12:1–12:83. ACM, New York, NY, USA.
12. Yizhou Zhang, Matthew C. Loring, Guido Salvaneschi, Barbara Liskov, Andrew C. Myers. Lightweight, Flexible Object-oriented Generics // Proceedings of the 36th ACM SIGPLAN Conference PLDI '2015, p. 436–445. ACM, New York, NY, USA, 2015.
13. Белякова Ю. В., Михалкович С. С. Средства обобщённого программирования в современных объектно-ориентированных языках. Часть 2. Обзор новых решений. // Труды научной школы И. Б. Симоненко. Второй выпуск. Ростов н/Д: Изд-во ЮФУ, 2015. С. 75–89.
14. A. Pelenitsyn. Associated types and constraint propagation for generic programming in Scala // Programming and Computer Software, 41(4), p. 224–230. Pleiades Publishing, 2015.
15. The Rust programming language, version 1.2.0. August 2015.
16. http://doc.rust-lang.org/stable/reference.html
17. The Swift Programming Language, version 2.0. August 2015.
18. http://developer.apple.com/swift/resources/
19. The Ceylon language specification, version 1.1.0. October 2014.
20. http://ceylon-lang.org/documentation/1.1/spec
21. Jeremy Siek, Andrew Lumsdaine. A Language for Generic Programming in the Large // Sci. Comput. Program., 76(5), p. 423–465, May 2011. Elsevier North-Holland, Inc., Amsterdam, The Netherlands.

Исаченко А.Н.[1], Исаченко Я.А.[2]

[1]Белорусский государственный университет, г. Минск, к.ф-м..н., доцент кафедры информационных систем управления, isachen@bsu.by

[2] ООО «Икстет», гендиректор, г. Москва, yisachenko@ixtet.com

О НЕКОТОРЫХ ХАРАКТЕРИЗАЦИЯХ МАТРОИДОВ И СВОЙСТВАХ ГАМИЛЬТОНОВЫХ МАТРОИДОВ

КЛЮЧЕВЫЕ СЛОВА

Матроид, информатика, связность, гамильтонов цикл, гамильтонов матроид, циклический граф, периметр, окружение матроида, оракул, полиномиальная сводимость.

АННОТАЦИЯ

Даётся обзор результатов авторов для гамильтоновых матроидов и характеристик матроидов. Приводятся необходимые определения и факты из теории матроидов. Рассматривается отношение связности в матроиде. В частности, эквивалентные определения связности матроида. Даются сведения о связности гамильтонова матроида, необходимых условиях и связности его циклического графа. Приведена аксиоматизация матроида в терминах периметров и окружения.

Введение

Матроиды встречаются во многих разделах теоретической информатики. В теории кодирования при рассмотрении булевых решёток двоичного простого равномерного кода, в криптографии при рассмотрении схем разделения секретов, в теории алгоритмов как основа «жадных» стратегий разработки алгоритмов, в теории сложности для исследования трудоемкости решения тех или иных задач и доказательства их полиномиальной разрешимости.

Теория матроидов тесно связана с теорией графов. Многие понятия теории матроидов появились как обобщения соответствующих графовых понятий и привели к возникновению отдельных направлений в исследовании матроидов. В свою очередь, описание свойств матроидов зачастую даётся в терминах свойств графов, построенных по определённым семействам матроида.

Широкое применение матроидов связано с простотой, удачным сочетанием комбинаторно-математического подхода и алгоритмическими обобщениями результатов дискретной математики.

Можно дать несколько эквивалентных определений матроида в терминах различных понятий [1,2]. Интересным является факт, что в алгоритмическом смысле, понятия матроида не равнозначны. Поэтому при применении матроидного подхода существенным становится форма задания матроида. Сложность задачи, рассматриваемой на матроиде, зависит от выбранного определяющего понятия. И оценивается, в том числе и количеством обращений к соответствующему форме задания оракулу [3-4]. Можно получить различную сложность для одной и той же задачи при разной форме задания матроида. Сложность задачи распознавания свойства матроида, также оценивается числом обращений к оракулу, в терминах которого определяется матроид.

Связность является общесистемной характеристикой, применяемой при синтезе систем любой природы. Отношение связности часто вводится в математическом моделировании для отражения факта наличия взаимодействия между элементами системы, или невозможности разбиения некоторого множества на составные части, или для отражения независимости элементов друг от друга. В сетевых моделях существуют понятия вершинной и рёберной связности. В топологических моделях понятие связного пространства. В проектировании программных систем связность модулей характеризует степень их независимости друг от друга. Отношение связности присутствует и в теории матроидов. В настоящей статье даётся обзор результатов, полученных авторами при исследовании гамильтоновых матроидов, матроидных оракулов и связанных с ними свойствами.

Основные определения

Основное определение матроида, как правило, даётся в терминах независимых множеств. Матроид на конечном множестве *S* определяется как пара $M=(S,F)$, где $F\subseteq 2^S$ - семейство, для которого выполняются следующие условия (аксиомы независимости):

(i1) $\emptyset\in F$;

(i2) если $A\subseteq B u B\in F, mo A\in F$;

(i3) если $A,B\in F u|A|=|B|+1$, то найдется $a\in A\setminus B$, такое что $B\cup a\in F$.

Подмножества из F называются независимыми, из $2^S\setminus F$ - зависимыми.

Ранг $\rho(A)$ подмножества $A\in 2^S$ в матроиде $M=(S,F)$ определяется как мощность максимального независимого множества, содержащегося в A. Максимальное по включению независимое множество называется базой, минимальное по включению зависимое множество - циклом матроида.

Если B база матроида $M=(S,F)$, то её дополнение $S\setminus B$ образует кобазу матроида *M*. При этом семейство всех кобаз F^* образует матроид $M^* = (S, F^*)$ двойственный к исходному.

В терминах циклов матроид определяется как пара $M = (S, \Sigma)$, где *S* конечное множество, $a\Sigma\subseteq 2^S$ - семейство, для которого выполняются следующие условия (аксиомы циклов):

(c1) если C_1, $C_2\in\Sigma, C_1\neq C_2$, то $C_1\not\subset C_2$;

(c2) если C_1, $C_2\in\Sigma, C_1\neq C_2$, и $e\in C_1\setminus C_2$, то для любого $x\in C_1\cap C_2$ существует $C_3\in\Sigma$ такое, что $e\in C_3\subseteq(C_1\cup C_2)\setminus x$.

Важное значение в теории матроидов имеет понятие связности. Элементы $a,b\in S$ в матроиде $M=(S,F)$ называют связными, если существует цикл C такой, что $a,b\in C$.

(1) Матроид $M=(S,F)$ называют связным, если любые два элемента множества *S* являются связными.

Можно дать эквивалентные определения связности, основанные на разложении матроида в произведение меньших матроидов и на понятии сепаратора [3]. Произведение матроидов $M_i=(S_i,F_i)$, определённых на попарно непересекающихся множествах $S_i, i=1,\ldots,t$, это матроид $M=(S,F)$, где S есть объединение множеств S_i, а F состоит из множеств, каждое из которых есть объединение ровно *t* подмножеств, по одному из каждого семейства F_i .

(2) Матроид связен тогда и только тогда, когда он не разлагается в произведение меньших матроидов.

Ограничением матроида $M=(S,F)$ на множество $A\in S$ называется матроид M . $A=(A,F(A))$, где $F(A)=\{X|X\subseteq A, X\in F\}$. Сепаратор матроида $M=(S,F)$ – это подмножество $T\in S$ такое, что

$$M=(S,F)=M.A\times M.(S\setminus T),$$

то есть является произведением ограничений матроида $M=(S,F)$ на множество T и множество $(S\setminus T)$.

(3) Матроид $M=(S,F)$ связен тогда и только тогда, когда он имеет только тривиальные сепараторы $\emptyset$ и S .

Гамильтоновы матроиды, связность и графы

Пусть $M=(S,F)$ - матроид ранга $\rho(S)=k, k<|S|$. Цикл C матроида M называют гамильтоновым, если $|C|=k+1$. Соответственно базу B матроида называют гамильтоновой, если существует содержащий её гамильтонов цикл. Матроид, содержащий гамильтонов цикл, так же называют гамильтоновым. Данные понятия введены в работах [4,5].

В работе [6] установлен следующий факт для гамильтонова матроида.

Теорема 1. Гамильтонов матроид является связным.

По неориентированному конечному графу $G=(V,E)$ можно построить матроид $M(G)$, взяв в качестве S множество рёбер E, и отнеся к F все подмножества рёбер образующие лес. Очевидно, что если граф $G=(V,E)$ имеет гамильтонов цикл, то матроид $M(G)$ является гамильтоновым. Однако, если граф имеет петли и изолированные вершины, он не является гамильтоновым, в то время как его матроид $M(G)$ может иметь гамильтонов цикл.

В гамильтоновом графе при добавлении к гамильтонову циклу ребра образуется ровно два цикла. При добавлении элемента к гамильтонову циклу матроида число образующихся циклов может быть больше двух. Для подтверждения этого рассмотрим следующий простой пример. Пусть $S=\{e_1,\ldots,e_7\}$, а Σ состоит из подмножеств $\{e_1,e_2,e_3,e_4,e_5,e_6\}$, $\{e_1,e_2,e_3,e_4,e_7\}$, $\{e_3,e_4,e_5,e_6,e_7\}$, $\{e_1,e_2,e_5,e_6,e_7\}$. Легко убедиться, что для Σ выполняются аксиомы циклов (c1), (c2). То есть пара $M=(S,\Sigma)$ является матроидом. Причём цикл $\{e_1,e_2,e_3,e_4,e_5,e_6\}$ гамильтонов. Добавление элемента e_7 даёт три цикла.

Хотя «почти все» графы являются гамильтоновыми, задача определения наличия гамильтонова цикла в графе является *NP*-полной [7]. Теорема Бонди-Хватала утверждающая, что граф G с n вершинами является гамильтоновым тогда и только тогда, когда его замыкание – гамильтонов граф, где замыкание определяется добавлением в G ребра (u,v) для каждой пары несмежных вершин u, v, сумма степеней которых не меньше n, имеет в большей степени теоретическое значение. Имеются [8] только достаточные условия Дирака, Оре и Поша, которые полиноминально проверяемы. Перенос этих результатов на матроиды не возможен в силу того, что они формируются в терминах степеней вершин, то есть отношения инцидентности вершин – рёбер, которое отсутствует в матроиде.

Для получения характеризации гамильтоновых матроидов вводится понятие степени его элемента.

Пусть $M=(S,F)$ – матроид. Степень $d(e)$ элемента $e\in S$ есть количество циклов матроида M, содержащих e.

В частности, если $d(e)=0$, то элемент e входит в любую базу матроида M.

В работе [9] установлено необходимое условие для гамильтонова матроида.

Теорема 2. Пусть $M=(S,F)$ – матроид на множестве S с n элементами и рангом $\rho(S)=k, 0<k<n$. Если матроид $M=(S,F)$ гамильтонов, то для любого $e\in S$ имеет место неравенство $d(e)\geq n-k$.

Как уже отмечалось в ведении, описание некоторых свойств матроидов можно дать в виде указания свойств графов, построенных по определённым семействам матроида. Приведём такой результат для гамильтоновых матроидов.

Пусть $M=(S,\Sigma)$ матроид на множестве S, определённый семейством циклов Σ. Циклический граф $G(M)=(V,E)$ матроида M это граф с множеством вершин $V=C$ и множеством рёбер E, состоящим из пар $(C1,C2)$ таких, что:

1) $C1\cup C2$ связное подмножество матроида M;

2) $\rho(C1\cup C2)=|C1\cup C2|-2$.

Одно из свойств гамильтонова матроида дает следующее утверждение [10].

Теорема 3. Циклический граф гамильтонова матроида является связным.

Перейдем к рассмотрению функций , определённых для матроидов.

Периметры и окружения матроида

Через циклы можно определить две функции периметра и получить соответствующую аксиоматизацию матроида [11,12].

Пусть $M=(S,F)$ матроид на множестве S с семейством независимых множеств F. H-периметром матроида (S,F) это функция $\gamma_H: 2^S \rightarrow \{0,\ldots,|S|\}$ со значениями $\gamma_H(A)=max\{|C|: C\subseteq A, C-цикл\}$, если A зависимое множество, и $\gamma_H(A)=0$, если $A\in F$.

Теорема 4. (Аксиомы *H*-периметра). Функция $\gamma_H: 2^S \rightarrow \{0,\ldots,|S|\}$ является функцией *H*-периметра некоторого матроида (S,F) тогда и только тогда, когда для неё выполняются условия:

*hp*1) если $\gamma_H(X)>0$, то существует множество $Y\subseteq X$, для которого $\gamma_H(X)=\gamma_H(Y)=|Y|$;

*hp*2) если $X\supseteq Y$, то $\gamma_H(X)\geq\gamma_H(Y)$;

*hp*3) если $\gamma_H(X)=|X|$, то $\gamma_H(X\setminus x)=0$ для любого $x\in X$;

*hp*4) если $\gamma_H(X)=|X|, \gamma_H(Y)=|Y|, X\neq Y, x\in X\cap Y$, то $\gamma_H((X\cup Y)\setminus x)>0$.

L-периметр матроида (S,F) это функцию $\gamma_L: 2^S \to \{0,\ldots,|S|\}$ со значениями $\gamma_L(A)=min\{|C|: C\subseteq A, C-цикл\}$, если A зависимое множество, и $\gamma_L(A)=0$, если $A\in F$. Аксиомы *L*-периметра даёт следующее утверждение.

Теорема 5. (Аксиомы *L*-периметра). Функция $\gamma_L: 2^S \to \{0,\ldots,|S|\}$ является функцией *L*-периметра некоторого матроида (S,F) тогда и только тогда, когда для неё выполняются условия:

*lp*1) если $\gamma_L(X)>0$, то существует множество $Y\subseteq X$, для которого $\gamma_L(X)=\gamma_L(Y)=|Y|$;

*lp*2) если $X\supseteq Y$, то $\gamma_L(X)\le\gamma_L(Y)$;

*lp*3) если $\gamma_L(X)=|X|$, то $\gamma_L(X\setminus x)=0$ для любого $x\in X$;

*lp*4) если $\gamma_L(X)=|X|, \gamma_L(Y)=|Y|, X\neq Y, x\in X\cap Y$, то $\gamma_L((X\cup Y)\setminus x)>0$.

Для того чтобы дать определение функций окружения напомним определение гиперплоскости матроида. Если $A\subset S$, то замыкание $\sigma(A)$ это объединение A с элементами из $S\setminus A$ добавление которых не изменяет ранг A. Множество A замкнуто, если $A=\sigma(A)$. Замкнутое множество называется плоскостью, а плоскость ранга $\rho(S)-1$ гиперплоскостью.

L-окружением матроида (S,F) это функция $\varphi_L: 2^S \to 1,\ldots,|S|,\infty$, заданная равенствами $\varphi_L(A)=min\{|L|: A\subseteq L, L-гиперплоскость\}$, если ранг $\rho(A)$ меньше ранга $\rho(S)$, и $\varphi_L(A)=\infty$, в противном случае.

Теорема 6. (Аксиомы *L*-окружения). Функция $\varphi_L: 2^S \to \{1,\ldots,|S|,\infty\}$ является *L*-окружением некоторого матроида (S,F) тогда и только тогда, когда для неё выполняются условия:

*ls*1) если $\varphi_L(X)<\infty$, то существует множество $Y\supseteq X$, для которого $\varphi_L(X)=\varphi_L(Y)=|Y|$;

*ls*2) если $X\supseteq Y$, то $\varphi_L(X)\le\varphi_L(Y)$;

*ls*3) если $\varphi_L(X)=|X|<\infty$, то $\varphi_L(X\cup x)=\infty$ для любого $x\in S\setminus X$;

*ls*4) если $\varphi_L(X)=|X|, \varphi_L(Y)=|Y|, X\neq Y, x\in X\cup Y$, то $\varphi_L((X\cap Y)\cup x)<\infty$.

H-окружением матроида является функция $\varphi_H: 2^S \to \{1,\ldots,|S|,\infty\}$, определяемая равенствами $\varphi_H(A)=max\{|H|: A\subseteq H, H-гиперплоскость\}$, если ранг A меньше ранга S, и $\varphi_H(A)=\infty$, в противном случае.

Теорема 7. (Аксиомы *H*-окружения). Функция $\varphi_L: 2S \to \{1,\ldots,|S|,\infty\}$ является *L*-окружением некоторого матроида (S,F) тогда и только тогда, когда для неё выполняются условия:

*hs*1) если $\varphi_L(X)<\infty$, то существует множество $Y\supseteq X$, для которого $\varphi_L(X)=\varphi_L(Y)=|Y|$;

*hs*2) если $X\supseteq Y$, то $\varphi_L(X)\ge\varphi_L(Y)$;

*hs*3) если $\varphi_L(X)=|X|<\infty$, то $\varphi_L(X\cup x)=\infty$ для любого $x\in S\setminus X$;

*hs*4) если $\varphi_L(X)=|X|, \varphi_L(Y)=|Y|, X\neq Y, x\in X\cup Y$, то $\varphi_L((X\cap Y)\cup x)<\infty$.

Функции *H*-периметр, *L*-периметр, *H*-окружение и *L*-окружение двойственного к матроиду $M=(S,F)$ матроида $M^*=(S,F^*)$ называют соответственно *H*-кориметром, *L*-кориметром, *H*-коокружением, *L*-коокружением и обозначают $\gamma_H{}^*, \gamma_L{}^*, \varphi_H{}^*, \varphi_L{}^*$. Получим для любого $X\notin F$:

$\gamma_H(X)=|S|-\varphi_L{}^*(S\setminus X), \gamma_L(X)=|S|-\varphi_H{}^*(S\setminus X)$.

Если $\rho(X)<\rho(S)$ то:

$\varphi_H(X)=|S|-\gamma_L^*(S\setminus X), \varphi_L(X)=|S|-\gamma_H^*(S\setminus X)$.

Матроидные оракулы

Предположим теперь, что рассматривается некоторая задача на матроиде $M=(S,F)$. С точки зрения вычислительной сложности [7] размер задачи следует определить через количество элементов множества S. При этом количество входных данных будет зависеть от способа задания матроида и выражаться либо через число подмножеств, определяющих то или иное семейство матроида, либо через число значения той или иной функций на всех подмножествах из 2^S. Но число подмножеств в семействах, определяющих матроид, может экспоненциально зависеть от мощности S. Что приводит к экспоненциальной временной сложности от мощности S построения входных данных для задачи. Тем самым становиться бессмысленным определение временной сложности алгоритма решения задачи как функции от длины входа, так как генерирование самого входа требует экспоненциального времени. Для устранения данной трудности вводится понятие оракула матроида.

Пусть дан матроид $M=(S,F)$, а u – некоторое понятие матроида. Если понятие определяет характер каждого множества $A\subseteq S$, то через $u(M)$ обозначим соответствующее семейство подмножеств. Если *u* определяет функцию, то через $u(A,M)$ обозначим её значение для подмножества A на матроиде M. Оракулом $O(u)$ назовём инъективное отображение

$$W_u(2^S,\mu(S))\to E(u),$$

где $\mu(S)$ – совокупность всех матроидов на множестве S, $E(u)$ – множество конкретное для каждого понятия. Например, для таких понятий как независимое, конезависимое, базис, кобазис, цикл, коцикл, плоскость, коплоскость, гиперплоскость, когиперплоскость $E(u)=\{ДА, НЕТ\}$ и

$W_u(A,M)=\{ДА, если\ A\in u(M); НЕТ, если\ A\notin u(M)\}$.

Для понятий замыкание, козамыкание $E(u)=2^S$, для понятий ранг, коранг, *H*-периметр, *H*-копериметр, *L*-периметр, *L*-копериметр $E(u)=\{0,\dots,|S|\}$, для понятий *H*-окружение, *H*-коокружение, *L*-окружение, *L*-коокружение

$E(u)=\{1,\dots,|S|,\infty\}$.

Для всех этих понятий $W_u(A,M)=u(A,M)$.

Если имеется задача Ω на матроиде $M=(S,F)$, заданном понятием *u*, то сложность её решения алгоритмом Λ относительно оракула $O(u)$ определяется как число элементарных операций, обозначим его через $m_u(\Omega,\Lambda,M)$, выполняемых алгоритмом. При этом одно обращение к оракулу $O(u)$, то есть получение значения $Wu(A,M)$, так же считается элементарной операцией. Сложность алгоритма Λ для задачи Ω относительно оракула $O(u)$ определяется как $max(m_u(\Omega,\Lambda,M))$ по всем матроидам из $\mu(S)$, при заданной мощности S.

Возникает вопрос: отличается ли сложность алгоритма (по полиномиальности) относительно разных оракулов?

При рассмотрении двух оракулов будем говорить, что оракул $O(u1)$ полиномиально сводим к оракулу $O(u2)$, если значение $W_{u1}(A,M)$ может быть получено за полиномиальное число обращений к оракулу $O(u2)$. Граф полиномиальной сводимости для двадцати двух матроидных оракулов приведен в работах [13,14]. В частности, для приведенных выше понятий матроида справедливо следующее утверждение.

Теорема 8. Оракулы независимое, конезависимое, базис, кобазис, цикл, коцикл, плоскость, коплоскость, гиперплоскость, когиперплоскость, замыкание, козамыкание полиномиально сводимы к оракулам *H*-периметр, *L*-периметр, *H*-окружение, *L*-окружение, *L*-коокружение, *H*-коокружение, *L*-копериметр, *H*-копериметр.

Литература

1. Welsh D.J.A. Matroid theory.– London: Acad. Press, 1976.– 433 с.
2. Oxley J. G. Matroid theory. – N.Y., Oxford University Press, 1992 и 2006. – 532 с.
3. Айгнер М. Комбинаторная теория. – М.: Мир, 1982. – 558 с.
4. Исаченко А.Н. Приложения теории матроидов и гамильтоновы матроиды //Третья международная научная конференция «Математическое моделирование и дифференциальные уравнения»: тезисы докладов, Брест, 17-22 сент. 2012 г. / Брест. гос. ун-т им. А.С. Пушкина. – Брест: БрГУ, 2012. – С. 29-30.
5. Исаченко А.Н., Исаченко Я.А. Периметр матроида и задача коммивояжёра на матроиде // XI Белорусская

математическая конференция: тезисы докладов междунар. науч. конф. Минск , 5-10 нояб. 2012 г. Ч.4. – Минск: Институт математики НАН Беларуси, 2012. – С. 87-88.
6. Исаченко А.Н., Исаченко Я.А. Свойства гамильтоновых матроидов // Международный конгресс по информатике: информационные системы и технологии International Congress on Computer Science: Information Systems and Technologies: материалы междунар. науч. конгресса, Республика Беларусь, Минск, 4-7 нояб. 2013 г. - Минск: БГУ, 2013. –С. 538-541.
7. Рейнгольд Э., Нивергейт Ю., Део Н. Комбинаторные алгоритмы. Теория и практика. – М.: Мир, 1980. – 476 с.
8. Оре О. Теория графов – М.: Наука, 1980. – 336 с.
9. Исаченко А.Н., Исаченко Я.А. Гамильтоновы циклы матроида // Проблемы теоретической кибернетики. Материалы XVII международной конференции (Казань 16-20 июня 2014 г.).- Казань: Отечество, 2014. – С. 116-118.
10. Исаченко А.Н., Исаченко Я.А. Циклический граф гамильтонова матроида// Дискретная математика, алгебра и их приложения: Тез. докл. Междунар. науч. конф. Минск, 14-18 сентября 2015 г.. - Мн.: Институт математики НАН Беларуси, 2015. – С. 108-109.
11. Исаченко А.Н., Исаченко Я.А. *H*-периметр и *L*-окружение матроида // Материалы XI Международного семинара «Дискретная математика и её приложения», посвященного 80-летию со дня рождения академика О.Б. Лупанова (Москва, МГУ, 18-23 июня 2012 г.) – М.: Изд-во механико-математического факультета МГУ, 2012. - С. 246-248.
12. Исаченко А.Н., Исаченко Я.А. О периметрах и окружениях для матроида // Вестник МГАДА. Серия «Экономика», №1 (20). – М.: МГАДА, 2013. – С. 63-67.
13. Исаченко А.Н. Полиномиальная сводимость матроидных оракулов // Известия АН БССР. Сер. физ.-мат. наук, 1984, №6. – С. 33-35.
14. Исаченко А.Н., Ревякин А.М. О сводимости матроидных оракулов // Вестник МГАДА. Серия «Философские, социальные и естественные науки», №3 (9). – М.: МГАДА, 2011. – С. 117-121.

Козлов С.В.

Смоленский государственный университет, г. Смоленск, к.п.н., доцент кафедры информатики,
svkozlov1981@yandex.ru

ИСПОЛЬЗОВАНИЕ СООТВЕТСТВИЯ ГАЛУА КАК ИНВАРИАНТА ОТБОРА КОНТЕНТА ПРИ ПРОЕКТИРОВАНИИ ИНФОРМАЦИОННЫХ СИСТЕМ

КЛЮЧЕВЫЕ СЛОВА

Информатика, информационные системы, программирование, информационно-коммуникационные технологии, соответствие Галуа, инвариантные методы, контент, графовая модель.

АННОТАЦИЯ

В статье обсуждается значимость использования инвариантов при проектировании информационных систем. Предлагаются инвариантные методы отбора контента с помощью соответствия Галуа. Рассматриваются структура и принципы функционирования информационных систем построенных на алгоритмах с использованием соответствий Галуа. Описываются особенности использования соответствия Галуа в совокупности с инвариантами теории графов на иерархических моделях информационных систем.

Задача проектирования информационных систем условно может быть разделена на два этапа. Первый составляет собственно разработка структуры информационной системы. Второй – выбор алгоритмов и определение сценариев функционирования ее основных и вспомогательных элементов [1, 2]. Данные этапы в последующем будут основополагающими при введении информационной системы в использование [3]. Они закладывают ее востребованность на рынке информационных услуг, устойчивость и эффективность использования как единой системы.

Основными структурными элементами информационных систем являются базы данных, большинство из которых имеют табличную организацию. Контент базы данных храниться во взаимосвязанных друг с другом таблицах, записи которых идентифицируются посредством ключевого поля. Однако табличная форма представления данных вовсе не единственная и далеко не всегда в полной мере отражает сущность взаимосвязей в информационной системе. В связи с этим вопрос о поиске инструмента проектирования инновационных информационных сред, который выступал бы инвариантом вне зависимости от области их представления и последующего применения, имеет важное теоретическое значение [4].

На разных этапах развития принципов построения информационных систем предлагались различные способы, базирующиеся на математических инвариантах. У каждого из них имеются как свои достоинства, так и недостатки. Одни успешно применяются на протяжении многих лет, другие не используются вовсе. Определяющую роль здесь играет область применимости метода относительно контента информационной системы [5]. При этом метод, используемый в качестве инварианта информационной системы, должен объяснять результаты практического ее применения и соответствовать данным полученным иным способом.

Использование соответствия Галуа в качестве метода, определяющего взаимодействие элементов системы, возможно в независимости от контентной области информационной среды и ею решаемых задач [6]. Соответствие Галуа задается на паре множеств A и B. Так, при табличной форме организации в качестве таковых можно использовать множества записей двух таблиц взаимосвязанных друг с другом или части этих таблиц как их подмножества. При иерархической форме организации можно рассматривать множества соподчиненных друг другу уровней или же опять их структурные подуровни. На заданных множествах A и B определяется множество пар R как подмножество их прямого произведения. При этом структура множеств естественным образом может разниться в информационных системах и быть как достаточно простой, так и сложной. Множества могут иметь базовые поля данных, например, текстовые или числовые, а могут представлять собой записи, в свою очередь состоящие из полей различного типа, то есть, по сути, представлять данные файлового типа с произвольным доступом. Таким образом, множество пар множества R соответствует одному из типов связей «один ко одному» или «один ко многим»

структурных элементов баз данных информационных систем.

Для отбора контента в информационной системе с использованием соответствия Галуа необходимо задать два множества X и Y. Первое из них является подмножеством множества A, а второе – подмножеством множества B соответственно. Тогда отображение Галуа $\Gamma(X)$ есть множество элементов y множества B, таких что пары (x, y) принадлежат множеству R для любого элемента x их множества X. Аналогичным образом строится обратное отображение Галуа $\Gamma'(Y)$ – это множество элементов x множества A, таких что пары (x, y) принадлежат множеству R для любого элемента y их множества Y.

Теперь, после определения отображений, можно определить и соответствие Галуа. Для этого строят отображения Γ и Γ' на булеанах множеств A и B соответственно. При этом данные отображения должны обладать двумя дополнительными свойствами.

Первое из них состоит из двух соотношений. Для любого подмножества X_1 множества X_2 следует, что множество $\Gamma(X_2)$ есть подмножество множества $\Gamma(X_1)$. И для любого подмножества Y_1 множества Y_2 следует, что множество $\Gamma'(Y_2)$ есть подмножество множества $\Gamma'(Y_1)$.

Второе свойство также включает два соотношения. Множество X есть подмножество множества, полученного как композиция отображения $\Gamma(X)$ и обратного отображения $\Gamma'(\Gamma(X))$. А множество Y есть подмножество множества, полученного как композиция обратного отображения $\Gamma'(Y)$ и отображения $\Gamma(\Gamma'(Y))$.

При этом следует заметить, что если выполняются два указанных свойства, то выполняется и третье – $\Gamma\Gamma'\Gamma(X) = \Gamma(X)$, $\Gamma'\Gamma\Gamma'(Y) = \Gamma\Gamma'(Y)$.

Итак, пара отображений между булеанами множеств A и B или, в общем случае, между двумя любыми множествами называется соответствием Галуа, если выполняются первые два свойства (и, следовательно, вытекающее из них третье свойство) [7].

Свойства соответствия Галуа выступают инвариантом отбора контента при составлении запросов в базах данных информационных систем. Они позволяют с использованием языка запросов SQL отбирать ключевое содержимое во взаимосвязанных блоках информационной системы. Так соответствие Галуа позволяет выявить базисные элементы всей информационной системы, от которых зависит ее стабильное функционирование [8, 9]. Помимо данной важной функции можно:

- выявить элементы-реперы информационной системы, влияющие друг на друга;
- выявить силу корреляционного взаимодействия между структурными элементами информационной системы;
- выявить элементы-записи в информационной системе, которые определяют поведение ее подсистем.

Соответствие Галуа позволяет выполнять данные операции на любом контенте информационной системы [10, 11]. По отношению к нему данный метод выступает инвариантом. Применение метода не ограничивается ни типом базы данных информационной системы ни ее содержательным направлением.

Рассмотрим несколько примеров, которые иллюстрируют применение метода.

Так, например, данная методология востребована при проектировании информационных мониторинговых систем [12, 13]. Вне зависимости от области использования такой информационной системы мониторинга соответствие Галуа дает действенный инструмент получения выводов о состоянии и функционировании среды исследования.

В области образования информационная система мониторинга, основанная на соответствии Галуа [14], позволяет, например, на основании:

- соответствия между уровнем знаний учащихся и их способностями сгруппировать обучаемых в группы и указать формы работы с классом;
- данных обучения формировать «подвижные» группы для продолжения обучения или ликвидации «проблемных» вопросов в обучении отдельных учеников;
- сведений о достигнутом уровне обученности и диагностики обучаемости определить истинную картину достижений ученика.

В области здравохранения подобная мониторинговая информационная система позволяет, например:

- отслеживая динамику роста заболеваемости выявлять факторы влияния на нее отдельных показателей;
- устанавливать степень воздействия компонентов лекарственных средств на состояние пациента;

•определять необходимость обеспечения лекарственными препаратами людей различных социальных групп.

В области строительства такого рода мониторинговые информационные системы дают возможность, например:

•отслеживать потребность социально-незащищенных слоев населения в различных мерах государственной целевой социальной поддержки;

•выявлять зависимость старения жилищного фонда капитальных строений от факторов коммунального хозяйствования;

•строить перспективные планы городского строительства домов, дорог, систем обеспечения электричеством и водоснабжением.

Однако следует заметить, что сам по себе инвариантный метод отбора контента, основанный на соответствии Галуа, будет оказывать заметное, но не решающее влияние без привлечения других инвариантных инструментов исследования информационных систем. Так продемонстрировать действенность рассматриваемого метода можно не только на информационных системах с табличной формой организации, но и на иерархических системах. В последнем случае для описания структуры системы наиболее целесообразно применять разнообразные графовые модели [15]. Это могут и *n*-арные деревья, и сети, и графы произвольной ориентации.

Рассмотрим различные инварианты теории графов применительно к инвариантному методу отбора контента, основанному на соответствии Галуа, иерархических моделей информационных систем.

Так инварианты «полустепень исхода вершины графа» и «полустепень захода вершины графа» позволяют задать два параметра соответственно. Первый – это параметр базовости вершины, второй – параметр выводимости вершины. Данных два инварианта позволяют в свою очередь определить такие инварианты как «вектор полустепеней исхода вершин графа» и «вектор полустепеней захода вершин графа». Эти инваринаты совместно с инвариантом соответствия Галуа отбора контента позволяют установить базисные элементы иерархической информационной системы. Они математически описывают набор распределения вершин информационной системы по параметрам базовости и выводимости. В этом случае соответствие Галуа в зависимости от изменения значений свойств выявляет зависимости в распределении существенных и несущественных значений записей вершин графовой модели. В частности в случаях представления некорректной информации с помощью соответствия Галуа на контенте информационной системы можно определить «пораженные» участки. Это дает возможность вовремя скорректировать процесс обработки системных данных.

Инварианты «полустепень исхода графа» и «полустепень захода графа» позволяют определить вершины иерархической информационной системы, имеющие наибольшие параметры базовости и выводимости. В общем случае эти максимальные значения могут быть приписаны разным вершинам. Наполнение системы контентом в свою очередь может менять данные параметры. При этом соответствие Галуа выступает инструментом определения дальнейшего поведения системы при добавлении в нее новых элементов. Возможность предвидеть развитие системы в конечном итоге задает ее устойчивость к воздействию внутренних и внешних факторов.

Инвариант «число слабых компонент графа» устанавливает классы эквивалентности подграфов графовой модели иерархической информационной системы. Он позволяет разбить информационную систему на подсистемы со слабыми связями. Соответствие Галуа в таком случае позволяет выявить элементы системы, обладающие недостающими значениями, а также недостающие элементы необходимые для внесения в ее структуру. Процесс поиска компонент системы со слабыми связями становится более эффективным по времени. Это предопределяет стабильное функционирование информационной системы.

Инвариант «число независимости» задает коэффициенты независимости подмножеств информационной системы и ее число «неплотности». Данные числовые значения определяют внутреннюю устойчивость подмножеств иерархической системы. Они характеризуют связи между отдельными ее элементами. Чем числа независимости подмножеств больше, тем слабее связь между их элементами. Анализ связей между элементами с помощью соответствия Галуа позволяет выявить такие элементы контента, которые максимально понижают «число независимости» добавлением минимального количества дополнительных связей. Применение соответствия Галуа также дает возможность определить те элементы системы, добавление связей между которыми не

влияет или не оказывает существенного влияния на поведение информационной системы в целом.

Дополняющие друг друга инварианты «число *t*-взаимозависимости» и «число *t*-взаимонезависимости» характеризуют разветвленность иерархической информационной системы. Многоступенчатость связей определяет коэффициент сложности системы. Большое число *t*-взаимозависимости указывает на сложность внутренней конструкции элемента системы. Как правило, в таких случаях элементами выступают записи с достаточно большим количеством свойств, каждое из которых в свою очередь представляет запись. Соответствие Галуа с одной стороны дает возможность выявить элементы информационной системы, влияющие на показатели взаимозависимости ее структурных подмножеств. С другой стороны оно выступает инструментом отбора контента, формирующего иерархическую информационную систему с заданными *t*-показателями.

Инвариант «вектор разделения» характеризуется двумя числовыми параметрами. Ими выступают числа внешнего и внутреннего разделения исследуемой вершины модели иерархической информационной системы. Первое число характеризует длину пути до наиболее удаленной вершины, второе – наибольшую из длин кратчайших путей в нее. Такие пары чисел, определяющие вектор для всех вершин графовой модели иерархической системы, описывают ее структуру, а, следовательно, и задают сложность ее контента. Соответствие Галуа на данном инварианте позволяет определить особенности структуры такого строения, выявить общие закономерности и частные случаи в ее проектировании и дальнейшем наполнении контентом.

С предыдущим инвариантом неразрывно связаны еще два инварианта «центр» и «радиус». Вершина графовой модели иерархической информационной системы с минимальным первым параметром является внешним центром, а с минимальным вторым – внутренним центром. Минимальные числовые значения, характеризующие данные величины выступают внешним и внутренним радиусами соответственно. Применительно к графовой модели иерархической информационной системы они определяют компоненты, которые являются базовыми для наибольшего числа других элементов системы. Соответствие Галуа собственными функциональными соотношениями также позволяет определить данные параметры, но с точки зрения контента рассматриваемых элементов. Это дает возможность сопоставить структурные и содержательные связи внутри информационной системы.

Инвариант «число полукомпонент диаметра *p*» характеризуется двумя числовыми величинами – собственно диаметром и полуплотностью. Это позволяет рассматривать подсистемы информационной системы, для графовой модели которых расстояние между любыми ее вершинами не превышает значение *p*. Максимальное число в таких полукомпонентах определяет *p*-полуплотность. Данные параметры позволяют анализировать структурные связи информационной системы длиной не больше p. Соответствие Галуа, определив параметр *p* в качестве входного значения, предоставляет функциональные возможности проанализировать содержательные связи в подграфе между рассматриваемыми элементами, их сложность и влияние друг на друга.

Инвариант «вектор надежности» характеризует значение коэффициента корреляции между эталонными числовыми параметрами и действительными значениями некоторой подсистемы контента информационной системы. Одному из числовых свойств каждого компонента контента системы придают определенное вероятностное значение. Оно определяет критическую точку между двумя состояниями изучаемого параметра. Затем на практике получают тем или иным исследовательским способом фактическое значение, которое переводят в вероятностную шкалу. Вектор надежности получают как результат применения мультипликативной функции к фактическим значениям элементов контента. Он будет характеризовать связь между составляющими контента. Вектора надежности, полученные для всех компонентов рассматриваемой подсистемы, позволяют построить матрицу надежности. Она будет определять функциональную устойчивость относительно задаваемых начальных вероятностных параметров. Применение соответствия Галуа к изучаемым подсистемам контента информационной системы дает возможность определить степень влияния одних содержательных компонентов подмножеств на другие подмножества. А в совокупности с другими параметрами выявить базовые и критические значения для рассматриваемых компонентов системы.

Инварианты можно рассматривать и между отдельными множествами информационной системы. Так инвариант «прочность связи» характеризует количество связей между элементами информационной системы, то есть число ребер между вершинами в ее графовой модели. Чем

больше таких связей, тем прочнее связь между элементами. Также определяют и прочность связи для всей графовой модели как наименьшее значение из значений прочности связей между всеми элементами. Это позволяет выделить подсистемы мультиграфовой модели целостной информационной системы и определить их взаимное влияние друг на друга. В этом случае соответствие Галуа выступает инструментом исследования, как структуры, так и ее содержимого системы. С одной стороны оно также оперирует количеством связей между системными элементами. С другой выделение контента с помощью соответствия Галуа в анализируемых подсистемах отвечает критериям конструктивной полноты, а, следовательно, и критериям ее стабильного функционирования.

Инвариант «слабая перемычка» задает множество всех связей между элементами двух подграфов информационной системы, которые не имеют общих вершин. Данный параметр определяет качество связей и в совокупности с соответствием Галуа отражает общую картину отношений среди различных подсистем системы в целом.

Инвариант «прочность слабой перемычки» задает параметр, который характеризуется количеством ребер содержащихся во множестве «слабой перемычки». Иными словами это количественный показатель инварианта «слабая перемычка», трактуемый как числовой показатель связи между рассматриваемыми множествами информационной системы. Данный показатель целесообразно анализировать совместно с показателем прочности связей между компонентами изучаемых подсистем с помощью соответствия Галуа. В этом случае данный инструмент анализа позволяет сопоставить связи между множествами связям между отдельными их элементами, выявить степень корреляционного влияния. При этом изначально количественный анализ получает качественные трактовки изучаемый связей на контенте информационной системы.

Инвариант «вектор прочности графа» определяет набор числовых значений, отражающий показатели прочности различных множеств «слабой перемычки» для рассматриваемых двух подсистем информационной системы. Этот вектор характеризует коэффициенты прочности, позволяет их сравнивать между собой. Это позволяет с использованием соответствия Галуа либо ослаблять, либо усиливать связи контента друг с другом в зависимости от ряда внутренних и внешних факторов, влияющих на устойчивое функционирование системы.

Итак, соответствие Галуа совместно с инвариантами теории графов, интерпретируемые к структуре и контенту информационных систем, позволяют предложить теоретическую основу их исследования. Соответствие Галуа на основе анализа собственных количественных показателей и числовых параметров инвариантов графовых моделей также представляет и инструмент изучения качественных интерпретаций контента информационной системы. Такая теоретико-практическая направленность инварианта соответствия Галуа, как инструмента исследования функционирования структурных компонентов системы от их наполнения контентом, определяет актуальность и значимость его использования при проектировании и сопровождении информационных систем.

Литература

1. Козлов С. В. Возможности и особенности построения автоматизированных дидактических систем // Математическая морфология: электронный математический и медико-биологический журнал. – Т. 10. – Вып. 3. – Смоленск: СГМА, 2011.
2. Баженов Р. И., Корнилков А. П., Лопатин Д. К. Проектирование web-ориентированной информационной системы университета на основе клиент-серверных технологий // Актуальные проблемы гуманитарных и естественных наук. – 2014. – № 4-1. – С. 68-71.
3. Козлов С. В. Актуальные вопросы развития инновационных информационных технологий и систем в образовании // Проблемы и перспективы инновационного развития территорий: материалы международной научно-практической конференции профессорско-преподавательского состава. – Ч.1. – Коломна: МГОСГИ, 2013. – С.173-176.
4. Емельченков Е. П., Бояринов Д. А., Козлов С. В. Информационные системы автоматизированной поддержки инновационной деятельности: модели, проектирование и реализация. – Смоленск: Изд-во СмолГУ, 2011. – 164 с.
5. Киселева О. М. Преподавание дисциплины «Современные системы управления базами данных» в ВУЗе для направления подготовки «Социальная работа» // Психология, социология и педагогика. – 2014. – № 7 (34). – С. 43-45.
6. Козлов С. В., Емельченков Е. П. Соответствия Галуа. САПР учителя // Системы компьютерной математики и их приложения. – Смоленск: СмолГУ, 2006. С. 100-102.
7. Кон П. М. Универсальная алгебра / П. М. Кон; пер. с англ. Т. М. Баранович; под ред. А. Г. Куроша. – М.: Мир, 1968. – 351 с.
8. Козлов С. В. Особенности применения системы индивидуального тестирования «Комплекс измерения обученности» в школьном курсе информатики // Системы компьютерной математики и их приложения. –

Смоленск: СмолГУ, 2008. С. 247-251.
9. Козлов С. В. Программный комплекс «Advanced Tester»: проектирование индивидуальных тестов в автоматизированной информационной системе // Современная педагогика. 2014. № 9 [Электронный ресурс]. URL: http://pedagogika.snauka.ru/2014/09/2696 (дата обращения: 29.10.2014).
10. Козлов С.В. Математические аспекты выбора оптимального набора тестовых заданий индивидуального теста // Психология, социология и педагогика. – 2014. – № 9 (36) [Электронный ресурс]. URL: http://psychology.snauka.ru/2014/09/3603 (дата обращения: 07.10.2014).
11. Андреева А.В., Максимова Н.А. ИСУ ВУЗ как инструмент управления качеством образования // В мире научных открытий. – 2013. – № 11.8 (47). – С. 22-28.
12. Козлов С. В. Система индивидуального тестирования «Комплекс измерения обученности» // Системы компьютерной математики и их приложения. – Смоленск: СмолГУ, 2007. С. 223-225.
13. Козлов С. В. Функциональные назначения и возможности информационно-образовательного ресурса «Advanced Tester» // Горизонты науки. – 2011. – №2 (6). – С. 9-12.
14. Козлов С. В. Актуальные вопросы использования адаптивных информационно-образовательных систем в профильной школе // Наука и образование в XXI веке: сборник научных трудов по материалам международной научно-практической конференции 30 сентября 2013 г.: в 34 частях. – Ч. 21. – Тамбов: Бизнес-Наука-Общество, 2013. – С. 48-51.
15. Козлов С. В. Вопросы проектирования инновационных автоматизированных информационных систем: анализ элементов графовых моделей // Система обеспечения качества образования: модели, технологии, анализ: материалы научно-методической конференции. – Смоленск: Смоленский филиал РГТЭУ, 2012. С. 53-57.

Пеленицын А.М.

Южный федеральный университет, Институт математики, механики и компьютерных наук им. И.И. Воровича, ассистент кафедры информатики и вычислительного эксперимента,
apel@sfedu.ru

КОНЦЕПТЫ C++17 В ИХ ОТНОШЕНИИ К КОНЦЕПТАМ C++0X

КЛЮЧЕВЫЕ СЛОВА

Обобщённое программирование, концепты C++, шаблоны C++, контроль типов

АННОТАЦИЯ

В статье проводится анализ текущего проекта концептов для C++ 2017 на основе выявления проблем концептов C++0x, которые планировалось включить в стандарт языка 2011 года. Показано, что неудача концептов C++0x напрямую повлияла на конкретные проектные решения, вошедшие в новый дизайн концептов.

Неограниченный параметрический полиморфизм, как он представлен в механизме шаблонов C++98, имеет ряд известных недостатков, главным из которых является невозможность генерации разборчивых сообщений об ошибках. На протяжении 2000-х и 2010-х рассматриваются средства введения ограниченного параметрического полиморфизма в язык C++. Мы исследуем наиболее актуальное из таких предложений — способ определения и применения ограничений на параметры шаблонов C++, так называемые концепты C++1z (также: Concepts Lite), планируемые к включению в стандарт языка 2017 года.

Все рассмотренные нами источники по дизайну концептов C++1z имеют общий недостаток, а именно, в описании нового механизма практически не говорится о том, какие факторы повлияли на принятие конкретных проектных решений. На наш взгляд, важнейшие из этих решений продиктованы проблемами, возникшими в старом дизайне концептов C++0x. Анализ этой связи между двумя проектами позволяет лучше понять основные черты готовящихся изменений в языке C++. Именно этот анализ предлагается в нашей статье.

В рассмотрении проекта концептов C++1z мы используем ряд документов комитета по стандартизации C++: некоторые примеры взяты из неформального введения в тему [1]; детали синтаксиса уточнены по более свежему формальному документу-описанию изменений в текст стандарта языка [2]; изменения в стандартной библиотеке пока ещё не пересмотрены в свете последней редакции концептов, но некоторое представление о них можно получить по отчёту [3]. Мнение автора языка C++, Бьярне Страуструпа приводится в основном по статье [4].

Статья состоит из пяти частей. В первой части приводится пример, который должен иллюстрировать основную проблему неограниченных шаблонов C++98 и преимущества, которые предполагается получить после введения концептов в новый стандарт языка. Во второй части приводится краткий обзор старого проекта концептов C++0x и основные причины его неудачи — невключения в стандарт 2011 года. В третьей части описываются главные черты нового проекта концептов и их связь с проблемами старого проекта. В четвёртой части рассматривается один из важных элементов ограниченного параметрического полиморфизма: перегрузка на основе концептов, и то, как авторам нового проекта удалось реализовать её в условиях отсутствия явных моделей концептов. В пятой части даны замечания по текущему состоянию нового проекта — деталям синтаксиса и доступности его реализации.

1. Пример и мотивация

Одна из главных проблем шаблонов C++98 состоит в длинных ошибках компиляции. Происходят они от того, что несоответствие типа неявным требованиям выясняется посреди процесса подстановки этого типа в определение шаблона. В результате текст ошибки ссылается на детали реализации шаблона, которым не смог удовлетворить тип-аргумент. Однако клиент шаблона, как правило, ничего не знает о его реализации.

Типичный пример такого несоответствия можно составить с помощью распространённой ошибки, которая состоит в попытке отсортировать список общим алгоритмом. В стандартной библиотеке C++ общий алгоритм сортировки подразумевает произвольный доступ к элементам сортируемого контейнера, в то время как тип списка такой доступ не предоставляет. Приведём

пример заголовка шаблона функции сортировки, который несколько проще стандартного алгоритма и принимает не пару итераторов, а контейнер целиком — в трёх вариантах (в своей реализации такой шаблон может, к примеру, вызывать стандартный алгоритм std::sort).

Листинг 1: Функция сортировки контейнера: неограниченный полиморфизм

```
template<typename Cont>
void sort(Cont & c);
```

В листинге 1 ничто не указывает на предположения о типе-параметре кроме его имени — типичная ситуация для неограниченного полиморфизма.

Листинг 2: Функция сортировки контейнера: ограниченный полиморфизм, полная и сокращённая версии

```
template<typename Cont>
   requires Sortable<Cont>()
void sort(Cont & c);

template<Sortable Cont>
void sort(Cont & c);
```

В листинге 2 приведены две эквивалентных версии в синтаксисе концептов, предлагаемом для C++17, причём вторая, сокращённая, совпадает также и с вариантом, предлагавшимся для концептов C++0x. В ней видно использование явного ограничения Sortable — это и есть именованный набор требований к шаблонному параметру Cont, который называется *концептом*. В данном случае набор требований должен обеспечивать произвольный доступ к элементам контейнера и наличие операции «меньше» для этих элементов.

Наиболее важным свойством концептов является то, что проверка соответствия аргумента заданным требованиям проводится до того, как начинается подстановка этого аргумента в тело шаблона. Таким образом, при возникновении несоответствий компилятор сообщит о них незамедлительно и в терминах ограничений, представленных в интерфейсе обобщённой компоненты, а не её реализации. Имеющиеся прототипы реализаций концептов C++0x и C++1z дают существенно более доступные сообщения об ошибках в подобных случаях, чем это было с неограниченными шаблонами C++98. Проиллюстрируем это на примере с сортировкой списка..

Сообщение об ошибке с вызовом стандартного алгоритма sort для стандартного списка в компиляторе GCC 4.9 занимает 67 строк, непосредственно текст ошибки плохо отражает существо проблемы, выглядит следующим образом:

no match for 'operator-' (operand types are 'std::_List_iterator<int>' and 'std::_List_iterator<int>')

Ошибка сопровождается большим количеством «замечаний» (англ. note), которые должны помочь программисту понять причину ошибки, но на деле имеет малую ценность. Аналогичная ошибка с концептами C++1z занимает около 6 строк, и выглядит следующим образом:

```
error: cannot call function 'void sort(const C&) [with C = std::__cxx11::list<int>]'
note:   constraints not satisfied
 void sort(C const & ) {}
      ^
note:   concept 'Sortable<std::__cxx11::list<int, std::allocator<int> > >' was not satisfied
```

В данном случае замечания совершенно точно отражают суть дела.

Концепты это новая сущность языка программирования, которая не укладывается в систему типов C++98. Это отличает механизм ограниченного параметрического полиморфизма, предлагаемый для C++, от ограниченного параметрического полиморфизма, основанного на подтипировании и характерного для чистых объектно-ориентированных языков, таких как Java и C#. (Отметим, что последний упомянутый подход неоднократно критиковался с предложением улучшений [5–7].) В связи с этим мы считаем важным всесторонний анализ дизайна концептов для C++ и предлагаем вариант такого анализа на основе сравнения текущего проекта концептов с проектом, предлагавшимся для включения в C++11. На наш взгляд основные черты нового проекта прямо вытекают из проблем, возникших в проекте концептов C++0x.

2. Концепты C++0x и их проблемы

Обсуждение концептов C++1z следует начать с анализа целей и причин неудачи концептов C++0x. Кратко перечислим основные элементы проекта концептов C++0x, которые одновременно являются достаточно общими требованиями к механизму ограниченного параметрического

полиморфизма.

1.Синтаксис для определения именованного набора ограничений — концепта.

2.Синтаксис для наложения ограничения (концепта) при объявлении шаблона класса или функции.

3.Способ проверки соответствия использования шаблонных параметров внутри шаблона класса или функции наложенным ограничениям.

4.Способ указания соответствия конкретных типов тем или иным концептам — так называемые *модели концептов* (concept map).

Оказалось, что основные сложности доставляли пункты 3 и 4. Первый из них требовал существенной модификации алгоритма поиска имён при обработке шаблонов компилятором. Такая модификация была выполнена в рамках проекта ConceptGCC [8], работала она весьма медленно и нестабильно. Мнение разработчиков этой версии состоит в том, что при необходимой доработке, проблем с производительностью компилятора быть не должно. Однако сам масштаб изменений, которые требовалось внести в распространённые компиляторы в отношении обработки шаблонов, негативно воспринимался членами комитета по стандартизации.

Второй проблемный пункт (4-й в списке выше) с моделями концептов неоднозначно воспринимался даже основным идеологом концептов C++0x, автором языка C++ Бьярне Страуструпом. Дело в том, что свидетельства соответствия конкретных типов некоторым концептов во многих случаях тривиальны и могут быть автоматически установлены компилятором. Однако такой автоматизм может плохо сочетаться с другими возможностями обновлённых шаблонов, как например, перегрузкой на основе концептов. Более того, в комитете по стандартизации высказывались мнения, что получение моделей концептов без участия, а значит, без явного намерения программиста просто неверно идеологически.

В целом, по состоянию на 2009 год проект концептов C++0x оценивался Б. Страуструпом как переусложнённый, рассчитанный на программистов-экспертов, в то время как изначальная идея состояла в создании механизма, облегчающего жизнь «среднему» программисту. Автор языка C++ предложил пять радикальных шагов по упрощению концептов, однако комитет по стандартизации не видел возможности обсуждать дополнительные предложения до принятия нового стандарта: сроки для внесения изменений подходили к концу. В итоге концепты были оставлены для будущих редакций стандарта языка.

3. Концепты C++1z в их отношении к концептам C++0x

Проект концептов для стандарта 2017 года был создан с нуля, но в соответствии с опытом, полученным при разработке концептов C++0x. Основные задачи, стоявшие перед авторами нового проекта, можно сформулировать следующим образом.

- Простота и понятность для «среднего» программиста.
- Отказ (возможно, временный) от средств, породивших наибольшие споры при обсуждении концептов C++0x.
- Учёт объективной сложности адаптации промышленных компиляторов.
- Обратная совместимость со стандартом языка 1998 года, в том числе, с неограниченными шаблонами (как и у концептов C++0x).

В этом разделе мы укажем две черты концептов C++1z, который вытекают из этих целей и отличают их от концептов C++0x: они связаны в первую очередь с двумя проблемными пунктами списка, приведённого в прошлом разделе. В следующих двух разделах обсуждаются другие черты концептов C++1z, важные с точки зрения обобщённого программирования.

Отсутствие контроля типов внутри шаблонов — возможно, самая неожиданная черта концептов C++1z, если смотреть на них без учёта изложенной выше предыстории. Контроль типов внутри шаблонов представлял потенциально особую сложность для адаптации промышленных компиляторов, как упоминалось в предыдущем разделе. Связано это, в частности, с тем, что новые ограниченные шаблоны должны были успешно сосуществовать и взаимодействовать со старыми, неограниченными шаблонами C++98. Если, к примеру, из шаблона первого типа вызывался шаблон второго, то проконтролировать корректность требований внешнего шаблона было достаточно сложно: фактически, задача свелась бы, как и прежде, поиску ошибок путём подстановки шаблонных аргументов. Обработка подобных сочетаний существенно усложняла комиплятор, не принося никакого выигрыша по сравнению со старым подходом стандарта 1998 года.

Таким образом, в концептах C++1z было решено полностью исключить проверку соответствия требований, наложенных в заголовке шаблона, и определения этого шаблона. То

есть проверяется лишь соответствие аргументов шаблона в месте его использования, в то время как в теле шаблона параметры обрабатываются компилятором так же, как это было с шаблонами C++98. Данный подход считается разумным компромиссом в связи с тем, что автор обобщённой компоненты зачастую более квалифицированный специалист, чем её пользователь: он должен сформулировать набор ограничений к аргументам своей компоненты и написать её определение так, чтобы оно соответствовало этим требованиям, не рассчитывая на помощь компилятора в этой задаче. Далее, многочисленные использования однажды написанной компоненты будут уже проверяться именно компилятором.

Этим изменением объясняется различие двух проектов концептов с точки зрения модулярного контроля типов. Модулярный контроль типов означает, что если определение обобщённой компоненты было однажды принято компилятором, то при любой подстановке аргументов, удовлетворяющих ограничениям, повторная проверка определения не требуется. В концептах C++0x это условие удовлетворялось лишь частично по причине возможного использования неограниченных шаблонов внутри ограниченных. В концептах C++1z вначале проверяется соответствие аргументов требованиям, однако во время подстановки могут произойти ошибки в том случае, если автор шаблона неточно описал эти требования.

Полностью автоматическая проверка соответствия типов наложенным ограничениям или отказ от моделей концептов — вторая важная черта, отличающая концепты C++1z. Как упоминалось в прошлом разделе, механизм моделей концептов вызывал много вопросов, связанных с запретом или разрешением генерации таких моделей. В проекте для C++1z было решено максимально упростить использование концептов с помощью удаления самого понятия моделей концептов.

Таким образом, если тип не может быть однозначно и автоматически отображён на нужный набор требований, то он будет отвергнут компилятором. Имеющийся способ решения этой проблемы — модифицировать рассматриваемый тип нужным образом, зачастую не представляет трудностей, но разрушает совместимость с кодом, который уже использует старое определение данного типа. Эта проблема непосредственно связана с одним из важных свойств обобщённого программирования, известным как ретроактивное моделирование. Ретроактивное моделирование предполагает возможность определения того, каким образом тип удовлетворяет концепту, без изменения типа. С удалением моделей концептов говорить о поддержке этого свойства если и можно, то только в очень ограниченном объёме, а именно: всегда можно добавлять перегруженные версии свободных функций, работающих с объектами данного типа. Однако наличие свободных функций является лишь одним из нескольких типов требований, которые могут накладывать концепты.

Завершая сравнение концептов C++0x и концептов C++1y, отметим, что в последних по сравнению с первыми были исключены также:

- аксиомы — средство описания семантических ограничений,
- синтаксическая поддержка уточнения (refinement) или, попросту, «наследования» концептов.

Аксиомы это достаточно высокоуровневые правила, которые в общем случае невозможно проверить с помощью компилятора. Пример аксиомы приведён в листинге 3: здесь показано, что полугруппой является пара из типа T и бинарной операции над элементами этого типа, причём операция должна быть ассоциативной — это условие сформулировано в виде аксиомы.

Аксиомы предполагались в помощь компилятору для проведения оптимизирующих преобразований. Соответствующие исследования проводились несколькими исследователями, поддерживавшими эту идею, однако число их было слишком мало, чтобы обеспечить вхождение механизма с таким нечётким назначением в новый упрощённый проект концептов.

Листинг 3: Аксиома ассоциативности — часть определения концепта полугруппы

```
concept Semigroup<typename Op, typename T> :
    CopyConstructible<T> {
  T operator()(Op, T, T);
  axiom Associativity(Op op, T x, T y, T z) {
    op(x, op(y, z)) == op(op(x, y), z);
  }
}
```

Пример уточнения также приведён на листинге 3: тип T в концепте полугруппы должен иметь конструктор копий (CopyConstructible<T>). Как замечено Дугом Грегором, одним из авторов концептов C++0x, в предложении по их упрощению [11], в отсутствии моделей концептов уточнения полностью эквивалентно ограничениям, которые можно накладывать на тип внутри определения концепта.

4. Перегрузка на основе концептов и связанная с ней проблема

В связи с удалением из нового дизайна концептов явных моделей возникает следующая проблема.

В техническом отчёте 2005 года ведущие специалисты в области обобщённого программирования, Дуг Грегор и Джереми Сик, высказали идею о том, что для успешной реализации перегрузки на основе концептов необходимо наличие механизма явного сопоставления концепта и типа, который должен ему удовлетворять. Пример Грегора и Сика достаточно прост и интересен, чтобы его рассмотреть и понять, как решают возникающую в нём проблему авторы концептов C++1z. Отметим сразу, что хотя упомянутое решение можно найти в описании проекта изменений в стандартную библиотеку в связи с запланированным добавлением концептов C++1z, само описание изменений библиотеки весьма обширно, а авторы новых концептов не вынесли это важное, по нашему мнению, решение ни в одну из известных нам статей или докладов, посвящённых новому элементу языка.

Пример, по существу, ставит проблему концептов, которые различаются только семантически. В качестве примера таких концептов можно взять широко распространённые в стандартной библиотеке C++ категории итераторов, которые планируется конвертировать в концепты в будущем стандарте. Хорошо известно, что итераторы ввода и прямые итераторы отличаются лишь семантикой операции инкремента.

Листинг 4: Перегрузка на основе концептов: конструктор типа vector с разной эффективностью

```
// O(lg n) allocations
template<Input_iterator Iter>
vector(Iter first, Iter last)
{
   while (first != last) push_back(*first++);
}

// 1 allocation
template<Forward_iterator Iter>
vector(Iter first, Iter last)
{
   auto n = distance(first, last);
   reserve(n);
   while (n--) push_back(*first++);
}
```

В листинге 4 приведён пример перегрузки функций-членов, а именно, конструкторов типа vector, принимающих итераторы начала и конца диапазона, элементы которого надо загрузить в создаваемый вектор. Перегрузка осуществляется на основе категории итератора, который задаётся концептом: для итератора ввода необходимо многократно выделять (O(lg$n$) выделений, если расширение происходит с помощью кратного увеличения резерва памяти), в то время как прямой итератор позволяет заранее вычислить количество элементов в полученном диапазоне и обойтись одним выделением.

Проблема состоит в том, что, как отмечалось, выше любой тип, (синтаксически) входящий в категорию итератора ввода, будет принадлежать также и категории прямого итератора. Как показал анализ предложения по редактированию стандартной библиотеки в связи с планируемым добавлением концептов, семантическое различие между разными типами авторы планируют, как и в старом стандарте кодировать заранее определёнными типами-метками (tag), для которых внутри типов-итераторов задаётся стандартный псевдоним iterator_category. На листинге 5 показано, как примерно должен выглядеть при таком подходе концепт итератора ввода.

Из листинга 5 видно, что данный концепт уточняет (refines) концепт WeakInputIterator, который включает некоторые элементарные свойства итераторов, такие как наличие операций разыменования и продвижения. Ограничения на тип-параметр I состоят в том, что объекты этого

типа должны быть сравнимы на равенство (EqualityComparable) и должны содержать вложенный синоним типа под названием iterator_category (его значение возвращает выражение IteratorCategory<I>), который наследуется (в частности, как часто бывает, равен) от стандартного типа-метки input_iterator_tag.

Листинг 5: Концепт итератора ввода

```
template<typename I>
concept bool InputIterator =
  WeakInputIterator<I>() && EqualityComparable<I>() &&
  Derived<IteratorCategory<I>, input_iterator_tag>();
```

Заметим, что строгое равенство типов обеспечивается аналогичным Derived отношением Same. Оба они вычисляются с помощью средств заголовочного файла type_traits стандартной библиотеки, появившегося в C++11, и, таким образом, не требуют существенной модификации существующих компиляторов.

5. Заключительные замечания относительно синтаксиса и реализации концептов C++1z

В предыдущих разделах мы не останавливались на специальном обсуждении синтаксиса для концептов, отдавая основное внимание семантике. Синтаксис концептов продолжает меняться и любые конкретные определения здесь могут быстро устареть. Однако для понимания языка описания концептов (пункт 1 из списка в разделе 2) следует сделать общее замечание относительно идеи введения новых синтаксический конструкций в язык программирования. Идея эта в целом заключалась в том, чтобы максимально использовать новые возможности из последних стандартов языка: 2011 и 2014 года.

Ключевое слово concept может появляться в одном из двух контекстов:

- в определении шаблона constexpr-функции (появились в C++11) — такой пример даётся в листинге 6,
- в определении шаблона переменной (появились в C++14) — такой пример приводится в листинге 5.

Листинг 6: Концепт Аллокатора (менеджера памяти)

```
template<typename A>
concept bool Allocator()
{
  return requires () {
    typename A::pointer;              // (1)
    requires Pointer<typename A::pointer>; // (2)
    // ...
  };
}
```

В обоих случаях это означает наложение некоторых ограничений на соответствующий шаблон функции или переменной. К примеру:

- тип возвращаемого значения шаблона функции, как и тип шаблона переменной должен быть логическим типом;
- у шаблона функции не должно быть параметров;
- в теле функции должен быть только оператор return, аргументом которого выступает логическое выражение, вычисляемое на этапе компиляции, возможно, включающее requires-подвыражение;
- и т. п.

Наиболее существенное добавление к синтаксису C++ происходит в связи с requires-выражением, оно может включать два вида предложений: обращение к вложенным типам (таким образом требуется их существование) или допустимые выражения. Пример второго вида требований приведён на листинге 7: requires позволяет вводить псевдопеременные параметра-типа T (в этом примере их имена a и b) и требовать, чтобы выражения сравнения на равенство и на неравенство были допустимыми для этих переменных и возвращали тип bool.

Листинг 7: Пример использования допустимых выражений

```
template<typename T>
concept bool Equality_comparable()
{
   return requires (T a, T b) {
      {a == b} -> bool;
      {a != b} -> bool;
   };
}
```

Приведённый синтаксис имеет ряд очевидных мелких недостатков. Например, указание типа bool в обоих видах концептов является избыточным. Кроме того: для использования концепта, определённого как функция, требуется указание круглых скобок (операции вызова функции), в то время как для переменной они не нужны и приведут к ошибке компиляции, если их добавить — это создаёт ненужное синтаксическое различие между разными концептами в то время как семантически они абсолютно эквиваленты.

Вполне возможно, что указанные недостатки будут сняты в будущих редакциях проекта нового стандарта [12].

В августе 2015 года прототипная реализация концептов на основе компилятора GCC была добавлена в основную ветвь разработки этого компилятора [13]. Это изменение планируется зафиксировать в версии 6 компилятора, которая запланирована к выходу весной 2016 года. Однако уже сейчас можно самостоятельно собрать данную версию компилятора из исходных кодов либо воспользоваться онлайн-ресурсами, предоставляющими веб-интерфейс к такой сборке [14]. Для включения механизма концептов при компиляции программ следует указывать ключ -std=C++1z.

Литература

1. *Sutton A., Stroustrup B., Dos Reis G.* Concepts Lite: тех. отч. — 28 июня 2013. — ISO/IEC JTC1/SC22/WG21 N3701.URL: http://www.open-std.org/jtc1/sc22/wg21/docs/papers/2013/n3701.pdf
2. *Sutton A.* C++ Extensions for Concepts PDTS: тех. отч. — 9 февр. 2015. — ISO/IEC JTC1/SC22/WG21 N4377.URL: http://www.open-std.org/jtc1/sc22/wg21/docs/papers/2015/n4377.pdf
3. A Concept Design for the STL: тех. отч. — 13 янв. 2012. — ISO/IEC JTC1/SC22/WG21 N3351. URL: http://www.open-std.org/jtc1/sc22/wg21/docs/papers/2012/n3351.pdf
4. *Stroustrup B.* The C++0x "Remove Concepts" Decision // Dr. Dobb's Journal. — 2009.URL: http://www.drdobbs.com/cpp/the-c0x-remove-concepts-decision/218600111
5. *Järvi J., Willcock J., Lumsdaine A.* Associated types and constraint propagation for mainstream object-oriented generics // OOPSLA '05: Proceedings of the 20th annual ACM SIGPLAN conference on Object- oriented programming systems languages and applications. — ACM Press. New York, NY, USA : ACM Press, 2005. — С. 1–19. — ISBN 1-59593-031-0. — DOI: 10.1145/1094811.1094813.
6. *Belyakova J., Mikhalkovich S.* Pitfalls of C# Generics and Their Solution Using Concepts // Proceedings of the Institute for System Programming. — Moscow, Russia, 2015. — Июнь. — Т. 27, No 3. — С. 29–45. — ISSN 2079-8156.
7. Lightweight, flexible object-oriented generics / Y. Zhang [и др.] // 36th ACM SIGPLAN Conf. on Programming Language Design and Implementation (PLDI). — Июнь 2015. — С. 436–445. URL: http://www.cs.cornell.edu/andru/papers/genus
8. ConceptGCC. — URL: http://www.generic-programming.org/software/ ConceptGCC/
9. Concepts: Linguistic Support for Generic Programming in C++ / D. Gregor [и др.] // SIGPLAN Not. — New York, NY, USA, 2006. — Окт. — Т. 41, No 10. — С. 291–310. — ISSN 0362-1340. — DOI: 10.1145/1167515.1167499.
10. *Tang X., Järvi J.* Axioms as generic rewrite rules in C++ with concepts // Science of Computer Programming. — 2015. — Т. 97, Part 3. — С. 320–330. — ISSN 0167-6423. — DOI: 10.1016/j.scico.2014.05.006. — Object-Oriented Programming and Systems (OOPS 2010)Modeling and Analysis of Compositional Software (papers from EUROMICRO SEAA'12).
11. *Gregor D.* Simplifying C++0x Concepts: тех. отч. — 9 апр. 2013. — ISO/IEC JTC1/SC22/WG21 N3629. URL: http://www.open-std.org/jtc1/sc22/wg21/docs/papers/2013/n3629.pdf
12. *Brown W.* Tweaks to Streamline Concepts Lite Syntax: тех. отч. — 10 апр. 2015. — ISO/IEC JTC1/SC22/WG21 N4434. URL: http://open-std.org/JTC1/SC22/WG21/docs/papers/2015/n4434.pdf
13. *Merrill J.* Huge C++ PATCH to merge c++-concepts branch — Mail archive for the GCC project. URL: https://gcc.gnu.org/ml/gcc-patches/2015-08/msg00377.html
14. Wandbox: Social Compilation Service. — URL: http://melpon.org/wandbox

Рамиль Альварес Х.[1], Владимирова Ю.С.[2]

[1]МГУ им. М.В.Ломоносова, ф-т ВМК, ramil@cs.msu.ru

[2]МГУ им. М.В.Ломоносова, ф-т ВМК, vladimirova@cs.msu.ru

ОБ ИЗВЛЕЧЕНИИ КВАДРАТНОГО КОРНЯ В ТРОИЧНОЙ СИММЕТРИЧНОЙ СИСТЕМЕ

КЛЮЧЕВЫЕ СЛОВА

Троичная симметричная система, извлечение квадратного корня.

АННОТАЦИЯ

В статье [1] описаны алгоритмы деления и извлечения квадратного корня типа «цифра за цифрой» в троичной симметричной системе (ТСС). К сожалению, в алгоритме извлечения квадратного корня автором допущена ошибка, которая исправлена при описании алгоритма в [2], в работе [3] приведено описание алгоритма на языке С [4]. Однако эти работы малодоступны, а в интернете [5] приводятся ошибочные алгоритмы деления и извлечения квадратного корня в ТСС. Эти ошибки носят методологический характер и связаны с перенесением с некоторой корректировкой на алгоритмы в ТСС методов и приемов из двоичной или десятичной систем.

Общие правила извлечения квадратного корня

Как и при вычислении квадратного корня в десятичной системе, подкоренное число N разбивается на пары цифр (тритов) влево и вправо от точки. Обозначим через N_i целое число, получаемое из первых слева направо i пар тритов, A_i – целое значение квадратного корня из этого числа, D_i – i-ая пара тритов, d_i – значение i-го трита корня и n – число пар тритов. Если старшая пара состоит из одного трита, добавим к ней слева трит 0. Отметим, что действия производятся над целыми числами, а точка в результате будет после k-го трита, если в подкоренном числе она стояла после k-ой пары.

Если рассмотреть теперь подкоренное число как число с точкой перед первой парой, имеем число в интервале (.166… : .5), корень из которого будет в интервале (.408… : .707…). В ТСС числа, удовлетворяющие $0 < x < .5$ имеют нулевой трит, равный 0, числа удовлетворяющие $.5 < x < 1.5$ – нулевой трит, равный 1, откуда, нулевой трит корня может быть 0 или 1. Таким образом, в ТСС корень из числа, состоящего из n пар тритов, может состоять из $n + 1$ тритов. Например, корень из числа из одной пары тритов 11 (4 в десятичной системе) равен $1\,\bar{1}$ (2 в десятичной системе), числу из двух тритов ($\bar{1}$ в ТСС обозначается –1).

Схему алгоритма извлечения квадратного корня «в столбик» для ТСС можно описать следующими формулами:

$A_0 = d_0$, где $d_0 = 0$ или 1,

$B_0 = -d_0$,

$N_i = A_i^2 + B_i$,

$C_i = B_{i-1}3^2 + D_i$,

$B_i = C_i - (2A_{i-1}3 + d_i)d_i$,

$A_i = A_{i-1}3 + d_i$.

Заметим, что $B_{i-1}3^2 + D_i$ при ручном использовании алгоритма выполняется приписыванием справа к значению B_{i-1} двух тритов пары D_i, а $2A_{i-1}3 + d_i$ и $A_{i-1}3 + d_i$ – трита d_i соответственно к числам $2A_{i-1}3$ и $A_{i-1}3$.

Для вычисления значения квадратного корня приведенную схему необходимо дополнить правилом выбора очередной цифры d_i.

Определение d_0

Если подкоренное число, рассматриваемое как число с точкой перед первой парой, больше .25, корень больше, чем .5, т.е. нулевой трит корня d_0 будет 1. В противном случае корень меньше, чем .5, и нулевой трит корня d_0 будет 0. Значение B_0 в первом случае равно $\bar{1}$, во втором 0, согласно схеме алгоритма. Заметим, что .25 не представляется точно в ТСС и имеет представление

в виде бесконечной последовательности .(1 $\bar{1}$).

Обоснование правила выбора d_i

Для получения правила определения d_i, i = 1, 2, … покажем, что при d_i – i-ой цифре A_i значение $|B_i|$ минимально из трех возможных. Рассмотрим функцию дискретной трехзначной переменной

$$b_i(x) = C_i - (2\cdot 3A_{i-1} + x)x,$$

$x \in \{\bar{1}, 0, 1\}$. Отметим, что $b_i(d_i) = B_i$, где d_i – i-ая цифра $A_i = [\sqrt{N_i}]$.

Утверждение 1. В ТСС значение

$|b_i(x)| = |C_i - (2\cdot 3A_{i-1} + x)x|$, $x \in \{-1, 0, 1\}$

достигает минимума при $x = d_i$, i = 1, …, n.

Доказательство. Отметим, что на i-м шаге ($i \geq 1$) $A_i \geq 1$. В самом деле, при i = 1 старшая пара цифр D_1 подкоренного числа по условию больше нуля, т.е. $1 \leq D_1 \leq 4$. В силу того, что $N_1 = D_1$ и $A_1 = [\sqrt{N_1}]$, $A_i \geq 1$. На следующих шагах, при i = 2, …, n, $A_i = A_{i-1}3 + d_i$, откуда $A_i > 1$.

Оценим $|B_i|$. Из того, что

$$A_i = [\sqrt{N_i}],$$

следует:

$$B_i = N_i - A_i^2 = N_i - [\sqrt{N_i}]^2.$$

Для того чтобы оценить модуль разности $|N_i - [\sqrt{N_i}]^2|$, представим $\sqrt{N_i} = A_i + z$, где z – дробная часть $\sqrt{N_i}$:

$$N_i = (A_i + z)^2 = A_i^2 + 2A_iz + z^2,$$

откуда:

$$B_i = N_i - A_i^2 = 2A_iz + z^2 \qquad (1)$$

Так как в ТСС $|z| < 0.5$,

$$|B_i| < A_i + 0.25. \qquad (2)$$

Отметим также, что поскольку

$$A_i^2 = (3A_{i-1})^2 + (2\cdot 3A_{i-1} + d_i)d_i,$$

то B_i можно представить как:

$$B_i = N_i - (3A_{i-1})^2 - (2\cdot 3A_{i-1} + d_i)d_i.$$

Докажем, что $|B_i| = |b_i(d_i)|$ – минимально.

Предположим обратное: $|B_i| = |C_i - 2\cdot(3A_{i-1} + d_i)d_i|$ – не минимально, $d_i \in \{\bar{1}, 0, 1\}$ является i-ой цифрой A_i. Тогда существует число $t = \pm 1, \pm 2$, такое что $(d_i + t) \in \{\bar{1}, 0, 1\}$ и

$$|b_i(d_i + t)| < |B_i|.$$

Преобразуем:

$$\begin{aligned} b_i(d_i + t) &= N_i - (3A_{i-1})^2 - (2\cdot 3A_{i-1} + d_i + t)(d_i + t) = \\ &= N_i - (3A_{i-1})^2 - ((2\cdot 3A_{i-1} + d_i)d_i + (2\cdot 3A_{i-1} + t)t + 2d_it) = \\ &= B_i - ((2\cdot 3A_{i-1} + t)t + 2d_it) = \\ &= B_i - 2\cdot(3A_{i-1} + d_i)t - t^2 = B_i - (2A_it + t^2). \end{aligned}$$

Обозначив

$$R = 2A_it + t^2,$$

получим

$$b_i(d_i + t) = B_i - R.$$

По предположению, $|B_i|$ не минимально, т.е.

$$|b_i(d_i + t)| = |B_i - R| < |B_i|. \qquad (3)$$

Нетрудно убедиться, что при $R > 0$ условие $|B_i - R| < |B_i|$ выполняется, если $R < 2B_i$, а при $R < 0$, если $R > 2B_i$.

При $A_i > 1$, если $t > 0$, то и $R > 0$, а если $t < 0$, то и $R < 0$. При $A_i = 1$ это верно не при всех значениях t. Рассмотрим в связи с этим раздельно случаи $A_i > 1$ и $A_i = 1$.

Рассмотрим случай $A_i > 1$.

Допустим, что $t > 0$, тогда и $R > 0$. Согласно предположению (3), при $R > 0$ имеет место $R < 2B_i$.

Используя оценку (2), получим

$$R < 2B_i < 2A_i + 0.5. \qquad (4)$$

С другой стороны, т.к. $R = 2A_it + t^2$, наименьшее значение $R > 0$

$R = 2A_i + 1 > 2A_i + 0.5$,

что несовместимо с (4).

Допустим, что $t < 0$, тогда и $R < 0$, и согласно предположению (3),

$$R > 2B_i. \qquad (5)$$

Отсюда следует, что $B_i < 0$. Отметим также, что при $t < 0$, т.е. $t = -1$ или $t = -2$, i-ая цифра результата $d_i \in \{0, 1\}$.

Из (2) следует оценка для $2B_i$:

$$2B_i > -2A_i - 0.5.$$

В силу того, что B_i и A_i – целые числа, полученная оценка сводится к:

$$2B_i \geq -2A_i. \qquad (6)$$

Выясним, при каких условиях неравенство (5) и (6) совместимы, т.е. возможно

$$-2A_i \leq 2B_i < R.$$

R может принимать одно из двух значений: $R = -4A_i + 4$ (при $t = -2$) или $R = -2A_i + 1$ (при $t = -1$).

При $A_i > 1$, $t = -2$ выполнено $-4A_i + 4 \leq -2A_i$, т.е.

$R \leq -2A_i \leq 2B_i$,

что несовместимо с (5).

При $t = -1$ значение $R = -2A_i + 1$, откуда (5) выполнено только в случае, если $2B_i = -2A_i$, т.е. $B_i = -A_i$. В силу (1)

$$B_i = 2A_iz + z^2.$$

В рассматриваемом случае

$$2A_iz + z^2 = -A_i,$$

откуда

$$A_i = \frac{-z^2}{2z+1}. \qquad (7)$$

По предположению, $A_i > 1$, откуда число в правой части (7) положительно, если $2z + 1 < 0$ или

$$z < -.5,$$

что невозможно в силу того, что z – дробная часть числа в ТСС, и $|z| < .5$. Таким образом, при $A_i > 1$ и $t < 0$ невозможно $B_i = -A_i$, и предположение, что $R > 2B_i$ неверно.

Рассмотрим случай $A_i = 1$, $i = 1, \ldots, n$. Очевидно, что $A_i = 1$ может быть только при $i = 1$ и $A_0 = 0$. Тогда $d_1 = 1$, и

$$B_1 = C_1 - 2A_0 3d_1 - d_1^2 = D_1 - 1.$$

Поскольку

$$b_1(x) = C_1 - (2 \cdot 3A_0 + x)x = D_1 - x^2, x \in \{-1, 0, 1\},$$

значение $D_1 - 1$ минимально для $b_1(x)$. Утверждение 1 доказано.

Утверждение 2. Докажем, что

1) $|B_i|$ единственный минимум $|b_i(x)|$ при $A_i \geq 1$ ($i = 2, \ldots, n$) и при $A_0 = 1$ ($i = 1$);

2) $|b_i(x)|$ может иметь два минимума при $A_0 = 0$ ($i = 1$), только один из которых соответствует старшему триту результата, а он положительный.

Доказательство. 1) Предположим, что для некоторой цифры $d_i' \neq d_i$, $d_i' \in \{\bar{1}, 0, 1\}$ соответствующее ей значение

$$|b_i(d_i')| = |N_i - (3A_{i-1})^2 - (2 \cdot 3A_{i-1} + d_i')d_i'|$$

также минимально, т.е. $|b_i(d_i')| = |B_i|$. Рассмотрим функцию вещественной переменной

$$f(x) = |b_i(x)| = |N_i - (3A_{i-1})^2 - (2 \cdot 3A_{i-1} + x)x|$$

и исследуем ее на сегменте $[-1; 1]$.

Преобразуем $f(x)$:

$$f(x) = |-x^2 - 2 \cdot 3A_{i-1}x + N_i - (3A_{i-1})^2| = |x^2 + 2 \cdot 3A_{i-1}x - N_i + (3A_{i-1})^2|.$$

Функция $f(x)$ имеет два минимума, равных нулю, в точках

$$x_{1,2} = -3A_{i-1} \pm \sqrt{N_i}.$$

В силу того, что на i-ом шаге при $i = 2, \ldots, n$, $A_{i-1} > 0$ или при $i = 1$, $A_0 = 1$,

$$x_1 = -3A_{i-1} - \sqrt{N_i} < -3 < -1,$$

т.е. точка x_1 лежит вне сегмента $[-1; 1]$. Откуда функция $f(x)$ имеет на сегменте $[-1, 1]$ единственный минимум.

2) Рассмотрим случай $i = 1$, $A_0 = 0$. Функция $f(x) = |x^2 - N_1|$ достигает минимума на концах сегмента $[-1; 1]$. Значение $f(x)$ в точке $x_2 = 1$ соответствует положительному старшему триту

результата $d_i = 1$. Утверждение 2 доказано.

Определение d_i

Согласно доказанным утверждениям, в качестве очередного d_i выбирается троичная цифра, для которой $|b_i(x)| = /C_i - 2\cdot(3A_{i-1} + x)x|$ минимально, т.е. минимальное из трех значений:

$$|b_i(-1)| = |C_i + (2\cdot 3A_{i-1} - 1)|, \quad (8)$$

$$|b_i(0)| = /C_i|, \quad (9)$$

$$|b_i(1)| = |C_i - (2\cdot 3A_{i-1} + 1)|. \quad (10)$$

Знак в выражениях под модулем в (8) и (10) зависит от того $A_{i-1} \geq 1$ или $A_{i-1} = 0$, поэтому рассмотрим эти случаи раздельно.

Отметим, что при $C_i \geq 0$ в случае $A_{i-1} \geq 1$, ($i = 1, 2, \ldots, n$).

$$|C_i + (2\cdot 3A_{i-1} - 1)| > C_i,$$

$$|C_i + (2\cdot 3A_{i-1} - 1)| > |C_i - (2\cdot 3A_{i-1} + 1)|,$$

т.е. (8) не может быть минимальным, и $d_i \neq \overline{1}$.

Аналогично, при $C_i < 0$:

$$|C_i - (2\cdot 3A_{i-1} + 1)| > |C_i|,$$

$$|C_i - (2\cdot 3A_{i-1} + 1)| > |C_i + (2\cdot 3A_{i-1} - 1)|,$$

т.е. (10) не может быть минимальным, $d_i \neq 1$. Таким образом, если d_i отлично от 0, его знак совпадает со знаком C_i.

При $C_i \geq 0$ для определения d_i необходимо сравнивать (9) и (10). Раскрывая модули в этих выражениях, получим, что если $C_i < 3A_{i-1}$, то минимально (9), т.е. $d_i = 0$, а если $C_i > 3A_{i-1}$, то минимально (10), т.е. $d_i = 1$.

Аналогично, при $C_i < 0$ для определения d_i необходимо сравнивать (8) и (9), в результате чего получаем, что если $C_i < -3A_{i-1}$, то минимально (9), и $d_i = 0$, а при $C_i > -3A_{i-1}$ то минимально (8), и $d_i = \overline{1}$.

Рассмотрим случай $i = 1$, $A_0 = 0$. При этом

$$|b_1(0)| = |C_1|,$$

$$|b_1(-1)| = |b_1(1)| = |C_1 - 1|.$$

Согласно схеме алгоритма,

$$C_1 = B_0 3^2 + D_1,$$

где $B_0 = -d_0 = -A_0 = 0$, D_1 – старшая пара подкоренного числа, для которой верно $1 \leq D_1 \leq 4$. Таким образом, $C_1 = D_1$, $1 \leq C_1 \leq 4$ и

$$|C_1| < |C_1 - 1|,$$

откуда $d_i = 1$.

Обобщая, получим следующий критерий выбора очередного d_i: $d_i = 0$ при $|C_i| < 3A_{i-1}$, и $d_i = sign(C_i)$ при $|C_i| > 3A_{i-1}$.

Рассмотрим случай $|C_i| = 3A_{i-1}$, тогда B_i можно представить в виде:

$B_i = C_i - 2A_{i-1}3d + (.5)^2 - .25$,

где $d = .5$ для $C_i > 0$ и $-.5$ для $C_i < 0$. Значение d_i зависит от значения n_i – необработанной части числа N, рассматриваемой как число с точкой перед ее первым тритом. То, что функция квадратного корня возрастающая, позволяет определить значение d_i. При $C_i > 0$ и $n_i > .25$ невычисленная часть корня больше .5, т.е. d_i будет 1. При $C_i < 0$ и $n_i < .25$ невычисленная часть корня меньше $-.5$, т.е. d_i будет $\overline{1}$.

Описание алгоритма

Целое троичное число представляется в виде структуры

struct *trin* {**int** *n*; **char** *trit* [*K*]};

где n – число тритов, а массив *trit* – значение числа: младший трит хранится в элементе с индексом 0, элементы массива принимают значения –1, 0 и 1. Для числа нуль n равно нулю. Параметр алгоритма K – максимальная длина числа.

Вспомогательные функции

В виду простоты реализации функций *sgn*(*w*), *red*(&*res*), *shftb*(&*bi*,*x*,*i*) и *shiftc*(&*y*) их определение на языке С не приводится, даются только словесные описания.

Результатом функции **int** *sgn*(**struct** *trin w*) является знак числа *w*. Если n равно нулю, то результат будет нуль, в противном случае, в соответствии со свойством ТСС и принятым представлением числа, результатом будет значение элемента массива *trit* с индексом n–1.

Функция *void red*(**struct** *trin* **res*) приводит число *res* к принятому представлению, т.е. удаляет старшие нулевые триты структуры *res*. Она используется в функциях *shftb*(&*bi*,*x*,*i*) и *sum*3(*x*, *y*, &*res*).

Функция *void shftb*(**struct** *trin *bi*, **struct** *trin x*, **int** *i*) определяет новое значение *bi*. Ненулевое значение *bi* сдвигается влево на два трита. Тритам с индексами один и нуль присваиваются значения тритов структуры *x* с индексами, соответственно, *i* и *i*–1, после чего производится приведение числа к принятому представлению, т.е. обращение к функции *red*().

Функция *void shiftc*(**struct** *trin *y*) сдвигает влево на один трит значение структуры *y* и обнуляет трит с индексом нуль.

Функция *acmp*(*x*, &*y*) сравнивает модуль *x* и *y* (*y* > 0). Если длина *x* больше длины *y*, результат будет 1, а если длина *x* меньше длины *y*, результат будет –1. При равных длинах *x* и *y* сравниваются триты чисел *x* и *y* с учетом знака *x*, начиная со старших, до первого несовпадения, которое определяет результат. Если все триты совпали, то результат – нуль.

```
int acmp(struct trin x, struct trin * y) {
  int i = x.n, sx = sgn(x);
          if (i > y->n) return 1;
          if (i-- == y->n) {
            while (sx * x.trit[i] == y->trit[i] && i-- >= 0);
            if (i<0) return 0;
            if (sx * x.trit[i] > y->trit[i]) return 1;
 }
          return -1;
}
```

Функция *cmp*(*x*, *k*) производит сравнение необработанной части структуры *x* с индексами с *k* по нуль (рассматриваемой как число с точкой перед *k*-ым тритом) с числом .25. Поэтому сравниваются последовательно пары тритов структуры x и пара $1\ \bar{1}$.

```
int cmp(struct trin x, int k)
{
          int i = k;
 while ((i > 0) && (x.trit[i] == 1) && (x.trit[i - 1] == -1))
    i -= 2;
 if ((i > 0)  &&  (x.trit[i] == 1) &&  (x.trit[i - 1] > -1))
    return 1;
 else return -1;
}
```

Функция *sum*3(*x*, *y*, &*res*) производит вычисление по формуле *res* = *x* – 1 – (*y* + *y*)*sign*(*x*). Практически это одновременное сложение трех слагаемых, заметим, что в этом случае, также как и при обычном сложении, перенос возможен только одной единицы (положительной или отрицательной). Максимальное значение суммы тритов будет при совпадении знаков суммируемых тритов и трита переноса и равно ±4, и это дает перенос в следующий трит единицы того же знака (1 или –1). Отметим, что при обычном сложении трех слагаемых начальное значение этого трита (переменная *c*1) равно нулю, в данном случае в связи с формулой вычисления *res* начальное значение трита равно –1.

Заметим, что трехвходовые сумматоры использовались в ЭВМ «Сетунь» [6] в множительном устройстве. Для умножения 18-тритных чисел на первом уровне использовалось 6 трехвходовых сумматоров, на втором – 2 трехвходовых сумматора для суммирования результатов первого уровня, а на третьем уровне – один обычный (двухвходовый) сумматор для суммирования результатов второго уровня. Это позволило для последовательности выполнения операций выполнять умножения за 320 машинных тактов, притом, что сложение выполнялось за 180 тактов.

```
void sum3 (struct trin x, struct trin * y, struct trin * res)
{
          int i; s; sx = sgn(x); c1 = -1; j = max(x.n; y-> n);
          res-> n = j ;
          for (i = 0; i < j; i ++)
 {
          s = c1;
          if (i < x.n) { s += x.trit[i]; }
          if (i < y-> n) { s -= (y-> trit[i] + y-> trit[i]) * sx; }
          if (s < -1) { res-> trit[i] = s + 3; c1 = -1; }
          else
```

```
            if (s > 1) { res-> trit[i] = s - 3; c1 = 1; }
            else {  res-> trit[i] = s; c1 = 0; }
    }
   if (c1 != 0)  { res-> trit[j] = c1; res-> n = j + 1; }
   else red(res);
}
```

Алгоритм извлечения квадратного корня

Функция *sqrt*(*x*, &*y*) производит вычисление квадратного корня числа *x*.

```
void sqrt (struct trin x, struct trin * y)
   int i = x.n; j, sbi, k, cmpbc;
   struct trin bi, cx = x;
   if (i == 0) y->n = 0; else  {
      if (x.trit[i - 1] < 0) printf ("Number less 0\n");
      else {
            if ((i&1) == 1) {
        // первая пара состоит из одного трита
          cx.n++; cx.trit[i] = 0;
            }
     }
   }
   j = cx.n >> 1; // j – индекс старшего трита корня
   if (cmp(cx, j + j - 1) == 1) // cx больше .25
     {bi.n = 1; bi.trit[0] = -1; y-> n = 1; y-> trit[0] = 1;} // bi=-1, ai=1
   else { bi.n = 0; y-> n = 0; } // bi=0 и ai=0
   for (j--; j >= 0; j--)
  { // определение j-го трита корня
    shiftb(&bi, x, i); shiftc(y);
    sbi = sgn(bi); cmpba = acmp(bi, y);
    if ((cmpba == 1) || (cmpba == 0) && (cmp1(cx, j + j - 1) == sbi))
    { // j –й трит корня не нуль
      sum3(bi, y, &bi);
      if (y-> n == 0) { y->n = 1; }
      y-> trit[0] = sbi;
      } // if
    } // for
  } // else
} // конец функции
```

Литература

1. Рамиль Альварес Х. Алгоритмы деления и извлечения квадратного корня в троичной симметричной системе // Вестн. Моск. Ун-та. Серия 15. Вычислительная математика и кибернетика. № 2, 2008. С. 42–45.
2. Рамиль Альварес Х. Троичный алгоритм извлечения квадратного корня // Программные системы и инструменты. №. 11. – М.: МАКС Пресс, Москва, 2010. С. 98–106.
3. Рамиль Альварес Х. Извлечение квадратного корня в троичной симметричной системе // Рамиль Альварес Х. Алгоритмы троичной арифметики. М: Фонд «Новое тысячелетие», 2012. С. 14-19.
4. Березин Б.И., Березин С.Б. Начальный курс С и С++. М: ДИАЛОГ МИФИ, 2005.
5. Balanced ternary [Электронный ресурс] Электрон. дан. – URL: https://en.wikipedia.org/wiki/Balanced_ternary (дата обращения 12.10.2015)
6. Брусенцов Н.П., Маслов С.П. и др. Малая цифровая вычислительная машина «Сетунь». М.: Изд-во МГУ, 1965.

Рамиль Альварес Х.

Московский государственный университет имени М.В. Ломоносова, факультет ВМК, г. Москва, вед. научн. сотр., к.ф.-м.н., ramil@cs.msu.su

АЛГОРИТМ ИЗВЛЕЧЕНИЯ КВАДРАТНОГО КОРНЯ ИЗ БОЛЬШИХ ЦЕЛЫХ ЧИСЕЛ

КЛЮЧЕВЫЕ СЛОВА

Большие числа извлечение квадратного корня деление с остатком.

АННОТАЦИЯ

В статье дано более простое обоснование алгоритма, чем описанное в работе [1] и предложена его модификация, позволяющая за счет использования операции деления с остатком существенно уменьшить число операций с большими числами. Описан простой способ определения очередной цифры при вычислении квадратного корня " в столбик"[2].

В работе [1] предложен алгоритм точного вычисления целого значения квадратного корня из больших чисел. При этом достигнуто уменьшение числа операций умножения и деления больших чисел. Подкоренное число разбивается справа налево на пары цифр с добавлением слева нуля к старшей паре, если она состоит из одной цифры. Идея алгоритма состоит в том, что на первом шаге вычисляется n_1 старших цифр корня. Начальное число цифр выбирается таким образом, чтобы имеющие средства (калькуляторы, компьютерные программы) позволяли вычислить точное целое значение корня из числа, образованного n_1 старшими парами цифр N.

На каждом последующем итерационном шаге при определенных n_i цифрах корня вычисляется группа следующих , показано, что эта группа состоит из n_{i-1} цифр. На последнем шаге может возникнуть ситуация когда n_i-1 будет больше чем n-n_i. По этому введем k_i - число определяемых на очередном шаге цифр равное min(n_i-1,n-n_i).

Используемые обозначения

N - исходное целое число, состоящее из n пар цифр,

$\widehat{N}_i$ - значение N рассматриваемое как вещественное число с запятой после n_i - ой пары старших цифр,

$\widehat{A}_i$ - вещественное значение корня из N_i,

N_i - число образованное n_i старшими парами числа N,

A_i - целое значение корня из числа N_i, то есть целая часть числа $\widetilde{A}_i$ (число с n_i цифрами), $A_i = \lfloor \widehat{A}_i \rfloor$,

x_i - дробная часть числа A_i ($x_i = \widehat{A}_i - A_i$),

R_i - остаток корня N_i, то есть $R_i = N_i - A_i^2$,

B_{i+1} - целое число образованное k_i цифрами N следующими за числом N_i,

C_{i+1} - целое число образованное k_i цифрами N следующими за числом B_{i+1},

D_{i+1} - целое число, определяющее k_i (min (n_i-1,n-n_i)) следующих верных цифр корня.

Утверждение

Если вы численных n_i старших верных цифр целого корня из N ,то на следующем шаге n_i-1 цифр после запятой вещественного числа $\dfrac{\widehat{N}_i - A_i^2}{2A_i}$ определяют следующие верные цифры корня.

Доказательство

Число $\widehat{N}_i$ можно представить в виде $\widehat{N}_i = (A_i + x_i)^2 = A_i^2 + 2 \bullet A_i \bullet x_i + x_i^2$.

Откуда x_i равно $\dfrac{\widehat{N}_i - A_i^2}{2 \bullet A_i} - \dfrac{x_i^2}{2 \bullet A_i}$.

Приняв приближенное значение x_i равным $\frac{\widehat{N}_i - A_i^2}{2 \cdot A_i}$, допускаем погрешность $\frac{x_i^2}{2 \cdot A_i}$. Оценим эту погрешность. Так как $x_i<1$ и A_i состоит из n_i цифр ($A_i > 10^{n_i-1}$), то

$$\frac{x_i^2}{2 \cdot A_i} < ½ \cdot 10^{n_i-1}.$$

Таким образом доказано, что n_i-1 цифр после запятой дроби $\frac{\widehat{N}_i - A_i^2}{2 \cdot A_i}$ определяют следующие верные цифры корня.

Вычисление следующих верных цифр корня

Целое число D_{i+1}, определяющее k_i следующих верных цифр корня, равно

$$\lfloor \frac{\widehat{N}_i - A_i^2}{2 \cdot A_i} \cdot 10^{k_i} \rfloor,$$

$$D_{i+1} = \lfloor \frac{\widehat{N}_i - A_i^2}{2 \cdot A_i} \cdot 10^{k_i} \rfloor = \lfloor \frac{\lfloor \left(\widehat{N}_ι - A_ι^2\right) \cdot 10^{κ_ι} \rfloor}{2 \cdot A_ι} \rfloor = \lfloor \frac{\lfloor \widehat{N}_ι \cdot 10^{κ_ι} \rfloor - A_ι^2 \cdot 10^{κ_ι}}{2 \cdot A_ι} \rfloor.$$

Заметим, что $\lfloor \widehat{N}_i \cdot 10^{k_i} \rfloor$ равно $N_i \cdot 10^{k_i} + B_{i+1}$, поэтому

$$D_{i+1} = \lfloor \frac{N_i \cdot 10^{k_i} + B_{i+1} - A_i^2 \cdot 10^{k_i}}{2 \cdot A_i} \rfloor = \lfloor \frac{\left(N_i - A_i^2\right) \cdot 10^{k_i} + B_{i+1}}{2 \cdot A_i} \rfloor = \lfloor \frac{R_i \cdot 10^{k_i} + B_{i+1}}{2 \cdot A_i} \rfloor.$$

При делении с остатком $R_i \cdot 10^{k_i} + B_{i+1}$ на $2 \cdot A_i$ получаем неполное частное D_{i+1} и остаток Q_{i+1}

$$(R_i \cdot 10^{k_i} + B_{i+1})/(2 \cdot A_i) = D_{i+1} \text{ост } (Q_{i+1}), \qquad Q_{i+1} = R_i \cdot 10^{k_i} + B_{i+1} - 2 \cdot A_i \cdot D_{i+1}.$$

Определение остатка корня N_i+1

$$R_{i+1} = N_{i+1} - A_{i+1}^2 = N_i \cdot 10^{2 \cdot k_i} + B_{i+1} \cdot 10^{k_i} + C_{i+1} - (A_i \cdot 10^{k_i} + D_{i+1})^2 =$$
$$= N_i \cdot 10^{2 \cdot k_i} + B_{i+1} \cdot 10^{k_i} + C_{i+1} - A_i^2 \cdot 10^{2 \cdot k_i} - 2 \cdot A_i \cdot D_{i+1} \cdot 10^{k_i} + D_{i+1}^2 - =$$
$$= ((N_i - A_i^2) \cdot 10^{k_i} + B_{i+1} - 2 \cdot A_i \cdot D_{i+1}) \cdot 10^{k_i} + C_{i+1} - D_{i+1}^2 =$$
$$= (R_i \cdot 10^{k_i} + B_{i+1} - 2 \cdot A_i \cdot D_{i+1}) \cdot 10^{k_i} + C_{i+1} - D_{i+1}^2$$

Заменив в последнем выражении скобку на Q_{i+1}, имеем

$$R_{i+1} = Q_{i+1} \cdot 10^{k_i} + C_{i+1} - D_{i+1}^2$$

Формулы алгоритма

Таким образом, получаем формулы

$$k_i = \min (n_i\text{-}1, n\text{-}n_i)$$

$$(R_i \cdot 10^{k_i} + B_{i+1})/(2A_i) = D_{i+1} \text{ост } (Q_{i+1}), \qquad A_{i+1} = A_i \cdot 10^{k_i} + D_{i+1}$$

$$R_{i+1} = Q_{i+1} \cdot 10^{k_i} + C_{i+1} - D_{i+1}^2, \qquad n_{i+1} = n_i + k_i.$$

Заметим, что если значение k_i на каждом итерационном шаге принять равным 1, получатся формулы модифицированной схемы вычисления квадратного корня "в столбик" [2]

$$(R_i \cdot 10 + b_{i+1})/(2 \cdot A_i) = d_{i+1} \text{ ост } (Q_{i+1})$$

$$A_{i+1} = A_i \bullet 10 + d_{i+1}, R_{i+1} = Q_{i+1} \bullet 10 + c_{i+1} - d_{i+1}^2$$

где d_{i+1}- очередная цифра корня b_{i+1} и c_{i+1}- цифры, следующие после числа N_i и цифры b_{i+1}, соответственно.

Пример вычисления корня

Вычислим целое значение корня из числа, состоящего из 24 цифр (n=12).

98 76 54 32'10 98 76'54 32 0 98 76

Апострофом (') выделены используемые на каждом шаге группы цифр .

Для определения начального приближения корня возьмем корень из числа, состоящего из 4 старших пар (98 76 54 32). Для нахождения корня из этого числа воспользуемся модифицированной схемой вычисления цифра за цифрой.

Начальное значение корня из числа 98 равно 9 (A_1), а остаток корня -17 (R_1). Для нахождения второй цифры d_2 надо разделить число $R_1 10+b_2$ (177) на $2A_1$ (18)

117/18 = 9 ост (15), то есть $d_2 = 9$ и $Q_2 = 15$.

Остаток корня из числа 9876 (R_2) равен разности $Q_2 10+c_2$ и квадрата d_2

156-81 = 75, то есть $A_2 = 99$ и $R_2 = 75$.

Повторяя эти вычисления, будут получены цифры d_3 и d_4 и остатки R_3 и R_4. Результаты вычислений представлены в Таблице 1.

Таблица 1

i	A_i	R_i	b_{i+1}	d_{i+1}	Q_{i+1}	c_{i+1}
1	9	17	7	9	15	6
2	99	75	5	3	161	4
3	993	1605	3	8	165	2
4	938	1588				

Таким образом, начальное значение корня A_1 из числа 98 76 54 32'10 98 76'54 32 0 98 76 равно 9 938, разность корня R_1 -1 588, n_1 равно 4 и k_1=3, то есть можно одновременно вычислить группу из трех цифр.

К остатку корня R_1 приписываются 3 следующие цифры числа B_2 (109). Полученное число 1 588 109 делится с остатком на удвоенный корень A_1 (19 876).

1 588 109/19 876 = 079 ост (17 905)

то есть $D_2 = 079$ и $Q_2 = 17\ 905$.

К остатку от деления Q_2 приписываются 3 следующие цифры числа C_2 (876), из полученного числа вычитается квадрат неполного частного D_2 (6 241).

17 905 876-6 241=17 899 635

Таким образом, после первого итерационного шага вычислены 7 старших цифр корня - A_2 (9 938 079) и остаток корня R_2 (17 899 635).

На втором итерационном шаге будут вычислены еще 5 цифр корня, так как $\min(n_i\text{-}1, n\text{-}n_i)$ равен 5. Результаты вычислений представлены в Таблице 2.

Таблица 2

i	n_i	A_i	R_i	k_i	B_{i+1}	D_{i+1}	Q_{i+1}	C_{i+1}
1	4	9 938	1 588	3	109	079	17 905	876
2	7	9 938 079	17 899 635	5	54 321	90 055	16 145 631	09 876
3	12	9938 079 90055	1 606 453 206 851					

Таким образом, целое значение корня из числа 987 654 321 098 765 432 109 876 равно 993 807 990 055, а остаток - 1 606 453 206 851.

Литература

1. 1. Колесов А. Н. Нахождение точного целочисленного корня квадратного Доклады ТУСУРа – Томск: Томский государственный университет систем управления и радиоэлектроники 2009. №2(20) С. 162-168.
2. 2. Соловьев Ю. П. Старый алгоритм // Квант – М. : Наука 1987. № 3 С. \34-37.

Фёдоров И.Г.

Российский Экономический Университет им. Г.И. Плеханова, кт.н., доцент кафедры экономических информационных систем и информационной безопасности, IgorFiodorov@mail.ru

СИНТАКСИС И СЕМАНТИКА ИСПОЛНЯЕМЫХ МОДЕЛЕЙ БИЗНЕС-ПРОЦЕССОВ

КЛЮЧЕВЫЕ СЛОВА

Синтаксис, семантика и прагматика Онтология Бунге-Ванда-Вебера, нотации моделирования бизнес-процессов, онтологическая выразительность языков моделирования бизнес-процессов..

АННОТАЦИЯ

Большую популярность приобрели языки и нотации визуального моделирования бизнес-процессов, в том числе BPMN и UML. Авторы нотаций и вендоры, создающие средства разработки, озабоченные вопросом стандартизации, активно работают над тем, чтобы дать научное обоснование синтаксиса, семантики и прагматики моделей, без которых аналитики не смогут их правильно применять. Для исполняемых моделей остро стоит вопрос формализации поведения, реализуемого моделью. В предлагаемой работе синтаксис, семантика и прагматика языков визуального моделирования бизнес-процессов объясняются с использованием онтологии Бунге-Ванда-Вебера.

Введение

Сегодня руководства по языкам UML и BPMN связывают стандартизацию синтаксиса, семантики и прагматики модели с метамоделью, которая описывает требуемые свойства при помощи языка обмена метаданными XMI [1], хотя известно, что метамодель не является ни синтаксисом ни семантикой языка моделирования [1]. Практики BPMN предлагают эмпирические правила стиля и метода моделирования [3], при этом остается не ясно, что они описывают – синтаксис или семантику, могут ли эти рекомендации быть теоретически обоснованы? Можно добавить, что до сих пор не решен вопрос, какой математический формализм следует использовать для описания поведения исполняемых моделей бизнес-процессов, так что часть специалистов считают, что в основе исполняемых моделей лежит формализм машины состояний, тогда как другие уверены, что в основе моделей лежит формализм сетей Петри [5]. Таким образом, вопрос теоретического обоснования синтаксиса, семантики и прагматики визуальных исполняемых моделей является актуальным. Целью данной работы является теоретическое обоснование синтаксиса, семантики и прагматики визуальных исполняемых моделей.

Синтаксис и семантика в исследованиях Д. Харела и Б. Румпе

По мнению Д. Харела и Б, Румпе синтаксис определяет правила, связывающие отельные знаки, принадлежащие к алфавиту языка, в лексически правильные конструкции и выражения, тогда как семантика определяет смысл высказывания [7], [2]. Они различают текстовые и иконические языки. Алфавит первых состоит из букв, которые соединяются в слова, передающие смысл, тогда как в иконических языках каждый знак обозначает отдельное понятие, так что синтаксис определяет правила, связывающие знаки в законченную, правильно структурированную модель. Семантика, по определению Д. Харел и Б. Румпе, включает семантический домен и семантическое отображение (см. рис.1). Семантический домен перечисляет все концепты, которые лежат в основе формализма представления модели бизнес процесса. Авторы делают очень интересное замечание о том, что семантический домен, образованный концептами, может быть использован для сравнения языков и нотаций моделирования, но дальше эту мысль не развивают. К сожалению, они не объясняют, как можно описать семантический домен, поэтому его практическое применение остается неясно. Семантическое отображение связывает домен с соответствующей предметной областью моделирования, которую иначе называют универсум дискурса. Правила построения семантического отображения авторы не приводят. Они отмечают, что большинство компьютерных средств разработки приложений позволяют проверять синтаксис соответствующего языка, тогда как смысл модели проверить автоматически невозможно.

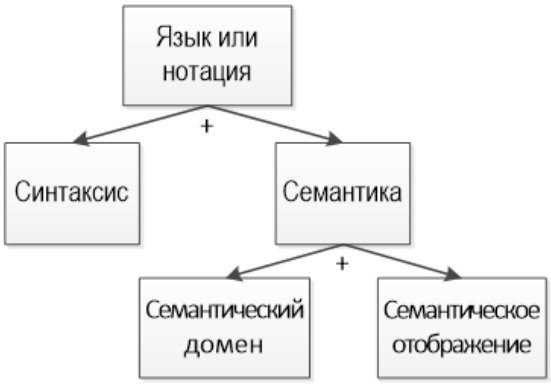


Рисунок 1. Синтаксис и семантика языка моделирования в исследованиях Д. Харела и Б. Румпе

Харел и Б. Румпе рекомендуют различать семантику, метамодель и поведение. Они считают, что семантику языка нельзя отождествлять с:

1)метамоделью, поскольку последняя является средством описания синтаксиса;

2)семантическим доменом, поскольку семантика описывает отображение на этот домен, но не совпадает с доменом;

3)контекстным условием, поскольку в этом утверждении не упоминаются ни семантический домен, ни семантическое отображение;

4)поведением исследуемого объекта, поскольку некоторые модели, например структурные, имеют семантику, но не описывают поведение;

5)исполняемостью, поскольку не все языки позволяют создавать исполняемые модели;

6)«математической нотацией», поскольку последняя, по их мнению, не имеет никакого отношения к семантике;

7)пониманием индивидуальных конструкций, поскольку семантика отдельной модели не есть семантика языка.

Подводя итог можно отметить – модель предлагаемая авторами является абсолютно правильной, однако она не дает четкого формализованного объяснения понятиям семантический домен и семантическое отображение, что ограничивает её практическое применение.

Семиотический подход к определению семантики языка

Модель бизнес-процессов можно рассматривать как средство коммуникации между аналитиком, который наблюдает окружающий мир, описывает его и кодирует свои наблюдения в виде модели, используя некоторый язык моделирования, и интерпретантом, который декодирует полученную модель и пытается понять, что хотел передать ему аналитик. Если интерпретантом является человек, то он может, переспросить непонятные ему места, понять из контекста, додумать и, таким образом, восстановить смысл модели. Если же интерпретант машина, то модель должна быть такой, чтобы исключить возможность неоднозначного толкования. Сегодня появились исполняемые модели бизнес-процессов, которые позволяют превратить модель в исполняемый машинный код без дополнительного программирования. Семантика таких моделей должна быть ясной и понятной как аналитику так и интерпретанту, чтобы исключить любую двусмысленность.

Рассмотрим треугольник Г. Фреге (см рис.2), который иллюстрирует принцип восприятия человеком окружающей реальности и связывает объект реального мира, соответствующий знак языка и. концепт, абстрагирующий понятие связанное со знаком Объект - есть предмет, вещь иначе денотант предметной области – то что мы собираемся моделировать. Совокупность всех денотантов образует предметную область моделирования, иначе называемую универсум дискурса. Концепт, иначе десигнант, это некоторое представление, которое связанно с моделируемой вещью. Совокупность всех десигнантов образует семантический домен. Знак, иначе экспонент, это логическое имя, присвоенное соответствующему концепту. Совокупность всех допустимых знаков образует алфавит языка. Таким образом, знак языка моделирования отображает сущность реального мира в том случае, если он способны отобразить концепт соответствующей предметной области, который, в свою очередь, абстрагирует некую сущность реального мира [7].

Теперь мы можем утверждать, что смысловое значение модели передается через систему понятий, называемых концептами. Поэтому нам необходимо восстановить смысл каждого из них.

Семантический домен натурального языка содержит счетное, и, по-видимому, бесконечное число концептов. Учитывая их синонимию и омонимию, построить семантический домен и отобразить все слова на соответствующие понятия затруднительно. Однако, как установили Я. Ванд и Р. Вебер, для графических языков визуального моделирования число концептов конечно.

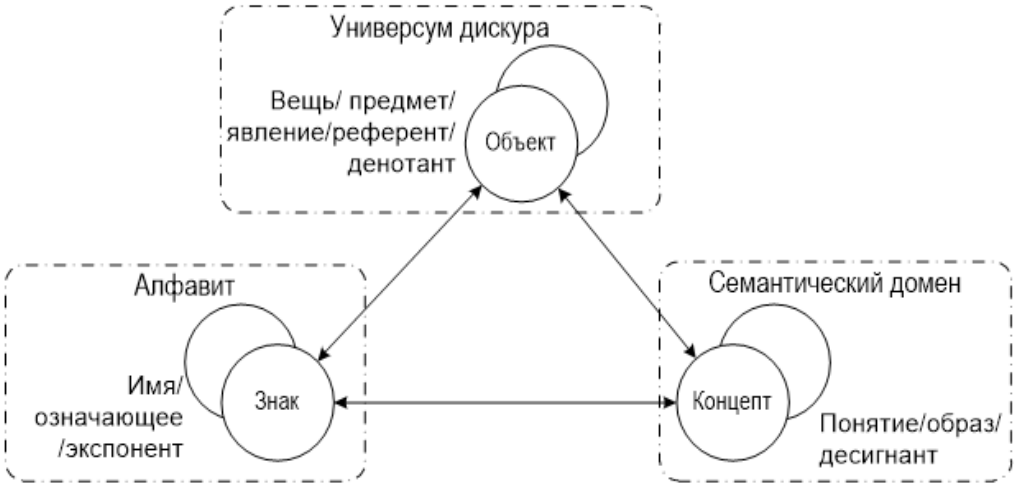


Рисунок 2. Отображение объектов реального мира

Модель представления Бунге-Ванда-Вебера

Модель представлений Бунге-Ванда-Вебера основывается на философских работах М. Бунге, который является приверженцем диалектического материализма. В основе его рассмотрения находится материальный мир [30], который образован вещами, последние принято трактовать как «отдельный объект материального мира, обладающий относительной независимостью, объективностью и устойчивостью существования» [195], поэтому далее мы будем пользоваться термином объект. Здесь и далее будем использовать термин объект, как синоним понятия вещь.

Свойством называют один из атрибутов объекта, оно не может иметь свойств. Свойство обусловливает различие или общность объекта с другими предметами, оно может быть частными, присущими отдельному объекту, например, габариты и вес характеризуют каждый предмет в отдельности, обобщающими, характеризующими совокупность объектов, например, должность характеризует группу сотрудников или общими, присущими сразу нескольким объектам, например, заказ и договор имеют общие атрибуты: клиент, адрес доставки, номенклатура заказа и т.д. [28].

Состояние объекта определяется как совокупность значений всех его атрибутов в определенный момент времени. При этом, не все состояния рассматриваются как допустимые и не все переходы между состояниями считаются разрешенными. Изменение состояния объекта в результате его взаимодействия с другими объектами называют трансформацией, она осуществляется всегда по заранее определенному закону, называемому правилом преобразования. Трансформацию можно интерпретировать как работу изменяющую объект или как операцию, выполняемую над объектом.

Рисунок 3. Основные концепты модели процесса

Событием Я. Ванда и Р. Вебера называют факт изменения состояния объекта, независимо

от причины возникновения. Событие предполагаются дискретными, его длительностью принято пренебрегать. Можно говорить, что объект переходит из одного разрешенного состояния в другое разрешенное состояние в результате срабатывания события, которое инициирует трансформацию этого объекта, выполняемую по определенному правилу (см. рис.3). Процессом они называют упорядоченную во времени последовательность смены состояний одного объекта.

Один объект может взаимодействовать с другим объектом, назовем его объектом окружения. Объект наблюдения может узнать об изменение состояния объекта окружения через оповещения. Механизм этих оповещений онтологией не определяется. Изменение состояния объекта окружения, о котором система узнает через оповещение, называется внешним событием. Говорят, что два объекта взаимодействуют, если история состояний одного из них зависит от присутствия другого объекта. Рисунок 4 показывает взаимодействие 2-х объектов. Изменение состояние объекта 1 может рассматривать как внешнее событие для объекта 2, которое инициирует трансформацию этого объекта.

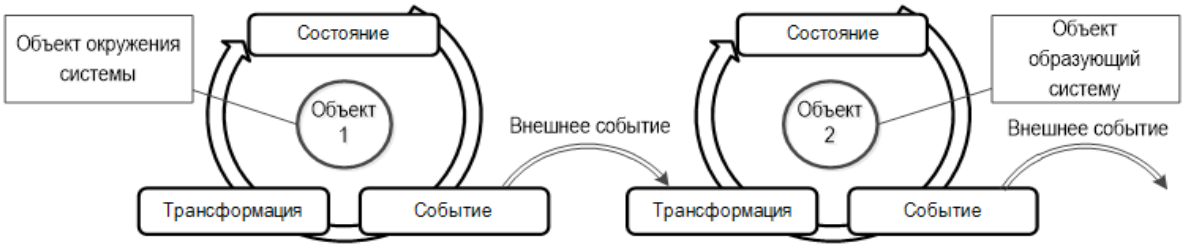


Рисунок 4. Взаимодействие 2-х объектов

Адаптация онтологии Бунге-Ванда-Вебера к описанию исполняемых моделей бизнес-процессов

Можно указать на следующие особенности онтологии Бунге-Ванда-Вебера. Во-первых, онтология не использует категорию время, хотя очевидно, что временные параметры исполнения процесса очень важны при моделировании. Во-вторых, онтология не включает концептов, отображающих участников процесса. Это, на первый взгляд, кажется странным и противоречит устоявшейся практике начинать моделирование с выявления участников процесса [19]. В третьих, остается непонятным, как классифицировать логические операторы, которые маршрутизируют объект, но не трансформирую. Адаптируем онтологию таким образом, чтобы её можно было использовать для моделирования бизнес-процессов.

Время в онтологии Бунге-Ванда-Вебера

М. Бунге определил событие как факт смены состояния объекта [8]. При этом возникают вопрос: чем событие отличается от состояния? В существующей трактовке событие объясняется как смена состояния, несет смысл «по причине этого» и отражает причинно-следственную связь. Как результат, термины состояние и событие оказываются трудноразличимы. Предложим следующую интерпретацию. Определим внутреннее событие как момент времени, когда произошла смена состояния, добавив ему смысл «после этого» – позже в хронологическом порядке. Таким образом, внутреннее событие фиксирует факт и момент времени, когда, система перешла в очередное состояние и готова к исполнению следующей операции. Однако наступления внутреннего события недостаточно для начала исполнения очередной трансформации. Если это интерактивная операция, то исполнение начнется после вмешательства оператора, последний рассматривается как внешний объект по отношению к системе, а если операция автоматическая, то после сигнала от внешнего управляющего устройства. Внешнее событие отражает факт изменения состояния объекта окружения, которое инициирует исполнение операции, оно фиксирует момент времени, когда начнется трансформация. Таким образом, промежуток времени между внутренним событием, означающим готовность к началу обработки и внешним событием, инициирующим трансформацию, определяет длительность ожидания начала исполнения операции, а интервал времени между внешним событием и внутренним событием означающим завершение трансформации, означает длительность выполнения операции. Рисунок 5 иллюстрирует соотношение между внутренним и внешним событием на диаграмме Ганта.

Тем самым в онтологию добавляется понятия темпоральной логики: момент времени и интервал времени между двумя последовательными событиями. Такая точка зрения хорошо соответствует современным представлениям об онтологии времени, используемой для описания темпоральных взаимоотношений между работами, составляющими процесс, которая использует

два базовых понятия: событие и интервал [20]. Под событием понимается точка на шкале времени, а интервалом называется отрезок на шкале времени, заключенный между начальным и конечным событиями.

Рисунок 5 Внутренние и внешние события на диаграмме Ганта

Трансформация и маршрутизация в онтологии Бунге-Ванда-Вебера

«Классическая» онтология Бунге-Ванда-Вебера предполагает, что трансформация изменяет свойства объекта. Однако существуют операции процесса, которые объект не трансформируют, а маршрутизируют, они не могут быть отображены в концепты онтологии. Например, логические операторы ветвления и слияния, объект не изменяют, но переключают возможные сценарии исполнения. Мы будем рассматривать трансформацию не только как преобразование, но и как маршрутизацию. Таким образом, логический оператор на модели процесса отображается на трансформацию в онтологии Бунге-Ванда-Вебера.

Семантика визуальных языков моделирования

Проведенный анализ позволил выделить шесть концептов онтологической модели бизнес-процесса, образующих семантический домен. К их числу относятся: объект, который имеет набор свойств, его состояние, которое характеризуется значением свойств в определенный момент времени, трансформации, которые изменяют свойства объекта, что приводит к изменению его состояния и работы, которые свойств объекта не изменяют, но маршрутизируют по нужным направлениям, события, (внутренние и внешние), которые инициируют трансформацию. Теперь необходимо определить семантическое отображение множества знаков языка моделирования на семантический домен.

Семантическое отображение

Я. Ванд и Р. Вебер предположили, что главным фактором успеха использования языка является его способность предоставить пользователям набор знаков, которые могут непосредственно выражать соответствующие концепты предметной области [190]. Они выделили следующие варианты соответствия между множествами знаков языка моделирования и концептов модели процесса (см. рис.6):

1. Каждому знаку языка можно сопоставить ровно один концепт онтологии – можно говорить о взимнооднозначном соответствии двух множеств или об их эквивалентности;
2. Один концепт может быть отображен сразу на несколько знаков нотации – имеет место омонимия или неразличимость соответствующих знаков языка моделирования;
3. Отдельные концепты онтологии не имеют соответствующего им знака – существует дефицит выразительной способности соответствующего языка, он не способен отобразить отдельные аспекты окружающей реальности;
4. Ни один концепт не может быть отображен в примитив нотации – имеется избыточность языка, он содержит знаки, которые не соответствуют объектам реальности;
5. Одному знаку соответствует несколько концептов – в наличии синонимия знаков или неоднозначность нотации.

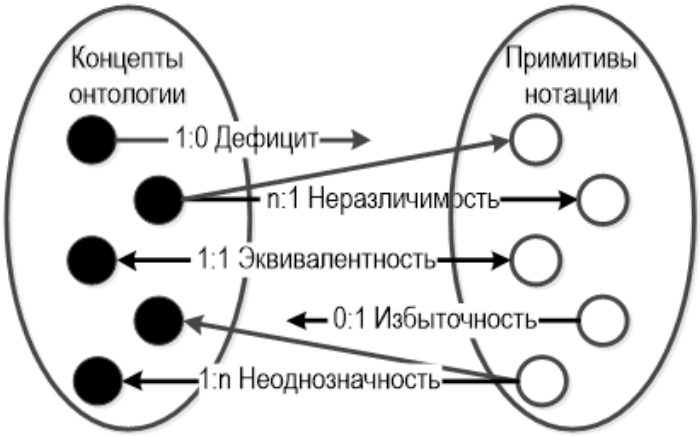


Рисунок 6. Соотношение примитивов языка моделирования и концептов модели бизнес-процесса

Конкретный и абстрактный синтаксис языка моделирования

Назовем идеальным язык моделирования, для которого между множеством знаков его алфавита и соответствующим семантическим доменом существует взаимно однозначное соответствие или биекция. Конкретный синтаксис языка описывает правила соединения знаков языка в лексически правильные выражения. Для существующих языков визуального моделирования правила синтаксиса постулируются, но не доказываются. Мы постараемся найти объяснение правилам синтаксиса.

Давайте внимательно рассмотрим фрагмент треугольника Фреге, изображенный на рис.7. Мы установили, что каждому знаку нотации соответствует ровно один концепт онтологии. Обратим внимание, что отношение знак-знак образует конкретный синтаксис данной нотации. Поскольку отображение занак-концепт является гомоморфным, мы можем утверждать, что конкретный синтаксис определяется связями концептов друг с другом. Будем называть связи между концептами абстрактным синтаксисом. Постараемся теоретически обосновать полученные правила синтаксиса.

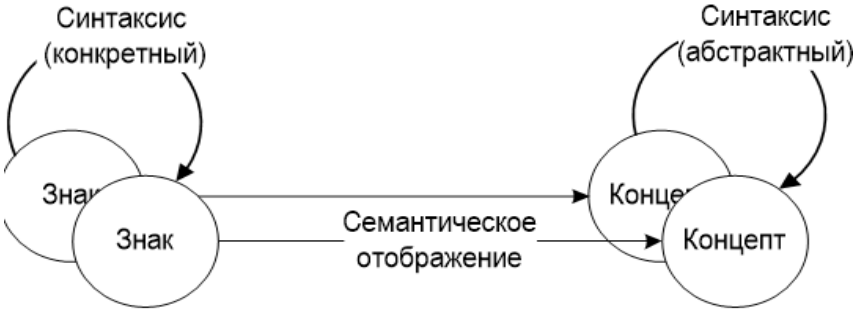


Рисунок 7 Абстрактный синтаксис языков визуального моделирования

Чтобы обосновать связи между концептами модели бизнес-процесса, рассмотрим, какие формальные модели позволяют описать выделенные нами концепты. Поскольку визуальные языки моделирования бизнес-процессов являются графовыми, будем искать формализмы среди графов. Мы рассмотрим пять типов формализмов, которые в совокупности описывают взаимосвязи между концептами семантического домена. Будем рассматривать только базовые формализмы, которые не имеют расширений. При этом мы будем обращать внимание, что соответствующая диаграмма может моделировать, а что она изображает для ссылки, связывающей её с другой формальной моделью. Поясним на примере, рассмотрим инфологическую модель, которая позволяет описать набор свойств некоторого объекта и, таким образом, установить, тот набор свойств, который соответствует каждому разрешенному состоянию этого объекта. Далее рассмотрим диаграмму состояний (STD), она моделирует переходы состояния объекта, кроме того она изображает (но не моделирует) работы (трансформации), в результате которых происходит изменение состояния. Используя имя работы как ссылку, мы сможем построить связь диаграммы состояний с диаграммой потоков данных (DFD), которая моделирует работы, трансформирующие объект. Диаграмма потоков данных не

позволяет отобразить моменты времени, когда начинается или заканчивается соответствующая операция. Используя имя трансформации как ссылку, можно перейти к диаграмме Ганта, которая моделирует интервалы между соответствующими событиями. Поскольку диаграмма потоков данных моделирует только те трансформации, которые изменяет состояние объекта, она не позволяет отобразить логические операторов. Поэтому добавим в рассмотрение обычные сети Петри. Очень важно точно определить отображение концептуальной модели на соответствующий формализм. Для рассмотренных выше формализмов отображения были тривиальными, но для сетей Петри отображение имеет решающее значение, поскольку одна модель при разных отображениях дает различные результаты. В случае моделирования бизнес-процессов, позиции сети Петри соответствует состояние объекта, а её переход – трансформации или операции процесса. Обратим внимание, что ни позиция, ни операция не позволяют смоделировать состояние или трансформацию, т.е. они используются для ссылки на другие модели. Однако смоделировать работу логического оператора сети Петри позволяют. Теперь рассмотрим, что означает маркер? Его можно представить себе как движущийся информационный объект, его прибытие инициирует выполнение операции. Однако свойства объекта смоделировать в обычных сетях Петри нельзя. Таким образом, модель сетей Петри, при соответствующем отображении, позволяет смоделировать логические операторы бизнес-процесса. Рисунок 8 иллюстрирует связи между соответствующими формализмами (инфологическая модель опущена). Те параметры, которые не моделируются и используются для ссылки, изображены с подчеркиванием.

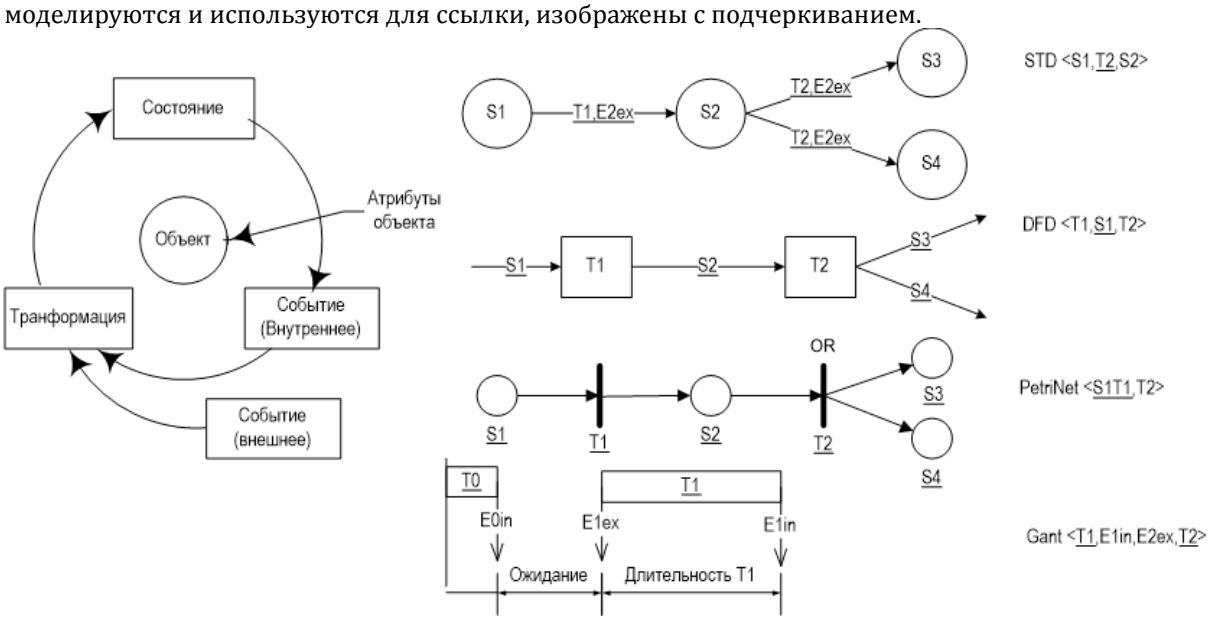


Рисунок 8 Формальные модели для описания концептов онтологии БВВ

Предлагаемый в работе подход, в целом, соответствует предложениям Э. Йордана, который в рамках структурного метода моделирования предлагал моделировать последовательно в нескольких моделях DFD, STD и ER [148]. Йордан не ставил целью проектировать системы реального времени, поэтому структурный метод опускает временной аспект, в нем не используется диаграмма Ганта, не моделируется бизнес логика, поэтому отсутствуют сети Петри. Поскольку мы рассматриваем наиболее общий случай, мы дополнили нашу модель таким образом, чтобы учесть все связи между концептами. Мы назовем отображение концептуальной онтологической модели на соответствующий формализм – дефиниционным.

Сделаем следующие выводы. Нами рассмотрен идеальный язык моделирования, в котором между множествами знаков алфавита языка и множеством концептов онтологической модели существует взаимно однозначное соответствие, таким образом, абстрактный синтаксис языка однозначно определяется отношениями между отдельными концептами. Нами выделено шесть концептов и построено шесть дефиниционных отображений, которые позволяют связать соответствующие концепты с формальными моделями. Каждый из рассмотренных формализмов позволяет определить некоторый набор правил. Совокупность всех правил всех формальных моделей образует конкретный синтаксис языка моделирования.

На практике ни один из известных языков моделирования не является идеальным. Используя онтологию как базис сравнения, была проверена выразительная способность ряда

популярных языков (нотаций) моделирования [11]. Исследование показало, что ни один из языков: STD, DFD, WFD, PERT, Petri Net, EPC, BPMN не способен без потерь передать все свойства окружающей реальности, только часть из них. А некоторые нотации даже не обеспечивают однозначного отображения отдельных знаков ровно на один концепт, имеет место неразличимость соответствующих знаков языка моделирования. Таким образом, можно сделать вывод о дефиците выразительности этих моделей.

Покажем, что дефицит изобразительной возможности преодолеваем, тогда как неразличимость недопустима. Введем понятие «хорошей» модели, в которой каждому знаку соответствует ровно один концепт, но не все концепты могут быть отображены на знаки этой нотации, т.е. множества знаков и концептов не являются равномощными. Можно утверждать, что в этом случае имеет место инъекция — такое отображение множества знаков в множество концептов, при котором каждому знаку соответствует свой концепт. При этом, конкретный синтаксис языка определяется не всеми дефинитивными отображениями, а только теми, для которых существует концепт.

Прагматика модели бизнес-процесса

В контексте языков моделирования бизнес-процессов прагматика понимается как практические способы достижения цели моделирования. Мы будем различать аналитические модели бизнес-процессов, предназначенные, чтобы понять, как работает организация и исполняемые модели, предназначенные для управления процессом достижения этого результата. Разные цели моделирования определяют различие требований, предъявляемые к моделям. Аналитические модели должны быть когнитивно понятны, тогда как исполняемые модели должны интерпретироваться соответствующим исполняющим устройством. Соответственно мы будем разделять два компонента прагматики модели бизнес-процесса. Экспрессивность есть характеристика, определяющая насколько модель является когнитивно понятной, так что она сможет визуально передать, заложенный в неё смыл. Этот аспект прагматики модели бизнес-процесса является объектом ряда исследований [8], мы не будем рассматривать его в данной работе. Второй аспект характеризует интерпретируемость – способность осуществить предписанную последовательность действий, заложенную в модель бизнес-процесса.

Поскольку исполняемую модель можно рассматривать как алгоритм, который, согласно определению А.П. Ершова, есть «точное предписание исполнителю осуществить последовательность действий, направленных на достижение указанной цели или на решение предлагаемой задачи», для оценки интерпретируемости модели бизнес-процесса применить следующие критерии эквивалентности [15]:

•Функциональная (трансформационная) – операции реализуемые процессами, имеющими одинаковые множества аргументов, совпадают;

•Операционно-логическая (структурная, трассовая) – в обоих процессах совпадают последовательности пройденных операций и логических операторов;

•Логико-термальная (поведенческая) – для любой трассы одного процесса существует трасса в другом, такая что логические условия проверяются в одной и той же последовательности для одних и тех же наборов значений переменных.

Если целью моделирования является реинжиниринг, то процессы «как есть» и «как должно быть» должны иметь функциональную эквивалентность, оба выполняют одинаковое преобразование, но по-разному. В тех случаях, когда стоит задача раскрыть и смоделировать неизвестный процесс (process mining), часто ставится задача достичь трассовой эквивалентности. Однако следует иметь в виду, что этот тип эквивалентности не гарантирует достижение требуемого для процесса результата. В случае создания исполняемой модели, необходимо добиться поведенческой эквивалентности, таким образом, чтобы модель процесса в точности повторяла реальную работу в организации.

Оба аспекта прагматики связаны с метамоделью, реализуемой на языке XMI. Первый аспект связан с тем, что первоначально преобразование модели процесса в формат XMI осуществлялось таким образом, что информация о взаимном расположении элементов терялась, так что воспроизводимость модели ухудшалась. Сегодня появляются такие средства преобразования в формат XMI, которые сохраняют информацию о взаимном расположении элементов.

Обобщенная модель синтаксиса, семантики и прагматики языка моделирования

Чтобы лучше понять соотношение между синтаксисом, семантикой и прагматикой модели процесса, мы рассмотрим квадрат Д.А. Поспелова [15], но внесем в него небольшое изменение –

вместо метамодели (вершина 4), которую ввел в рассмотрение Д.А. Поспелов, мы используем боле общее понятие дефиниция, раскрывающее содержание (смысл) знака посредством описания существенных отличительных признаков предметов или явлений, обозначаемых данным именем (см. рис.9).

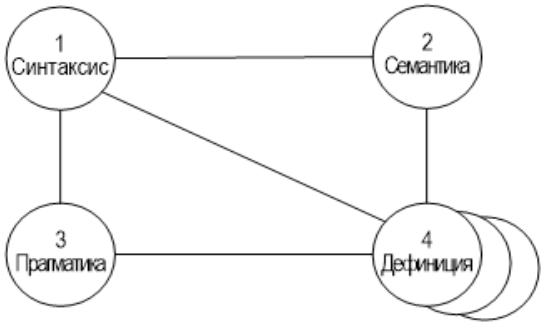


Рисунок 9. Квадрат Поспелова

Можно предположить, что дефиниция может быть выполнена разными способами. Во-первых, можно создать текстовое описание, которое определяет содержание понятия, смысловое значение его имени. Во-вторых, можно определить объект при помощи некоторой метамодели (как это изначально предлагал Д.А. Поспелов), в этом случае, мы делаем модель переносимой между разными средами моделирования и определяем одинаковое поведение модели в различных средах исполнения. В-третьих, можно выполнить формальную дефиницию с помощью математической модели.

Совместим квадрат Поспелова и треугольник Фреге. Заметим, что синтаксис есть отношения, существующие между знаками языка, а семантика есть понятие или смысл, заложенные в модель. Будем полагать, что семантика предполагает выбор определенного формализма, который определяет синтаксис, семантику и прагматику модели бизнес-процесса. Отметим, что у модели может быть несколько дефиниций с каждой связан свой формализм. Таким образом, модель распадается на набор субмоделей, каждая описывает взгляд на прототип с определенной точки зрения. Для нескольких частных моделей может существовать одна общая для них всех метамодель, так что связь между частным формализмом и общей метамоделью будет односторонней или гомоморфной (см. рис.10). Формализм каждой из частных моделей определяет её синтаксис и поведение (если оно данным формализмом описывается).

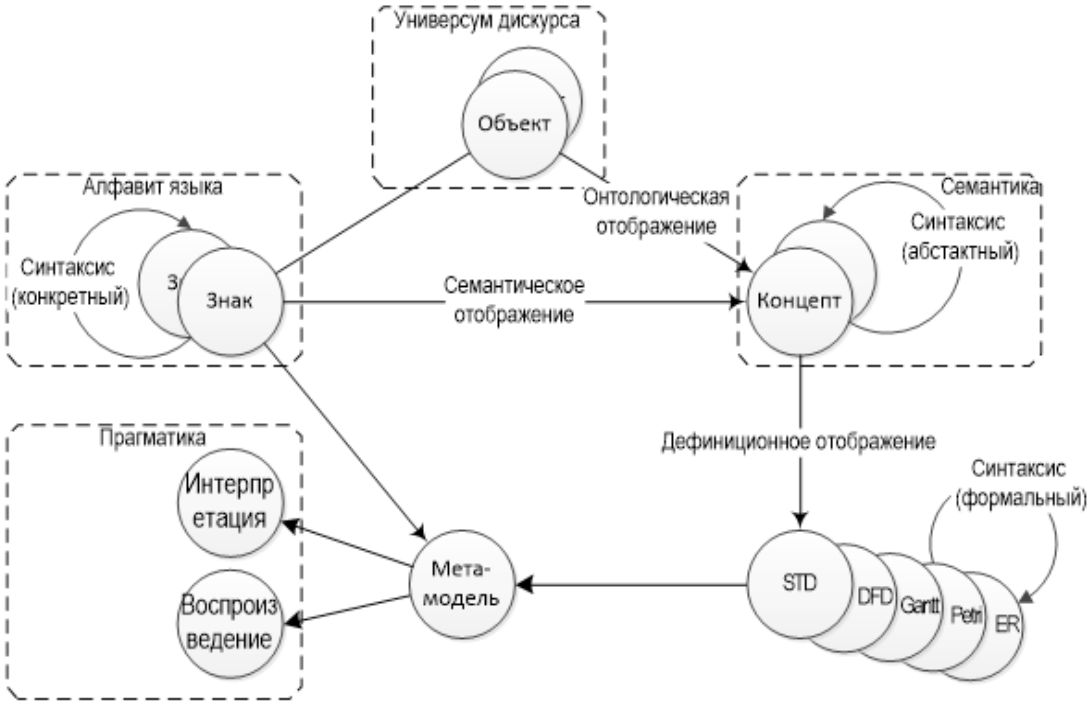


Рисунок 10. Треугольник Фреге, совмещенный с квадратом Поспелова

Выводы

Проведенный анализ позволяет сформулировать теоретические основы моделирования бизнес-процесса. В качестве фундамента для анализа языков и нотаций моделирования выбрана онтология Бунге-Ванда-Вебера. Главный результат заключается в выделении концептов, образующих онтологию модели процесса, что позволяет установить основные понятия предметной области моделирования, определить аксиоматику моделирования. Важный вывод, который можно сделать на основе рассмотрения — процесс есть последовательность смены состояний объекта, происходящая в результате выполнения операций процесса.

Новизна исследования заключается в уточнении различия между внутренними и внешними событиями. Внутренне событие отражает факт готовности объекта к очередной трансформации. Внешнее событие отображает момент времени, когда трансформация будет начата или остановлена. В онтологию добавлена категория времени – ранее событие толковалось как смена состояния, таким образом, оно несло смысл «по причине этого», отражающее причинно-следственную связь. Как следствие термины состояние и событие оказывались трудноразличимы. Мы определили событие как момент времени, когда произошла смена состояния, добавив ему смысл «после этого» — позже в хронологическом порядке. Тем самым в онтологию добавляется понятие времени: момента времени и интервала времени между двумя последовательными событиями. Теперь понятие событие трактуется в соответствии с представлениями темпоральной логики [23]. Такой подход позволяет объяснить, проанализировать и выявить ошибки в моделях бизнес-процессов, когда внешнее событие происходит ранее, чем связанное с ним внутреннее событие.

Мы установили, что в число концептов модели процесса не включен исполнитель операций процесса (актор), который генерирует внешние события, инициирующие / завершающие исполнение трансформаций. Если операция является интерактивной, то именно актор генерирует внешнее событие, которое инициирует старт её обработки. Из этого следует, что модель процесса должна быть отображена на множество сотрудников организации. Сделан важный вывод о том, что организационная модель предприятия не является имманентной частью модели процесса. Обе модели являются независимыми, причем исполнители на модели процесса должны быть отображены на организационную модель предприятия. Этим может быть достигнута инвариантность модели процесса изменениям организационно штатного расписания предприятия.

Предложено трактовать понятие трансформация из онтологии Бунге-Ванда-Вебера не только как работу, которая трансформирует состояние некоторого информационного объекта, но также как работу, которая информационный поток не изменяет, но маршрутизирует. Таким образом, логические операторы на модели процесса могут быть отображены в концепты онтологии Бунге-Ванда-Вебера.

Сравнение различных языков и нотаций моделирования, проведенное с использованием описанного семантического подхода, позволило установить, что ни одна из известных нотаций не в состоянии отобразить сразу все концепты онтологии Бунге-Ванда-Вебера, налицо дефицит онтологической выразительности [24]. Это дало основание сделать вывод, что моделирование процесса следует проводить одновременно в нескольких нотациях, так чтобы каждая частная модель показывала ограниченный набор свойств моделируемого явления, а все вместе они давали полное и исчерпывающее представление о моделируемой реальности.

Литература

1. Stevens P. // XMI and MOF: a mini-tutorial. 2001. URL: http://homepages.inf.ed.ac.uk/perdita/XMI/tutslides2up.pdf (дата обращения: 15.01.2015).
2. Harel D., Rumpe B., "Meaningful Modeling What's the Semantics of Semantics," Journal Computer, Vol. 37, No. 10, October 2004. pp. 64-72.
3. Silver B., "BPMN Method and Syle," 2009.
4. OMG, "Business Process Model and Notation (BPMN) Version 2.0," Object Management Group, OMG Document Number: formal/2011-01-03, 2012.
5. Börger E., "Approaches to modeling business processes: a critical analysis of BPMN, workflow patterns and YAWL, "Journal Software and Systems Modeling (SoSyM), Vol. 11, No. 3, July 2012. pp. 305-318.
6. Harel D., Rumpe B, "Modeling Languages: Syntax, Semantics and All That Stuff, Part I: The Basic Stuff," Weizmann Science Press of Israel©, Jerusalem, Israel, Technical Report 2000. 1-28 pp.
7. Ullmann S., "Semantics: An Introduction to the Science of Meaning, ". Oxford,: Basil Blackwell, 1972. 278 pp.
8. Bunge M. Ontology I: "The Furniture of the World, Treatise on Basic Philosophy, " Vol 3. Boston, MA,: D. Reidel Publishing Company, 1977.
9. Уемов А. И. Вещи, свойства и отношения. -М.: 1963.

10. Soffer P., Wand Y., "On the Notion of Soft-Goals in Business Process Modeling," Business Process Management Journal, Vol. 11, No. 6, 2005. pp. 663 - 679.
11. Wand Y., Weber R., "Research Commentary: Information Systems and Conceptual Modeling -- A Research Agenda," Information Systems Research, Vol. 13, No. 4, 2002. pp. 363-376.
12. Yourdon E. Modern Structured Analysis. Prentice Hall, 1988. 688 pp.
13. Сотников А.Н., Фёдоров И.Г., "Создание исполняемой модели бизнес-процесса без программирования, миф или реальность? " // Инжиниринговые науки, телекоммуникационные и информационные технологи (En&T-2014). Долгопрудный. 2014.
14. Moody D.L., "The “Physics” of Notations: Toward a Scientific Basis for Constructing Visual Notations in Software Engineering Software Engineering,» Software Engineering, IEEE Tr., Vol. 35, No. 6, 2009. pp. 756 - 779.
15. Ерщов А.П., "Основы информатики и вычислительной техники: Пробное учебное пособие для средних учебных заведений," В 2-х частях / Под редакцией А. П. Ершова и В. М. Монахова..," 1985.
16. Поспелов Д.А., Осипов Г.С., "Введение в прикладную семиотику" // Новости искусственного интеллекта. 2002. Vol. 54. No. 6. pp. 28-35.
17. OMG. // XML Metadata Interchange (XMI)® [Электронный ресурс]. URL: http://www.omg.org/spec/XMI/ (дата обращения: 15.01.2015).
18. Энгельс Ф. Диалектика природы // In: Соч., т.20. / Ed. by Маркс К. Э.Ф. pp. 339-626.
19. Soffer P., Wand Y., "On the Notion of Soft-Goals in Business Process Modeling," Business Process Management Journal, Vol. 11, No. 6, 2005. pp. 663 - 679.
20. van der Aalst W., "Verification of workflow nets" // In: Application and theory of Petri nets. Heidelberg: Springer, Lecture notes in computer science, 1997. pp. 407–426.
21. Репин В.В., Елиферов В.Г. "Процессный подход к управлению. Моделирование бизнес- процессов". -М. : Манн, Иванов и Фербер, 2013. 544 pp.
22. Nissen M. A., "Configuration-Contingent Enterprise Redesign Model" // In: Business Process Engineering. Advancing the State of the Art. Kmuwer Academic Publishing, 1999. P. 391.
23. Фёдоров И.Г. "Формальное представление поведенческой перспективы модели бизнес-процесса" // Открытые системы. 2012.

Афанасьевский Л.Б.[1], Горин А.Н.[2], Фадин А.Г.[3]

[1]ВУНЦ ВВС «Военно-воздушная академия им. проф. Н.Е. Жуковского и Ю.А. Гагарина», г. Воронеж, к.т.н., доцент кафедры АСУ, afleonid@yandex.ru

[2]ВУНЦ ВВС «Военно-воздушная академия им. проф. Н.Е. Жуковского и Ю.А. Гагарина», г. Воронеж, к.т.н., старший преподаватель кафедры АСУ, algorin.algoral@mail.ru

[3]ВУНЦ ВВС «Военно-воздушная академия им. проф. Н.Е. Жуковского и Ю.А. Гагарина», г. Воронеж, к.т.н., профессор кафедры АСУ, afleonid@yandex.ru

РЕАЛИЗАЦИЯ АНАЛИТИЧЕСКОЙ И ИМИТАЦИОННОЙ МОДЕЛЕЙ СИСТЕМЫ МАССОВОГО ОБСЛУЖИВАНИЯ

КЛЮЧЕВЫЕ СЛОВА

Система массового обслуживания, поток заявок, канал, состояние, характеристика СМО, очередь заявок.

АННОТАЦИЯ

В статье обсуждается порядок разработки аналитической модели систем массового обслуживания (СМО) в среде MathCad, характеристики эффективности СМО. Рассмотрена методика построения имитационной модели многофазной многоканальной системы массового обслуживания (СМО) при наличии нескольких различных источников заявок. В основу модели положено событийное продвижение модельного времени в соответствии с принципом σz. При программной реализации модели большинство параметров в фазах обслуживания и в СМО в целом определяется по рекуррентным алгоритмам.

К системам массового обслуживания (СМО) может быть отнесено достаточно большое количество реальных систем, например, системы связи, ремонтные и сервисные фирмы, компьютерные сети, аэропорты, морские порты, кассы по продаже билетов и т.д.

Рассмотрена программная реализация в MathCad решения моделей динамики и статики многоканальной марковской СМО с ограниченной очередью типа M/M/n/m, где n – число каналов обслуживания заявок, m – длина очереди.

На вход СМО поступает поток заявок с интенсивностью λ. Интенсивность обслуживания в каждом из каналов равна μ.

Процесс функционирования рассматриваемой СМО состоит в следующем. Заявки поступают в систему случайным образом и в какой-то момент времени они могут занять все свободные каналы. Тогда последующие заявки будут ставиться в очередь. Если же число необслуженных заявок превысит $n + m$, то части из них будет отказано в обслуживании, так как все каналы и места в очереди будут заняты. Поступление заявок в каналы обслуживания и в очередь происходит с интенсивностью входящего потока λ. Интенсивности обслуживания заявок системой с увеличением числа занятых каналов возрастают и равняются $i \cdot \mu$, где i =1, 2,..., n – количество занятых каналов в системе.

Если все каналы обслуживания заняты, то интенсивность продвижения заявок в очереди постоянна, совпадает с интенсивностью обслуживания заявок n каналами и, следовательно, равна $n \cdot \mu$. Общее число состояний рассматриваемой системы равно $n+m+1$.

Граф состояний и переходов рассматриваемой системы представлен следующим образом (рис. 1).

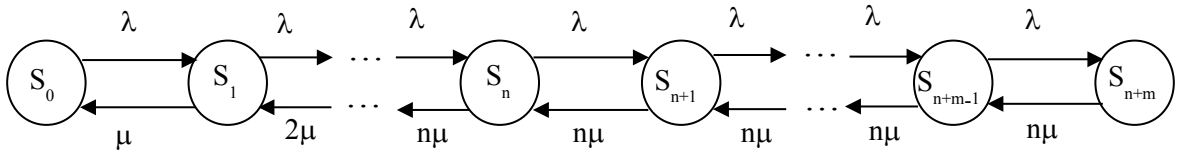


Рис. 1. Граф состояний и переходов многоканальной СМО с очередью

Состояния в рассматриваемой СМО имеют следующий смысл:

S_0 – все каналы свободны; S_1 – один канал занят обслуживанием заявки, остальные каналы свободны; S_2 – заняты два канала; S_n – заняты все n каналов для обслуживания n поступивших заявок; S_{n+1} – заняты n каналов и одна заявка находится в очереди; S_{n+m} – заняты n каналов и m заявок находятся в очереди.

Динамика СМО рассматриваемого типа описывается системой линейных дифференциальных уравнений Колмогорова-Чепмена [1], в которой искомыми переменными являются вероятности $P_i(t)$, i = 0, 1, ..., n+m нахождения СМО в каждом из n+m+1 состояний как функции времени t, а коэффициенты являются функциями интенсивностей %lambda и % m u .

Начальные условия для решения уравнений Колмогорова-Чепмена имеют вид

$$P_0(0)=1;\ P_1(0)=P_2(0)=\ldots P_n(0)=0\ .$$

Формулы для расчета значений вероятностей состояний в стационарном режиме легко получаются из системы уравнений Колмогорова-Чепмена при приравнивании производных нулю, исключении зависимостей вероятностей от времени и замене одного из уравнений выражением

$$P_0+P_1+P_2+\ldots+P_n=1$$

и имеют следующий вид:

$$P_0=\frac{1}{1+\frac{\lambda}{\mu}+\frac{\lambda^2}{n!\cdot\mu^2}+\ldots+\frac{\lambda^n}{n!\cdot\mu^n}+\frac{\lambda^n}{n!\cdot\mu^n}\cdot\left[\frac{\lambda}{n\cdot\mu}+\frac{\lambda^2}{n^2\mu^2}+\ldots+\frac{\lambda^m}{n^m\mu^m}\right]};$$

$$P_1=\frac{\lambda}{1!\,\mu}P_0,\quad P_2=\frac{\lambda^2}{2!\,\mu^2}P_0,\ \ldots,\quad P_n=\frac{\lambda^n}{n!\ \lambda}P_0\ ;$$

$$P_{n+1}=\frac{\lambda^{n+1}}{n\cdot n!\ \mu^{n+1}}P_0,\ \ P_{n+2}=\frac{\lambda^{n+2}}{n^2\cdot n!\ \mu^{n+2}}P_0,\ \ldots,\quad P_{n+m}=\frac{\lambda^{n+m}}{n^m\cdot n!\ \mu^m}P_0.$$

Заметим, что в выражении для P_0 квадратная скобка отсутствует, если СМО представляет собой систему с отказами.

К характеристикам СМО относят:

$P_{обс}$ – вероятность обслуживания заявок;

$P_{отк}$ – вероятность отказа в обслуживании;

Q – абсолютную пропускную способность (среднее число заявок, обслуженных за единицу времени);

q – относительную пропускную способность (вероятность обслуживания в момент времени t, характеризуемую отношением среднего числа обслуженных заявок Q к среднему числу поступивших заявок %lambda за единицу времени);

$n_{зан}$ – среднее число занятых каналов;

$k_{ис}$, $k_{пр}$ – коэффициенты использования и простоя каналов;

$t_{ож}$ – среднее время ожидания заявки в очереди;

$m_{оч}$ – среднее количество заявок, ожидающих обслуживания;

$m_{обс}$ – среднее количество заявок, находящихся на обслуживании;

m_c – среднее количество заявок в системе (в очереди и на обслуживании);

t_c – среднее время пребывания заявки в системе.

С учетом формул, приведенных выше, можно записать выражения для вычисления характеристик СМО, например

$$P_{отк}=P_{n+m};\qquad q=1-P_{отк}=P_{обс};\quad Q=\lambda q;\qquad n_{зан}=\frac{Q}{\mu};$$

$$k_{ис}=\frac{n_{зан}}{n};\quad k_{пр}=1-k_{ис};\ \ m_c=m_{оч}+m_{обс};\ \ t_{ож}=\frac{m_{оч}}{\lambda};\ \ m_{обс}=\frac{q}{\mu}.$$

Фрагмент рабочего окна MathCad, в котором динамическая модель марковской СМО оформлена как программный блок, представлен на рис. 2. Эта модель носит общий характер и позволяет путем задания значений структурных параметров получить описание одноканальной и многоканальной СМО с отказами и с очередями. На основе этой модели могут быть получены

оценки всех характеристик СМО. Подход, принятый при формировании модели, может быть распространен и на исследование с помощью аналитических моделей немарковских СМО.

Решение уравнений Колмогорова-Чепмена в MathCad с помощью функции Rkadapt, реализующей метод Рунге-Кутта с автоматическим выбором шага интегрирования, представлено на рис. 3.

История создания и развития методов аналитического описания систем массового обслуживания (СМО) занимает значительный промежуток времени [2, 3]. Основным применением этой теории являются однофазные системы. Для многофазных многоканальных СМО аналитические результаты получены только для марковских систем при обслуживании одного типа заявок [3].

Исследование динамики и статики марковской СМО

$n := 2$ - количество каналов в СМО $m := 1$ - максимальная длина очереди в СМО

$\lambda := 0.67$ - интенсивность входящего потока заявок

$\mu := 0.15$ - интенсивность потока обслуженных заявок

$$P := \left| \begin{array}{l} \text{for } i \in 0..n+m \\ \quad S_i \leftarrow \left| \begin{array}{ll} 1 & \text{if } i = 0 \\ 0 & \text{otherwise} \end{array} \right. \quad \text{- формирование вектора начальных условий} \\ S \end{array} \right.$$

Уравнения Колмогорова - Чепмена

$$D(t,P) := \left| \begin{array}{l} \text{for } i \in 0..n+m \\ \quad a_i \leftarrow \left| \begin{array}{ll} P_{i-1}\cdot\lambda & \text{if } 0 < i \le n+m \\ 0 & \text{if } i = 0 \end{array} \right. \\ \quad b_i \leftarrow \left| \begin{array}{ll} P_{i+1}\cdot\mu\cdot(i+1) & \text{if } i < n \\ P_{i+1}\cdot\mu\cdot n & \text{if } n \le i < n+m \\ 0 & \text{if } i = n+m \end{array} \right. \\ \quad c_i \leftarrow \left| \begin{array}{ll} -P_i\cdot(\lambda + i\cdot\mu) + a_i + b_i & \text{if } i < n \\ -P_i\cdot(\lambda + n\cdot\mu) + a_i + b_i & \text{if } n \le i \le n+m-1 \\ -P_{n+m}\cdot n\cdot\mu + a_i + b_i & \text{if } i = n+m \end{array} \right. \\ c \end{array} \right.$$

Рис. 2. Программа формирования уравнений Колмогорова-Чепмена

Имитационное моделирование по-прежнему остается основным инструментом исследования сложных многофазных многоканальных СМО с ограниченными и бесконечными очередями, при этом описанные в литературе методики моделирования носят скорее концептуальный характер, что делает эти методики практически мало пригодными для программной реализации [2, 4].

В последние годы большинство публикаций, например [5, 6], посвящены применению для имитационного моделирования инструментальных систем GPSS World, AnyLogic, Simulink и др., при этом описание методики моделирования как таковой отсутствует, а возможности систем иллюстрируются решением достаточно простых задач.

При изучении имитационного моделирования сложных систем с использованием инструментальных средств универсального назначения основные усилия и время затрачиваются на освоение этих средств, при этом изложение методики также ограничивается ориентацией ее на решение достаточно простых задач.

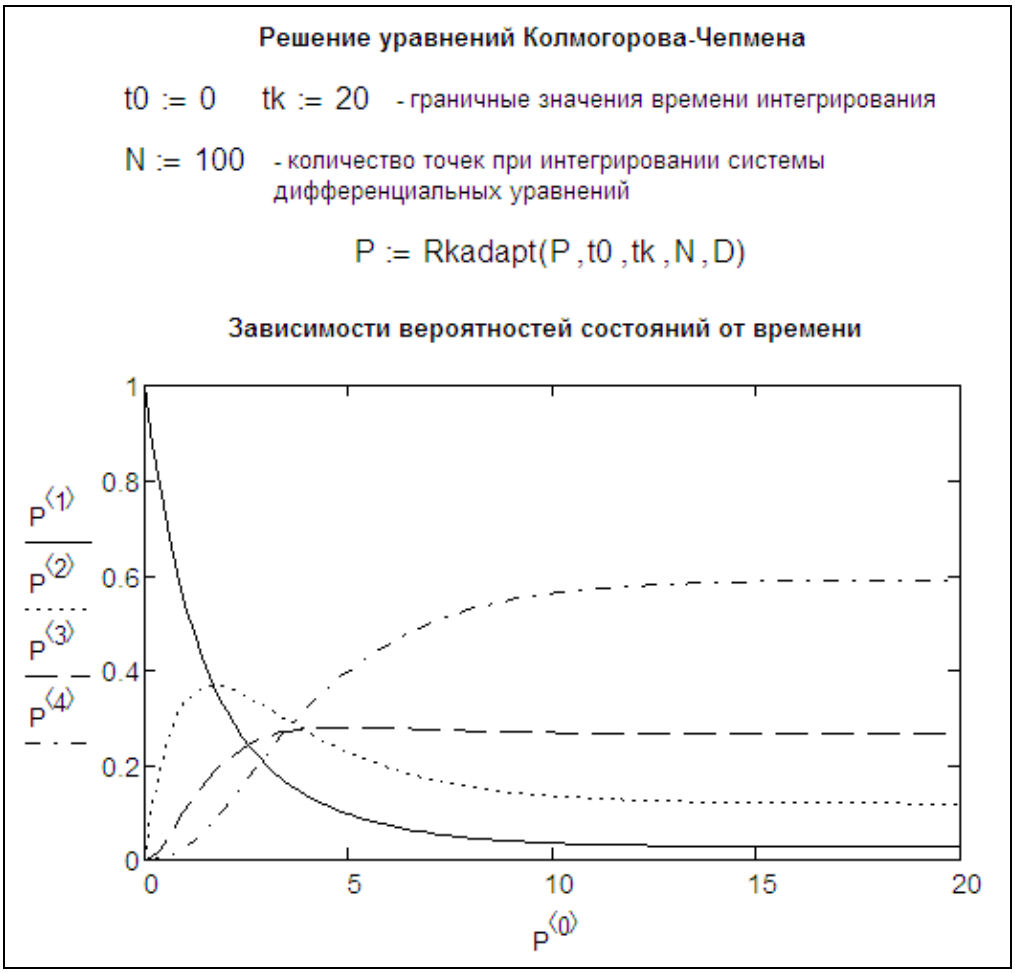


Рис. 3. Решение уравнений динамики СМО

Для имитационного моделирования дискретных событийных систем в среде моделирования Simulink, интегрированной в MatLab, имеется библиотека SimEvents. Модель многофазной многоканальной СМО, построенная с использованием SimEvents, оказывается очень громоздкой и жестко привязанной к конкретной структуре СМО. Пакет моделирования событийно-управляемых систем Stateflow, также входящий в Simulink, в своей основе имеет теорию конечных автоматов. Он также может быть использован для имитационного моделирования СМО, но с целым рядом ухищрений.

С учетом сказанного достаточно актуальной является разработка методики имитационного моделирования многофазных СМО.

Рассматриваемая ниже методика предполагает использование в качестве инструмента для реализации модели универсального языка программирования и ориентирована на имитацию работы СМО при следующих достаточно общих предпосылках и допущениях:

- режим поступления заявок и дисциплина их обслуживания являются случайными, при этом для них предполагаются заданными законы распределения вероятностей и их параметры;
- продвижение модельного времени осуществляется по событиям (принцип σz);
- каналы в фазах обслуживания являются однотипными для каждого типа заявок;
- в первой фазе обслуживания длины очередей для каждого типа заявок ограничены, т.е. в СМО возможны потери заявок;
- во всех фазах, кроме первой, очереди считаются бесконечными, т.е. заявка, принятая на обслуживание в первой фазе, обязательно будет обслужена;
- исходные данные для моделирования должны формироваться в файлах текстового формата, что обеспечит гибкость модели.

Суть методики моделирования заключается в следующем.

Раздельно генерируются времена наступления событий в активных элементах СМО (источниках заявок и каналах при их освобождении). Полученные данные записываются в четыре массива, различающиеся их назначением и способом модификации элементов.

Два из этих массивов (назовем их tz и tk) используются для определения ближайшего события в СМО. Эта процедура включает в себя определение минимального элемента tzmin в одномерном массиве tz и соответствующего ему типа заявки, определение минимального элемента tkmin в двумерном массиве tk и соответствующего ему номера фазы обслуживания и типа каналов. Целесообразность разделения времен наступления событий на два массива обусловлена различием процессов обработки информации при появлении очередной заявки и освобождения канала обслуживания. Результаты указанных операций являются начальными условиями для циклического процесса имитации функционирования СМО.

В цикле выполняется сравнение tzmin и tkmin. При tzmin ≤ tkmin увеличивается модельное время и реализуется передача заявки в первую фазу обслуживания, при этом возможны следующие ситуации:

1) имеются свободные каналы, один из которых будет занят новой заявкой;
2) свободные каналы отсутствуют, длина очереди меньше допустимой, заявка занимает место в очереди (очередь моделируется целым числом, равным количеству заявок в ней);
3) свободные каналы отсутствуют, очередь заполнена, заявка теряется.

Если tkmin < tzmin, то модифицируется модельное время, выполняются операции освобождения канала в текущей фазе и передача заявки в следующую фазу, если фаза не является последней, при этом возможны следующие ситуации:

- в текущей фазе часть каналов занята, в следующей фазе имеются свободные каналы, освобождается канал в текущей фазе, заявка передается в канал следующей фазы;
- в текущей фазе часть каналов занята, в следующей фазе свободных каналов нет, освобождается канал в текущей фазе, заявка передается в очередь следующей фазы;
- в текущей фазе все каналы заняты, в следующей фазе имеются свободные каналы, уменьшается длина очереди в текущей фазе, заявка передается в канал следующей фазы;
- в текущей и следующей фазах все каналы заняты, освобождается канал или уменьшается длина очереди в текущей фазе, заявка передается в очередь следующей фазы;
- в текущей фазе все каналы свободны, состояния каналов и очереди не изменяются, определяется среднее время простоя каналов.

В последней фазе при соответствующих условиях выполняется только операция освобождения каналов, при этом определяются средние значения параметров, характеризующих работу фазы.

Для определения в каждой фазе и в СМО в целом средних значений параметров, имеющих смысл количества (среднее количество занятых каналов, средняя длина очереди и др.), целесообразно использовать рекуррентную формулу

$$m_x(n)=m_x(n-1)+\frac{1}{n}[x(n)-m_x(n-1)],\quad n=1, 2, \ldots, k, \qquad (1)$$

где n – количество событий, модифицируемое при наступлении соответствующего события;

m_x – среднее значение параметра x;

x(n) – значение параметра x при n-ом событии.

Определение средних значений параметров, имеющих смысл времени (среднее время нахождения заявок в очереди, среднее время простоя каналов и др.), выполняется с использованием формулы вида (1), но в ней параметр x(n) является временем между последними событиями соответствующего типа. Эти времена для всего множества типов событий хранятся в двух других массивах (tzs – время между заявками; tks – время между событиями в каналах). При наступлении какого-либо события в указанных массивах модифицируется только значение времени, связанное с этим событием.

После завершения описанных действий, связанных с обслуживанием заявок, значения в массивах tz и tk уменьшаются на величину tzmin при tzmin ≤ tkmin или на величину tkmin, если tkmin < tzmin, при этом на место нулевого элемента в модифицированных массивах записывается вновь генерируемое время наступления события соответствующего типа.

Далее идентифицируется новое ближайшее событие (определяются tzmin и tkmin, устанавливаются тип заявки, номер фазы обслуживания и тип каналов) и для него выполняются все описанные выше операции, приводящие к изменению состояния каналов обслуживания или

очередей.

Описанные вычислительные процедуры повторяются в цикле до тех пор, пока не будет достигнуто предельное значение модельного времени, указанное в файле исходных данных.

Укрупненная схема алгоритма моделирования СМО, реализующего описанную методику, имеет следующий вид (рис. 4)

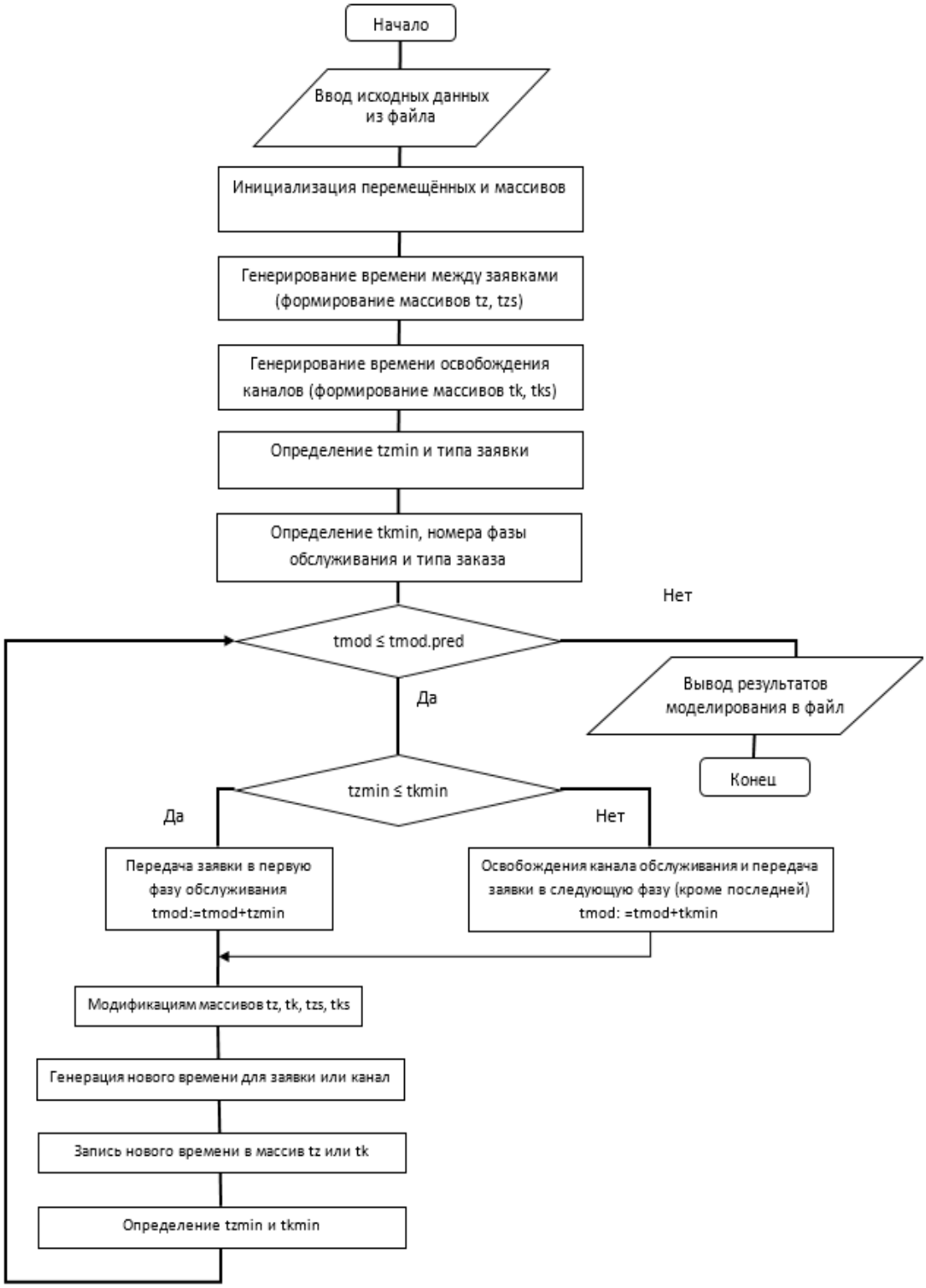


Рис. 4. Укрупненная схема алгоритма имитационного моделирования СМО

Для работы программы файл входных данных должен содержать значения следующих параметров модели:

- предельное время моделирования;
- количество типов заявок (каналов обслуживания в фазах);
- вид закона распределения вероятностей для времени между поступлениями заявок каждого типа (1 – экспоненциальный; 2 – равномерный);
- параметр экспоненциального распределения для каждого типа заявок (0, если закон отличается от экспоненциального);
- параметр (нижняя граница) равномерного распределения (0, если закон отличается от равномерного);
- параметр (верхняя граница) равномерного распределения (0, если закон отличается от равномерного);
- количество фаз обслуживания;
- количество каналов каждого типа во всех фазах;
- вид плотности распределения для времени обслуживания заявок в каналах каждого типа во всех фазах (треугольный; аппроксимированный двумя параболами, сопрягаемыми в точке экстремума и др.);
- параметры для каждого вида закона распределения;
- предельная длина очереди для заявок каждого типа на входе первой фазы обслуживания.

Можно сформировать несколько вариантов исходных данных в разных файлах и выполнить анализ функционирования СМО для каждого из них. Такой подход целесообразен для вновь разрабатываемых систем.

Результаты работы программы целесообразно выводить в файл текстового формата, которые должны содержать:

1)время моделирования;

2)общее количество заявок, поступивших в СМО, количества обслуженных и потерянных заявок;

3)абсолютную и относительную пропускную способность СМО;

4)средние значения для разных параметров (количества занятых каналов, длины очереди, времени простоя каналов, времени нахождения заявок каждого типа в очереди на входе каждой фазы, времени обслуживания заявок каждого типа в каждой фазе и в СМО в целом и др.).

Программная реализация методики имитационного моделирования вследствие цикличности большинства операций является относительно простой и выполнена в Delphi.

В MathCad реализована имитационная модель только однофазной многоканальной марковской СМО, для которой выше рассмотрена аналитическая модель.

Наращивание возможностей программы без изменения методики моделирования может выполняться по следующим направлениям:

- расширение состава законов распределения для времени между поступлениями заявок и процессов их обслуживания в каналах;
- использование разнотипных каналов для обслуживания одного и того же типа заявок;
- применение динамических массивов для исключения ограничений по количеству типов заявок и фаз обслуживания.

Программа, разработанная по описанной методике, может быть использована для анализа функционирования сложных систем, в частности, для выявления в них «узких» мест, например для:

- средств приема и развертывания подразделений МЧС в зоне стихийного бедствия;
- многостадийных дискретных производств с разными типами технологического оборудования.

Литература

1. Фадин А.Г. Моделирование радиоэлектронных систем на ЭВМ. – Воронеж: ВИРЭ, 2000. – 496 с.
2. Иващенко А.В., Сыпченко Р.П. Основы моделирования сложных систем на ЭВМ. – Л.: ЛВВИУС, 1988. – 272 с.
3. Вишневский В.М. Теоретические основы проектирования компьютерных сетей. – М.: Техносфера, 2003. – 512 с.
4. Шелухин О.И., Тенякшев А.М., Осин А.В. Моделирование информационных систем. – М.: Сайнс-пресс, 2005. – 368 с.
5. Кудрявцев Е.М. GPSS World. Основы имитационного моделирования различных систем. – М.: ДМК Пресс, 2004. – 320 с.
6. Дьяконов В.П. MatLab 6.5 SP1/7.0 + Simulink 5/6. Основы применения. – М.: СОЛОН-Пресс, 2005. – 800 с.

НАУЧНОЕ ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ
В ОБРАЗОВАНИИ И НАУКЕ

Лазовская Т.В.[1], Тархов Д.А.[2]

[1] Санкт-Петербургский государственный политехнический университет Петра Великого, г.Санкт-Петербург, ст. преподаватель кафедры Высшая математика, ИПММ, tatianala@list.ru

[2] Санкт-Петербургский государственный политехнический университет Петра Великого, г.Санкт-Петербург, д.т.н., д.т.н., профессор кафедры Высшая математика, ИПММ, dtarkhov@gmail.com

ИСПОЛЬЗОВАНИЕ АСИМПТОТИЧЕСКИХ РЕШЕНИЙ ПРИ ПОСТРОЕНИИ НЕЙРОСЕТЕВОЙ МОДЕЛИ В ЖЕСТКОЙ ЗАДАЧЕ

КЛЮЧЕВЫЕ СЛОВА

Нейронные сети, дифференциальные уравнения, параметризованная математическая модель, двусторонняя асимптотика.

АННОТАЦИЯ

В статье рассматриваются новые подходы к нейросетевому решению параметризованных дифференциальных уравнений на примере модельной задачи. Применяются такие методы, как частичная перегенерация тестовых точек, использование дополнительной информации различных типов, в том числе алгоритм, использующий условия двусторонней асимптотики. Рассмотрены возможности расширения интервала изменения параметра.

Методология Васильева-Тархова [1] построения нейросетевых моделей на основе дифференциальных уравнений и прочих данных различного типа позволяет решать сложные, в том числе некорректные, задачи математической физики. В данной статье рассматриваются параметризованные нейронные сети, позволяющие одновременно решать совокупность задач, зависящих от одного или нескольких параметров, сразу. Апробация новых методов проводилась на одной модельной задаче [2].

Рассмотрим параметризованную задачу Коши

$$\left\{\begin{matrix} y'=-\alpha\left(y-\cos x\right) \\ y(0)=0 \end{matrix}\right\} \qquad (1)$$

При значении параметра $\alpha=50$ и более задача (1) является жесткой в том смысле, что решение ее методом Эйлера нестабильно в окрестности нуля. Таким образом, достаточно ограничиться рассмотрением $x\in[0,1]$. Что касается интервала изменения параметра, то рассмотрены как результаты для ранее рассматривавшегося [2,3] интервала $\alpha\in[0.5,50]$, так и для расширенного по сравнению с предыдущими исследованиями [2,3] до [0.1, 100].

Приближенное решение задачи ищется в виде выхода искусственной нейронной сети заданной архитектуры

$$y(x)=\sum_{i=1}^{n} c_i v(x,\alpha,\mathbf{a}_i),$$

веса которой $\{c_i,\mathbf{a}_i\}_{i=1}^{n}$ определяются в процессе минимизации функционала ошибки

$$\sum_{j=1}^{m}\left(y'(\xi_j)+\alpha_j\left(y(\xi_j)-\cos(\xi_j)\right)\right)^2+\delta\, y^2(0).$$

Пары тестовых точек (α_j,ξ_j) моделируются как равномерно распределенные в двумерной области изменения переменной *x* и параметра α случайные величины. После нескольких итераций алгоритма оптимизации такие точки выбираются заново. Новый случайный выбор пробных точек на некотором шаге будем называть перегенерацией тестовых точек.

Что касается дополнительных данных, их введение в нейросетевую модель происходит путем добавления в функционал ошибки соответствующих новых слагаемых. Так, в случае точечной информации о решении вида $(x_j,f(x_j))$ такое дополнительное слагаемое будет иметь

вид $\delta_1 \sum_{j=1}^{m} (f(x_j) - y(x_j))^2$, где константа $\delta_1 > 0$ означает штрафной вес, выбираемый в зависимости от свойств данных или иных условий. Дополнительные данные в виде уравнений учитываются аналогично.

Оптимизация функционала ошибки проводилась по алгоритму, сочетающему RProp и метод облака; точки случайным образом перегенерировались каждые три шага, облако состояло из трех частиц. Выбор задачи позволяет оценить согласованность полученного результата с аналитическим решением уравнения (1).

В предыдущих работах [2,3] были исследованы два типа нейронных сетей для решения задачи с интервалом изменения параметра α равным [0.5, 50]. В качестве базисных функций рассматривались сигмоиды и умноженные на *x* гауссианы, причем последние изначально удовлетворяют краевому условию задачи (1). Для обеих сетей рассматривались две модели, зависящие от типа дополнительных данных. В первом случае использовалась точечная информация о решении, роль которой играли поточечные решения задачи явным методом Эйлера для значений параметра α равных 5 и 50. Как уже упоминалось, при $\alpha = 50$ такое решение получается с достаточно большой ошибкой, что позволяет походу исследовать робастность данного метода. Эксперименты с различным сочетанием данных в модели, а также разным количеством функций активации показали, что сеть-персептрон более восприимчива к введению новой информации и требует меньшего числа итераций.

Остановимся более подробно на модели, где в качестве дополнительных данных используется информация в виде уравнений, которые мы получим непосредственно из условия решаемой задачи. Отметим, что при достаточно малых α имеет место приближение

$$y \simeq \alpha \sin x . \qquad (2)$$

Данное асимптотическое условие мы будем учитывать в функционале ошибки, добавив в него слагаемое вида $\delta_2 \sum_{k=1}^{m_2} (y(x_k, \alpha_k / M) - \alpha_k / M \sin x_k)^2$, где *M*>0 достаточно велико. Гетерогенная модель, учитывающая данные обоих типов, была исследована для *n*=20 базисных функций и показала улучшение результатов по сравнению с базовой. Таким образом, использование даже неточной дополнительной информации позволяет улучшить нейросетевую модель.

Как было отмечено выше, в ходе реализации алгоритма оптимизации происходит полная перегенерация тестовых точек. Если ее не проводить, то нейросетевой метод можно считать аналогом метода, называемого метод коллокаций, где приближение строится для набора фиксированных точек. В статье [3] был введен алгоритм частичной перегенерации тестовых точек. Вводится параметр d_t, принимающий любые значения в промежутке от 0 до 1 и отражающий долю точек, фиксирующихся от одной итерации к другой. Например, $d_t = 0$ означает полную перегенерацию (традиционно используемую Васильевым и Тарховым), т. е. все точки заново выбираются случайным образом перед каждой итерацией, $d_t = 1$ соответствует методу коллокаций. Для промежуточных значений параметра используется следующее правило: фиксируется $d_t m$ точек из всех *m* тестовых точек с наибольшими значениями слагаемых в соответствующей сумме функционала ошибки, а остальные перегенерируются случайным образом. При первой итерации во всех случаях точки выбираются так, чтобы они были случайно и равномерно распределены на рассматриваемой области.

Первая группа экспериментов была проведена для нейросетевой модели с параметрами $d_t = 0.0, 0.3, 0.5, 0.7, 1.0$, *n*=20 персептронными базисными функциями, как более восприимчивыми, в случае точечных дополнительных данных. Число тестовых точек m выбиралось равным 20, общее число итераций *k*=300. Результаты показали, что частичная перегенерация снижает зависимость от начального выбора точек, а также в ряде задач помогает получить более стабильное решение. Другая серия экспериментов исследовала уже модель с гетерогенными данными. Вариация асимптотического условия выражалась в рассмотрении параметра $M = 50, 100, 200$, параметр d_t изменялся аналогично предыдущим моделям, число тестовых точек *m*=20, 50. Результаты для модели при *M*=50 подтвердили гипотезу, что при малом числе тестовых точек целесообразно использовать полную перегенерацию точек ($d_t = 0$). При

этом, выбор достаточно большого числа пробных точек обеспечивает наилучшее приближение решения задачи при их фиксации. Отметим, что при использовании асимптотического условия с параметром *M*=100, 200 задача (1) становится неклассической. Так, если $\alpha_j \in [0.5, 50]$, то $\alpha_j / M \in [0.005, 0.5]$ даже при *M*=100, то есть условие накладывается вне области поиска решения. Как показали результаты вычислительных экспериментов, в таких случаях предпочтительно выбирать модель с частичной перегенерацией тестовых точек. Уточнение модели при прочих равных наглядно демонстрирует сравнение рис.1 и рис.2.

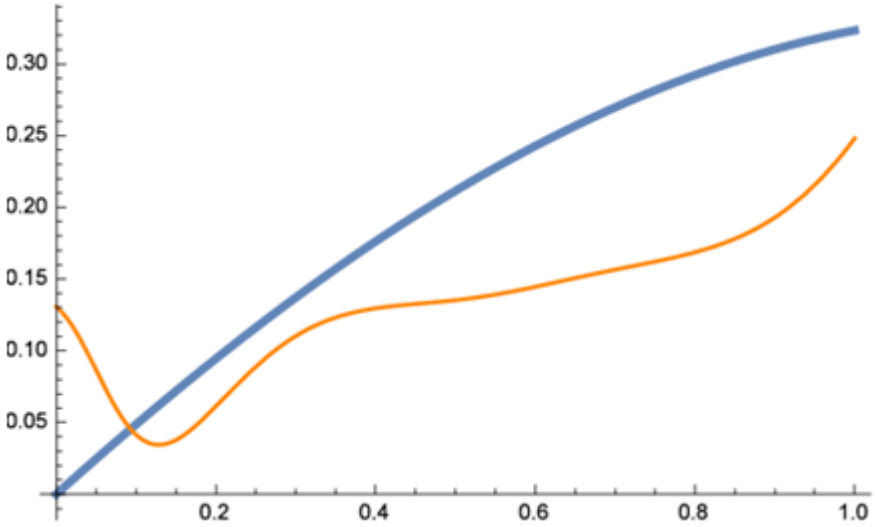


Рис.1. Нейросетевое приближение (красным) решения задачи (1). Гетерогенная модель с параметрами $\alpha = 0.5$, $d_t = 0$, $x \in [0,1]$, $\alpha_j \in [0.5, 50]$, *M=100*

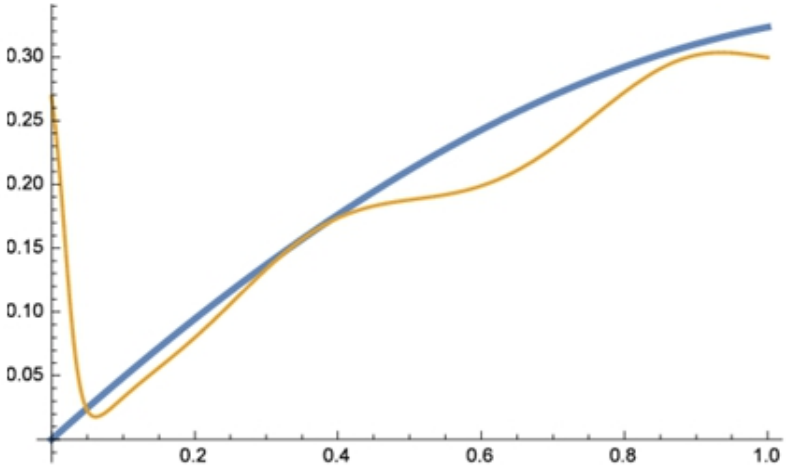


Рис.2. Нейросетевое приближение (красным) решения задачи (1). Гетерогенная модель с частичной перегенерацией с параметрами $\alpha = 0.5$, $d_t = 0.3$, $x \in [0,1]$, $\alpha_j \in [0.5, 50]$, *M=100*

Расширим теперь интервал изменения параметра α до [0.01, 100] и рассмотрим новый тип алгоритма построения модели, где будут использоваться только асимптотические условия. Для малых α уточним условие, использованное в предыдущей модели, применяя к уравнению задачи (1) метод разложения по малому параметру [4]. Тогда асимптотическое разложение решения, например, третьего порядка будет иметь вид

$$y \cong As_1(x,\alpha) := \alpha \sin x + \alpha^2(\cos x - 1) - \alpha^3(\sin x - x). \quad (3)$$

Для больших α, то в качестве малого параметра можно рассматривать $1/\alpha$. В этом случае разложение примет форму

$$y \cong \cos x + \frac{\sin x}{\alpha} - \frac{\cos x}{\alpha^2} - \frac{\sin x}{\alpha^3}. \quad (4)$$

Отметим, что в окрестности нуля разложение (4) не удовлетворяет граничному условию задачи (1). Скорректируем условие следующим образом. Рассматривая в качестве малого параметра саму переменную *x*, получим вышеупомянутым методом новое разложение

$$y \cong \alpha x - \frac{(\alpha x)^2}{2} + (\alpha^3 - \alpha)\frac{x^3}{6}.$$

Тогда вместо условия (4) будем использовать следующее

$$y \cong As_2(x,\alpha) = \min\left(\cos x + \frac{\sin x}{\alpha} - \frac{\cos x}{\alpha^2} - \frac{\sin x}{\alpha^3}, \alpha x - \frac{(\alpha x)^2}{2} + (\alpha^3 - \alpha)\frac{x^3}{6} \right) \quad (5)$$

Изменения алгоритма естественно вносятся в нейросетевую модель путем модификации исходного функционала ошибки введением дополнительных слагаемых

$$\delta_1 \sum_{k=1}^{m_1} \left(y(x_i, \frac{\alpha_i}{M_1}) - As_1(x_i, \frac{\alpha_i}{M_1}) \right)^2 + \delta_2 \sum_{k=1}^{m_2} \left(y(x_i, \alpha_i + M_2) - As_2(x_i, \alpha_i + M_2) \right)^2$$

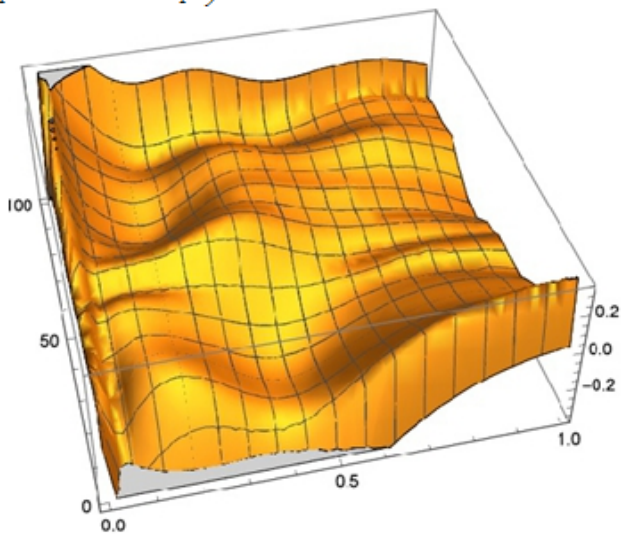


Рис.3. Ошибка нейросетевого приближения решения задачи (1). Базовая модель с параметрами $d_t = 0$, $x \in [0,1]$, $\alpha \in [0.01, 100]$.

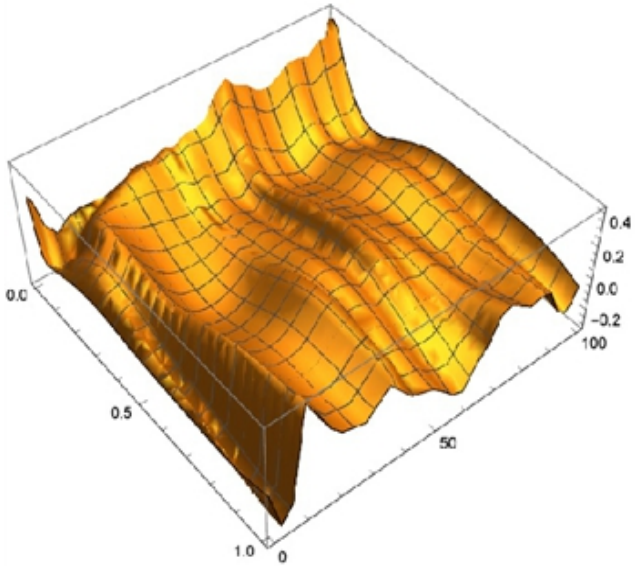


Рис.4. Ошибка нейросетевого приближения решения задачи (1). Модель двусторонней асимптотики с параметрами $d_t = 0$, $x \in [0,1]$, $\alpha \in [0.01, 100]$.

Алгоритм двусторонней асимптотики был опробован для решения задачи (1) с помощью нейросетевой модели с параметрами $n=50$, $m=100$, $k=100$ и уже упомянутого расширенного интервала изменения параметра [0.01,100]. Сравнение результатов по среднему и минимальному значению среднеквадратичной ошибки полученного приближенного решения в 10^5 равномерно распределенных на соответствующей области точках показало значимую эффективность асимптотической модели. Трехмерные изображения ошибки на всей области изменения переменной *x* и параметра α представлены для базовой модели (рис.3) и асимптотической (рис.4), значения оценки *c* равны 0.12 и 0,11 соответственно.

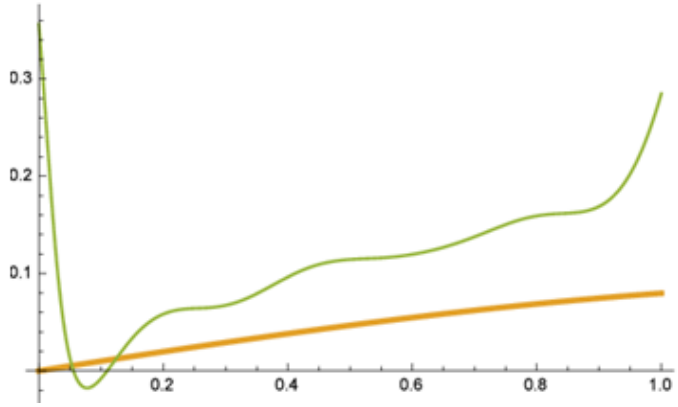


Рис.5. Нейросетевое приближение (зеленым) решения задачи (1). Модель двусторонней асимптотики с параметрами $d_t=0$, $x\in[0,1]$, $\alpha=0.1$, $c=0.07$

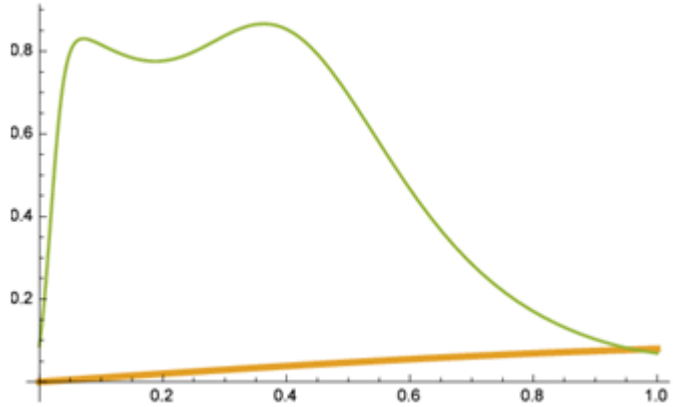


Рис.6. Нейросетевое приближение (зеленым) решения задачи (1). Базовая модель с параметрами $d_t=0$, $x\in[0,1]$, $\alpha=0.1$, $c=0.12$

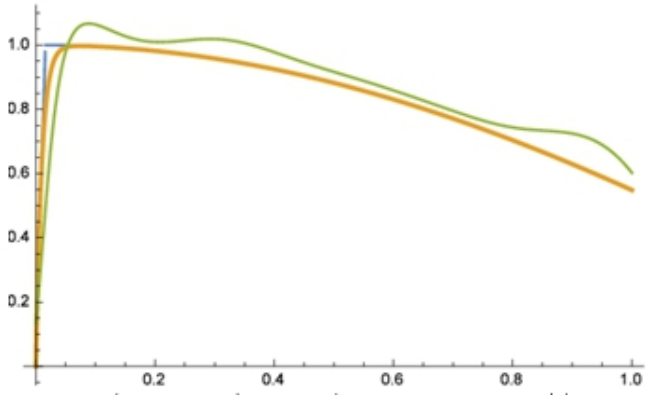


Рис.7. Нейросетевое приближение (зеленым) решения задачи (1). Модель двусторонней асимптотики с параметрами $d_t=0$, $x\in[0,1]$, $\alpha=100$, $c=0.07$

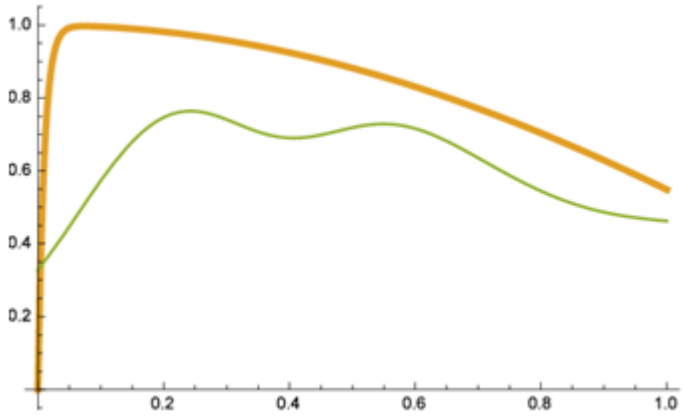


Рис.8. Нейросетевое приближение (зеленым) решения задачи (1). Базовая модель с параметрами $d_t = 0$, $x \in [0,1]$, $\alpha = 100$, $c = 0.12$

Здесь важно отметить более равномерный характер ошибки для двусторонне-асимптотической модели. Слабыми местами базовой модели являются области малых значений параметра, а также малых значений переменной в комбинации с большими α. Более наглядно этот факт демонстрируют графики приближенных и истинных решений в упомянутых случаях для моделей обоих типов (рис.5-8). Отметим, представлены наилучшие результаты в серии экспериментов.

Итак, мы можем заключить, что нейросетевой подход способен учитывать в приближенной модели дополнительные данные различных типов, приводящие к улучшению результата. Для внесения изменений в модель не требуются значимые изменения в алгоритме. Отметим, рассмотренные в статье подходы естественно обобщаются на системы обыкновенных дифференциальных уравнений, дифференциальные уравнения в частных производных и дифференциальные уравнения более высокого порядка.

Работа выполнена при финансовой поддержке Российского фонда фундаментальных исследований (грант №14-01-00660).

Литература

1. Васильев А.Н., Тархов Д.А. Нейросетевое моделирование. Принципы. Алгоритмы. Приложения. – СПб.: Изд-во Политехнического университета, 2009. – 528 с.
2. Лазовская Т.В., Тархов Д.А. Об использовании методов нейронных сетей для одного жесткого уравнения первого порядка. – Проблемы информатики в образовании, управлении, экономике и технике: Сб. статей XIV Междунар. научно-техн. конф. – Пенза: ПДЗ. – 2014. –С. 171-175.
3. Лазовская Т.В., Тархов Д.А. Варианты использования дополнительной информации при построении параметризованного нейросетевого решения жесткого дифференциального уравнения. – Научно-технические ведомости Санкт-Петербургского государственного политехнического университета. Физико-математические науки. – 2015. – № 2 (218). С. 138-147.
4. Эльсгольц Л.Э. Дифференциальные уравнения и вариационное исчисление. – Наука. – 1969.
5. Тархов Д.А. Нейросетевые модели и алгоритмы. – М.: Радиотехника. – 2014. – 348 с.
6. Васильев А.Н., Тархов Д.А. Построение приближенных нейросетевых моделей по разнородным данным. – Математическое моделирование. 2007. – Т. 19. № 12. – С. 43-51.

Васильев А.Н.[1], Тархов Д.А.[2], Шемякина Т.А.[3]

[1]Санкт-Петербургский политехнический университет Петра Великого, Санкт-Петербург, д.т.н., профессор кафедры «Высшая математика», a.n.vasilyev@gmail.com

[2]Санкт-Петербургский политехнический университет Петра Великого, Санкт-Петербург, д.т.н., профессор кафедры «Высшая математика»,dtarkhov@gmail.com

[3]Санкт-Петербургский политехнический университет Петра Великого, Санкт-Петербург, к.ф.-м.н., доцент кафедры «Высшая математика», sh_tat@mail.ru

МНОГОУРОВНЕВЫЕ МОДЕЛИ ОКРУЖАЮЩЕЙ СРЕДЫ В МЕГАПОЛИСАХ

КЛЮЧЕВЫЕ СЛОВА

Экологический мониторинг, иерархическая система, загрязнение, вредные вещества, прогнозирование, концентрация, данные, дифференциальные уравнения, искусственная нейронная сеть (ИНС), настройка ИНС, глобальная оптимизация, гибридный метод.

АННОТАЦИЯ

Рассмотрены теоретические основы построения иерархических систем мониторинга экологической обстановки, основанные на сочетании классических и нейросетевых методов. Обсуждаются особенности моделирования загрязнений воздушной среды в крупных городах с учётом влияния транспортных потоков.

Работа поддержана Российским фондом фундаментальных исследований (гранты №14-01-00660 и №14-01-00733).

Математическое моделирование таких сложных явлений, как экологические, требуется для мониторинга состояния окружающей среды, анализа и прогнозирования процессов и явлений, поиска оптимальных решений в обычных условиях и при чрезвычайных ситуациях. Для обоснованного прогнозирования часто необходимо проводить исследования на различных уровнях от локального (в масштабах отдельного объекта) до глобального (мегаполис, регион, Земля в целом). Возникает необходимость создания иерархической системы экологического мониторинга, моделирования и прогнозирования на разных уровнях. Модели сложной системы разбиваются на подсистемы и уровни, каждому из которых соответствует свой набор математических моделей.

Рассмотрим классы моделей, применяемые в [1], и возможности их сочетания с искусственными нейронными сетями.

Гауссовы модели. Среди моделей распространения примеси в атмосфере городов наиболее известными являются статистические модели распространения, основанные на функции распределения Гаусса. Данные модели предназначены для ровной подстилающей поверхности.

Концентрация загрязняющих веществ над гладкой подстилающей поверхностью в нестационарном случае описывается выражением:

$$\frac{Q}{(2\pi)^{3/2}\sigma_x\sigma_y\sigma_z}\exp\left[\frac{-(x-ut)^2}{2\sigma_x^2}\right]\exp\left[\frac{-y^2}{2\sigma_y^2}\right]\left\{\exp\left[\frac{-(x-H)^2}{2\sigma_z^2}\right]+\exp\left[\frac{-(x+H)^2}{2\sigma_z^2}\right]\right\}$$

Здесь H – конечный подъём шлейфа над землёй, u – скорость ветра на высоте H, причём ось абсцисс совпадает с направлением ветра, Q – мощность точечного источника. Учёт возможного повышения концентрации в застойных зонах вблизи зданий и сооружений проводится путём введения эмпирических коэффициентов.

Для использования соответствующих формул необходимо определить параметры, которые в них используются. В силу сложности рассматриваемых реальных процессов распространения загрязнений и воздушных течений практически это можно сделать на основе результатов наблюдений (экспериментальные данные). Возникающие при этом задачи идентификации могут решаться в рамках нейросетевой технологии с учётом регуляризующих свойств нейросетевых разложений.

Легко видеть, что при рассмотрении ряда источников формула для концентрации загрязняющих веществ является выходом нейронной сети с радиальными базисными функциями

[2-4], что позволяет использовать для подбора параметров (весов сети) всю палитру алгоритмов построения и обучения искусственных нейронных сетей.

Транспортно-диффузионная модель. В монографии [1] для описания распространения загрязнений для многослойной расчётной области сложной конфигурации предлагается использовать уравнение:

$$\frac{\partial c}{\partial t}+\operatorname{div}(c\,\mathbf{w})-\operatorname{div}(K\nabla c)-\frac{\partial}{\partial z}K_z\frac{\partial c}{\partial z}+rc=Q \quad \text{для } z>h \text{ и } c=0 \text{ при } z\leqslant h\,. \qquad (1)$$

Здесь c – концентрация примеси, K – коэффициент горизонтальной турбулентной диффузии атмосферного воздуха, K_z – коэффициент вертикальной турбулентной диффузии, $\mathbf{w}$ – скорость ветра, Q – поле эмиссии примеси, r – скорость распада примеси, функция $h(x,y)$ – описывает рельеф местности и равна высоте возвышения, если оно есть, и нулю, если возвышения в данной точке нет.

Однако, реальная постановка задачи, формализуемой уравнением (1) и соответствующими граничными условиями, состоит в том, что входящие в неё параметры не являются константами и известны с некоторой погрешностью. Для многоуровневой системы такие параметры (скорость воздушного потока, концентрация загрязнений на границе региона и т.д.) часто определяются решением задачи на другом уровне иерархии. Вычисления при большом числе наборов параметров требуют нерационально больших затрат вычислительных ресурсов.

При этом модели на каждом уровне иерархии имеют свои особенности. На самом нижнем уровне (отдельные дома, перекрёстки и отрезки улиц между перекрёстками) профиль скоростей, поле выбросов имеют большую вариабельность, связанную с обтеканием воздушным потоком отдельных зданий и влиянием отработавших газов единичных автомобилей в случае несплошного транспортного потока.

На следующем уровне (район города или небольшой город) влиянием отдельных автомобилей и обтеканием отдельных домов можно пренебречь, большее влияние приобретает суточный, недельный (будни и выходные) и годовой ритмы, а также наложение выбросов отдельных загрязняющих объектов.

Для крупных городов имеет смысл выделить следующий уровень – всего города, для которого важными являются задачи прогноза и управления экологической ситуацией в зависимости от планирования транспортной инфраструктуры, графика ремонта улиц, оптимизация транспортных потоков с помощью управления работой светофоров и информационных табло, строительство объездных дорог и т.д.

Возвращаясь к задаче (1) отметим, что в реальных ситуациях трудно определить саму границу региона и условия на ней. Обычно, вместо них, наряду с дифференциальными уравнениями, бывает задана дополнительная информация, например, в виде приближённо известных данных наблюдений (измерений с помощью датчиков, размещённых в некотором наборе точек изучаемой области). Предлагаемый нами подход позволяет объединить разнородную информацию о системе в нейросетевой модели.

Иерархическая система мониторинга должна быть построена на нескольких уровнях, каждому из которых соответствует свой набор математических моделей, описывающих распространение загрязнений и другие величины, важные для рассматриваемых задач, например, скорость ветра, температура воздуха и т.д. Данные модели обычно имеют вид или уравнений в частных производных с граничными и начальными условиями, или таблиц (баз данных) с результатами наблюдений (измерения интересующих параметров). Авторами разработан обобщённый унифицированный подход к построению иерархии нейросетевых моделей для каждого из случаев: «уравнения» или «наборы данных» – и в смешанной гетерогенной ситуации: «уравнения+данные».

Построение моделей на каждом из указанных выше уровней имеет свои особенности. При моделировании воздушных течений в уличных «каньонах» учитываются только здания, расположенные вблизи источника. Такие же предпосылки вводятся при решении уравнений термической гидродинамики и так называемых транспортно-диффузионных уравнений. Достаточно широкие площади и современные магистрали при моделировании требуют усреднения влияния отдельных зданий. Расчёт загрязнений в более высоких слоях воздушной среды над мегаполисом разумно проводить, считая источники загрязнений распределёнными, а ландшафт – сглаженным.

Моделирование распределения загрязнений на каждом из этих уровней и течений в

каньонах на основе решения уравнений термической гидродинамики сопряжено с известными принципиальными трудностями для всех подобных моделей – заданием входных параметров:

– условий на границах;

– начальных значений, которые должны зависеть от временного интервала, для которого производится расчёт.

Очевидно, что граничными условиями для одних подсистем являются решения для других (примыкающих к ним), что делает практически невозможным их последовательное решение. Решение полной замкнутой системы требует суперкомпьютерных мощностей, что затрудняет достаточно точное предсказательное моделирование. Выход видится в параллельном построении нейросетевых моделей [2-4] для каждой подсистемы с активным использованием параметрических нейросетевых моделей, пересылаемых от одного вычислительного узла к другому [5-12].

Реальные начальные значения также задать невозможно, вместо них для построения адекватных моделей предлагается использовать результаты измерений. Подобная методология достаточно хорошо отработана и протестирована на модельных задачах [2-10].

Отдельные проблемы возникают при моделировании воздушных потоков, от которых зависит скорость распространения загрязнений. Метеорологические модели в условиях больших городов включают, в том числе, зависимость от времени и от метеоусловий. Кроме того, проблема, возникающая при решении уравнений переноса, заключается в необходимости задавать коэффициенты, зависящие от энергии турбулентных движений – функции многих величин. Адекватность приводимых моделей реальным условиям во многом определяется выбором значений эмпирических констант. Для описания формирования полей концентраций примеси используется полуэмпирическое уравнение переноса и диффузии.

В [1] предлагается следующий алгоритм формирования ветровых полей:

1) Задаётся начальное значение на высоте анемометра;

2) Рассчитывается вертикальный профиль;

3) Решается уравнение $\mathrm{div}(\mathbf{w}h)=0$, где h - толщина горизонтального слоя в данной точке;

4) Производится итерационное сглаживание поля ветра.

Основная проблема при реализации данного алгоритма состоит в том, что расчёты для всего мегаполиса (или даже одного его района) слишком трудоёмки и требуют пересчёта при изменении внешних ветровых потоков, а разбиение на подсистемы не даёт возможность проводить расчёты по отдельности.

Более перспективным представляется обучение параметрических нейросетевых моделей для типичных конфигураций систем зданий с дальнейшим замыканием системы соответствующих нейросетей и её непрерывное дообучение с учётом меняющихся метеоусловий и данных натурных измерений.

При этом результаты вычислений по указанному выше алгоритму могут быть использованы в качестве дополнительных данных для ускорения обучения параметризованных нейросетевых моделей [2-12].

Несомненно, большое значение имело бы создание базы данных, алгоритмов и программ, информационных (в частности, нейросетевых) моделей. Сюда относится и построение нейросетевых моделей типовых городских элементов – шаблоны «Магистраль», «Каньон», «Перекресток», «Двор», «Автостоянка» и др. для загрязняющих веществ разного типа.

В Институте математического моделирования РАН в течение ряда лет разрабатывались вопросы, связанные с построением моделей распространения примесей в турбулентной атмосфере над местностью, имеющей сложный рельеф, и в условиях городской застройки. Для численного моделирования процессов переноса в транспортно-диффузионной модели малых примесей в ветровом поле был разработан специальный метод, являющийся развитием классического метода частиц. Итогом этих работ стало построение транспортно-диффузионной модели и создание на ее основе программного комплекса «TIMES», который обеспечивает решение системы уравнений, описывающей процесс распространения загрязняющих веществ в городе. Важное место в создании программного комплекса занимает физическая модель атмосферы, которая влияет на построение поля ветра и на описание адекватных процессов.

Предлагается развить данный пакет в иерархическую систему мониторинга и прогнозирования состояния окружающей среды, которая должна решать следующие задачи:

1) Более точный прогноз загрязнений окружающей среды с доступом к этим данным

широким кругам населения;

2) Прогноз возможных экологических катастроф, путей их предотвращения и минимизации их неблагоприятных последствий;

3) Выработка оптимальных способов управления состоянием окружающей среды на разных пространственных (локальный, региональный, глобальный) и временных (часы, сутки, времена года, десятилетия) масштабах и различными субъектами такого управления. При этом субъекты более высокого уровня могут управлять экологическим состоянием не только непосредственно (природоохранные мероприятия), но и опосредованно (изменение законодательной базы и стимулирование субъектов нижнего уровня);

4) Прогноз и минимизация последствий чрезвычайных ситуаций, создание системы автоматического оповещения населения, которое может подвергнуться риску различного уровня, своевременная эвакуация населения, для которого эта мера является оптимальной;

5) Объективная оценка ущерба окружающей среде и здоровью граждан и адекватное наказание для виновников такого ущерба.

Литература

1. Калиткин Н.Н. и др. Математические модели природы и общества. – М.: Физматлит, 2005. – 360с.
2. Васильев А.Н., Тархов Д.А. Нейросетевое моделирование. Принципы. Алгоритмы. Приложения. – СПб.: изд-во СПбГПУ, 2009. – 528с.
3. Васильев А.Н., Тархов Д.А Нейросетевые методы и алгоритмы математического моделирования. – СПб.: Изд-во Политехн. ун-та, 2014. – 582 с.
4. Тархов Д.А. Нейросетевые модели и алгоритмы. – М.: Радиотехника, 2014. – 348 с.
5. Васильев А.Н., Идрисова Д.И., Романова А. Г., Тархов Д.А. Применение параметрических нейросетевых моделей к построению системы мониторинга загрязнений атмосферы промышленными выбросами и условий труда в строительстве. Материалы IX Международной конференции по неравновесным процессам в соплах и струях (NPNJ'2012). – Алушта. – М.: Изд-во МАИ-ПРИНТ, 2012. – С. 468 – 470.
6. Романова А.Г., Тархов Д.А, Шемякина Т.А. Нейросетевое моделирование в экологии. Нейрокомпьютеры. Разработка. Применение. – М.: ЗАО изд-во «Радиотехника». №2, 2014, С. 16–21.
7. Васильев А.Н., Осипов В.П., Тархов Д.А. Унифицированный процесс моделирования физико-технических объектов с распределенными параметрами// Научно-технические ведомости СПбГПУ. Физ.-мат. науки. – 2010. – №3(104). – С.39-52.
8. N. U. Kainov, D. A. Tarkhov, T. A. Shemyakina. Application of neural network modeling to identification and prediction problems in ecology data analysis for metallurgy and welding industry// Nonlinear Phenomena in Complex Systems. vol. 17, no. 1, pp. 57-63 (2014).
9. Романова А.Г., Тархов Д.А, Шемякина Т.А. О применении нейросетевых моделей в экологии «Соврем. Информац. технологии и ИТ-образование». – М.: ИНТУИТ.РУ, 2013, Т.1. № 1(8). С.534 –539.
10. Романова А.Г., Тархов Д.А, Шемякина Т.А. Нейросетевое моделирование в обратных задачах экологического мониторинга Материалы XVIII Междунар. конф. по выч. механике и совр. приклад. програм. системам (ВМСППС'2013) Алушта. – М.: Изд-во МАИ-ПРИНТ, 2013, С.228 –231.
11. Васильев А.Н., Лазовская Т.В., Тархов Д.А. Иерархические нейросетевые модели в случае вычислительных комплексов высокой производительности// Системы компьютерной математики и их приложения. 2015. № 16. С. 70-72.
12. Лазовская Т.В., Тархов Д.А. Новые подходы к построению параметризованного нейросетевого решения жёсткого дифференциального уравнения// Научно-технические ведомости Санкт-Петербургского государственного политехнического университета. Физико-математические науки. 2015. № 2 (218). С. 138-147.

Васильев А.Н.[1], Тархов Д.А.[2], Шемякина Т.А.[3]

[1]Санкт-Петербургский политехнический университет Петра Великого, Санкт-Петербург, д.т.н., профессор кафедры «Высшая математика», a.n.vasilyev@gmail.com

[2]Санкт-Петербургский политехнический университет Петра Великого, Санкт-Петербург, д.т.н., профессор кафедры «Высшая математика», dtarkhov@gmail.com

[3]Санкт-Петербургский политехнический университет Петра Великого, Санкт-Петербург, к.ф.-м.н., доцент кафедры «Высшая математика», sh_tat@mail.ru

МОДЕЛЬ НЕИЗОТЕРМИЧЕСКОГО ХИМИЧЕСКОГО РЕАКТОРА НА ОСНОВЕ ПАРАМЕТРИЧЕСКИХ НЕЙРОННЫХ СЕТЕЙ. ГИБРИДНЫЙ МЕТОД

КЛЮЧЕВЫЕ СЛОВА

Неизотермический химический реактор, начально-краевая задача, уравнение диффузии, нейросетевое моделирование, искусственная нейронная сеть (ИНС), настройка ИНС, глобальная оптимизация, гибридный метод.

АННОТАЦИЯ

Статья посвящена построению приближённого параметрического нейросетевого решения краевой задачи для обыкновенного дифференциального уравнения, описывающего процессы в неизотермическом химическом реакторе. Изучаются свойства приближённого нейросетевого решения задачи в области изменения параметра. Анализируется влияние дополнительной информации о решении, заданной в виде приближённых данных точечных измерений или вычислений, на результат обучения нейронной сети.

В работе рассматривается макрокинетическая модель неизотермического химического реактора [1]. Процессы теплового взрыва, самовоспламенения, зажигания, распространения волн горения и т.д., происходящие в реакторе, описываются системой квазилинейных (параболических, эллиптических) уравнений в частных производных или обыкновенных дифференциальных уравнений.

В безразмерных переменных модель химического реактора описывается системой квазилинейных уравнений теплопроводности и диффузии:

$$\begin{cases} \dfrac{\partial \theta}{\partial t} = \dfrac{1}{\delta}\Delta\theta + \varphi(a)\exp\dfrac{\theta}{1+\beta\theta} \\ \dfrac{\partial a}{\partial t} = \dfrac{1}{\delta}\Delta y - \gamma\varphi(a)\exp\dfrac{\theta}{1+\beta\theta} \end{cases} \tag{1}$$

где a – относительная концентрация реагирующей компоненты смеси;

θ – изменение относительной температуры (безразмерная температура);

x – относительное изменение длины реактора (безразмерная координата);

Δ – оператор Лапласа в безразмерных координатах;

γ – безразмерная энергия активации;

β – безразмерный коэффициент теплопередачи;

δ – безразмерный параметр, называемый числом Франк-Каменецкого;

$\varphi(a)$ – зависимость, характеризующая скорость реакции от концентрации.

В работах авторов [2-4] исследовалась нейросетевая модель катализатора: анализ баланса тепла и массы в плоской грануле пористого катализатора при каталитической реакции приводил к краевой задаче для обыкновенного дифференциального уравнения второго порядка, которое является частным случаем стационарной системы дифференциальных уравнений (1)

$$\frac{d^2\theta}{dx^2} = \alpha(1+\theta)\exp[-\frac{\gamma\beta\theta}{1-\beta\theta}], \qquad \frac{d\theta}{dx}(0) = 0, \quad \theta(1) = 0. \tag{2}$$

В данной работе продолжается исследование частных вариантов модели химических

реакторов – системы уравнений (1) – с помощью нейросетевого моделирования. Далее изучается стационарная задача о тепловом взрыве в плоскопараллельном случае [1] в предположении, что реакция является одностадийной, необратимой, не сопровождается фазовыми переходами, протекает в неподвижной среде. Эта задача интересна тем, что известно точное решение, область существования решения и значения параметра, при которых решение задачи не существует ($\delta > \delta_{кр} \approx 0.878458$), – рассматриваем её в качестве модельной для алгоритмов обучения нейронных сетей.

Ищется приближённое решение краевой задачи:

$$\frac{d^2\theta}{dx^2} + \delta \exp(\theta) = 0, \quad \frac{d\theta}{dx}(0) = 0, \quad \theta(1) = 0 \tag{3}$$

в виде нейросетевого разложения вида

$$\theta(x, \delta) = \sum_{k=1}^{N} c_k \exp(-\alpha_k (x - \tilde{x}_k)^2) th(\beta_k (\delta - \tilde{\delta}_k)^2),$$

параметры которого находятся в процессе поэтапного обучения сети на основе минимизации функционала ошибки $J(\theta) = J_0 + \lambda_1 J_1 + \lambda_2 J_2$, состоящего из невязок в удовлетворении уравнению $J_0 = \sum_{j=1}^{M} \left(\frac{d^2\theta}{dx^2} + \delta \exp(\theta) \right)^2 (x_j, \delta_j)$ краевым условиям $J_1 = \sum_{j=1}^{M} \left(\left(\frac{d\theta}{dx}(0, \delta_j) \right)^2 + (\theta(1, \delta_j)) \right)^2$ и *дополнительным условиям* в форме $J_2 = \sum_{j=1}^{M} \left(\theta(x_i', \delta') - \theta_i \right)^2$ где x_j – периодически перегенерируемые тестовые точки из [0;1]; δ_j – тестовые точки из $[\delta_{min}; \delta_{max}]$; x_i' – фиксированные точки, где известны значения искомой функции $\theta(x_i', \delta') = \theta_i$; $\lambda_1 \lambda_2$ – положительные штрафные параметры. Нейросетевой подход применялся с учётом (гибридный метод) и без учёта дополнительной информации, полученной на основе измерений или грубого численного решения в фиксированном наборе точек, учитываемой в слагаемом $\lambda_2 J_2$.

Минимизация функционала $J(\theta)$ проводилась методом **RProp** с перегенерацией *M*=50 и *M*=100 тестовых точек каждые 5 шагов работы алгоритма; делалось 200 перегенераций.

Для применения гибридного метода строилось приближённое поточечное решение при значениях параметра: $\delta' = 0.5, \delta' = 0.8, \delta' = 0.87$, которое рассматривалось как дополнительные данные для решения задачи.

Мы использовали сети из *N*=30, *N*=50, *N*=100 нейроэлементов. Мы выбирали следующие интервалы изменения параметра $\delta \in [\delta_{min}; \delta_{max}]$: [0,4;1] и [0.85;0.9].

Мы провели два численных эксперимента, результаты которых приведены в виде графиков, где представлены нейросетевое решение $\theta(x, \delta)$ с сетью из *N* элементов (в работе приведены данные для *N*=30), аналитическое решение $\hat{\theta}(x, \delta)$ и функция $\tilde{\theta}(x, \delta)$, найденная в пакете «*Mathematica*». При этом параметр δ может принимать значения, при которых точное решение отсутствует.

В ***первом эксперименте*** мы применяем нейросетевой подход при значениях параметра $\delta \in [0.4; 1]$. При этом рассматриваем случаи:

а) без дополнительных условий (БДУ),

б) с дополнительным условием (ДУ) при значении параметра $\delta' = 0.5$,

в) с дополнительным условием (ДУ) при значении параметра $\delta' = 0.8$.

Графики Рис.1 показывают нейросетевое решение $\theta(x, \delta)$ и аналитическое решение $\hat{\theta}(x, \delta)$ при значении параметра $\delta = \delta_{min} = 0.4$ в случаях а), б) и в).

Графики Рис.2 показывают нейросетевое решение $\theta(x, \delta)$ и функцию $\tilde{\theta}(x, \delta)$, полученную с помощью пакета «*Mathematica*», при значении параметра $\delta = \delta_{max} = 1.0$ в случаях а), б) и в).

Рис.1 показывает графики функций при малом значении параметра $\delta=\delta_{min}=0.4$. Результат получается лучше, чем на Рис.2. Погрешность приближения меньше для варианта с дополнительным условием б) при $\delta'=0.5$.

Рис.2 показывает графики функций с достаточно большим параметром $\delta=\delta_{max}=1.0$. Точное решение задачи $\hat{\theta}(x,\delta)$ не существует при этом значении параметра. Мы получаем приближенную функцию $\tilde{\theta}(x,\delta)$. Результат плохой для всех вариантов а), б), в), хуже, чем на Рис.1. В этом случае пакет «*Mathematica*» выдает функцию $\tilde{\theta}(x,\delta)$, которая не удовлетворяет граничному условию на правом конце отрезка. В то же время нейронная сеть $\theta(x,\delta)$ даёт более сбалансированную функцию. Мы отмечаем, что можно управлять точностью удовлетворения граничных условий задачи с помощью штрафных множителей.

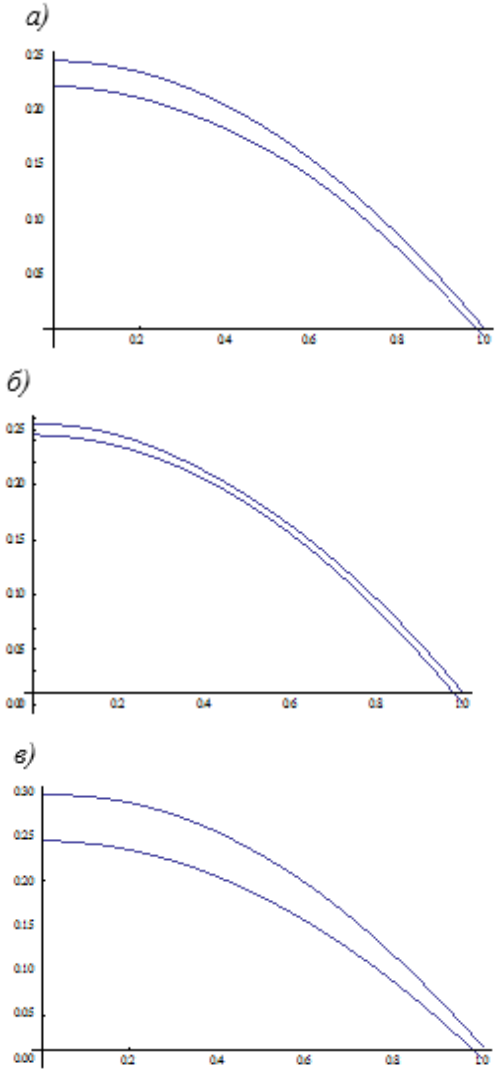


Рис.1. Графики решений

$\theta(x,\delta),\ \hat{\theta}(x,\delta),\ \delta=\delta_{\min}=0.4$

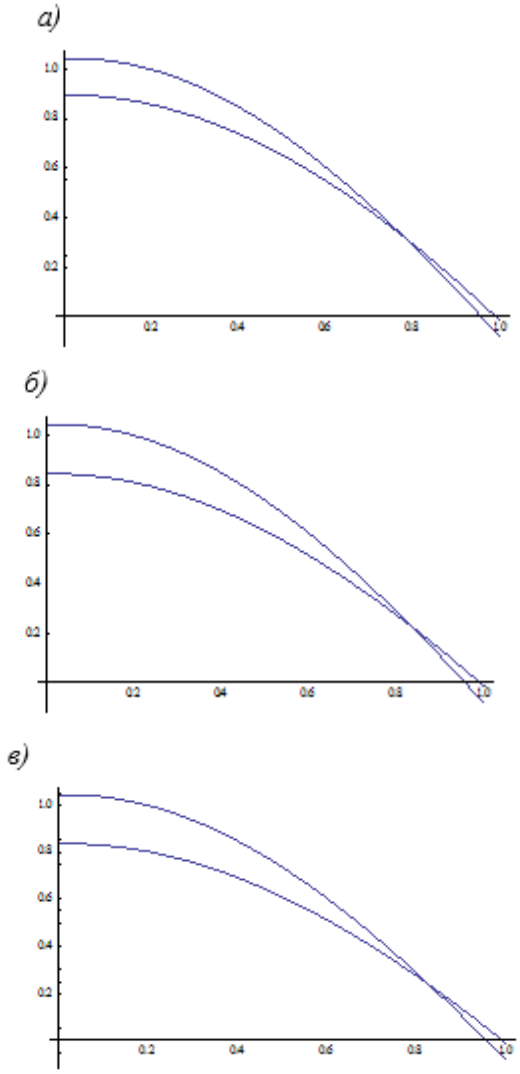


Рис.2. Графики решений

$\theta(x,\delta),\ \tilde{\theta}(x,\delta),\ \delta=\delta_{\max}=1$

Пусть погрешность (невязка) – это разность между левой и правой частью дифференциального уравнения из постановки (3), вычисленная в определённой точке, например, при *x*=0.5. Таким образом, нейросетевое решение $\theta(x,\delta)$, $\delta\in[0.4;1]$ удовлетворяет уравнению (3) с некоторой погрешностью. Рис.3 показывает график погрешности (невязки) $\Delta(\theta(x,\delta))$ в зависимости от параметра $\delta\in[0.4;1]$. Результаты вычислений приведены для случаев а), б) и в).

Погрешность $\Delta(\theta(x,\delta))$ растет, когда параметр δ принимает значения больше

критического: $\delta > \delta_{кр} \approx 0.878458$. Из книги [1] известно, что в этом случае точного решения задачи $\hat{\delta}$ не существует. Тогда построенная нейросеть не удовлетворяет уравнению. Резкий рост ошибки указывает на возможное отсутствие точного решения.

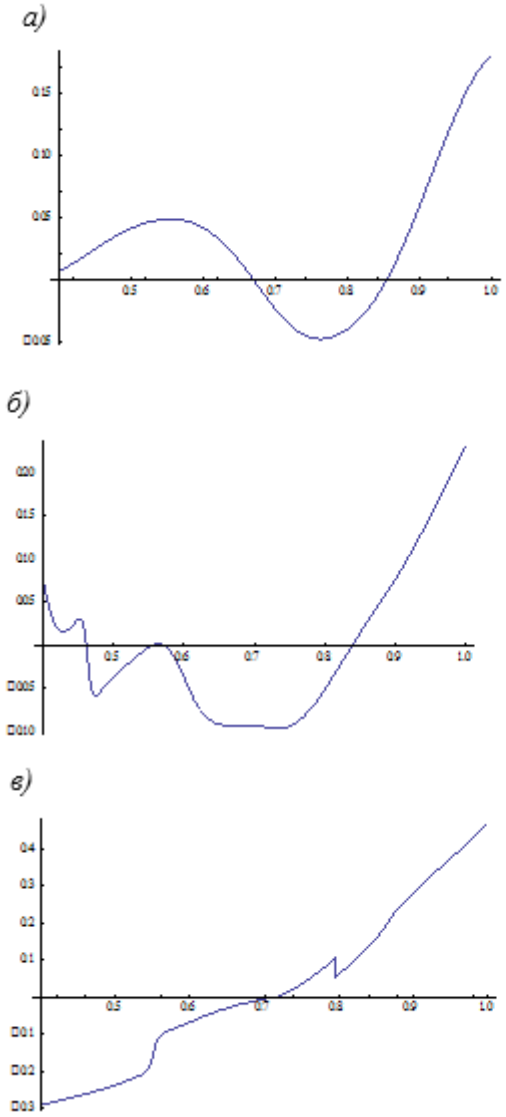


Рис.3. Графики погрешности

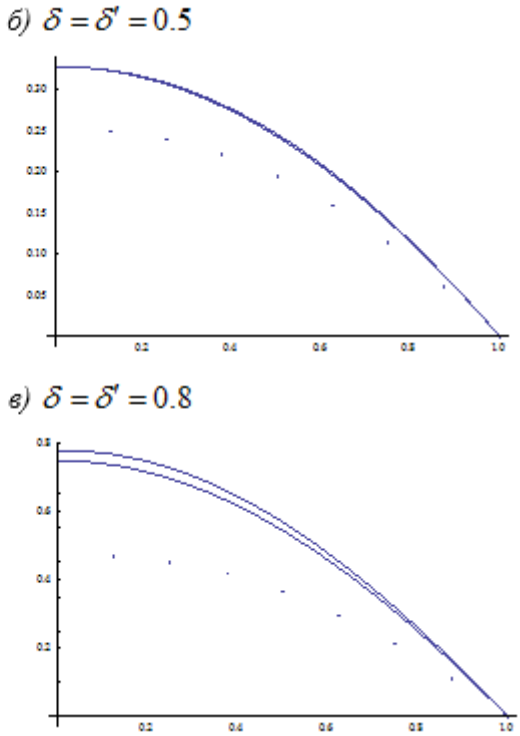


Рис.4. Графики решений

$\Delta(\theta(x,\delta)), \delta \in [0.4;1.0], x = 0.5$ $\theta(x,\delta),\ \hat{\theta}(x,\delta),\ \theta_i$

Рис.4 показывает график нейросетевого решения $\theta(x,\delta)$, график аналитического решения $\hat{\theta}(x,\delta)$ и график поточечного решения θ_i. Поточечное решение изображено на графике точечной линией. Мы задаём параметр $\delta \in [0.4;1]$, совпадающим с параметром δ' для дополнительного условия: б) $\delta = 0.5$ или в) $\delta = 0.8$. В этом случае нейросеть $\theta(x,\delta)$ практически совпадает с аналитическим решением $\hat{\theta}(x,\delta)$. Поточечное решение используем в качестве дополнительных данных. Оно значительно отличается от аналитического решения для всех вариантов Рис.4.

Во ***втором эксперименте*** мы сужаем интервал изменения параметра и применяем нейросетевой подход для значений параметра $\delta \in [0.85;0.9]$. Значения параметра δ близки при этом к критическому значению $\delta = \delta_{кр} \approx 0.878452$. Как и ранее, рассматриваем три случая:

а) без дополнительных условий (БДУ),

б) с дополнительным условием (ДУ) при значении параметра $\delta' = 0.5$,

в) с дополнительным условием (ДУ) при значении параметра $\delta' = 0.87$.

Графики Рис.5 показывают нейросетевое решение $\theta(x,\delta)$ и аналитическое решение $\hat{\theta}(x,\delta)$ при значении параметра $\delta = \delta_{\min} = 0.85$ в случаях а), б) и в).

Графики Рис.6 показывают нейросетевое решение $\theta(x,\delta)$ и функцию $\tilde{\theta}(x,\delta)$, полученную с помощью пакета «*Mathematica*», при значении параметра $\delta = \delta_{\max} = 0.9$ в случаях а), б) и в).

Рис.5 показывает графики функций при значении параметра $\delta = \delta_{\min} = 0.85$. Результат получается лучше, чем на Рис.6. Погрешность приближения меньше для варианта с дополнительным условием б) при $\delta' = 0.5$.

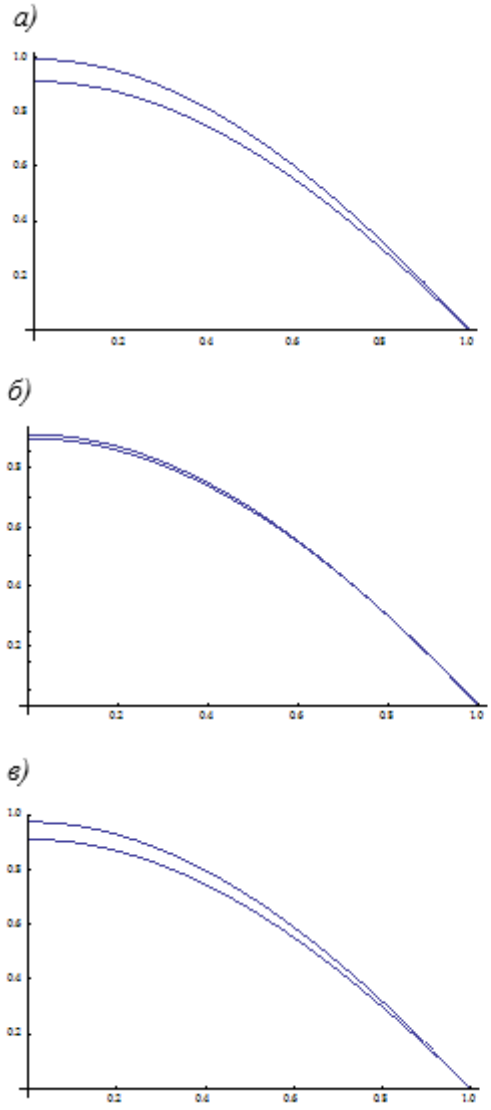


Рис.5. Графики решений

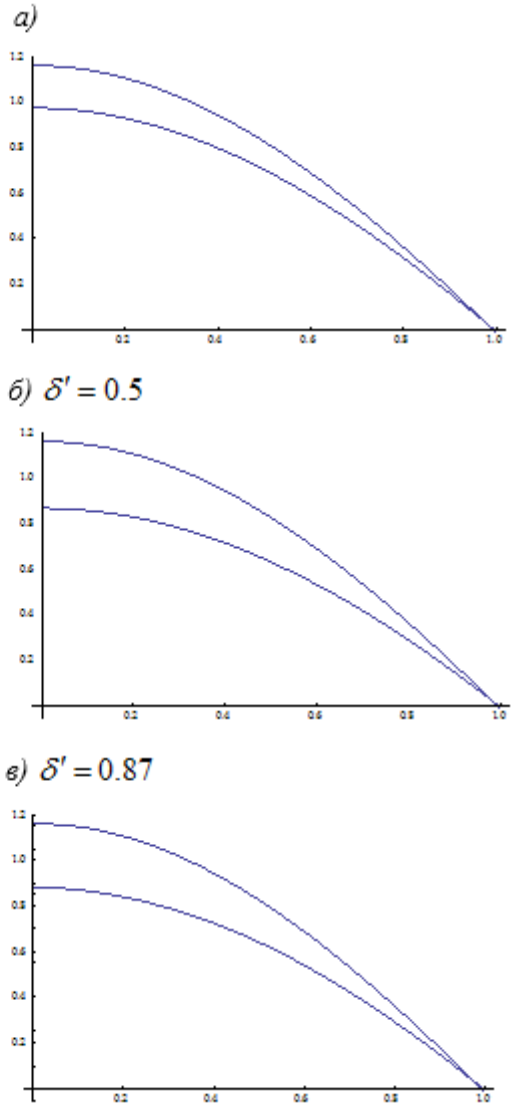


Рис.6. Графики решений

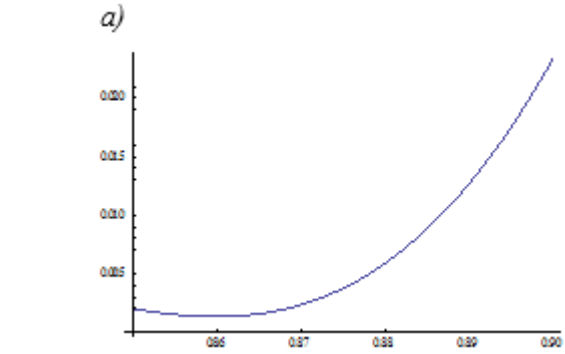
$\theta(x,\delta),\ \hat{\theta}(x,\delta),\ \delta = \delta_{\min} = 0.85$ $\qquad$ $\theta(x,\delta),\ \tilde{\theta}(x,\delta),\ \delta = \delta_{\max} = 0.9$

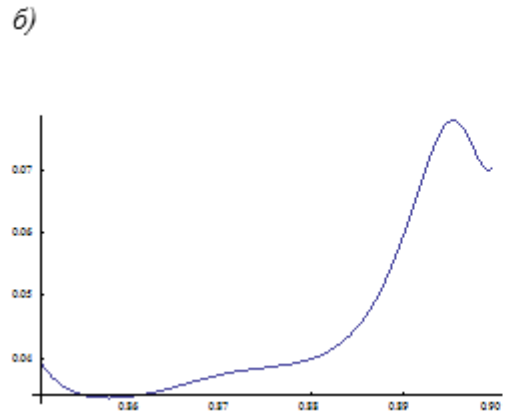
Рис.6 показывает графики функций с достаточно большим параметром $\delta = \delta_{\max} = 0.9$. Точное решение задачи не существует при этом значении параметра. В этом случае мы получаем приближенную функцию $\tilde{\theta}(x,\delta)$. Результат получается плохим для всех вариантов, хуже, чем на Рис.5. В этом случае пакет «*Mathematica*» выдает функцию $\tilde{\theta}(x,\delta)$, которая не удовлетворяет

граничному условию на правом конце отрезка. В то же время нейронная сеть $\theta(x,\delta)$ даёт более сбалансированную функцию. Мы отмечаем, что можно управлять точностью удовлетворения граничных условий задачи с помощью штрафных множителей.

Рис.7 показывает график погрешности (невязки) $\Delta(\theta(x,\delta))$ в зависимости от параметра $\delta\in[0.85;0.9]$. Рассматривается окрестность контрольной точки, в качестве которой выбирается x=0.5. Нейросетевое решение $\theta(x,\delta)$, $\delta\in[0.85;0.9]$ удовлетворяет уравнению из (3) с некоторой погрешностью. Результаты вычислений указаны для случаев а), б) и в).

Погрешность $\Delta(\theta(x,\delta))$ растет, если параметр принимает значения больше критического: $\delta>\delta_{кр}\approx 0.878452$. Известно, что в этом случае точного решения задачи не существует. Резкий рост ошибки указывает на возможное отсутствие точного решения.





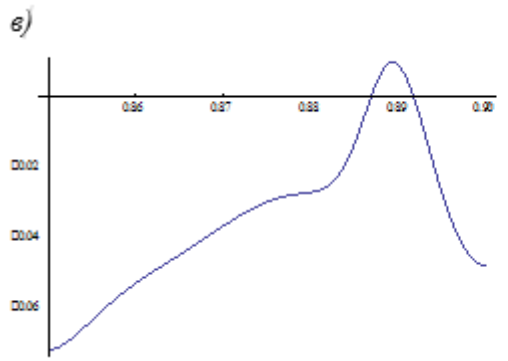
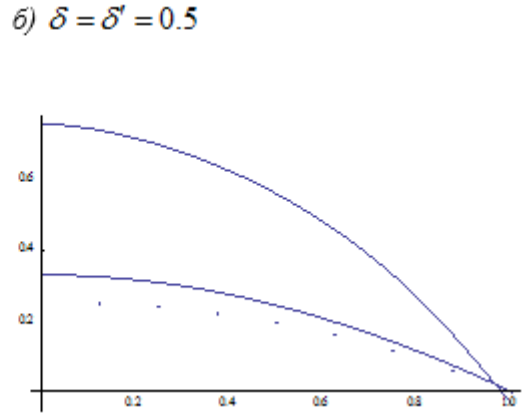


Рис.7. Графики погрешности



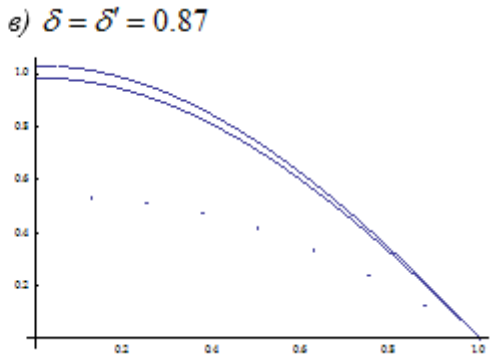


Рис.8. Графики решений

$\Delta(\theta(x,\delta)),\delta\in[0.85;0.9],x=0.5$ $\theta(x,\delta),\ \hat{\theta}(x,\delta),\ \theta_i$

Рис.8 показывает график нейросетевого решения $\theta(x,\delta)$, график аналитического решения $\hat{\theta}(x,\delta)$ и график поточечного решения θ_i. Поточечное решение изображено на графике точечной линией. Параметр $\delta\in[0.85;0.9]$ задается совпадающим с параметром δ' для дополнительного условия: б) $\delta=0.5$ или в) $\delta=0.87$.

Рассмотрим случай в) $\delta = 0.87$. Параметр принадлежит рассматриваемой области $[0.85; 0.9]$. Нейросеть $\theta(x, \delta)$ практически совпадает с аналитическим решением $\hat{\theta}(x, \delta)$. Поточечное решение используем в качестве дополнительных данных. Оно значительно отличается от аналитического решения.

Рассмотрим случай б) $\delta = 0.5$. Параметр δ выходит за интервал рассмотрения $[0.85; 0.9]$. Нейросеть $\theta(x, \delta)$ отличается от аналитического решения $\hat{\theta}(x, \delta)$. При этом отмечается резкий рост ошибки после критического значения параметра, что может служить признаком отсутствия точного решения задачи.

Таблицы 1 и 2 показывают относительные погрешности вычисления для двух экспериментов. Относительная погрешность достаточно мала: в пределах $10^{-3}\%$.

Таблица 1.

Относительная погрешность (10^{-2} %), $N = 30$			
$\delta \in [0.4; 1.0]$	БДУ	ДУ, $\delta' = 0.5$	ДУ, $\delta' = 0.8$
$\theta(x,\delta),\ \hat{\theta}(x,\delta)$ $\delta = \delta_{\min} = 0.4$	0.102	0.041	0.204
$\theta(x,\delta),\ \tilde{\theta}(x,\delta)$ $\delta = \delta_{\max} = 1.0$	0.143	0.190	0.190
$\theta(x,\delta),\ \hat{\theta}(x,\delta)$ $\delta = 0.5$	–	0	0
$\theta(x,\delta),\ \hat{\theta}(x,\delta)$ $\delta = 0.8$	–	0	0.267

Таблица 2.

Относительная погрешность (10^{-2} %), $N = 30$			
$\delta \in [0.85; 0.9]$	БДУ	ДУ, $\delta' = 0.5$	ДУ, $\delta' = 0.87$
$\theta(x,\delta),\ \hat{\theta}(x,\delta)$ $\delta = \delta_{\min} = 0.85$	0.110	0	0.022
$\theta(x,\delta),\ \tilde{\theta}(x,\delta)$ $\delta = \delta_{\max} = 0.9$	0.174	0.259	0.217
$\theta(x,\delta),\ \hat{\theta}(x,\delta)$ $\delta = 0.5$	–	1.273	–
$\theta(x,\delta),\ \hat{\theta}(x,\delta)$ $\delta = 0.87$	–	–	0.048

Численные эксперименты показали:

- методы построения нейросетевых моделей сложных систем существенно улучшают нейросетевое решение, если используется дополнительная информация. Её можно получить, например, с помощью приближений на основе традиционных численных методов (даже не очень точных);
- нейронная сеть позволяет построить приближённое решение параметрической задачи. При этом параметр может принимать значения, при которых точного решения задачи не существует. Эффект отсутствия решения проявляется резким ростом ошибки удовлетворения уравнению;
- в сложных задачах, где неизвестен интервал существования решения, нейронные сети позволяют уточнить приближённое решение. Для этого используются дополнительные данные – измерения или численное решение при одном значении параметра. При этом эффект уточнения теряется при приближении к критическому значению параметра.

Работа поддержана Российским фондом фундаментальных исследований (гранты №14-01-00660 и №14-01-00733).

Литература

1. Худяев С.И. Пороговые явления в нелинейных уравнениях. –М.: Наука, 2003. – 268 с.
2. Васильев А.Н., Тархов Д.А., Шемякина Т.А. Нейросетевая модель решения задачи о катализаторе. Гибридный метод.// Проблемы информатики в образовании, управлении, экономике и технике: сб. статей XIV Междунар. научно-техн. конф. – Пенза: Приволжский Дом знаний, 2014. – С.58 – 62.
3. Васильев А.Н., Тархов Д.А., Шемякина Т.А. Гибридный метод построения параметрической нейросетевой модели катализатора.//«Современные информац. технологии и ИТ-образование». – М.: ИНТУИТ.РУ, 2014. Т.1. №1(9). С.476–484.
4. A.N. Vasilyev, D.A. Tarkhov Mathematical Models of Complex Systems on the Basis of Artificial Neural Networks// Nonlinear Phenomena in Complex Systems. - vol. 17, no.3, pp. 327-335 (2014).
5. Васильев А.Н., Тархов Д.А Нейросетевые методы и алгоритмы математического моделирования. – СПб.: Изд-во Политехн. ун-та, 2014. – 582 с.

Васильев А.Н.[1], Тархов Д.А.[2], Шемякина Т.А.[3]

[1]Санкт-Петербургский политехнический университет Петра Великого, Санкт-Петербург, д.т.н., профессор кафедры «Высшая математика», a.n.vasilyev@gmail.com

[2]Санкт-Петербургский политехнический университет Петра Великого, Санкт-Петербург, д.т.н., профессор кафедры «Высшая математика»,dtarkhov@gmail.com

[3]Санкт-Петербургский политехнический университет Петра Великого, Санкт-Петербург, к.ф.-м.н., доцент кафедры «Высшая математика», sh_tat@mail.ru

МЕЗО-УРОВНЕВАЯ НЕЙРОСЕТЕВАЯ МОДЕЛЬ ЗАГРЯЗНЕНИЯ АТМОСФЕРНОГО ВОЗДУХА САНКТ-ПЕТЕРБУРГА ПО ДАННЫМ МОНИТОРИНГА

КЛЮЧЕВЫЕ СЛОВА

Экологический мониторинг, иерархическая система, загрязнение, вредные вещества, прогнозирование, концентрация, данные, дифференциальные уравнения, искусственная нейронная сеть (ИНС), настройка ИНС, глобальная оптимизация.

АННОТАЦИЯ

Нейросетевой подход используется для построения многоуровневых моделей загрязнения воздуха в мегаполисах. Данные модели могут быть использованы для оценки и минимизации воздействия транспортных потоков на окружающую среду в больших городах. Построена нейросетевая модель мезо-уровня загрязнённости атмосферного воздуха Санкт-Петербурга на основании данных, полученных автоматизированным мониторингом. Рассчитаны диаграммы распределения загрязнения по районам города, отмечены самые загрязненные части города и выявлены основные источники загрязнений, учтена зависимость уровня загрязнения разных районов как функции от направления и скорости ветра.

Работа поддержана Российским фондом фундаментальных исследований (гранты №14-01-00660 и №14-01-00733).

Проведены оценка состояния атмосферного воздуха в Санкт-Петербурге и исследование влияния скорости и направления ветра на распределения загрязнения в атмосферном воздухе по городу Санкт-Петербургу на основании данных, полученных от 21 стационарной станции Автоматизированной системы мониторинга атмосферного воздуха Санкт-Петербурга. Для этого использованы показания скорости и направления ветра с 1 по 31 июля 2014 года, а также общий уровень загрязнения атмосферы по данным мониторинга.

В Санкт-Петербурге основную часть загрязнений воздушной среды дает автотранспорт, поэтому особенно сильно загрязнен воздух вблизи магистралей с интенсивным движением автомашин.

Перенос выброса в атмосферном воздухе, как правило, относится к гауссову типу переноса. Согласно гауссовой модели изменение концентрации примеси в атмосфере подчиняется нормальному закону распределения. Используя принцип суперпозиции и заменяя интеграл кубатурной формулой, приходим к выражению

$$q_n(t,V_x,V_y)=\sum_{i=0}^{n} C_i \frac{Q\exp\left\{-\left[\frac{(x_i-x_0-V_x t)^2}{2(\sigma_x)^2 t}+\frac{(y_i-y_0-V_y t)^2}{2(\sigma_y)^2 t}+\frac{(z_i-z_0)^2}{2(\sigma_z)^2 t}\right]\right\}}{(\sqrt{2\pi t})^3\sigma_x\sigma_y\sigma_z},$$

где $x_{0,}y_{0,}z_0$ – координаты источника примеси; Q - мощность источника; $(V_x,V_y,0)$ – компоненты вектора скорости ветра; $\sigma_x,\sigma_y,\sigma_z$ – средние квадратичные отклонения частиц примеси в момент времени t соответственно вдоль координатных осей OX, OY, OZ; C_i – числовой коэффициент, x_i, y_i, z_i, $i=1=0,1,\ldots,n$ – узлы интегрирования. Очевидно, что эта модель соответствует нейронным сетям с радиальными базисными функциями в случае

использования в качестве базисной функции гауссиана [1, 2].

На основе данных мониторинга были построены нейронные сети с разным количеством нейронов: $n=5; n=10; n=15; n=20$, – с применением ***метода RProp*** и комбинации ***метода облака*** из 3 частиц ($n_1=3$) и ***метода RProp*** [1, 2].

Получены диаграммы, отражающие зависимость уровня загрязнения в разных районах города от векторного параметра – скорости и направления ветра. Наилучшие результаты получились в случае применения комбинации методов при $n=5$ и $n_1=3$.

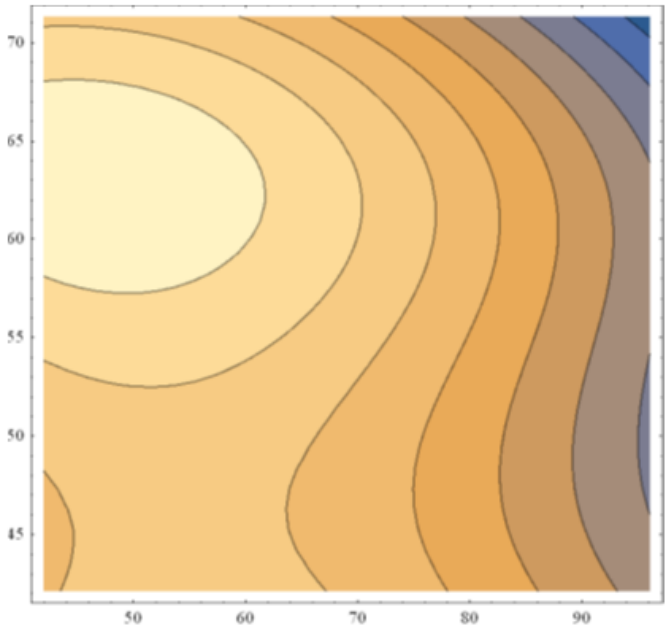


Диаграмма 1

Данная диаграмма показывает уровень загрязнения атмосферы в разных районах Санкт-Петербурга в безветренную погоду. Из диаграммы можно сделать вывод, что наиболее загрязненным является атмосферный воздух над Финским заливом. Это объясняется диффузией загрязнения в атмосферном воздухе с других районов города.

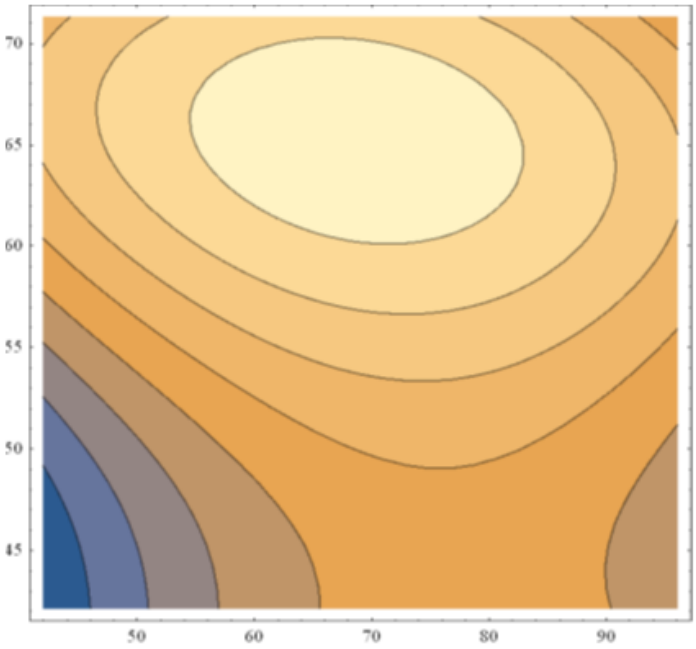


Диаграмма 2

На диаграмме 2 представлено распределение загрязнения атмосферы по городу Санкт-

Петербургу при южном ветре со скоростью – 2 м/с $(V_x=0, V_y=2)$. Общий уровень загрязнения перенесен с Финского залива в северную часть города – Приморский район.

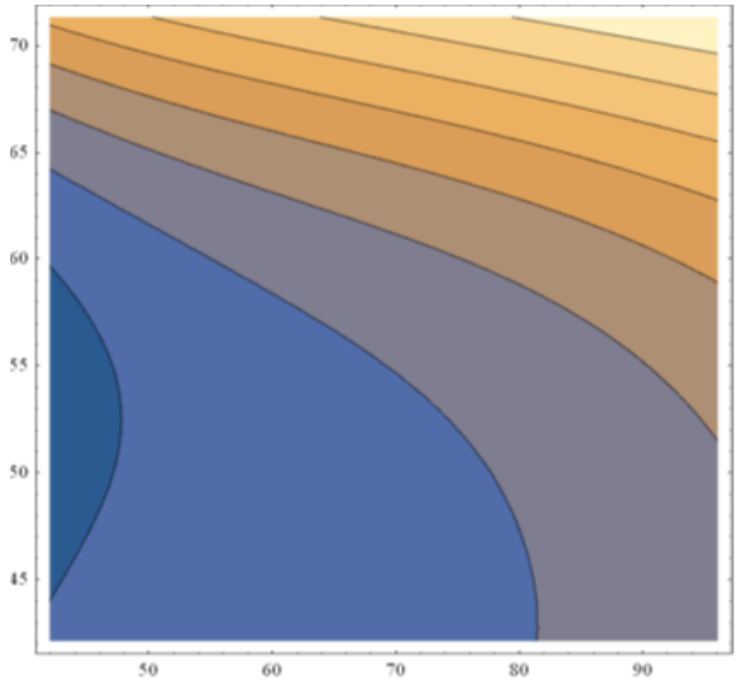


Диаграмма 3

По диаграмме 3 заметно, что загрязнение атмосферного воздуха переместилось из центра на северо-восточную часть Санкт-Петербурга при скорости ветра – 2м/с и западном направлении ($V_x=2, V_y=0$). Наивысший уровень загрязнения наблюдается в Выборгском районе.

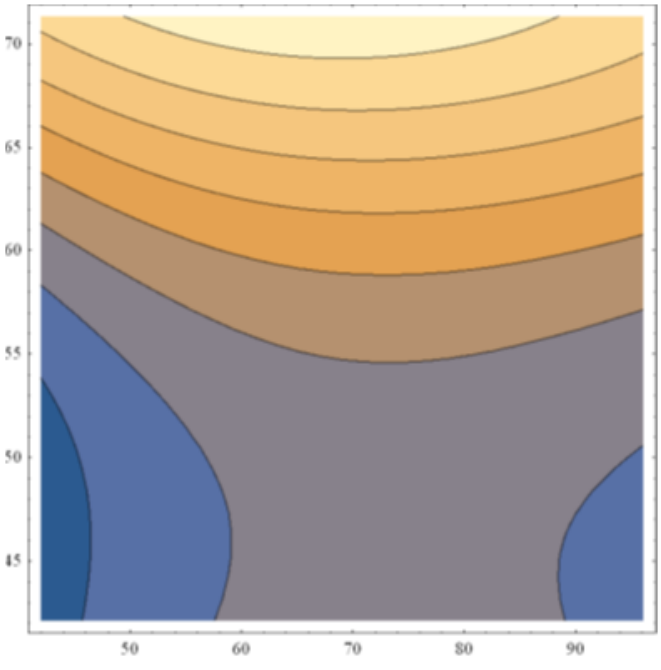


Диаграмма 4

Из диаграммы 4 заметно, как загрязнение перемещается от Финского залива в северную часть города Санкт-Петербурга. Это обусловлено преобладающим юго-западным ветром со скоростью 1,41 м/с ($V_x=1, V_y=1$). При данных характеристиках ветра наиболее загрязненным районом является Приморский.

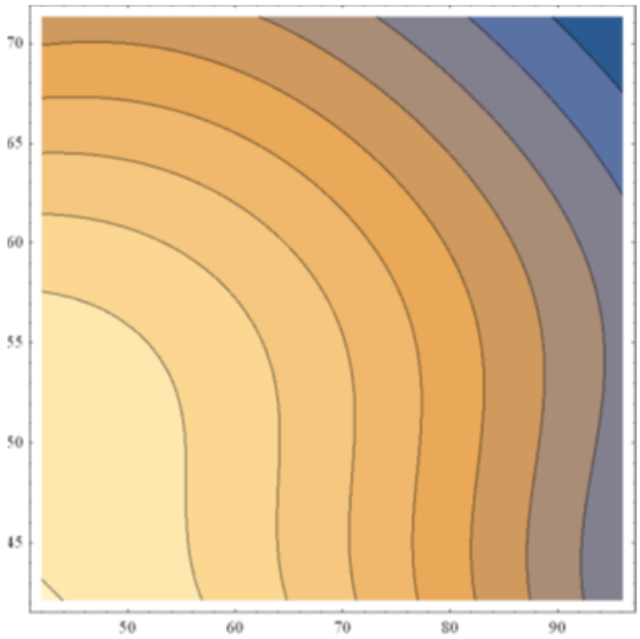


Диаграмма 5

Согласно диаграмме 5, наивысший уровень загрязнения в Петродворцовом районе, в юго-западной части города Санкт-Петербурга, при северо-восточном ветре со скоростью 1,41 м/с ($V_x = -1, V_y = -1$).

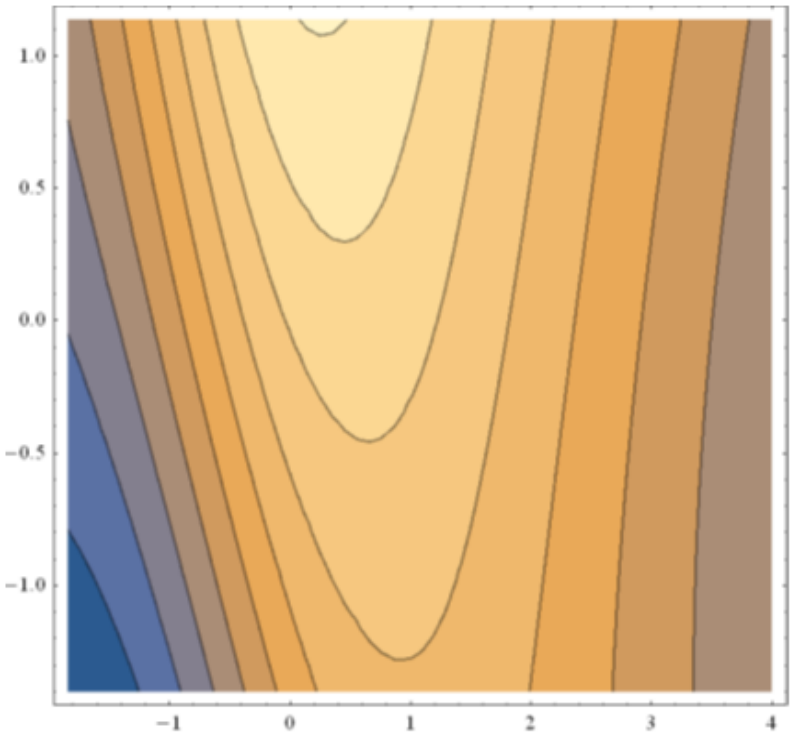


Диаграмма 6

На диаграмме 6 представлена зависимость уровня загрязнения атмосферного воздуха в центре города Санкт-Петербурга от компонент скорости ветра V_x и V_y .

При северо-восточном ветре концентрация загрязняющих веществ в центре минимальна, так как на северо-востоке мало заводов и других источников загрязнения. При западном и юго-восточном ветре эта тенденция есть, но менее выражена. При ветре южных направлений картина более сложная, всё, видимо, зависит от того, приносит ли ветер загрязнения с ижорских заводов.

Северный ветер приводит только к возрастанию загрязнений, что свидетельствует о больших загрязнениях к северу от центра. Это можно объяснить тем, что в Выборгском районе находятся промышленные предприятия, расположенные, в основном, вдоль Выборгской набережной и в промзоне «Парнас»; основными стационарными источниками загрязнения являются в общей сложности 16 фабрик и заводов: «Двигатель», ЦНИИ «Гидроприбор», АООТ «Светлана», ОАО «Позитрон», на территории района проходит более 10 магистралей, такие, как Выборгское шоссе, Светлановский, Суздальский, Северный и другие проспекты.

На основании полученных в ходе исследования результатов можно сделать вывод, что распределение загрязнения атмосферного воздуха в городе Санкт-Петербурге напрямую зависит от скорости и направления ветра. В безветренную погоду наибольший уровень загрязнения – в районе Финского залива, а с появлением ветра это загрязнение вместе с воздушными массами переносится по городу в зависимости от направления ветра.

Литература

1. Васильев А.Н., Тархов Д.А. Нейросетевое моделирование. Принципы. Алгоритмы. Приложения. – СПб.: Изд-во СПбГПУ, 2009. – 528 с.
2. Тархов Д.А. Нейросетевые модели и алгоритмы. М.: Радиотехника, 2014. – 348 с.

Зыкина А.В.[1], Запорожец Д.Н.[2]

[1] Омский государственный технический университет, г. Омск, д.ф.-м.н., зав. кафедрой прикладной математики и фундаментальной информатики, avzykina@mail.ru

[2] Омский государственный технический университет, г. Омск, к.ф.-м.н., доцент кафедры прикладной математики и фундаментальной информатики, demlcf@mail.ru

ПАКЕТ ПРИКЛАДНЫХ ПРОГРАММ ДЛЯ МОДЕЛИРОВАНИЯ И РЕШЕНИЯ ПРОЦЕССОВ С ИСПОЛЬЗОВАНИЕМ АППАРАТА ВАРИАЦИОННЫХ НЕРАВЕНСТВ

КЛЮЧЕВЫЕ СЛОВА

Веб-разработка, веб-приложение, вариационные неравенства, экстраградиентные методы.

АННОТАЦИЯ

В статье обсуждается структура и основные принципы работы программного комплекса ExtaPMP, предназначенного для решения задач планирования и организации производства с использованием аппарата вариационных неравенств. Главная особенность программного комплекса заключается в его расширяемости. В том числе, программный комплекс может быть использован для решения задач линейного стохастического программирования с нормально или равномерно распределенными случайными параметрами путем сведения к эквивалентной детерминированной задаче. Программный комплекс может быть полезен научным сотрудникам, аспирантам и студентам как инструмент исследования и приобретения дополнительных навыков в моделировании и программировании. Программно-аппаратные требования: операционная система Windows 7, оперативная память 256 Мб, объем программного комплекса 3072 Кб. Разработанный программный комплекс отвечает требованиям новизны и является вкладом в объединенный фонд электронных ресурсов «Наука и образование».

Исследование выполнено при поддержке РФФИ, проект № 15-41-04436_р_сибирь_а.

Введение

В настоящее время происходит стремительное развитие наук, как фундаментальных, так и прикладных, использующих сложные математические модели и, как следствие, требующих сложной, но быстрой обработки информации. Это вместе с быстрым технологическим прогрессом приводит к тому, что значительно возросла потребность в применении мощных вычислительных средств и суперкомпьютерных технологий. Вариационные неравенства – это современный вычислительный инструмент, активно развивающийся в последние десятилетия, работа с задачами в такой формулировке позволяет использовать для их решения эффективные численные методы. Актуальность разработки новых итерационных методов решения вариационных неравенств обусловлена тем, что вариационные неравенства являются обобщением классических постановок задач оптимизации и имеют многочисленные приложения (примеру, равновесие транспортных потоков, вопросы ценового равновесия, баланса спроса и предложения, выбор портфеля ценных бумаг и др.). Актуальность использования технологии параллельного программирования в итерационных методах вообще, и в итерационных методах решения вариационных неравенств в частности, обусловлена, прежде всего, появлением новой архитектуры вычислительных систем с многопроцессорной конфигурацией.

Описание программного комплекса ExtaPMP

Программный комплекс [1, 2, 3] предназначен для решения задач планирования и организации производства экстраградиентными методами и может служить учебным пособием для студентов, изучающих методы оптимизации и программирование, а также инструментом исследования для научных сотрудников. Программа написана на языке C# (.Net 2.0), в качестве среды разработки использовалась Microsoft Visual Studio 2010. Для работы программы необходима операционная система Windows и .NET Framework 2.0 или выше.

Программный комплекс состоит из 3 основных модулей: модуль MatrixCalc, модуль Task, модуль Method. В первом модуле содержится программная реализация основных действий над векторами и матрицами. Во втором модуле содержится описание интерфейса вариационного неравенства и задачи, сводимой к вариационному неравенству. Благодаря этому модулю исследователь может запрограммировать собственную задачу и решить задачу при помощи данного программного комплекса. Третий модуль содержит описание интерфейса итерационного метода решения вариационного неравенства. Таким образом, предоставляется возможным использовать в программном комплексе ExtaPMP не только собственные задачи, но и методы, причем к каждому методу автоматически добавится еще один метод с памятью. Методы с памятью являются параллельными и позволяют находить решение существенно быстрее.

Программный комплекс содержит несколько задач, сводящихся к вариационному неравенству: система линейных алгебраических уравнений, линейная задача дополнительности, нелинейное вариационное неравенство минимизация функции Розенброка, минимизация функции Химмельблау, задача линейного программирования, транспортная задача и задача планирования производства. Каждую из вышеперечисленных задач можно решить при помощи градиентного метода, одношагового экстраградиентного метода или двухшагового экстраградиентного метода. Методы решения вариационного неравенства, как и сами задачи, реализованы в виде подключаемых модулей, поэтому в программе дополнительно возможно выбрать следующие методы: градиентный метод с памятью, одношаговый экстраградиентный метод с памятью, двухшаговый экстраградиентный метод с памятью.

Описание модуля MatrixCalc

На рисунке 1 показана диаграмма классов «матрица» и «вектор». В этом модуле содержатся классы матрица и вектор, определены основные операции над ними, такие, как умножение, сложение векторов и матриц, произведение матрицы на вектор, произведение матриц, скалярное произведение векторов.

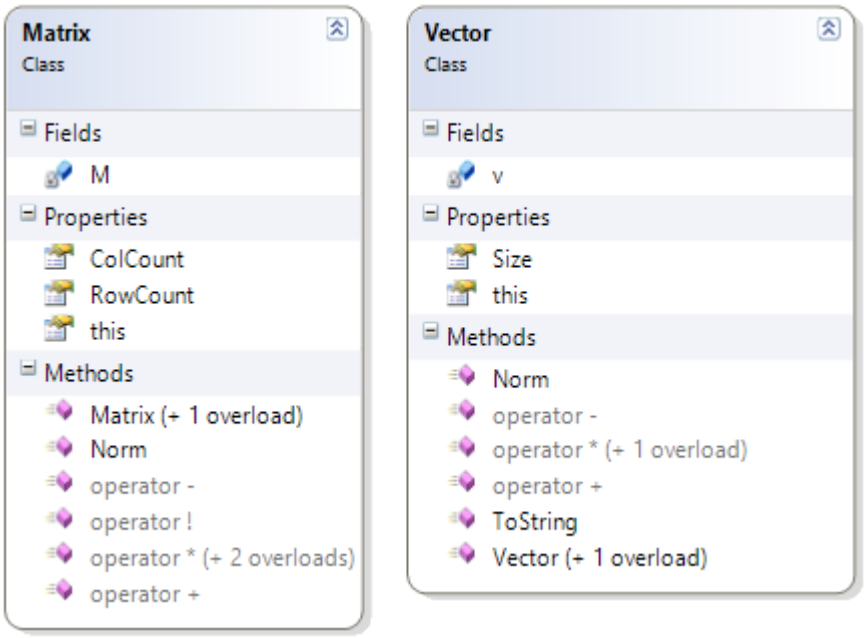


Рис.1. Диаграмма классов модуля MatrixCalc

Описание модуля Task

Модуль Task содержит интерфейс задачи ITask, класс VI, реализующий функционал вариационного неравенства.

На рисунке 2 изображена диаграмма классов модуля Task.

Так как в этом модуле содержится интерфейс ITask, то это позволяет любому программисту или исследователю реализовать собственную, интересующую его задачу в виде отдельного модуля программного комплекса ExtaPMP. Для этого достаточно написать класс, реализующий интерфейс ITask, и оформить этот класс в виде динамически подключаемой библиотеки (dll). Интерфейс ITask имеет ряд интересных свойств и методов. Свойство *Name* содержит название задачи, а *Overview* – ее краткое описание. Свойство *CanUseFillDlg* отвечает за то, можно ли использовать специальную

форму заполнения задачи. Если таковой нет, то задача должна быть полностью создана в конструкторе класса, иначе программист должен реализовать визуальную форму, в которой будет предоставлена возможность пользователю ввести все необходимые данные. На этой форме обязательно должна быть кнопка «создать», по нажатию которой необходимо выставить значение свойства *Created* в *true*, а так же послать сообщение *WM_TASKCREATED* родительскому окну. В методе *ToVarIneq* реализована логика приведения задачи к вариационному неравенству (класс VI).

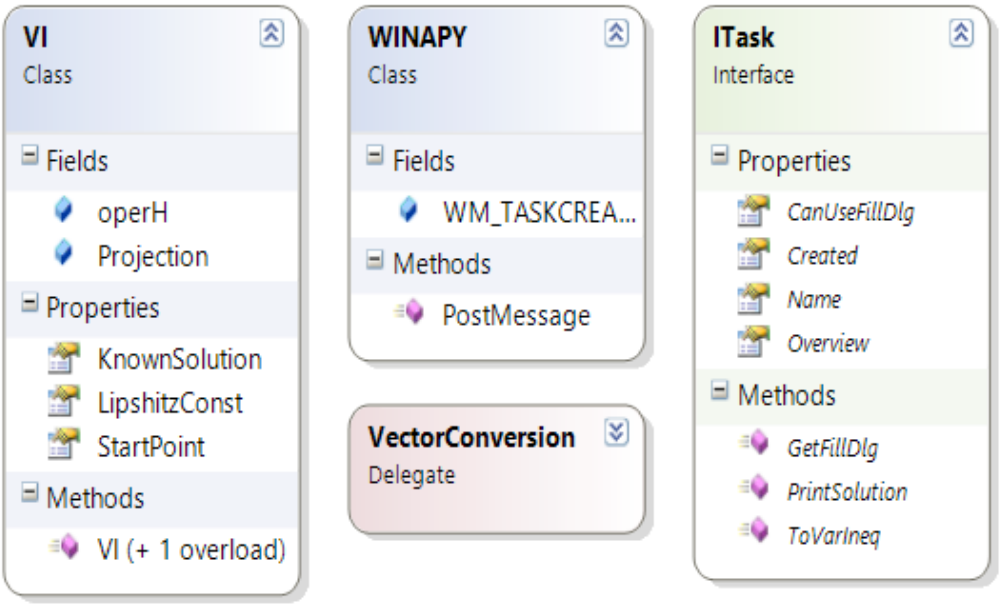


Рис.2 Диаграмма классов модуля Task

Описание модуля IMethod

Для того чтобы решать задачи новым методом, не включенным в программный комплекс, необходимо написать класс, реализующий интерфейс IMethod.

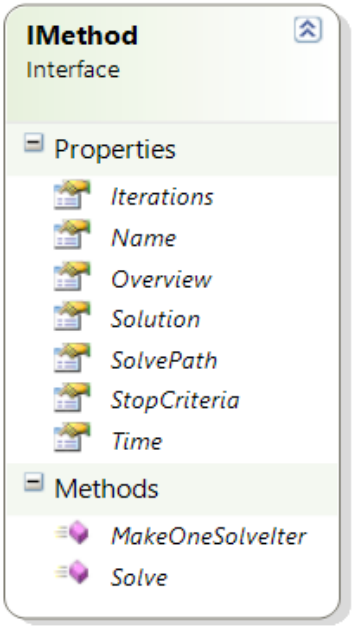


Рис.3. Диаграмма классов модуля IMethod

На рисунке 3 изображена диаграмма классов модуля IMethod.

При этом необходимо выделить два решающих метода-класса: *Solve* – находит решение вариационного неравенства, а *MakeOneSolveIter* – делает одну итерацию метода. Свойство *Solution* содержит решение, найденное методом, а свойство *Iterations* – количество итераций, сделанное методом, *Time* – показывает время в секундах, затраченное на решение задачи, , *SolvePath* – хранит

траекторию метода, *StopCriteria* – содержит пороговое значение критерия остановки *StopCriteria, Name* – название метода, *Overview* – его описание.

На рисунке 4 показано окно, которое пользователь увидит при первом запуске программного комплекса.

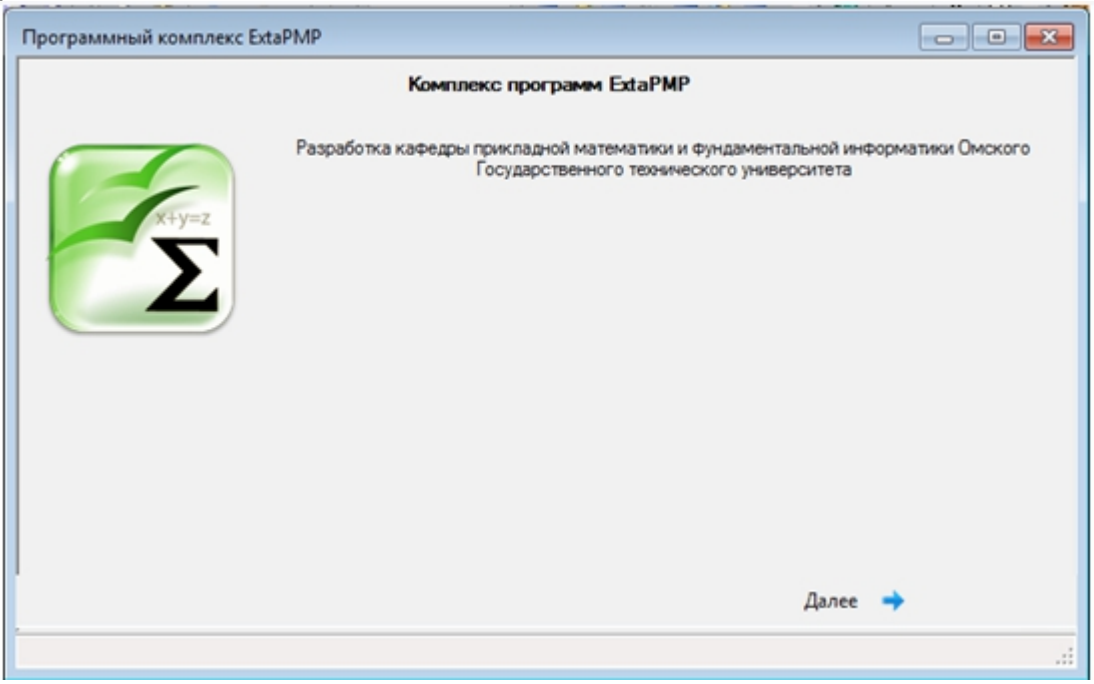


Рис.4. Окно первого запуска

При нажатии на кнопку «Далее» пользователь попадет на страницу выбора метода решения задачи.

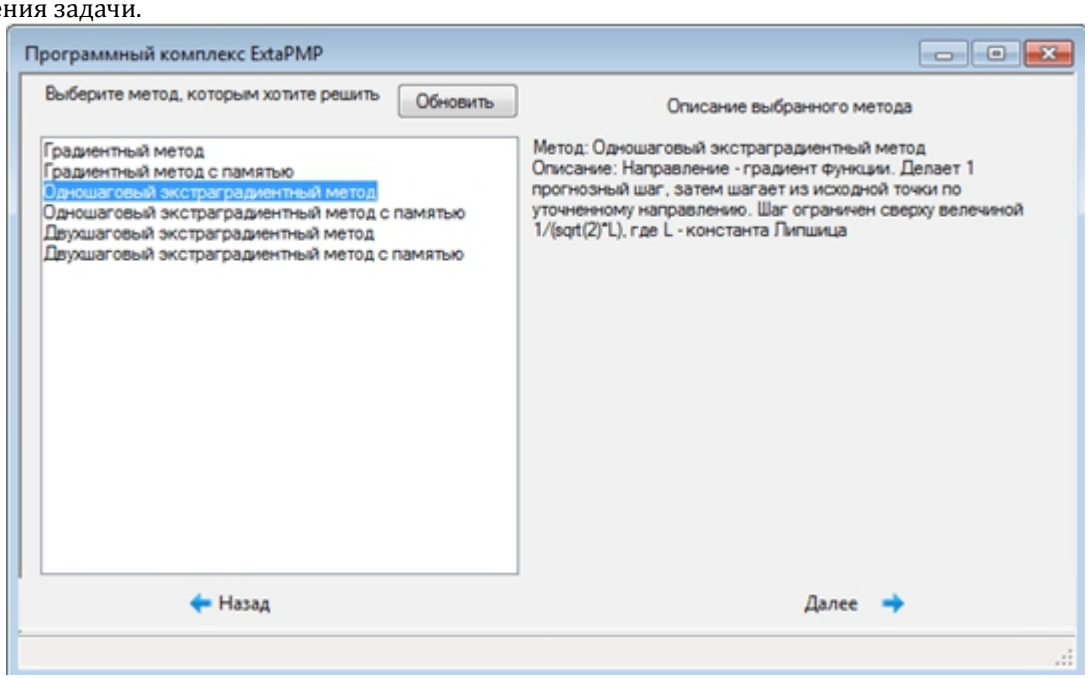


Рис.5. Окно выбора метода решения задачи

На рисунке 5 показано окно выбора метода решения задачи.

При выборе метода решения задачи пользователю показывается краткое описание метода. Кнопка обновить сканирует каталог Methods и служит для перестроения списка методов. При нажатии на кнопку «Далее» пользователь попадает на страницу выбора задачи.

На рисунке 6 показано окно выбора решаемой задачи.

При выборе задачи пользователю будет показано краткое описание задачи. При нажатии

на кнопку «Далее» пользователь попадает на страницу заполнения данных, или сразу переходит на страницу отображения результатов решения, если заполнение данных задачи не требуется и соответствующая форма отсутствует.

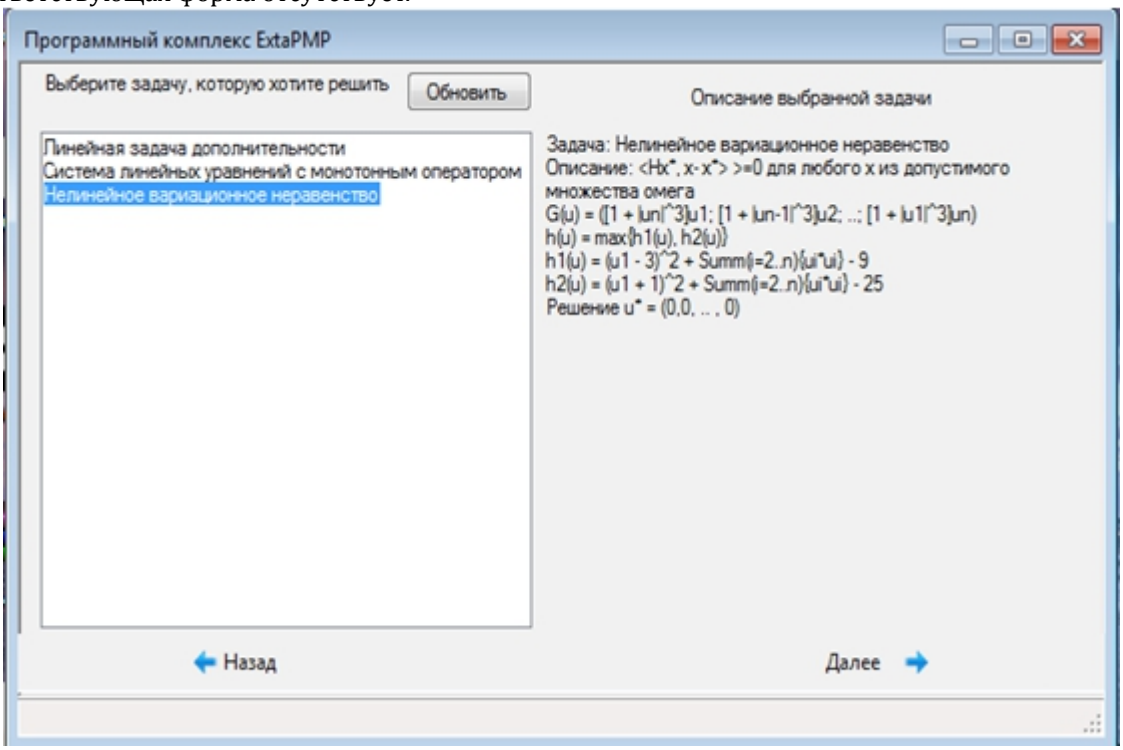


Рис.6. Окно выбора решаемой задачи

На рисунке 7 показано окно с результатами решения задачи. Здесь отображается информация о выбранных методе и задаче, количество сделанных итераций, время, затраченное на поиск решения, стартовая точка, из которой метод начинал решение задачи, и найденное решение.

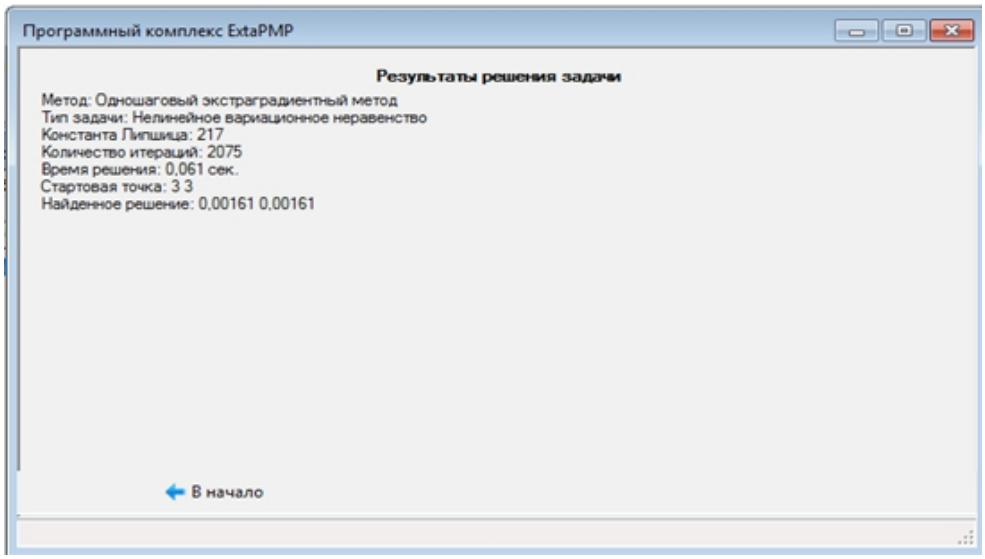


Рис.7. Окно с результатами решения задачи

Выводы

Программный комплекс MMSolver позволяет решать сводящиеся к вариационным неравенствам актуальные задачи математического моделирования из различных областей знаний: задачи теории игр, задачи о седловой точке, задачи линейного и выпуклого программирования, линейные задачи дополнительности и многие другие. В программном комплексе реализованы технологии параллельного программирования в численных методах

решения вариационных неравенств. В ходе вычислительных экспериментов, проведенных на построенных алгоритмах, могут быть получены новые закономерности, характеризующие построенные математические модели социально-экономических, технических и информационных систем. В том числе программный комплекс может быть использован для решения задач линейного стохастического программирования с нормально или равномерно распределенными случайными параметрами путем сведения к эквивалентной детерминированной задаче [4]. Детерминированный эквивалент является задачей выпуклого программирования, которая сводится к вариационным неравенствам.

Литература

1. Запорожец Д.Н., Меленьчук Н.В. Экстраградиентные методы (программа) // М.: ОФЭРНиО ФГПУ ИНИПИ РАО, 2010. – № 50201050071. – Св-во о регистрации электронного ресурса № 16330 от 27 октября 2010 года.
2. Запорожец Д.Н. Программный комплекс MMSolver (программный комплекс) // М.: ОФЭРНиО ФГПУ ИНИПИ РАО, 2012. – № 50201250473. – Св-во о регистрации электронного ресурса № 18137 от 17 апреля 2012 года.
3. Запорожец Д.Н. Программный комплекс ExtaPMP (программный комплекс) // М.: ОФЭРНиО ФГПУ ИНИПИ РАО, 2012. – № 50201251393. – Св-во о регистрации электронного ресурса № 18705 от 26 ноября 2012 года.
4. Kaneva O., Atink D. Research of statistic methods in finding the solutions of a linear stochastic problems // VI International conference on optimization methods and applications «Optimization and applications» (OPTIMA-2015): Proceedings include abstracts of reports presente. – Petrovac, Montenegro, 2015. – С. 91.

Бабичева Т.С.[1,2]

[1]Московский физико-технический институт (государственный университет), ассистент, eriright@gmail.com

[2]Институт прикладной математики им. М.В. Келдыша, м. н. с.

ТРАНСПОРТНЫЕ ПОТОКИ: МАТЕМАТИЧЕСКОЕ И ИМИТАЦИОННОЕ МОДЕЛИРОВАНИЕ

КЛЮЧЕВЫЕ СЛОВА

Транспортное моделирование, имитационное моделирование, теория систем массового обслуживания, автотранспортное средство, перекрёстки, управление движением.

АННОТАЦИЯ

В статье описываются методы математического и имитационного моделирования процессов локального взаимодействия в транспортных системах. Описаны разработанные автором модели транспортных потоков, основанные на модификации микроскопической модели Трайбера и на теории систем массового обслуживания, более полно, чем существующие, описывающие поведение участников дорожного движения на управляемых перекрёстках.

Микромодель движения АТС. Появление ЭВМ позволило производить сложные численные эксперименты с помощью имитационного моделирования процессов, и, что особо важно, появилась возможность учитывать случайный характер транспортного потока. К моделированию обычно прибегают в тех случаях, когда изучаемые системы невозможно анализировать с помощью прямых или формальных аналитических методов.

В микроскопических моделях каждое АТС рассматривается как отдельный элемент транспортной системы. Полагается, что ускорение конкретного АТС зависит от соседних АТС. Наибольшее влияние на поведение водителя оказывает АТС, движущееся впереди, *лидирующее АТС, лидер*.

Автором разработана микроскопическая модель транспортных потоков, обобщающая существующую "модель разумного водителя" Трайбера [1] на случай многополосных дорог и перекрёстков. В этой модели для каждого АТС задаются желаемая скорость в диапазоне $0 \ldots V_{max}$, величина d – расстояние между текущим АТС и следующим перед ним, V_n – скорость текущего АТС. Правила обновления задаются моделью Трайбера.

Модель Трайбера была обобщена на двумерный случай путём введения вероятностных характеристик возможности смены полосы и необходимости смены полосы, что позволило учитывать, в том числе, пересечение потоков транспортных средств при поворотах на перекрёстке. Для рассмотренных методов был разработан комплекс компьютерных программ BTSSIM, реализующий математические модели, с помощью которого проведён ряд вычислительных экспериментов.

Рассмотренная микроскопическая модель была использована для решения задачи о движении АТС на регулируемом перекрёстке. Рассматривалось движение АТС на перекрёстке с заранее заданными неоднородными многоцелевыми потоками. Решалась задача оптимизации длительности фаз светофорного режима на сетке значений, обеспечивающих максимальную пропускную способность перекрёстка при заданных интенсивностях потоков на каждом из направлений. Полученные распределения фаз были проверены на устойчивость к небольшим изменениям интенсивностей входящих потоков с различных сторон, что показало возможность работы с неточными данными.

Введём понятие *эффективного числа полос*, которое поясним следующим примером. Предположим, что поток АТС, двигающийся по шестиполосной дороге, доходит до Т-образного перекрестка. Предположим, что целью трети водителей является поворот налево, а целью остальных - движение прямо. В этом случае две крайние левые полосы займут АТС, у водителей которых цель - поворот налево. Поток АТС через светофор при включения зеленого сигнала для

прямого движения будет не 6 000 АТС/час, а только 4 000 АТС/час. Эффективное число полос в этом случае равно 4 [3].

Сформулируем следующее свойство и его следствие:

Свойство о равновесной максимальной пропускной способности на управляемом перекрёстке.

Пусть имеется управляемый многополосный перекрёсток. Пусть имеется поток АТС с одного направления на фиксированной светофорной фазе. Пусть N – число полос на исходящей дороге, N_i – число полос на целевых дорогах, S – максимальная пропускная способность одной полосы, $\Omega_i(t)$ – очередь АТС с исходящей дороги в направлении i в момент времени t, $\Omega(t)$ – очередь АТС на исходящей дороге в момент времени t, $\Omega_a(t)$ - очередь тех АТС на исходящей дороге, которые могут продолжить движение на данной фазе в момент времени t. Тогда:

Математическое ожидание максимального проходящего потока с одного направления на управляемом перекрёстке на данной фазе за время T равно:

$$E[S_{max} \cdot T] = \int_0^T min(S \cdot N \cdot \frac{\Omega_a(t)}{\Omega(t)}, \sum_i S \cdot N_i \cdot \frac{\Omega_i(t)}{\Omega(t)})dt.$$

Следствие об эффективном числе полос в отсутствии асимметрии очередей.

Пусть имеется управляемый многополосный перекрёсток. Пусть имеется поток АТС с одного направления на фиксированной светофорной фазе. Пусть в начальный момент времени очередь на перекрёстке отсутствует. Пусть q – математическое ожидание входящего потока с данного направления, k – математическое ожидание потока АТС, которые могут продолжить движение на данной фазе, S – максимальная пропускная способность одной полосы, N – число полос на исходящей дороге, N_i – число полос на целевых дорогах, k_i – математическое ожидание потока АТС, способных продолжить движение в направлении i на данной фазе. Тогда:

Математическое ожидание максимального проходящего потока с одного направления на данном управляемом перекрёстке на данной фазе равно:

$$E[S_{max}] = \min\left(S \cdot N \cdot \frac{k}{q}, \sum_i S \cdot N_i \cdot \frac{k_i}{q} \right).$$

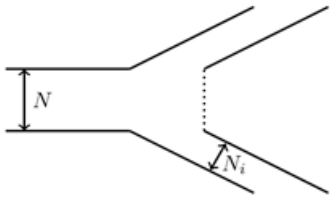


Рис. 1. Иллюстрация к свойству

Для решения задачи поиска ожидаемых задержек, возникающих при пересечении управляемого перекрёстка с фиксированными длительностями фаз применён аппарат теории систем массового обслуживания, который к транспортным задачам на многополосных дорогах практически не применялся. В исследовании данных задач, например, необходимо учитывать, что время обслуживания на многополосных перекрёстках - величина, зависящая от текущей светофорной фазы и распределения целей движущихся АТС.

Вводятся следующие сокращения и обозначения:

c – сигнальный цикл;

c_i – время длительности i -й фазы;

q – поток входящего трафика;

S – максимальный поток проходящего перекресток трафика;

Ω_0 - ожидаемая очередь переполнения из предыдущих циклов.

Рассмотрен простой поток - Пуассоновский процесс АТС, прибывающих к светофору с фиксированными временами фаз. Задержка АТС на светофоре равна времени, потраченному АТС на перекрёстке на ожидание. Общая задержка АТС во время одного сигнального цикла как сумма компонент, получающихся при каждой фазе цикла, выражается как

$$W=\sum_{i=1}^{4} W_i ,$$

где W_i – общая задержка АТС, вызванная i -й фазой,

$$W_i = \int_{\sum_{j=1}^{i-1} c_j}^{\sum_{j=1}^{i} c_j} \Omega(t)dt.$$

Воспользовавшись понятием эффективного числа полос, получаем, что, если k_i – интенсивности потоков, которые могут продолжать движение в *i*-той фазе, то проходящий поток будет равен

$$E[S_i]=S\cdot\frac{k_i}{q}.$$

Общий результат можно сформулировать следующим образом:

$$E[W]=c\Omega_0+\frac{P_1c_1^2}{2}+\frac{P_2c_2^2}{2}+c_2P_1c_1+\frac{P_3c_3^2}{2}+c_3(P_1c_1+P_2c_2)+\frac{P_4c_4^2}{2}+c_4(P_1c_1+P_2c_2+P_3c_3),$$

где $P_i=q-E[S_i]$ – так называемые "избыточные потоки".

Пусть P_{ij} – избыточный поток для дороги с номером *i* на фазе *j*, Ω_{i0} – начальная очередь АТС на данной дороге.

При рассмотрении математического ожидания суммарной задержки АТС за время *T* со всех направлений для одного перекрёстка с четырьмя фазами, то есть за *T/c* светофорных циклов, получается [2]:

$$E[W]=c(\sum_{i=0}^{\frac{T}{c}}(\sum_{j=1}^{4}\max(\Omega_{j0}+\sum(c_iP_{ji})i,0)+c_1\sum_{i=1}^{4}P_{i1})(c_1/2+c_2+c_3+c_4)+$$

$$+(c_2\sum_{i=1}^{4}P_{i2})(c_2/2+c_3+c_4)+(c_3\sum_{i=1}^{4}P_{i3})(c_3/2+c_4)+(c_4\sum_{i=1}^{4}P_{i4})(c_4/2))\cdot\frac{T}{c}.$$

Для конкретных перекрёстков величины P_{ij} можно считать постоянными. Решение этой задачи в общем виде достаточно громоздко и разбивается на множество случаев в зависимости от этих величин.

Для перекрестка двухполосных дорог, просчитанного численно в работе [1], была рассчитана общая задержка за светофорный цикл со всех направлений.

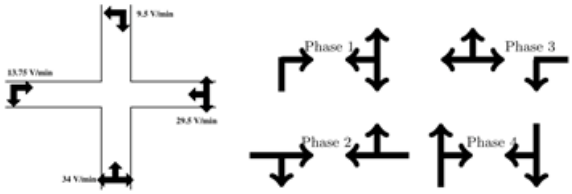


Рис. 2. Перекресток и фазы светофора

Задача была сведена к решению проблемы минимизации общих задержек в единицу времени: $E(W)/T$, то есть, к минимизации функции

$$F = (c \cdot \frac{1}{2}(\frac{T}{c}+1)\frac{T}{c}((\max(14c_1 - 26c_2 + 34c_3 - 6c_4, 0) +$$
$$+\max(13.75c_1 + 13.75c_2 - 46.25c_3 + 13.75c_4, 0) +$$
$$+\max(9.5c_1 + 9.5c_2 + 9.5c_3 - 51.5c_4, 0) + \max(-31.5c_1 + 9.5c_2 - 11.5c_3 + 29.5c_4, 0)) +$$
$$+(c_1(5.75)(c_1/2 + c_2 + c_3 + c_4) + c_2(6.75)(c_2/2 + c_3 + c_4) +$$
$$+c_3(-14.25)(c_3/2 + c_4) + c_4(-14.25)(c_4/2)) \cdot \frac{T}{c}) / T$$

на пирамиде

$c_1 \ldots 0$,

$c_2 \ldots 0$,

$c_3 \ldots 0$,

$c - c_1 - c_2 - c_3 \ldots 0$.

Решение этой задачи, в общем, зависит от отношения T/c , количества рассматриваемых циклов, но полученное решение асимптотическое и практически не подвержено колебаниям начиная с $T/c = 50$ (табл. 1).

c_1^*/c	c_2^*/c	c_3^*/c	c_4^*/c	T/c
0.208502	0.368045	0.229767	0.193685	40
0.209071	0.367474	0.2292	0.194255	50
0.209071	0.367473	0.229201	0.194255	60
0.209071	0.367474	0.2292	0.194255	70
0.209071	0.367473	0.2292	0.194255	80
0.209071	0.367474	0.2292	0.194255	90
0.209071	0.367474	0.2292	0.194255	100

Таблица 1. Зависимость оптимального значения c_i^*/c от количества циклов T/c.

Сравнение результатов анализа перекрёстков аналитической и микромодели для данного перекрёстка (таб. 2, 3) показывают, что результаты, полученные с помощью аналитического аппарата коррелируют с результатами микромоделирования.

Фаза 1, сек	Фаза 2, сек	Фаза 3, сек	Фаза 4, сек	CPT
35	61.5	38.4	32.5	0.741

Таблица 2. Результаты, полученные аналитически с помощью аппарата систем массового обслуживания

Фаза 1, сек	Фаза 2, сек	Фаза 3, сек	Фаза 4, сек	CPT
60	30	45	30	0.737
45	30	30	30	0.737
30	75	45	30	0.737
...	...	...	...	...
45	30	30	90	0.566
30	30	30	90	0.560

Таблица 3. Результаты, полученные численным микромоделированием

Движение на кольцевой автодороге. Далее рассматривается моделирование движения на кольцевой автостраде, закрученной в одну сторону. Рассматриваемая автострада представлена в виде, рассмотренном на рис. 3.

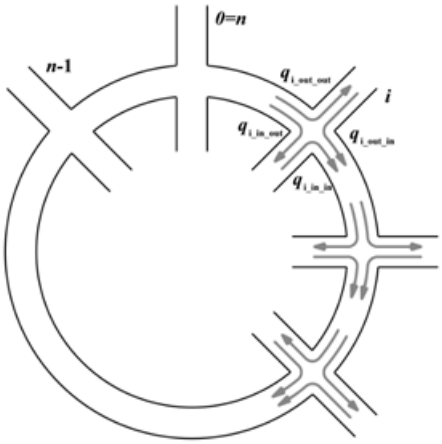


Рис. 3. Кольцевая автострада

Здесь:

- под q_i подразумевается поток трафика от узла с номером *i* к узлу с номером *i+1*;
- под $q_{i,out,out}$ – поток исходящего трафика из *i*-того узла в направлении от кольца;
- под $q_{i,\text{in},out}$ – поток исходящего трафика из *i*-того узла в направлении внутрь кольца;
- под $q_{i,out,\text{in}}$ – поток входящего трафика из *i*-того узла в направлении от кольца;
- под $q_{i,\text{in},\text{in}}$ – поток входящего трафика из *i*-того узла в направлении внутрь кольца.

Моделирование движения осуществляется для некоторой "матрицы корреспонденций" *Q*, имеющей следующий вид:

	0_{in}	0_{out}	...	...	$(n-1)_{out}$
0_{in}	×	$Q[0,in,0,out]$			
0_{out}	$Q[0,out,0,in]$	×			
⋮			×		
⋮					
$(n-1)_{out}$					×

Для удобства вычислений, ячейки нумеруются не в виде $2n \times 2n$, а в виде *Q*[*i, in/out, j, in/out*], то есть, представление матрицы корреспонденции есть четырёхмерная матрица $n \times 2 \times n \times 2$. Для численной записи в данной матрице *in* обозначает 0, а *out* обозначает 1. Под *Q*[*i, in, j, out*] подразумевается число АТС в единицу времени, желающих попасть с внутренней дороги *i*-го узла на внешнюю дорогу *j*-го узла.

Пусть N_i – число полос у соответственных побочных дорог и *N* – число полос на кольцевой автостраде. Число полос у въездов полагается всегда достаточной для пропуска всех желающих в случае полностью зелёного сигнала, в таком случае можно использовать следствие об эффективном числе полос. Автострада принимается постоянной ширины, максимальная пропускная способность одной полосы – константа, равная *S*. Рассчитана максимальная пропускная способность рассматриваемой автострады в зависимости от матрицы корреспонденций и пропускных способностей главной и побочной дорог.

Ранее рассмотренная модель движения на изолированном перекрёстке обобщена и расширена на данный случай.

При движении по ранее показанной схеме математическое ожидание максимального проходящего потока с главной дороги на *n*-той фазе есть

$$E[S_i^n] = S \cdot N \cdot \frac{k_i^n}{q_{i-1}}$$

с побочных дорог

$$E[S_{i,u}^n] = S \cdot N_i \cdot \frac{k_{i,u}^n}{q_{away,i,u}}.$$

Здесь $q_{away,i,u}=\sum_{j=0}^{n-1}\sum_{l=0}^{1}Q[j,l,i,u]$ – поток АТС, желающий выехать на *i*-м узле в направлении *u(in/out)*.

k_i^n – соответственные интенсивности потоков АТС, которые могут продолжать движение в *n*-й фазе с главного направления,

$k_{i,u}^n$ – соответственные интенсивности потоков АТС, которые могут продолжать движение в *n*-й фазе с побочного направления *u(in/out)*.

Для регулируемых светофоров на магистрали с односторонним движением обычно используется цикл из 3 светофорных фаз.

Рис. 4. Возможная схема организации движения на узле магистрали с односторонним движением

Далее рассматриваются светофорные фазы рис. 4.

Пусть $q_{\text{in},i,u}$ – поток, приходящий по второстепенной дороге узла *i* с направления *u*.

$$q_{\text{in},i,u}=\sum_{j=0}^{n-1}\sum_{l=0}^{1}Q[i,u,j,l]$$

Тогда:

$$c^i\cdot E[q_i]=(c_1^i+c_2^i)\cdot\left(S\cdot N\cdot\frac{q_{i-1}-\sum_{u=0}^{1}q_{away,i,u}}{q_{i-1}}\right)+$$

$$+c_2^i\cdot(S\cdot N_i\cdot\frac{q_{in,i,0}-Q[i,0,i,1]}{q_{in,i,0}}+S\cdot N_i\cdot\frac{q_{in,i,1}-Q[i,1,i,0]}{q_{in,i,1}}),$$

что даёт рекуррентное соотношение, позволяющее от *i*-го узла перейти к *i+1*-му, и это позволяет замкнуть модель.

Задача минимизации общих задержек, возникающих при преодолении кольцевой автострады.

"Избыточные потоки" с каждого направления для каждой фазы равны

$$P_i^n=\max(q_{i-1}-E[S_i^n],0),$$

$$P_{i,u}^n=\max(q_{i,u}-E[S_{i,u}^n],0).$$

Зависимость *P* от *Q* и q_0 получается подстановкой найденных значений и рекуррентным получением q_{i-1}.

Задержка, возникающая при пересечении *i*-го узла всех АТС равна:

$$E[W_{sum}^i]=c\Omega_{sum0_i}+\frac{P_{sum1_i}(c_1^i)^2}{2}+\frac{P_{sum2_i}(c_2^i)^2}{2}+c_2^iP_{sum1_i}c_1^i+$$
$$+\frac{P_{sum3_i}(c_3^i)^2}{2}+c_3(P_{sum1_i}c_1^i+P_{sum2_i}c_2^i). \quad (1)$$

Важно учесть, что "избыточные потоки" зависят от потоков по главным дорогам, зависящих от сигнального цикла, а также, что, в отличие от задачи об изолированном перекрёстке, изначальные очереди в общем виде зависят от ранее лежащих узлов, а, значит, и от сигнального цикла.

Программный комплекс моделирования движения на кольцевой дороге основан на формуле (1) и предназначен для определения оптимальных фаз светофоров, расположенных на кольцевой дороге.

Входными данными программы являются значения коэффициентов $\Omega_{i,j}$, где $I \in \{1,\ldots,N\}$ и $j \in \{\text{In},\text{Out}\}$, а так же коэффициентов $\Omega_{i,j,k,l}$, где $i.k \in 1,\ldots,N; j,l \in \{\text{in},\text{out}\}$ и коэффициентов $q_{0i}, i \in \{1,\ldots,N\}$.

Первый режим работы - нахождение оптимального решения в предположении, что все светофоры на кольцевой дороге имеют одинаковое распределение фаз, заключается в нахождении вектора коэффициентов (c_1, c_2, c_3), связанных условием $c_1+c_2+c_3=1$, при заданных значениях коэффициентов $\Omega_{i,j}$, $\Omega_{i,j,k,l}$ и q_{0i}, который минимизирует функцию задержек (1). Эта задача решается методом расчёта по сетке возможных значений c_1, c_2, c_3 с заданным шагом дискретизации.

Второй режим работы, нахождение оптимального решения в предположении, что для каждого из светофоров определяются собственные распределения фаз, заключается в определении набора векторов ($(c_{1,i}, c_{2,i}, c_{3,i}, i \in \{1,..,N\})$) при тех же исходных данных, минимизирующего функцию задержек (1). Данная задача есть задача оптимизации функции с $2 \cdot N$ переменными (переменная $c_{3,i}$ связана с остальными переменными выражением $c_{3,i}=1-c_{1,i}-c_{2,i}$).

Заключение. В работе впервые проведена модификация модели Трайбера путём внедрения в модель поведения интеллектуального агента, обеспечивающего многовариантное поведение, характерного для движения на многополосной магистрали и перекрёстках, в том числе, сложной формы, а также на модели кругового движения. Модель апробирована на конкретных перекрёстках, результаты апробации опубликованы в ряде работ, в том числе [1-3].

Литература

1. Babicheva T.S., Babichev D.S., Numerical Methods for Modeling of Traffic Flows at Research and Optimization of Traffic on the Signal-controlled Road Intersections// Procedia Computer Science, 3rd International Conference on Information Technology and Quantitative Management, ITQM 2015, Vol. 55, pp 461–468, 2015
2. Babicheva T.S., The Use of Queuing Theory at Research and Optimization of Traffic on the Signal-controlled Road Intersections// 3rd International Conference on Information Technology and Quantitative Management, ITQM 2015, Vol. 55, pp 469–478, 2015
3. Обидина Т.С., О возможной реорганизации в строительстве перекрестков на примере г. Королев (МО)// Современные проблемы фундаментальных и прикладных наук. Часть VII Управление и прикладная математика: Труды 55-й научной конференции МФТИ./ М.-Долгопрудный, 2012.

Идрисова Д.И.[1], Каверзнева Т.Т.[2], Тархов Д.А.[3], Лазовская Т.В.[4]

[1] Санкт-Петербургский государственный политехнический университет, г. Санкт-Петербург, старший преподаватель кафедры «Безопасность жизнедеятельности»

[2] Санкт-Петербургский государственный политехнический университет, г. Санкт-Петербург, к.т.н., доцент кафедры «Безопасность жизнедеятельности»

[3] Санкт-Петербургский государственный политехнический университет, г. Санкт-Петербург, д.т.н., профессор кафедры «Высшая математика» dtarkhov@gmail.com

[4] Санкт-Петербургский государственный политехнический университет, г. Санкт-Петербург, старший преподаватель кафедры «Высшая математика»

МОДЕЛИРОВАНИЕ РАСПРЕДЕЛЕНИЯ ОПАСНОГО ВЕЩЕСТВА В ТУПИКОВОМ ТОННЕЛЕ С ИСПОЛЬЗОВАНИЕМ НЕЙРОСЕТЕВОГО ПОДХОДА

КЛЮЧЕВЫЕ СЛОВА

Нейросетевое моделирование, диффузия вредных веществ, безопасность, условия труда, строительство.

АННОТАЦИЯ

В статье рассматривается применения нейросетевого подхода для моделирования распространения вредных веществ по воздуху рабочей зоны на этапе строительства и эксплуатации тоннелей в случае аварийного отключения системы вентиляции.

Эксплуатация подземных сооружений различного назначения требует повышенного внимания с точки зрения обеспечения безопасности. Наиболее важной задачей является обеспечение допустимых параметров воздуха рабочей зоны в тоннелях особенно в их тупиковых участках. В основе имеющихся решений лежит использование рациональной системы естественной и искусственной вентиляции.

В данной работе рассматривается обеспечение безопасности в аварийной ситуации, когда система вентиляции выведена из строя или отключена. Особенно актуально решение такой задачи в тоннелях без естественной вентиляции, к ним относятся тупиковые участки (например, тупиковые пути в метрополитене, используемые для очистки, дезинфекции, экипировки и ремонта вагонов), а также на этапе строительства тоннелей, когда сооружен только один портал.

Для обеспечения безопасности работников, находящихся в тоннеле, от воздействия вредных веществ воздуха рабочей зоны наиболее важно моделирование и мониторинг распространения по тоннелю вредных веществ в опасных концентрациях.

В качестве уравнения, описывающего процессы массопереноса, рассмотрим уравнение диффузии, использующееся для описания и анализа массообменных процессов,

$$\frac{\partial w}{\partial t} = a \frac{\partial^2 w}{\partial x^2}, \tag{1}$$

где $w(x,t)$ – концентрация вещества в момент времени t на расстоянии x от источника выделений. Для простоты решения задачи и без потери общности будем считать, что исследование процесса происходит в тоннеле единичной длины и в течение промежутка времени также единичной длины. Выход из тоннеля соответствует значению x=0. Параметр a соответствует постоянному коэффициенту диффузии процесса, его тоже рассмотрим равным 1.

Так как предполагается, что мы ничего не знаем об источнике, находящемся в конце тоннеля, краевая задача содержит одно начальное и одно граничное условие. Это отсутствие выделений на всем протяжении тоннеля в нулевой момент времени

$$w(x,0) = 0 \text{ при } 0 \le x \le 1, \tag{2}$$

и отсутствие какой-либо концентрации на выходе из тоннеля

$$w(0,t) = 0 \text{ при } 0 \le t \le 1. \tag{3}$$

Отметим, что классическая постановка задачи подразумевает наличие граничных условий на правом конце тоннеля, которое в нашей постановке задачи отсутствует. Нами решается задача

восстановления функции $w(x,t)$ в области $[0,1]\times[0,1]$, удовлетворяющей уравнению (1) и неклассической краевой задаче (2)-(3), по измерениям – значениям искомой функции вблизи выхода из тоннеля. Будут рассмотрены два типа таких измерений: точные, полученные, например, с помощью специального прибора, и бинарные, моделирующие ситуацию оценки наличия и отсутствия выделений человеком «на нюх».

Чтобы данные модельной задачи соответствовали реальной ситуации, рассмотрим в качестве генерирующей функции некоторое решение классической задачи, когда к уравнению (1) и условиям (2)-(3) добавлено граничное условие на правом конце. В работе рассмотрены два варианта такой функции.

Первая получается при использовании условия линейного роста производной $\partial_x w(1,t)=t$ и имеет вид

$$R_1(x,t)=tx-\frac{x}{2}+\frac{x^3}{6}+\sum_{n=0}^{+\infty}\frac{\sin\frac{\pi(2n+1)x}{2}}{(2n+1)^4}(-1)^n\exp\frac{-\pi^2(2n+1)^2t}{4}.$$

В реальном вычислительном эксперименте сумма ряда заменялась частичной суммой от $n=0$ до $n=100$, дающей достаточно точное приближение.

Во втором случае рассматривается условие $\partial_x w(1,t)=0$, что соответствует непроницаемости правого конца тоннеля, а на правом конечном участке тоннеля длины d происходит выделение постоянной интенсивности. Тогда решением уравнения будет функция

$$R_2(x,t)=\frac{16}{\pi^3}\sum_{n=0}^{+\infty}\frac{\sin\frac{\pi(2n+1)x}{2}\sin\frac{\pi(2n+1)d}{2}}{(2n+1)^3}(-1)^n\exp\frac{-\pi^2(2n+1)^2t}{4},\ \text{где } d=0{,}1.$$

Данная сумма ряда (так же, как и предыдущая) заменялась частичной суммой от $n=0$ до $n=100$.

Эти функции используются для проверки результата решения поставленной задачи.

Для моделирования предлагаются две ситуации. Если измерения данных ведет прибор, фиксируя точные модельные значения, мы используем значения функций $R_i(x,t)$ в равноотстоящих точках рассматриваемого временного отрезка при фиксированном x. Если предполагается, что прибора нет, и с некоторой погрешностью (индивидуальное восприятие) человек учитывает концентрации выше критического уровня как наличие и ниже как отсутствие вредных выделений в воздухе, создаем бинарные данные, тогда генерация данных проводится по правилу

$$G(t_i)=\begin{cases}\beta, & \text{при } R[1,t_i]+\varepsilon(2\xi-1)>\alpha\\ 0, & \text{иначе}\end{cases}, \qquad (4)$$

Где ε – погрешность измерения, которая может быть очень большой, ξ – стандартная равномерно распределенная случайная величина, α – порог чувствительности, t_i из [0,1]. Значение β выбиралось равным 2α.

Построим нейросетевую модель для нашей задачи с учетом всех имеющихся данных. Это краевые и граничные условия, данные измерений и само уравнение (1). Подбор параметров модели (весов нейронной сети) осуществлялся на основе минимизации функционала ошибки, составленного из следующих слагаемых.

Первое слагаемое отвечает за соответствие построенного решения $u(x,t)$ исходному уравнению диффузии

$\delta_{eq}=\sum_{i=1}^{m}\left(\frac{\partial u(x_i,t_i)}{\partial t}-\frac{\partial^2 u(x_i,t_i)}{\partial x^2}\right)^2$, где m – число тестовых точек (x_i,t_i);

Второе слагаемое характеризует согласованность решения с данными измерений

$\delta_g=A_1\sum_{i=1}^{m_1}(u(x_g,t_i)-G(t_i))^2$, где m_1 – количество измерений, $0\le t_i\le 1$ – рассматриваемые моменты времени, x_g – точка тоннеля, где находится измеритель, в вычислительных экспериментах рассматривался случай x_g=0.1;

Третье слагаемое отвечает за выполнение граничного условия

$\delta_0 = A_2 \sum_{i=1}^{m_2} \left(u\left(0, \frac{i}{m_2}\right) - G\left(\frac{i}{m_2}\right)\right)^2$, где m_2 – число равноотстоящих точек на всем промежутке времени;

Четвёртое – характеризует соответствие краевому условию

$\delta_1 = A_3 \sum_{i=1}^{m_3} u^2\left(\frac{i}{m_3}, 0\right)$, где m_3 – число равноотстоящих точек на всей длине тоннеля.

Минимизация построенного функционала ошибки

$$\delta = \delta_{eq} + \delta_g + \delta_0 + \delta_1 \quad (5)$$

производится методом облака в сочетании с методом RProp [1]

Для построения приближённого решения первой задачи применяется нейросетевой подход [1-7]. Для данной задачи используется два типа базисных функций. Первый - универсальные функции сигмоидного типа

$$u(x, t, x_c, t_c, a, b, c, d) = c\left(\mathrm{th}\left(a(x - x_c) + b(t - t_c)\right) + d\right),$$

где x_c, t_c, a, b, c, d – параметры (веса) нейронной сети. В результате получается персептрон с одним скрытым слоем. Второй - специальные базисные функции, являющимися решением уравнения диффузии (1),

$$u(x, t, x_c, t_c, a, c) = c\left(\frac{1}{\sqrt{t - t_c}} \exp \frac{-0,25(x - x_c)^2}{t - t_c} + a\right). \quad (6)$$

В последнем случае функционал ошибки (5) примет вид $\delta = \delta_g + \delta_0 + \delta_1$, так как $\delta_{eq} = 0$ автоматически.

Для сравнения приведем результаты работы нейронной сети с задачей, в которой данные генерируются с помощью функции $R_1(x,t)$. Число нейронов для сети персептрон и сети со специальными функциями взято равным 10. На рисунке 1 изображены наложения контурных графиков истинной функции $R_1(x,t)$ и построенных нейросетевых приближений на уровне значения концентрации вредных веществ 0,05. В данном модельном случае этот уровень выступает примером критического, превышение которого опасно для человека. Графики отражают распространение повышенной концентрации по тоннелю от источника (x=1) в сторону выхода (x=0) с течением времени ($0 \le t \le 1$). Как и ожидалось, специальные функции дают более точное решение (несовпадающие области отмечены темным цветом).

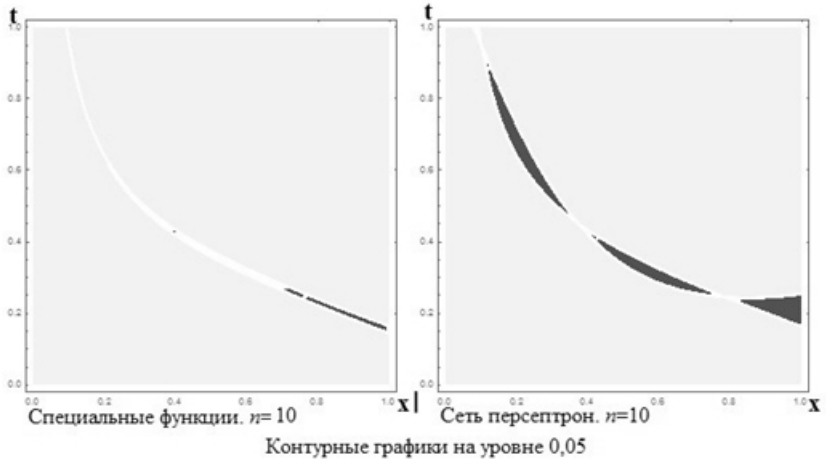


Рис.1 Нейросетевое решение задачи для функции $R_1(x,t)$ в случае точных данных

При увеличении числа нейронов до 20 универсальная сеть дает хорошее приближение функции $R_1(x,t)$ в области, находящейся у выхода из тоннеля, для дальнего конца тоннеля происходит накопление ошибки.

В случае выбора в качестве функции концентрации $R_2(x,t)$ задача решается менее успешно. Значительное увеличение числа нейронов (до 100) сети позволяет построить довольно точное приближение функции концентрации на левой части тоннеля. Подобный результат для $R_1(x,t)$ получается уже при n=20.

Нейросетевое решение второй задачи строится для данных, сгенерированных по формуле (4). В качестве параметров рассматривались $\varepsilon = \alpha = 0{,}03$ и $\beta = 2\alpha$.

Хороший результат, учитывая характер данных и выбранную погрешность, показала нейронная сеть, в которой в качестве базисных функций рассматриваются специальные функции (6).

Соответствующие контурные графики результатов работы сети с n=5 нейронами представлены на рисунке 2 для обоих рассматриваемых источников выделения вредных веществ.

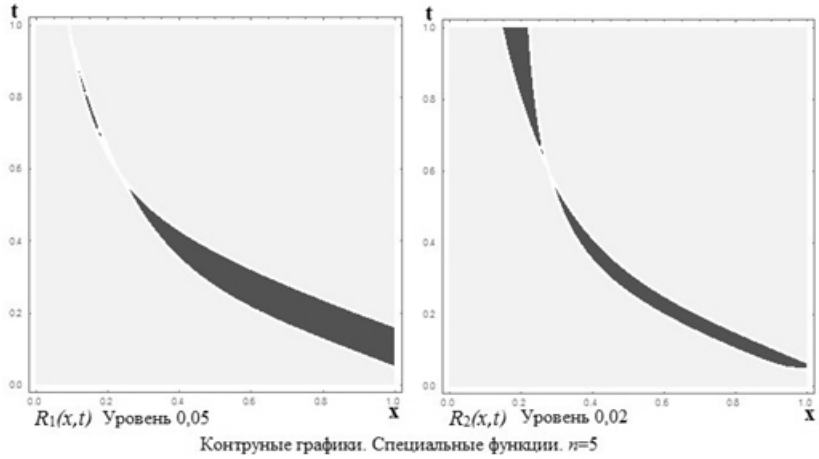


Рис.2 Нейросетевое решение задачи в случае бинарных данных

Отметим, что при увеличении числа нейронов возникает эффект переобучения сети и ошибка возрастает.

Выводы

Как видно из полученных графиков, более точные результату получаются при использовании данных поученных в результате измерений с помощью приборов или датчиков. Расхождение между границами областей с превышениями ПДК, построенными с использованием специальных функций, не значительное.

Однако в случае невозможности использования приборов или экономической нецелесообразности установки датчиков в тоннелях, применимо использование результатов решения второй задачи на основе бинарных данных. Учитывая то, что мы основывались на неточных данных полученных исходя из обонятельной чувствительности человека, полученный в этой задаче результат считаем удовлетворительным.

Работа поддержана Российским фондом фундаментальных исследований (грант №14-01-00660А).

Литература

1. Тархов Д.А. Нейросетевые модели и алгоритмы. М.: Радиотехника, 2014. – 348 с.
2. Васильев А.Н., Тархов Д.А. Нейросетевое моделирование. Принципы. Алгоритмы. Приложения. – СПб.: Изд-во СПбГПУ, 2009. – 528 с.
3. Васильев А.Н., Тархов Д.А. Построение приближённых нейросетевых моделей по разнородным данным// Математическое моделирование. – 2007. – Том 19, №12. – С.43-51
4. Васильев А.Н., Осипов В.П., Тархов Д.А. Унифицированный процесс моделирования физико-технических объектов с распределенными параметрами// Научно-технические ведомости СПбГПУ. Физ.-мат. науки. – 2010. – №3(104). – С.39-52.
5. Васильев А.Н., Тархов Д.А. Параметрические нейросетевые модели классических и неклассических задач для уравнения теплопроводности// Научно-технические ведомости Санкт-Петербургского государственного политехнического университета. Физико-математические науки. – 2012. – Т. 3, № 153. – С.136-144.
6. Tarkhov D.A., Vasilyev A.N. Mathematical Models of Complex Systems on the Basis of Artificial Neural Networks// Nonlinear Phenomena in Complex Systems. – vol.17, №3, 2014, pp. 327-335
7. Ефремов С.В., Каверзнева Т.Т., Тархов Д.А. Нейросетевое моделирование в охране труда. – СПб: Изд-во Политехн. ун-та, 2014. – 136 с.
8. Цодиков В. Я. Вентиляция и теплоснабжение метрополитенов. – Недра, 1975. – 574с.

Пишкинас А.О.[1], Оносов И.А.[1], Корчагин С.А.[2], Романчук С.П.[3], Терин Д.В.[4]

[1]Энгельсский технологический институт (филиал) Саратовского государственного технического университета им. Гагарина Ю.А., г. Энгельс, магистрант
[2]Энгельсский технологический институт (филиал) Саратовского государственного технического университета им. Гагарина Ю.А., г. Энгельс, аспирант, korchaginser@gmail.com
[3]Энгельсский технологический институт (филиал) Саратовского государственного технического университета им. Гагарина Ю.А., г. Энгельс, ассистент кафедры технической физики и информационных технологий, romanchuk_sergey@bk.ru
[4]Энгельсский технологический институт (филиал) Саратовского государственного технического университета им. Гагарина Ю.А., г. Энгельс, к.ф.-м.н., заведующий кафедрой технической физики и информационных технологий, terinden@mail.ru

РАЗРАБОТКА ПРОГРАММНЫХ СРЕДСТВ МОДЕЛИРОВАНИЯ КОМПОЗИТНЫХ НАНОМАТЕРИАЛОВ

КЛЮЧЕВЫЕ СЛОВА

Математическое моделирование, программный комплекс, базы данных, гетерогенные среды, композитные наноматериалы, электродинамические свойства, теория эффективной среды.

АННОТАЦИЯ

В работе представлен разработанный программный комплекс для проведения численного анализа электродинамических свойств композитных наноматериалов, рассмотрена структура базы данных для хранения значений экспериментальных характеристик различных веществ. Приведены результаты моделирования влияния анизотропии формы включений на свойства нанокомпозита. Продемонстрирована зависимость эффективной диэлектрической проницаемости от формы и ориентации включений Au в матрице Al2O3 на основе обобщенной модели Максвелла-Гарнетта.

При создании новых функциональных материалов особую роль играют программные инструменты математического моделирования. Такие программные комплексы позволяют ещё на стадии проектирования провести ряд численных экспериментов по подбору оптимальной морфологии материала. В текущее время ощущается недостаток в инструментах для проведения численного экспресс анализа электродинамических свойств композитных сред. Для широкого применения программных комплексов такого рода при разработке необходимо учесть следующие требования: широкий спектр функциональности, многообразие экспорта и импорта данных, по возможности низкие системные требования, а также простой в освоении визуальный интерфейс.

Моделирование свойств гетерогенных сред проводят на основе различных подходов, например метод эквивалентных схем и метод конечных элементов. Особое место занимает теория эффективной среды [1]. Главным преимуществом данной теории является то, что при расчете электродинамических свойств нанокомпозитной среды нет необходимости решать уравнение Максвелла в каждой точке пространства. Ограничением применимости теории эффективной среды является условие малости размера частиц включений по отношению к длине волны внешнего электромагнитного излучения [2].

Таблица 1. Пример реализованных в программном комплексе моделей

№ п/п	Вид среды	Модель диэлектрической проницаемости гетерогенной среды
1	Слоистая структура, поле направлено параллельно слоям	$\varepsilon = \sum_{i=1}^{n} \varepsilon_i \nu_i$
2	Слоистая структура, поле направлено перпендикулярно слоям	$\varepsilon = 1 \Big/ \sum_{i=1}^{n} \frac{\nu_i}{\varepsilon_i}$

3	Матричная смесь, включения одинакового радиуса расположены в узлах простой кубической решетки, модель Релея	$\varepsilon = \varepsilon_m \left[1 + \frac{\nu}{\frac{\varepsilon_p + 2\varepsilon_m}{\varepsilon_p - \varepsilon_m} - \nu - 1{,}31 \frac{\varepsilon_p - \varepsilon_m}{\varepsilon_p + 4/3\,\varepsilon_m} \nu^{10/3}} \right]$
4	Матричная смесь, модель Лоренц-Лорентца	$\varepsilon = \frac{\varepsilon_m(\varepsilon_p + 2) + 2\nu_p(\varepsilon_m + \varepsilon_p)}{(\varepsilon_p + 2) + \nu_p(\varepsilon_m - \varepsilon_p)}$
5	Матричная смесь, модель Максвелла	$\varepsilon = \varepsilon_m \left[1 + \frac{\varepsilon_p - \varepsilon_m}{2\varepsilon_m + \varepsilon_p} 3\nu \right]$
6	Матричная смесь, модель Максвелла-Гарнетта	$\varepsilon = \frac{2\varepsilon_m^2(1-\nu) + \varepsilon_m\varepsilon_p(1+2\nu)}{\varepsilon_m(2+\nu) + \varepsilon_p(1-\nu)}$
7	Статистическая смесь, модель Бруггемана	$2\varepsilon^2 + \varepsilon\left(\varepsilon_p - 2\varepsilon_m + 3\nu(\varepsilon_m - \varepsilon_p)\right) - \varepsilon_m\varepsilon_p = 0$
8	Статистическая смесь, модель Оделевского	$\varepsilon = a + \sqrt{a^2 + \frac{\varepsilon_1\varepsilon_2}{2}}$ $a = \frac{(3\nu - 1)\varepsilon_1 + (2 - 3\nu)\varepsilon_2}{4}$

Разработанный программный комплекс «Математическое моделирование и многокритериальный анализ нелинейных свойств композиционных материалов на основе эффективной среды» [3] позволяет проводить исследования электродинамических свойств гетерогенных сред на основе ряда моделей. Анализ свойств проводится исходя из введенных в систему данных о параметрах среды: объемные доли компонент, комплексная диэлектрическая проницаемость веществ, форма, структура и ориентация в пространстве частиц включений.

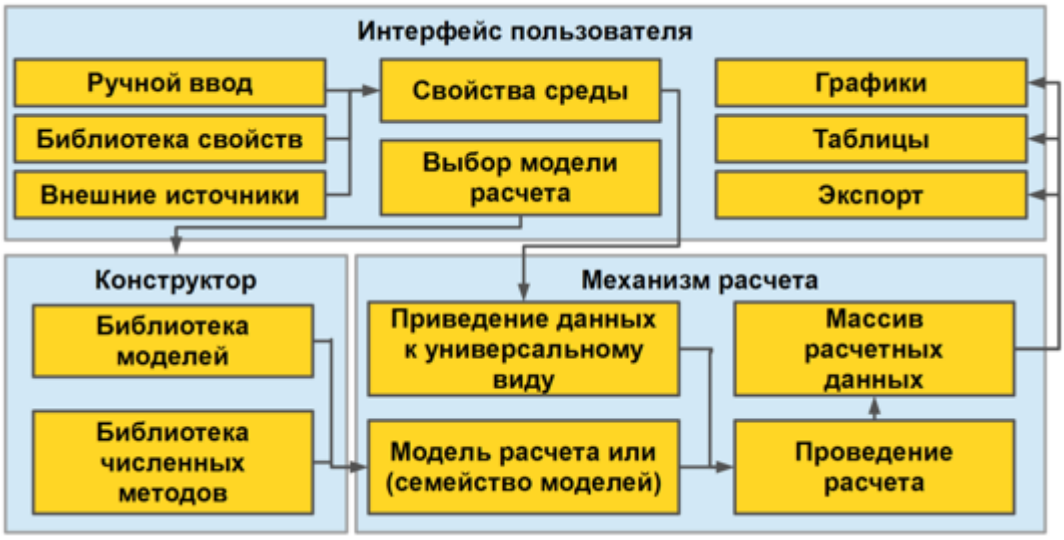


Рис. 1 Структура программного комплекса

Модульность разработки (рис. 1) позволяет расширять функционал, в особенности важным фактом является возможность добавления новых моделей гетерогенных сред и новых численных методов. На данный момент программно реализовано более 20 моделей (Релея, Лихтенекера, Лоренц-Лорентца, Максвелла-Гарнетта, Бруггемана, Оделевского и др.) для различной морфологии композитов. Часть моделей представлена в таблице 1. При разработки библиотеки численных методов были программно реализованы алгоритмы поиска комплексных корней полиномов [4].

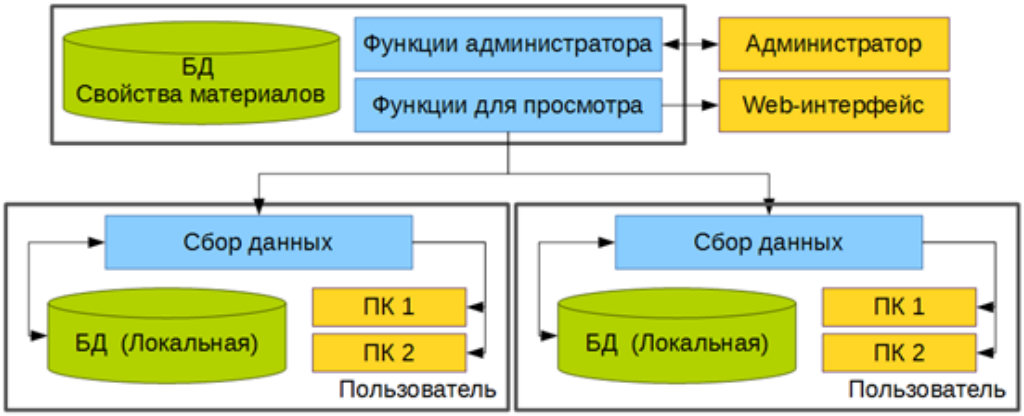


Рис. 2 Структура взаимодействия с использованием единой базы данных свойств материалов

В ходе разработки программного комплекса обнаружилась потребность в источнике исходных данных свойств материалов. В такой роли наиболее удобно использовать табличные данные экспериментальных исследований свойств материалов. Для этих целей спроектирована схема работы нескольких программных комплексов с использованием единой базы данных свойств материалов (рис. 2).

Удаленная база данных свойств материалов имеет интерфейс для выполнения административных функций и интерфейс для выгрузки данных внешнему клиенту. В свою очередь, программный комплекс на стороне клиента получает данные через унифицированный модуль сбора данных из глобальной базы данных или из локальной. Взаимосвязь приложений с удаленной базой данных происходит посредством локальной или глобальной сети [5].

Такой подход позволяет создать инструментальную среду для проведения различного рода исследований в области анализа свойств гетерогенных сред и осуществить возможность удаленного подключения к всегда актуальной базе данных свойств материалов.

При исследовании эффективных характеристик композитного наноматериала особый интерес представляет обобщенная модель Максвелла-Гарнетта [6] с учетом фактора деполяризации [7], т.к. модель охватывает широкий спектр форм частиц и учитывает ориентацию включений в пространстве. Данную модель применяют к матричным средам с включениями в форме сфероидов, что позволяет использовать в качестве частиц включений пластины, стержни, сферы и эллипсоиды вращения [8].

Рассмотрим варианты композитов с ориентированными частицами нормально вектору электромагнитного поля, коллинеарно вектору электромагнитного поля и с хаотически расположенными частицами. Для учета анизотропии формы включений вычисляется фактор деполяризации:

$$N_x = \frac{abc}{3}\int_0^{\infty} \frac{ds}{(s+a^2)^{3/2}(s+b^2)^{1/2}(s+c^2)^{1/2}}, \qquad (1)$$

где a, b, c - длина полуосей эллипсоида вдоль осей координат x, y, z.

Для одиночного эллипсоида с диэлектрической проницаемостью ε_p, который находится в однородной среде с диэлектрической проницаемостью ε_m, напряженность поля E_1 внутри эллипсоида при направлении внешнего поля E_0 вдоль оси x выражается следующим уравнением [7]:

$$E_1 = \frac{\varepsilon_m}{\varepsilon_m + (\varepsilon_p - \varepsilon_m) N_x} E_0 . \qquad (2)$$

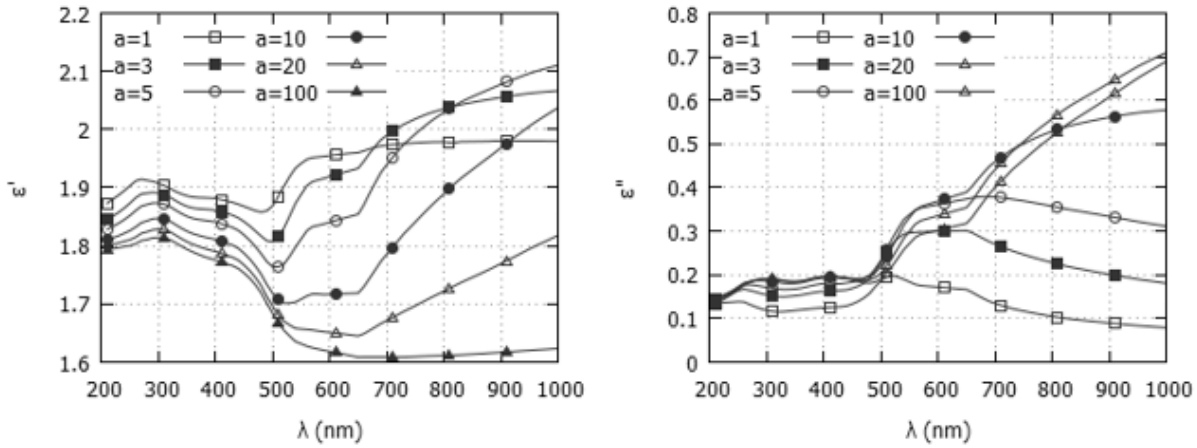


Рисунок 1. Частотная зависимость диэлектрической проницаемости композита при коллинеарном расположении частиц относительно вектора поля

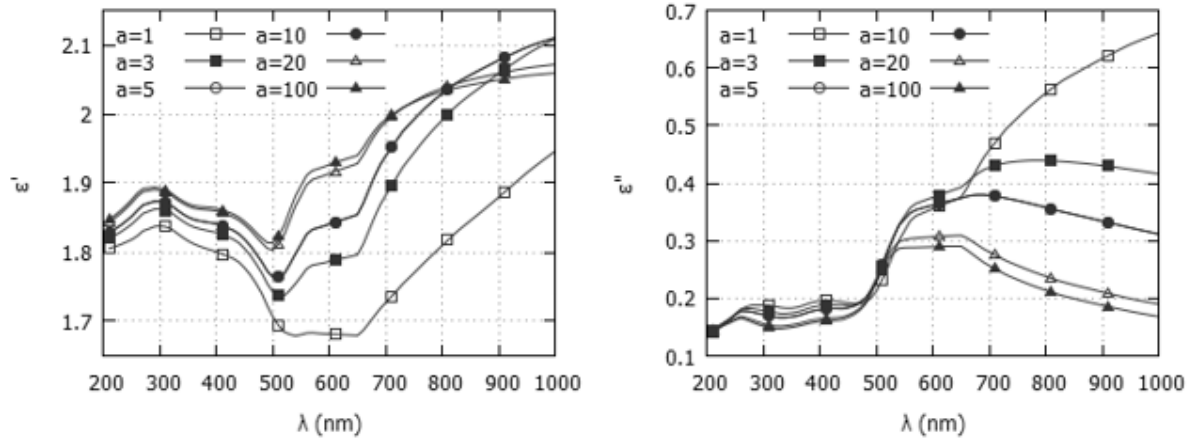


Рисунок 2. Частотная зависимость диэлектрической проницаемости композита при нормальном расположении частиц относительно вектора поля

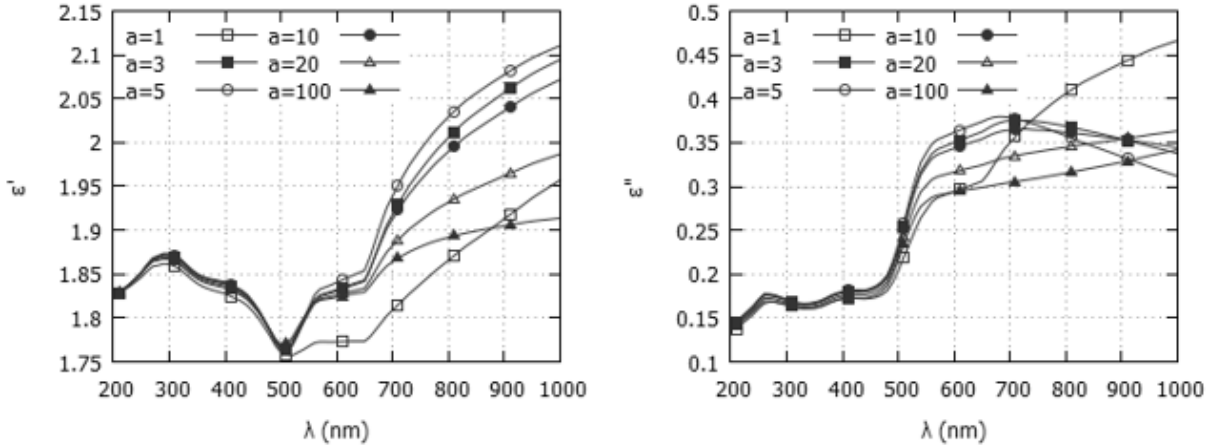


Рисунок 3. Частотная зависимость диэлектрической проницаемости композита при хаотическом расположении частиц относительно вектора поля

Факторы деполяризации для вытянутого и сплюснутого эллипсоида вращения (сфероид) можно получить по следующим соотношениям соответственно (3) и (4) [7].

$$N_x = \frac{(1-\alpha^2)(Arth(\alpha)-\alpha)}{\alpha^3}; \qquad (3)$$

$$N_x = \frac{(1-\beta^2)(\beta - arctg(\beta))}{\beta^3}, \quad (4)$$

где $\alpha=(1 - b_2/a_2)1/2$, $\beta=(b_2/a_2-1)1/2$.

Используя выражения (2), (3) и (4) исследовано влияние анизотропии формы включений в матричном композите на эффективную диэлектрическую проницаемость материала. Численное моделирование проводилось для смеси проводящих наночастиц золота (Au) взвешенных в диэлектрической матрице оксида алюминия (Al_2O_3), объемная доля включений – 0.1, при диапазоне длин волн от 200 нм до 1000 нм. Форма частиц от сплюснутых (a=1 нм, a=3 нм, b=5 нм) до сильно вытянутых (a=10 нм, a=20 нм, a=100 нм, b=5 нм) сфероидов.

Графики 1, 2 и 3 демонстрируют зависимость свойств композита от ориентации и формы наночастиц. Значения диэлектрической проницаемости среды с включениями форме сфер (a=5 нм, b=5 нм) на всех графиках остается неизменными.

Литература

1. Виноградов А.П. Электродинамика композитных материалов./А.П. Виноградов. -М.: Эдиториал УРСС, 2001, - 208 с.
2. Головань Л.А., Тимошенко В.Ю., Кашкаров П.К. Оптические свойства нанокомпозитов на основе пористых систем// Успехи физических наук . 2007. Т. 177, № 6. С. 619-638.
3. Свидетельство № 2014615533 Российская Федерация, Программный комплекс "Математическое моделирование и многокритериальный анализ нелинейных свойств композиционных материалов на основе моделей эффективной среды": Свидетельство о государственной регистрации программы для ЭВМ / Романчук С.П., Терин Д.В. ; заявитель и патентообладатель Романчук Сергей Петрович, Терин Денис Владимирович. - № 2014612918/69 ; заявл. 02.04.2014 ; зарегистр. 28.05.2014. - [1] с.
4. Исследование численных схем одновременного поиска корней полиномов с комплексными коэффициентами применительно к моделям эффективной среды /Романчук С.П., Терин Д.В., Шатурная О.С. // Вестник Саратовского государственного технического университета. 2013. Т. 4. № 1 (73). С. 181-188.
5. Комплекс программ для проведения исследований композитных сред различной структуры / Оносов И.А., Пишкинас А.О., Корчагин С.А., Романчук С.П.,Терин Д.В.// Математическое моделирование и информационные технологии в научных исследованиях и образовании. Саратов, 2015. С. 86-89.
6. Maxwell-Garnett J C Philos. Trans. R. Soc. London 203 385 (1904)
7. Нетушил А.В. и др. Высокочастотный нагрев в электрическом поле. М.: Высшая школа, 1961.
8. Челидзе Т.Л., Деревянко А.И., Кириленко О.Д. Электрическая спектроскопия гетерогенных систем. Киев: Наукова думка, 1977. 230 с.

Семененко М.Г.

Калужский филиал МГТУ им. Н.Э.Баумана, г.Калуга, к.ф.-м.н., доцент кафедры высшей математики,
msemenenko2009@yandex.ru

МОДЕЛЬ МАРКОВИЦА: МАТЕМАТИЧЕСКИЕ АСПЕКТЫ И КОМПЬЮТЕРНАЯ РЕАЛИЗАЦИЯ

КЛЮЧЕВЫЕ СЛОВА

Инвестиционный анализ, портфельная теория, модель Марковица, электронные таблицы Excel.

АННОТАЦИЯ

В статье рассматриваются вопросы, связанные с изучением модели Марковица. Для компьютерной реализации модели предлагается использовать электронные таблицы Excel. Приводится пример практических расчетов.

Введение

В различных курсах экономического вуза рассматриваются элементы инвестиционного анализа. В частности, следующая задача: составить оптимальный инвестиционный портфель, составленный из заданных активов (например, акций). Теоретические основы решения данной задачи заложены в портфельной теории Марковица, сформулированной в 1950-х годах, которая предлагает методику формирования инвестиционного портфеля с оптимальным выбором долей заданных активов. С точки зрения инвестора, оптимальный портфель должен обеспечивать максимально возможную доходность с минимальным риском. Однако математически такая постановка задачи является противоречивой и решения не имеет. Поэтому, как правило, рассматривается один из двух возможных вариантов данной задачи:

- рассчитать доли активов, обеспечивающих максимальную доходность при заданном уровне риска;
- рассчитать доли активов, обеспечивающих минимальный риск при заданном уровне доходности.

В данной статье рассматриваются математические аспекты данной задачи и компьютерная реализация ее решения в электронных таблицах Excel.

Математические аспекты модели Марковица

Теоретической базой модели Марковица является аппарат теории вероятностей. В рамках модели доходность каждого актива рассматривается как случайная величина, соответственно, доходность портфеля является случайным вектором.

В основе модели лежат следующие предпосылки:

- стационарность модели «реального мира». Множество факторов, от которых зависит конкретное значение доходности финансового актива, принято называть «состоянием экономики». Предполагается, что число состояний конечно и экономика может находиться в одном из этих известных состояний, число которых равно n (например, в одном из трех состояний: предкризисное, кризис, нормальное). Каждому состоянию приписывается некоторое положительное число, которое является вероятностью данного состояния. Предполагается, что вероятности состояний экономики и сами состояния не меняются;
- статичность модели. Портфель формируется в момент t = 0 и до получения финансового результата в момент t = T не меняется.

Основными числовыми характеристиками доходности актива являются его математическое ожидание и дисперсия, которая характеризует риск, связанный с приобретением данного актива. При решении практических задач также удобно использовать среднее квадратическое отклонение (ско) доходности, которое вычисляется как корень из дисперсии.

Пусть портфель включает N ценных бумаг. Обозначим долю j-й бумаги в портфеле через x_j. Доходность портфеля R_p вычисляется как средневзвешенное значение доходностей ценных бумаг, включенных в портфель, причем в качестве весов используются соответствующие доли:

$$R_p = \sum_{j=1}^{N} R_j x_j .$$

Здесь R_j – доходность j-й ценной бумаги в портфеле.

Тогда математическое ожидание доходности портфеля m_p равно

$$m_p = \sum_{j=1}^{N} m_j x_j ,$$

где m_j – математическое ожидание доходности j-й ценной бумаги в портфеле.

Дисперсия доходности портфеля V_p рассчитывается по формуле

$$V_p = \sum_{i=1}^{N} \sum_{j=1}^{N} V_{ij} x_i x_j ,$$

где V_{ij} – ковариация доходностей ценных бумаг в портфеле.

Рассмотрим один из возможных вариантов математической постановки задачи согласно модели Марковица, соответствующий условию минимизации риска при заданном уровне доходности портфеля:

$$V_p = \sum_{i=1}^{N} \sum_{j=1}^{N} V_{ij} x_i x_j \quad ,$$
$$\sum_{j=1}^{N} m_j x_j = m_p ,$$
$$\sum_{j=1}^{N} x_j = 1 .$$

В матричной форме данную задачу можно записать как

$X^T V X \rightarrow \min$,

$m^T X = m_p$,

$I^T X = 1$.

Здесь $X = (x_1\ x_2\ \ldots\ x_N)^T$, I - единичная матрица, $m = (m_1\ m_2\ \ldots\ m_N)^T$.

Один из методов решения данной задачи на условный экстремум - метод Лагранжа. Чтобы записать функцию Лагранжа, необходимо ввести два множителя Лагранжа λ_1 и λ_2в соответствии с числом ограничений задачи. Тогда функция Лагранжа имеет вид:

$$L(X, \lambda_1, \lambda_2) = X^T V X + \lambda_1 (m^T X - m_p) + \lambda_2 (I^T X - 1) .$$

Точка безусловного экстремума функции Лагранжа совпадает с точкой условного экстремума исходной задачи. Чтобы найти эту точку необходимо найти частные производные функции Лагранжа по всем переменным, приравнять их нулю и решить соответствующую систему алгебраических уравнений. В матричной форме полученная система алгебраических уравнений будет иметь вид

$$\begin{pmatrix} 2V & m & I \\ m^T & 0 & 0 \\ I^T & 0 & 0 \end{pmatrix} \begin{pmatrix} X \\ \lambda_1 \\ \lambda_2 \end{pmatrix} = \begin{pmatrix} O \\ m_p \\ 1 \end{pmatrix} . \qquad (1)$$

Здесь V – ковариационная матрица, m – вектор-столбец ожидаемых доходностей, I – вектор-столбец, все N элементов которого равны единице, $\boldsymbol{O}$ – вектор-столбец, все N элементов которого равны нулю.

Введем обозначение:

$$A = \begin{pmatrix} 2V & m & I \\ m^T & 0 & 0 \\ I^T & 0 & 0 \end{pmatrix} , \quad X = \begin{pmatrix} X \\ \lambda_1 \\ \lambda_2 \end{pmatrix} , \quad b = \begin{pmatrix} \boldsymbol{O} \\ m_p \\ 1 \end{pmatrix} .$$

Тогда система (1) имеет вид

$$A \cdot x = b .$$

Решение этой системы

$$x = A^{-1} \cdot b . \qquad (2)$$

Эффект диверсификации

Если доходности различных видов ценных бумаг в портфеле взаимно-независимы, то риск (ско) портфеля тем меньше, чем больше N. Это свойство портфеля ценных бумаг называется эффектом диверсификации.

Эффективная диверсификация портфеля по Марковицу предусматривает такое распределение ценных бумаг в портфеле, которое соответствует минимальному уровню риска при заданной доходности.

Компьютерная реализация модели Марковица

Компьютерную реализацию модели Марковица удобно выполнить в электронных

таблицах Excel (можно также использовать эектронные таблицы других производителей программного обеспечения, например, PlanMaker в Ashampoo Office). Рассмотрим несколько примеров.

Пример 1.

Получим ковариационную матрицу доходности двух активов А и В. После ввода данных вычисляем среднее значение с помощью функции СРЗНАЧ. Затем вычисляем сумму произведений отклонений от среднего (рис.1, данные взяты из [2]). При составлении ковариационной матрицы учитывается, что cov(A,B) = cov(B,A) и cov(A,A) = σ^2(A). Диагональные элементы матрицы можно вычислить с помощью функции ДИСПР.

Активы						**Среднее**
Период	1	2	3	4	5	
Доходность А,%	10	-12	23	11	-2	**6**
Доходность В,%	14	3	17	22	6	**12,4**
Ковариация						**Итого**
Отклонения А	4	-18	17	5	-8	
Отклонения В	1,6	-9,4	4,6	9,6	-6,4	
Произведение	6,4	169,2	78,2	48	51,2	**70,6**

	A	*B*
A	143,6	70,6
B	70,6	49,04

Рис. 1. Пример вычисления ковариационной матрицы

В электронных таблицах Excel для вычисления ковариационной матрицы можно также использовать опцию Ковариация меню Анализ данных (вкладка Данные), выбрав Группирование по строкам.

Пример 2.

Найдем структуру оптимального портфеля, состоящего из акций трех предприятий с известной динамикой доходностей (данные взяты из [2]).

Вычисление средней доходности акций показано на рис. 2.

	A	B	C	D	E	F	G	H	I	J	K	L
1	**Эмитент**	**Янв**	**фев**	**мар**	**апр**	**май**	**июн**	**июл**	**авг**	**сен**	**окт**	**Среднее**
2	**Акрон**	0,13	0,12	-0,01	-0,11	-0,19	-0,02	0,05	0,11	0,02		**0,0111111**
3	**Автоваз**	-0,01	0,01	0,32	-0,03	-0,21	0,04	0,1	0,4	0,06	-0,03	**0,065**
4	**Дорогобуж**	0,01	0,08	0,09	-0,1	-0,12	-0,05	-0,03	0,03	0,06	-0,01	**-0,004**
5												
6												

Рис. 2. Ввод данных и вычисление среднего значения доходности

Матрица ковариации *V* вычисляется аналогично примеру 1. Затем формируется матрица А и вычисляется обратная матрица с помощью функции МОБР. Оптимальная структура портфеля определяется по формуле (2) с помощью функции умножения матриц МУМНОЖ.

Фрагмент вычислений показан на рис. 3. Состав оптимального портфеля: ~17,5% акций Акрона, 31% акций АвтоВАЗа и ~51,5% акций Дорогобужа.

Следует отметить следующие особенности решения задачи по предлагаемой методике.

Операция обращения матриц, вообще говоря, численно неустойчива. Если матрица плохо обусловлена [3], в частности, близка к вырожденной, результаты вычислений сильно зависят от различных вычислительных погрешностей, особенностей работы программы и т.п. В частности, у матрицы из примера 2 определитель равен 0,005. Для плохо обусловленных матриц результатам вычислений доверять нельзя. Для примера 2 в этом можно убедиться, проведя вычисления в программах Excel и Plan Maker.

	A	B	C	D	E	F	G
8		**Акрон**	**Автоваз**	**Дорогобуж**		**mp =**	**0,02**
9	**Акрон**	0,0104321	0,0090827	0,00534815		**m1**	0,0111111
10	**Автоваз**	0,0090827	0,028145	0,00712		**m2**	0,065
11	**Дорогобуж**	0,0053481	0,00712	0,004684		**m3**	-0,004
12							
13							
14		**Матрица А**					
15							
16	0,0208642	0,0181654	0,0106963	0,01111111	1		
17	0,01816543	0,05629	0,01424	0,065	1		
18	0,0106963	0,01424	0,009368	-0,004	1		
19	0,01111111	0,065	-0,004	0	0		
20	1	1	1	0	0		
21							
22		**Обратная матрица**					
23							
24	105,427721	-23,08884	-82,33888	8,47227259	0,0063387		0
25	-23,088841	5,0564933	18,032347	12,6373123	0,0565828		0
26	-82,33888	18,032347	64,306533	-21,109585	0,9370785	**b =**	0
27	8,47227259	12,637312	-21,10958	-7,1280237	-0,101335		0,02
28	0,00633867	0,0565828	0,9370785	-0,1013348	-0,010057		1
29							
30		**0,1757841**					
31		**0,3093291**					
32	**X =**	**0,5148868**					
33		-0,243895					
34		-0,012084					
35							
36	**Контроль**	1					

Рис. 3. Фрагмент вычислений оптимального портфеля ценных бумаг

Наши вычисления с различными исходными данными (в том числе с хорошо обусловленными матрицами) также показали следующее.

Доходность портфеля должна лежать в пределах некоторого интервала, в противном случае решение системы может содержать отрицательные значения долей.

Выводы

Модель Марковица является удобным инструментом для анализа экономических задач математическими методами в различных курсах финансово-экономических ВУЗов. На примере этой модели можно показать практическое применение таких элементов высшей математики как матричное исчисление (нахождение обратной матрицы, произведение матриц), численные методы и особенности их применения, понятие устойчивости численного решения математических задач и т.п.

Результаты проведенного нами моделирования позволяют сформулировать следующие выводы, которым уделяется мало внимания в существующих учебниках.

Применение модели Марковица в практических примерах обязательно должно сопровождаться исследованием обусловленности матрицы А в системе (1), в частности, вычислением определителя этой матрицы.

Исходные данные задачи (1) должны удовлетворять некоторым дополнительным требованиям. В частности, значение ожидаемой доходности портфеля должно лежать в некотором интервале. В противном случае решение задачи может содержать отрицательные значения долей ценных бумаг в портфеле.

Литература

1. Бабешко Л.О. Математическое моделирование финансовой деятельности. Уч.пос. – М.: «КноРус», 2013. – 212с.
2. Ремезов М. Семь раз отмерь, чтобы не отрезали//D, №23, 2010.
3. Семушин И. В. Численные методы алгебры: учебное пособие для вузов. – Ульяновск: УлГТУ, 2006. – 178 с.

Сенчилов В.В.

Смоленский государственный университет, г. Смоленск, к.ф.-м.н., доцент кафедры информатики,
senchilov.vv@gmail.com

О РЕШЕНИИ ОДНОЙ ВИДОИЗМЕНЕННОЙ КРАЕВОЙ ЗАДАЧИ ТИПА НЕЙМАНА В КЛАССАХ МЕТААНАЛИТИЧЕСКИХ В КРУГЕ ФУНКЦИЙ С ПРИМЕНЕНИЕМ СИСТЕМЫ КОМПЬЮТЕРНОЙ МАТЕМАТИКИ MAPLE

КЛЮЧЕВЫЕ СЛОВА

Краевая задача, метаналитическая функция, условие нетеровости, система компьютерной математики Maple.

АННОТАЦИЯ

В статье рассмотрен алгоритм решения видоизмененной краевой задачи типа Неймана для метааналитических функций в круге при помощи общего подхода, основывающегося на представлении метааналитических функций через аналитические функции комплексного переменного, а также на теории задачи типа Римана для аналитических функций. Описаны необходимые и достаточные условия разрешимости задачи типа Неймана, а также указаны условия, при которых рассматриваемая задача является нетеровой. Описана процедура, реализующая этот алгоритм в среде Maple.

1. Постановка задачи. Пусть T^+ – односвязная область на плоскости комплексного переменного $z=x+iy$, ограниченная простым гладким замкнутым контуром L, уравнение которого имеет вид: $t = x(s) + iy(s)$, $0 \le s \le l$, где s – натуральный параметр. Через T^- будем обозначать дополнение $T^+ \cup L$ до полной комплексной плоскости.

Напомним (см., например, [1], [4]), что функция $F(z)=U(x,y)+i\,V(x,y)$ называется метааналитической в области T^+, если она имеет в T^+ непрерывные частные производные (по x и y) до второго порядка включительно и удовлетворяет там уравнению

$$\frac{\partial^2 F}{\partial \bar{z}^2} + a_1 \frac{\partial F}{\partial \bar{z}} + a_0 F = 0, \tag{1}$$

где $\partial/\partial\bar{z}=(\partial/\partial x+i\,\partial/\partial y)/2$ – дифференциальный оператор Коши-Римана, а a_0 и a_1 – некоторые комплексные постоянные.

Как известно (см. [1], с. 139), если λ_1 и λ_2 – корни квадратного уравнения

$$\lambda^2 + a_1\lambda + a_0 = 0, \tag{2}$$

то всякую метааналитическую в области T^+ функцию $F^+(z)$ можно задать в виде

$$F^+(z)=[\,\phi_0^+(z)+\bar{z}\,\phi_1^+(z)\,]\exp\{\lambda_0\bar{z}\}\ ,\text{если}\ \lambda_1=\lambda_2=\lambda_0\ , \tag{3}$$

или

$$F^+(z)=\phi_0^+(z)\exp\{\lambda_1\bar{z}\}+\phi_1^+(z)\exp\{\lambda_2\bar{z}\},\ \text{если}\ \lambda_1\neq\lambda_2 \tag{4}$$

где $\phi_k^+(z)$ $(k=0,\,1)$ – аналитические в T^+ функции, называемые аналитическими компонентами метааналитической функции $F^+(z)$.

Рассматривается следующая краевая задача. *Требуется найти все метааналитические функции $F^+(z)$ класса $M_2(T^\pm)\cap H^{(2)}(L)$, удовлетворяющие на L следующему краевому условию:*

$$\frac{\partial F^+(t)}{\partial n}+G(t)\,\overline{F^+(t)}=g(t), \tag{5}$$

где $\partial/\partial n$ – производная по внутренней нормали к L, а $G(t)$, $g(t)$ – заданные на L функции класса H(L).

Отметим сразу, что при $G(t)\equiv 0$ задача (5) представляет собой *задачу Неймана* в классе метааналитических функций. Поэтому при $G(t)\neq 0$, $t\in L$, сформулированную задачу будем

называть *видоизмененной задачей типа Неймана для метааналитических функций* или короче *задачей N*. При этом, если $g(t)\equiv 0$, то задачу (5) называем *однородной.*

В данной статье будет установлено, что задача Неймана (т.е. задача (5) при $G(t)\equiv 0$) в классе метааналитических функций *не является нетеровой* (см. также [2], [3]). Однако, как будет установлено ниже, видоизмененная задача типа Неймана для метааналитических функций *является нетеровой.*

Всюду в дальнейшем в качестве области T^+ будем рассматривать единичный круг с центром в начале координат, т.е. $T^+=\{t:|t|<1\}$.

Сначала покажем, что задача Неймана для метааналитических функций не является нетеровой. Для этого достаточно заметить, что все метааналитические в круге $T^+=\{t:|t|<1\}$ функции вида

$$F^+(z)=(z^{n-1}+\bar{z}\cdot z^n)\exp\{\lambda_0\bar{z}\},$$

где n – произвольное натуральное число, а λ_0 – любое комплексное число, являются решениями однородной задачи Неймана (5). Следовательно, число линейно независимых решений однородной задачи Неймана не является конечным, а значит, задача Неймана для метааналитических функций не нетерова.

2. О решении видоизмененной задачи Неймана в круге. Для полноты исследования необходимо рассмотреть два случая в зависимости от того, в каком виде будем искать решение задачи: в виде (3) или (4). Однако в данном сообщении будем рассматривать лишь первый случай, т.е. случай $\lambda_1=\lambda_2=\lambda_0$, и будем искать решение задачи **N** в виде (3).

Так как (см., например, [1], с. 102)

$$\frac{\partial}{\partial n}=i\left(t'\frac{\partial}{\partial t}-\overline{t'}\frac{\partial}{\partial \bar{t}}\right),\quad t'=\frac{dt}{ds},\tag{6}$$

и $t'=it$, $\bar{t}=\frac{1}{t}$, $t\in L$, то с учетом представления (3) краевое условие (5) можно переписать в виде

$$t\frac{d\phi_0^+(t)}{dt}+\frac{d\phi_1^+(t)}{dt}+\frac{\lambda_0}{t}\phi_0^+(t)+\frac{t+\lambda_0}{t^2}\phi_1^+(t)-\widetilde{G}_1(t)[\overline{\phi_0^+(t)}+t\,\overline{\phi_1^+(t)}]=\widetilde{g}_1(t)\,,\tag{7}$$

где

$$\widetilde{G}_1(t)=G(t)\cdot e^{\overline{\lambda_0}t-\lambda_0\bar{t}},\quad \widetilde{g}_1(t)=-g(t)\cdot e^{-\lambda_0\bar{t}}\,.\tag{7a}$$

Вводя в рассмотрение новые аналитические соответственно в T^+ и T^- функции вида

$$\Phi^+(z)=z^3\frac{d\phi_0^+(z)}{dz}+z^2\frac{d\phi_1^+(z)}{dz}+\lambda_0 z\phi_0^+(z)+(z+\lambda_0)\phi_1^+(z),\quad z\in T^+,\tag{8}$$

$$\Phi^-(z)=\frac{1}{z}\overline{\phi_0^+}\left(\frac{1}{z}\right)+\overline{\phi_1^+}\left(\frac{1}{z}\right),\quad z\in T^-,\tag{9}$$

краевое условие (7) можно записать так:

$$\Phi^+(t)=G_1(t)\Phi^-(t)+g_1(t),\quad t\in L,\tag{10}$$

где $G_1(t)=t^3\widetilde{G}_1(t)$, $g_1(t)=-t^2\widetilde{g}_1(t)\cdot e^{-\lambda_0\bar{t}}$.

Замечание 1. Отметим, что для всякого z, не лежащего на контуре L, из равенства (9) вытекает “условие симметрии”:

$$\Phi^-(z)=\overline{z_*\phi_0^+(z_*)}+\overline{\phi_1^+(z_*)}\,,\tag{*}$$

где z_* – точка, симметричная z относительно контура L.

Равенство (10) есть краевое условие скалярной задачи Римана относительно кусочно-аналитической функции $\Phi(z)=\{\Phi^+(z),\Phi^-(z)\}$.

Пусть $\chi_1=IndG_1(t)$. Тогда, как известно (см., например, [1], с. 54), при $\chi_1\geq 0$ решение задачи Римана (10) задается в виде

$$\Phi^{\pm}(z)=X^{\pm}(z)\left[\Psi^{\pm}(z)+P_{\chi_1}(z)\right], \tag{10а}$$

где $X^{\pm}(z)$ – канонические функции задачи, $\Psi(z)=\frac{1}{2\pi i}\int_L \frac{g_1(\tau)}{X^{+}(\tau)}\frac{d\tau}{\tau-z}$, а $P_{\chi_1}(z)$ – многочлен степени не выше χ_1 с произвольными комплексными коэффициентами. Если же $\chi_1<0$, то решение задачи (10) по-прежнему задается формулой (10а), при выполнении следующих $-\chi_1$ условий разрешимости:

$$\int_L \frac{g_1(\tau)}{X^{+}(\tau)}\tau^{k-1}d\tau=0,\ \ k=1,\ 2,\ \ldots,\ -\chi_1. \tag{10б}$$

Следовательно, решая задачу (10) (в случае ее разрешимости) определим аналитические функции $\Phi^{+}(z)$ *и* $\Phi^{-}(z)$ по формулам (10а).

Далее, заменяя z на $1/\overline{\zeta}$ из (9) будем иметь:

$$\overline{\zeta}\cdot\overline{\phi_0^{+}(\zeta)}+\overline{\phi_1^{+}(\zeta)}=\Phi^{-}(1/\overline{\zeta})\ \ \textit{или}\ \ \zeta\phi_0^{+}(\zeta)+\phi_1^{+}(\zeta)=\overline{\Phi^{-}(1/\overline{\zeta})},\ |\zeta|<1. \tag{11}$$

Наконец, заменив в (11) ζ на z и умножив обе части полученного равенства на z^2, получим

$$z^3\phi_0^{+}(z)+z^2\phi_1^{+}(z)=z^2\overline{\Phi^{-}}(1/z),\ \ z\in T^{+}. \tag{12}$$

Дифференцируя по z из (12) находим

$$z^3\frac{d\phi_0^{+}(z)}{dz}+z^2\frac{d\phi_1^{+}(z)}{dz}+3z^2\phi_0^{+}(z)+2z\phi_1^{+}(z)=\frac{d}{dz}\left[z^2\overline{\Phi^{-}}(1/z)\right]. \tag{13}$$

В свою очередь, из (8)

$$z^3\frac{d\phi_0^{+}(z)}{dz}+z^2\frac{d\phi_1^{+}(z)}{dz}+\lambda_0(z\phi_0^{+}(z)+\phi_1^{+}(z))+z\phi_1^{+}(z)=\Phi^{+}(z). \tag{14}$$

Вычитая из (13) равенство (14) и учитывая в полученном равенстве соотношение (12), будем иметь

$$z^2\varphi_0^{+}(z)-z\varphi_1^{+}(z)=(\lambda_0-2z)\cdot\overline{\Phi^{-}}(1/z)+\frac{d}{dz}\left[z^2\overline{\Phi^{-}}(1/z)\right]-\Phi^{+}(z). \tag{15}$$

На основании равенств (12) и (15) получается следующая система

$$\begin{cases} z^2\phi_0^{+}(z)+z\phi_1^{+}(z)=z\,\overline{\Phi^{-}}(1/z),\ \ z\in T^{+},\\ z^2\phi_0^{+}(z)-z\phi_1^{+}(z)=\Phi_1^{+}(z),\ \ z\in T^{+}, \end{cases} \tag{16}$$

где $\Phi_1^{+}(z)=(\lambda_0-2z)\cdot\overline{\Phi^{-}}(1/z)+\frac{d}{dz}\left[z^2\overline{\Phi^{-}}(1/z)\right]-\Phi^{+}(z)$.

Решая систему (16) относительно $\phi_0^{+}(z)$ *и* $\phi_1^{+}(z)$ получаем

$$\phi_0^{+}(z)=\frac{1}{2z^2}W_0^{+}(z),\ \ \phi_1^{+}(z)=\frac{1}{2z}W_1^{+}(z), \tag{17}$$

где

$$W_0^{+}(z)=z\overline{\Phi^{-}}(1/z)+\Phi_1^{+}(z),\ \ W_1^{+}(z)=z\,\overline{\Phi^{-}}(1/z)+\Phi_1^{+}(z),\quad z\in T^{+}. \tag{17а}$$

Из (17 и (17а) видно, что для того, чтобы функции $\varphi_0^{+}(z)$ и $\varphi_1^{+}(z)$ были аналитическими в круге T^{+} (т.е. для разрешимости задачи **N**) функция $W_0^{+}(z)$ должна иметь в точке z = 0 нуль не ниже второго порядка, а функция $W_1^{+}(z)$ - нуль не ниже первого порядка. Очевидно, что последние условия можно записать соответственно в виде:

$$\int_L \frac{W_0^+(t)}{t^p}dt=0,\quad p=1,\,2,$$
$$\int_L \frac{W_1^+(t)}{t}dt=0. \qquad (18)$$

Замечание 2. Важно отметить, что при $\chi_1\geq 0$ выражения для функций $W_0^+(z)$ и $W_1^+(z)$ (в силу (10а) и (17а)) линейно зависят от χ_1+1 произвольных комплексных постоянных. Поэтому равенства (18) можно записать в виде трех линейных алгебраических уравнений относительно указанных произвольных постоянных. Следовательно, выполнение условий (18) можно добиваться за счет определенного выбора некоторых произвольных постоянных, входящих в выражения $W_0^+(z)$ и $W_1^+(z)$.

В силу (3), при выполнении условий (18) (а также условий (10б) в случае $\chi_1<0$), решение исходной задачи **N** можно задавать формулой

$$F^+(z)=\left[\frac{1}{2z^2}W_0^+(z)+\bar{z}\frac{1}{2z}W_1^+(z)\right]\cdot e^{\lambda_0\bar{z}}, \qquad (19)$$

где $W_0^+(z),\ W_1^+(z)$ определяются по формулам (17а).

Таким образом, получили следующий результат.

Теорема 1. *Пусть* $T^+=\{z:\ |z|<1\}$ *и уравнение (2) имеет один (двукратный) корень. Тогда решение задачи* ***N*** *сводится к решению обычной скалярной задачи Римана (10). Кроме того, если* $\chi_1\geq 0$ *, то задача* ***N*** *разрешима тогда и только тогда, когда выполняются три условия вида (18) и ее общее решение, задаваемое формулой (19), линейно зависит не менее чем* χ_1-3 *(при* $\chi_1>3$ *) произвольных комплексных постоянных. Если же* $\chi_1<0$ *, то для разрешимости задачи* ***N*** *необходимо и достаточно выполнение условий (10б) и (18), причем в этом случае задача* ***N*** *будет иметь единственное решение.*

3. Пример. Пусть $L=\{z:|z|=1\}$. *Требуется найти все метааналитические в области* $T^+=\{z:|z|<1\}$ *функции* $F^+(z)$*, являющиеся решениями уравнения*

$$\frac{\partial^2 F(z)}{\partial \bar{z}^2}-2\frac{\partial F(z)}{\partial \bar{z}}+F(z)=0 \qquad (20а)$$

и удовлетворяющие на L *следующему краевому условию:*

$$\frac{\partial F^+(t)}{\partial n_+}+e^{\bar{t}-t}\cdot\overline{F^+(t)}=-\frac{e^{\bar{t}}}{t^2}. \qquad (20б)$$

Имеем $G(t)=e^{\frac{1}{t}-t}$, $g(t)=-\dfrac{e^{\frac{1}{t}}}{t^2}$.

Рассмотрим характеристическое уравнение для уравнения (20а):

$$\lambda^2-2\lambda+1=0.$$

Это уравнение имеет один (двукратный) корень $\lambda_0=1$, поэтому метааналитическую в области T^+ функцию $F^+(z)$ будем искать в виде

$$F^+(z)=\left[\phi_0^+(z)+\bar{z}\cdot\phi_1^+(z)\right]\cdot e^{\bar{z}}, \qquad (20в)$$

где $\phi_0^+(z)$, $\phi_1^+(z)$ – аналитические в T^+ функции.

С учетом представления (20в), вводя в рассмотрение вспомогательные аналитические соответственно в T^+ и T^- функции вида

$$\Phi^+(z)=z^3\frac{d\,\phi_0^+(z)}{d\,z}+z^2\frac{d\,\phi_1^+(z)}{d\,z}+z\phi_0^+(z)+(z+1)\phi_1^+(z),\quad z\in T^+, \qquad (20г)$$

$$\Phi^-(z)=\frac{1}{z}\overline{\phi_0^+}\left(\frac{1}{z}\right)+\overline{\phi_1^+}\left(\frac{1}{z}\right),\quad z\in T^-\ , \tag{20д}$$

краевое условие (20б) можно записать так:

$$\Phi^+(t)=t^3\cdot\Phi^-(t)+1,\quad t\in L\ . \tag{20е}$$

Равенство (20е) есть краевое условие скалярной задачи Римана относительно кусочно-аналитической функции $\Phi(z)=\{\Phi^+(z),\ \Phi^-(z)\}$. Найдем индекс задачи: $\chi=\mathrm{Ind}(t^3)=3$.

Так как $\chi\geq 0$, то задача безусловно разрешима и ее общее решение задается формулой (10а).

Следовательно, решение задачи (20б) имеет вид:

$$\begin{aligned}&\Phi^+(z)=c_3z^3+c_2z^2+c_1z+c_0+1,\qquad z\in T^+,\\&\Phi^-(z)=c_3+c_2\frac{1}{z}+c_1\frac{1}{z^2}+c_0\frac{1}{z^3},\qquad z\in T^-,\end{aligned} \tag{20ж}$$

где $c_0,\ c_1,\ c_2,\ c_3$ – произвольные комплексные постоянные.

Далее, из (16) с учетом (20ж), будем иметь:

$$\begin{cases}z^2\phi_0^+(z)+z\phi_1^+(t)=\overline{c_3}z+\overline{c_2}z^2+\overline{c_1}z^3+\overline{c_0}z^4,\ \ z\in T^+,\\ z^2\phi_0^+(z)-z\phi_1^+(t)=\Phi_1^+(z),\ \ z\in T^+,\end{cases} \tag{20з}$$

где

$$\Phi_1^+(z)=3\overline{c_0}z^4+\left(\overline{c_0}+2\overline{c_1}-c_3\right)z^3+\left(\overline{c_1}+\overline{c_2}-c_2\right)z^2+\left(\overline{c_2}-c_1\right)z+\overline{c_3}-c_0-1\ .$$

Решая систему (20з) относительно $\phi_0^+(z)$ и $\phi_1^+(z)$, получаем

$$\phi_0^+(z)=\frac{1}{2z^2}W_0^+(z),\ \ \phi_1^+(z)=\frac{1}{2z}W_1^+(z),\ \ z\in T^+\ , \tag{20и}$$

где

$$\begin{aligned}W_0^+(z)&=4\overline{c_0}z^4+\left(\overline{c_0}+3\overline{c_1}-c_3\right)z^3+\left(\overline{c_1}+2\overline{c_2}-c_2\right)z^2+\left(\overline{c_2}+\overline{c_3}-c_1\right)z+\overline{c_3}-c_0-1,\\W_1^+(z)&=-2\overline{c_0}z^4-\left(\overline{c_0}+\overline{c_1}-c_3\right)z^3-\left(\overline{c_1}-c_2\right)z^2-\left(\overline{c_2}-\overline{c_3}-c_1\right)z-\overline{c_3}+c_0+1,\ \ z\in T^+.\end{aligned}$$

Так как функция $W_0^+(z)$ должна иметь в точке $z=0$ нуль не ниже второго порядка, а функция $W_1^+(z)$ – нуль не ниже первого порядка, то решим следующую систему:

$$\begin{cases}\overline{c_2}+\overline{c_3}-c_1=0\\ \overline{c_3}-c_0-1=0\\ \overline{c_3}-c_0-1=0\end{cases}\Leftrightarrow\begin{cases}\overline{c_1}=c_3+c_2\\ \overline{c_0}=c_3-1\end{cases},$$

то есть функции $\phi_0^+(z)$ и $\phi_1^+(z)$ имеют вид:

$$\begin{aligned}&\phi_0^+(z)=2\left(c_3-1\right)z^2+\frac{1}{2}\left(3c_3+3c_2-1\right)z+\overline{c_2}+\frac{1}{2}c_3,\ \ z\in T^+,\\&\phi_1^+(z)=\left(1-c_3\right)z^3-\frac{1}{2}\left(c_3+c_2-1\right)z^2-\frac{1}{2}c_3z+\overline{c_3},\ \ z\in T^+.\end{aligned} \tag{20л}$$

Таким образом, общее решение задачи (20а), (20б) имеет вид:

$$\begin{aligned}F^+(z)=&[2\left(c_3-1\right)z^2+\frac{1}{2}\left(3c_3+3c_2-1\right)z+\overline{c_2}+\frac{1}{2}c_3+\\&+\overline{z}\cdot\left(\left(1-c_3\right)z^3-\frac{1}{2}\left(c_3+c_2-1\right)z^2-\frac{1}{2}c_3z+\overline{c_3}\right)]\cdot e^{\overline{z}},\end{aligned} \tag{21}$$

где $c_2,\ c_3$ – произвольные комплексные постоянные.

4. Описание процедуры ExactNeumann

Опишем основные особенности процедуры ExactNeumann, реализующую алгоритм решения рассмотренной выше задачи.

Процедура имеет 4 входных параметра: числовые параметры *a1, a0* условия (1), а также функциональные параметры *gb* и *gs* условия (5). Учитывая результаты, полученные в статье Адукова В.М. [5], можно сделать вывод о необходимости накладывания дополнительных требований к параметрам a_0 и a_1 уравнения (1), а также функциям $G(t)$, $g(t)$ условия (5). Эти требования позволят применить возможности символьных вычислений системы Maple и получить результат в виде, доступном для дальнейшего использования.

Перед обращением к процедуре требуется подключить пакет *LinearAlgebra*. Тогда обращение к процедуре будет иметь вид:

```
>with (LinearAlgebra)
>a1:=-2; a0:=1; gb:=exp(1/t - t); gs:=(-1)*exp(1/t)/(t^2);
> ExactNeumann (a1, a0, gb, gs).
```

Рис.1. Способ обращения к процедуре ExactNeumann

В результате использования указанных начальных данных получен следующий результат:

$$\left((2c_1-2)z^2+\left(\frac{3c_1}{2}+\frac{3c_2}{2}-\frac{1}{2}\right)z+\frac{1}{2}c_3+a,\quad (c_1-1)z^3+\left(-\frac{c_1}{2}-\frac{c_2}{2}+\frac{1}{2}\right)z^2-\frac{1}{2}c_3z+b\right)$$

Рис.2. Результат работы процедуры ExactNeumann

Поскольку точные вычисления предполагают использование рациональной арифметики, то контроль за рациональностью коэффициентов задачи Римана (10) на данном уровне решения проблемы обеспечивает пользователь, однако дальнейшее решение, в том числе и проверка условий разрешимости (10б) и (18), осуществляется програмно, и в случае их не выполнения процедура выдает соответствующее сообщение и прекращает выполнение.

Литература

1. Расулов К.М. Краевые задачи для полианалитических функций и некоторые их приложения. – Смоленск: Изд-во СГПУ, 1998. – 345 с.
2. Бицадзе А.В. Некоторые классы уравнений в частных производных. – М.: Наука, 1981.
3. Закарян А.А. Корректные граничные задачи для уравнения Бицадзе / Ереванск. Политехн. Ин-т. – Ереван, 1988. – 34 с. – Деп. в АрмНИИНТИ 24.08.88, № 66-Ар88.
4. Расулов К.М., Сенчилов В.В. О решении одной видоизмененной краевой задачи типа Рикье для метааналитических функций в круге // Дифференц. уравнения. 2005. Т. 41. № 3. С. 415–418.
5. Адуков В.М. О точном и приближенном решении задачи факторизации Винера-Хопфа для мероморфных матриц-функций // Вестн. Юж.-Урал. гос. ун-та, серия «Математика, физика, химия». – 2008. – № 7(107), вып. 10 – С. 3–12.

Скрипачев В.О.[1], Пирхавка А.П.[2], Полушковский Ю.А.[3], Суровцева И.В.[4], Жуков А.О.[5], Яковлев О.В.[6]

[1] Московский государственный технический университет МИРЭА, г. Москва,ст. преподаватель, skripatchevv@inbox.ru

[2]Московский государственный технический университет МИРЭА, г. Москва,к.т.н., доцент

[3]Московский государственный технический университет МИРЭА, г. Москва,к.т.н., ст. преподаватель

[4]НТЦ «Космонит» АО «Российские космические системы», г. Москва,вед.инженер

[5] Экспертно-аналитический центр Минобрнауки, д.т.н., доцент

[6]Вычислительный центр РАН им. А.А. Дородницына, г. Москва, д.т.н., с.н.с

АСПЕКТЫ СОЗДАНИЯ ИНФОРМАЦИОННОЙ СИСТЕМЫ ДЛЯ ОБРАБОТКИ ИОНОСФЕРНЫХ ДАННЫХ

КЛЮЧЕВЫЕ СЛОВА

Ионосфера, обработка ионосферных данных, информационная система, информационно-технологический процесс, UML диаграмма.

АННОТАЦИЯ

В статье обоснована необходимость создания информационной системы обработки ионосферных данных, представляющая унифицированные возможности для различных корпоративных систем потребителей ионосферной информации.

Для диагностики состояния ионосферы используются различные методы измерений как наземные, так и с привлечением бортовой аппаратуры космических аппаратов (КА). Обзоры методов диагностики ионосферы рассматриваются в работах [1].

Состояние ионосферы (положение границ ее слоев, электронная концентрация, ионосферные неоднородности и пр.) необходимо учитывать при решении целого ряда научно-исследовательских задач:

- организация связи на низких частотах [1];
- повышение точности навигационного определения посредством космических навигационных систем;
- передача данных по радиоканалам «космос-земля» [2];
- планирование работы высокочувствительной бортовой аппаратуры КА;
- обеспечение информацией потребителей о наличии возмущений геофизического и геологического характера, например о процессах подготовки землетрясений, извержений вулканов [3, 4].

С учетом большого числа разнотипных научно-исследовательских задач целесообразно рассмотреть вопросы создания информационных систем, предоставляющих конечным пользователям возможности доступа, обработки и использования ионосферной информации.

В настоящее время не существует единой информационной системы, предоставляющей такие возможности. Как правило, доступ к ионосферным данным, включая архивные, ограничен, что приводит к необходимости составления и согласования различных документов, регламентирующих доступ. Однако даже при получении такого доступа существуют сложности с обработкой ионосферной информации, заключающиеся в создании инструментария обработки исходных данных.

Тем самым обусловлена необходимость разработки информационной системы обработки данных ионосферных параметров, которая представляла бы унифицированные возможности для различных корпоративных систем потребителей ионосферной информации. Поскольку ионосферная информация достаточно разнородна, то ее обработка требует применения специальных аппаратных и программных средств.

Таким образом, под обработкой информации необходимо понимать информационно-технологические процессы (ИТП), сопряженные в единую систему. При этом каждый вид

рассматриваемой ионосферной информации должен аккумулироваться единичным ИТП, состоящим из алгоритмов получения, обработки и распространения информации.

Основу ИТП составляют операции преобразования информации. Каждая операция получает входную информацию и обеспечивает получение выходной информацию. При этом выходная информация поступает на вход следующей операции обработки данных и т.д., до получения конечного результата, предоставляемого потребителям. При комплексирование таких операций в определенном порядке позволяет получить необходимые потребителям данные [5].

При разработке системы ИТП необходимо опираться на опыт применения объектно-ориентированного программирования (ООП), широко используемого при создании информационных систем, где используется понятие класса, представляющего собой спецификацию, описывающую новую форму данных, и объекта, представляющего собой индивидуальную реализацию структуры данных, созданную в соответствии с этой спецификацией [6].

Целесообразно рассмотреть ИТП с точки зрения ООП.

В состав ИТП могут входить как аппаратные, так и программные средства, т.е.

$$C_{ИТП} = \{ H_{ИТП}, S_{ИТП} \},$$

где $C_{ИТП}$ – совокупность средств, составляющих ИТП;

$H_{ИТП}$ – совокупность аппаратных средств;

$S_{ИТП}$ – совокупность программных средств.

Опуская различия в аппаратных средствах получения ионосферной информации, рассматривая программные части ИТП $S_{ИТП}$, и опираясь на принципы наследования, полиморфизма и инкапсуляции необходимо создать «абстрактный» ИТП $Э_i^q$, в котором используется ряд операций, необходимых для достижения целевой функции обеспечения потребителей информацией. При этом операции и данные должны инкапсулироваться в ИТП [5].

Таким образом, систему обработки ионосферных данных можно представить как множество экземпляров процесса $Э_i^q$, т.е.

$$П^q = \{ Э_1^q, Э_2^q, \ldots, Э_i^q \}$$

Основными компонентами экземпляра ИТП являются: входная информация I_i^q, операции преобразования Q_i^q в виде технических T_i^q, программных P_i^q, организационных O_i^q средств и ресурсов, а также целевые компоненты C_i^q (требования, ограничения).

В результате формализации каждый экземпляр ИТП может быть представлен как:

$$Э_i^q = < I_i^q, Q_i^q(T_i^q, P_i^q, O_i^q), C_i^q > .$$

Для пояснения выше изложенного используем возможности языка UML (Unified Modeling Language). Диаграммы UML позволяют рассмотреть проектируемую систему с разных точек зрения [7].

Система ИТП представляется диаграммой прецедентов. Прецедент – описание множества последовательностей действий (включая их варианты), которые выполняются системой ИТП для того, чтобы потребитель получил значимый результат. Чаще всего с помощью прецедентов моделируют «поведение» элементов системы в целом или подсистемы.

Рассмотрим диаграмму прецедентов для системы ИТП, имеющей четыре потребителя (P_1, P_2, P_3, P_4) и четыре источника входных данных (A_1, A_2, A_3, A_4) (Рис.1). На диаграмме абстрактный ИТП ассоциирован с источниками данных A_1, A_2, A_3, A_4. Реализация $ИТП_i$ связана с абстрактным ИТП. Прецедент комплексирования информации зависит от реализации $ИТП_i$ и ассоциирован с потребителями.

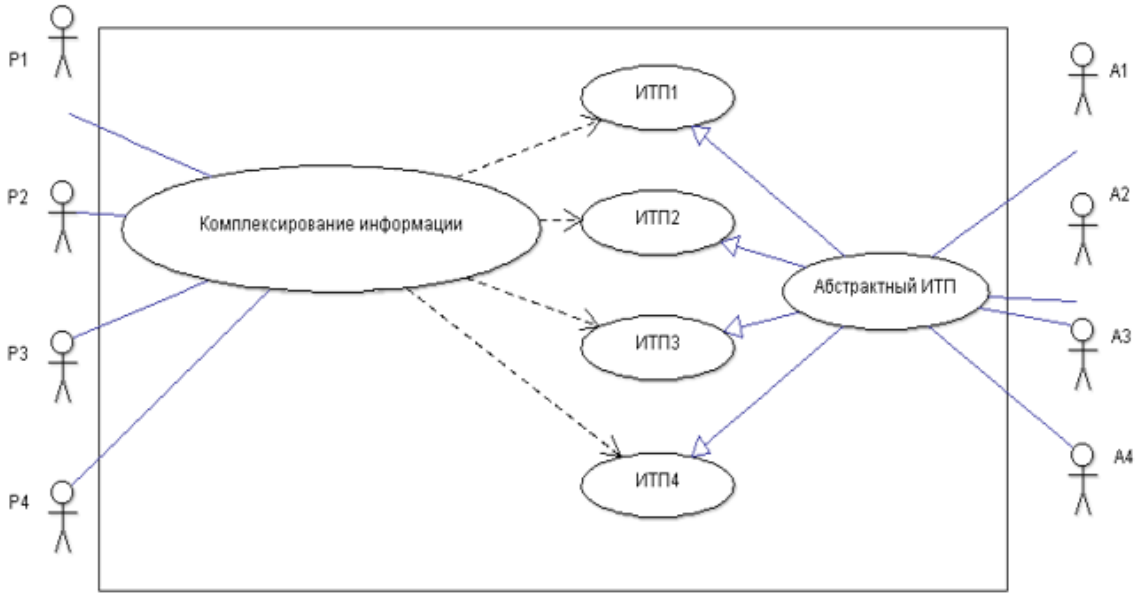


Рис.1 – UML диаграмма прецедентов

Прецедент представляет собой функциональные требования к системе в целом. При этом каждый прецедент описывает алгоритм взаимодействия входных параметров A_1, A_2, A_3, A_4 с системой в целом и ее абстракциями. Соответственно основными функциями, возлагаемыми на разрабатываемую информационную систему являются:

- обеспечение доступа к данным посредством соответствующих ИТП;
- возможность интеграции новых источников ионосферных параметров;
- преобразование исходных данных;

Для обеспечения функции доступа к данным необходимо учитывать, что работоспособность системы ИТП должна реализовываться с учетом использования потребителями разнородных систем обработки информации, поэтому в предлагаемой системе должны присутствовать API для работы с СУБД для различных языков программирования, наиболее распространенных в научно-исследовательской среде:

1. C/C++;
2. Java;
3. Fortran;
4. Interactive Data Language (IDL);
5. Python.

Для хранения ионосферных данных могут быть использованы СУБД MySQL или PostgreSQL.

Так, например, для работы с базами данных из языка Java необходимо использовать технологию JDBC (JavaDataBaseConnectivity), реализованную в виде пакета java.sql, входящего в состав Java SE. Основу JDBC составляют драйверы, позволяющие получать соединение с базой данных по соответствующему URL. В тоже время программный модуль IDL DataMiner используется для организации соединения с базами данных из IDL приложений посредством технологии ODBC (Open Database Connectivity).

Функция интеграции новых источников ионосферных данных должна обеспечиваться путем предоставления пользователю программных инструментов, позволяющих применять существующие структуры данных с минимальными трудозатратами, что можно достичь за счет широкого применения механизмов объектно-ориентированного программирования (инкапсуляция, наследование, полиморфизм). При этом необходимо учесть особенности применения объектно-ориентированной парадигмы в разных языках, например в IDL, о чем подробно изложено в [8].

Функция преобразования исходных данных заключается в их приведении к формату данных, удобном для применения в пользовательских системах обработки информации. Другими словами, данная функция должна обеспечиваться набором программных модулей, осуществляющих автоматизированное преобразование исходных данных, например преобразование файлов формата HDF в плоские файлы.

Помимо программных модулей, обеспечивающих реализацию вышеуказанных функций, целесообразно в состав системы ИТП ввести набор программных библиотек, позволяющих использовать их при работе корпоративных систем обработки ионосферной информации. В состав таких библиотек могут входить функции статистической обработки данных, функции фильтрации данных, функции доступа к тематическим интернет ресурсам.

Статистическая обработка необходима для получения наиболее информативных данных о каком-либо параметре ионосферы с течением времени, то есть необходимы процедуры обработки временных рядов наблюдений. При этом целесообразно использовать различные методы обработки в зависимости от их физической природы и шкал измерений: дихотомической, номинальной, интервальной [9].

Как правило, используются такие методы, как:

- корреляционный анализ для поиска тесноты связи между данными полученными различными источниками данных;
- регрессионный анализ для поиска количественных значений параметров;
- пробит-анализ - частный случай регрессионного анализа, позволяющий оценить не количественные значения, а вероятность возникновения события;
- спектральный анализ – метод, базирующийся на преобразовании Фурье.

Отдельно следует отметить необходимость проверки значимости в проводимых статистических исследованиях. Для определения уровня значимости, согласно теории математической статистики, требуется задание статистических гипотез. Как правило, используются: H_0 – нулевая гипотеза и H_1 – альтернативная гипотеза, при этом статистическая значимость определяется только на основе нулевой гипотезы.

Под фильтрацией данных необходимо понимать процедуры выборки данных с учетом заданных ограничений. Например, при анализе данных о сейсмической активности целесообразно вводить фильтр, определяемый формулой Добровольского [10]

$$\rho = 10^{0.43M},$$

где ρ – радиус зоны подготовки землетрясения;

M – магнитуда предполагаемого события.

Таким образом будут отобраны данные из целого набора, попадающие в указанный регион, что уменьшает объем обрабатываемых данных.

Под функциями доступа к тематическим Интернет ресурсамподразумевается доступ к ионосферным моделям, доступ к которым возможен в режиме онлайн, например ресурс *http://omniweb.gsfc.nasa.gov/vitmo*. Данный ресурс предоставляет возможность воспользоваться международными моделями ионосферы IRI и верхней атмосферы MSIS. При использовании в предлагаемой системе различных ионосферных моделей целесообразно исходить из следующего:

1. Модель MSIS (Mass Spectrometer Incoherent Scatter) описывает температуру и плотность нейтралов верхней атмосферы на высотах свыше 100 км. MSIS-86 составляет верхнюю часть модели CIRA-86 (COSPAR International Reference Atmosphere). Модель MSIS базируется на широком спектре данных, предоставленных A. E. Hedin и др. Источники данных включают [11]. Входными параметрами для моделиMSIS являются год, день от начала года, всемирное время (UT), высота, геодезические координаты (широта и долгота), местное время, поток $F_{10.7}$ (предыдущего дня и осредненный за три месяца, и магнитный A_p (дневной или за последние 59 часов).

2. Модель ионосферы IRI (International Reference Ionosphere) в настоящее время широко используется и совершенствуется уже более двух десятилетий и ежегодно обновляется специальными рабочими группами, спонсируемые комитетом космических исследований (COSPAR) и Международным Союзом радионаук (URSI). Эти организации сформировали рабочую группу для создания эмпирической модели ионосферы, базирующейся на данных из всевозможных источников. Было выпущено несколько переизданий модели, последнее переиздание произошло в 2012 году. Для заданного местоположения и времени IRI описывает электронную плотность, электронную температуру, ионную температуру и ионный состав в диапазоне высот от 50 до 2000 км, а также электронное содержание. Модель представляет усредненные месячные значения для неполярной ионосферы в спокойных магнитных условиях [12].

В целях получения модельных параметров на большом объеме исходных данных необходимо реализовать программные модули, позволяющие устанавливать HTTP соединение, вводить данные и сохранять полученные модельные результаты. На Рис. 2 приведена упрощенная

UML диаграмма классов разработанных программных модулей для работы с моделями MSIS и IRI.

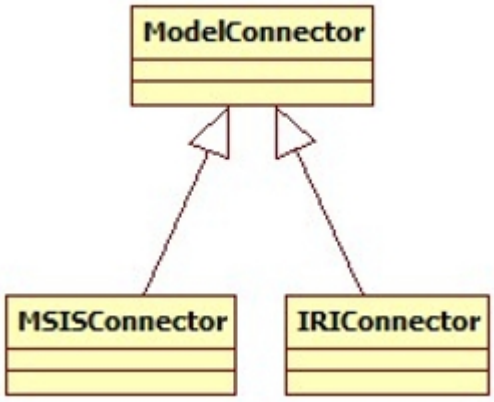


Рис.2 – UML диаграмма классов

Основным классом является ModelConnector, от которого наследуются классы MSISConnector и IRIConnector необходимые для работы с моделями MSIS и IRI соответственно.

Выводы

1. Показана необходимость создания информационной системы для обработки ионосферных данных, которая представляла бы унифицированные возможности для различных корпоративных систем потребителей.

2. Обоснованы рекомендации по созданию информационной системы обработки ионосферной информации с учетом:

- требований потребителей;

- условий организации информационно-технологических процессов и определяющих их факторов;

- состояния и перспектив развития технических, программных средств обработки и передачи ионосферных данных.

Литература

1. Nathan Blaunstein, Eugeniu Plohotniuc. Ionosphere and Applied Aspects of Radio Communication and Radar. CRC Press, 2008, 600p.
2. Peter Fortescue, Graham Swinerd, John Stark. Spacecraft Systems Engineering. Wiley; 4 edition, 2011, 724p.
3. С.И. Авдюшин, Н.П. Данилкин, С.В. Журавлев, Н.М. Давыдов, В.И. Лукьященко, В.А. Шувалов, А.И. Коваленко. Исследование и отработка многофункционального радиофизического комплекса для зондирования ионосферы, поверхности и подповерхности структур Земли с борта космических аппаратов. // Космонавтика и ракетостроение. 2007. №4 (49). С.158-163.
4. John Keith Hargreaves. The Solar-Terrestrial Environment: An Introduction to Geospace - the Science of the Terrestrial Upper Atmosphere, Ionosphere, and Magnetosphere. Cambridge University Press; 1995, 436p.
5. Скрипачев В.О., Чулкин М.О. Разработка прототипа информационно-технологического процесса обработки информации с учетом его стоимости // Программные продукты и системы. 2014. № 108. С. 100-102.
6. Stephen Prata. C++ Primer Plus (6th Edition). Addison-Wesley Professional; 6 edition. 2011. 1200p.
7. Язык UML. Руководство пользователя / Грейди Буч, Джеймс Рамбо, Айвар Джекобсон: Пер. с англ. Слинкин А.А. – 2-е изд., стер. – М.: ДМК Пресс; СПб.: Питер, 2004. – 432с.
8. Ronn Kling. Object Oriented Programming with IDL. - Kling Research and Software, 2010. – 126p.
9. Скрипачев В.О., Яковлев О.В. Методические аспекты реализации технологий мониторинга риска в технических системах // //Технологии техносферной безопасности: интернет-журнал. 2014. № 4 (56). С.26. http://ipb.mos.ru/ttb.
10. Dobrovolsky I.P., Zubkov S.I. and MiachkinV.I.:1979, Estimation of the size of earthquake preparation zone, Pure Appl.Geophys. 117,1025-1044.
11. A. E. Hedin, Extensionofthe MSIS Thermospheric Model into the Middle and Lower Atmosphere, J. Geophys. Res. 96, 1159, 1991.
12. Коррекция модели ионосферы по данным спутникового радиозондирования со сверхнизких орбит: дис. к.ф.м.н. / Азизбаев М.Р.

Юмагулов М.Г.[1], Беликова О.Н[2], Исанбаева Н.Р.[3]

[1]Башкирский Государственный Университет, г.Уфа, д.ф. -м.н., профессор, заведующий кафедрой дифференциальных уравнений, yum_mg@mail.ru

[2] Сибайский институт Башкирского Государственного Университета, г.Сибай, к. ф.-м. н., belikova-oksana@yandex.ru

[3]Башкирский Государственный Университет, г.Уфа, аспирант кафедры дифференциальных уравнений, nurgizarifovna@mail.ru

МОДЕЛИРОВАНИЕ ОБЛАСТЕЙ УСТОЙЧИВОСТИ ТОЧЕК ЛИБРАЦИИ ОГРАНИЧЕННОЙ ЗАДАЧИ ТРЕХ ТЕЛ С ПОМОЩЬЮ СИСТЕМ КОМПЬЮТЕРНОЙ МАТЕМАТИКИ

КЛЮЧЕВЫЕ СЛОВА

Устойчивость, задача трех тел, точки либрации, малый параметр, системы компьютерной математики.

АННОТАЦИЯ

В статье предлагается новая общая схема построения областей устойчивости треугольных точек либрации ограниченной эллиптической задачи трех тел с использованием систем компьютерной математики MATLAB и Maple.

Введение

Системы компьютерной математики (MATLAB, Maple, MathCad, Mathematica и др.) становятся все более популярными для исследования сложных задач теории и практики. Эта популярность вызвана не только мощью численных расчетов и возможностью символьных вычислений. В технологии проведения научных расчетов указанные системы стали по сути суперкалькуляторами, позволяющими быстро и эффективно проводить анализ сложных проблем, на решение которых еще 10-20 лет назад требовались бы много часов ручного труда и машинного времени и необходимость создавать специальные программные продукты.

Сказанное в полной мере относится и к классической задаче трех тел. Дифференциальные уравнения задачи трех тел занимают одно из центральных мест в математике и механике. Несмотря на относительную простоту формулировок и прозрачность основных формул эти уравнения представляют собой чрезвычайно сложный объект исследования, привлекающий повышенное внимание многих поколений ученых - математиков, механиков, физиков и др. Здесь разработан ряд методов исследования, нашедших многочисленные приложения в математике, небесной механике, астрономии и других науках (см., например, [1, 2] и имеющуюся там библиографию).

Неугасающее внимание к исследованию дифференциальных уравнений задач небесной механики связано не только с тем, что они находят свое применение при изучении движения небесных тел. Эти уравнения демонстрируют огромное многообразие качественного поведения решений, от самых простых - стационарных решений (точек либрации) - до сложных хаотических движений. Дифференциальные уравнения задач небесной механики зависят от различных параметров, что может приводить к тем или иным сценариям бифуркационного поведения.

Одной из наиболее актуальных как с теоретической, так и практической точек зрения представляется исследование областей устойчивости стационарных решений (точек либрации) дифференциальных уравнений ограниченной задачи трех тел и различных ее модификаций. Известные здесь наиболее полные результаты получены во второй половине прошлого столетия (см. [1,3]). Они основаны на сложных конструкциях и чрезвычайно утомительных компьютерных вычислениях. В настоящей статье предлагается новая общая схема построения областей устойчивости с использованием систем компьютерной математики MATLAB и Maple.

Схема М.Розо

Этот пункт носит вспомогательный характер, хотя приведенные в нем результаты имеют самостоятельный теоретический и практический интерес в задачах анализа устойчивости

стационарных и периодических решений дифференциальных уравнений с периодическими правыми частями. В этом пункте приводится общая схема исследования таких задач. Схема является развитием метода, предложенного в [4].

Рассмотрим зависящее от малого параметра ε дифференциальное уравнение

$$\frac{dz}{dt} = [A + \varepsilon P(t) + \varepsilon^2 Q(t,\varepsilon)]z + a(z,t,\varepsilon), \qquad z \in C^N, \tag{1}$$

где A - постоянная матрица, $P(t)$ и $Q(t,\varepsilon)$ - непрерывные матрицы, $a(z,t,\varepsilon)$ - вектор-функция, удовлетворяющая условию: $\|a(z,t,\varepsilon)\| = O(\|z\|^2)$ при $z \to 0$, при этом $P(t)$, $Q(t,\varepsilon)$ и $a(z,t,\varepsilon)$ являются *T*-периодическими по ε. Элементы матриц A, $P(t)$ и $Q(t,\varepsilon)$ вектор-функции $a(z,t,\varepsilon)$, а также параметр ε могут быть как вещественными, так и комплексными.

Предполагается, что матрица A имеет одно или несколько собственных значений с нулевой вещественной частью, а остальные собственные значения имеют отрицательные вещественные части. В этом случае при малых ε вопрос об устойчивости нулевого решения уравнения (1) зависит от мультипликаторов *T*-периодической матрицы $A+\varepsilon P(t)$. Отметим в этой связи, что явное построение мультипликаторов матрицы $A+\varepsilon P(t)$ возможно лишь в самых простых случаях.

Одним из эффективных способов исследования устойчивости нулевого решения уравнения (1) является предложенная М.Розо [4] схема перехода к равносильному уравнению вида

$$\frac{dy}{dt} = [A + \varepsilon S + \varepsilon^2 \tilde{Q}(t,\varepsilon)]y + \tilde{a}(y,t,\varepsilon), \ y \in C^N, \tag{2}$$

где *S* - постоянная матрица. Переход осуществляется с помощью замены

$$y = (I - \varepsilon H(t))z \tag{3}$$

где $H(t)$ - невырожденная *T*-периодическая матрица. Такой переход позволяет решить задачу об устойчивости путем построения собственных значений постоянной матрицы $A+\varepsilon S$, например, методами теории возмущений.

Подставляя (3) в (1), получим уравнение

$$\frac{dy}{dt} = \{A + \varepsilon[P(t) + AH(t) - H(t)A - H'(t)] + \varepsilon^2 \tilde{Q}(t,\varepsilon)\}y + a(y,t,\varepsilon). \tag{4}$$

Уравнение (4) будет уравнением вида (2), если удастся подобрать такую *T*-периодическую матрицу $H(t)$, для которой матрица $S = P(t) + AH(t) - H(t)A - H'(t)$ будет постоянной. С этой целью рассмотрим уравнение

$$\frac{dH}{dt} = AH(t) - H(t)A + P(t) - S. \tag{5}$$

Считая сначала, что постоянная матрица *S* задана, определим условия, при которых уравнение (5) имеет *T*-периодическое решение. Общее решение уравнения (5) имеет вид

$$H(t) = e^{At}\left[\int_0^T e^{-A\tau}(P(\tau) - S)e^{A\tau}d\tau + C_0\right]e^{-At}, \tag{6}$$

где C_0 - произвольная матрица.

Матрица (6) будет *T*-периодической тогда и только тогда, когда $H(0) = H(T)$. Отсюда получим равенство

$$C_0 = e^{AT}C_0e^{-AT} + e^{AT}\int_0^T e^{-A\tau}(P(\tau) - S)e^{A\tau}d\tau e^{-AT}. \tag{7}$$

ТЕОРЕМА 1. Пусть существуют матрицы C_0 и *S*, связанные равенством (7). Тогда замена (3), где $H(t)$ - матрица (6), приводит уравнение (1) к виду (2).

Естественен вопрос: существуют ли такие матрицы C_0 и *S*? Считая C_0 фиксированной, перепишем (7) в виде уравнения относительно *S*:

$$\int_0^T e^{-A\tau}Se^{A\tau}d\tau = \int_0^T e^{-A\tau}P(\tau)e^{A\tau}d\tau - e^{-AT}C_0e^{AT} + C_0. \tag{8}$$

Будем говорить, что для матрицы *A* выполняется условие отсутствия *T*-резонанса, если

любая пара λ_1 и λ_2 различных собственных значений этой матрицы удовлетворяет соотношению: $\lambda_1-\lambda_2\neq\frac{2\pi}{T}qi$ при целых q.

ТЕОРЕМА 2. *Уравнение (8) имеет единственное решение S тогда и только тогда, когда для матрицы A выполняется условие отсутствия T -резонанса.*

В частности, при $C_0=0$ уравнение (8) имеет вид

$$\int_0^T e^{-A\tau}Se^{A\tau}d\tau=\int_0^T e^{-A\tau}P(\tau)e^{A\tau}d\tau. \tag{9}$$

ТЕОРЕМА 3. *Пусть выполнено условие отсутствия T- резонанса. Тогда замена (3), где* $H(t)=e^{At}\left[\int_0^t e^{-A\tau}(P(\tau)-S_0)e^{A\tau}d\tau\right]e^{-At}$ *, а* S_0 *- единственное решение уравнения (9), приводит (1) к виду (2).*

ТЕОРЕМА 4. *Пусть выполнено условие отсутствия T- резонанса. Тогда единственным решением уравнения (8) будет матрица* $S=S_0-C_{0A}+AC_0$*, где* S_0 *- решение уравнения (9).*

СЛЕДСТВИЕ. *Если для матрицы A выполнено условие отсутствия T - резонанса, то существует бесконечно много различных замен вида (3), приводящих (1) к виду (2).*

Пусть теперь для матрицы *A* не выполнено условие отсутствия *T* -резонанса. В этом случае можно указать предварительное преобразование $x=U(t)z$ с *T*-периодической невырожденной матрицей $U(t)$, с помощью которых исходную систему (1) можно свести к равносильному виду

$$\frac{dx}{dt}=[\tilde{A}+\varepsilon\tilde{P}(t)+\varepsilon^2\tilde{Q}(t,\varepsilon)]x+\tilde{a}(x,t,\varepsilon),\qquad x\in C^N \tag{10}$$

для которого указанное условие уже выполнено.

Пусть, например, для некоторых двух различных полупростых собственных значений λ_1 и λ_2 матрицы *A* выполнено равенство $\lambda_1-\lambda_2\neq\frac{2\tau}{T}qi$ при некотором целом *q*. Обозначим через E_1 спектральное подпространство оператора A, отвечающее собственному значению λ_1, а через $P_1:C^N\to E_1$ -- оператор спектрального проектирования. Полагая $\widetilde{P}_1=\frac{-2\pi}{T}qiP_1$, произведем в (1) замену $x=e^{\widetilde{P}_1}z$ с *T*- периодической матрицей $e^{\widetilde{P}_1t}$. Тогда получим систему требуемого вида (10), в которой $\widetilde{A}=A+\widetilde{P}_1$.

3. Задача трех тел

В этом пункте приводится постановка основной задачи – построение областей устойчивости точек либрации задачи трех тел.

Рассматривается плоская ограниченная эллиптическая задача трех тел, уравнения движения которой в координатах Нехвила имеют вид (см., например, [1]):

$$\begin{cases} x''-2y'=\rho\left(x-\mu+\frac{\mu-1}{u^3}x-\frac{\mu}{v^3}(x-1)\right), \\ y''+2x'=\rho\left(y+\frac{\mu-1}{u^3}y-\frac{\mu}{v^3}y\right), \end{cases} \tag{11}$$

Здесь $u=(x^2+y^2)^{1/2}$, $v=[(x-1)^2+y^2]^{1/2}$, $\rho=\frac{1}{1+\varepsilon\cos t}$, μ -параметр масс, ε -эксцентриситет кеплеровской орбиты.

Система (11) имеет пять постоянных решений – точек либрации. Прямолинейные точки либрации неустойчивы при всех μ и малых значениях ε. В то же время треугольные точки либрации $L_4\left(\frac{1}{2},\frac{\sqrt{3}}{2}\right)$ и $L_5\left(\frac{1}{2},-\frac{\sqrt{3}}{2}\right)$ могут быть при указанных значениях μ и ε как устойчивыми, так и неустойчивыми.

Задача исследования устойчивости треугольных точек либрации оказалась чрезвычайно

сложной. Здесь особо важными представляются определение в плоскости параметров (μ,ε) областей устойчивости и неустойчивости в линейном приближении и, соответственно, границы между ними. Известные здесь наиболее полные результаты получены во второй половине прошлого столетия (см. [1,3]). Эти результаты схематично (для малых значений параметра масс μ) могут быть изображены так, как это показано на Рис. 1.

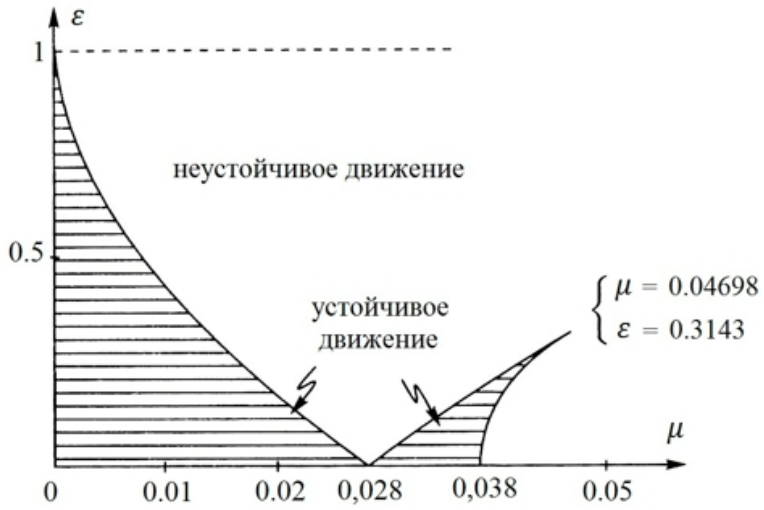


Рис. 1. Область устойчивости треугольных точек либрации

Заштрихованная область соответствует устойчивости точки либрации L_4 в линейном приближении. Граница области устойчивости образована совокупностью непрерывных линий, при переходе параметров ε и μ через которые в системе (11) возможны различные бифуркации [2, 5].

Ниже предлагается новая схема определения границы области устойчивости треугольных точек либрации в линейном приближении, основанная на построениях п. 2 настоящей статьи.

4. Построение областей устойчивости

Приведем основные положения определения границы области устойчивости треугольных точек либрации в линейном приближении применительно для построения одной из кривых $\varepsilon=f(\mu)$, образующих указанную границу. Эта кривая (см. Рис. 1) определена и монотонно убывает на отрезке $[0,\mu^*]$, где $\mu^*=\frac{1}{2}-\frac{\sqrt{2}}{3}$ ≈ 0,028, при этом $f(0)=1$ и $f(\mu^*)=0$. Искомую кривую $\varepsilon=f(\mu)$ предлагается строить в параметрическом виде: $\varepsilon(\delta)=\varepsilon_1\delta+\varepsilon_2\delta^2+\ldots,\mu(\delta)=\mu^*+\mu_1\delta+\mu_2\delta^2+\ldots$ где δ - малый параметр, а $\varepsilon_1,\varepsilon_2,\ldots,\mu_1,\mu_2,\ldots$ - требуют определения.

Ограничимся здесь схемой построения коэффициентов ε_1 и μ_1. Для построения остальных коэффициентов схема аналогична, но требует более громоздких построений.

Можно считать, что $\varepsilon_1=\sin\varphi$ и $\mu_1=\cos\varphi$ при некотором φ. Переходя от (11) посредством стандартной замены $h_1=x, h_2=y, h_3=x', h_4=y'$ к системе дифференциальных уравнений первого порядка и, линеаризуя затем правую часть полученной системы в окрестности треугольной точки либрации L_4, получим систему вида

$$h' = A_0 h + \delta[M_1(\varphi) + M_2(\varphi)\cos t]h + \delta^2 M_3(\delta,\varphi,t)h,\ h \in R^4, \qquad (12)$$

где матрицы $A_0, M_1(\varphi), M_2(\varphi)$ и $M_3(\delta,\varphi,t)$ могут быть получены в явном виде. При этом матрица A_0 имеет чисто мнимые собственные значения $\lambda_{1,2}=\pm i$ и $\lambda_{3,4}=\pm i\frac{\sqrt{3}}{2}$.

Дальнейшее исследование проводится по схеме, изложенной в п. 2. Система (12) имеет вид (1). При этом для матрицы A_0 не выполняется условие отсутствия *T*–резонанса при *T=2π.* Поэтому на первом этапе применяется преобразование, указанное в конце п. 2; здесь существенно используются системы MATLAB и Maple.

На втором этапе конструируется и затем решается уравнение (9). Этот этап также связан с использованием систем MATLAB и Maple. В результате получаем решение указанного уравнения - матрицу $S_0(\varphi)$; она зависит от угла φ.

Наконец, на третьем этапе определяются собственные значения матрицы $S_0(\varphi)$. Границе области устойчивости соответствуют те углы φ, при переходе через которые изменяется топологический тип матрицы $S_0(\varphi)$. В результате вычислений получаем следующие значения: $\varepsilon_1 = \sqrt{\frac{10368}{10401}} \approx 0{,}9884$, $mu_1 = -\sqrt{\frac{33}{10401}} \approx -0{,}0563$. Полученный результат соответствует изображенной на Рис. 1 области устойчивости, а именно, найденные числа ε_1 и μ_1 определяют касательную к кривой $\varepsilon = f(\mu)$ в точке μ^*.

Литература

1. Маркеев А.П. Точки либраций в небесной механике и космодинамике. М.: Наука. Глав. ред. физ.-мат. лит., 1978. – 312 с.
2. Маршал К. Задача трех тел. М.-Ижевск: Институт компьютерных исследований, 2004. – 640 с.
3. Nayfeh A.H., Kamel A.A. Stability of the triangular points in the elliptic restricted problem of three bodies. // AIAA Journal, 1970, v. 8, № 2
4. Розо М. Нелинейные колебания и теория устойчивости. - М.: Наука, 1971. – 288 с.
5. Юмагулов М.Г., Беликова О.Н. Бифуркация 4π -периодических решений плоской ограниченной эллиптической задачи трех тел. // Астрономический журнал, 2009 г., Т. 86, № 2. С. 170-174.

Лесников С.В.[1], Булыгина Д.С.[2], Лесников А.В.[3], Лесников Г.С.[4]

[1] Филиал ФГАОУ ВПО «Северный (Арктический) федеральный университет имени М.В. Ломоносова», г.Коряжма Архангельской области, доцент кафедры математики и информатики (http://ЛСВ.РФ) serg@lsw.ru

[2] Санкт-Петербургский государственный университет, аспирант.

[3] СыктГУ, инженер-программист, helpt@mail.ru

[4]СГМУ (Архангельск), студент, gleb@lsw.ru

КОНСТРУИРОВАНИЕ ГИПЕРТЕКСТОВОГО ИНФОРМАЦИОННО-ПОИСКОВОГО ТЕЗАУРУСА МЕТАЯЗЫКА ЛИНГВИСТИКИ

КЛЮЧЕВЫЕ СЛОВА

Гизаурус, гипертекст, дефиниция, классификация, лингвистика, математика, метаязык, поиск, рубрикатор, словарь, справочник, тезаурус, текст, толкование, указатель, экономика, энциклопедия, язык, языковедение, языкознание.

АННОТАЦИЯ

В статье рассмотрены основы конструируемого гипертекстового информационно-поискового тезауруса /ИПТ/ метаязыка лингвистики при финансовой поддержке Российского фонда фундаментальных исследований по проекту N11-07-00733 (2011-2013) «Гипертекстовый информационно-поисковый тезаурус «Метаязык науки» (структура; математическое, лингвистическое и программное обеспечения; разделы лингвистика, математика, экономика)» (научный руководитель - С.В. Лесников). Промежуточные итоги исследования изложены в ряде публикаций [1-15].Особенностью конструируемого тезауруса является то, что, помимо традиционного поиска по ключевым словам, тезаурус может помочь пользователю выделять смысловые акценты, соответствующие поисковому запросу, обеспечивая тем самым релевантность и пертинентность запросов.

При сравнении гипертекстовых (электронных, компьютерных, машинных, автоматизированных) версий и бумажных (книжных - традиционных в домашинную эпоху) тезаурусов можно отметить предоставляемые компьютером, информационными системами (технологиями) методические, организационные и технологические преимущества, связанные со следующими возможностями и характеристиками ИПТ:

1) мультимедиа (текст, графика, аудио, анимация, видео - комплексное воздействие на различные каналы восприятия информации; мультисенсорность) и гипертекст (нелинейное представление информации посредством гиперссылок);

2) персональность (адаптивность - способность информационной системы автоматически изменять свой дизайн и своё функционирование в интерактивном режиме в зависимости от потребностей и запросов пользователя, его профессионального уровня, а также от конкретных аспектов и режимов работы, состояния и/или внешних условий; индивидуальный, личный алгоритм нелинейной последовательности чтения источника) и интерактивность (диалог, интеракция, человеко-машинное взаимодействие);

3) выход в коммуникативные ситуации и актуализации дефиниций, интерпретаций, объяснений, определений, понятий, пояснений, разъяснений, толкований, трактовок, формулировок, экскурсов и эксцерпций терминов метаязыка (авторство, выбор, история, оценка, парадигма, поиск, разграничение, создание, уточнение, этимология);

4) поиск (точный, нечёткий, контекстный, полнотекстовый; многократный, скоростной; история поиска) и копирование, сохранение и редактирование материалов тезауруса;

5) дружественный интерфейс и художественно-эстетический дизайн;

6) множество входов в тезаурус (веб-формат, база данных; базовый и генеральный словники; систематический и пермутационный, авторский, предметный, хронологический указатели; ключевые слова и словосочетания; терминологизмы);

7) сетевой доступ (возможность использовать тезаурус в локальной, учебной, библиотечной, корпоративной сетях и в Интернете), а также оперирование большими корпусами текстов, неограниченное количество обращений как к фрагментам, так и целиком к тезаурусу; предоставление программой обратной связи с авторами-разработчиками и др.

Исследованиям метаязыка уделяли внимание следующие ученые: О.С. Ахманова, А.Н. Баранов, В.А. Виноградов, Н.В. Гвишиани, В.З. Демьянков, Д.О. Добровольский, Н.Н. Дурново, А.В. Иванов, М.И. Исаев, В.А. Ицкович, А.И. Киселевский, И.С. Куликова, А.В. Лемов, А.Т. Липатов, Л.Е. Макарова, Т.В. Матвеева, В.В. Налимов, В.Н. Немченко, Г.А. Нечаев, Т.М. Николаева, Р.В. Одеков, Е.Д. Поливанов, В.В. Потапов, И.А. Ребрушкина, А.А. Реформатский, Д.Э. Розенталь, О.И. Романова, Д.В. Салмина, М.Н. Славятинская, Н.А. Слюсарева, В.Д. Стариченок, М.А. Теленкова, В.А. Шаймиев, А.М. Шахнарович, В.Н. Ярцева.

«Thesaurus (греч.) = научная. коллекция; капитальный словарь» [16. С.730].

«Тезаурус (греч. thesauros - сокровище) - словарь для поиска какого-либо слова по его смысловой связи с другими словами… 1) лингвистический тезаурус - словарь, содержащий перечень слов естественного языка, отобранных в результате содержательного анализа текстов и систематизированных в соответствии с принятой классификационной системой, 2) статистический тезаурус - информационно-поисковый словарь, содержащий перечень слов, отобранных в результате статистического анализа текстов по какой-либо определённой тематике и сгруппированных в словарные статьи на основе частоты совместной встречаемости этих слов в одних и тех же текстах (документах)» [17. С.587].

Методологической и теоретической базой настоящей статьи послужили работы отечественных и зарубежных лингвистов в области теории текста: Арнольд И.В., Валгина Н.С., Гальперин И.Р., Колшанский Г.В., Кострова О.А., Тураева З.Я.; лексикографии и компьютерной лингвистики: Апресян Ю.Д., Гак В.Г., Лукьянова Н.А., Мандрикова Г.М., Назарова Т.Б., Ольшанский И.Г., Сивакова Н.А., Щерба Л.В.; теории современного гипертекста: Баранов О., Барст О.В., Брейдо Е., Буш В., Визель М., Давыдова К.В., Дедова О.В., Зыкова Е.В., Ковалева Н.Ф., Купер И.Р., Ларионова Н.С., Левоненко О.А., Лория А.В., Масалова М.В., Потапова Р.К., Рябов Г., Рязанцева Т.И., Субботин М.М., Хартунг Ю., Шехтман Н.А., Эпштейн В.Л., а также справочники, энциклопедии и словари лингвистики (в том числе и смежных с нею наук) и отдельных разделов языкознания.

Немного предыстории о гипертексте и некоторые определения:

Первая попытка сконструировать, точнее, описать "гипертекстовую машину" была предпринята в 1945 году [Bush V. 1945]. Ванневар Буш (1890-1974), научный советник президента Ф.Д. Рузвельта, считается первым, кто дал описание гипертекста. Он сделал это в своей статье 1945 г. “Как мы можем думать” (“As We May Think”, Atlantic Monthly, July 1945, p. 101-108 [http://www.isg.sfu.ca/~duchier/misc/vbush]), в которой призвал не пожалеть в послевоенное время усилий для механизации системы поиска данных в научной литературе. Рассматривая историю развития счетных машин, начиная с Лейбница и Бэббиджа, В.Буш предложил проект электронно-механического устройства, названного "memex" (от английского MEMory EXtension), которое должно дать человеку инструмент для усиления мыслительных способностей. Memex позволяет человеку определенным способом организовать свои записи, статьи, книги. С помощью механических средств (рычагов, кнопок и т.п.) тексты, записанные на микрофильмы, могут быть связаны между собой так, что переход от одного к другому осуществляется непосредственно, даже если физически они расположены далеко друг от друга. При чтении таким способом возможны пропуски страниц, возврат назад или просмотр каждых десяти страниц. Одновременно на экран могут вызываться несколько микрофильмов. Дополнительно к существующим материалам могут записываться новые - статьи, заметки, а также комментарии, ссылки и т.п. Буш считал, что бумажные формы представления информации значительно отстали от уровня требований, предъявляемых современной ему наукой. Архаичность средств хранения информации приводит к тому, что важнейшие результаты оказываются невостребованными.

Буш пишет: “Основная проблема - проблема доступа к информации. Она гораздо глубже и заключается не просто в несовершенстве библиотечной техники или неумении пользоваться ею. Причина нашей недостаточной способности к информационному поиску заключается в искусственности принятой системы индексации. Она предполагает, что собранные данные должны быть рассортированы, упорядочены по алфавиту или по порядковым номерам. Прослеживая цепочку переходов между подклассами, спускаясь по иерархическому дереву, мы обнаруживаем единственное место хранения, если не предусмотрено специальное дублирование. Эти действия отличаются неукоснительным подчинением строгим и весьма обременительным

правилам, причем, обнаружив один элемент данных, для поиска нового вам придется повторить все сначала.

Сознание работает совсем не так, оно оперирует ассоциациями. Захватив один объект, мозг немедленно переходит к другому, что предполагает наличие механизма мысленных ассоциаций, некоторой паутины (web) ячеек мозга, связанных сложными путями.

Эта паутина характеризуется высокой динамикой связей и непостоянством, однако обладает несравненным быстродействием" [http://www.computer-museum.ru/galglory/4.htm].

Буш одним из первых обнаружил несовершенство механизма индексации как метода систематизации и доступа к данным. Здесь позволительно сделать небольшое отступление. Дело в том, что индексация, которая очень часто представляется чуть ли не единственным механизмом поиска, с исторической точки зрения появилась относительно недавно. Индексация родилась в XV веке одновременно с изобретением книгопечатания и бухгалтерии. Поиск по индексу стал замечательным изобретением и отражал соответствующий уровень развития технологий и потребности того времени. Он был вызван необходимостью в унификации и стандартизации документооборота, как мы сказали бы сегодня.

Далее необходимо вспомнить работы Дугласа Энгельбарта (род. в 1925 г.) из Стэнфордского научно-исследовательского института. В 1963 г. Энгельбарт опубликовал статью "Концептуальная схема усиления человеческого интеллекта" ("A Conceptual Framework for the Atigmentation of Man's Intellect"). В ней. он развил представление о том, что компьютер возвещает переход человеческой эволюции в новую стадию, которая характеризуется "автоматизацией обращения с внешними символами". Энгельбарт предложил систему H-LAM/T - Human using Language, Artifacts and Methodology, in which he is trained (система подъема способностей человека посредством языка, артефактов и методологии), которая включала человека-пользователя как необходимый элемент. Пользователь и компьютер рассматриваются в этой системе как симбиоз динамических.компонентов, благодаря которому усиливается природный интеллект пользователя. Такого взгляда на гипертекст придерживались многие поколения разработчиков гипертекстовых систем. Свои взгляды он воплотил в созданной им системе NLS (On-Line System). В число основных функций NLS вошли: редактирование текстов в онлайновом режиме; гипертекстовые ссылки; телеконференция; электронная почта; конфигурирование рабочего места в соответствии с потребностями пользователя.

Эти функции невозможно было реализовать теми средствами, которые имелись в распоряжении ученых. Поэтому Энгельбарту пришлось создать целый ряд других программных и аппаратных средств, которые сегодня стали классикой: мышь для указания позиции на экране; многооконную систему вывода информации на экран; онлайновую систему подсказок (help) с контекстной привязкой; мультимедиа; архитектуру клиент-сервер; универсальный пользовательский интерфейс [18, с.130-133].

Следующее слово в истории гипертекста сказал Теодор Нельсон -программист, придумавший термин "гипертекст", дав следующее определение гипертекста: текст, не имеющий линейной последовательности ("нелинейное письмо") [Нельсон 1964]. Нельсон стремится реализовать идеи удачного, по его мнению, использования компьютера в виде всеобщей системы электронных публикаций и архива. А в основе подобной системы должно лежать понятие гипертекста. По мнению Т.Нельсона, компьютер призван сделать человека свободным, а поэтому нужно разработать систему, максимально облегчающую работу с информацией [Nelson T. 1974.]. Литература (понимаемая в широком смысле как информация, предназначенная для сохранения) представляет собой непрерывную систему взаимосвязанных произведений, и это надо рассматривать не как определение, а как обнаруженную данность, то есть "просто не-последовательное письмо (writing), текст, который разветвляется" и предоставляет читателю выбор [Nelson T. 1993].

Наибольшую известность **Теодору Нельсону** принесла работа над проектом "Xanadu". Это система динамического, непрерывно растущего гипертекста. Основная цель проекта состояла в том, чтобы способствовать революционному процессу перевода всего корпуса мировой литературы в такую форму, при которой возможен непосредственный доступ к любому произведению, извлечение из него любого фрагмента. Реализация "Xanadu" потерпела фиаско. В книге "Dream Machines", опубликованной в 1974 г., Нельсон обещал, что она будет готова в 1976 г. В издании "Literary Machines" 1987 г. этой датой стал 1988 г. Xartadu получила мощную поддержку в начале 1988 г. от "Autodesk", которая вложила в разработку 5 млн. долл. Нельсон ожидал завершения работ в 1991 г., и все опять кончилось ничем. "Autodesk" потеряла интерес к "Xanadu".

Идеи Нельсона всегда носили более литературный, чем научный характер. Нельсон - не инженер. Его “Xanadu” так и осталась мифом, как и тот волшебный дворец, в честь которого она названа. Однако именно он в самый ранний период информационной эры придумал название и сформировал ключевую концепцию гипертекста. Его интеллектуальное присутствие в огромной степени влияло на эволюцию гипертекстовых систем. Мало найдется исследователей, которые бы отрицали это.

В 1994 году в Саппоро (Япония) специально для Нельсона была создана лаборатория, финансируемая некоей японской корпорацией и получившая название HyperLab. Кроме того, Нельсон продолжает работать над “Xanadu” в Университете Кейо недалеко от Токио. В Австралии поддерживается проект “Xanadu Australia” (xanadu.com.au/xanadu/), в рамках которого продолжается разработка открытой системы электронных публикаций на основе идей “Xanadu”. Планируется всемирная сеть публикаций, основанная на этой разработке. Эта система литературы (“Xanadu Docuverse”) должна позволять людям создавать виртуальные копии (“transclusions”) любой информации в системе независимо от прав собственности.

Чтобы это было возможно, система должна гарантировать владельцу информации выплату авторского гонорара за любую порцию его документов, независимо от ее размера, времени и места использования (“transpublishing and transcopyright”).

В 1980-е годы создается множество коммерческих гипертекстовых систем. Согласно Conklin J., гипертекстовые системы классифицируются так: 1) библиотечные макросистемы; 2) средства исследования проблем ("инструментальные средства для поддержки неструктурированного мышления на ранней стадии исследования проблем, когда человек имеет дело с многочисленными, плохо связанными одна с другой идеями"); 3) системы для просмотра ("browsing") базы данных (то есть системы, в которых добавление данных пользователями не предусматривается); 4) системы широкого назначения (для экспериментирования в различных областях применения гипертекста). Механизм, заключающийся в возможности связать отрывки текста, переходить от одного к другому, называется гипертекстом, или нелинейным текстом: а) "надтекст" – некая единица информации, частями которой являются тексты, и/или б) текст, части которого имеют "сверхсвязи", то есть соединены друг с другом не линейным отношением в одномерном пространстве (отношением следования, как в обычном тексте естественного языка), а множеством различных отношений, представляемых в многомерном пространстве [Conklin J. 1987, p.20].

На русском языке о гипертексте одним из первых заявил публично Субботин М.М.: гипертекст - это соединение смысловой структуры, структуры внутренних связей некоего содержания, и технической среды, технических средств, дающих возможность человеку осваивать структуру смысловых связей, осуществлять переходы между взаимосвязанными элементами [19, с..2]. Под руководством М.М. Субботина группа исследователей, занимаясь вопросами организации электронных текстов, предложила метод логико-смыслового моделирования является как бы особой версией гипертекстовой технологии. Авторы этого метода утверждают, что помимо чтения гипертекста, существует еще одна качественно новая форма интеллектуальной деятельности - "создание линейного текста путем линеаризации нелинейного" [20. 1993].

В 1989 году Тим Бернерс-Ли предложил глобальную гипертекстовую систему, основанную на проекте “Enquire” и позволившую соединять связями не только текст, но и графику, звуки, видео. Глобальность этой системы, в отличие, например, от системы Теда Нельсона, предполагала, что данные будут распределяться по всему миру, а ее основой станет Интернет.

Бернерс-Ли был по образованию физиком. В 1984 г. он пришел в CERN, где занимался разработкой многозадачных операционных систем и компьютерных приложений. Изначальная цель создания новой системы, как и всей сети Интернет, состояла в обеспечении ученых (прежде всего физиков) новыми, удобными средствами коммуникации, обеспечивающими обмен результатами экспериментов, публикациями; совместную работу в режиме он-лайн.

Через год Бернерс-Ли написал первое клиент-серверное обеспечение для того, что теперь известно как World Wide Web (WWW или W3). Эта система впервые была задействована в декабре 1990 г. и дорабатывалась вплоть до 1993 г. За это время был предложен гипертекстовый протокол передачи данных (HTTP), Международной организацией стандартов ISO был разработан и установлен стандарт HTML; была создана универсальная система адресов (URL) для адресации документов в WWW.

Решающее значение для всемирного распространения WWW имело то, что все эти стандарты были сделаны общедоступными и использовались де-факто. К началу 2000г.

разработан и используется уже стандарт HTML 4.01 [http://www.w3.org/TR/REC-html40/].

Британская энциклопедия [The New Encyclopedia Britannica 1994] упоминает о гипертексте в разделе "информатика" (information science), при обсуждении методов семантического представления информации. Гипертекст определяется как метод, при котором документы объединяются посредством "связей" в сеть. Условием объединения может служить сходство идей, выводов, одинаковое расположение частей и так далее.

В энциклопедии языкознания гипертексту посвящена целая статья, смысл которой приблизительно соответствует смыслу статьи в британской энциклопедии с тем добавлением, что гипертекст - это структурированный текст [The Encyclopedia of Language and Linguistics 1994].

В толковом словаре по информатике [21] гипертекст трактуется как информационный массив, на котором заданы и автоматически поддерживаются ассоциативные и смысловые связи между выделенными элементами, понятиями, терминами или разделами; наряду с этим приводится дополнительное пояснение, что гипертекст - это текст, в котором отдельные термины на экране дисплея выделены подсветкой и их определение сразу может быть выдано на экран.

Л.В. Шуткин структуры гипертекстов представляет моделью в виде регулярной конфигурации теории паттернов, при этом структуры гипертекстовых тем рассматриваются как образующие темы [22].

Безусловно, суть гипертекста можно уяснить лишь непосредственно в компьютерном варианте, однако для примера можно привести опубликованную книжную версию словаря-гипертекста, где авторы определяют: «Гипертекст - это метод построения и организации электронной компьютерной документации, позволяющей отразить ассоциативные связи между ее отдельными разделами, понятиями и терминами. Читая такую документацию, как электронную книгу на экране ЭВМ, и, встретив новое, неясное или забытое понятие, можно немедленно заставить машину провести Вас по всем ассоциативным связям этого понятия и показать любые участки текста, поясняющие его смысл или раскрывающие взаимодействие с др. объектами, категориями, терминами» [23, с.3]. Элементы гипертекста и использование соответствующих гипертекстовых технологий в лингвистике можно наблюдать также в виде паралингвистических реалий и средств, которые могут быть представлены в виде поля паралингвистических средств, определяющих внешнюю организацию письменного текста и создающих его (текста) «оптический образ». Собственно паралингвистическое поле может детерминироваться как графическим представлением текста (например, сегментацией и расположением), так и использованием разнообразных типографских знаков, шрифтов, средств иконического и пиктографического языка, схем, рисунков, таблиц, фотографий и др. [24].

Для полноты представления понятия «гипертекст» приведем цитату из «Словаря культуры XX века: ключевые понятия и тексты» В.П. Руднева: «ГИПЕРТЕКСТ - текст, устроенный таким образом, что он превращается в систему, иерархию текстов, одновременно составляя единство и множество текстов. Простейший пример Г. - это любой словарь или энциклопедия, где каждая статья имеет отсылки к другим статьям этого же словаря. В результате читать такой текст можно по-разному: от одной статьи к другой, по мере надобности, игнорируя гипертекстовые отсылки; читать статьи подряд, справляясь с отсылками; наконец, пуститься в гипертекстовое плавание, то есть от одной отсылки переходить к другой. Настоящий словарь является подчеркнуто гипертекстовым, то есть построен так, чтобы постоянно провоцировать именно третий путь чтения. Например, сама статья “Г.” безусловно связана со статьями текст[15] и реальность (существует еще понятие гиперреальности), а стало быть, с пространством и временем, с мифом и неомифологизмом, виртуальными реальностями и измененными состояниями сознания, от которых путь недалек к психическим отклонениям, к шизофрении, психозу, неврозу, от чего один шаг до психоанализа, от него к Эдипову комплексу, оттуда к сновидению, от сновидения к проблеме индивидуального языка, от него к аналитической философии в целом, от нее к теории речевых актов, оттуда к прагматике, к семиотике, мотивному анализу, к интертексту и Г. И тут цепочка (одна из возможных гипертекстовых цепочек — ибо Г. нелинеен) замкнется. ... это имеет отношение к Эдипову комплексу, ибо кого же мы хотим перехитрить, как не самого себя и свои чудовищные бессознательные стремления, и это безусловно индивидуальный язык, потому что его, кроме нас, никто не поймет, и это аналитическая философия, потому что мы все время так или иначе анализируем значения употребляемых нами слов, и это теория речевых актов, потому что гипертекстовое путешествие - это очень сложная разновидность речевого акта, а раз так, то он относится к области прагматики

15 Подчеркнуты слова, имеющие отдельные словарные статьи в словаре В.П. Руднева.

и, стало быть, к семиотике, поскольку прагматика - одна третья часть семиотики. И это, конечно, семиотика интертекста. Надеюсь, что более наглядно и убедительно трудно было объяснить, что такое Г. Лит.**:** Бодрийар Ж. Войны в заливе не было // Художественный журнал. 1993. № 4. Интервью с Жаном Бодрийаром // Там же. 1995. № 8. Руднев В. Морфология реальности: Исследование по “философии текста”. М., 1996» [Руднев В.П. 1997, с.69-72 Руднев В.П. Словарь культуры XX века. Ключевые понятия и тексты. М.: Аграф, 1997. 384с. Указ..7-10.]. Словарь культуры XX века Руднева представляет собой уникальный словарь-гипертекст и содержит 140 словарных статей, посвященных наиболее актуальным понятиям и текстам культуры XX века. Для нас актуальны словарные статьи: *автокоммуникация, виртуальные реальности, генеративная лингвистика, гипотеза лингвистической относительности, знак, метаязык, многозначные логики, принцип дополнительности, структурная лингвистика, текст в тексте, формальная школа, хазарский словарь* и др. Текст словаря Руднева в полном объеме доступен в Интернете.

«Гипертекст обеспечивает нелинейную организацию содержания со стороны продуциента и возможность нелинейного восприятия со стороны реципиента. Формально гипертекст - это вся совокупность текстов (…), связанных с воспринимаемым текстом посредством ссылочного аппарата. Гипертексты, “технически” соединенные друг с другом ссылочным аппаратом, следует отличать от тематически и организационно (но не “технически”) объединенной последовательности текстов, например совокупности реплик всех участников какой-нибудь дискуссионной группы. В гипертексте на место двумерного сочетания горизонтального и вертикального контекстов приходит контекстуальная трех- и даже n-мерность (n>3). Каждая ссылка представляет собой вектор-указатель на другой текст, который, в свою очередь, либо задает (n-1)-мерность, имея выход на другие тексты (т. е. соединяется с n текстов минус один первичный текст), либо является двумерным, обладая только горизонтальным и вертикальным контекстами (имеет вход, но не имеет выхода на другие тексты). В связи с этим культурно-речевой аспект гипертекстуальности может включать среди прочего исследование способов оптимизации ссылок и установление определенных ссылочных норм. Идея формального и семантического взаимоотношения между различными текстами, идея интер- и гипертекстуальности не нова. В применении к компьютерным документам ее возводят к 1965 г. Типичным же гипертекстом досетевой эпохи является совокупность текстов Священного Писания. Их можно читать не только линейно, но и нелинейно, следуя эксплицитно указанному во многих изданиях книг Ветхого и Нового Завета порядку ассоциативных ссылок и параллельных мест (о Священном Писании как гипертексте ср.: [Иванов Л.Ю. 2001]. "Гипертекст позволяет связывать текст, аудио, фотографии, чертежи, карты, движущиеся картинки и другие формы информации в осмысленное целое, к которому может осуществляться доступ при помощи системы индексации, ориентированной на конкретные идеи, а не на конкретные слова в тексте" [25].

Таким образом, под гипертекстом может пониматься как: - метод объединения документов; - текст, организованный по-особому, при этом, важную роль в этой организации играют "связи" между текстами, устанавливающиеся произвольно; - механизм, позволяющий эти тексты связывать гиперссылками, однако связанные тексты здесь рассматриваются как единое целое, большее, чем текст, то есть гипер-текст; - форма организации материала (электронная, компьютерная); - средство организации текстов; - документация, которая ветвится. В разных контекстах трактовка гипертекста может приобретать различные оттенки и рассматриваться как метод, средство, форма, механизм актуализации информации и т.д.

Практическим итогом гипертекстовых разработок стала компьютерная сеть INTERNET.

ИНТЕРНеТ= (англ. Internet, от Interconnected Networks - объединённые сети]; сленг. ИНеТ, НЭТ)= глобальная телекоммуникационная сеть информационных и вычислительных ресурсов. Служит физической основой для Всемирной паутины. Часто упоминается как Всемирная сеть, Глобальная сеть, либо просто сеть. Когда слово Интернет употребляется в обиходе, чаще всего имеется в виду Всемирная паутина и доступная в ней информация, а не сама физическая сеть.

«ИНТЕРНеТ= Единый интерактивный гипертекст» [26].

Для компьютерных технологий в Интернете принято следующее определение. Гипертекст - hypertext - «метод подготовки и распространения текстов, который идеально подходит для компьютера, на котором читатель может выбирать свой путь прохождения по материалу. Для подготовки гипертекста вы сначала “нарезаете” информацию в виде небольших частей, которыми легко манипулировать, таких, как одна страница текста. Затем вы встраиваете в текст гиперсвязи. Когда читатель щелкает мышью на гиперсвязи, программное обеспечение поддержки гипертекста

отображает информацию, предоставляемую другим узлом. Процесс управления прохождения среди узлов, связанных в этот путь, называется просмотром. Набор узлов, взаимосвязанных посредством гиперсвязи, называется паутиной WWW - World Wide Web и представляет собой гипертекстовую систему мирового масштаба.

Отметим еще несколько базовых определений, используемых в информационных технологиях в связи с понятием «гипертекст»: 1) http - Hypertext transfer protocol - протокол передачи гипертекстов - способ передачи страниц WWW в Интернете. 2) HTML - Hypertext markup language - язык ссылок гипертекста - позволяет включать в текст коды (тэги, признаки, параметры, разметку) для определения шрифтов, стилей, графики, аудио и видео данных как непосредственно, так и посредством гипертекстовых ссылок. Процесс конструирования HTML-текстов называется авторизацией. Любая HTML страничка может быть создана и соответственно отредактирована практически в любом текстовом редакторе, который позволяет обрабатывать файлы в коде ASCII = American Standart Code for Information Interchange - американский стандартный код для обмена информацией (версия ISO).

Основные понятия из предметной области "гипертекст": - браузер (программа просмотра гипертекста), - связь (отношение между фрагментами текста, выраженное некоторым образом в данных информационной системы, - ссылка (указатель при ... на другой фрагмент), - текстура (текст, оформленный некоторым образом посредством подбора шрифтов, цветов, размещения компонентов, фрагмент (фрем - часть текста, имеющая смысловую законченность, ссылаемая как целое) - язык разметки гипертекста (Бурьяк А. Системы поддержки решений на основе гипертекста. 2005. http://bouriac.narod.ru/DSS.htm).

По мере развития сети Интернет в ее оборот вовлекались все большие объемы информации, ориентироваться в которой становилось все труднее. Возникла задача создать простой и понятный способ организации Информации, размещенной на узлах сети Интернет.

В 1991 году Европейская лаборатория практической физики (CERN) объявила о создании новой глобальной информационной среды World Wide Web. В WWW используются следующие механизмы: 1. Гипертекст для простого перемещения по ресурсам (HTML). 2. Протоколы для доступа к именованным ресурсам через Web (HTTP). 3. Единая схема наименования для поиска ресурсов в Web (URI).

Конструирование тезауруса обусловлено необходимостью единства в терминологии - нередко учёные одно и то же явление называют по-разному, с другой стороны, один и тот же термин бывает полисемичен даже в рамках одной научной области. Кроме этого, при анализе уже опубликованных терминологических словарей определённой науки бросается в глаза лакунарность состава словников и некая тенденциозность подачи научных материалов авторами-составителями. Фактически речь идет о составлении метаязыка науки - особого языка, объектом которого является содержание и выражение другого языка, т.е. в нашем случае метаязык - это язык второго порядка как специальная семиологическая система, употребляемая тогда, когда надо говорить о естественном (или искусственном) языке же, выступающем в качестве «языка-объекта». При этом можно отдельно выделить в метаязыке следующее: 1) собственно термины (слова, которые или совсем не употребляются в языке-объекте, либо приобретают, после заимствования из языка-объекта, особое значение), 2) сочетания слов, которые характеризуют собственно метаязык определенной науки в его полном виде (напр., составные термины), 3) определенный социально-предметный аспект (напр., когда метаречь отражает специфику эпохи, того или иного научного направления, школы).

Литература

1. Кузнецова И.В., Лесников С.В. Компетентностно-ориентированные задания как средство формирования профессиональной компетентности будущего учителя математики и информатики // Казанская наука. №8. 2011. Казань: Изд-во Казанский Издательский Дом, 2011. С.268-271.
2. Кузнецова И.В., Лесников С.В. Разработка и описание гипертекстового информационно-поискового тезауруса по алгебре // Вестник Российского университета дружбы народов. Серия: информатизация образования. Москва: РУДН, 2011. №3. С.70-76.
3. Лесников С.В. Конструирование гипертекстового генерального свода лексики русского языка: русский гизаурус / Международная научная конференция "Язык и культура". 14-17 сентября 2001. М.: ОЛЯ РАН, научн. журнал "Вопросы филологии", Инст. иностранных языков, 2001, с.280-281. (1000 экз.) ISSN 1562-1391.
4. Лесников С.В. Автоматизированная информационно-справочная система поиска лексико-семантических вариантов вокабул русского языка // Проблемы истории, филологии, культуры. 2009. №24. Магнитогорск: Магнитогорский ГУ, 2009. С.622-630. ISSN 1991-9484 http://pifk.science74.ru/ http://elibrary.ru/item.asp?id=16863201.
5. Лесников С.В. Базовые блоки автоматизированной лексикографической системы // Вестник Челябинского

государственного университета. Серия: филология, искусствоведение. Вып. 60. №33(248). Челябинск: ЧелГУ, 2011. С.200-202. (1000 экз.) ISSN 1994-2796. http://www.lib.csu.ru/elbibl/vestnik.shtml.
6. Лесников С.В. Гипертекстовый тезаурус метаязыка науки // Проблемы истории, филологии, культуры. №3(33) 2011. Москва-Магнитогорск-Новосибирск: Российская академия наук (отделение историко-филологических наук РАН); Изд-во ООО «Аналитик», 2011. С.30-34. (500 экз.) ISSN 1991-9484..
7. Лесников С.В. Моделирование тезауруса метаязыка лингвистики на базе гипертекстовых фреймов // Вестник Вятского государственного гуманитарного университета. Филология и искусствоведение. №3(2). Киров: ВятГГУ, 2011. С.51-54. (1000 экз.) ISSN 1997-4280. http://www.vggu.ru/sites/default/files/public/vestnik_322011_m.pdf.
8. Лесников С.В. Основные латинские терминоэлементы и термины метаязыка лингвистики // Научные ведомости БелГУ. Серия: гуманитарные науки. Белгород: БелГУ, 2011. №12(107). Вып.10. С.37-45. ISSN 1990-5327 http://unid.bsu.edu.ru/upload/iblock/04a/g%2012_107_vip_10-last.pdf.
9. Лесников С.В. Тезаурус как отражение системности языка // Вестник Челябинского государственного университета. Серия: филология, искусствоведение. Вып.59. №28(243). Челябинск: ЧелГУ, 2011. С.52-61. (1000 экз.) ISSN 1994-2796. http://www.lib.csu.ru/vch/243/vcsu11_28.pdf.
10. Лесников С.В. Типология русских словарей лингвистической терминологии // Мир науки, культуры, образования. №6(31) 2011. Часть 2, декабрь 2011. Раздел: Филология. Горно-Алтайск: (типография «Концепт», Барнаул) 2011. С.6-10. (500 экз.) ISSN 1991-5497. http://amnko.ru/index.php/download_file/view/111/82 http://amnko.ru
11. Лесников С.В. Фреймовое конструирование тезауруса метаязыка лингвистики // Вестник Поморского университета. Серия "Гуманитарные и социальные науки". №4. Архангельск: ПГУ, 2011. С.84-89. (1000 экз.) ISSN 1728-7391..
12. Лесников С.В. Базовые операторы языка поисковых запросов тезауруса метаязыка лингвистики // В мире научных открытий. Красноярск: Научно-инновационный центр, 2012. №7.2(31) (Гуманитарные и общественные науки). С.39-53. ISSN 2072-0831. (1000 экз.) http://nkras.ru/vmno/issues/articles/2012/7.2/sod7.22012.pdf.
13. Лесников С.В., Кузнецова И.В. Концепция учебной деятельности в сетевых сообществах при обучении алгебраическим структурам, направленной на формирование методической компетентности учителя математики // Вестник Российского университета дружбы народов. Серия: информатизация образования. Москва: РУДН (Российский университет дружбы народов), 2012. №4. С.47-56. (500 экз.) ISSN 0869-8732 http://www.ido.rudn.ru/vestnik/2012/2012_4/7.pdf.
14. Лесников С.В., Найденов Н.Д., Новокшонова Е.Н. Классификации и их совершенствование в экономических науках // Современные исследования социальных проблем. № 3(07). Красноярск: Научно-инновационный центр, 2011. С.138-150. (1000 экз.) ISSN 2077-1770 (print). ISSN 2218-7405 (online) http://sisp.nkras.ru/issues/32011/11.pdf.
15. Лесников С.В., Найденов Н.Д., Новокшонова Е.Н., Швецова Н.Н. Совершенствование классификации экономических наук в свете теории социальных коммуникаций // Философия социальных коммуникаций. Научно-теоретический журнал. 3(20) 2012. Волгоград, 2012. С.18-30.
16. Смирнов В. Полный словарь иностранных слов, вошедших в русский язык с общедоступным толкованием их значения и употребления, и со включением отдельных слов и фраз, употребляющихся в устной и письменной речи в их оригинальной иностранной форме. М.: Издание торгового дома В.Коновалова и К, 1908. 800 с.
17. Кондаков Николай Иванович. Логический словарь-справочник / Отв. ред. Д.П. Горский. М.: Наука, 1975. 720 с.
18. Коряковцева Н.А. Техники информационно-библиотечной работы / Предисл. и науч. ред. В.А. Фокеева. М.: Либерея, 2004. 136с.
19. Субботин М.М. Новая информационная технология: создание и обработка гипертекста // Научно-техническая информация. Сер.2. 1988. N5.
20. Субботин М.М. Теория и практика нелинейного письма (взгляд сквозь призму "грамматологии" Ж. Деррида) // Вопросы философии. 1993. N3. С.36-45.
21. Першиков В.И., Савинков В.М. Толковый словарь по информатике. М.: Фин. и статистика, 1995.
22. Шуткин Л.В. Паттерновое моделирование гипертекстов // НТИ. Сер. 2. 1995. № 9. С.20-25.
23. Евсеев О.В., Савицкая Е.В. Словарь-гипертекст экономики и бизнеса. М.: Изд-во МГОУ А/О «Росвузнаука», 1993. 163с.
24. Анисимова Е.Е. Паралингвистика и текст (к проблеме кролизованных и гибридных текстов) // ВЯ. 1992. № 1. С.71-78.
25. Эпштейн В.Л. Введение в гипертекст и гипертекстовые системы. 02.07.1998 http://www.ipu.ru/publ/epstn.htm
26. Абрегов А.Н. (ред.) Словарь русской лингвистической терминологии. 2003.

Понятский В.М.[1], Егоров Д.Б.[2]

[1] АО Конструкторское бюро приборостроения им. академика А.Г. Шипунова, г. Тула, Тульский государственный университет, к.т.н., доцент кафедры Проектирование автоматизированных комплексов, pwmru@rambler.ru

[2] Тульский государственный университет, г. Тула, аспирант кафедры радиоэлектроники, gosha_fighten@mail.ru

ПРОГРАММНЫЙ КОМПЛЕКС МОДЕЛИРОВАНИЯ ПОСЛЕДОВАТЕЛЬНОСТИ МЕТОДОВ ВИДЕООБРАБОТКИ ДЛЯ ЗАДАЧ УПРАВЛЕНИЯ

КЛЮЧЕВЫЕ СЛОВА

Изображение, обработка, метод, модуль, время, пакетный режим обработки, покадровый режим обработки.

АННОТАЦИЯ

В данной работе рассматривается задача моделирования процесса обработки видеопотока в реальных информационно-измерительных системах для задач управления.

Программный комплекс обработки видеоизображений DIP Studio 2.5 [1] предоставляет возможность выбора последовательности методов обработки, осуществить настройку их параметров и режимов работы для обработки и селекции изображений. Комплекс имеет встроенные модули для ввода или моделирования последовательности видеоизображений, их предварительной обработки, селекции и отображения результатов обработки. Комплекс позволяет выбрать последовательность методов обработки видеокадров из предоставленного набора модулей и скомпоновать составленный набор методов обработки в проект с возможностью его сохранения и корректировки. Данный комплекс выполняет выбранный набор методов модуль за модулем, т.е. сначала первый модуль обрабатывает всю последовательность изображений, затем также второй модуль, третий и т.д. Таким образом, на момент выполнения текущего модуля все предыдущие модули обрабатывают всю заданную последовательность видеоизображений. Проект можно запустить на выполнение либо целиком, либо по шагам. Шаг – минимальное действие для выполняемого проекта. В комплексе DIP Studio 2.5 одним шагом является выполнение одного модуля для всей последовательности изображений или всего интервала времени, т.е. обеспечивается пакетная обработка. Таким образом, DIP Studio 2.5 реализует пакетный режим обработки проекта.

В видеосистемах реального времени пакетный процесс обработки не соответствует реальному процессу обработки. В реальных системах с определенной частотой дискретизации в процессор поступают изображения, и процессор выполняет весь набор операций над конкретным видеоизображением. В терминах программного комплекса такой процесс обработки выражается в том, что для конкретного отсчета времени выполняются все модули проекта. Далее выполняются все модули проекта для следующего отсчета времени и т.д. То есть обеспечивается покадровая обработка изображений.

Разработан программный комплекс DIP Studio 3, обеспечивающий обработку изображений в соответствии с реальными процессами в видеосистемах слежения и управления (рис. 1).

Программный комплекс DIP Studio 3 поддерживает как пакетный режим обработки, так и покадровый. Для покадрового режима понятие шага выполнения определяется следующим образом: выполнение одного модуля для данного отсчета времени. Следующий шаг – выполнение следующего модуля для этого же отсчета времени, и так до последнего модуля в проекте. Когда последний модуль проекта для данного отсчета времени выполнен, следующим шагом является выполнение первого модуля проекта для следующего отсчета времени. И так далее.

Для задания времени начала и времени конца выполнения проекта, а также его шага дискретизации была введена дополнительная панель времени (рис. 2).

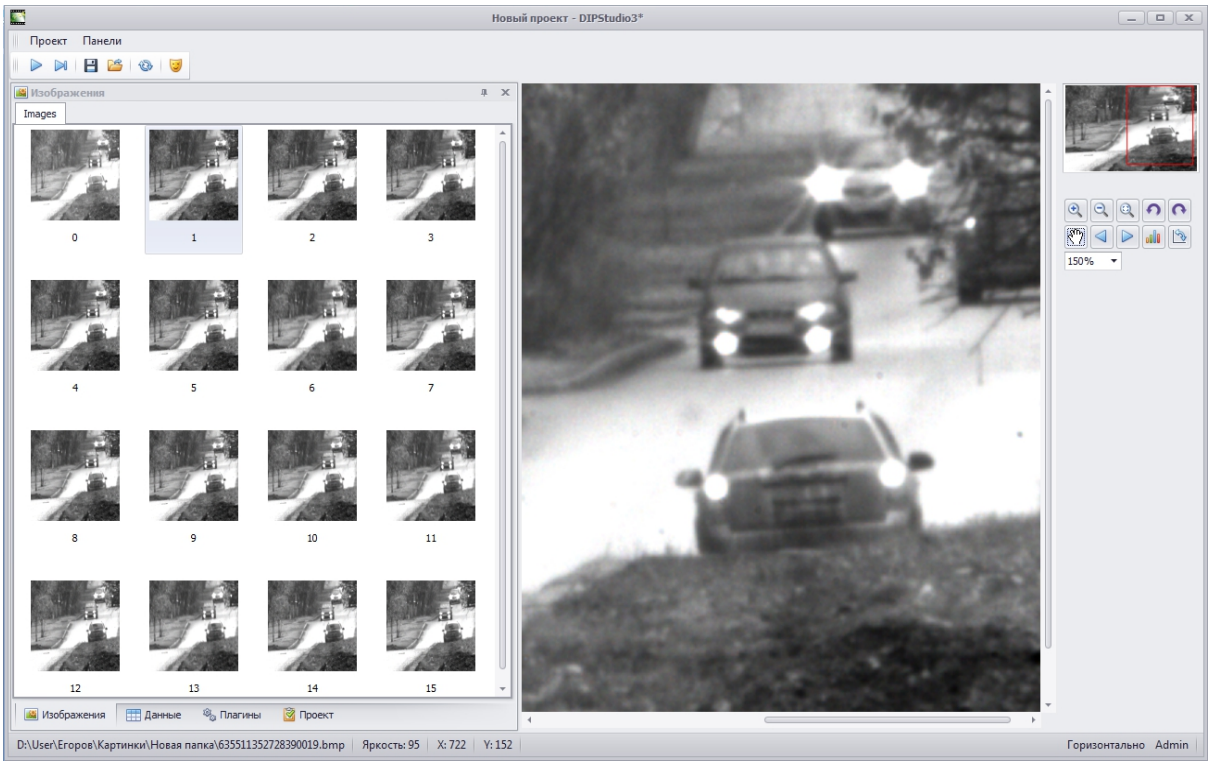


Рис. 1. Интерфейс программного комплекса моделирования последовательности методов видеообработки для задач управления

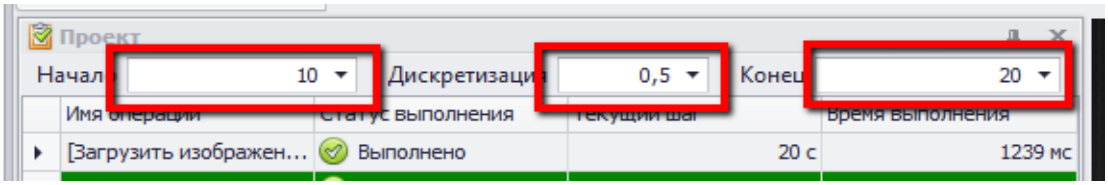


Рис. 2. Панель ввода времени начала обработки, конца и дискретизации

Задание времени выполнения, специфичного для конкретного плагина, производится в окне настройки этого плагина. На следующем рисунке (рис. 3) представлено, как задается время начала и время дискретизации для плагина «Загрузка изображений».

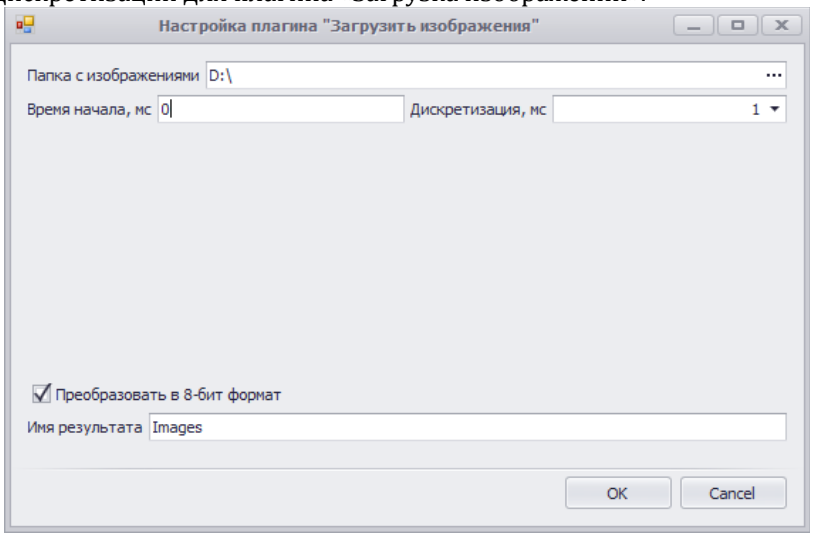


Рис. 3. Окно настройки плагина «Загрузка изображений»

В основе модулей DIP Studio 3 лежит тот же принцип динамически подключаемых плагинов, как и в DIP Studio 2.5. Подключаемые модули представляют собой отдельные сборки.

Программный комплекс автоматически загружает их из подпапки Plugins папки, где установлен программный комплекс. Программный комплекс предоставляет набор предустановленных модулей. Также разработчик может реализовать свой собственный плагин, реализующий требуемый метод обработки.

Программный комплекс согласно настройкам, заданным в панели времени (рис. 3), передает отсчет времени, для которого выполняется данное действие, в плагин. Плагин же, в свою очередь, может обрабатывать полученный отсчет времени согласно своей логике выполнения. Можно выделить три основных режима работы плагина с полученным отсчетом времени. Например, плагин может иметь свою собственную частоту дискретизации обрабатываемых видеокадров, свое время начала и конца. В данном случае, плагин может выдавать результирующие изображения со своей частотой, а не с частотой выполнения проекта. Такой режим реализован в плагине «Загрузка изображений», представленном выше. Например, время дискретизации для проекта установлено в 0,5 с, а для плагина «Загрузка изображений» задано время дискретизации 0,2 с. Таким образом, для отсчета времени 0,5 с плагин «Загрузка изображений» загрузит два видеокадра для 0,2 и 0,4 секунд. Данный подход может имитировать поступление видеокадров в систему из стороннего источника с его собственной частотой дискретизации.

Плагины, реализующие описанный выше режим, могут создать несколько видеопотоков с разной частотой дискретизации. Поэтому встает задача комплексирования этих потоков. Режим комплексирования является еще одним режимом работы плагина с поступившим отсчетом времени. Типичным примером является сложение двух видеорядов разных частот с третьей частотой сложения. Например, время дискретизации первого видеопотока – 0,4 с, второго – 0,3 с, производится сложение двух видеопотоков с частотой 0,5 с. Время начала обоих потоков и начала сложения одинаково и равно 0 с. Таким образом, суммой изображений для первой секунды будет сумма изображения первого видеопотока, полученного на 0,8 с, и изображения второго видеопотока, полученного на 0,9 с. Такой режим реализован в плагине «Сложение изображений» (рис. 4). Данный плагин не имеет поля ввода для частоты сложения и производит сложение для отсчета времени, переданного программным комплексом.

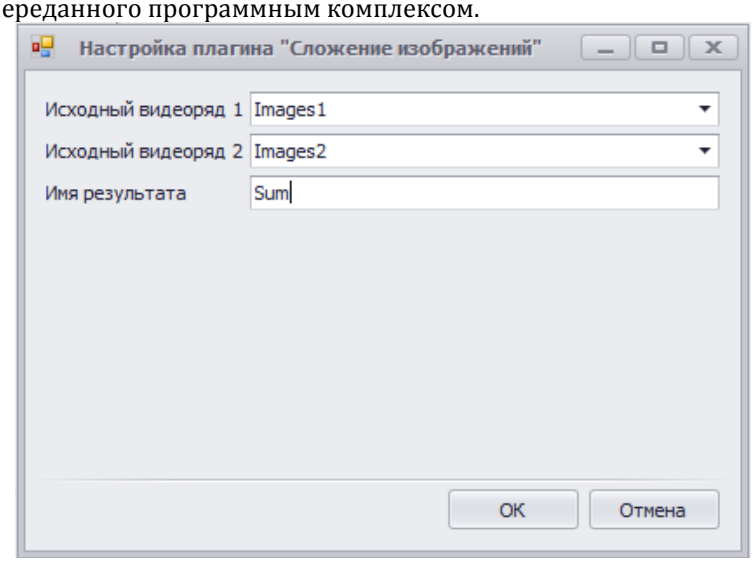


Рис. 4. Окно настройки плагина «Сложение изображений»

Однако большинство предустановленных плагинов реализуют третий режим работы. Плагин выдает результирующие данные с шагом, равным времени дискретизации проекта. В качестве примера рассмотрим плагин «Размерная фильтрация», чей интерфейс приведен на рис.5. Входными данными плагина является таблица выделенных на видеокадрах объектов, результатом работы является таблица объектов, отфильтрованных по величине площади. Допустим, что исходные объекты поступают с частотой 0,3 с, время дискретизации проекта – 0,5 с. Таким образом, результатом работы плагина является таблица с шагом 0,5 с, а, например, для первой секунды обработки плагин вернет отфильтрованные объекты, выделенные на 0,9 с.

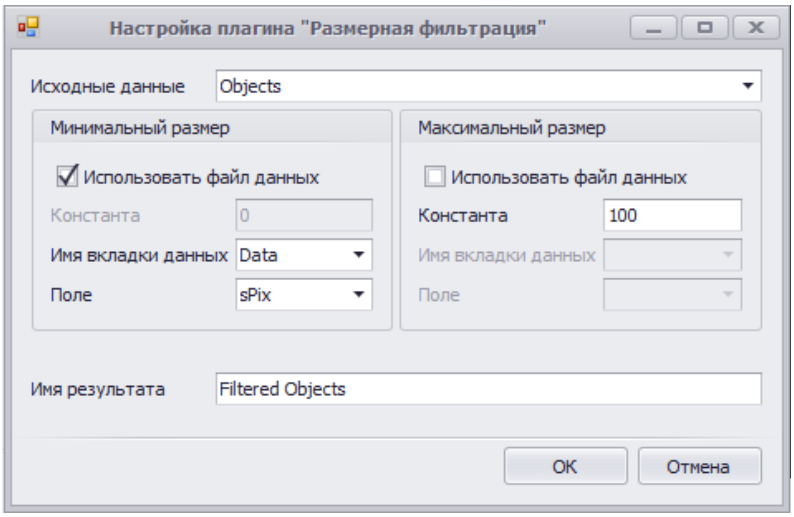
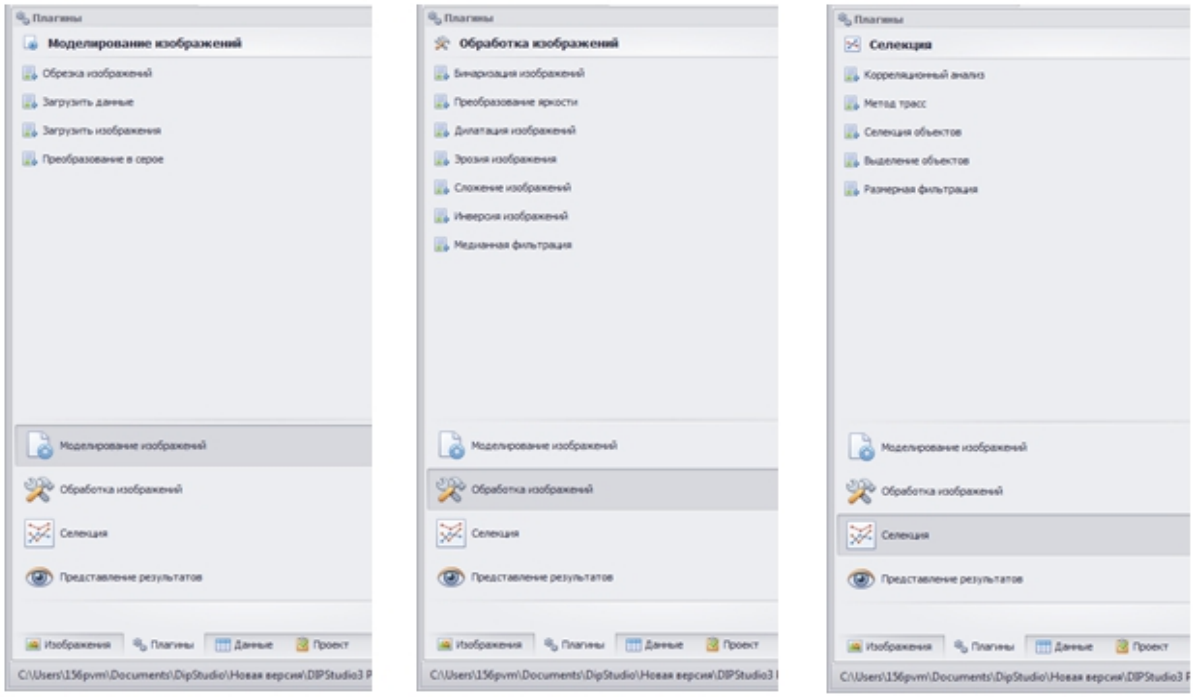


Рис. 5. Окно настройки плагина «Размерная фильтрация»

Интерфейс программного комплекса DIP Studio 3 предлагает четыре основных панели:
- Плагины (рис. 6);
- Проект (рис. 7);
- Изображения (рис. 8);
- Данные (рис. 9)

и окно просмотра изображения (рис. 10).



а *б* *в*

Рис.6.Панель «Плагины» для режима: а – моделирование изображений; б – обработка изображений; в - селекция

Панель «Плагины» (рис. 6) предоставляет пользователю возможность выбрать нужные модули и добавить их в проект. Плагины разделены на несколько категорий согласно их задачи. Так, выделены группы:

- «Моделирование изображений», в которой представлены модули, моделирующие источник видеокадров (см. рис. 6а);
- «Обработка изображений», в которой представлены плагины для предварительной обработки входящих видеокадров (см. рис. 6б);
- «Селекция», в которой представлены модули для траекторного анализа и селекции

полученных траекторий (см. рис. 6в);

- «Представление результатов», в которой собраны плагины для отображения результатов работы программы в удобном для пользователя виде.

Панель «Проект» (рис. 7) отображает список выбранных для текущего проекта плагинов. Пакетный или покадровый режим работы задается до начала формирования проекта в главном меню. Во время выполнения проекта на данной панели идет индикация плагина, который выполняется в данный момент, и отсчет времени, для которого он выполняется (для покадрового режима), а также общее время выполнения плагина.

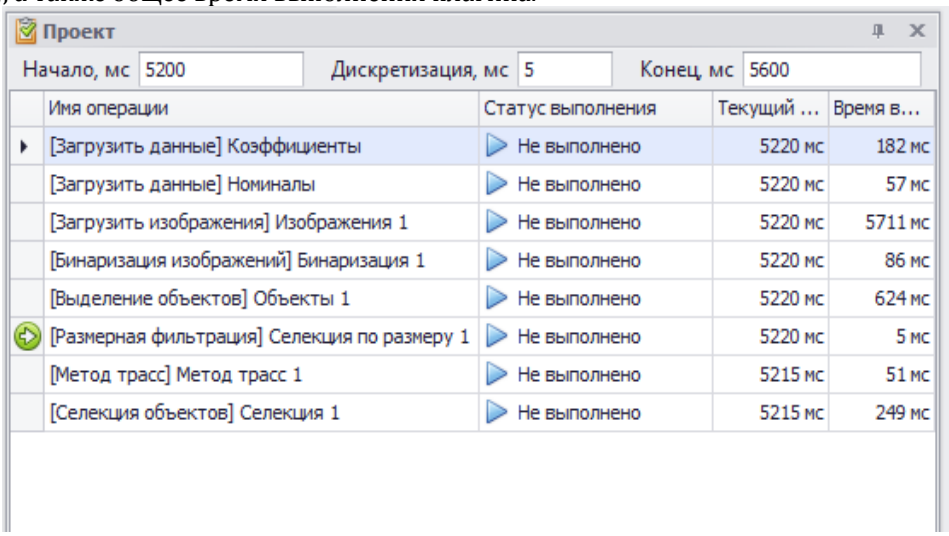


Рис. 7.Панель «Проект»

На панель «Изображения» (рис. 8) выводятся результирующие изображения работы плагинов. Изображения сгруппированы во вкладки для каждого плагина.

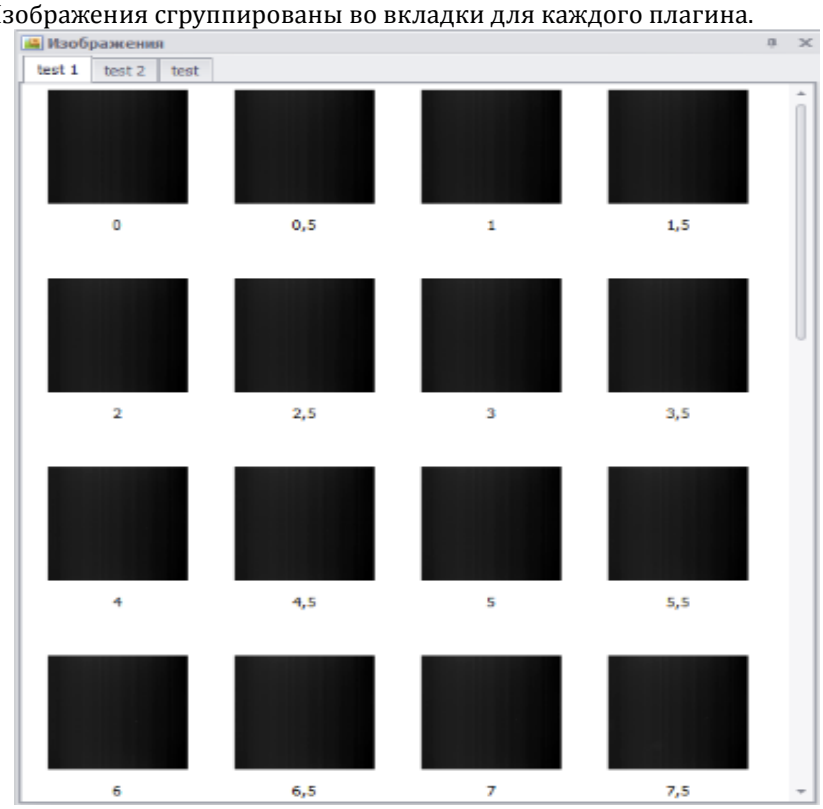


Рис. 8.Панель «Изображения»

На панель «Данные» (рис. 9) выводятся результаты работы плагинов в табличном виде. Здесь можно вывести координаты объектов на видеокадрах, их траектории.

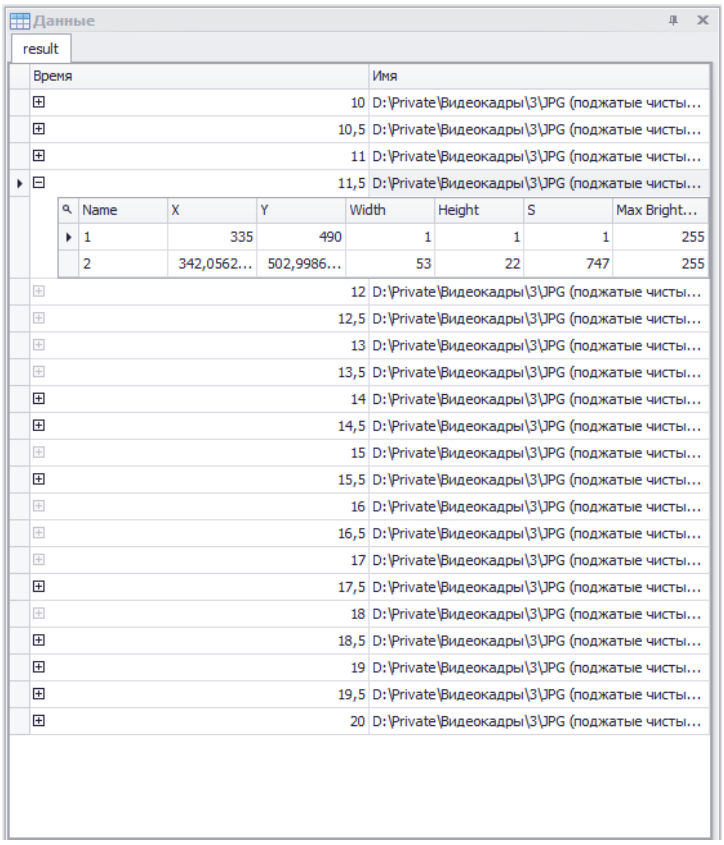


Рис. 9.Панель «Данные»

В окне просмотра изображений (рис. 10) обеспечивается увеличение и уменьшение масштаба изображения, поворот изображения, выделение части изображения, навигация между изображениями, построение гистограммы, инверсия изображения.

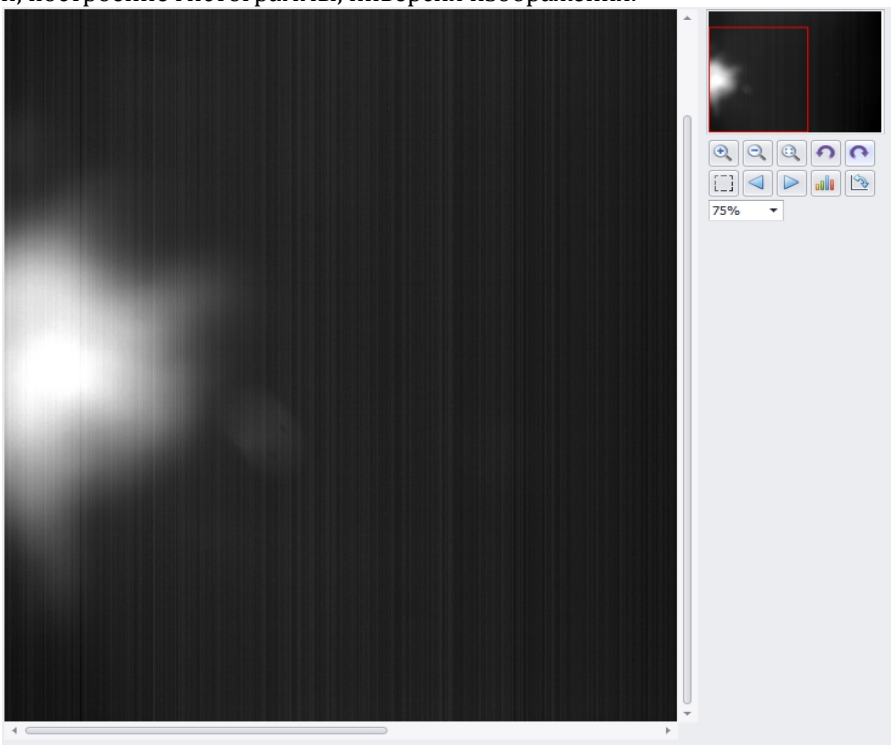


Рис. 10.Окно просмотра изображений

Таким образом, разработан программный комплекс моделирования последовательности

методов видеообработки для задач управления. Основным преимуществом реализованной возможности покадрового выполнения проекта является моделирование выполнения методов обработки, чья логика зависит от выходных данных, полученных от других методов для текущего момента времени. Примером, где может понадобиться такая связь ранних и поздних действий, является формирование размеров и наложение строба на изображение по текущим выделенным координатам объекта и дальнейшая обработка только той области, которая входит в пределы строба.

Литература

1. Егоров Д.Б., Понятский В.М. Программный комплекс обработки последовательности видеоизображений // VIII Междунар. Конф. «Современные информационные технологии и ИТ-образование», М.: Изд-во МГУ, 2013. Ч 2.

Тархов Д.А[1]., Симакина А.А.[2], Суднева А.И.[3]

[1]Государственное образовательное учреждение высшего профессионального образования «Санкт-Петербургский Политехнический университет Петра Великого», г. Санкт-Петербург, dtarkhov@gmail.com

[2]Государственное образовательное учреждение высшего профессионального образования «Санкт-Петербургский Политехнический университет Петра Великого», г. Санкт-Петербург, студент, Aleksandra.simakina@gmail.com

[3]Государственное образовательное учреждение высшего профессионального образования «Санкт-Петербургский Политехнический университет Петра Великого», г. Санкт-Петербург, студент, Sudneva-anastasiya@gmail.com

ОБРАБОТКА ДАННЫХ МЕТОДОМ ТРЕУГОЛЬНЫХ ПРИБЛИЖЕНИЙ

КЛЮЧЕВЫЕ СЛОВА

Нейронные сети, новый метод аппроксимации.

АННОТАЦИЯ

Предлагается новый метод построения нейронной сети с радиальными базисными функциями или персептрона с одним скрытым слоем по зашумлённым данным. Полученную нейронную сеть можно использовать для последующего дообучения (уточнения параметров). Метод основан на приближении выборки суммой функций, графиками которых являются равнобедренные треугольники. Для приближения каждой такой функции гауссианом или парой сигмоид используется заранее полученное оптимальное приближение стандартной треугольной функции. Рассматриваются четыре варианта организации алгоритма, проведено их сравнительное тестирование в ряде численных экспериментов. Данный подход может быть применён к построению гибридных алгоритмов поиска решений дифференциальных уравнений, сочетающего классические и нейросетевые методы.

Введение

Нейронные сети достаточно часто применяются для поиска скрытых зависимостей в данных [1-3]. Этими данными могут быть финансовые показатели, результаты экспериментов, медицинские наблюдения и иные временные ряды, связанные с наблюдением за некоторой сложной системой. Главная проблема, которую приходится решать при их обучении состоит в большой длительности этого процесса, если используются сети с десятками, а не единицами нейронов, что является необходимостью при достаточно точной аппроксимации сложных зависимостей. Причина этого состоит в необходимости поиска глобального экстремума функционала ошибки, поверхности уровня которого обычно бывает сильно изрезанными, что затрудняет процесс оптимизации. Метод, предлагаемый в данной работе, позволяет построить достаточно точное приближение к искомой нейронной сети, что сильно сокращает и упрощает процесс обучения.

Идея метода заключается в приближении данных кусочно-линейной функцией, которая представляется линейной комбинацией базисных функций, графики которых имеют вид равнобедренного треугольника. Дальше, используя заранее вычисленное приближение стандартной функции линейной комбинацией базисных функций нейросетевого типа, получаем искомую предобученную нейронную сеть.

Данный метод разумно применять в гибридных алгоритмах построения параметризованных нейросетевых решений [2-5] дифференциальных уравнений и систем. При этом сперва с помощью метода сеток или конечных элементов строятся поточечные или кусочно-линейные решения для набора фиксированных значений параметров, далее с помощью рассмотренных в данной работе методов строятся нейросетевые приближения, которые затем уточняются обычным образом.

Был проведён ряд численных экспериментов с четырьмя алгоритмами, реализующими указанный подход. Приведём их результаты в случае, когда искомая нейронная сеть имеет один

вход и один выход. Обобщение на многомерный случай получается естественным образом. Пусть при наблюдении наблюдений за некоторым процессом был получен набор точек $\{x_i, y_i\}$.

Алгоритм 1. Треугольник с вершиной в экстремуме

Шаг 1. Среди полученных точек выбираем точку с максимальной по модулю ординатой, которую М, а её абсциссу X_m. Зеркально отражаем все точки, абсциссы которых больше X_m относительно М.

Шаг 2. Используя метод наименьших квадратов, строим линейную зависимость, график которой проходит через точку М ($y=ax+b$).

Шаг 3. Отражаем получившуюся прямую так, чтобы образовался равнобедренный треугольник, основание которого лежит на ОХ. Описываем треугольник, как кусочно-заданную функцию.

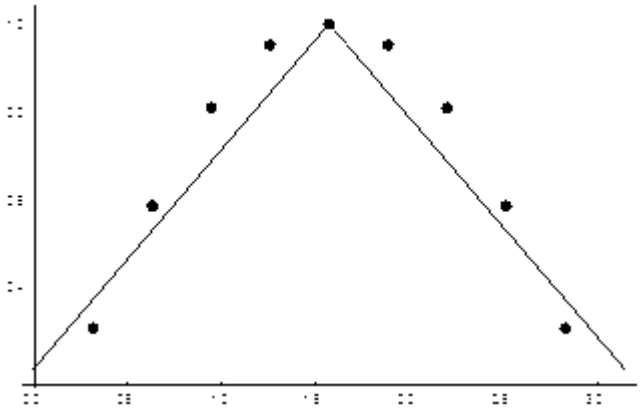


Рис.1 Приближение поточечной зависимости треугольной функцией

Шаг 4. Рассматриваем разность ординат y_i точек, не лежащих на треугольнике, и получившихся треугольных функций (стоит учесть, что функция за пределами треугольника равна 0).

Шаг 5. В результате получаем новый набор точек, с которым проделываем вышеперечисленные итерации N раз. Следует заметить, что чем больше N, тем меньше получим ошибку на выходе, однако при больших N мы приближаем не только искомую функцию, но и шум.

Шаг 6. В итоге получается N треугольников, каждый из которых аппроксимируем гауссианами или сигмоидами. Приближение гауссианами даёт RBF-сеть (нейронную сеть с радиальными базисными функциями), а сигмоидами – персептрон с одним скрытым слоем.

Метод треугольных приближений был протестирован на различных наборах точек. Рассмотрим подробнее случай синуса с добавкой белого шума, имеющее равномерное распределение с амплитудой 0.3.

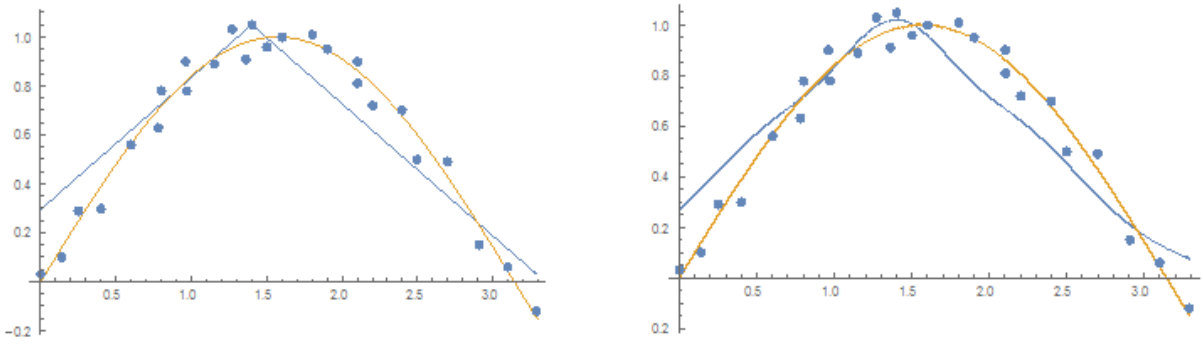


Рис.2 График синуса, зашумлённая выборка, приближение её треугольной функцией и RBF-сетью

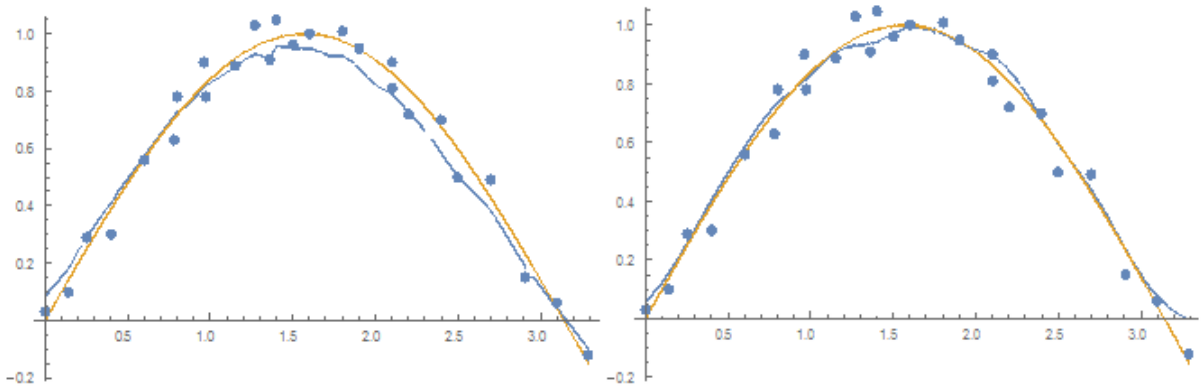


Рис.3 График синуса, зашумлённая выборка, приближение её ломаной (N=20) и RBF-сетью.

Из приведённых графиков видно, что после оптимального числа шагов получается приближение, точность которого превышает точность исходных данных. Дальнейшая работа алгоритма только ухудшает конечный результат.

Было интересно проверить поведение рассматриваемого алгоритма для более сложных функций. Приведем пару графиков, иллюстрирующих его работу.

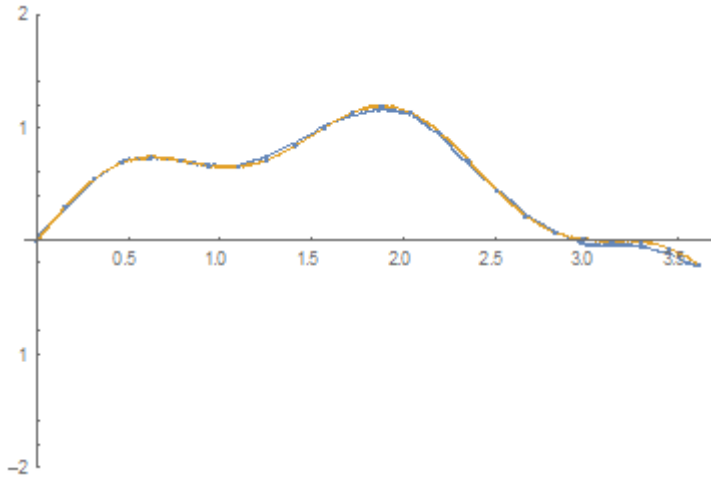


Рис.4 График Sin(x)+Sin(4x)/4, выборка, приближение RBF-сетью с помощью алгоритма 1 (для 30 точек и 30 базисных функций)

Данный график показывает, что усложнение приближаемой функции не слишком сказывается на работоспособности алгоритма.

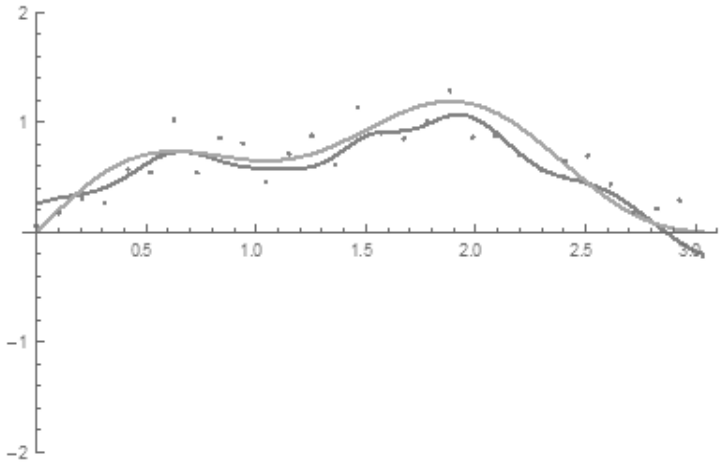


Рис.5 sin(x)+sin(4x)/4,зашумленная выборка, приближение RBF-сетью с помощью алгоритма 1 (для 30 точек и 30 базисных функций)

Видно, что добавление шума вносит искажение, которое, однако, меньше амплитуды шума. Вид графика позволяет сделать вывод, что получившаяся RBF сеть приближает не только целевую функцию, но и шум. Это может свидетельствовать о неоптимальном числе слагаемых. Данный вопрос будет далее исследован подробнее.

Следующие алгоритмы являются вариациями первого.

Алгоритм 2. В первом алгоритме изменим второй шаг. Как и прежде будем строить линейную зависимость, но теперь, строя её, мы не требуем, чтобы точка, соответствующая максимальной координате по оси ординат являлась вершиной треугольника.

Приведём графики для данногоалгоритма:

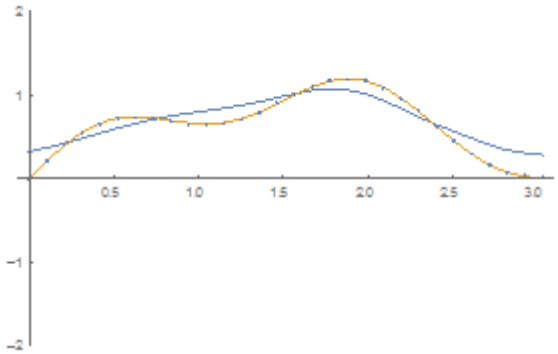


Рис.6 sin(x)+sin(4x)/4,выборка, приближение RBF-сетью с помощью алгоритма 2 (для 30 точек и 30 базисных функций)

На первый взгляд, данный график свидетельствует о неработоспособности рассматриваемого алгоритма, но этот вывод представляется преждевременным. Во-первых, построенную нейронную сеть можно дообучить и заранее не ясно, какой алгоритм окажется лучше с учётом доучивания. Во-вторых, алгоритм, оказавшийся хуже для одних данных, может оказаться лучше для других. Результаты дальнейшего тестирования приведены ниже.

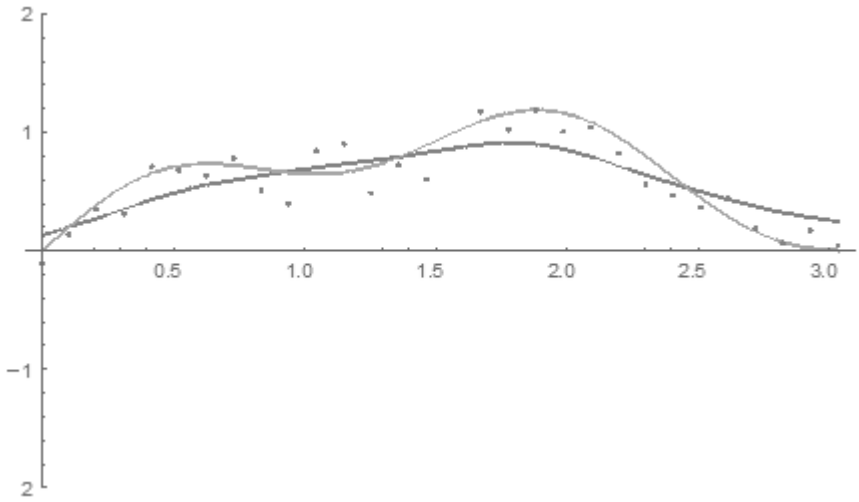


Рис.7 sin(x)+sin(4x)/4,зашумленная выборка, приближение RBF-сетью с помощью алгоритма 2 (для 30 точек и 30 базисных функций)

В данном случае оказалось, что ошибка меньше амплитуды шума, но функция оказалась чрезмерно сглаженной. Насколько этот результат хуже или лучше представленного на рис. 5 заранее неясно, это зависит от целей исследования.

Алгоритм 3. Возьмем первые четыре шага первого алгоритма. После четвертого шага запомним сумму квадратов разности ординат точек, не лежащих на треугольнике, и получившихся прямых, также запомним точку с максимальной ошибкой М. Проделаем ещё раз шаги 1-4, но теперь рассматриваем не все точки (отбросим одну самую отдалённую точку от М). Эту процедуру мы проделываем для того, чтобы посмотреть, когда будет минимальная сумма квадратов ошибок по выборке. Затем посмотрим, чему будет равна ошибка, если мы отбросим еще одну точку. Будем повторять эту процедуру до тех пор, пока не дойдем до того, что останется две точки: М и еще

одна. Исходя из нашего метода, мы можем построить треугольник минимум по двум точкам: одна из точек будет вершиной треугольника, а по второй построим одну из сторон. Если в результате эксперимента количество точек достаточно велико или мы хотим упростить вычисления, то можно “откидывать” не одну, а несколько точек, однако в этом случае можно не получить минимальную ошибку. После того, как мы получим все ошибки, ищем наименьшую, и строим соответствующий треугольник. Дальнейшие действия будут как в исходном методе.

Графики для данного метода:

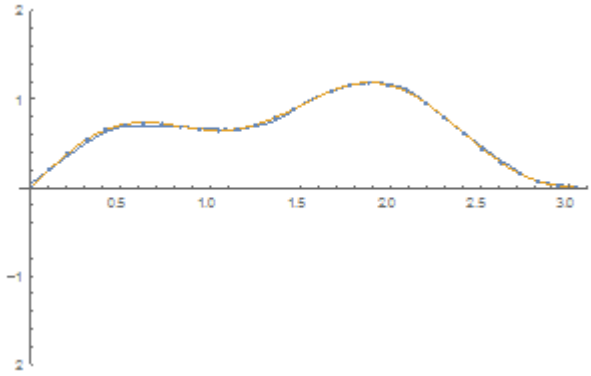


Рис.8 Sin(x)+Sin(4x)/4,выборка, приближение RBF-сетью с помощью алгоритма 3

Очевидно, что результат работы этого алгоритма при работе с точными данными лучше, чем у предыдущих двух, однако, его трудоёмкость тоже существенно выше.

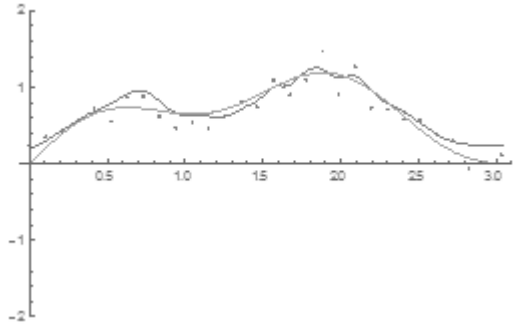


Рис.9 sin(x)+sin(4x)/4,зашумленная выборка, приближение RBF-сетью с помощью алгоритма 3

При работе с зашумлёнными данными преимущества рассматриваемого алгоритма становятся неочевидными.

Алгоритм 4. В этом методе мы объединим второй и третий. Стороны будем искать с помощью линейной регрессии, не требуя, чтобы точка, соответствующая максимальной ординате, являлась вершиной треугольника. А из метода 3 мы возьмем поиск наименьшей суммы квадратов ошибки, то есть будем откидывать точки и искать такой треугольник, для которого ошибка по выборке будет минимальна.

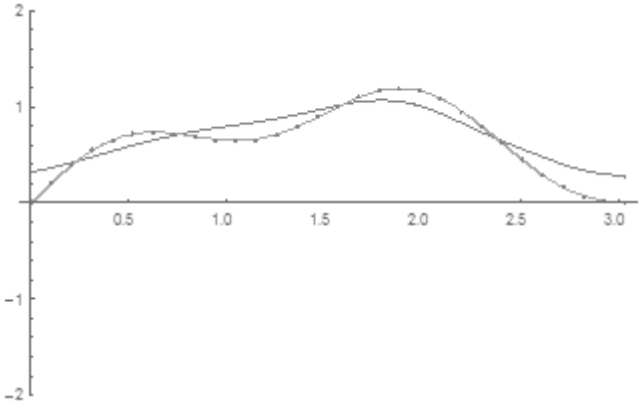


Рис.10 Sin(x)+Sin(4x)/4, выборка, приближение RBF-сетью с помощью алгоритма 4

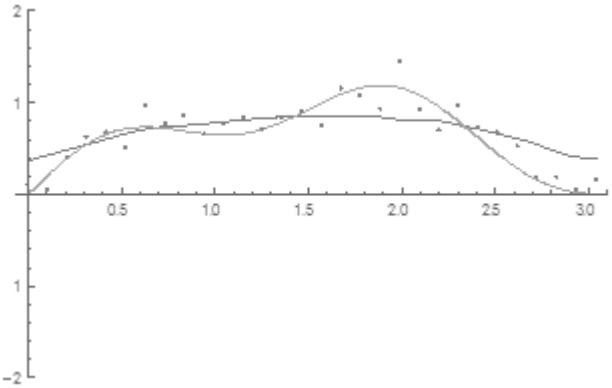


Рис.11 sin(x)+sin(4x)/4,зашумленная выборка, приближение RBF-сетью с помощью алгоритма 4

Данные графики позволяют сделать те же выводы, что и графики 6 и 7.

Результаты численных экспериментов

Рассмотрим таблицу среднеквадратических ошибок, вычисленных по истинному значению приближаемой функции для каждого метода на конкретных примерах. *Числа во втором столбце (5,15,30)- количество треугольников, по которым построена RBF-функция, δ – амплитуда ошибки, L-количество точек.*

Таблица 1.

		Sin(x), L=30		Sin(4x), L=30	Sin(x)+Sin(4x)/4, L=30			Sin(x)+Sin(4x)/4, L=200
δ		0.2	0.5	0	0	0.3	0.5	0.3
1 метод	5	0.02	0.026	0.16	0.03	0.04	0.04	0.05
	15	0.006	0.03	0.1	0.003	0.01	0.08	0.03
	30	0.006	0.02	0.05	0.006	0.02	0.06	0.009
2 метод	5	0.006	0.026	0.41	0.03	0.048	0.053	0.02
	15	0.006	0.026	0.33	0.03	0.046	0.053	0.017
	30	0.006	0.026	0.2	0.03	0.046	0.053	0.017
3 метод	5	0.005	0.01	0.06	0.01	0.02	0.1	0.05
	15	0.006	0.05	0.05	0.004	0.16	0.026	0.02
	30	0.007	0.056	0.03	0.003	0.02	0.023	0.04
4 метод	5	0.007	0.053	0.084	0.025	0.063	0.03	0.029
	15	0.007	0.053	0.06	0.025	0.02	0.03	0.027
	30	0.007	0.053	0.06	0.025	0.02	0.03	0.027

Формула для вычисления ошибок по выборке: $\sum_{i=1}^{m}(y_i - p_m(x_i))^2$, где x_i и y_i - координаты *i*-й точки выборки.

Формула для вычисления ошибок по истинному значению: $\sum_{i=1}^{L}(f(x_i) - p_m(x_i))^2$, где $f(x)$ -истинная функция, p_m - функция, равная сумме треугольных функций на m-ом шаге.

Формула для вычисления среднеквадратической ошибок по истинному значению получается из предыдущей формулы делением на количество используемых точек *L*.

По таблице видно, что результаты во многих методах неоднозначны, однако наименьшая среднеквадратическая ошибка по истинному значению функции, наблюдаемая в результате метода 3, всегда остается одной из минимальных. Так же можно заметить, что при увеличении числа экспериментальных точек, при относительно небольшом разбросе, ошибка при разных

количествах треугольников в разных методах может, как и уменьшается, так и увеличиваться. К примеру, в методе 1 минимальная ошибка будет такой же, но достигаться будет при 30, а не 15 треугольниках.

Рассмотрим графики полученных ошибок по выборке и графики ошибок по истинному значению (ось X-номера треугольников, ось Y-соответствующие ошибки).

Вначале проанализируем случаи, для которых лучше работает не только 3ий метод. Рассмотрим sin(x) с разбросом 0.5 (из таблицы видно, что для него лучше подходят методы 1 и 3).

Алгоритм 3:

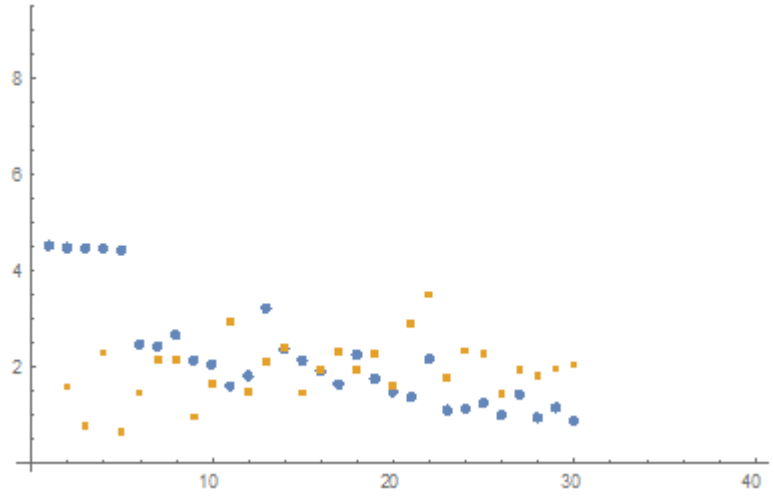


Рис.12 графики ошибки по выборке (синяя) и по истинному значению (желтая)

Алгоритм 1:

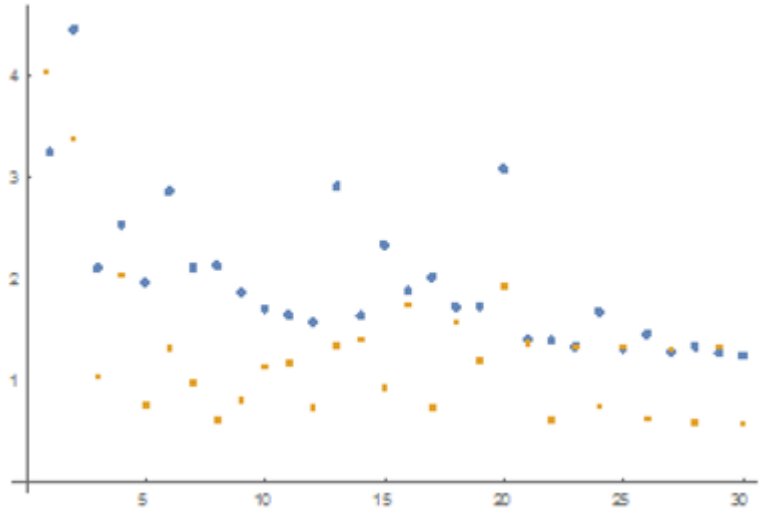


Рис.13 графики ошибки по выборке (синяя) и по истинному значению (желтая)

Во-первых, следует отметить, что ошибки по истинному значению функции ниже, чем по выборке, что говорит о хороших сглаживающих качествах нейронной сети и о возможности использовать ошибку по выборке как индикатор ошибки по истинному значению. Во-вторых, ошибка по истинному значению меньше при работе 3-го алгоритма, чем 1-го для $N \leq 5$, дальше она начинает возрастать. Этой тенденции не видно по ошибке по выборке, но она после седьмого и восьмого шагов начинает колебаться и убывает очень медленно, что говорит о нецелесообразности добавления новых функций. То же можно сказать и про первый алгоритм после пятого шага.

Рассмотрим функцию sin(x)+sin(4x)/4 с разброс 0.5. Это ещё один спорный случай из таблицы 1, проанализируем его, используя методы 3 и 4.
Алгоритм 3:

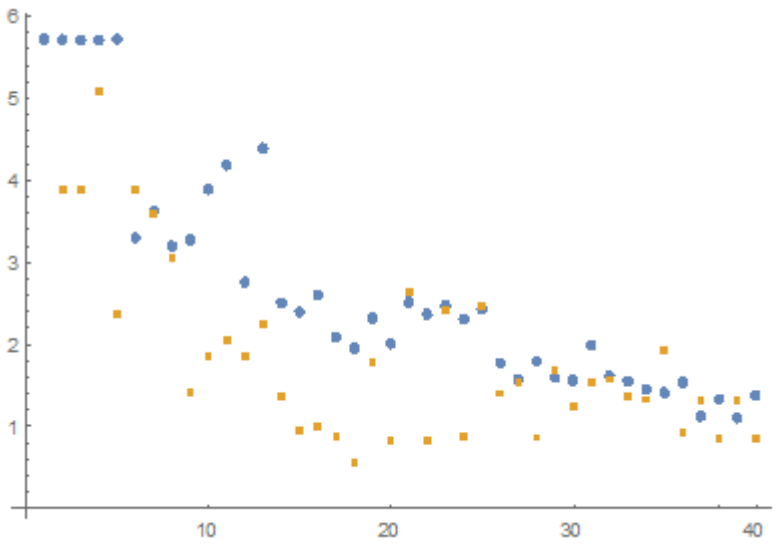


Рис.14 графики ошибки по выборке (синяя) и по истинному значению (желтая) при работе третьего алгоритма

Метод 4:

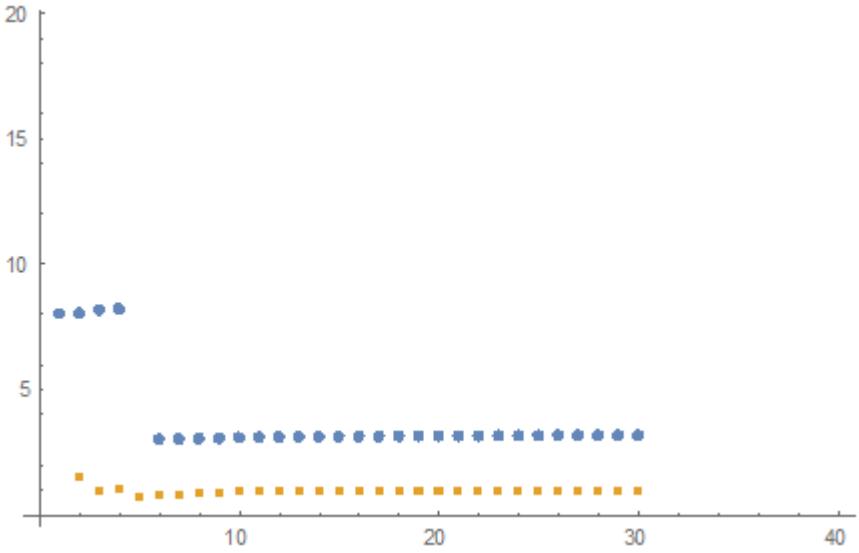


Рис.15 графики ошибки по выборке (синяя) и по истинному значению (желтая) при работе четвёртого алгоритма

При работе четвёртого алгоритма минимальная ошибка достигается после трёх шагов и дальше не изменяется. При работе третьего алгоритма ошибки по выборке и по истинному значению будет меньше, чем при работе четвёртого алгоритма, но она будет достигаться на 18 шаге. В этом случае оба алгоритма работают достаточно эффективно, алгоритм 3 точнее показывает полученную зависимость, но потребует больших расчетов, чем алгоритм 4. Следует заметить, что данные алгоритмы целесообразно применять для достаточно сложных целевых функций, для которых первый алгоритм оказывается недостаточно эффективным.

Проанализируем подробнее работу 3 алгоритма.

Возьмем функцию sin(x)+sin(4x)/4 без разброса.

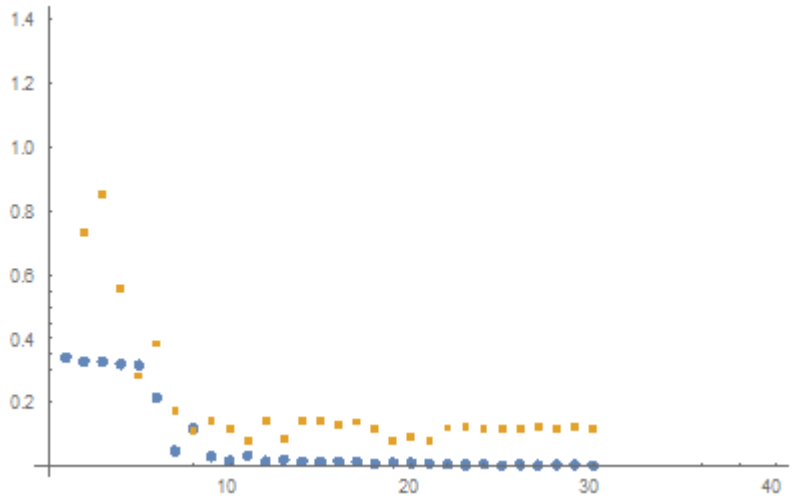


Рис.16 графики ошибки по выборке (синяя) и по истинному значению (желтая)

В данном случае работу алгоритма можно закончить после 7 шагов, причём ошибка по выборке является достаточно хорошим индикатором необходимости прервать его работу. Аналогичные выводы можно сделать и для других численных экспериментов:

- Алгоритм целесообразно применять для заметно меньшего числа шагов, чем число точек выборки.
- Резкое замедление убывания ошибки по выборке является хорошим критерием окончания работы алгоритма.
- Дальнейшее продолжение работы алгоритма целесообразно, если предполагается доучивание получившейся нейронной сети, а погрешность измерений (наблюдений) намного меньше достигнутого среднеквадратичного отклонения.

Можно сделать общий вывод, что третий алгоритм лучше всего приближает нашу функцию, но в случаях с большим разбросом можно также воспользоваться алгоритмами 1 и 4, причём приведённые выше выводы справедливы и для них. Целесообразность применения алгоритма 1 вытекает из его существенно меньшей трудоёмкости, в особенности это актуально при обработке данных в режиме реального времени. Алгоритм 3 является более универсальным в случае ограниченности данных и достаточных вычислительных ресурсов, так как он работает для всех случаев достаточно точно.

Заметим, что изученные алгоритмы достаточно легко распространяются на многомерный случай.

Благодарность: *Статья подготовлена по результатам исследования, выполненного при финансовой поддержке гранта Российского Научного Фонда (проект 14-38-00009) «Программно-целевое управление комплексным развитием Арктической зоны РФ (Санкт-Петербургский политехнический университет Петра Великого).*

Литература

1. Бастенс Д.Э., Ван ден Берг, Вуд Д. Нейронные сети и финансовые рынки: принятие решений в торговых операциях. – М: ТВП, 1998. – 236с.
2. Васильев А.Н., Тархов Д.А .– Нейросетевое моделирование. Принципы. Алгоритмы. Приложения. – СПб.:Изд-во Политехн. Ун-та. 2009. - 528 с.
3. Тархов Д.А. Нейросетевые модели и алгоритмы. – М.: Радиотехника, 2014. – 352 с.
4. Vasilyev A.N., Tarkhov D.A. Mathematical Models of Complex Systems on the Basis of Artificial Neural Networks// Nonlinear Phenomena in Complex Systems. - vol. 17, no.3, pp. 327-335 (2014).
5. N. U. Kainov, D. A. Tarkhov, T. A. Shemyakina. Application of Neural Network Modeling to Identification and Prediction Problems in Ecology Data Analysis for Metallurgy and Welding Industry// Nonlinear Phenomena in Complex Systems. vol. 17, no. 1, pp. 57-63 (2014).

Тарасенко Ф.Д.[1], Тархов Д.А.[2]

[1]Санкт-Петербургский политехнический университет Петра Великого, Санкт-Петербург, Россия, студент института прикладной математики и механики, oudi@mail.ru

[2]Санкт-Петербургский политехнический университет Петра Великого, Санкт-Петербург, Россия, д.т.н., профессор, dtarkhov@gmail.com

СРАВНИТЕЛЬНЫЙ АНАЛИЗ ПРИМЕНЕНИЯ РАЗЛИЧНЫХ БАЗИСНЫХ ФУНКЦИЙ В АЛГОРИТМАХ ПОСЛЕДОВАТЕЛЬНОГО СГЛАЖИВАНИЯ ДАННЫХ

КЛЮЧЕВЫЕ СЛОВА

Последовательные алгоритмы, обработка данных, RBF-сети, сплайн.

АННОТАЦИЯ

Исследованы алгоритмы построения зависимости по экспериментальным данным, построенные на основе последовательной обработки точек по одной. Рассмотрены четыре алгоритма обработки экспериментальных данных и проведен их сравнительных анализ для разных исходных данных, базисных функций и наборов параметров. Исходные данные имели не только статический, но и динамический характер, когда искомая модель менялась в процессе работы алгоритма.

Проведены численные эксперименты, позволившие на основе результатов сравнительного анализа сделать выводы об эффективности алгоритмов в тех или иных случаях, и дать рекомендации по их использованию.

Введение. В данной работе, вслед за [1-3] рассматриваются методы нахождения зависимости $y=f(x)$ по экспериментальным данным $(x_1, y_1),(x_2, y_2),\dots,(x_N, y_N)$ в ситуации, когда точки (x_i, y_i) поступают и обрабатываются по одной, что может быть связано с необходимостью обрабатывать данные в режиме реального времени. Будем искомую зависимость представлять в виде $y=\sum_{j=1}^{n} c_j \varphi_j(x)$, где $\varphi_j=\varphi(\alpha_j|x-z_j|)$ или $\varphi_j=\varphi(\alpha_j(x-z_j)^2)$. Функции такой конструкции принято называть RBF-сетями [3,4]. Подбор параметров c_j, α_j и z_j называется обучением сети. В данной статье рассматривается случай, когда при обучении подбираются только коэффициенты c_j.

Как известно, любую кусочно-линейную функцию можно представить в виде суммы $\sum_{j=1}^{n} c_j \varphi(\alpha_j(x-z_j))$, если выбрать в качестве базисной функции треугольную шапочку. Аналогичным образом можно любой сплайн разложить по базисным функциям, соответствующим его степени и гладкости [6].

В численных экспериментах использовались следующие базисные сплайны [6]: парабола, кубическая парабола, гауссиан.

Нами исследовано несколько алгоритмов сглаживания данных:

1) Обработка точек по одной с адаптацией веса базисной функции с ближайшим центром.

2) То же, но с адаптацией весов двух ближайших функций.

3) то же, что и в 2) но предполагается линейная зависимость скорости адаптации веса от близости абсциссы добавляемой точки к центру базисной функции [1].

4) Нахождение оптимальных коэффициентов разложения по базисным функциям решением линейной системы.

Для подходов 1 и 3 исследовалась также вариация алгоритма с заданным заранее законом изменения шага.

В подходах 1-3 основой для адаптации коэффициентов является минимизация

квадратичного функционала ошибки.

Подход 1.

Начальные значения коэффициентов $c_j = 0$. При переходе от N-1 наблюдений к N изменение весового коэффициента базисной функции с ближайшим центром, обозначенное $\Delta_k(N)$, получается равным

$$\Delta_k(N) = \frac{Q_k}{S_k}, \tag{1}$$

где

$$S_k = \sum_{i=1}^{N} \varphi_k^2(x_i), \quad Q_k = \sum_{i=1}^{N} \varphi_k(x_i)\delta_i(N). \tag{2}$$

При переходе к следующему шагу ошибки нужно пересчитать по формуле

$$\delta_i(N+1) = \delta_i(N) - \Delta_k \varphi_k(x_i). \tag{3}$$

Подход 2.

В отличие от первого алгоритма выбираются две базисные функции, между центрами которых находится поступившее наблюдение.

Изменение весовых коэффициентов производится по формулам

$$\Delta_k = \frac{Q_k S_{k+1} - Q_{k+1} P_k}{S_k S_{k+1} - P_k^2}, \ \Delta_k = \frac{Q_k S_{k+1} - Q_{k+1} P_k}{S_k S_{k+1} - P_k^2} \tag{4}$$

где

$$P_k = \sum_{i=1}^{N} \varphi_k(x_i)\varphi_{k+1}(x_i). \tag{5}$$

При обращении в ноль знаменателя (2) используются формулы (7) и (8).

При переходе к новому шагу слагаемые в суммы (2) и (5) добавляются для всех базисных функций, для которых $\varphi_k(x_i) \neq 0$.

Ошибки δ_i при переходе к следующему шагу нужно пересчитать по формуле

$$\delta_i(N+1) = \delta_i(N) - \Delta_k \varphi_k(x_i) - \Delta_{k+1}\varphi_{k+1}(x_i). \tag{6}$$

Подход 3.

Пусть x_N находится между центрами функций с номерами k и $k+1$. В прежних обозначениях используем формулы [1]:

$$\Delta_k(N) = \Delta(N)\lambda \text{ и } \Delta_{k+1}(N) = \Delta(N)(1-\lambda), \tag{7}$$

где в [1] рекомендуется выбирать $\lambda = \frac{z_{k+1} - x_N}{z_{k+1} - z_k}$.

Число $\Delta(N)$ находится минимизацией функционала ошибки и получается равным

$$\Delta(N) = \frac{\lambda Q_k + (1-\lambda)Q_{k+1}}{\lambda^2 S_k + 2\lambda(1-\lambda)P_k + (1-\lambda)^2 S_{k+1}}. \tag{8}$$

Ошибки δ_i при переходе к следующему шагу нужно пересчитать по формуле

$$\delta_i(N+1) = \delta_i(N) - \Delta(N)(\lambda\varphi_k(x_i) + (1-\lambda)\varphi_{k+1}(x_i)). \tag{9}$$

Для алгоритмов 1 и 3 можно вместо формул (1), (7) и (8) использовать убывающие по заданному закону $\Delta(N)$, например $\Delta(N) = \left(1 - \frac{2}{T}\right)^N \delta_k(N)$, где $T = 5n + N_{max}$ [1]. Это существенно сокращает объём вычислений.

Подход 4.

Этот подход предусматривает построение линейной регрессии и предполагает решение системы линейных уравнений, которое можно получать рекуррентным образом при появлении нового наблюдения [3,4,7].

Результаты численных экспериментов. Проведено сравнительное тестирование алгоритмов, реализующих подходы 1-4 для указанных выше базисных функций для разных величин ошибки «измерений», количества точек и числа базисных функций. Также, проанализированы случаи динамических (переменных во времени) данных при различном наборе

параметров и выборе базисных функций. Интервал изменения аргумента [0;1].

Ниже приведены иллюстрации работы алгоритмов, некоторые наиболее характерные данные численных экспериментов вы и выводы, полученные при анализе результатов этих экспериментов.

На первом рисунке представлен результат обработки экспериментально зависимости, полученной генерацией точек около функции $\sin(\pi x)+0.1\sin(10\pi x)$, с использованием первого алгоритма и формулы из [1] для 20-ти базисных функций, 400-сот экспериментальных точек, ошибки «измерений» – 0,1 и 10-ти прогонов алгоритма.

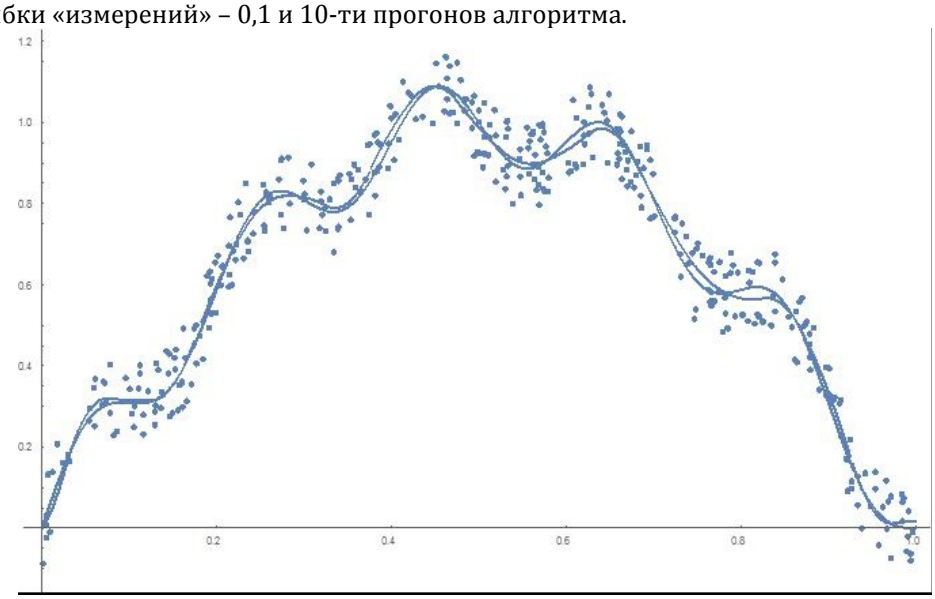


Рисунок 1. Результат работы первого алгоритма для статических данных

На втором рисунке представлены результаты применения второго подхода для динамических данных. Приближаемая функция – *sin(wπx)*, где *w* меняется за время вычислений от 1 до 4,5. Базисная функция – кубическая парабола. 300 экспериментальных точек. 10 базисных функций. Ошибка измерений –0,1. Изображены три момента в которые исходная функция имеет вид соответственно: *sin(πx), sin(2,25πx), sin(4,5πx).*

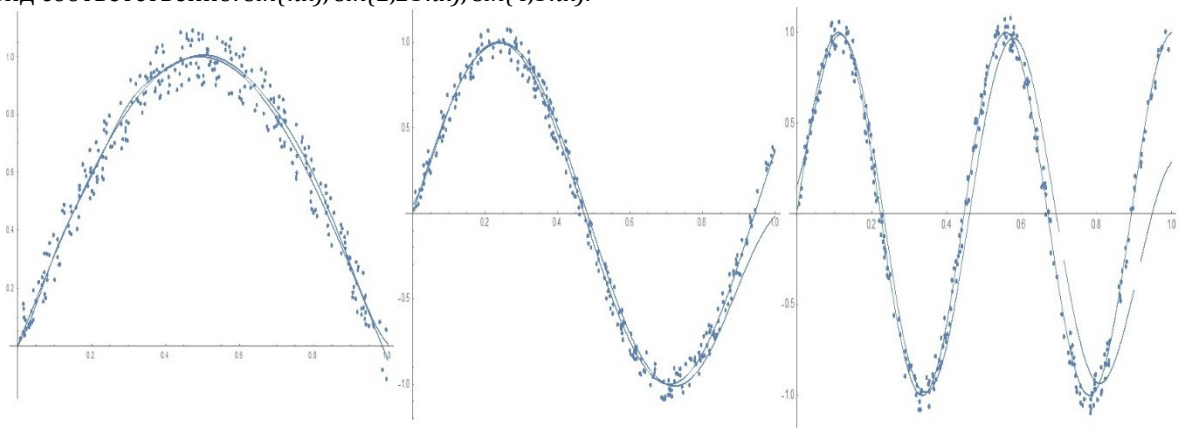


Рисунок 2: Результаты применения второго подхода для динамических данных

Таблица 1: Математическое ожидание ошибки для восьми тестов алгоритма при 50 точках наблюдений. Статические данные. Приближаемая функция - sin(πx). Число базисных функций – 10. Число экспериментальных точек – 50.

ε	*m*	Подход	Треугольная шапочка	Парабола	Кубическая функция	Гауссиан
0.01	10	1 Т.	0.0045	0.0359	0,0251	0,0337
1	10		0.2880	0.3115	0,0404	0,0308
0.01	1		0.0247	0.0363	0,0536	0,0209

0.01	10	1 Р.	0,0047	0,0050	0,0056	0,3342
1	10		0,0320	0,0406	0,0361	0,0355
0.01	1		0,1888	0,1126	0,1130	0,1313
0.01	10	2	0,0039	0,0039	0,0054	0,0038
1	10		0,0331	0,0330	0,0281	0,0343
0.01	1		0,0057	0,0150	0,0195	0,0146
0.01	10	3 Т.	0,0132	0,0125	0,0238	0,0217
1	10		0,0292	0,0378	0,0387	0,0398
0.01	1		0,0464	0,0403	0,0365	0,0487
0.01	10	3 Р.	0,0044	0,0044	0,0053	0,0422
1	10		0,0278	0,0342	0,0358	0,0579
0.01	1		0,1016	0,0975	0,0945	0,1041
0.01		4	0,0031	0,0035	0,0040	0,0080
1			0,2838	0,2542	0,2633	0,2901

Здесь и дальше, ε - погрешность наблюдений, m - число прогонов алгоритма, n – кол-во точек наблюдений, Р. – регулируемый шаг в соответствии с формулой из [1], Т. – точный в соответствии с формулами, приведёнными выше для каждого алгоритма.

Таблица 2: Математическое ожидание ошибки для четырех тестов алгоритма. Динамические данные. Приближаемая функция – sin(wπx), где w меняется за время вычислений от 1 до 2 и до 4,5. Число базисных функций – 10. Ошибка измерений – 0,1.

m	*n*	*w*	Подход	Треугольная шапочка	Парабола	Кубическая функция	Гауссиан
100	300	4,5	1	0,4252	0,4084	0,4060	0,4360
50	100	2		0,1908	0,1910	0,1755	0,1811
100	300	4,5	2	0,2880	0,2624	0,2481	0,2348
50	100	2		0,1454	0,1361	0,1321	0,1233
100	300	4,5	3	0,4176	0,4296	0,4422	0,4767
50	100	2		0,1854	0,1815	0,1779	0,1817

Выводы

- Рассмотренные алгоритмы и базисные функции показали хорошую работоспособность в рассматриваемых задачах.
- Ни один из алгоритмов не имеет решающего преимущества над другими. Это позволяет рекомендовать самый простой из них - первый.
- Точное вычисление шага не имеет преимуществ над формулой из [1], которые могли бы компенсировать существенное увеличение вычислительной сложности, если выборка достаточно велика или продублирована необходимое число раз.
- При больших ошибках и малом числе экспериментальных точек все методы работают неудовлетворительно. Несколько лучше других получаются результаты сглаживания при использовании кубической БФ для большого количества точек и гауссиана для малого.
- Результаты, полученные при использовании разных базисных функций примерно одинаковы. Выбор конкретной функции диктуется условиями на гладкость. Если таких условий нет, то предпочтительной является треугольная функция в силу минимальности своего носителя (что приводит к минимальной вычислительной сложности). Если такие условия заранее неизвестны или достаточно жёстки, то предпочесть следует гауссиан, как имеющий бесконечную гладкость.
- Наилучшим из рассмотренных алгоритмов для сглаживания динамических данных является подход 2, так как он значительно лучше обрабатывает конец промежутка.

Благодарность: Статья подготовлена по результатам исследования, выполненного при финансовой поддержке гранта Российского Научного Фонда (проект 14-38-00009) «Программно-целевое управление комплексным развитием Арктической зоны РФ (Санкт-Петербургский

политехнический университет Петра Великого).

Литература

1. Хакимов Б.В. Моделирование корреляционных зависимостей сплайнами на примерах в геологии и экологии. М.: МГУ; СПб: Нева. 2003, 144 с.
2. Хакимов Б.В., Михеев И.М. Нелинейная модель нейрона – многомерный сплайн // Нейрокомпьютеры: разработка, применение. 2012. № 7. С. 36–40.
3. Тархов Д.А. Последовательные алгоритмы сглаживания данных// Нейрокомпьютеры: разработка, применение. 2015. № 3. С. 11–18.
4. Тархов Д.А. Нейросетевые модели и алгоритмы. – М.: Радиотехника, 2014. – 352 с.
5. Хайкин С. Нейронные сети: полный курс, 2-е издание. Пер. с англ. – М.: Изд. дом «Вильямс», 2006. – 1104 с.
6. Свиньин С.Ф. Базисные сплайны в теории отсчётов сигналов. СПб., Наука, 2003. – 118с.
7. Алберт А. Регрессия, псевдоинверсия и рекуррентное оценивание. - М.: Наука, 1977. 224с.

Тархов Д.А. [1], **Шаньшин И.К.**[2], **Шаханов Д.О.**[3]

[1]Санкт-Петербурский политехнический университет Петра Великого, г.Санкт-Петербург, д.т.н., профессор кафедра высшей математики, dtarkhov@gmail.com

[2] Санкт-Петербурский политехнический университет Петра Великого, г.Санкт-Петербург, студент института прикладной математики и механики, ivan.fizik92@yandex.ru

[3] Санкт-Петербурский политехнический университет Петра Великого, г.Санкт-Петербург, студент института прикладной математики и механики , trupsaveman@gmail.com

СРАВНЕНИЕ НЕЙРОСЕТЕВОГО И КЛАССИЧЕСКОГО ПОДХОДА К ЗАДАЧЕ ИДЕНТИФИКАЦИИ МИГРАЦИОННЫХ ПРОЦЕССОВ

КЛЮЧЕВЫЕ СЛОВА

Социодинамика, миграция, динамические системы, нейронные сети.

АННОТАЦИЯ

Рассмотрены методы решения обратной задачи моделирования миграционных потоков. Проведён сравнительный анализ классического подхода (основанного на дискретизации и линейной регрессии) и подхода, основанного использовании нейронных сетей.

ВВЕДЕНИЕ

Имеются две миграционные модели в виде [1]:

$$\begin{cases} \dfrac{dx}{dt} = sh(kx + k_1 y) - x\ ch(kx + k_1 y) \\ \dfrac{dy}{dt} = sh(k_2 x + ky) - y\ ch(k_2 x + ky) \end{cases} \quad (1)$$

$$\begin{cases} \dfrac{dx}{dt} = (k-1)x + k_1 y \\ \dfrac{dy}{dt} = k_2 x + (k-1)y \end{cases} \quad (2)$$

Здесь (1) исходная нелинейная, а так же линеаризованная в окрестности нулевого положения равновесия - система (2). На основе этих моделей [1] можно построить прогноз миграционной динамики. Для этого необходимо определить коэффициенты моделей по имеющимся данным.

Рассмотрим три способа идентификации коэффициентов рассматриваемых моделей.

Первый способ подразумевает решение системы (2) аналитически, а затем определение коэффициентов по наилучшему соответствию решения и данных. У такого подхода есть два недостатка: невозможность его распространения на нелинейную систему (1) и необходимость решать систему заново при распространении на модели более высокого порядка.

Второй способ представляет собой дискретизацию системы (2) и определение её коэффициентов по формулам для двумерной линейной регрессии. Такой подход можно распространить на более высокие порядки, но случай нелинейной системы (1) требует построения нелинейной регрессии, что также снижает его универсальность.

Третий способ подразумевает использование нейронной сети [2,3]. Для этого можно численно решить систему для достаточно большого набора параметров, затем обучить нейронную сеть для определения каждого из них по данным, после этого вычисление необходимых коэффициентов модели по наблюдениям сведётся просто к подстановке данных в нейронную сеть. Такой подход имеет ряд преимуществ, например, большую универсальность – его применение в линейном и нелинейном случае практически не различается.

РЕЗУЛЬТАТЫ ВЫЧИСЛЕНИЙ

Примем следующие обозначения: красным цветом выделен расчетный график, синим -

теоретический. Рассматривался период прогноза в 5 раз больший, чем период времени, для которого строилась модель.

Рассмотрим сначала случай, когда период времени, для которого строится модель, составляет от нуля до единицы.

Приведём результаты применения второго и третьего методов для значений коэффициентов $k=1.2$; $k_1=-0.5$; $k_2=0.5$.

1) Второй метод (линейная регрессия).

При количестве трёх входных точках – 0, 0.5 и 1, мы получили сильное расхождение расчетной модели с теоретической, однако форма графика не искажена, что легко видеть из рис.1.

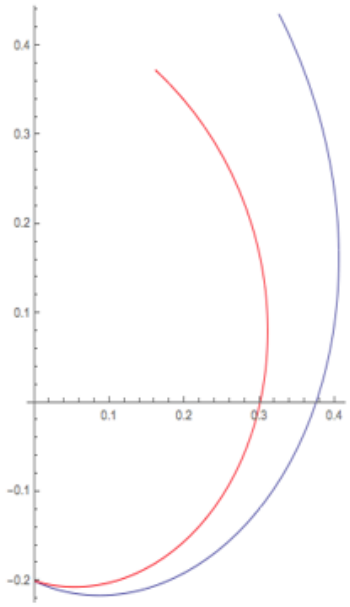


Рисунок 1

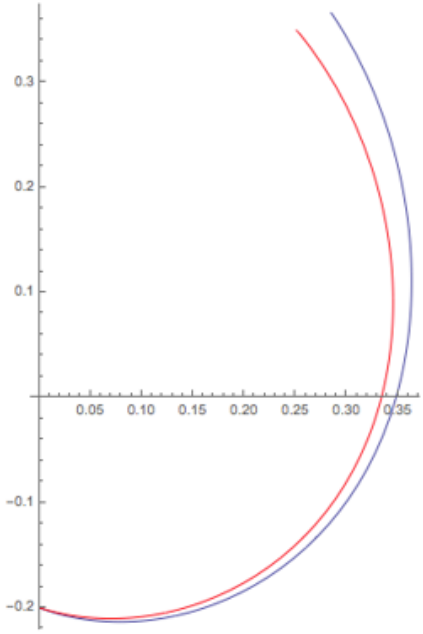


Рисунок 2

С увеличением количества точек до 10 в окрестности начала координат имеем неплохое приближение, но погрешность все еще велика (рис.2). Далее, увеличивая количество точек до 100, мы достигаем отличной точности, графики функций теоретической и расчетной модели практически совпадают. Это хорошо видно на рис.3:

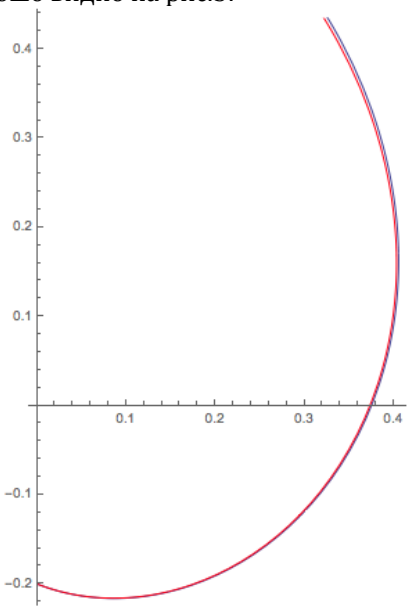


Рисунок 3

2) Метод нейросетевого моделирования. Сети обучались для набора из 300 значений

параметров из области $k \in [0,2]$; $k_1 \in [-1,0]$; $k_2 \in [0,1]$.

Для начала выберем количество нейронов, равное пяти. Модель полученная с использованием такой сети по трём точкам из интервала [0,1] (момент времени 0, 0.5 и 1) отражает основную тенденцию, однако существует значительная погрешность и искажена скорость миграционной динамики, что можно увидеть на рис.4.

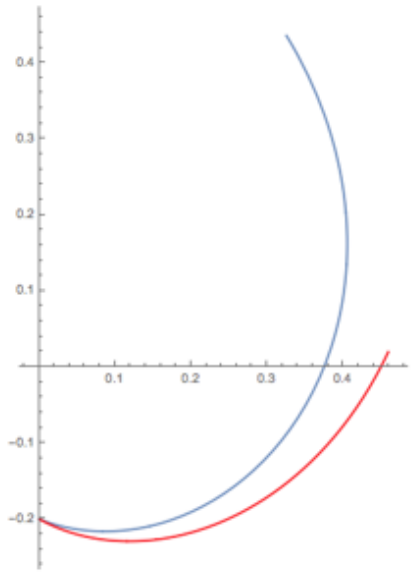


Рисунок 4

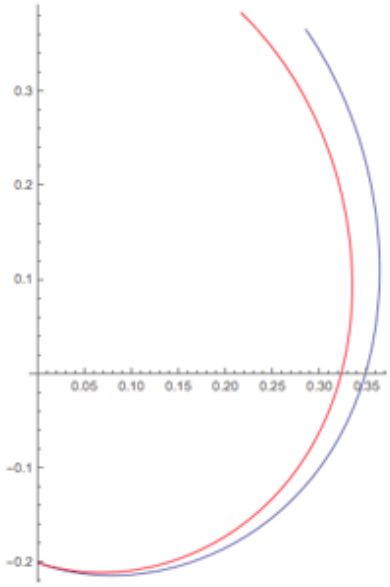


Рисунок 5

Однако, если использовать уравнения (1), то есть нелинейную модель, точность значительно лучше и искажение скорости несущественно (рис.5). Это является преимуществом нейронных сетей: нелинейная модель, которая является более точной, аппроксимируется лучше. При увеличении числа нейронов до 15 достигается хорошая точность для случая нелинейной модели, особенно в окрестности начала координат (рис.6).Также хорошо отражена основная тенденция миграции и её скорость.

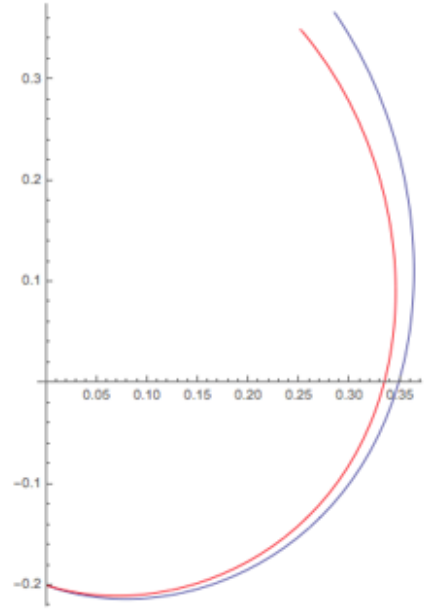


Рисунок 6

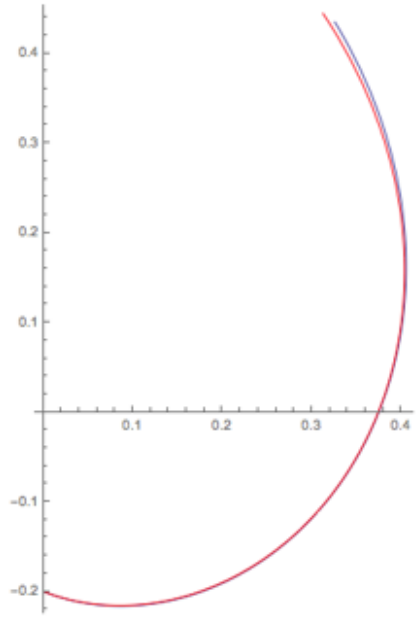


Рисунок 7

Далее, увеличив размер нейронной сети до 25-ти нейронов, при работе с линейной моделью, увидим, что она очень хорошо приближает теоретическую модель, результат сопоставим с результатом метода линейной регрессии при ста точках (рис.7).

Теперь рассмотрим интервал времени, увеличенный в 10 раз. Значения коэффициентов k , k_1, k_2 остались прежними.

1) Второй метод (линейная регрессия).

Начнем рассмотрение со случая 100 точек. Исходя из рис. 8 можно сделать вывод, что погрешность становится очень большой, это говорит о том, что по 100 точкам не получается хорошо прогнозировать тенденцию миграции на достаточно большой промежуток времени (напомним, что рассматривался период прогноза в 5 раз больший, чем период времени, для которого строился прогноз) и, чтобы достичь приемлемого приближения, необходимо увеличение количества точек.

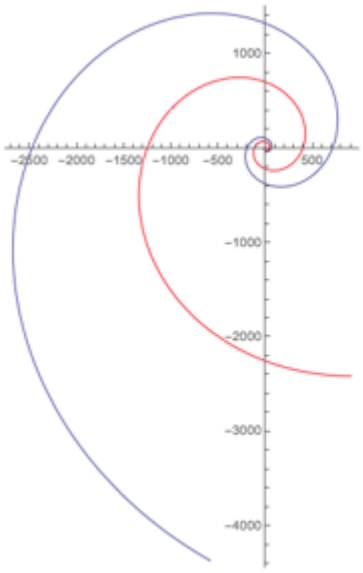


Рисунок 8

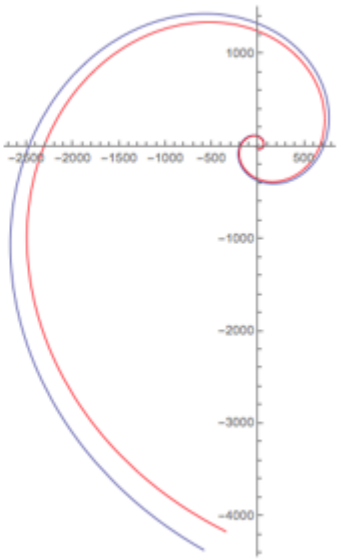


Рисунок 9

При увеличении числа точек до 1000 точность стала достаточно высока, особенно в сравнении с моделью, построенной по 100 точкам (рис.9) . Однако в подобных задачах трудно получить такое большое количество входных данных.

2) Метод нейросетевого моделирования.

Рассмотрим случай нейронной сети, состоящей из 15-ти нейронов, нелинейная модель. Если увеличить период времени в 10 раз, то точность приближения падает, но при этом расчетная модель хорошо повторяет вид графика функции теоретической модели и общая тенденция миграции сохранена (рис.10) .

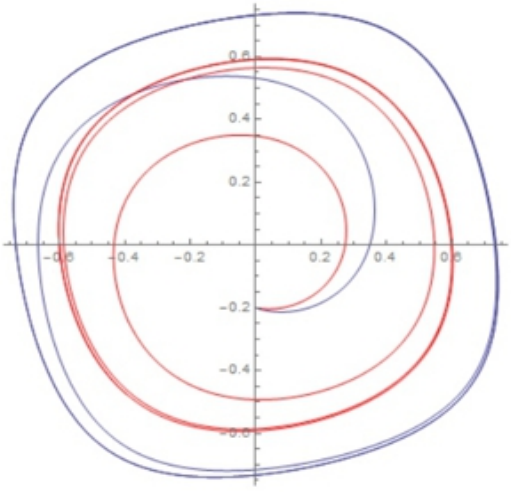


Рисунок 10

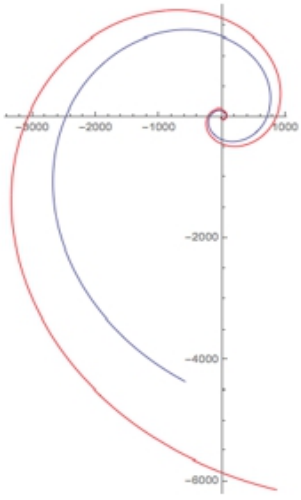


Рисунок 11

Увеличим размер нейронной сети до 25-ти нейронов. Для случая линейной модели результат будет в несколько раз лучше линейной регрессии, построенной по ста точкам, но хуже, чем по тысяче точек, погрешность достаточно велика, однако это не так важно, поскольку линейная модель сама по себе имеет некоторую неточность (рис. 11). При работе с нелинейной моделью, мы достигаем даже при увеличении периода времени в 10 раз, мы получаем отличное приближение, причем на всех участках, что видно на рисунке 12.

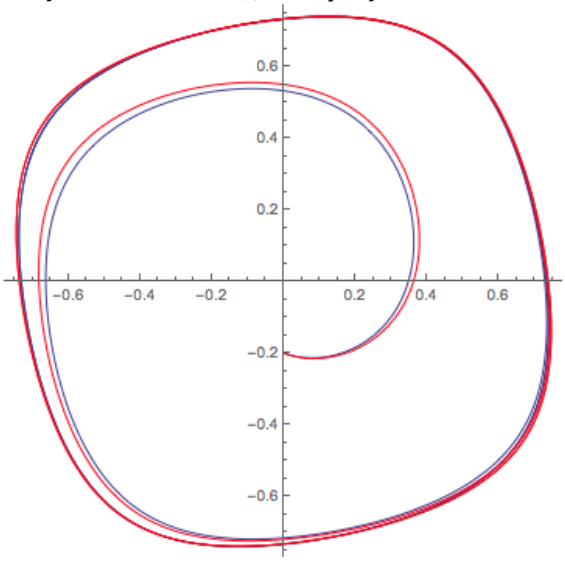


Рисунок 12

ВЫВОДЫ

Исходя из приведенных данных , можно сделать несколько выводов:

- Нейронные сети при достаточном количестве нейронов (в данной задаче - больше или равном 25) позволяют хорошо восстановить модель. Для того чтобы достичь той же точности с использованием линейной регрессии, необходимо большое количество входных данных. Кроме того, такой метод вызывает существенные затруднения для случая нелинейной модели, которая является более точной;
- Нейронные сети могут работать как и линейной моделью, так и с нелинейной, точность аппроксимации существенно зависит от количества нейронов, и не сильно - от количества входных данных, это дает нейронным сетям большое преимущество над методом линейной регрессии .У нейронных сетей имеется ряд заметных преимуществ по сравнению с другими методами приближения и прогнозирования: универсальность, точность, способность к обучению. Нейронные сети уже являются универсальным методом решения многих сложных прикладных задач, полученные результаты лишь подтвердили это.

Проведённые вычислительные эксперименты показывают, что применение нейронных сетей для решения обратной задачи моделирования миграционных потоков является оптимальным способом.

Благодарность: *Статья подготовлена по результатам исследования, выполненного при финансовой поддержке гранта Российского Научного Фонда (проект 14-38-00009) «Программно-целевое управление комплексным развитием Арктической зоны РФ (Санкт-Петербургский политехнический университет Петра Великого).*

Литература

1. Вайдлих В. Социодинамика: системный подход к математическому моделированию в социальных науках. М.: Едиториал УРСС, 2005. 480с.
2. А.Н Васильев, Д.А Тархов – Нейросетевое моделирование. Принципы. Алгоритмы. Приложения. – СПб.:Изд-во Политехн. Ун-та. 2009. - 528 с.
3. Тархов Д.А. Нейросетевые модели и алгоритмы. – М.: Радиотехника, 2014. – 352 с.

Васильев А.Н.[1], Кузнецов Е.Б.[2], Леонов С.С.[3]

[1] Санкт-Петербургский политехнический университет Петра Великого, Санкт-Петербург, д.т.н., профессор кафедры «Высшая математика», a.n.vasilyev@gmail.com

[2] Московский авиационный институт, Москва, д.ф.-м.н., профессор кафедры «Дифференциальные уравнения», kuznetsov@mai.ru

[3] Московский авиационный институт, Москва, аспирант кафедры «Дифференциальные уравнения», powerandglory@yandex.ru

НЕЙРОСЕТЕВОЙ МЕТОД ИДЕНТИФИКАЦИИ И АНАЛИЗА МОДЕЛИ ДЕФОРМИРОВАНИЯ МЕТАЛЛИЧЕСКИХ КОНСТРУКЦИЙ В УСЛОВИЯХ ПОЛЗУЧЕСТИ

КЛЮЧЕВЫЕ СЛОВА

Ползучесть, разрушение, параметр поврежденности, искусственные нейронные сети, нейросетевой базис, продолжение решения по параметру, системы дифференциальных уравнений.

АННОТАЦИЯ

В работе рассматривается задача идентификации параметров модели, описывающей процессы деформирования и разрушения металлических конструкций в условиях ползучести. В качестве определяющих соотношений используются уравнения теории структурных параметров Ю.Н. Работнова с одним скалярным параметром поврежденности, имеющие две особые точки. Для решения указанной задачи применяется метод, в основе которого лежат принципы нейросетевого моделирования. Данный подход позволяет как определить параметры модели, так и получить приближенное решение задачи. Для ускорения процесса идентификации также предложено использовать метод продолжения решения по параметру. В работе найдены параметры модели растяжения образцов из стали 45 в условиях ползучести при постоянных напряжении и температуре. Соответствующие этим параметрам решения хорошо согласуются с экспериментальными данными и результатами других авторов.

1. Введение. В последние десятилетия все более широкое применение находят модели, способные описывать процессы деформирования и разрушения конструкций при сложном напряженном состоянии. Особое внимание уделяется возможности учета ползучести материала в области высоких и умеренных температур. Это связано как с возросшими требованиями к точности прогнозирования длительной прочности конструкций, так и с увеличением количества практических задач, требующих решения, в числе которых исследование поведения конструкций из металлических, керамических и композитных материалов при сложных температурно-силовых режимах. Одним из наиболее эффективных подходов к решению подобных задач является использование уравнений известных теорий ползучести (теории старения, упрочнения, наследственности, теории структурных параметров Ю.Н. Работнова и т. д.). Однако, все такие уравнения, как правило, содержат несколько материальных констант (характеристик ползучести), которые необходимо определять на основании информации о протекании процесса деформирования, основным источником которой является эксперимент. Учитывая то, что характеристики ползучести могут зависеть от вида используемого материала и его состояния, режима нагружения, температуры, типа анизотропии и других факторов, задача их идентификации имеет весьма сложный характер. В данной работе представлен унифицированный метод идентификации параметров модели, описывающей процессы ползучести и разрушения конструкций по результатам эксперимента. В качестве основы для разрабатываемого подхода приняты принципы и методы нейросетевого моделирования.

2. Метод нейронных сетей. Следуя монографии [1], рассмотрим построение нейронной сети для системы m обыкновенных дифференциальных уравнений (ОДУ) r -ого порядка с p

неизвестными скалярными параметрами, задаваемыми вектором $\alpha=(\alpha_1,\ldots,\alpha_p)^T$

$$\mathbf{F}\left(t,\mathbf{y},\mathbf{y}',\ldots,\mathbf{y}^{(r)},\alpha\right)=0,\ t\in\left[t_0,t^*\right] \tag{1}$$

\ с начальными условиями

$$\begin{cases}\mathbf{y}(t_0)=\mathbf{y}_0,\\ \mathbf{y}'(t_0)=\mathbf{y}_1,\\ \quad\vdots\\ \mathbf{y}^{(r-1)}(t_0)=\mathbf{y}_{r-1}.\end{cases} \tag{2}$$

Здесь $\mathbf{F}(t,\mathbf{y},\mathbf{y}',\ldots,\mathbf{y}^{(r)},\alpha)=(f_1(t,\mathbf{y},\mathbf{y}',\ldots,\mathbf{y}^{(r)},\alpha),\ldots,f_m(t,\mathbf{y},\mathbf{y}',\ldots,\mathbf{y}^{(r)},\alpha))^T$ – вектор-функция векторного аргумента; $f_i(t,\mathbf{y},\mathbf{y}',\ldots,\mathbf{y}^{(r)},\alpha),\ i=1,\ldots,m$ – скалярные функции векторного аргумента; $\mathbf{y}(t)=(y_1(t),\ldots,y_m(t))^T$ – искомое решение, вектор-функция скалярного аргумента, в общем случае неявно зависящая от параметров $\alpha_1,\ldots,\alpha_p$; t_0 – начальная точка; t^* – правый конец интервала изменения аргумента; $\mathbf{y}_j(t)=(y_{j1},\ldots,y_{jm})^T,\ j=0,\ldots,r-1$ – векторы значений функции $\mathbf{y}(t)$ и ее $r-1$ первых производных в точке t_0; $\mathbf{y}^{(j)}(t)=\dfrac{d^j\mathbf{y}(t)}{dt^j}$; $\mathbf{y}^{(0)}(t)=\mathbf{y}(t)$.

При описании физических процессов на области изменения параметров $\alpha_1,\ldots,\alpha_p$ могут накладываться ограничения вида

$$\alpha_i\in A_i\subseteq\mathrm{R},\quad i=1,\ldots,p. \tag{3}$$

Пусть для задачи (1)-(2) также выполнены условия теоремы Коши о существовании и теоремы о единственности решения начальной задачи. Кроме того, имеются l экспериментальных наблюдений

$$\mathbf{y}(t_q)=\mathbf{y}_q^e,\quad t_q\in\left[t_0,t^*\right],\quad q=1,\ldots,l, \tag{4}$$

\где $\mathbf{y}_q^e=(y_{q1}^e,\ldots,y_{qm}^e)^T$ – векторы значений $\mathbf{y}(t)$ в точках $t_1,\ldots,t_l$ соответственно.

Решение начальной задачи (1)-(2), удовлетворяющее соотношениям (4), будем искать, используя методы нейросетевого моделирования. Для этого каждую компоненту вектор-функции $\mathbf{y}(t)$ разложим по нейросетевому базису

$$\hat{y}_i(t,\mathbf{w}_i)=\sum_{j=1}^{N_i}c_{ij}v_j(t,\mathbf{a}_{ij}),\quad i=1,\ldots,m, \tag{5}$$

где $\mathbf{w}_i=(\mathbf{w}_{i1},\ldots,\mathbf{w}_{iN_j})^T$ – матрицы настраиваемых параметров (нейросетевых коэффициентов); $\mathbf{w}_{ij}=(c_{ij}\,\mathbf{a}_{ij})$; c_{ij} – линейно входящие параметры; $\mathbf{a}_{ij}=(a_{ij}^1,a_{ij}^2)$ – нелинейно входящие параметры; N_i – количество нейронов в разложении (5).

Тип нейросетевого базиса $c_{ij}v_j(t,\mathbf{a}_{ij})$ задается скалярной функцией скалярного аргумента $\varphi_j(x)$, так называемой функцией активации

$$v_j(t,\mathbf{a}_{ij})=\varphi_j(x),\ x=\psi(t,\mathbf{a}_{ij}),$$

где $\psi(\cdot)$ – заданная функция (например, $\psi(t,\mathbf{a}_{ij})=a_{ij}^1\cdot t+a_{ij}^2$). Активационная функция может быть задана в форме гиперболического тангенса, радиальной базисной функции (например, в виде гауссианы $\varphi(x)=\exp\{-x^2\}$) или иной форме, описанной в монографии [1].

Заменяя в задаче (1)-(2) и экспериментальных данных (4) компоненты вектор-функции $\mathbf{y}(t)$ на нейросетевые разложения (5), составим нормированный функционал ошибки в интегральной форме, как сумму взвешенных невязок

$$J(\alpha,\mathbf{w}_1,\dots\mathbf{w}_m)=R_0^{-1}\cdot\sum_{i=1}^{m}\left(\beta_i\int_{t_0}^{t^*}\left|f_i(\xi,\hat{\mathbf{y}},\hat{\mathbf{y}}',\dots\hat{\mathbf{y}}^{(n)},\alpha)\right|^2 d\xi+\right.$$
$$\left.+\gamma_i\sum_{j=0}^{r-1}\left|\hat{y}_i^{(j)}(t_0,\mathbf{w}_i)-y_{ji}\right|^2+\delta_i\sum_{q=1}^{l}\left|\hat{y}_i(t_q,\mathbf{w}_i)-y_{qi}^e\right|^2\right). \quad (6)$$

Здесь $\hat{\mathbf{y}}(t,\mathbf{w_1},\dots,\mathbf{w_m})=(\hat{y}_1(t,\mathbf{w_1}),\dots,(t,\mathbf{w_m}))^T$ – вектор нейросетевых разложений; $\beta_i,\gamma_i,\delta_i>0$ – штрафные множители; $R_0=\sum_{i=1}^{m}(\beta_i\cdot(t^*-t_0)+\gamma_i\cdot r+\delta_i\cdot l)$.

На практике обычно используется дискретное представление [1] функционала ошибки (6)

$$J(\alpha,\mathbf{w}_1,\dots\mathbf{w}_m)=R_1^{-1}\cdot\sum_{i=1}^{m}\left(\beta_i\sum_{h=1}^{M}\left|f_i(\xi_h,\hat{\mathbf{y}},\hat{\mathbf{y}}',\dots\hat{\mathbf{y}}^{(n)},\alpha)\right|^2+\right.$$
$$\left.+\gamma_i\sum_{j=0}^{r-1}\left|\hat{y}_i^{(j)}(t_0,\mathbf{w}_i)-y_{ji}\right|^2+\delta_i\sum_{q=1}^{l}\left|\hat{y}_i(t_q,\mathbf{w}_i)-y_{qi}^e\right|^2\right). \quad (7)$$

В данной формуле вектор-функция $\mathbf{F}(t,\hat{\mathbf{y}},\hat{\mathbf{y}}',\dots,\hat{\mathbf{y}}^{(n)},\alpha)$ вычисляется на множестве пробных точек $\{\xi_h\}_{h=1}^{M}$, генерируемых случайным образом по равномерному закону распределения на отрезке $[t_0,t^*]$ [1]; M – количество пробных точек; $R_1=\sum_{i=1}^{m}(\beta_i\cdot M+\gamma_i\cdot r+\delta_i\cdot l)$.

Для нахождения неизвестных параметров $\alpha_1,\dots,\alpha_p$ и нейросетевых коэффициентов $\mathbf{w_1},\dots,\mathbf{w_m}$ решается задача минимизации функционала ошибки (6) (или (7)) с учетом ограничений (3)

$$J(\alpha,\mathbf{w}_1,\dots,\mathbf{w}_m)\to\min. \quad (8)$$

В результате решения задачи (8) получим значения параметров $\alpha_1,\dots,\alpha_p$ и вектор настраиваемых параметров $\mathbf{w}_1^*,\dots,\mathbf{w}_m^*$, которые доставляют минимум функционалу (6) (или (7)). Нейросетевое решение задачи запишется в виде

$$\hat{\mathbf{y}}(t)=\hat{\mathbf{y}}\left(t,\mathbf{w}_1^*,\dots,\mathbf{w}_m^*\right),t\in\left[t_0,t^*\right]. \quad (9)$$

Отметим, что минимизация функционала ошибки ведется не до глобального минимума, а до момента, когда его значение становится меньше наперед заданного значения точности η, т.е. $J<\eta$. И именно это значение функционала принимается за приближенное минимальное J_*. Для того чтобы избежать остановки процесса минимизации в точке локального минимума, производится периодическая (после нескольких итераций алгоритма минимизации) перегенерация пробных точек $\{\xi_h\}_{h=1}^{M}$ [1].

Для иллюстрации описанного подхода рассмотрим задачу идентификации параметров модели одноосного растяжения прутков (диаметр *d=42* мм) из стали 45, широко используемой в авиации (например, для создания деталей трубопроводной арматуры после закалки и отпуска), при постоянных напряжении σ_0 и температуре T.

3. Растяжение образцов из стали 45. Для описания поведения металлов в условиях ползучести вплоть до разрушения будем использовать уравнения теории структурных параметров Ю.Н. Работнова [2] с одним скалярным параметром поврежденности ω в форме системы двух ОДУ первого порядка [3, 4]:

$$\begin{cases} \dfrac{d\varepsilon}{dt} = f(\sigma,T)\Psi(\omega,T), \\ \dfrac{d\omega}{dt} = g(\sigma,T)\Psi(\omega,T). \end{cases} \tag{10}$$

Где ε – деформация ползучести; σ – действующее напряжение; t – время; T – температура; функциональные зависимости, входящие в правые части, определяются по результатам эксперимента.

В статье [3] показано, что функция $\Psi(\omega,T)$ может быть выбрана в виде

$$\Psi(\omega,T) = \omega^{-\alpha}\left(1-\omega^{\alpha+1}\right)^{-m}, \tag{11}$$

где α и m – параметры модели, зависящие в общем случае от температуры T.

Таким образом, подставляя соотношение (11) в систему ОДУ (10), в случае постоянной температуры получим:

$$\begin{cases} \dfrac{d\varepsilon}{dt} = \dfrac{f(\sigma)}{\omega^{\alpha}\left(1-\omega^{\alpha+1}\right)^{m}}, \\ \dfrac{d\omega}{dt} = \dfrac{g(\sigma)}{\omega^{\alpha}\left(1-\omega^{\alpha+1}\right)^{m}}. \end{cases} \tag{12}$$

Функции $f(\sigma)$ и $g(\sigma)$ для стали 45 выберем в виде [3]

$$f(\sigma) = B_{\varepsilon}\sigma^{n},\ g(\sigma) = B_{\omega}\sigma^{k}.$$

где $B_{\varepsilon}, B_{\omega}, n, k$ – характеристики ползучести материала.

При постоянном напряжении $\sigma=\sigma_0=const$ получим следующую систему ОДУ для данной задачи

$$\begin{cases} \dfrac{d\varepsilon}{dt} = \dfrac{B_{\varepsilon}\sigma_0^{n}}{\omega^{\alpha}\left(1-\omega^{\alpha+1}\right)^{m}}, \\ \dfrac{d\omega}{dt} = \dfrac{B_{\omega}\sigma_0^{k}}{\omega^{\alpha}\left(1-\omega^{\alpha+1}\right)^{m}}. \end{cases} \tag{13}$$

В качестве начальных условий для системы (13) берутся однородные

$$t = 0 : \omega = \varepsilon = 0. \tag{14}$$

Аналитическое решение задачи (13)-(14) можно записать в виде [5]

$$\omega(t) = \left(1-\left(1-(\alpha+1)(m+1)B_{\omega}\sigma_0^{k}t\right)^{\frac{1}{m+1}}\right)^{\frac{1}{\alpha+1}}, \quad \varepsilon(t) = \frac{B_{\varepsilon}\sigma_0^{n}}{B_{\omega}\sigma_0^{k}}\cdot\omega(t). \tag{15}$$

Учитывая, что при разрушении параметр поврежденности принимает значение равное единице, из первого соотношения (15) получим значение длительной прочности t^{*} данной конструкции

$$t^{*} = \left[(m+1)(\alpha+1)B_{\omega}\sigma_0^{k}\right]^{-1}.$$

4. Идентификация параметров. Введем обозначения: $\mathbf{y}(t)=(\varepsilon(t),\omega(t))^{T}$;

$$f_1(\mathbf{y},\mathbf{y}',B_{\varepsilon},\alpha,m,n) = \frac{d\varepsilon}{dt} - \frac{B_{\varepsilon}\sigma_0^{n}}{\omega^{\alpha}\left(1-\omega^{\alpha+1}\right)^{m}};\quad f_2(\mathbf{y},\mathbf{y}',B_{\omega},\alpha,m,k) = \frac{d\omega}{dt} - \frac{B_{\omega}\sigma_0^{k}}{\omega^{\alpha}\left(1-\omega^{\alpha+1}\right)^{m}}.$$

Видно, что система ОДУ (13) с начальными условиями (14) по структуре схожа с задачей (1)-(2) ($m=2,\ r=1,\ p=6$), поэтому будем проводить идентификацию параметров системы (13), используя подход, описанный в разделе 2. В качестве дополнительных данных будем использовать

результаты эксперимента на одноосное растяжение цилиндрических образцов из стали 45 при некотором уровне напряжения σ_0 и температуре $T=850^oC$ [3].

$$\varepsilon\left(t_q\right)=\varepsilon_q,\ \omega\left(t_q\right)=\omega_q,\quad t_q\in\left(0,t^*\right],\quad q=1,\ldots,l, \tag{16}$$

где t_q – момент времени снятия q-ого экспериментального значения, ε_q и ω_q – экспериментальное значение деформации ползучести и параметра поврежденности в момент времени t_q соответственно.

Используя в нейросетевом разложении (5) базисные функции специального вида, мы получим

$$\begin{aligned}\hat{\varepsilon}(t,\mathbf{c},\mathbf{a})&=\sum_{j_1=1}^{N_1}c_{j_1}\cdot\mathrm{th}\left(a_{1j_1}t+a_{2j_1}\right)\cdot\left(1+a_{3j_1}t+a_{4j_1}\right)^{-1},\\ \hat{\omega}(t,\mathbf{b},\mathbf{d})&=\sum_{j_2=1}^{N_2}b_{j_2}\cdot\mathrm{th}\left(d_{1j_2}t+d_{2j_2}\right)\cdot\left(1+d_{3j_2}t+d_{4j_2}\right)^{-1},\end{aligned} \tag{17}$$

Здесь $\mathbf{c}=\{c_{j_1}\}$, $\mathbf{b}=\{b_{j_2}\}$ – векторы линейно входящих параметров; $\mathbf{a}=\{a_{ij_1}\}$, $\mathbf{d}=\{d_{ij_2}\}$ – матрицы нелинейно входящих параметров; $i=1,\ldots,4$.

Для задачи (13)-(14) функционал ошибки (7) запишется в виде

$$J(B_\varepsilon,B_\omega,n,k,\alpha,m,\mathbf{c},\mathbf{a},\mathbf{b},\mathbf{d})=R\cdot\left\{\delta\sum_{h=1}^{M}\left[\left|\frac{d\hat{\varepsilon}}{dt}-\frac{B_\varepsilon\sigma_0^n}{P(\hat{\omega})}\right|^2+\left|\frac{d\hat{\omega}}{dt}-\frac{B_\omega\sigma_0^k}{P(\hat{\omega})}\right|^2\right]\right. \tag{18}$$

$$\left.+\delta_1\left|\hat{\varepsilon}(0,\mathbf{c},\mathbf{a})\right|^2+\delta_2\left|\hat{\omega}(0,\mathbf{b},\mathbf{d})\right|^2+\delta_3\sum_{q=1}^{l}\left|\hat{\varepsilon}(t_q,\mathbf{c},\mathbf{a})-\varepsilon_q\right|^2+\delta_4\sum_{q=1}^{l}\left|\hat{\omega}(t_q,\mathbf{b},\mathbf{d})-\right.\right.$$

Здесь $\{\xi_h\}_{h=1}^{M}$ – набор пробных точек, равномерно распределенных на отрезке $[0,t^*]$; выражение под знаком первой суммы берется в точке $t=\xi_h$; $P(\hat{\omega})=\hat{\omega}^{\alpha}(1-\hat{\omega}^{\alpha+1})^m$; $R=(2\cdot\delta\cdot M+\delta_1+\delta_2+(\delta_3+\delta_4)\cdot l)^{-1}$.

Для идентификации параметров $B_\varepsilon, B_\omega, n, k, \alpha, m$ решается задача

$$J(B_\varepsilon,B_\omega,n,k,\alpha,m,\mathbf{c},\mathbf{a},\mathbf{b},\mathbf{d})\to\min. \tag{19}$$

Учитывая особенности протекания процесса деформирования, на параметры модели необходимо наложить следующие ограничения

$$B_\varepsilon>0,\ B_\omega>0,\ n>0,\ k>0,\ \alpha>0,\ m>0. \tag{20}$$

Таким образом, имеем задачу минимизации с ограничениями типа неравенств (19)-(20), в результате решения которой найдем значения параметров $B_\varepsilon, B_\omega, n, k, \alpha, m$ и коэффициенты нейросетевого разложения $\mathbf{c}^*, \mathbf{a}^*, \mathbf{b}^*, \mathbf{d}^*$, доставляющие минимум функционалу (18). Подставляя нейросетевые коэффициенты в (17), получим приближенное решение задачи.

5. Параметры нейросети и результаты эксперимента. Идентификация параметров системы ОДУ (13) проводится для трех значений напряжений $\sigma_0=35;40;45$ МПа. Число базисных функций в нейросетевом разложении, количество пробных и экспериментальных точек, значения штрафных множителей, а также другие параметры нейросети приведены в табл. 1.

Таблица 1. Параметры нейросети

σ_0, МПа	δ	δ_1	δ_2	δ_3	δ_4	N_1	N_2	M	l	η
35	10	10	10	10	10	3	3	30	14	$5\cdot10^{-4}$
40	1	1	1	1	1	4	3	20	10	$5\cdot10^{-4}$

σ_0, МПа	δ	δ_1	δ_2	δ_3	δ_4	N_1	N_2	M	l	η
45	10^4	10^5	10^5	10^5	10^5	4	3	20	12	$5\cdot10^{-4}$

Экспериментальные данные для деформации ползучести взяты из работы [3], для параметра поврежденности результаты эксперимента пересчитаны по формуле $\omega=\varepsilon/\varepsilon_*$, где ε_* – значение деформации ползучести в момент разрушения.

6. Результаты вычислений. Задача минимизации (19)-(20) решалась в вычислительной среде Mathcad 14 методом сопряженных градиентов [6]. Параметры, входящие в систему ОДУ (13), полученные в результате решения задачи (19)-(20) и приведенные в работе [3], представлены в табл. 2. В табл. 3 даны основные сведения о процессе деформирования, где $\delta t^*, \delta\varepsilon_*$ – относительная погрешность длительной прочности и деформации ползучести в момент разрушения соответственно; t_n – время счета.

Таблица 2. Характеристики ползучести

Результаты	σ_0, МПа	$B_\varepsilon\sigma_0^n$, $ч^{-1}$	$B_\omega\sigma_0^k$, $ч^{-1}$	α	m
Статья [3]	35	0.0115	0.0202	0.849	2.83
	40	0.034	0.0512		
	45	0.0883	0.1163		
Задача (19)-(20)	35	0.0079	0.0157	0.947	3.873
	40	0.0383	0.0619	0.806	1.966
	45	0.1603	0.2552	0.297	1.479

Таблица 3. Основные сведения о процессе деформирования

	Результаты	t^*, ч	ε_*	δt^*, %	$\delta\varepsilon_*$, %	$J_*\cdot10^5$	t_n, с
35 МПа	Параметры [3]	7.0005	0.5716	4.39	10.77	–	–
	Задача (19)-(20)	6.7348	0.5072	0.43	1.73	4.691	934
	Эксперимент [3]	6.7061	0.5161	–	–	–	–
40 МПа	Параметры [3]	2.7601	0.6647	7.37	7.98	–	–
	Задача (19)-(20)	3.0158	0.6185	1.21	0.47	12.517	762
	Эксперимент [3]	2.9796	0.6155	–	–	–	–
45 МПа	Параметры [3]	1.2145	0.7593	0.82	19.9	–	–
	Задача (19)-(20)	1.2187	0.6281	0.47	0.82	17.16	973
	Эксперимент [3]	1.2245	0.6333	–	–	–	–

Ниже приводятся приближенные нейросетевые решения для деформации ползучести

$$\hat{\varepsilon}_{35}(t)=0.112\cdot\frac{\mathrm{th}(0.496t+0.047)}{(0.93-0.088t)}+7.022\cdot10^{-3}\cdot\frac{\mathrm{th}(0.019t+0.167)}{(1.629-0.241t)}$$
$$+0.022\cdot\frac{\mathrm{th}(9.197t+1.65\cdot10^{-3})}{(1.091-0.017t)};$$
$$\hat{\varepsilon}_{40}(t)=2.356\cdot10^{-3}\cdot\frac{\mathrm{th}(2.804t+0.289)}{(1.957-0.652t)}+0.172\cdot\frac{\mathrm{th}(1.034t+6.096\cdot10^{-3})}{(0.95+0.754t)}-$$
$$-346.182\cdot\frac{\mathrm{th}(-0.103t+2.26\cdot10^{-4})}{(351.428-33.064t)}-2.297\cdot\frac{\mathrm{th}[(4.289t+2.292)\cdot10^{-3}]}{(1.922+5.239\cdot10^{-3}t)};\qquad(21)$$
$$\hat{\varepsilon}_{45}(t)=5.789\cdot\frac{\mathrm{th}(-0.814t-0.137)}{(1.034-0.26t)}+0.065\cdot\frac{\mathrm{th}(1.826\cdot10^{-3}t+0.026)}{(1.221+0.127t)}+$$
$$+0.669\cdot\frac{\mathrm{th}(0.012t-2.744\cdot10^{-3})}{(0.937-0.734t)}+24.711\cdot\frac{\mathrm{th}(0.183t+0.027)}{(0.868-8.041\cdot10^{-4}t)}$$

и параметра поврежденности

$$\hat{\omega}_{35}(t) = -0.241 \cdot \frac{\text{th}(-0.426t - 0.034)}{(1.1 - 0.106t)} + 6.1 \cdot \frac{\text{th}(0.013t - 1.256 \cdot 10^{-5})}{(0.131 + 1.109t)} +$$
$$+0.023 \cdot \frac{\text{th}(0.032t + 4.209 \cdot 10^{-3})}{(1.959 - 0.29t)};$$

$$\hat{\omega}_{40}(t) = 0.338 \cdot \frac{\text{th}(0.14t + 2.207 \cdot 10^{-3})}{(0.626 - 0.107t)} - 0.076 \cdot \frac{\text{th}[(-443.59t + 9.983) \cdot 10^{-5}]}{[(7.328 + 8.929t) \cdot 10^{-4}]} + \quad (22)$$
$$+1.24 \cdot 10^{-3} \cdot \frac{\text{th}(0.224t - 0.059)}{(0.417 - 0.139t)};$$

$$\hat{\omega}_{45}(t) = 6.316 \cdot \frac{\text{th}(0.108t + 5.949 \cdot 10^{-4})}{(0.998 + 7.411t)} + 0.032 \cdot \frac{\text{th}(0.245t + 9.74 \cdot 10^{-4})}{(0.992 - 0.783t)} +$$
$$+1.384 \cdot \frac{\text{th}(0.253t - 2.433 \cdot 10^{-5})}{(1.001 - 0.28t)}.$$

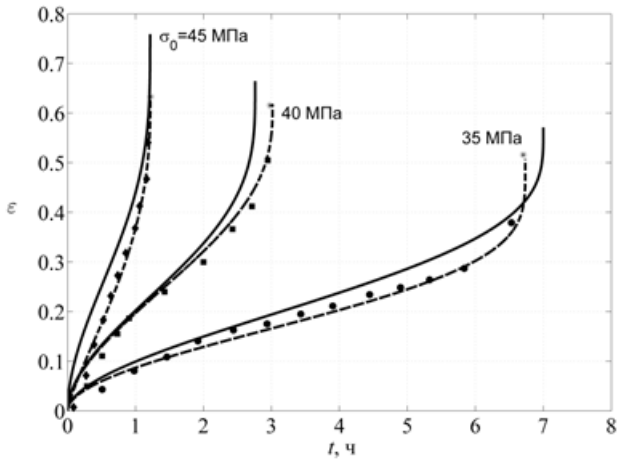


Рис. 1. Деформация ползучести

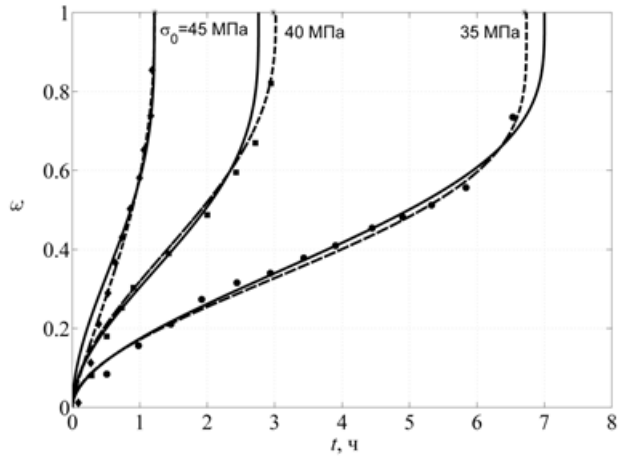


Рис. 2. Параметр поврежденности

Графики зависимостей деформации ползучести и параметра поврежденности от времени показаны на рис. 1-2, где точками, квадратами и ромбами обозначены результаты эксперимента для значений напряжения $\sigma_0 = 35; 40; 45$ МПа соответственно, непрерывные и пунктирные линии – аналитические зависимости, соответствующие параметрам модели, полученным в работе [3] и при решении задачи (19)-(20) соответственно. Значения напряжения указаны на рисунках около соответствующих графиков.

На рис. 3-4 представлены нейросетевые решения (21)-(22), изображаемые пунктирными линиями, остальные обозначения сохраняются.

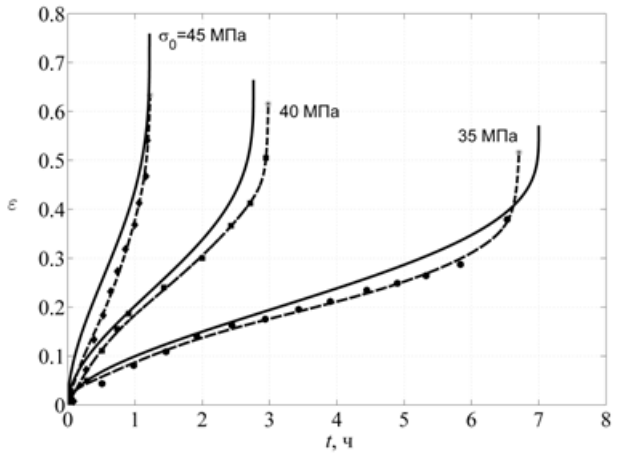


Рис. 3. Деформация ползучести, нейросетевые решения

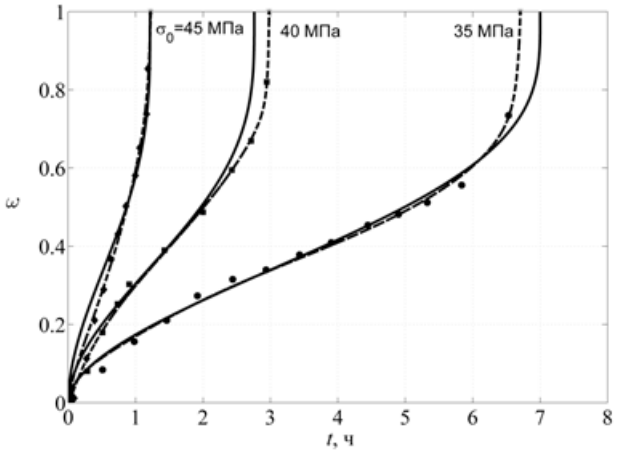


Рис. 4. Параметр поврежденности, нейросетевые решения

7. Метод продолжения решения по параметру. Анализируя начальную задачу (13)-(14) можно видеть, что она содержит две особые точки $\omega=0$ и $\omega=1$ в которых правые части уравнений системы (13) обращаются в бесконечность. Метод нейронных сетей позволяет обойти данную особенность и получить параметры модели, а также приближенное решение данной задачи. Однако особенность остается, и можно ожидать, что если нам удастся устранить ее, то это позволит сократить временные затраты на минимизацию функционала ошибки и позволит использовать более простые формы активационной функции в (17). Для этого предварительно преобразуем исход-ную задачу (13)-(14), используя метод продолжения решения по параметру [7].

Параметризуем уравнения системы (13), полагая, что деформация ползучести, параметр поврежденности и время являются функциями параметра κ

$$\varepsilon = \varepsilon(\kappa),\ \omega = \omega(\kappa),\ t = t(\kappa). \tag{23}$$

Параметр κ можно выбирать различными способами (например, как *наилучший параметр* [7]). Для задачи (13)-(14) выберем его в форме

$$(d\kappa)^2 = (d\varepsilon)^2 + (d\omega)^2 + \left(\omega^{-\alpha}\left(1-\omega^{\alpha+1}\right)^{-m} dt\right)^2. \tag{24}$$

Тогда параметризованная система примет вид

$$\begin{cases} \dfrac{d\varepsilon}{d\kappa} = \dfrac{B_\varepsilon \sigma_0^n}{\sqrt{1+B_\varepsilon^2\sigma_0^{2n}+B_\omega^2\sigma_0^{2k}}}, \\ \dfrac{d\omega}{d\kappa} = \dfrac{B_\omega \sigma_0^k}{\sqrt{1+B_\varepsilon^2\sigma_0^{2n}+B_\omega^2\sigma_0^{2k}}}, \\ \dfrac{dt}{d\kappa} = \dfrac{\omega^\alpha\left(1-\omega^{\alpha+1}\right)^m}{\sqrt{1+B_\varepsilon^2\sigma_0^{2n}+B_\omega^2\sigma_0^{2k}}}. \end{cases} \tag{25}$$

Начальные условия (14) для системы (25) перепишутся в виде

$$\kappa = 0 : \varepsilon = \omega = t = 0. \tag{26}$$

Отметим следующие преимущества использования параметра κ:

1. Обусловленность полученной системы (25) близка к наилучшей;
2. Система (25) имеет более простой вид по сравнению с (13), так как правые части первых двух уравнений и знаменатель третьего уравнения системы (25) являются постоянными величинами;
3. Можно получить значения параметра в экспериментальных точках (16) κ_q и в момент разрушения κ^* по приведенным формулам

$$\kappa_q = \varepsilon_q \cdot \left(B_\varepsilon \sigma_0^n\right)^{-1} \cdot \sqrt{1+B_\varepsilon^2\sigma_0^{2n}+B_\omega^2\sigma_0^{2k}}, \quad \kappa^* = \left(B_\omega \sigma_0^k\right)^{-1} \cdot \sqrt{1+B_\varepsilon^2\sigma_0^{2n}+B_\omega^2} \tag{27}$$

Для определения параметров системы (25) также будем использовать метод нейронных сетей. Выбирая активационную функцию в форме гиперболи-ческого тангенса, запишем нейросетевое разложение (5) в виде

$$\hat{\varepsilon}(\kappa,\mathbf{c},\mathbf{a}) = \sum_{j=1}^{N_1} c_j \cdot \operatorname{th}\left(a_{0j}\kappa + a_{1j}\right), \quad \hat{\omega}(\kappa,\mathbf{b},\mathbf{d}) = \sum_{j=1}^{N_2} b_j \cdot \operatorname{th}\left(d_{0j}\kappa + d_{1j}\right),$$
$$\hat{t}(\kappa,\mathbf{r},\mathbf{s}) = \sum_{j=1}^{N_3} r_j \cdot \operatorname{th}\left(s_{0j}\kappa + s_{1j}\right). \tag{28}$$

Для задачи (25)-(26) функционал ошибки (7) запишется в виде

$$J_p(B_\varepsilon, B_\omega, n, k, \alpha, m, \mathbf{c}, \mathbf{a}, \mathbf{b}, \mathbf{d}, \mathbf{r}, \mathbf{s}) = R' \cdot \left\{ \delta \sum_{h=1}^{M} \left[\left| \frac{d\hat{\varepsilon}}{d\kappa} - \frac{B_\varepsilon \sigma_0^n}{Q} \right|^2 + \left| \frac{d\hat{\omega}}{d\kappa} - \frac{B_\omega \sigma_0^k}{Q} \right|^2 + \right.\right.$$
$$\left.\left. + \left| \frac{d\hat{t}}{d\kappa} - \frac{P(\hat{\omega})}{Q} \right|^2 \right]\Bigg|_{\kappa=\xi_h} + \delta_1 \left|\hat{\varepsilon}(0,\mathbf{c},\mathbf{a})\right|^2 + \delta_2 \left|\hat{\omega}(0,\mathbf{b},\mathbf{d})\right|^2 + \delta_3 \left|\hat{t}(0,\mathbf{r},\mathbf{s})\right|^2 + \tag{29}$$
$$+ \delta_4 \sum_{q=1}^{l} \left|\hat{\varepsilon}(\kappa_q,\mathbf{c},\mathbf{a}) - \varepsilon_q\right|^2 + \delta_5 \sum_{q=1}^{l} \left|\hat{\omega}(\kappa_q,\mathbf{b},\mathbf{d}) - \omega_q\right|^2 + \delta_6 \sum_{q=1}^{l} \left|\hat{t}(\kappa_q,\mathbf{r},\mathbf{s}) - t_q\right|^2 \Bigg\}.$$

Здесь $R' = \left(3 \cdot \delta \cdot M + \sum_{i=1}^{3} \delta_i + \sum_{i=4}^{6} \delta_i\right)^{-1}$; $Q = \sqrt{1+B_\varepsilon^2\sigma_0^{2n}+B_\omega^2\sigma_0^{2k}}$; $\{\xi_h\}_{h=1}^M$ – набор пробных точек, равномерно распределенных на отрезке $[0, \kappa^*]$.

Для идентификации параметров $B_\varepsilon, B_\omega, n, k, \alpha, m$ решается задача

$$J_p(B_\varepsilon, B_\omega, n, k, \alpha, m, \mathbf{c}, \mathbf{a}, \mathbf{b}, \mathbf{d}, \mathbf{r}, \mathbf{s}) \to \min. \tag{30}$$

с ограничениями типа неравенств (20).

Задача минимизации (30), (20) решалась для $\sigma_0 = 35$ МПа. Число базисных функций в нейросетевом разложении, количество пробных и экспериментальных точек, значения штрафных

множителей, а также другие параметры нейросети для параметризованной задачи приведены в табл. 4. Параметры, входящие в систему ОДУ (25), полученные в результате решения задачи (30), (20) и приведенные в работе [3], представлены в табл. 5. В табл. 6 даны основные сведения о процессе деформирования.

Таблица 4. Параметры нейросети

σ_0, МПа	δ	δ_i, $i=1,\ldots,6$	N_1	N_2	N_3	M		η
35	10	100	1	1	2	30	14	$5\cdot10^4$

Таблица 5. Характеристики ползучести

Результаты	$B_\varepsilon \sigma_0^n$, $ч^{-1}$	$B_\omega \sigma_0^k$, $ч^{-1}$	α	m
Статья [3]	0.0115	0.0202	0.849	2.83
Задача (30), (20)	0.0108	0.0209	0.811	2.903

Таблица 6. Основные сведения о процессе деформирования

Результаты	t^*, ч	ε_*	δt^*, %	$\delta\varepsilon_*$, %	$J_*\cdot10^5$	t_n, с
Статья [3]	7.00049	0.5716	4.2	9.72	–	–
Задача (30), (20)	6.78742	0.5185	1.21	0.46	7.061	385
Эксперимент [3]	6.70612	0.5161	–	–	–	–

Приближенное нейросетевое решение задачи (25)-(26) имеет вид

$$\hat{\varepsilon}(\kappa) = 1.227\cdot \mathrm{th}(9.157\cdot10^{-3}\kappa + 4.833\cdot10^{-3});$$
$$\hat{\omega}(\kappa) = 2.327\cdot \mathrm{th}(9.494\cdot10^{-3}\kappa + 2.259\cdot10^{-3}); \qquad (31)$$
$$\hat{t}(\kappa) = 3.656\cdot \mathrm{th}(0.072\kappa + 1.065) + 3.069\cdot \mathrm{th}(0.099\kappa - 1.722).$$

Графики для задачи (25)-(26) аналогичны представленным на рис. 1-4.

8. Выводы. В качестве основных результатов работы можно выделить:

1. Применение метода нейронных сетей к решению задачи Коши для систем ОДУ r-ого порядка с неизвестными скалярными параметра-ми и набором дополнительных данных. Описан процесс построения нейронной сети, а также алгоритм решения;
2. Использование указанного подхода для решения задачи идентифи-кации параметров модели ползучести и разрушения образцов из стали 45 при постоянных напряжении и температуре – задачи, описываемой системой двух ОДУ первого порядка теории структурных парамет-ров Ю.Н. Работнова;
3. Нахождение параметров и нейросетевых решений для рассматри-ваемой модели в случае трех начальных напряжений $\sigma_0 = 35,40,45$ МПа. Результаты расчетов показывают хорошее качественное и количественное согласование с экспериментальными данными и результатами, представленными в работе [3] (табл. 2-3, рис. 1-4), что говорит о применимости нейросетевого подхода к решению задач данного класса;
4. Применение метода продолжения решения по параметру к рассмат-риваемой задаче привело к значительному уменьшению времени счета (табл. 6), что достигается за счет возможности использования более простых форм активационных функций. При этом точность полученного решения не уменьшается (табл. 5-6).

Работа выполнена при финансовой поддержке Российского Фонда Фундаментальных Исследований, проекты 13-08-00473, 14-01-00660 и 14-01-00733.

Литература

1. Васильев А.Н., Тархов Д.А. Нейросетевое моделирование. Принципы. Алгоритмы. Приложения. – СПб.: Изд-во СПбГПУ, 2009. – 528 с.
2. Работнов Ю.Н. Ползучесть элементов конструкций. – М.: Наука, 1966. – 752 с.
3. Горев Б.В., Захарова Т.Э., Клопотов И.Д. К описанию процесса ползучести и разрушения материалов с немонотонным изменением деформационно-прочностных свойств // Физическая мезомеханика. – 2002. – Т. 5. – № 2. – С. 17–22.
4. Gorev B.V., Klopotov I.D., Lyubashevskaya I.V. Creep and damage behavior of AK4-1T and VT-9 alloys under different stress states // Theor. and Appl. Fract. Mech. 1998. Vol. 29. P. 1–10.
5. Горев Б.В., И.В. Любашевская, Панамарев В.А., Иявойнен С.В. Описание процесса ползучести и разрушения

современных конструкционных материалов с использованием кинетических уравнений в энергетической форме // ПМТФ. – 2014. – Т. 55. – № 6. – С. 132–144.
6. Полак Э. Численные методы оптимизации. Единый подход. – М.: Мир, 1974. – 376 с.
7. Шалашилин В.И., Кузнецов Е.Б. Метод продолжения решения и наилучшая параметризация. – М.: Эдиториал УРСС, 1999. – 224 с.

Бурцев А.А.[1], Сидоров С.А.[2]

[1] Федеральный Научный Центр НИИ Системных Исследований (ФГУ ФНЦ НИИСИ РАН), г. Москва, к.ф-м.н., доцент, старший научный сотрудник Отдела Главного Конструктора (ОГК) изделий на базе модулей сигнальной обработки, burtsev@niisi.msk.ru

[2] Федеральный Научный Центр НИИ Системных Исследований (ФГУ ФНЦ НИИСИ РАН), г. Москва, к.ф-м.н., заведующий Отделом Сопровождения Разработок (ОСР), sidorov@niisi.msk.ru

ПРОГРАММНЫЙ КОМПЛЕКС ДССП-ТВМ ДЛЯ СТРУКТУРИРОВАННОГО ПРОГРАММИРОВАНИЯ ТРОИЧНОЙ [ВИРТУАЛЬНОЙ] МАШИНЫ

КЛЮЧЕВЫЕ СЛОВА

Троичная симметричная система счисления, троичный компьютер, ТВМ – троичная виртуальная машина, структурированное программирование, ДССП – Диалоговая система структурированного программирования.

АННОТАЦИЯ

В статье подчёркивается значимость троичной информатики, объясняются преимущества троичных машин (перед двоичными), и в качестве одного из возможных путей дальнейшего развития отечественной компьютерной индустрии рассматривается предпринятый (в 2010-2013 гг.) в научно-исследовательской лаборатории троичной информатики (НИЛ ТИ) ВМК МГУ вариант построения троичного компьютера в виде имитационной программной модели (ТВМ – троичной виртуальной машины), сопровождаем ой комплексом программных инструментов (ДССП-ТВМ), предназначенных для разработки структурированных программ для троичной машины на языке ДССП-Т – троичном варианте языка ДССП – Диалоговой системы структурированного программирования, созданной в начале 80-х годов XX века в проблемной лаборатории ЭВМ ВМК МГУ под руководством Николая Петровича Брусенцова (07.04.1925–04.12.2014) – творца первых в мире троичных компьютеров «Сетунь» и «Сетунь-70».

Рассматривается состав и структура комплекса, даётся характеристика системы команд троичной виртуальной машины, её языка ассемблера, а также языка ДССП-Т.

Описываются основные возможности кросс-компилятора и наполняющей ДССП-словарь библиотеки модулей (на языке ДССП-Т), обеспечивающие построение троичных программ с последующим их исполнением не только на самой «голой» ТВМ (автономно), но и в среде диалогового командного монитора (ДКМОН) или диалогового ДССП-интерпретатора, способных функционировать на троичной машине в качестве её резидентного ПО (программного оснащения).

ДССП-ТВМ уже сейчас может использоваться в качестве учебной среды программирования, служить своеобразным «полигоном» для освоения «искусства программирования» троичных машин, и тем самым способствовать подготовке квалифицированных специалистов в области троичной информатики.

Статья сопровождается двумя приложениями, в одном из которых демонстрируются примеры программ на языке ДССП-Т, а в другом – вариант учебного плана (или учебной программы) спецкурса с условным названием «Программирование в среде ДССП для троичной машины», который предлагался (и предлагается!) для изучения студентам старших курсов (университетов или технических вузов), освоившим базовый курс программирования.

В настоящее время, когда традиционная (кремниевая) микроэлектроника, построенная на двоичной системе счисления, подходит к пределу своих технологических возможностей, возрождается интерес к исследованию принципиально иных путей построения вычислительных средств, основанных на других системах счисления. Как одна из наиболее перспективных, многих

исследователей давно привлекает троичная симметричная система счисления (с цифрами -1,0,+1), которую когда-то Д.Кнут назвал “самой изящной” [1], и которая была полвека назад успешно воплощена в троичных цифровых машинах (ЦМ) “Сетунь” и “Сетунь-70” [2], разработанных в НИЛ ЭВМ МГУ под руководством Брусенцова Н.П.

Использование симметричного троичного кода и трёхзначной логики даёт ряд неоспоримых преимуществ (перед двоичными системами), которые были подтверждены практической эксплуатацией этих машин. Среди множества достоинств рассматриваемой троичной симметричной системы счисления можно выделить следующие:

- тем же количеством разрядов можно кодировать больший диапазон чисел;
- единообразное представление отрицательных и положительных чисел;
- непосредственное воплощение логики с тремя значениями: да (+1), нет (−1), не знаю [может быть?] (0);
- упрощённая реализация ряда арифметических операций (сдвига, сравнения чисел, смены знака); трёхзначность функции “знак числа”;
- оптимальное округление чисел простым отсечением младших разрядов и взаимокомпенсируемость погрешностей округления в процессе вычислений.

С осознанием этих преимуществ приходит понимание, что с троичными вычислениями и троичными компьютерами связано будущее вычислительной техники. И горизонты этого будущего открываются сегодня всё большему кругу научных работников и специалистов, причём не только сферы информационных технологий. Так, например, в Интернете появились статьи на тему: “Будущее квантовых компьютеров — в троичных вычислениях”[3]. Встречаются также публикации о преимуществах троичных машин, в которых в качестве одного из доводов в пользу троичности приводится упоминание о троичности нейрона человеческого мозга [4].

В НИЛ ТИ ВМК МГУ (ранее НИЛ ЭВМ), где продолжаются исследования возможных способов реализации троичного процессора на современной элементной базе [5-6], была предпринята попытка создания имитационной модели троичного процессора, а также системы разработки троичных программ для него. В результате проведённой (в 2010-2013 гг.) работы был создан программный комплекс ТВМ (Троичная Виртуальная Машина) [7], имитирующий функционирование современного варианта троичного процессора двухстековой архитектуры с поддержкой структурированного программирования на уровне машинных команд, аналогичной той, что была обеспечена в троичной ЦМ “Сетунь-70”.

В той же НИЛ ЭВМ МГУ ранее (в 80-х годах XX века) была разработана система программирования ДССП [8], которая на протяжении ряда лет применялась для программирования двоичных мини- и микрокомпьютеров самой разнообразной архитектуры и в различных операционных средах. ДССП убедительно показала, как важны для успешного построения компьютерных программ интерактивная среда разработки, двухстековая архитектура, структурированный набор команд управления и поддержка нисходящей разработки по принципу “сверху-вниз” (top-down). Поэтому в качестве основного средства разработки троичных программ для ТВМ было решено создать троичный вариант этой системы программирования, т.е. троичную ДССП с языком ДССП-Т – троичным вариантом языка ДССП.

Работа по созданию системы разработки программ для ТВМ на языке ДССП-Т была выполнена в два этапа. Сначала (на 1-ом этапе) была построена кросс-система ДССП-ТВМ [9], позволяющая создавать программы для ТВМ на языке ДССП-Т с помощью кросс-компилятора. Затем (на 2-ом этапе), используя уже саму ДССП-ТВМ как среду разработки, был создан диалоговый интерпретатор ДССП/ТВМ [10], способный функционировать на троичной машине (ТВМ) в качестве её резидентной системы программирования.

В настоящее время в языке ДССП-Т предусмотрены конструкции для структурированного, модульного, объектно-ориентированного и параллельного программирования, а также средства для структурированной обработки исключительных ситуаций.

Подробнее с возможностями ТВМ, ДССП-ТВМ и выразительными средствами языка ДССП-Т можно ознакомиться на Интернет-сайте НИЛ троичной информатики по адресу http://ternarycomp.cs.msu.ru/, а также по материалам серии статей, опубликованных в тематических сборниках «Программные системы и инструменты» факультета ВМК МГУ» [9-11] и «Труды НИИСИ РАН» ФНЦ НИИСИ РАН [12], а также доклада [13], представленного на научно-практической конференции «Посткремниевые вычисления», которая состоялась в рамках прошедшего в г. Переславль-Залесский в ноябре 2014 года Национального Суперкомпьютерного Форума (НСКФ-2014).

Созданный в НИЛ ТИ ВМК МГУ программный комплекс ДССП-ТВМ может использоваться не только в качестве среды опережающей подготовки и отладки программного оснащения для создаваемой «в кремнии» (т.е. аппаратно реализованной) троичной машины. В настоящее время ДССП-ТВМ может применяться ещё и в образовательном процессе в качестве учебной среды программирования, служить своеобразным «полигоном» для освоения специфического «искусства программирования» троичных машин и тем самым способствовать подготовке квалифицированных специалистов в области троичной информатики.

Опыт использования (студентами кафедр АСВК и алгоритмических языков во время прохождения ими преддипломной практики в НИЛ ТИ) ДССП-ТВМ в деле создания ряда учебно-исследовательских программ для ТВМ, с одной стороны, подтвердил, что и сама система разработки, и её язык относительно легко могут усваиваться новичками. Но с другой стороны, всё же выявил потребность во внедрении в учебный процесс особого спецкурса, благодаря которому заинтересованные студенты могли бы гораздо быстрее освоить специфику структурированной разработки программ для троичной машины.

В целях удовлетворения обозначенной потребности и последующего внедрения ДССП-ТВМ в учебный процесс (на факультете ВМК МГУ) был составлен и предложен учебный план годового (или полугодового) спецкурса с условным названием "Программирование в ДССП для троичной машины" (см. приложение №2). Этот спецкурс нацелен на ознакомление слушателей с троичной симметричной системой счисления, характеристиками троичных ЦМ "Сетунь" и "Сетунь-70", архитектурой и системой команд троичной виртуальной машины (ТВМ), а также возможностями среды ДССП-ТВМ и её языка ДССП-Т. Основная задача спецкурса заключается в освоении слушателями выразительных средств и приёмов разработки структурированных программ в ДССП для троичной машины. Спецкурс ориентирован на слушателей, усвоивших базовый курс программирования (например, на языке Паскаль), и может быть предложен для освоения студентам 2-4 курсов университетов или технических ВУЗов, обучающихся по направлениям: «Прикладная математика и информатика» и/или «Информатика и вычислительная техника.

Литература

1. Кнут Д. Искусство программирования на ЭВМ. т.2 Получисленные алгоритмы. п. 4.1. с. 216-219.
2. Брусенцов Н.П., Рамиль Альварес Х. Троичные ЭВМ “Сетунь” и “Сетунь 70”. // Первая Международная конференция "Развитие вычислительной техники в России и странах бывшего СССР: история и перспективы" SORUCOM-2006 (3-7 июля 2006 г., г. Петрозаводск, Россия): Труды конференции в 2-х ч. Петрозаводск: Изд-во ПетрГУ, 2006. ч.1, с. 45-51.
3. Будущее квантовых компьютеров — в троичных вычислениях. URL: http://www.infuture.ru/news.php?news_id=475 (дата обращения: 02.11.2014).
4. Малашевич Д.Б. Недвоичные системы в вычислительной технике. URL: http://www.computer-museum.ru/books/archiv/sokcon27.pdf (дата обращения: 02.11.2014)
5. Маслов С.П. Об одной возможности реализации троичных цифровых устройств // Программные системы и инструменты. Тематический сборник №12, М.: Изд-во факультета ВМК МГУ, 2011. С.222-227.
6. Маслов С.П. Троичная схемотехника // Программные системы и инструменты. Тематический сборник №13, М.: Изд-во факультета ВМК МГУ, 2012. С.152-158.
7. Сидоров С.А., Владимирова Ю.С. Троичная виртуальная машина // Программные системы и инструменты. Тематический сборник №12, М.: Изд-во ф-та ВМК МГУ, 2011. С.46-55.
8. Бурцев А.А., Сидоров С.А. История создания и развития ДССП: от "Сетуни-70" до троичной виртуальной машины. // Вторая Международная конференция "Развитие вычислительной техники и её программного обеспечения в России и странах бывшего СССР" SORUCOM-2011 (12-16 сентября 2011 г., г. Великий Новгород, Россия): Труды конференции. В.Новгород: Изд-во НовГУ, 2011. с. 83-88.
9. Бурцев А.А., Рамиль Альварес Х. Кросс-система разработки программ на языке ДССП для троичной виртуальной машины // Программные системы и инструменты. Тематический сборник №12, М.: Изд-во факультета ВМК МГУ, 2011. С.183-193.
10. Бурцев А.А., Рамиль Альварес Х. Реализация средств объектно-ориентированного программирования в кросс-компиляторе языка ДССП-Т. // Программные системы и инструменты. Тематический сборник №13, М.: Изд-во факультета ВМК МГУ, 2012. с.28-37.
11. Бурцев А.А., Рамиль Альварес Х. Сопрограммный механизм в системе структурированного программирования для троичной машины. // Программные системы и инструменты. Тематический сборник №14, М.: Изд-во факультета ВМК МГУ, 2013. с.193-208.
12. Бурцев А.А., Бурцев М.А. ДССП для троичной виртуальной машины. // Труды НИИСИ РАН, т.2, №1. М.: Изд-во НИИСИ РАН, 2012. с.73-82.
13. А.А. Бурцев, С.А. Сидоров. ТРОИЧНАЯ ВИРТУАЛЬНАЯ МАШИНА И ТРОИЧНАЯ ДССП. URL: http://2014.nscf.ru/TesisAll/0_PostMoore_Plenar/06_208_BurtsevAA.pdf (дата обращения 20.10.2015)

Приложение состоит из двух частей:

Приложение №1. Примеры программ на языке ДССП-Т:

Примеры использования цикла DW с условием

```
PROGRAM DEMO_DW
LOAD kern
 ." примеры использования цикла DW  " CR
{{ ------------ Наибольший общий делитель
 : NOD {{  [ x,y ] => [ z ] , z= nod(x,y)
     x<>y? DW x-y|y-x D ;
  : x<>y? {{ [ x,y ]} C2 C2 <> ;
  : x-y|y-x {{ [ x,y ]} x>y? BR+ x-y y-x ;
   : x>y?  {{ [ x,y ]} C2 C2 > ;
   : x-y {{ [ x,y ]} E2 C2 - E2 {{ [ x-y,y ]} ;
   : y-x {{ [ x,y ]} C2 - {{ [ x,y-x ]} ;
{{ ------------ Число Фиббоначи (n-ое по порядку)
{{           F(0)=1, F(1)=1; F(k+1)= F(k)+F(k-1)
 : Fib {{  [ n ] => [ F(n) ] 1 1 E3 1- {{ [ FP,FT,k ]} >0? DW newF D E2D ;
   : >0? C 0 > ;
   : newF {{ [ FP,FT,k ]} E2 E3 C3 + E2 1- {{ [ FT,FN,k-1 ]} ;
{{ ------------ Число Фиббоначи (первое по порядку, превыщающее M)
{{           F(0)=1, F(1)=1; F(k+1)= F(k)+F(k-1)
 : Fib> {{  [ M ] => [ Fi ] , Fi=F(i) > M
      1 1 {{ [ M,FP,FT ]} <=M? DW newFi E3 DD ;
   : <=M? C C4 <= ;
   : newFi {{ FP,FT ]} E2 C2 + {{ FT,FN ]} ;
{{ ------------ Факториал
 : Fctrl {{  [ n ] => [ n! ] , n!= 1*2*...*n   1 E2 {{ F,k ]} >0? DW F*k D ;
   : F*k {{ F,k ]} E2 C2 * E2 1- {{ F*k, k-1 ]} ;
 : DEMO_DW 77 55 NOD . 7 Fib . { 21 } 20 Fib> . 5 Fctrl . ;
UNDEF CLEAR
```

Примеры использования цикла DO- с параметром-счетчиком

```
PROGRAM DEMO_DO- { DKMON }
[] [*****   примеры использования цикла DO- с параметром-счетчиком.  *****]
[***   Вывод строки заглавных латинских букв ***]
:: : .LatString 26 DO- .LatLetter ;
   : .LatLetter [i] 'Z' C2 - TOB [i] ; [печать i-ой латинской буквы]
[***   Сумма квадратов всех целых чисел от 0 до N ***]
:: : SumKv [N] 0 E2 [S,N] 1+ DO- SumI [S] ;
   : SumI [S,i] E2 C2 C * [i,S,i*i] + E2 [S+i*i,i] ; [ i=N,N-1,..,1,0. ]
[***   Сумма квадратов всех целых чисел заданного диапазона (M..N) ***]
:: : SumKvD [M,N] C2 - 1+ 0 E2 [M,0,N-M+1] DO- SumID [M,S] E2 D [S] ;
   : SumID [M,S,i] E2 C3 C3 + [M+i] C *
        [M,i,S,(M+i)*(M+i)] + E2 [M,S+(M+i)*(M+i),i] ; [ i=N-M,..,1,0. ]
[***   Вычисление факториала ***]
::  : Factor [N] 1 E2 [1,N] DO- FI [N!] ;
   : FI [F,I] E2 C2 1+ * E2 [F*(I+1),I] ; [при I=N-1,..,1,0]
[***   Вычисление полинома по схеме Горнера ***]
   [ P(X) = (..(A9*X+A8)*X+ ..)*X+A0 ]
    9 VALUE n  [задает размер массива коэффициентов ]
   [массив коэффициэнтов A(0..9), определяющих полином : ]
   LONG CNST A 5 -2 0 1 0 0 0 0 0 0 ; [ X^3-2X+5 ]
  : Plnm [X] n A [A(n)] n [X,A(n),n] DO- PI [P(X)] ;
  : PI [X,S,i] E2 C3 * [X,i,S*X] C2 A + [S*X+A(i)] E2 [X,S,i] ;
[***   Сортировка массива методом простого выбора ***]
 9 VALUE N [размер массива]  N VCTR S [сортируемый массив S(0..N) ]
::  : SORT N 1+ [N+1] DO- SORTi ;
   : SORTi [i] MAXi  [i,k] EXCHANGEi D [i] ; [i=N,..,0]
    : MAXi [определяет индекс максим.элемента среди S0..Si ]
         [i] C [i,k] C2 [i] DO- MAXj [i,k] ;
     : MAXj [k,j] C2 S C2 S [S(k),S(j)] < IF+ k:=j [k,j] ;
      : k:=j [k,j] E2 D C [j,j] ;
    : EXCHANGEi [ переставляет i-ый и k-ый элементы массива ]
        [i,k] C S C3 S [i,k,S(k),S(i)] C3 ! S [S(i)=>S(k)]
            [i,k,S(k)] C3 ! S [S(k)=>S(i)]  [i,k] ;
[***   ввод массива S из стека ***]
::  : GETS [S0,S1,..,SN] N 1+ [N+1] DO- GETSi ;
   : GETSi [Si,i] E2 C2 ! S [i] ;
 [***  вывод массива S на терминал ***]
::  : PUT  N 1+ [N+1] DO- PUTi ;
   : PUTi [i] N C2 - S .N "," [i] ;
    : .N C IF- .- .N2 ;   : .N2 2 TON10 ; : .-  ."-" NEG ;
[***   ввод массива S с терминала ***]
::  : GET  N 1+ [N+1] DO- GETi ;
```

```
        : GETi [i] ."=" TIN N C3 - ! S [i] ;
    ::  : DEMO_DO-          ." LatString=" .LatString CR
          ." SumKv(10)=" 10 SumKv 4 TON10 CR
          ." Factor(7)=" 7 Factor 6 TON10 CR
          ." Plnm(3)="   3 Plnm   6 TON10 CR
          ." Сортировка массива " CR 55 23 67 12 99 07 66 78 33 47 GETS
          ."S[0:9]=" PUT CR ."SORT_S[0:9]" SORT CR ."S[0:9]=" PUT CR   ;
     UNDEF  CLEAR
    $$
```

Примеры программ перестановок в одномерном массиве

```
PROGRAM Test_PrstVctr { DKMON }
LOAD DKMON
{----- Ввод массива из стека -----}
::  : GetVctr {A0,A1,..,An,'A,k} {{'A=адрес,k=длина (кол-во эл-тов)
  {A0,..,An,'A,n+1} DO- Get_A[i] D { } ; {for i=n,..,1,0 A[i]:=value from stack}
    : Get_A[i] {.., X, 'A, i } E2 E3 {.., 'A, i, X } C3 C3 *3 +
       {.., 'A, i, X, 'A+i*3 } !W { A[i]:=X } {.., 'A, i } ;
{----- Печать массива как десятичных чисел -----}
::  : PrintVctr {'A,k} DO- Print_A[i] D { } ; {{ 'A=адрес,k=длина (кол-во эл-тов)
    : Print_A[i] { 'A[i], i } C2 3+ { 'A[i+1] } E3
       { 'A[i+1], i, 'A[i] } @W 0 TON10 SP  { 'A[i+1], i } ;
{------- Инверсия одномерного массива ----------}
:: : InvrVctr { A,k } {{ A - адрес массива, k= кол-во элементов , n=k-1
      {{ массив A[0:n]= [ A0,A1,..,An ] => [ An,..,A1,A0 ]
      { A,k } C 1- E2 /2 { A,n, k div2 } DO- A[i]<=>A[n-i] DD { } ;
   : A[i]<=>A[n-i] { for i= k div2,..,1,0 }
      { A,n,i } C3 C3 C3 - *3 + {..,A+(n-i)*3 } C @W {..,A[n-i] }
      { A,n,i,'A[n-i],A[n-i] } 5 CT C4 *3 + {..,A+i*3 } C @W {..,A[i] }
      { A,n,i,'A[n-i],A[n-i],'A[i],A[i] } E3 E2 !W { A[i]:=A[n-i] }
      { A,n,i,'A[n-i],A[i] } E2 !W { A[n-i]:=A[i] }     ;
{--- Циклический сдвиг одномерного массива влево. Простой вариант. ---}
:: : LShiftVctr {A,n,s} {{A-адрес массива, k=кол-во эл-тов, s=число позиций сдвига
      {{ массив A[0:n]= [ A0,A1,..,A(n-1) ] => [ As,..,A(n-1),A0,..,A(s-1) ]
      { A,n,s } DO- LShift1Vctr DD { } ;
   : LShift1Vctr { A,n,i } C3 @W { A0 } C4 {'A[0] } C4 { n } 1- DO- LShift_A[j]
      { A,n,i, A0, 'A[n] } !W { A[n]:= A0 } { A,n,i } ; { for i= s-1,.,1,0 }
   : LShift_A[j] {.., 'A[j],j } { j=n-1,..,1,0 }
      {.., 'A[j],j } C2 3+ C @W {.., 'A[j],j, 'A[j+1], A[j+1] } E2 E4
      {.., 'A[j+1],j, A[j+1], 'A[j] } !W { A[j]:=A[j+1] } {.., 'A[j+1],j } ;
{--- Циклический сдвиг одномерного массива влево. Эффективный вариант. ---}
:: : LShift!Vctr {A,n,s}{{A=адрес массива, k= кол-во эл-тов,s=число позиций сдвига
       { A,n,s } C3 C2 {.., A,s } InvrVctr
       { A,n,s } C3 C2 *3 + C3 C3 - {.., 'A[s],n-s } InvrVctr
       { A,n,s } D { A,n } InvrVctr ;
  100 VCTR A  : 'A 0 ' A ;
:: : Test_PrstVctr CR ." --- Перестановки в одномерном массиве (векторе) ---" CR
      10 11 22 33 44 55 66 77 88 99  'A 10 GetVctr  'A 10 PrintVctr CR
      'A 10 InvrVctr ." A[0:9]=> "  'A 10 PrintVctr           CR
      10 11 22 33 44 55 66 77 88 99 'A 10 GetVctr  'A 10 PrintVctr CR
      'A 10 3 LShiftVctr  'A 10 PrintVctr CR
      'A 10 2 LShift1Vctr 'A 10 PrintVctr CR
      'A 10 1 LShift1Vctr 'A 10 PrintVctr CR
      'A 10 0 LShift1Vctr 'A 10 PrintVctr CR
        10 11 22 33 44 55 66 77 88 99 'A 10 GetVctr  'A 10 PrintVctr CR
      'A 10 4 LShift!Vctr 'A 10 PrintVctr CR   ;
 UNDEF  CLEAR
$$
```

Троичная проверка попадания точки в отрезок

```
  PROGRAM Test_TSEG
  LOAD KERN
   ." Test_TSEG Троичная проверка попадания точки в отрезок " CR
  {{ { a,b, x } TSEG { r } {{ r = +1 (a<x<b) или 0 (x=a|x=b) или -1 (x<a|b<x)
    :: : TSEG { a,b,x } C E4 { x,b, x,a } CMP { x?a } E3
          { x?a, b,x } CMP { x?a,b?x } TMIN { r } ;
   10 VALUE N   { константа, задающая количество точек }
   VAR A VAR B { переменные, задающие границы отрезка [A,B] }
   9  VCTR V   { одномерный массив точек V[0:9] }
  { приём значений точек из стека данных в массив V }
  : GetV {V0,V1,..,V9} N DO- GetV[i] ;
```

```
   : GetV[i] {..,Vi,i} E2 C2 ! V {..,i} ; { i=9,8,..,1,0 }
 : .10 {Num} 0 TON10 {Num} ; { печать 10-ного числа }
 { проверка попадания точек массива в отрезок: }
 : .CheckV ."--- для отрезка [" A .10 ',' TOB B .10 ']' TOB CR CheckV ;
 : CheckV N DO- CheckV[i] { i=9,8,..,1,0 } ;
  : CheckV[i] {i} A B ." точка " C3 V {i,A,B,V[i]} C .10
   {i,A,B,V[i]} TSEG {-1/0/1} BRS )A,B( |A,B| (A,B) CR {i} ;
   : )A,B( ." вне отрезка " ;   : |A,B| ." на границе отрезка " ;
   : (A,B) ." внутри отрезка " ;
:: : Test_TSEG ."===== Проверки попадания точек в отрезок: =====" CR
    { задание массива точек V[0:9]=} 5 4 7 3 8 -4 -7 0 6 -6 GetV
    { задание границ отрезка: }  4 ! A 7 ! B  { A:= 4; B:= 7 }
    { проверки попадания точек массива в отрезок:  [4,7] } .CheckV
    { задание границ отрезка: } -6 ! A 6 ! B  { A:=-6; B:=+6 }
    { проверки попадания точек массива в отрезок: [-6,6] } .CheckV  ;
 CLEAR UNDEF
 $$
```

Новые типы данных очередь (QUEUE) и дек (DEQ) как классы

```
 PROGRAM DKMON {QDEMO}
 []
  [* ОЧЕРЕДЬ как Пример построения нового типа данных с помощью КЛАССА *]
 LOAD DKMON
   CR ." Новый Тип данных QUEUE (Очередь) " CR :: : QDEMO? 1 ;
  [--- ОЧЕРЕДЬ элементов типа TWORD (3-х трайтовых целых) ---]
  100 VALUE MaxLen [ максимальная длина очереди ]

              [--- Объявление класса ОЧЕРЕДЬ ---]
 ." -------- Объявление класса  QUEUE " CR
 :: CLASS:  QUEUE  SUBCLASS TCLASS
    [--- поля данных, составляющих очередь ---]
     VAR .cnt [ кол-во элементов в очереди ]
     VAR .n  [ индекс-указатель начала очереди в массиве ]
     VAR .k  [ индекс-указатель конца очереди в массиве ]
     100 VCTR .MI  [ массив для хранения элементов очереди ]
    [--- методы , представляющие операции над очередью ---]
   ::  METHOD Init   [ инициировать, начать работу очереди ]
   ::  METHOD Done   [ завершить, закончить работу очереди ]
   ::  METHOD Show   [ показать на экране состояние очереди ]
   ::  METHOD Put    [ поместить элемент x из вершины стека в очередь Q ]
               [ Пример вызова: {x} Put Q  ]
   ::  METHOD Get    [ взять элемент y из очереди Q на вершину стека ]
               [ Пример вызова:  Get Q {y} ]
   ::  METHOD Empty? [ проверить, пуста ли очередь ]
   ::  METHOD Full?  [ проверить, полна ли очередь ]
   ::  METHOD= ! =>  [  копирование очереди ]
   ;CLASS

 ." -------- Объявление ситуаций " CR
     SITUATION  Empty! .Empty!   : .Empty! ." Empty Queue! " HALT ;
     SITUATION  Full!  .Full!    : .Full!  ." Full Queue! "  HALT ;

          [--- Реализация методов класса ОЧЕРЕДЬ ---]
 ." -------- Реализация методов QUEUE " CR
 QUEUE :M: Init [Q] 0 C2 ! .cnt              [ Q.cnt:=0 ]
              0 C2 ! .k 1 E2 [Q] ! .n  [ ] ; [ Q.k:=0; Q.n:=1 ]
 QUEUE :M: Done [Q] D ;
 QUEUE :M: Show [Q] ."Queue:"  ShowQueue [] ;
        : ShowQueue [Q] ." (" C .cnt 2 TON10 .")= < "
             C .n C2 .cnt [Q,n,cnt] DO- ShowItem DD ."> " []  ;
        : ShowItem [Q,t,i] C2 C4 [...,t,Q] .MI [M(t)] 3 TON10 ., "
              [Q,t,i] E2 +1mod E2 [Q,(t+1)mod,i] ;
 QUEUE :M: Put [x,Q] C .Full? IF+ Full! [ проверка на искл.ситуацию ]
              C .k +1mod C C3 ! .k [ Q.k:=(Q.k+1)mod Maxlen ]
           [x,Q,k] C2 .cnt 1+ C3 ! .cnt     [ Q.cnt:=Q.cnt+1 ]
                E2 [x,k,Q] ! .MI         [ Q.M(k):=x ]
             [ ] ;
 QUEUE :M: Get [Q] C .Empty? IF+ Empty! [ проверка на искл.ситуацию ]
              C .n C2 .MI E2    [ y:= Q.M(Q.n) ]
         [y,Q] C .cnt 1- C2 ! .cnt [ Q.cnt:=Q.cnt-1 ]
          C .n +1mod E2 [y,n+1,Q] ! .n [Q.n:=(Q.n+1)mod Maxlen]
```

```
            [y] ;
      : .Empty? [Q] .cnt 0 <= [1/0] ;    [ Q.cnt<=0 ? ]
 QUEUE :M= Empty? .Empty?
      : .Full? [Q] .cnt MaxLen >= [1/0] ;    [ Q.cnt>=MaxLen ? ]
 QUEUE :M= Full? .Full?

                  [--- Вспомогательные процедуры  ---]
 : +1mod [z] 1+ C MaxLen >= IF+ T0 [z+1 или 0, если z>=MaxLen-1 ] ;
 : -1mod [z] C IF0 +MaxLen 1-      [z-1 или MaxLen-1, если z=0 ]  ;
  : +MaxLen MaxLen + ;


[-------  Примеры объявленных экземпляров классов QUEUE ----------]
." Примеры использования QUEUE " CR
 QUEUE VAR Q    QUEUE VAR Q2
 9 QUEUE VCTR VQ  [ массив очередей VQ(0:9) ]
 3 4 2 QUEUE ARR AQ  [ матрица очередей AQ(0:3,0:4) ]
:: : QDEMO CR ."Демонстрация операций с очередями " CR
     Init Q
     33 Put Q 44 Put Q 55 Put Q 66 Put Q 77 Put Q ."Q= " Show Q CR
     9 DO- InitVQ(i)
     Q 3 => VQ [ VQ(3):= Q ] -99 3 Put VQ  ."VQ(3)= " 3 Show VQ CR
     -44 4 Put VQ  3 Get VQ . SP 4 Put VQ 3 Get VQ . SP 4 Put VQ CR
     ."VQ(3)= " 3 Show VQ CR ."VQ(4)= " 4 Show VQ CR
     [ действия с одним элементом матрицы очередей: ]
     1 2 Init AQ     [ A(1,2).Init  ]
     4 VQ 1 2 => AQ  [ A(1,2):=VQ(4) ]
     666 1 2 Put AQ  [ AQ(1,2).Put(666) ]
      ."AQ(1,2)= " 1 2 Show AQ CR   ;
    : InitVQ(i) [i] C Init VQ [i] ; [ инициализация i-ой очереди ]
```

```
." Новый Тип данных DEQ на основе QUEUE " CR  :: : DQDEMO? 1 ;
            [--- Объявление класса ДЕК ---]
 [--- ДЕК элементов типа LONG (4-х байтовых целых) ---]
:: CLASS:  DEQ  SUBCLASS QUEUE [ наследует класс QUEUE ]
    [--- методы , дополняющие операции над очередью ---]
  ::  METHOD Put_ [ поместить элемент из вершины стека в начало очереди ]
  ::  METHOD Get_  [ взять элемент с конца очереди на вершину стека ]
  ::  METHOD Count [ определить количество элементов в очереди ]
  ;CLASS
          [--- Реализация методов класса ДЕК ---]
 DEQ :M= Count .cnt
 DEQ :M: Show [Q] ."Double-End-"  Show AS QUEUE [] ;
 DEQ :M: Put_ [x,Q] C .Full? IF+ Full! [ проверка на искл.ситуацию ]
              C .n -1mod C C3 ! .n [ Q.n:=(Q.n-1)mod MaxLen ]
            [x,Q,n] C2 .cnt 1+ C3 ! .cnt [ Q.cnt:=Q.cnt+1 ]
                 E2 [x,n,Q] ! .MI     [ Q.M(n):=x ] [ ] ;
DEQ :M: Get_ [Q] C .Empty? IF+ Empty! [ проверка на искл.ситуацию ]
              C .k C2 .MI E2   [ y:= Q.M(Q.k) ]
        [y,Q] C .cnt 1- C2 ! .cnt  [ Q.cnt:=Q.cnt-1 ]
        C .k -1mod E2 [y,k+1,Q] ! .k  [Q.k:=(Q.k-1)mod MaxLen] [y] ;
[-------  Примеры объявленных экземпляров класса DEQ ----------]
." Примеры использования  QUEUE и DEQ " CR
 DEQ VAR DQ  [ дек , который может использоваться и как очередь ]
 104 VCTR ZA [ произвольная область памяти, которая используется как дек ]
     : Z 0 ' ZA ;  [ её адрес ]
:: : DQDEMO CR ."Демонстрация операций с очередями и деком " CR
     Init Q
     33 Put Q  44 Put Q  55 Put Q  66 Put Q  77 Put Q ."Q= " Show Q CR
     Init DQ  Q DQ => AS QUEUE [ DQ:= Q as QUEUE ]
     88 Put DQ 22 Put_ DQ 11 Put_ DQ 99 Put DQ ."DQ= " Show DQ CR
     Get DQ 3 TON10 ." <- " Get_ DQ 2 TON10 ." -> "
     Get DQ 3 TON10  ." <- " CR  ."DQ= " Show DQ CR
     DQ => Q [ Q:= DQ as QUEUE ] Get Q 3 TON10 CR ."Q= " Show Q CR
     9 DO- InitVQ(i)
     Q 3 => VQ [ VQ(3):= Q ] -99 3 Put VQ  ."VQ(3)= " 3 Show VQ CR
     -44 4 Put VQ  3 Get VQ . 4 Put VQ SP
     3 Get VQ .  4 Put VQ CR ."VQ(3)= " 3 Show VQ CR
     ."VQ(4)= " 4 Show VQ CR ."VQ(4)= " 4 VQ Show AS DEQ CR ;
CR ." Для запуска демонстрации выполните QDEMO или DQDEMO " CR
UNDEF CLEAR
$$
```

Учебный план спецкурса "Программирование в ДССП для троичной машины" ("Programming in DSSP for ternary computer"):

Учебный план спецкурса

"Программирование в ДССП для троичной машины"

("Programming in DSSP for ternary computer")

Спецкурс "Программирование в ДССП для троичной машины" нацелен на ознакомление слушателей с троичной симметричной системой счисления, характеристиками троичных ЦМ "Сетунь" и "Сетунь-70", архитектурой и системой команд троичной виртуальной машины (ТВМ), а также возможностями Диалоговой Системы Структурированного Программирования (ДССП), созданными в НИЛ троичной информатики факультета ВМК МГУ.

Основная задача спецкурса заключается в освоении слушателями выразительных средств и приёмов разработки структурированных программ в ДССП для троичной машины.

1. Введение. Особенности троичной симметричной системы счисления. Краткая характеристика ЦМ "Сетунь" и "Сетунь-70". История создания и развития ДССП.

2. Краткая характеристика ТВМ, ДССП-ТВМ, ДССП/ТВМ . ТВМ — программный комплекс, имитирующий современный вариант троичного процессора двухстековой архитектуры с поддержкой структурированного программирования на уровне машинных команд. ДССП-ТВМ — кросс-система разработки программ для него на языке ДССП-Т. Структура и состав ДССП-ТВМ, прогон ДССП-программ в ней. Диалоговый интерпретатор ДССП/ТВМ и приёмы простейшей работы в нём.

3. Словарная организация ДССП. Программирование в ДССП как наращивание словаря исполнителя. Простейшие приёмы определения новых слов.

4. ДССП как стековая машина. Операции целочисленной арифметики над арифметическим стеком. Представление числовых констант, символов и строк. Работа с ДССП как со стековым калькулятором. Отладочные средства просмотра содержимого стеков и памяти.

5. Процедуры в ДССП. Приёмы определения новых процедур. Принципы структурированного программирования. Пошаговая нисходящая разработка программ в ДССП.

6. Организация ветвлений в ДССП. Команды условного вызова процедур. Операции для вычисления логических условий (выражений). Поразрядные операции троичной логики и арифметики. Примеры разработки простейших программ с ветвлением.

7. Организация циклов в ДССП. Простейшие команды многократного вызова процедур. Примеры разработки простейших программ с циклами.

8. Переменные и массивы в ДССП. Объявления для определения новых слов-переменных и слов-массивов. Простейшие типы данных. Простейшие операции над переменными и массивами. Примеры программ обработки массивов.

9. Доступ к памяти по адресам. Примеры разработки программ для обработки произвольных массивов, задаваемых в качестве параметров. Приёмы создания реентерабельных программ в ДССП.

10. Простейшие операции ввода и вывода с терминалом. Ввод и вывод символов, строк, чисел. Операции анализа символов и обработки строк. Операции преобразования строки в число и числа в строку. Примеры программ обработки текстовых строк.

11. Команды управления загрузкой, чисткой и сохранением словаря. Управление видимостью имён.

12. Словарь ядра и ДССП-библиотека. Характеристика ассортимента слов базового словаря и возможностей модулей ДССП-библиотеки. Простой диалоговый командный монитор (ДКМОН).

13. Команды доступа к двоичным файлам среды. Примеры разработки программ обработки файлов. Особенности обработки текстовых файлов.

14. Механизм обработки исключительных ситуаций. Объявление слов-ситуаций. Команды установки реакций на ситуации. Команды возбуждения ситуаций. Примеры разработки программ с применением механизма ситуаций.

15. Команды циклов с выходами путём возбуждения ситуации. Примеры разработки программ, применяющих циклы с выходами по ситуации.

16. Объявление новых типов как структур. Операции над объектами-структурами и отдельными их полями. Примеры программ с применением структур.

17. Объявление новых типов как классов. Определение новых операций как методов класса. Определение процедур реализации методов. Объявление класса-наследника. Переопределение методов. Спецоперации (AS, CONSTRUCT) для работы с объектами классов в

произвольном участке памяти. Примеры определения новых типов как классов.

18. Сопрограммный механизм в ДССП. Объявление сопрограммы. Операция контекстного переключения. Примеры применения сопрограмм.

19. Простейший монитор управления параллельными процессами. Объявление процесса и операции с ним. Примеры применения монитора. Приёмы его реализации на основе сопрограммного механизма.

20. Определение слов в коде ассемблера ТВМ. Характеристика архитектуры и системы команд ТВМ. Особенности входного языка ассемблера ТВМ. Примеры реализации слов ядра ДССП на ассемблере.

21. Особенности сшитого кода ДССП для ТВМ. Строение тел процедур, констант, переменных, массивов. Операции доступа к содержимому управляющего стека. Приёмы определения особого вида слов с получением доступа к содержимому управляющего стека и позициям тел сшитого кода.

Литература, предлагаемая слушателям спецкурса:

1. Кнут Д. Искусство программирования на ЭВМ. Т. 2: Получисленные алгоритмы. — М.:Мир,1977. п. 4.1. с. 216–219.

2. Дал У., Дейкстра Э., Хоор К. Структур(ирован)ное программирование.— М.: Изд-во Мир, 1975.— 247 с.

3. Брусенцов Н. П., Рамиль Альварес Х. Троичные ЭВМ «Сетунь» и «Сетунь-70» // Труды 1-ой международной конференции «Развитие вычислительной техники в России и странах бывшего СССР: история и перспективы» SORUCOM-2006, г. Петрозаводск, Россия, 3–7 июля 2006 г.— Петрозаводск: Изд-во ПетрГУ, 2006.— Ч.1, с. 45–51.

4. Бурцев А. А., Сидоров С. А. История создания и развития ДССП: от «Сетуни-70» до троичной виртуальной машины // Труды 2-ой международной конференции «Развитие вычислительной техники и её программного обеспечения в России и странах бывшего СССР» SORUCOM-2011, г. Великий Новгород, Россия, 12–16 сентября 2011 г.— В.Новгород: Изд-во НовГУ, 2011.— С. 83–88.

5. Сидоров С. А., Владимирова Ю. С. Троичная виртуальная машина // Программные системы и инструменты. Тематический сборник №12.— М.: Изд-во ф-та ВМК МГУ, 2011, с. 46–55.

6. Бурцев А. А., Рамиль Альварес Х. Кросс-система разработки программ на языке ДССП для троичной виртуальной машины // Программные системы и инструменты. Тематический сборник №12.— М.: Изд-во факультета ВМК МГУ, 2011, с. 183–193.

7. Бурцев А. А., Бурцев М. А. ДССП для троичной виртуальной машины // Труды НИИСИ РАН, т. 2, № 1.— М.: Изд-во НИИСИ РАН, 2012.— С. 73–82.

8. Бурцев А. А., Рамиль Альварес Х. Реализация средств объектно-ориентированного программирования в кросс-компиляторе языка ДССП-Т // Программные системы и инструменты. Тематический сборник № 13.— М.: Изд-во факультета ВМК МГУ, 2012, с. 28–37.

9. Бурцев А. А., Шумаков М. Н. Сопрограммный механизм в ДССП как основа для построения мониторов параллельных процессов // Вопросы кибернетики. Сборник статей / ред. В .Б. Бетелин.— М.: Изд-во НИИСИ, 1999.— С. 45–63.

10. Бурцев А. ., Рамиль Альварес Х. Сопрограммный механизм в системе структурированного программирования для троичной машины // Программные системы и инструменты. Тематический сборник № 14.— М.: Изд-во факультета ВМК МГУ, 2013.— С. 193–208.

11. Баранов С.Н., Ноздрунов Н.Р. Язык Форт и его реализации.— Л.: Машиностроение, 1988. — 157 с.

12. Материалы по троичной информатике на Интернет-сайте Научно-Исследовательской Лаборатории Троичной Информатики (НИЛ ТИ) факультета ВМК МГУ. URL: http://ternarycomp.cs.msu.ru/ (дата обращения 20.10.2015).

ПАРАЛЛЕЛЬНОЕ И РАСПРЕДЕЛЕННОЕ ПРОГРАММИРОВАНИЕ, ГРИД-ТЕХНОЛОГИИ, ПРОГРАММИРОВАНИЕ НА ГРАФИЧЕСКИХ ПРОЦЕССОРАХ

Курочкин И.И.[1], Гуменный Д.Г.[2]

[1]ИППИ РАН, г.Москва, к.т.н., с.н.с., kurochkin@iitp.ru

[2]НИТУ «МИСиС», г.Москва, студент, koldanst@hotmail.com

БЕЗОПАСНОСТЬ СЕТЕЙ SDN. КЛАССИФИКАЦИЯ АТАК

КЛЮЧЕВЫЕ СЛОВА

SDN, безопасность, сетевые атаки, программно-конфигурируемые сети.

АННОТАЦИЯ

Software-defined Networking (SDN) – новый подход к построению архитектуры компьютерных сетей. Уровень управления сетью и уровень передачи данных разделяются, а функции управления сетью переносятся с маршрутизаторов и коммутаторов на программные приложения, запущенные на отдельном сервере – контроллере. В статье приведен обзор наиболее распространенных атак на сети SDN, их классификация и цели, которых хотят достигнуть злоумышленники.

Введение

Software-defined Networking (SDN) это новая идеология построения компьютерных сетей, в которой весь «интеллект», все управление сетью вынесено на отдельную аппаратно-программную базу (контроллер), а все управление трафиком происходит с использованием специальных протоколов, оперирующих понятием «поток» (flow) и управляющих им [1]. На контроллере устанавливается политика управления сетью на основе заданных правил и особенностей работы специализированных приложений.

По результатам экспериментов было определено, что с помощью технологии SDN можно повысить эффективность сетевого оборудования, а также снизить затраты на эксплуатацию сетей. Основная цель – превратить управление сетями из искусства в инженерию [2], повысив безопасность. Внедрение SDN должно оказать значительное влияние на сети дата-центров, корпоративные сети и пр.

Основные идеи, заложенные в SDN, заключаются в следующем:

- Разделение уровня передачи и уровня управления данными.
- Единый, унифицированный, независящий от поставщика интерфейс между уровнем управления и уровнем передачи данных.
- Логически централизованный уровень управления данными. Централизованное управление осуществляется с помощью контроллера с установленной сетевой операционной системой и набором сетевых приложений, развернутых поверх ее.
- Виртуализация физических ресурсов сети.

OpenFlow является основной движущей силой концепции программно-конфигурируемых сетей и наиболее широко применимым стандартом их построения.

Главной особенностью сетей SDN являются широкие возможности по виртуализации, динамическому применению сетевых политик и лучшему контролю над различными элементами сети при малой требовательности к вычислительным ресурсам. Однако централизованный контроллер сети требует повышенного внимания со стороны администратора для обеспечения безопасности и корректного функционирования всей сети. Подвергшиеся атаке элементы сети могут быть использованы для утечки информации, атаки на других пользователей или для вывода сети из строя.

Существующие на момент написания статьи атаки на сети SDN в том или ином виде эксплуатируют основную их особенность – централизованный контроллер. Их можно классифицировать по типу угрозы:

- Атаки на топологию сети:
  - ARP Poisoning (подмена ARP пакетов);
  - Fake topology (перестройка топологии сети);
- Атаки на уровень управления данными:
  - DoS контроллера;

- TCAM exhaustion (переполнение памяти коммутаторов);
- Switch blackhole (создание «черной дыры» в сети);

- Атака на уровень SDN.
  - Атаки на топологию сети.

SDN контроллер обрабатывает множество пакетов (ARP, IGMP, LLDP и другие), отправляемых с коммутаторов в рамках протокола OpenFlow, чтобы сконструировать свое представление о сетевой топологии. Эти сообщения можно подменить, что ведет к появлению возможности атаковать сеть.

1.1. ARP Poisoning

Хосты, с которых проводится атака, могут подменять информацию о физических хостах в сети с помощью поддельных ARP запросов. В результате, контроллер устанавливает вредоносные правила для потоков в сети и меняет их направление для перехвата трафика, предназначавшегося другому хосту. Также, злоумышленник может инициировать случайные потоки, чтобы заставить контроллер и коммутаторы создавать петли и «черные дыры» в сети или для осуществления IP splicing атаки (атака, при которой установившаяся сессия перехватывается и злоумышленник выдает себя за уже авторизованного пользователя подменой IP).

1.2. Fake topology

Один зараженный хост пытается создать поддельное звено в сети, используя линейную топологию, состоящую из трех коммутаторов X, Y и Z. Сервер A соединен с коммутатором X. Сервер отправляет вредоносный LLDP пакет, как будто он пришел от коммутатора Z. В результате атаки создается поддельное однонаправленное ребро от Z к X в представлении контроллера и происходит пересчет путей маршрутизации, что может нарушить маршрутизацию и правильное направление потоков в сети.

2. Атаки на уровень управления данными

Зараженные хосты и коммутаторы могут провести DoS (Denial-of-service) атаку, загрузив сеть трафиком к случайным хостам, чтобы вывести из строя ресурсы на уязвимых коммутаторах и/или сам SDN контроллер, влияя на пересылку данных.

2.1. DoS контроллера

Протокол OpenFlow требует от коммутаторов отправлять пакеты к контроллеру, если очереди на вход заполнены. Загрузив сеть большим числом пакетов можно значительно увеличить вычислительную нагрузку на контроллер или даже прервать его работу.

2.2. TCAM exhaustion

Данные о маршрутизации потоков в сети хранятся в быстрой ассоциативной памяти TCAM. Злоумышленник может нацелиться на эту память коммутатора, чтобы атаковать другие хосты. Зараженный хост отправляет множество сообщений в сеть и вынуждает контроллер устанавливать большое количество правил для потоков, расходуя TCAM. Впоследствии, новые правила не могут быть запомнены коммутатором, пока старые не выполнятся. Если коммутатор находится в ключевой части сети, высокая задержка или утеря пакетов гарантированы.

2.3. Switch blackhole

«Черная дыра» (blackhole) это состояние сети, когда потоки резко обрываются и пакеты не получается доставить до пункта назначения. Зараженный коммутатор может сбрасывать или перенаправлять пакеты и в результате поток не доходит. Это может значительно нарушить работу сети и бизнес-приложений, которые не получат нужные данные.

3. Атаки на уровень SDN

Злоумышленник выбирает целью своей атаки API сервера. Сервер может использовать любой из множества языков программирования. Если получится использовать уязвимость в API, а на контроллере не будет никакой защиты, тогда можно получить полный контроль за сетью SDN через контроллер. К примеру, администратор забыл поменять пароль по умолчанию. В таком случае, злоумышленник посылает пакеты напрямую в управляющий интерфейс контроллера, получает сведения о конфигурации сети или меняет ее.

Заключение

Безопасность играет ключевую роль в дальнейшей миграции сетевых сервисов в «облака» и для дальнейшего развития SDN. Список областей, где пока не стоит использовать SDN достаточно велик, однако со временем технология будет усовершенствована и для нее будут разработаны надежные системы управления. Вместе с тем, следует отметить, что ЦОД должны сами обеспечивать защиту вычислительной архитектуры и целостность SDN, а программный контроллер, управляющий сетью, является частью общей управляющей системы всего ЦОД.

Преимуществом технологии SDN и ее слабым местом является централизованное управление сетью и маршрутизацией в ней. Обманув контроллер или получив над ним контроль можно в той или иной мере нарушить работу всей сети.

Литература

1. Abusing Software Defined Networks, Gregort Pickett
2. Программно-конфигурируемые сети – Новый подход к построению компьютерных сетей, Р.Л. Смелянский
3. Martin Casado, Tal Gerfinkel, Aditya Akella, Michael J. Freedman Dan Boneh. Nick Mckeown, Scott Shenker SANE: A protection Architecture for Enterprise Networks 15-th Usenix Secutiry Symposium, Vancouver, Canada, August 2006.
4. N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker and J. Turner, “Openflow. Enabling innovation in campus networks.” SIGCOMM Computer Communication Review, vol. 38, no. 2, pp. 69-74, 2008.
5. SND Security Considerations in the Data Center, ONF Solution Brief, October 8, 2013
6. Open Networking Foundation. OpneFlow Switch Specification:Version 1.3.0. https://www.opennetsummit.org/why.html
7. Rob Sherwood, Glen Gibby, Kok-Kiong Yapy, Guido Appenzellery, Martin Casado, Nick McKeowny, Guru Parulkary, “FlowVisor: A Network Virtualization Layer”, 2009
8. Детские болезни SDN, Валерий Коржов, “Открытые системы”, №6, 2013.

Захаров В.Н.[1], Мунерман В.И.[2]

[1]Федеральний исследовательский центр «Информатика и управление» РАН, г. Москва, д.т.н., ученый секретарь, VZakharov@ipiran.ru,

[2]Смоленский государственный университет (СмолГУ), к.т.н., доцент, vimoon@gmail.com

ПАРАЛЛЕЛЬНЫЙ АЛГОРИТМ УМНОЖЕНИЯ МНОГОМЕРНЫХ МАТРИЦ

КЛЮЧЕВЫЕ СЛОВА

Многомерные матрицы, параллельное программирование, модели данных

АННОТАЦИЯ

В статье рассматривается параллельный алгоритм умножения многомерных матриц, необходимый для построения программно-аппаратного комплекса, ориентированного на массовую параллельную обработку больших объемов структурированных данных на основе многомерно-матричной модели данных. Выбор этой модели данных целесообразен в том случае, когда свойства данных в конкретной задаче таковы, что соответствующие им многомерные матрицы не будут разреженными. Приводится описание алгоритма и архитектура программно-аппаратного комплекса.

**Работа выполнена при частичной поддержке гранта РФФИ № 13-07-00579.*

***Часть экспериментальных результатов получена в рамках гранта, предоставленного Смоленскому государственному университету корпорацией Microsoft.*

Для параллельного умножения обычных матриц, как правило, используются два вида алгоритмов [1]: ленточные, реализующие поэлементное умножение матриц и блочные, основанные на методе Фробениуса [2]. Многие авторы, проводившие анализ эффективности параллельного умножения матриц [3, 4, 5], отдают предпочтение блочным алгоритмам, утверждая, что последние обладают высокой степенью масштабируемости. Учитывая тот факт, что масштабируемость – это наиболее важное свойство параллельных алгоритмов и вычислительных комплексов их реализующих, в дальнейшем будет рассматриваться параллельный алгоритм блочного умножения многомерных матриц. Метод Фробениуса естественно обобщается на многомерные матрицы. Известны различные реализации параллельного умножения матриц: алгоритмы Фокса и Кэннона [1, универсальный алгоритм для параллельных вычислительных комплексов с распределенной памятью PUMMA [5], масштабируемый универсальный алгоритм SUMMA [4]. Для обобщения на многомерные матрицы далее рассматривается алгоритм Кэннона, однако, предложенный метод обобщения может быть распространен и на другие алгоритмы блочного умножения матриц.

Многомерная матрица определяется как система ($n_1 \times ... \times n_p$) элементов $a_{i_1,\dots,i_p}(i_\alpha = 1,..,n_\alpha, \alpha = 1,\dots,p)$, расположенных в точках p-мерного пространства, определяемых координатами $i_1 \dots i_p$, и обозначается $A = \left\| a_{i_1,\dots,i_p} \right\|$ [6]. Элементы многомерных матриц могут быть произвольной природы: логические значения, числа, строки, кортежи, при условии, что над ними определены две операции, одна из которых трактуется как аддитивная, а другая – как мультипликативная.

Умножение многомерных матриц определяется следующим образом. Пусть даны p-мерная матрица $A = \left\| a_{i_1,\dots,i_p} \right\|$ и q-мерная матрица $B = \left\| b_{i_1,\dots,i_q} \right\|$. Можно разбить совокупности индексов $i_1 \dots i_p$ и $i_1 \dots i_q$ на четыре группы, содержащие соответственно k, λ, μ и ν индексов ($k, \lambda, \mu, \nu > 0$). Причем $k + \lambda + \mu = p$, а $\lambda + \mu + \nu = q$. Разбиение порождает четыре группы индексов: $l = (l_1, \dots, l_k)$, $s_1, \dots, s_\lambda$, $c = (c_1, \dots, c_\mu)$ и $m = (m_1, \dots, m_\nu)$. Индексы разбиений s и c (они называются скоттовыми и кэлиевыми индексами) принадлежат обеим матрицам. Тогда матрицы A и B можно представить в виде $A = \left\| a_{lsc} \right\|$ и . $B = \left\| b_{scm} \right\|$ Очевидно, что если индексу $s_\gamma(c_\gamma)$ соответствуют индекс i_α в матрице A и индекс i_β в матрице B, то $n_{i_\alpha} = n_{i_\beta}$. Матрица

$C=\left\|c_{lsm}\right\|$, элементы которой вычисляются по формуле $c_{lsm}=\sum_{(c)} a_{lsc}\times b_{scm}$, называется (λ,μ)-свернутым произведением и обозначается $^{\lambda,\mu}(A\times B)$.

Для построения алгоритма параллельного умножения многомерных матриц необходимо определить понятие сечения многомерной матрицы. Пусть в *m* индексах ($1 \le m \le p$) совокупности индексов $(i_1,\dots,i_p)$ матрицы зафиксировано по одному значению. Для простоты и без ограничения общности можно считать, что это индексы $(i_1,\dots,i_m)$. $(p-m)$-мерная матрица $\left\|A_{(i_1^0,\dots,i_m^0 i_{m+1},\dots,i_p)}\right\|$, состоящая только из тех элементов матрицы $A=\left\|a_{i_1,\dots,i_p}\right\|$, в которых индексы $(i_1,\dots,i_m)$ имеют единственное фиксированное значение $(i_1^0,\dots,i_m^0)$, называется простым *m*-кратным сечением матрицы *A* ориентации $(i_1,\dots,i_m)$.

Особенность умножения многомерных матриц, отличающая его от умножения обычных (плоских) матриц и тензоров, заключатся в наличии скоттовых индексов, которые присутствуют в обеих матрицах-операндах и в матрице-результате. Эта особенность существенно сказывается на алгоритме умножения многомерных матриц. При отсутствии скоттовых индексов этот алгоритм мало отличается от алгоритма умножения плоских матриц. Далее рассматривается алгоритм параллельного умножения многомерных матриц при наличии скоттовых и кэлиевых индексов $(\lambda>0,\mu>0)$. Для простоты, но без ограничения общности, удобно считать, что количества значений всех индексов во всех разбиениях *l*, *s*, *c*, *m* матриц-сомножителей *A* и *B* одинаковы, то есть равны числу *n*, которое делится на целое число *E*.

Пусть даны: *p*-мерная матрица *A* (A_{lsc}), *q*-мерная матрица *B* (B_{scm}) и *r*-мерная матрица *C* (C_{lsm}) $(r=p+q-\lambda-2\mu)$ такая, что $C=^{\lambda,\mu}(A\times B)$. Если зафиксировать по одному значению каждого из скоттовых индексов, одинаковому во всех трех матрицах ($s_1^0,\dots,s_\lambda^0$), то будут получены: ($p-\lambda$)-кратное сечение матрицы *A*, ($q-\lambda$)-кратное сечение матрицы *B* и ($r-\lambda$)-кратное сечение матрицы *C*. Все эти сечения будут иметь ориентацию $s_1,\dots,s_\lambda$. Пусть $s_1^1,\dots,s_\lambda^1$ и $s_1^2,\dots,s_\lambda^2$ фиксированные наборы значений скоттовых индексов многомерных матриц *A*, *B* и *C*, отличающиеся друг от друга, по крайней мере, одним значением. Этим наборам значений скоттовых индексов соответствуют два различных сечения каждой из матриц. Элементы соответствующих сечений матрицы *C* вычисляются по следующим формулам:

$$c_{l_1\dots l_\kappa s_1^1\dots s_\lambda^1 m_1\dots m_\nu}=\sum_{c_1,\dots,c_\mu=0}^{n} a_{l_1\dots l_\kappa s_1^1\dots s_\lambda^1 c_1\dots c_\mu}\cdot b_{s_1^1\dots s_\lambda^1 c_1\dots c_\mu m_1\dots m_\nu},$$

$$c_{l_1\dots l_\kappa s_1^2\dots s_\lambda^2 m_1\dots m_\nu}=\sum_{c_1,\dots,c_\mu=0}^{n} a_{l_1\dots l_\kappa s_1^2\dots s_\lambda^2 c_1\dots c_\mu}\cdot b_{s_1^2\dots s_\lambda^2 c_1\dots c_\mu m_1\dots m_\nu}.$$

Из приведенных формул вытекает следующее утверждение:

Результат (λ,μ)-свернутого произведения двух многомерных матриц *A* и *B* размерностей *p* и *q*, при наличии скоттовых индексов ($\lambda>0$), есть *r*-мерная матрица $C(r=p+q-\lambda-2\mu)$, составленная из ($r-\lambda$)-кратных сечений, каждое из которых есть произведение ($p-\lambda$)-кратного сечения матрицы *A* и ($q-\lambda$)-кратного сечения матрицы *B*. При этом наборы значений индексов $s_1,\dots,s_\lambda$ сечений-сомножителей и сечения-результата совпадают.

Как и в случае умножения плоских матриц, алгоритм умножения многомерных матриц – простых сечений матриц *A* и *B* ориентации $(s_1,\dots,s_\lambda)$ реализуется параллельно выполняемыми процессами. Поскольку умножения этих сечений не зависят друг от друга, далее рассматривается умножение в рамках только одного сечения. Каждый процесс выполняет умножение блоков этих матриц *A* и *B*, в результате которого получается блок матрицы *C*.

Блок матрицы *A* – это ее сечение, в котором индексы разбиения *l* принимают *E* последовательных значений, индексы разбиения *c* принимают все возможные значения, индексы разбиения *s* – единственное значение.

Блок матрицы *B* – это ее сечение, в котором индексы разбиения *m* принимают *E* последовательных значений, индексы разбиения *c* принимают все возможные значения, индексы разбиения *s* – единственное значение.

Блок матрицы *C* – это ее сечение, в котором индексы разбиения *l*, *c* и *m* принимают *E* последовательных значений, индексы разбиения *s* – единственное значение.

Общее число процессов, необходимых для вычисления одного сечения ориентации $(s_1, \dots, s_\lambda)$ матрицы-результата *C* определяется, общим числом индексов разбиений *l* и *m*, равным $\kappa + \nu$ и, в рассматриваемом случае, числом *E*, которому кратно *n* – количество значений каждого индекса. Тогда индекс процесса можно представить в виде триады: $N_E^l < s_1^0, \dots, s_\lambda^0 > N_E^m$, где N_E^1 и N_E^m – числа в *E*-ричной системе счисления, принимающие значения от 0 до $E^\kappa - 1$ и $E^\mu - 1$. Следовательно, количество процессов, необходимых для вычисления одного сечения матрицы *C* равно $E^{\kappa+\nu}$.

Как и большинство алгоритмов умножения матриц, алгоритм умножения многомерных матриц состоит из двух этапов.

На первом этапе производится "обнуление" (заполнение значениями нейтрального элемента) блоков сечения матрицы *C* в соответствующих процессах и распределение по процессам блоков сечений матриц *A* и *B*. Распределение может быть физическим, если каждый процесс связан с автономным запоминающим устройством, или логическим, если все процессы обрабатывают данные расположенные на одном запоминающем устройстве.

В начальном состоянии предполагается, что каждый процесс связан с блоками матриц *A* и *B*, индексы которых соответствуют индексам процесса. Это означает, что для сечения ориентации $s_1^*, \dots, s_\lambda^*$ при фиксированных *i* и *j* $(i = 0, \dots, \overline{(E-1)\dots(E-1)}_\kappa, j = 0, \dots, \overline{(E-1)\dots(E-1)}_\nu,$ где $\overline{(E-1)\dots(E-1)}_\kappa = E^\kappa - 1, \quad \overline{(E-1)\dots(E-1)}_\nu = E^\nu - 1)$ процессу $P_{l_1^i \dots l_k^i s_1^* \dots s_\lambda^* m_1^j \dots m_\nu^j}$ будут соответствовать блоки $A_{L_1^i \dots l_k^i s_1^* \dots s_\lambda^* c_1^j \dots c_\nu^j}$ и $B_{L_1^i \dots l_k^i c_1^* \dots c_\lambda^* m_1^j \dots m_\nu^j}$ матриц *A* и *B*. Начальная привязка блоков матриц *A* и *B* к процессам приведена в таблице 1.

Таблица 1. Начальная привязка блоков матриц-операндов к процессам

$P_{\underbrace{0\dots0}_{\kappa} s_1^* \dots s_\lambda^* \underbrace{0\dots0}_{\nu}}$ $A_{\underbrace{0\dots0}_{\kappa} s_1^* \dots s_\lambda^* \underbrace{0\dots0}_{\mu}}$ $B_{s_1^* \dots s_\lambda^* \underbrace{0\dots0}_{\mu} \underbrace{0\dots0}_{\nu}}$	$P_{\underbrace{0\dots1}_{\kappa} s_1^* \dots s_\lambda^* \underbrace{0\dots0}_{\nu}}$ $A_{\underbrace{0\dots1}_{\kappa} s_1^* \dots s_\lambda^* \underbrace{0\dots0}_{\mu}}$ $B_{s_1^* \dots s_\lambda^* \underbrace{0\dots1}_{\mu} \underbrace{0\dots0}_{\nu}}$	…	$P_{\underbrace{E-1\dots E-1}_{\kappa} s_1^* \dots s_\lambda^* \underbrace{0\dots0}_{\nu}}$ $A_{\underbrace{E-1\dots E-1}_{\kappa} s_1^* \dots s_\lambda^* \underbrace{0\dots0}_{\mu}}$ $B_{s_1^* \dots s_\lambda^* \underbrace{E-1\dots E-1}_{\mu} \underbrace{0\dots0}_{\nu}}$
$P_{\underbrace{0\dots0}_{\kappa} s_1^* \dots s_\lambda^* \underbrace{0\dots1}_{\nu}}$ $A_{\underbrace{0\dots0}_{\kappa} s_1^* \dots s_\lambda^* \underbrace{0\dots1}_{\mu}}$ $B_{s_1^* \dots s_\lambda^* \underbrace{0\dots0}_{\mu} \underbrace{0\dots1}_{\nu}}$	$P_{\underbrace{0\dots1}_{\kappa} s_1^* \dots s_\lambda^* \underbrace{0\dots1}_{\nu}}$ $A_{\underbrace{0\dots1}_{\kappa} s_1^* \dots s_\lambda^* \underbrace{0\dots1}_{\mu}}$ $B_{s_1^* \dots s_\lambda^* \underbrace{0\dots1}_{\mu} \underbrace{0\dots1}_{\nu}}$	…	$P_{\underbrace{E-1\dots E-1}_{\kappa} s_1^* \dots s_\lambda^* \underbrace{0\dots1}_{\nu}}$ $A_{\underbrace{E-1\dots E-1}_{\kappa} s_1^* \dots s_\lambda^* \underbrace{0\dots1}_{\mu}}$ $B_{s_1^* \dots s_\lambda^* \underbrace{E-1\dots E-1}_{\mu} \underbrace{0\dots1}_{\nu}}$
…	…	…	…
$P_{\underbrace{0\dots0}_{\kappa} s_1^* \dots s_\lambda^* \underbrace{E-1\dots E-1}_{\nu}}$ $A_{\underbrace{0\dots0}_{\kappa} s_1^* \dots s_\lambda^* \underbrace{E-1\dots E-1}_{\mu}}$ $B_{s_1^* \dots s_\lambda^* \underbrace{0\dots0}_{\mu} \underbrace{E-1\dots E-1}_{\nu}}$	$P_{\underbrace{0\dots1}_{\kappa} s_1^* \dots s_\lambda^* \underbrace{E-1\dots E-1}_{\nu}}$ $A_{\underbrace{0\dots1}_{\kappa} s_1^* \dots s_\lambda^* \underbrace{E-1\dots E-1}_{\mu}}$ $B_{s_1^* \dots s_\lambda^* \underbrace{0\dots1}_{\mu} \underbrace{E-1\dots E-1}_{\nu}}$	..	$P_{\underbrace{E-1\dots E-1}_{\kappa} s_1^* \dots s_\lambda^* \underbrace{E-1\dots E-1}_{\nu}}$ $A_{\underbrace{E-1\dots E-1}_{\kappa} s_1^* \dots s_\lambda^* \underbrace{E-1\dots E-1}_{\mu}}$ $B_{s_1^* \dots s_\lambda^* \underbrace{E-1\dots E-1}_{\mu} \underbrace{E-1\dots E-1}_{\nu}}$

Распределение блоков основано та том, что наборы фиксированных значений индексов разбиений *l* и *m* можно рассматривать как числа в *E*-ричной системе счисления $(\overline{l_1^i \dots l_\kappa^i})_E$ и $(\overline{m_1^j \dots m_\nu^j})_E$. Тогда первый этап алгоритма состоит из следующих шагов:

1. Блок матрицы *A* с набором значений индексов $(\overline{m_1^j \dots m_\nu^j})_E$ разбиения *m* связывается с процессом, набор значений индексов разбиения *m* которого вычисляется по формуле
$$(E^\nu + (\overline{m_1^j \dots m_\nu^j})_E - j) \bmod E^\nu \ (j = 0, \dots, \underbrace{(E-1)\dots(E-1)}_{\kappa}).$$
2. Блок матрицы *B* с набором значений индексов $(\overline{l_1^i \dots l_\kappa^i})_E$ разбиения *l* связывается с процессом, набор значений индексов разбиения *m* которого вычисляется по формуле
$$(E^\kappa + (\overline{l_1^i \dots l_\kappa^i})_E - i) \bmod E^\kappa \ (i = 0, \dots, \underbrace{(E-1)\dots(E-1)}_{\nu}).$$

Привязка блоков матриц *A* и *B* к процессам после завершения первого этапа алгоритма приведена в таблице 2.

Второй этап алгоритма состоит из *E* итераций, на каждой из которых выполняются три действия:

1. Каждый процесс $P_{l_1^i \dots l_k^i s_1^* \dots s_\lambda^* m_1^j \dots m_\nu^j}$ выполняет умножение связанных с ним блоков матриц *A* и *B* и складывает результат умножения с соответствующим ему блоком матрицы *C*.
2. Блок матрицы *A* с набором значений индексов $(\overline{m_1^j \dots m_\nu^j})_E$ разбиения *m* связывается с процессом, набор значений индексов разбиения *m* которого вычисляется по формуле
$$(E^\nu + (\overline{m_1^j \dots m_\nu^j})_E - 1) \bmod E^\nu \ (j = 0, \dots, \underbrace{(E-1)\dots(E-1)}_{\kappa}).$$
3. Блок матрицы *B* с набором значений индексов $(\overline{l_1^i \dots l_\kappa^i})_E$ разбиения *l* связывается с процессом, набор значений индексов разбиения *m* которого вычисляется по формуле
$$(E^\kappa + (\overline{l_1^i \dots l_\kappa^i})_E - 1) \bmod E^\kappa \ (i = 0, \dots, \underbrace{(E-1)\dots(E-1)}_{\nu}).$$

Таблица 2. Привязка блоков матриц-операндов к процессам после выполнения первого этапа алгоритма

$P_{\underbrace{0\dots0}_{\kappa} s_1^* \dots s_\lambda^* \underbrace{0\dots0}_{\nu}}$ $A_{\underbrace{0\dots0}_{\kappa} s_1^* \dots s_\lambda^* \underbrace{0\dots0}_{\mu}}$ $B_{s_1^* \dots s_\lambda^* \underbrace{0\dots0}_{\mu} \underbrace{0\dots0}_{\nu}}$	$P_{\underbrace{0\dots1}_{\kappa} s_1^* \dots s_\lambda^* \underbrace{0\dots0}_{\nu}}$ $A_{\underbrace{0\dots1}_{\kappa} s_1^* \dots s_\lambda^* \underbrace{0\dots1}_{\mu}}$ $B_{s_1^* \dots s_\lambda^* \underbrace{0\dots1}_{\mu} \underbrace{0\dots0}_{\nu}}$	…	$P_{\underbrace{E-1\dots E-1}_{\kappa} s_1^* \dots s_\lambda^* \underbrace{0\dots0}_{\nu}}$ $A_{\underbrace{E-1\dots E-1}_{\kappa} s_1^* \dots s_\lambda^* \underbrace{E-1\dots E-1}_{\mu}}$ $B_{s_1^* \dots s_\lambda^* \underbrace{E-1\dots E-1}_{\mu} \underbrace{E-1 0\dots0}_{\nu}}$
$P_{\underbrace{0\dots0}_{\kappa} s_1^* \dots s_\lambda^* \underbrace{0\dots1}_{\nu}}$ $A_{\underbrace{0\dots0}_{\kappa} s_1^* \dots s_\lambda^* \underbrace{0\dots1}_{\mu}}$ $B_{s_1^* \dots s_\lambda^* \underbrace{0\dots1}_{\mu} \underbrace{0\dots1}_{\nu}}$	$P_{\underbrace{0\dots1}_{\kappa} s_1^* \dots s_\lambda^* \underbrace{0\dots1}_{\nu}}$ $A_{\underbrace{0\dots1}_{\kappa} s_1^* \dots s_\lambda^* \underbrace{0\dots2}_{\mu}}$ $B_{s_1^* \dots s_\lambda^* \underbrace{0\dots2}_{\mu} \underbrace{0\dots1}_{\nu}}$	…	$P_{\underbrace{E-1\dots E-1}_{\kappa} s_1^* \dots s_\lambda^* \underbrace{0\dots1}_{\nu}}$ $A_{\underbrace{E-1\dots E-1}_{\kappa} s_1^* \dots s_\lambda^* \underbrace{0\dots0}_{\mu}}$ $B_{s_1^* \dots s_\lambda^* \underbrace{0\dots0}_{\mu} \underbrace{0\dots1}_{\nu}}$
…	…		…

$P_{\underbrace{0\ldots0}_{\kappa}s_1^*\ldots s_\lambda^*\underbrace{E-1\ldots E-1}_{\nu}}$ $A_{\underbrace{0\ldots0}_{\kappa}s_1^*\ldots s_\lambda^*\underbrace{E-1\ldots E-1}_{\mu}}$ $B_{s_1^*\ldots s_\lambda^*\underbrace{E-1\ldots E-1}_{\mu}\underbrace{E-1\ldots E-1}_{\nu}}$	$P_{\underbrace{0\ldots1}_{\kappa}s_1^*\ldots s_\lambda^*\underbrace{E-1\ldots E-1}_{\nu}}$ $A_{\underbrace{0\ldots1}_{\kappa}s_1^*\ldots s_\lambda^*\underbrace{0\ldots0}_{\mu}}$ $B_{s_1^*\ldots s_\lambda^*\underbrace{0\ldots0}_{\mu}\underbrace{E-1\ldots E-1}_{\nu}}$	…	$P_{\underbrace{E-1\ldots E-1}_{\kappa}s_1^*\ldots s_\lambda^*\underbrace{E-1\ldots E-1}_{\nu}}$ $A_{\underbrace{E-1\ldots E-1}_{\kappa}s_1^*\ldots s_\lambda^*\underbrace{E-1\ldots E-1\,E-2}_{\mu}}$ $B_{s_1^*\ldots s_\lambda^*\underbrace{E-1\ldots E-1\,E-2}_{\mu}\underbrace{E-1\ldots E-1}_{\nu}}$

Последовательности произведений блоков матриц *A* и *B*, участвующих вычислениях блоков матрицы *C* на всех *E* итерациях приведены в таблице 3.

Таблица 3. Выполнение действий на итерациях второго этапа алгоритма

Индекс процесса	Итерация	Слагаемое (произведение блоков матриц *A* и *B*)
$\underbrace{0\ldots0}_{\kappa}s_1^*\ldots s_\lambda^*\underbrace{0\ldots0}_{\nu}$	0	$A_{\underbrace{0\ldots0}_{\kappa}s_1^*\ldots s_\lambda^*\underbrace{0\ldots0}_{\mu}} \times B_{s_1^*\ldots s_\lambda^*\underbrace{0\ldots0}_{\mu}\underbrace{0\ldots0}_{\nu}}$
	1	$A_{\underbrace{0\ldots0}_{\kappa}s_1^*\ldots s_\lambda^*\underbrace{0\ldots1}_{\mu}} \times B_{s_1^*\ldots s_\lambda^*\underbrace{0\ldots1}_{\mu}\underbrace{0\ldots0}_{\nu}}$
	…	
	E-1	$A_{\underbrace{0\ldots0}_{\kappa}s_1^*\ldots s_\lambda^*\underbrace{E-1\ldots E-1}_{\mu}} \times B_{s_1^*\ldots s_\lambda^*\underbrace{E-1\ldots E-1}_{\mu}\underbrace{0\ldots0}_{\nu}}$
$\underbrace{0\ldots0}_{\kappa}s_1^*\ldots s_\lambda^*\underbrace{0\ldots1}_{\nu}$	0	$A_{\underbrace{0\ldots0}_{\kappa}s_1^*\ldots s_\lambda^*\underbrace{0\ldots1}_{\mu}} \times B_{s_1^*\ldots s_\lambda^*\underbrace{0\ldots1}_{\mu}\underbrace{0\ldots1}_{\nu}}$
	1	$A_{\underbrace{0\ldots0}_{\kappa}s_1^*\ldots s_\lambda^*\underbrace{0\ldots2}_{\mu}} \times B_{s_1^*\ldots s_\lambda^*\underbrace{0\ldots2}_{\mu}\underbrace{0\ldots1}_{\nu}}$
	…	
	E–1	$A_{\underbrace{0\ldots0}_{\kappa}s_1^*\ldots s_\lambda^*\underbrace{0\ldots0}_{\mu}} \times B_{s_1^*\ldots s_\lambda^*\underbrace{0\ldots0}_{\mu}\underbrace{0\ldots1}_{\nu}}$
…		
$\underbrace{0\ldots0}_{\kappa}s_1^*\ldots s_\lambda^*\underbrace{E-1\ldots E-1}_{\nu}$	0	$A_{\underbrace{0\ldots0}_{\kappa}s_1^*\ldots s_\lambda^*\underbrace{E-1\ldots E-1}_{\mu}} \times B_{s_1^*\ldots s_\lambda^*\underbrace{E-1\ldots E-1}_{\mu}\underbrace{E-1\ldots E-1}_{\nu}}$
	1	$A_{\underbrace{0\ldots0}_{\kappa}s_1^*\ldots s_\lambda^*\underbrace{0\ldots0}_{\mu}} \times B_{s_1^*\ldots s_\lambda^*\underbrace{0\ldots0}_{\mu}\underbrace{E-1\ldots E-1}_{\nu}}$
	…	
	E–1	$A_{\underbrace{0\ldots0}_{\kappa}s_1^*\ldots s_\lambda^*\underbrace{E-1\ldots E-1\,E-2}_{\mu}} \times B_{s_1^*\ldots s_\lambda^*\underbrace{E-1\ldots E-1\,E-2}_{\mu}\underbrace{E-1\ldots E-1\,E-2}_{\nu}}$
$\underbrace{0\ldots1}_{\kappa}s_1^*\ldots s_\lambda^*\underbrace{0\ldots0}_{\nu}$	0	$A_{\underbrace{0\ldots1}_{\kappa}s_1^*\ldots s_\lambda^*\underbrace{0\ldots1}_{\mu}} \times B_{s_1^*\ldots s_\lambda^*\underbrace{0\ldots1}_{\mu}\underbrace{0\ldots0}_{\nu}}$
	1	$\lvert A_{\underbrace{0\ldots1}_{\kappa}s_1^*\ldots s_\lambda^*\underbrace{E-1\ldots E-1\,E-3}_{\mu}} \times B_{s_1^*\ldots s_\lambda^*\underbrace{E-1\ldots E-1\,E-3}_{\mu}\underbrace{0\ldots0}_{\nu}}$
	…	
	E–1	$A_{\underbrace{0\ldots1}_{\kappa}s_1^*\ldots s_\lambda^*\underbrace{0\ldots0}_{\mu}} \times B_{s_1^*\ldots s_\lambda^*\underbrace{0\ldots0}_{\mu}\underbrace{0\ldots0}_{\nu}}$

$\underbrace{0\ldots1}_{\kappa}\overset{*}{s}_1\ldots\overset{*}{s}_\lambda\underbrace{0\ldots1}_{\nu}$	0	$A_{\underbrace{0\ldots1}_{\kappa}\overset{*}{s}_1\ldots\overset{*}{s}_\lambda\underbrace{0\ldots2}_{\mu}} \times B_{\overset{*}{s}_1\ldots\overset{*}{s}_\lambda\underbrace{0\ldots2}_{\mu}\underbrace{0\ldots1}_{\nu}}$
	1	$A_{\underbrace{0\ldots1}_{\kappa}\overset{*}{s}_1\ldots\overset{*}{s}_\lambda\underbrace{0\ldots3}_{\mu}} \times B_{\overset{*}{s}_1\ldots\overset{*}{s}_\lambda\underbrace{0\ldots30\ldots1}_{\mu\ \nu}}$
	E–1	$A_{\underbrace{0\ldots1}_{\kappa}\overset{*}{s}_1\ldots\overset{*}{s}_\lambda\underbrace{0\ldots1}_{\mu}} \times B_{\overset{*}{s}_1\ldots\overset{*}{s}_\lambda\underbrace{0\ldots1}_{\mu}\underbrace{0\ldots1}_{\nu}}$
...		
$\underbrace{0\ldots1}_{\kappa}\overset{*}{s}_1\ldots\overset{*}{s}_\lambda\underbrace{E-1\ldots E-1}_{\nu}$	0	$A_{\underbrace{0\ldots1}_{\kappa}\overset{*}{s}_1\ldots\overset{*}{s}_\lambda\underbrace{0\ldots0}_{\mu}} \times B_{\overset{*}{s}_1\ldots\overset{*}{s}_\lambda\underbrace{0\ldots0}_{\mu}\underbrace{E-1\ldots E-1}_{\nu}}$
	1	$A_{\underbrace{0\ldots1}_{\kappa}\overset{*}{s}_1\ldots\overset{*}{s}_\lambda\underbrace{0\ldots1}_{\mu}} \times B_{\overset{*}{s}_1\ldots\overset{*}{s}_\lambda\underbrace{0\ldots1}_{\mu}\underbrace{E-1\ldots E-1}_{\nu}}$
		...
	E–1	$A_{\underbrace{0\ldots1}_{\kappa}\overset{*}{s}_1\ldots\overset{*}{s}_\lambda\underbrace{E-1\ldots E-1E-1}_{\mu}} \times B_{\overset{*}{s}_1\ldots\overset{*}{s}_\lambda\underbrace{E-1\ldots E-1E-1}_{\mu}\underbrace{E-1\ldots E-1}_{\nu}}$
...		
$\underbrace{E-1\ldots E-1}_{\kappa}\overset{*}{s}_1\ldots\overset{*}{s}_\lambda\underbrace{0\ldots0}_{\nu}$	0	$A_{\underbrace{E-1\ldots E-1}_{\kappa}\overset{*}{s}_1\ldots\overset{*}{s}_\lambda\underbrace{E-1\ldots E-1}_{\mu}} \times B_{\overset{*}{s}_1\ldots\overset{*}{s}_\lambda\underbrace{E-1\ldots E-1E-10\ldots0}_{\mu\ \nu}}$
	1	$A_{\underbrace{E-1\ldots E-1}_{\kappa}\overset{*}{s}_1\ldots\overset{*}{s}_\lambda\underbrace{0\ldots0}_{\mu}} \times B_{\overset{*}{s}_1\ldots\overset{*}{s}_\lambda\underbrace{0\ldots0}_{\mu}\underbrace{0\ldots0}_{\nu}}$
		...
	E–1	$A_{\underbrace{E-1\ldots E-1}_{\kappa}\overset{*}{s}_1\ldots\overset{*}{s}_\lambda\underbrace{E-1\ldots E-1E-2}_{\mu}} \times B_{\overset{*}{s}_1\ldots\overset{*}{s}_\lambda\underbrace{E-1\ldots E-1E-20\ldots0}_{\mu\ \nu}}$
$\underbrace{E-1\ldots E-1}_{\kappa}\overset{*}{s}_1\ldots\overset{*}{s}_\lambda\underbrace{0\ldots1}_{\nu}$	0	$A_{\underbrace{E-1\ldots E-1}_{\kappa}\overset{*}{s}_1\ldots\overset{*}{s}_\lambda\underbrace{0\ldots0}_{\mu}} \times B_{\overset{*}{s}_1\ldots\overset{*}{s}_\lambda\underbrace{0\ldots0}_{\mu}\underbrace{0\ldots1}_{\nu}}$
	1	$A_{\underbrace{E-1\ldots E-1}_{\kappa}\overset{*}{s}_1\ldots\overset{*}{s}_\lambda\underbrace{0\ldots1}_{\mu}} \times B_{\overset{*}{s}_1\ldots\overset{*}{s}_\lambda\underbrace{0\ldots1}_{\mu}\underbrace{0\ldots1}_{\nu}}$
		...
	E–1	$A_{\underbrace{E-1\ldots E-1}_{\kappa}\overset{*}{s}_1\ldots\overset{*}{s}_\lambda\underbrace{E-1\ldots E-1}_{\mu}} \times B_{\overset{*}{s}_1\ldots\overset{*}{s}_\lambda\underbrace{E-1\ldots E-10\ldots1}_{\mu\ \nu}}$
...		
$\underbrace{E-1..E-1}_{\kappa}\overset{*}{s}_1\ldots\overset{*}{s}_\lambda\underbrace{E-1\ldots E-1}_{\nu}$	0	$A_{\underbrace{E-1\ldots E-1}_{\kappa}\overset{*}{s}_1\ldots\overset{*}{s}_\lambda\underbrace{E-1\ldots E-1E-2}_{\mu}} \times B_{\overset{*}{s}_1\ldots\overset{*}{s}_\lambda\underbrace{E-1\ldots E-1E-2}_{\mu}\underbrace{E-1\ldots E-1}_{\nu}}$
	1	$A_{\underbrace{E-1\ldots E-1}_{\kappa}\overset{*}{s}_1\ldots\overset{*}{s}_\lambda\underbrace{E-1\ldots E-1}_{\mu}} \times B_{\overset{*}{s}_1\ldots\overset{*}{s}_\lambda\underbrace{E-1\ldots E-1}_{\mu}\underbrace{E-1\ldots E-1}_{\nu}}$
		...
	E–1	$A_{\underbrace{E-1\ldots E-1}_{\kappa}\overset{*}{s}_1\ldots\overset{*}{s}_\lambda\underbrace{E-1\ldots E-1E-3}_{\mu}} \times B_{\overset{*}{s}_1\ldots\overset{*}{s}_\lambda\underbrace{E-1\ldots E-1E-3}_{\mu}\underbrace{E-1\ldots E-1}_{\nu}}$

Для определения характеристик эффективности предложенного алгоритма можно использовать стандартные методы, описание которых приведено в [7, 8].

Независимость умножений сечений матриц-сомножителей по скоттовым индексам (ориентации $(s_1, \ldots, s_\lambda)$) порождает дополнительный уровень параллелизма, на котором гиперпроцессы $P_1, \ldots, P_s$ (наибольшее значение $s = n_{s_1} \times \ldots \times n_{s_\lambda}$, если все сечения простые) параллельно реализуют параллельные алгоритмы умножения сечений матриц *A* и *B*. Эти сечения имеют меньшую размерность или с меньшее количество элементов, чем матрицы *A* и *B*.

Из сказанного следует, что программно-аппаратный комплекс, реализующий умножение многомерных матриц можно рассматривать как пронумерованную совокупность параллельных подкомплексов (слоев), имеющих архитектуру многомерных решеток, или в идеальном случае торов, каждый из которых реализует умножение одной пары сечений матриц A и B. Если размерность каждого подкомплекса N, то размерность всего комплекса N+1.

На рисунке 1 показана архитектура программно-аппаратного комплекса, который реализует операцию (1, 1)-свернутого произведения трехмерных матриц A_{lsc} и B_{scm}. Очевидно, что слои процессорной решетки, в которых производятся умножения двумерных сечений матриц A_{lsc} и B_{scm} могут иметь топологию – трехмерный тор. Выполнение этого требования желательно потому, что пересылки блоков сечений матриц между процессорами слоя циклические как по "вертикали", так и по "горизонтали". На этом рисунке тороидальные слои обозначены символами TL_i (i=0, …, s–1).

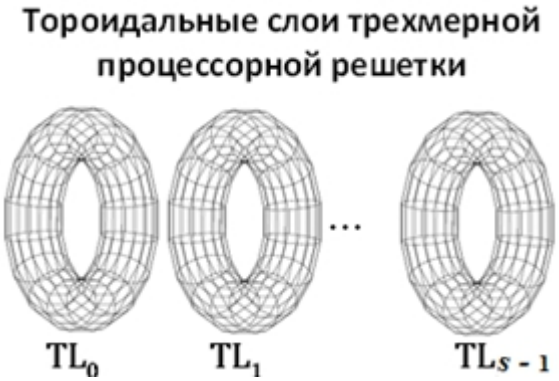


Рис. 1. Программно-аппаратный комплекс для умножения трехмерных матриц

Таким образом, можно утверждать, что предложенный алгоритм умножения многомерных матриц позволяет определить архитектуру программно-аппаратного комплекса, которая позволяет решать проблему массовой параллельной обработки больших объемов структурированных данных в тех случаях, когда эти данные могут быть представлены в виде не разреженных многомерных матриц.

Литература

1. Ефимов С.С. Обзор методов распараллеливания алгоритмов решения некотоых задач вычислительной дискретной математики / Журнал «Математические структуры и моделирование». – Омск: Омск. гос. техн. ун-т, 2007. – Вып. 17. – 125 с.
2. Гантмахер Ф.Р. Теория матриц 2-е изд., доп. – М.: "Наука", 1966. – 576 с.
3. А.В. Малышев Паралеллизация умножения матриц. – Электронное научное периодическое издание "Электроника и информационные технологии", 2(4), 2008 г. – fetmag.mrsu.ru.
4. Robert A. van de Geijn and Jerrell Watts SUMMA: scalable universal matrix multiplication algorithm, Concurrency: Practice and Experience. Volume 9, Issue 4, pages 255–274, April 1997.
5. Choi, J., J. J. Dongarra, and D. W. Walker PUMMA: Parallel Universal Matrix Multiplication Algorithms on distributed memory concurrent computers," Concurrency: Practice and Experience, Vol 6(7), 543-570, 1994.
6. Мунерман В.И. Многомерно-матричная модель массовой обработки данных. – Системы высокой доступности, № 3, 2012, т.8, с. 19-22.
7. Kumar V., Grama A., Gupta A., Karypis G. Introduction to Parallel Computing. Second Edition. Addison Wesley, 2003. 856 с.
8. Roosta, S.H. Parallel Processing and Parallel Algorithms: Theory and Computation. Springer-Verlag, NY. 2000.

Мунерман В.И.[1], Мунерман Д.В.[2]

[1]ФГБОУ ВПО Смоленский государственный университет (СмолГУ), г. Смоленск, к.т.н., доцент, vimoon@gmail.com

[2]ФГБОУ ВПО Смоленский государственный университет (СмолГУ), г. Смоленск, danvmoon@gmail.com

АЛГЕБРАИЧЕСКИЙ ПОДХОД К ПОСТРОЕНИЮ ПРОГРАММНО-АППАРАТНЫХ КОМПЛЕКСОВ ДЛЯ ПОВЫШЕНИЯ ЭФФЕКТИВНОСТИ МАССОВОЙ ОБРАБОТКИ ДАННЫХ

КЛЮЧЕВЫЕ СЛОВА

Базы данных, параллельное программирование, архитектуры вычислительных комплексов.

АННОТАЦИЯ

В статье рассматривается проблема оптимизации массовой обработки структурированных больших данных. Обработка такого типа данных часто встречается во многих информационных системах и имеет высокую вычислительную сложность. Для решения проблемы повышения эффективности массовой обработки данных предлагается подход, основанный на промежуточных моделях данных, и рассматриваются SQL-реляционная и файловая модели данных. Показано, что на основе этих моделей можно проектировать программно-аппаратные комплексы, ориентированные на эффективное решение конкретных задач в информационных системах.

Работа проводилась в рамках грантов, предоставленных Смоленскому государственному университету корпорацией Microsoft.

В статье рассматривается проблема оптимизации массовой обработки структурированных больших данных (МОД). Традиционно такой вид обработки данных широко используется для решения многих задач в различных предметных областях в тех случаях, когда в вычисления включается значительная часть содержащихся в базе данных. К числу таких в задач экономике относятся: оперативная статистическая обработка данных, задача "Завершение операционного дня банка", ежедневные задачи учета и планирования производства в современных системах управления, задачи статистического анализа и синтеза подсистем послепродажного обслуживания в системах интегрированной логистической поддержки наукоемкой продукции.

Современное состояние вычислительной техники и возможности систем программирования позволяют проводить оптимизацию рассматриваемого вида обработки на различных уровнях и различными методами. Рассматриваемая в статье технология МОД основана на сочетании файловой (теоретико-множественной) модели данных [1, 2, 3], связывающей исходную модель данных и модель вычислений, и SQL-реляционной модели. Между этими двумя моделями может быть установлено, в зависимости от требований решаемой задачи, как гомоморфное, так и изоморфное соответствие. В статье рассматривается параллельная реализация алгебраических выражений запросов, представляющих собой цепочки операций слияния нестрого упорядоченных файлов. В SQL-реляционной модели этой операции соответствует операция соединения JOIN.

Строго упорядоченным файлам X_L и Y_M соответствуют таблицы P и R во второй или третьей нормальной форме по составным ключам, соответствующим множествам ключей L и M. Операции слияния нестрого упорядоченных файлов X_L и Y_M по множеству ключей K, которому соответствует простой или составной ключ K, в таблицах P и R, соответствует операция P INNER JOIN R ON π ($P.K$, $R.K$). На практике операция JOIN реализуется при следующих условиях, которые соответствуют требованиям к операции слияния нестрого упорядоченных файлов:

- на множестве экземпляров ключа K задано отношение эквивалентности;
- таблицы P и R имеют разные схемы, в каждую из которых входит ключ K;
- предикат π определен на множествах экземпляров ключа K в таблицах P и R и имеет

вид $P.K = R.K$.

В общем случае, файлы X_L и Y_M нестрого упорядочены, тогда соответствующие им таблицы $P(K, \ldots)$ и $R(K, \ldots)$ будут в первой нормальной форме. В этом случае, таблицы, как и файлы, состоят из классов эквивалентности, в которых все строки содержат один и тот же экземпляр ключа K. Тогда для параллельной реализации операции JOIN можно метаданные, определяющие распределение строк таблицы по классам эквивалентности, основанное, как и в случае файловой модели данных, на индексно-последовательном методе доступа (ISAM).

Пусть $K_1, \ldots, K_n$ – множество экземпляров ключа K. Таблицы $P(K, \ldots)$ и $R(K, \ldots)$ имеют схемы, которые содержат ключ K и произвольные наборы полей (как правило, различные). Таблице P можно поставить в соответствие индексную таблицу со схемой $indP(K, I, M)$. Эта таблица – есть множество строк $ind\,P=\{\langle K_1, i_1, m_1\rangle, \ldots, \langle K_n, i_n, m_n\rangle\}$. Здесь i_j – индекс первой записи класса эквивалентности, строки которого содержат экземпляр K_j ключа K, m_j – количество записей в этом классе эквивалентности. Поле I имеет тип Счетчик строк, а поле M – целочисленное. Таблица $indP$ может быть получена в результате двух запросов:

1. $Q1$=INSERT INTO *indPt* (*K*, *I*, *M*)
   SELECT *P.K*, 1 AS *M* FROM *P* ORDER BY *P.K*;
2. $Q2$=SELECT *indPt.K*, First(*indPt.I*) AS *I*, Sum(*IndPt.M*) AS *M*
   INTO *indP* FROM *indPt*
   GROUP BY *indPt.K* ORDER BY *indPt.K*

Запрос $Q1$ формирует промежуточную таблицу *indPt*, которая содержит столько же строк, что и таблица *P*, схема этой таблицы состоит из трех полей: ключ *K*, номер строки и поле *M*, содержащее значение 1. Запрос $Q2$ завершает построение таблицы *indP* посредством операции группировки. Этому двухпроходному алгоритму соответствует простой однопроходный алгоритм, который легко разработать на любом языке программирования, связанным с СУБД, или на языках манипулирования данными, такими как Tranzact-SQL или PL-SQL. Листинг 1 демонстрирует реализацию однопроходного алгоритма построения индексного файла для индексно-последовательного доступа к таблице *P*.

Листинг 1. Построение индексного файла средствами языка Tranzact-SQL

```
DECLARE @K <тип>, @Kcurrent <тип>, @I int, @M int, @RowCount int, @j int
DELETE FROM indP
DECLARE cursorP CURSOR FOR SELECT K FROM P ORDER BY K
OPEN cursorP
SET @RowCount =@@Cursor_Rows
SET @j=0
FETCH NEXT FROM cursorP INTO @K
SET @Kcurrent =@K
SET @I=1
SET @M=1
WHILE @j<@ RowCount π1
BEGIN
        FETCH NEXT FROM cursorP INTO @K
        IF @Kcurrent =@K
        BEGIN
                SET @M=@M+1
        END
        ELSE
        BEGIN
                INSERT INTO indP (K, I, M) VALUES (@Kcurrent, @I, @M)
                SET @Kcurrent =@K
                SET @I=@I+@M
                SET @M=1
        END
        SET @j=@j+1
END
```

Таблица *indR* для таблицы *R* может быть получена аналогично.

Очевидно, что таблица-результат операции JOIN будет содержать только те классы

эквивалентности, которые принадлежат пересечению индексных таблиц *indPy∩indR*. Запрос, в результате которого получается это пересечение, имеет вид:

```
SELECT indP.K, indP.M*indR.M AS MM, indP.I, indR.I,
        indP. M, indR. M
  INTO ComInd FROM indP, indR
  WHERE indP.K= indR.K
```

Результат запроса – таблица *ComInd*. Она содержит составной ключ *K*, поля *I* и *M* обеих индексных таблиц, а также поле *MM*. Значение поля *MM* вычисляется как произведение количества строк в обоих классах эквивалентности, строки которых содержат одно и то же значение составного ключа *K*. Это значение определяет число выходных строк, которое получится в результате обработки этих классов эквивалентности операцией JOIN.

Использование таблицы *ComInd* позволяет распределить таблицы *P* и *R* между несколькими независимыми базами данных. Если имеется *N* таких баз данных, то эти таблицы разделяются на *N* фрагментов. Каждая база данных $DBSnp_i$ содержит пару фрагментов $<P_i, R_i>$ таблиц *P* и *R*. При этом классы эквивалентности, записи которых содержат одинаковые значения ключа *K*, полностью расположены в одном и только одном фрагменте. Такой способ распределения таблиц называется принципом симметричного горизонтального распределения данных. Для реализации симметричного горизонтального распределения таблиц *P* и *Q* используется алгоритм бустрофедона, который в простейшем случае может быть реализован так, как показано в листинге 2. Для простоты таблица *ComInd* читается одновременно от начала до середины и от конца до середины.

Листинг 2. Реализация симметричного горизонтального распределения средствами языка Tranzact-SQL

```
DECLARE @KA <тип>, @KD <тип>, @RowCount int, @i int
DECLARE cursorASC CURSOR FOR SELECT K, MM FROM ComInd
ORDER BY MM
DECLARE cursorDSC CURSOR FOR SELECT K, MM FROM ComInd
ORDER BY MM DESC
OPEN cursorASC
OPEN cursorDSC
SET @ RowCount =@@Cursor_Rows
IF @ RowCount %2=0
        SET @RowCount =@RowCount /2
ELSE
        SET @RowCount =@RowCount /2+1
SET @i=0
WHILE @i<@RowCount
BEGIN
  FETCH NEXT FROM cursorASC INTO @ KA
  FETCH NEXT FROM cursorDSC INTO @ KD
   IF @i % N=0
   BEGIN
        INSERT INTO PSnp1 SELECT * FROM P WHERE K=@KA
        INSERT INTO PSnp1 SELECT * FROM P WHERE K=@KD
        INSERT INTO RSnp1 SELECT * FROM R WHERE K=@KA
        INSERT INTO RSnp1 SELECT * FROM R WHERE K=@KD
   END
        …
   IF @i % N=N-1
   BEGIN
        INSERT INTO PSnpN SELECT * FROM P WHERE K=@KA
        INSERT INTO PSnpN SELECT * FROM P WHERE K=@KD
        INSERT INTO RSnpN SELECT * FROM R WHERE K=@KA
        INSERT INTO RSnpN SELECT * FROM R WHERE K=@KD
   END
        SET @i=@i+1
END
```

После выполнения этого алгоритма количества строк таблицы-результата, получаемые в

результате операции JOIN над фрагментами P_i и R_i таблиц P и R, будут минимально отличаться друг от друга. Симметричное горизонтальное распределение фрагментов таблиц P и R между базами данных $DBSnp_i$ позволяет проектировать программно-аппаратные комплексы, позволяющие повысить эффективность МОД за счет распараллеливания операции JOIN.

Программно-аппаратная реализация МОД. Рассматриваемому варианту МОД присущи регулярные запросы. Такие запросы разрабатываются прикладным программистом в процессе проектирования автоматизированной информационной системы, после чего существуют практически без изменений на протяжении всего жизненного цикла системы. Следовательно, затраты на трансляцию запроса, его оптимизацию, создание библиотечной программы, аналогичной хранимой процедуре, имеют разовый характер и не наносят ущерба как разработчику – профессиональному прикладному программисту, – так и самой информационной системе во время ее эксплуатации.

Особенность современного состояния вычислительной техники состоит в том, что, во-первых, используются гибкие архитектуры современных аппаратных средств, основанные на многоядерности и многопроцессорности, и, во-вторых, вычислительные сети обеспечивают такую простоту коммуникаций, которая позволяет легко проектировать различные топологии сетей. Эти два фактора – регулярность запросов и гибкость вычислительных средств – позволяют решать задачи оптимизации архитектуры программно-аппаратного вычислительного комплекса на этапе разработки автоматизированной информационной системы. Причем для каждой системы, входящей в ее состав задачи или даже отдельного запроса может быть разработан индивидуальный программно-аппаратный вычислительный комплекс.

Таким образом, в задачах МОД можно выделить две основные цели оптимизации: ускорение отдельных операций обработки одного или нескольких файлов, и построение оптимальных последовательностей операций (процессов или запросов) для решения конкретных задач.

На основе файловой модели становится возможной оптимизация отдельных операций за счет применения различных параллельных архитектур вычислительных комплексов, реализующих выполнение операций. То есть, она позволяет производить двухуровневую оптимизацию, как на уровне операций, так и на уровне процессов. Наибольшую сложность при решении проблемы параллельной обработки больших объемов данных имеют операция слияния нестрого упорядоченных файлов и процессы, состоящие из последовательностей таких операций.

Далее рассматривается построение программно-аппаратного комплекса, который может эффективно решать различные задачи МОД. В терминах файловой модели данных алгебраической моделью процесса МОД служит алгебраическое выражение вида $A = E(A_1, ..., A_p)$, правая часть которого состоит из файлов $A_1, ..., A_p$, соединенных знаками операций над файлами, а левая – из выходного файла. Далее рассматриваются только аддитивная операция слияния строго упорядоченных файлов и мультипликативная операция слияния нестрого упорядоченных файлов. Тогда правая часть алгебраического выражения $E(A_1, ..., A_p)$ может быть сведена к "сумме произведений" исходных файлов. Как было сказано, в реляционной модели данных операции слияния нестрого упорядоченных файлов соответствует операция JOIN. Далее рассмотрена архитектура вычислительного комплекса для параллельного вычисления цепочки "произведений" файлов, то есть, в терминах SQL-реляционной модели данных, вычисления выражения запроса вида:

SELECT <список полей> FROM
(A_p INNER JOIN SELECT <список полей> FROM
(A_{p-1} INNER JOIN ...
SELECT <список полей> FROM
(A_2 INNER JOIN A_1 ON π_1) ...) ON π_{p-1})

Для вычисления этого запроса может быть эффективно использована конвейерная (MISD) архитектура вычислительного комплекса [11, 12] (рисунок 1).

Этот комплекс состоит из p-1 *Join*-процессора, первый из которых начинает обработку первых двух "сомножителей". Как только готов очередной фрагмент (один или несколько классов эквивалентности) результата, он передается следующему *Join*-процессору, который обрабатывает этот фрагмент с соответствующим классом эквивалентности файла A_3, и так далее по конвейеру. Последний *Join*-процессор конвейера принимает классы эквивалентности, полученные в результате предыдущих произведений и обрабатывает их с соответствующими классами эквивалентности последней таблицы в цепочке.

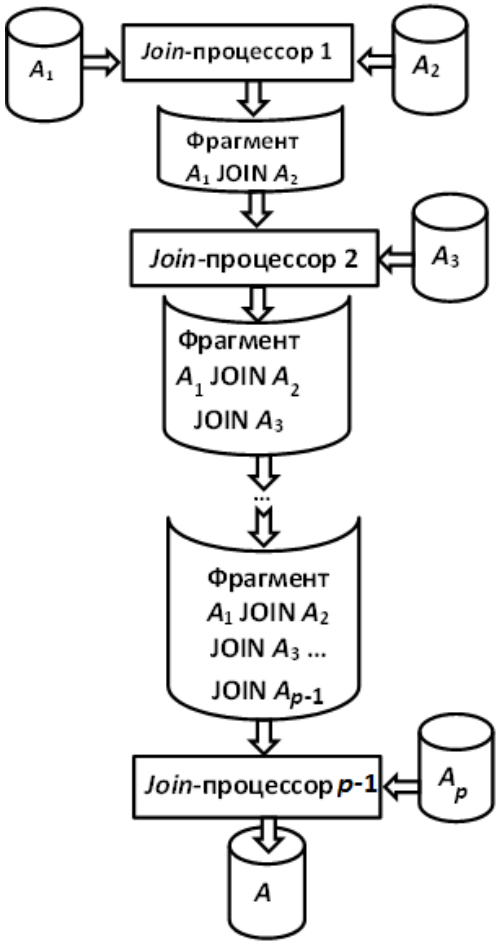


Рис.1. Конвейерный вычислительный комплекс для вычисления цепочки операций Join

Среди операций МОД операция JOIN имеет наибольшую вычислительную сложность. На основе файловой модели *Join*-процессор можно организовать, как вычислительный комплекс на основе SIMD-архитектуры (рисунок 2).

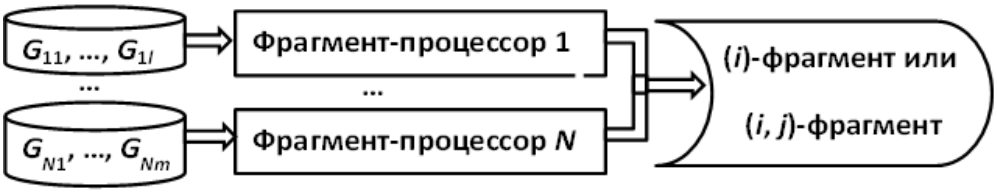


Рис. 2. SIMD-архитектура Join-процессора

Фрагменты таблиц-операндов, обрабатываемых *i*-тым фрагмент-процессором, можно рассматривать как совокупность пар G_{i1}, ..., G_{ik}, каждая из которых содержит два соответствующих друг другу класса эквивалентности таблиц-операндов. Размещение данных в памяти фрагмент-процессоров производится на основе принципа симметричного горизонтального распределения.

Фрагмент-процессор выполняет операцию JOIN над фрагментами файлов и передает на выход либо весь результат операции, либо фрагмент, содержащий результат декартова произведения классов эквивалентности очередной пары G_{ij}.

При таком сочетании конвейерной (MISD) и SIMD-архитектур вычислительных комплексов [4, 5] можно добиться значительного повышения эффективности процессов МОД.

Для проверки предложенных методов повышения эффективности МОД проведен ряд вычислительных экспериментов [6, 7, 8, 9], в которых на основе теоретико-множественной и многомерно-матричной моделей были построены программно-аппаратные комплексы с различными архитектурами:

- симметричное мультипроцессирование (SMP) на основе рабочей станции с четырехядерным процессором Intel Core i7 (с технологией Hyper Threading);
- массово-параллельная архитектура на основе локальной вычислительной сети с однотипными рабочими станциями;
- массово-параллельная архитектура на основе облачной системы Windows Azure с распределением вычислений между виртуальными машинами и отдельными базами данных.

В результате применения принципа симметричного горизонтального распределения данных на виртуальных вычислительных комплексах с предложенной архитектурой было получено от 6-кратного до 20-кратного ускорение решения задач МОД (в зависимости от количества используемых вычислительных ресурсов).

Из сказанного можно сделать вывод о том, что применение файловой (теоретико-множественной модели) в качестве средства, связывающего высокоуровневую SQL-реляционную модель данных с архитектурами программно-аппаратных вычислительных комплексов, позволит существенно повысить эффективность массовой обработки структурированных данных. Кроме того, предложенные методы параллельной реализации задач массовой обработки данных открывают следующие возможности:

1. Создание на основе гибких архитектур современных средств вычислительной техники виртуальных программно-аппаратных комплексов индивидуально для каждой конкретной задачи массовой обработки интенсивно используемых данных, наилучшим образом приспособленных для реализации алгоритмов решения именно этой задачи, и учитывающих характер обрабатываемых данных. Эти комплексы могут быть построены на основе серийных вычислительных средств, что существенно снижает уровень материальных затрат;
2. Использование для разработки и реализации программного обеспечения стандартных систем программирования и СУБД. При этом разработка не требует высокого уровня искусства программирования, так как основана на достаточно простой технологии программирования. Использование любого программного обеспечения, уже приобретенного или открытого, также снижает материальные затраты, необходимые для решения задач МОД;
3. Сокращение времени решения задач повышает степень актуальности информации, что особенно важно при работе с большими данными.

Литература

1. Левин Н.А., Мунерман В.И. Модели обработки больших объемов данных в системах массового параллелизма. – Системы высокой доступности, № 1, 2013, т.9, с. 35-43.
2. Левин Н.А., Мунерман В.И. Реализация объектно-ориентированной модели массовой обработки данных. – Системы высокой доступности, № 3, 2012, т.8, с. 23-25.
3. Мунерман В.И. Объектно-ориентированная модель массовой обработки данных. – Системы высокой доступности, № 4, 2011, т.7, с. 72-74.
4. Flynn M.J. Computer Organization and Architecture // Operating Systems, An Advanced Course. – Springer, 1978 (Lecture Notes in Computer Science; Vol.60). – P. 17-98.
5. Flynn M.J. Rudd K. W. Parallel architectures // ACM Computing Surveys. 1996. – Vol. 28, No. 1. – P. 67-70.
6. Макаров Д.И., Мунерман В.И. Параллельная реализации операции соединения для массовой обработки данных. – Системы высокой доступности, № 3, 2012, т.8, с. 26-28.
7. Мунерман В.И. Построение архитектур программно-аппаратных комплексов для повышения эффективности массовой обработки данных. – Системы высокой доступности, № 4, 2014, т.10, с. 3-16.
8. Мунерман В.И. Опыт массовой обработки данных в облачных системах (на примере Windows Azure). – Системы высокой доступности, № 2, 2014, т.9, с. 3-8.
9. Мунерман В.И. Реализация обработки больших объемов данных на симметричных мультипроцессорных системах. – Системы высокой доступности, № 2, 2013, т.9, с. 36-9.

Орлов Ю.В.[1], Посыпкин М.А.[2]

[1]АО ВНИИРТ, 105082, г. Москва, ул. Б.Почтовая, 22, выпускник бакалавриата факультета ВМК МГУ имени М.В. Ломоносова, инженер 2-й категории, justice1786@gmail.com

[2] Институт проблем передачи информации РАН, 117218, г. Москва, Нахимовский пр-т, д.36, корп. 1, ведущий научный сотрудник, д.ф.-м.н., mposypkin@gmail.com

СРЕДА КОМПЛЕКСНОГО АНАЛИЗА ПРОИЗВОДИТЕЛЬНОСТИ АЛГОРИТМОВ БАЛАНСИРОВКИ В ПАРАЛЛЕЛЬНОМ МЕТОДЕ ВЕТВЕЙ И ГРАНИЦ

КЛЮЧЕВЫЕ СЛОВА

Метод ветвей и границ; параллельные и распределенные вычисления; анализ производительности алгоритмов.

АННОТАЦИЯ

В работе описывается среда комплексного анализа производительности алгоритмов балансировки вычислительной нагрузки в параллельном методе ветвей и границ. Основное назначение разработанной среды – поддержка исследования производительности алгоритмов балансировки и выяснение причин ее снижения. Рассмотрены основные подходы к визуализации производительности, продемонстрированы их достоинства и недостатки. Приведена общая схема и принципы работы разработанного программного инструментария.

Введение

Многие задачи глобальной оптимизации относятся к классу NP и их решение требует значительных вычислительных ресурсов. Поэтому представляется целесообразным применение методов параллельных [1,2] и распределенных вычислений [3]. Метод ветвей и границ (МВГ) является одним из наиболее распространенных алгоритмов решения задач дискретной оптимизации. В его основе лежит идея декомпозиции, которая делает естественным применение параллельных вычислений. Этим фактом можно объяснить значительный интерес исследователей к вопросам параллельной реализации МВГ. Обзор различных подходов можно найти в работах [4-6]. Основной проблемой при параллельной реализации методов типа ветвей и границ является адекватная балансировка вычислительной нагрузки между параллельными процессорами. Так как информационный граф алгоритма [7] в данном случае представляющий собой дерево, заранее не известен, особую важность получают методы динамической балансировки, перераспределяющие вычисления в процессе работы в зависимости от загрузки процессоров.

В ближайшее время ожидается, что суперкомпьютеры обретут производительность порядка экзафлопс (около 10^{18} операций с плавающей точкой в секунду). Основным способом увеличения производительности является наращивание количества ядер, число которых в некоторых современных системах [8] уже превосходит 10^{16}. В такой ситуации балансировка нагрузки становится весьма нетривиальной задачей. Следовательно, необходим развитый инструментарий для анализа производительности алгоритмов балансировки.

Одной из важнейших задач комплексного анализа производительности таких алгоритмов является выявление причин потерь производительности. Часто без визуализации, используя одну лишь трассу, сделать это практически невозможно, так как трассы выполнения параллельных программ обычно очень велики по объему и слабо поддаются визуальному анализу.

Для наиболее полного анализа производительности параллельных алгоритмов оптимизации необходимо иметь возможность:

- осуществлять визуализацию работы многопроцессорной системы в динамике;
- осуществлять визуализацию коммуникаций между процессами;
- вычислять ускорение и эффективность системы в целом.

Существующие решения

На сегодняшний день уже существуют визуальные программные комплексы, позволяющие анализировать производительность процессоров и ядер на современной вычислительной технике. К таким системам можно отнести: Ganglia[9], Nagios[10], NVIDIA Visual Profiler[11], PGPROF[12] и другие. Однако данные программные продукты являются средствами общего назначения. Они не позволяют осуществлять визуализацию низкоуровневых коммуникаций между процессами. Поэтому, проблема разработки и реализации визуальной среды для анализа производительности параллельных алгоритмов оптимизации остается актуальной.

Основные принципы работы среды

Важнейшей составляющей разработки алгоритмов управления распределением вычислений в многопроцессорной системе является отладка, позволяющая выявить слабые стороны тестируемого алгоритма. Отладка на реальной многопроцессорной системе дает наиболее полную информацию о производительности алгоритма. Но такая отладка весьма трудоемка, так как требует большого числа запусков параллельного приложения.

Для того, чтобы проводить тестирование алгоритмов балансировки без проведения ресурсоемких вычислений, был разработан симулятор многопроцессорной системы BNB-Simulator[13] на базе компонентов библиотеки BNB-Solver[14,15]. Данная библиотека предлагает набор модулей для разработки параллельных приложений. Симулятор имитирует выполнение реального параллельного приложения, а его трасса имеет тот же формат, что и трасса параллельного приложения, разработанного на основе BNB-Solver. Таким образом, симулятор позволяет отлаживать алгоритм, заложенный в программе, который затем можно использовать как отдельный модуль параллельного приложения, запускаемого на многопроцессорной системе.

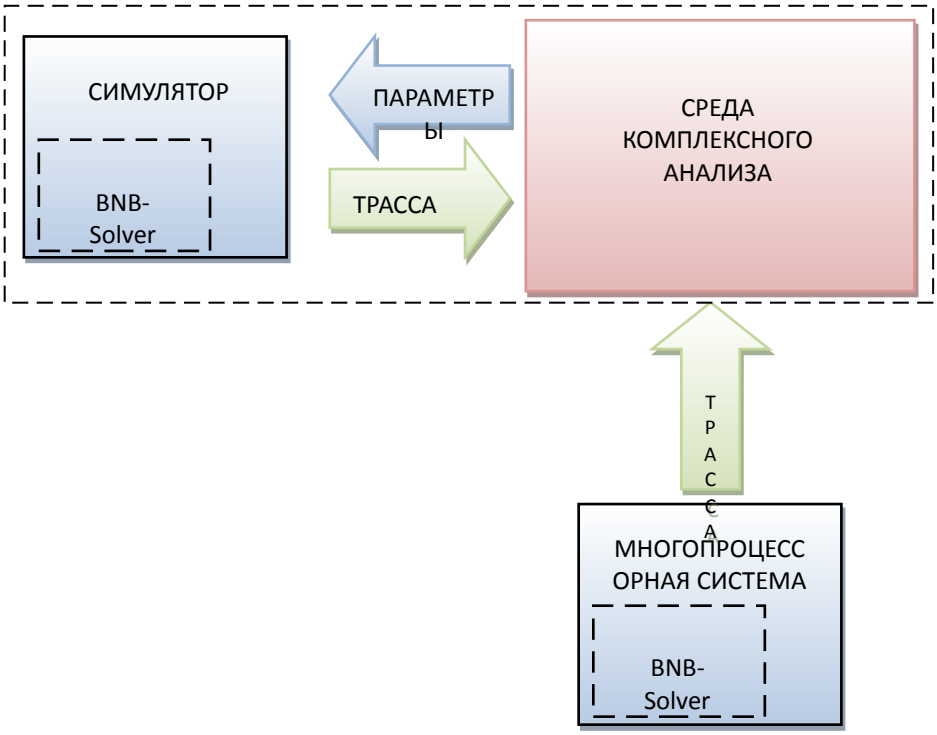


Рис. 1. Общая схема работы визуальной среды

Симулятор представляет собой приложение, управляемое через файл настроек settings.json. Это в свою очередь позволяет непосредственно через графический интерфейс визуальной среды:

- запускать симулятор, передавая ему параметры и получая от него в ответ трассу в виде потока байт;
- изменять настройки симулятора.

Также среда комплексного анализа позволяет загружать трассу из файла с расширением «.trc». Таким образом, визуальная среда позволяет работать с трассой установленного формата, независимо от источника ее получения (реального приложения или симулятора).

Разработанная среда предоставляет следующую функциональность:

- загрузка и разбор трассы;

- визуализация графиков работы процессоров;
- визуализация графиков таблицы процессоров;
- визуализация обмена данными между процессами;
- расчет и вывод статистики.

Трасса содержит информацию о низкоуровневых командах и событиях каждого из процессов. Разбор трассы происходит в несколько этапов. Сначала из полученных текстовых данных формируется список строк. Далее с помощью специального метода разбора из списка строк формируется внутреннее представление трассы, состоящие из списка объектов, содержащих информацию об обмене данными, и объектов, содержащих информацию о каждом из процессов в каждый момент времени работы многопроцессорной системы. Далее окно компонентов управления использует внутреннее представление для извлечения и передачи информации окнам визуализации. Также данные объекты используются средой для формирования статистики.

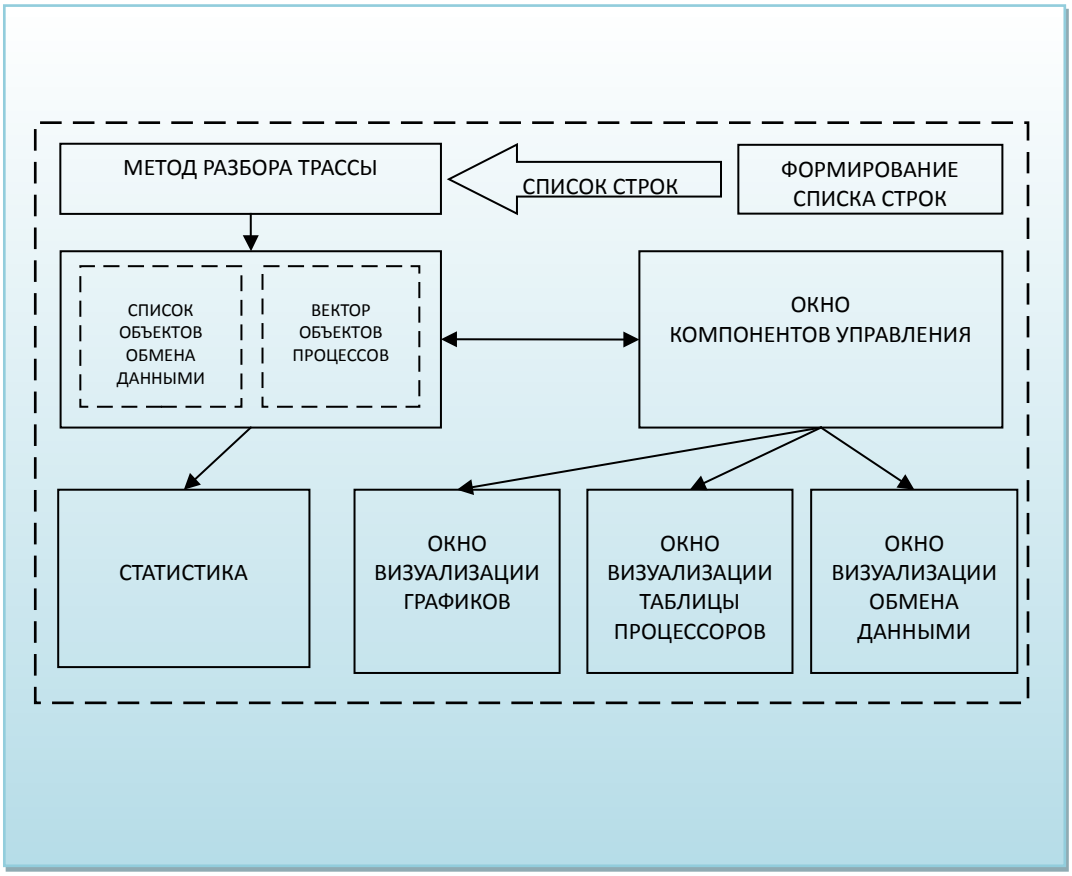


Рис.2. Схема отображения трассы

Основные компоненты визуальной среды

Окно компонентов управления (рис. 3) представляет собой интерфейс, напоминающий мультимедийный проигрыватель. Данное окно содержит полосу прокрутки, максимальное значение которой равно максимальному значению логического времени работы системы, кнопки для управления прокруткой («Rewind», «Play», «Pause», «Stop», «Forward») и слайдер для регулирования скорости воспроизведения действий процессоров. При изменении состояния полосы прокрутки обновляется открытое окно визуализации. Такой способ организации управления визуализацией позволяет абстрагироваться от времени работы многоядерной системы, с целью наиболее детального исследования поведения каждого процессора и системы в целом.

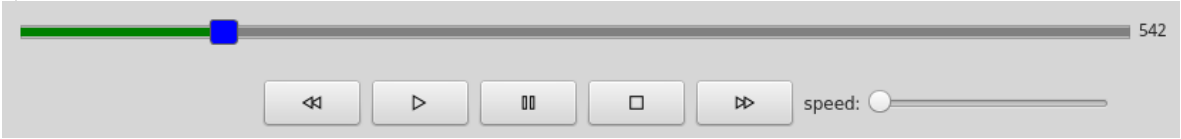


Рис.3. Окно компонентов управления

Для более детального исследования поведения процессов на определенном промежутке времени удобно использовать окно визуализации графиков производительности (рис. 4). В данном окне поведение каждого процесса представлено в виде набора графиков на координатной плоскости, отображающими состояние процесса в конкретный момент времени. Красным цветом отображается состояние ожидания, синим – состояние счета, зеленым – состояние отправки данных другому процессу.

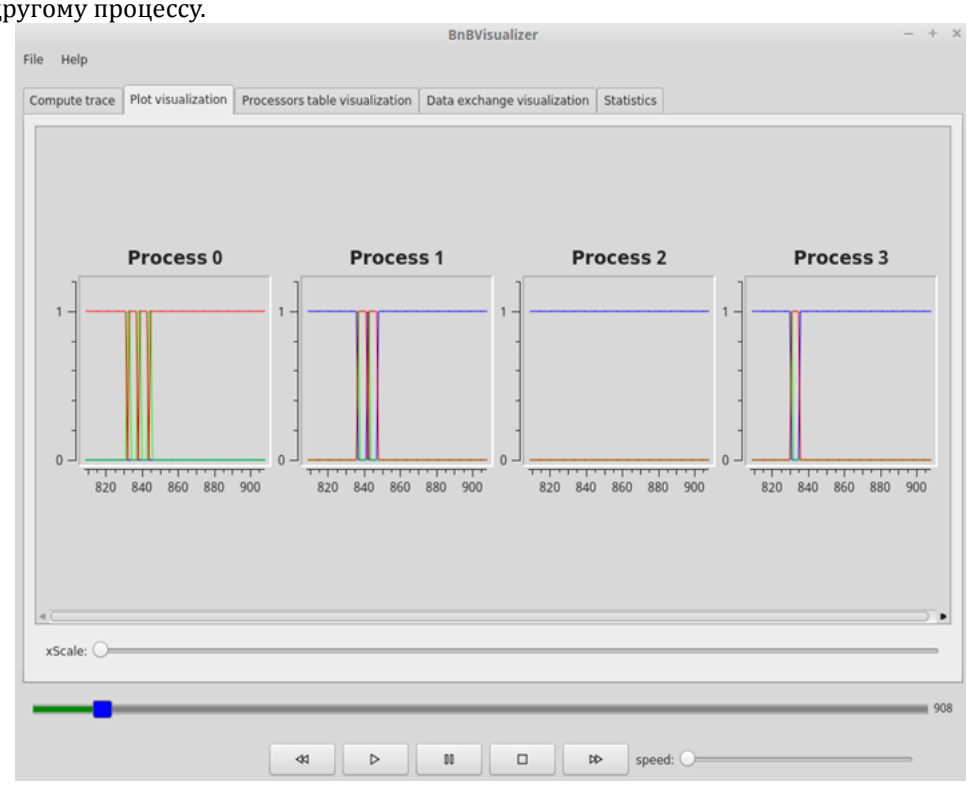


Рис.4. Визуализация графиков загруженности процессоров

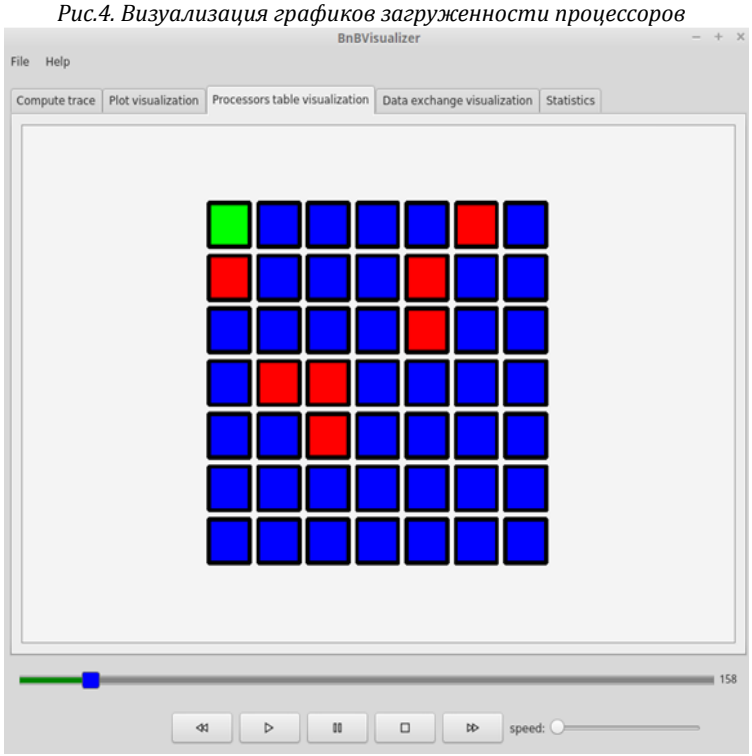


Рис.5. Визуализация таблицы загруженности процессоров

Такой подход удобен при небольшом (не более 10) числе процессоров. Если запуск параллельного приложения осуществляется на довольно большом количестве узлов, удобнее

использовать окно визуализации таблицы процессоров (рис. 5). Каждый процесс в данном окне изображен в виде квадрата. В зависимости от своего состояния квадрат меняет цвет. Такой способ визуализации не позволяет нам исследовать поведение процесса на отрезке времени, зато дает возможность видеть состояние всей системы в каждый момент времени ее работы. Чтобы узнать номер процесса, соответствующий квадрату, достаточно навести на квадрат курсор мыши, номер отобразится во всплывающей подсказке.

Для полноценного анализа параллельного алгоритма оптимизации одной лишь информации о состоянии процессов недостаточно. Часто при отладке алгоритма по трассе нужно обладать информацией о коммуникации процессов. С этой целью в среду комплексного анализа было добавлено окно визуализации обмена данными (рис. 6).

В данном окне каждому процессу соответствует два квадрата меняющих цвет в зависимости от состояния процесса. Квадраты выстраиваются в два ряда, перпендикулярных по отношению друг к другу, от левого нижнего угла, что позволяет изображать процесс передачи сообщений при помощи цветных линий. Квадраты из горизонтального ряда соответствуют процессорам-отправителям, а квадраты из вертикального ряда – процессорам-получателям. Зеленая линия, исходящая вверх от отправителя, символизирует отправку сообщения, синяя линяя, исходящая вправо от получателя, обозначает получение сообщения.

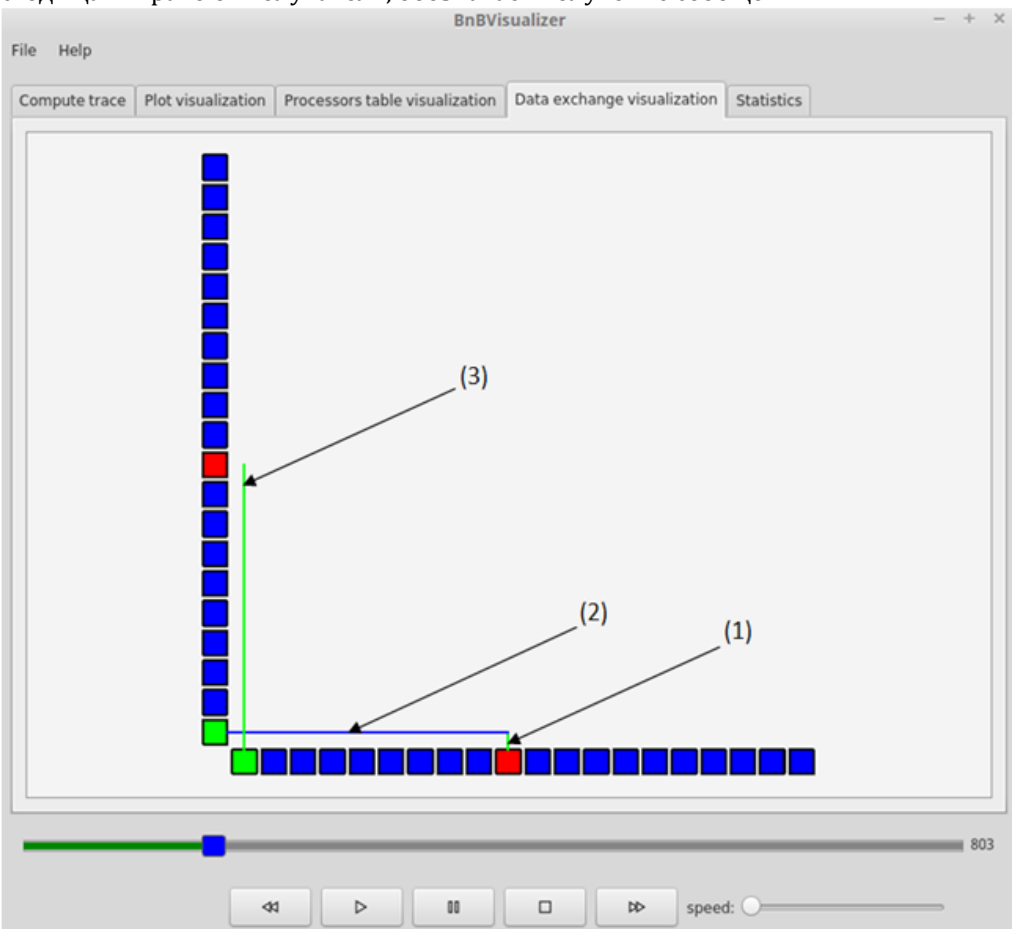


Рис.6. Визуализация обмена данными. (1) – процесс с номером 9 отправил данные процессу с номером 0 и находится в состоянии ожидания; (2) – процесс с номером 0 получил отправленные ему процессом 9 данные; (3) – процесс с номером 0 отправил новые данные процессу 9

Такая визуализация дает возможность выявлять логические ошибки в алгоритме балансировки нагрузки. Так, например, можно выявлять тупиковые ситуации по трассе, используя простейший алгоритм, приведенный на рисунке 7. Если все процессы находятся в ожидании на последней метке времени, и при этом не зафиксировано ни одной передачи между ними, можно сделать вывод, что один или несколько процессов захватили ресурсы и по какой-то причине не передали их другим процессам, ожидающим данные ресурсы.

На рисунке 8 (справа) видно, как процесс с номером 3 (начиная с 0) находится в состоянии

ожидания, отправив перед этим данные процессу с номером 0. Процесс с номером 0 сразу же получает от него результаты и направляет новую порцию данных для счета. На том же рисунке (слева) видна ситуация, когда все три рабочих процесса завершили счет и ушли в ожидание, не отправив перед этим результаты главному процессу, ожидающему эти данные.

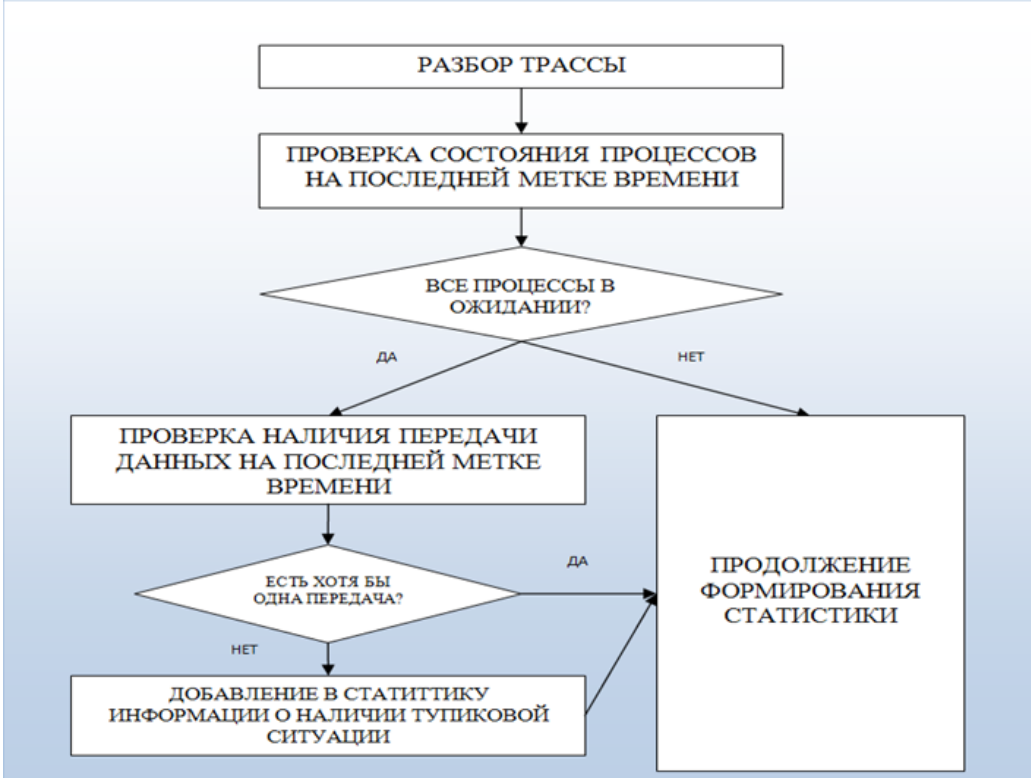


Рис.7. Схема алгоритма обнаружения тупиковых ситуаций

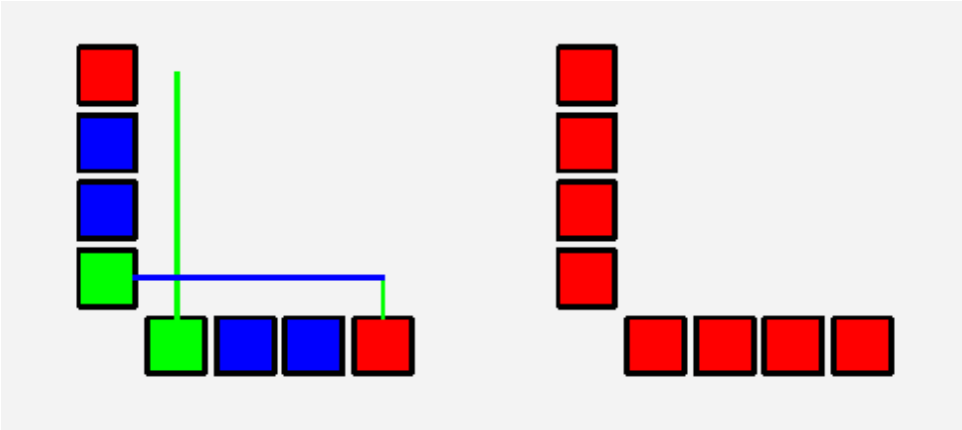


Рис.8. Демонстрация обмена данными между процессами (слева) и отсутствия такого обмена между процессами, находящимися в ожидании (справа)

Для проведения сравнительного анализа работы параллельных программ на основе различных алгоритмов балансировки важными показателями являются такие метрики как: параллельная эффективность *E* и параллельное ускорение *S* [16].

$$S=\frac{t_{посл}}{t_{пар}}\ ;\ E=\frac{S}{n}\ .$$

Среда комплексного анализа автоматически формирует статистику для всей системы и по каждому отдельному процессу. Статистика состоит из графиков соотношения пиковой и реальной производительности, информации о выявленных логических ошибках (выводится, если среде комплексного анализа удается автоматически их обнаружить), информации об ускорении и

эффективности вычислительной системы и информации по каждому из процессов.

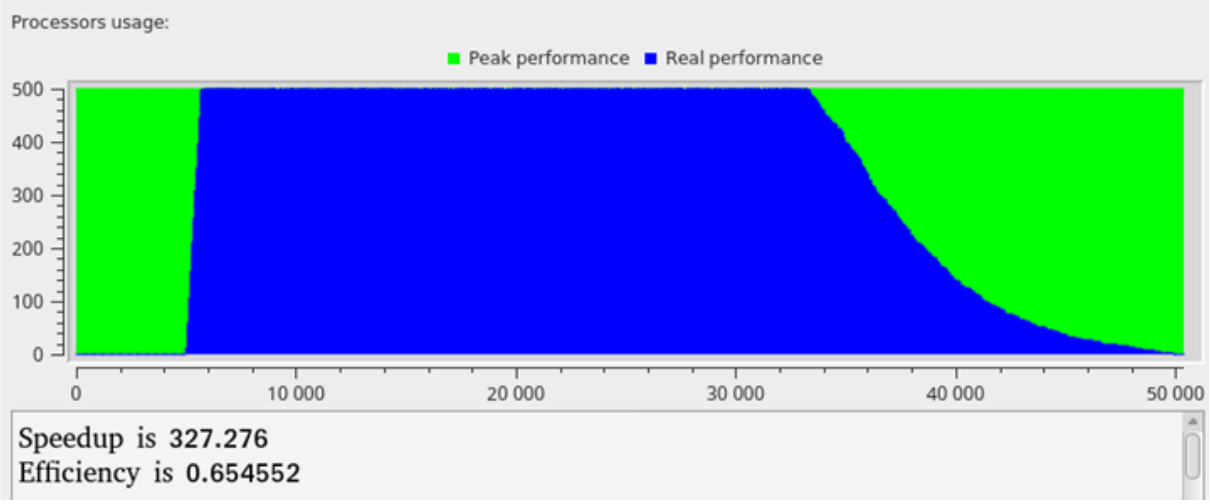


Рис.9. Статистика по результатам работы вычислительной системы

Заключение

В данной работе рассмотрены программная реализация и основной базовый функционал среды комплексного анализа производительности алгоритмов балансировки в параллельном методе ветвей и границ.

Среда комплексного анализа производительности параллельных алгоритмов оптимизации является удобным и многофункциональным расширяемым средством для исследования алгоритмов балансировки нагрузки процессоров при решении задач глобальной оптимизации методом ветвей и границ. Данное программное средство позволяет осуществлять как визуальный, так и автоматизированный анализ производительности алгоритмов.

Работа выполнена при поддержке РФФИ (проекты № 13-07-00768 А и № 0115РК00554)

Литература

1. М.А. Посыпкин, И. Х. Сигал, Исследование алгоритмов параллельных вычислений в задачах дискретной оптимизации ранцевого типа // Ж. вычисл.матем. и матем. физ., 2005, том 45, номер 10, С. 1801–1809.
2. И.Х. Сигал, Я. Л. Бабинская, М. А. Посыпкин Параллельная реализация метода ветвей и границ в задаче коммивояжера на базе библиотеки BNB-Solver комплексах // Труды ИСА РАН 2006. Т. 25, С.26-36.
3. Distributed Computing and Its Applications. // Felicity Press, Bristol,USA, 2005.ISBN: 0-931265-10-2, 298p. Монография (соавторы: S.V. Emelyanov, A.P. Afanasiev, Y.R. Grinberg, V.E. Krivtsov,B.V. Peltsverger, O.V. Sukhoroslov, R.G. Taylor, V.V. Voloshinov).
4. Лупин С. А., Посыпкин М. А. Технологии параллельного программирования: Учеб. пос //Сер. Высш. образ-ние. М.: Форум Инфра-М. – 2008. – Т. 208. – С. 2000.
5. Gendron B., Crainic T. G. Parallel branch-and-branch algorithms: Survey and synthesis //Operations research. – 1994. – Т. 42. – №. 6. – С. 1042-1066.
6. Стронгин Р. Г., Гергель В. П., Баркалов К. А. Параллельные методы решения задач глобальной оптимизации //Известия высших учебных заведений. Приборостроение. – 2009. – Т. 52. – №. 10. – С. 25-33.
7. Воеводин В.В., Воеводин Вл.В. Параллельные вычисления. БХВ-Петербург. 2002. С. 329.
8. Рейтинг по состоянию на июнь 2015 года с официального сайта Топ 500: http://www.top500.org/lists/2015/06/.
9. Официальный сайт проекта Ganglia: http://ganglia.sourceforge.net/.
10. Официальный сайт проекта Nagios: http://www.nagios.org/
11. Страница официального сайта NVIDIA с описанием основных возможностей профилировщика NVIDIA Visual Profiler: https://developer.nvidia.com/nvidia-visual-profiler.
12. Страница официального сайта PGI с описанием основных возможностей профилировщика PGPROF: https://www.pgroup.com/products/pgprof.htm.
13. Страница проекта BNB-Simulator https://github.com/fominandrey/bnb-simulator.
14. Evtushenko Y., Posypkin M., Sigal I. A framework for parallel large-scale global optimization //Computer Science-Research and Development. 2009. Т. 23. №. 3-4. С. 211-215.
15. Страница проекта BNB-Solver https://github.com/mposypkin/BNB-solver.
16. Воеводин В.В., Воеводин Вл.В. Параллельные вычисления. БХВ-Петербург. 2002. С. 82-83.

Тархов Д. А., Радченко Д.С.

Санкт-Петербургский государственный политехнический университет, Санкт-Петербург, Россия,
доктор технических наук, профессор, e-mail: dtarkhov@gmail.com, darya.darya.radchenko@mail.ru

РАСПРЕДЕЛЁННЫЕ АЛГОРИТМЫ ОПТИМИЗАЦИИ

КЛЮЧЕВЫЕ СЛОВА

Нейронные сети, распределенные вычисления, глобальный экстремум, кластеризация, генетический алгоритм.

АННОТАЦИЯ

Рассмотрены методы и алгоритмы оптимизации, реализуемые на нескольких (многих) компьютерах, сообщающихся между собой через Интернет, которые могут применяться при обучении нейронных сетей [10,15]. Рассмотрена ситуация, когда вычислительные процессы на разных компьютерах тесно взаимодействуют друг с другом. При этом работа алгоритмов не сильно меняется в случае, когда отдельные компьютеры могут выходить из процесса вычислений.

Введение

Под распределёнными вычислениями мы будем понимать процессы решения одной задачи на нескольких (многих) компьютерах, сообщающихся между собой через Интернет. Такая постановка задачи влечёт за собой ряд особенностей, первой из которых является то, что каналы связи оказываются медленными и ненадёжными.

Опишем специфику задачи более конкретно.

1. Данные, которые мы хотим обработать, возникают в разных частях Интернет, при этом пересылку их в одно место с последующей там обработкой будем считать невозможной или нерациональной;

2. Вычисления производятся на нескольких (многих) различных компьютерах, при этом вычислительные процессы на разных компьютерах взаимодействуют друг с другом, скорость такого взаимодействия и производительность разных компьютеров плохо прогнозируема и меняется со временем;

3. Отдельные компьютеры могут временно выходить из процесса вычислений, причём это время может быть непредсказуемым;

Подобные алгоритмы и результаты их тестирования приводились во многих работах [2,1,3,4,5]. Результаты являются удовлетворительными, но, как показывает практика, до сих пор актуальной является задача разработки распределенных алгоритмов оптимизации, позволяющего добиться максимально быстрой оптимизации целевого функционала, в случае его многоэкстремальности и большой размерности пространства варьируемых параметров. Такая задача обычно актуальна при обучении нейронной сети не слишком малой размерности, позволяющего организовать эффективные распределенные вычисления для как можно большего числа вычислительных узлов. Рассмотрению таких алгоритмов и посвящена данная статья.

Распределённый вариант метода рестартов

Проблема обучения нейронной сети с вычислительной точки зрения это проблема нахождения глобального экстремума сложной многоэкстремальной функции, обычно с большим числом аргументов. Самый простой способ её решения, состоит в том, чтобы использовать метод рестартов, состоящий в том, что на каждом из вычислительных узлов запускается процесс поиска локального экстремума из своей точки (набора точек), после чего выбирается лучший результат. Даже в этом простом случае возникает ряд интересных алгоритмических особенностей, если рассматривать не каждый локальный алгоритм в отдельности, а механизм взаимодействия локальных процессов.

Алгоритм 1

Шаг 1. На каждом из компьютеров запускаем процесс поиска локального экстремума из своей точки.

Шаг 2. Каждый узел через определённые промежутки времени посылает в другой узел, выбранный случайно или по определённому закону, свой идентификатор и значение функционала

ошибки.

Шаг 3. Узел, получивший такую информацию, сравнивает присланное значение функционала с тем, которое получено локальным процессом и отсылает далее информацию о лучшем результате. Является ли адрес посылки сообщения случайным или подчиняется определённому закону, надо чтобы информация поступала в узлы максимально равномерно.

Шаг 4. Если какой-либо узел долго не получает информацию из других узлов, то он начинает посылать свою более часто – таким образом система будет адаптироваться к замедлению работы сети и возможным сбоям в ней.

Шаг 5. Полученная информация сразу или через заданный промежуток времени сказывается на ходе работы локального алгоритма. Самый простой способ работы с поступившей информацией состоит в том, чтобы после истечения заданного времени работы сравнить свой результат с лучшим из присланных результатов.

- если свой результат хуже – можно запросить информацию о весах (и структуре сети, если она модифицировалась локальными алгоритмами) из узла с лучшим результатом, если этот узел не отвечает, запросить узел со следующим от минимального значением функционала и т.д.;
- если свой результат лучше, то его надо переслать в другие узлы. Более сложный вариант шага 5 – построить авторегрессию по значениям лучших вариантов и времени их прихода и с её помощью спрогнозировать результаты, которые могут прийти далее. Если прогнозное время прихода очередного лучшего результата слишком велико или прогнозное значение функционала мало отличается от уже полученного, тогда можно поступить, как и выше;

Шаг 6. Когда критерием остановки вычислительного процесса является достижение некоторого фиксированного заранее значения минимизируемого функционала, тогда возможны следующие варианты.

- если это значение достигнуто локально, то процесс оптимизации останавливается и происходит рассылка результатов;
- если в другом узле – то этот узел должен прислать свои результаты – веса и т.д.;
- если соответствующий узел их ещё не прислал, то их следует запросить;

Возможен вариант, когда пересылается информация не только о лучшем из присланных в данный узел результатов.

Распределённый вариант метода многогранника

Интересные вопросы возникают при использовании в качестве основы метода многогранника [8,9]. Главная проблема его лобового распараллеливания состоит в том, что на каждом шаге пересчитывается одна худшая точка. Более перспективным представляется вариант, при котором в каждом узле вычисляется функционал в своём наборе точек и запускается локальная версия метода многогранника, при этом время от времени вычисляется общее среднее и оно используется вместо локального среднего. Однако, в такой ситуации нет необходимости использовать в качестве метода, реализуемого на отдельном узле именно метод многогранника – в качестве такового может выступать любой достаточно эффективный алгоритм.

Приведём соответствующий адаптивный алгоритм, устойчивый к сбоям в сети и возможным выпадениям компьютеров из совместной работы.

Алгоритм 2

Шаг 1. На каждом из компьютеров запускается процесс поиска локального экстремума из своей точки.

Шаг 2. Каждый компьютер посылает на другой компьютер из этой совокупности результаты своей работы – веса сети и величину ошибки.

Шаг 3. Получивший эту информацию компьютер пересчитывает среднее значение весов по известной рекуррентной формуле $\mathbf{c}_N = \mathbf{c}_{N-1} + \frac{1}{N}(\mathbf{w}_N - \mathbf{c}_{N-1})$ и пересылает далее. Если мы учитываем устаревание информации, тогда множитель $\frac{1}{N}$ можно заменить числом $\alpha : 0 < \alpha < 1$.

Шаг 4. Если выполнены заданные условия (например, значение ошибки в этом узле больше, чем заданное число ошибок в предыдущих узлах), то совершается отражение весов относительно среднего в соответствии с методом многогранника и с этой точки продолжается работа локального метода минимизации. При этом цепочка пересылки данных может обрываться,

продолжаться дальше или начинаться новая.

Шаг 5. Если сеть ненадёжна и компьютер долго не получает таких «посланий» от других узлов, он сам инициирует новую цепочку.

Такой вычислительный процесс обладает рядом полезных свойств.

- во-первых, сохраняется нелокальный характер поиска, если не получается так, что все локальные процессы стягиваются к одной точке, являющейся точкой локального экстремума. Отражение относительно среднего для разного множества вершин и изменение самих вершин локальными методами оптимизации предотвращает зацикливание метода;
- во-вторых, найденный глобальный минимум уже не потеряется, а из локальных, в особенности из тех, которым соответствуют большие ошибки, локальный процесс будет «выброшен» рано или поздно;
- в-третьих, выпадение части узлов из процесса не сильно сказывается на его эффективности, если значительная часть пакетов теряется, то это приведёт только к более частой их генерации, слабо сказавшись на общем ходе процесса оптимизации;

Распределённый вариант метода кластеризации промежуточных результатов

Одним из наиболее известных и результативных направлений повышения эффективности метода рестартов является процедура кластеризации получившихся промежуточных результатов. После кластеризации выбирается один представитель кластера и производится спуск из этой точки и рестарт из остальных [10]. Прежде чем осуществлять спуск можно вычислить значение функционала в центре кластера и сравнить с лучшим значением функционала в точках кластера. При этом заведомо неперспективные кластеры можно выкидывать совсем.

Для реализации такого алгоритма в распределённой системе необходим соответствующий распределённый алгоритм кластеризации – например построение распределённой сети Кохонена.

Алгоритм 3

Шаг 1. Каждому узлу при старте минимизации присваивается некоторый номер кластера.

Шаг 2. После определённого числа шагов локальной оптимизации номер кластера вместе со значениями весов пересылается в другой случайный узел.

Шаг 3. Если этот номер совпадает с номером получившего пакет узла, то получатель пересчитывает полученное среднее значение весов, запоминает его и отсылает номер и среднее дальше. Если номер другой, то полученное среднее сравнивается с последним сохранённым средним своего класса и своими текущими весами. Если своё среднее ближе, то к пакету присоединяются свой номер и среднее, и составной пакет пересылается дальше. Если ближе чужое среднее, то локальный номер меняется на новый, пересчитывается новое среднее и пакет пересылается дальше.

Шаг 4. Повторяем шаги 2 и 3 с очевидной модификацией – сравнение происходит со всеми полученными средними в пакете.

Шаг 5. После заданного промежутка времени работы алгоритма пересылаем не только веса, но и лучшие в кластере значения функционала ошибки. Если локальное значение ошибки больше лучшего в кластере, то в данном узле происходит рестарт (возможно учитывающий накопленную информацию).

Распределённые варианты других глобальных алгоритмов

Ещё одним подходом к глобальной минимизации многоэкстремальной функции является минимизация её свёртки с унимодальной функцией. Если $E(\mathbf{w})$ - функция ошибки, которую мы минимизируем, то вместо неё мы можем минимизировать функцию $G(\mathbf{u})=\int E(\mathbf{w})-p(\mathbf{u}-\mathbf{w})d\mathbf{w}$, где $p(\mathbf{u})$ - сглаживающая функция, например плотность некоторого вероятностного распределения. Чем ближе $p(\mathbf{u})$ к δ - функции, т.е. чем в большей степени она локализована в окрестности нуля, тем ближе $G(\mathbf{u})$ к $E(\mathbf{w})$. Чем дальше $p(\mathbf{u})$ от δ - функции, т.е. чем более она размазана, тем более гладкой получается $G(\mathbf{u})$ и тем проще её минимизировать. Обычно рассматривается семейство функций $G_\alpha(\mathbf{u})=\int E(\mathbf{w})-p_\alpha(\mathbf{u}-\mathbf{w})d\mathbf{w}$, причём $p_\alpha(\mathbf{u}) \underset{\alpha\to 0}{\to} \delta(\mathbf{u})$. Работа алгоритма начинается при достаточно больших значениях α и постепенно этот параметр уменьшается, устремляясь к нулю с ростом номера шага. Следует заметить, что такое уменьшение не обязательно проводить на каждом шаге.

Такой алгоритм достаточно удобно распределяется, если эту свёртку вычислять методом

Монте-Карло, т.е. минимизировать функцию $G_\alpha(\mathbf{u})=\frac{1}{N}\sum_{k=1}^{N}E(\mathbf{w}_k)$, где точки $\mathbf{w}_k$ выбраны случайно в соответствии с распределением $p_\alpha(\mathbf{u}-\mathbf{w})$. Если функция $p_\alpha(\mathbf{u})$ выбрана достаточно удачно, то можно вычислить градиент $G_\alpha(\mathbf{u})$ в соответствии с формулой $\nabla G_\alpha(\mathbf{u})=\int E(\mathbf{w})\nabla p_\alpha(\mathbf{u}-\mathbf{w})d\mathbf{w}$, причём этот интеграл тоже можно вычислить по методу Монте-Карло. Например, если взять $p_\alpha(\mathbf{u})=\frac{1}{(2\pi\alpha)^{n/2}}e^{\frac{-(u)^2}{2\alpha}}$, что соответствует многомерному нормальному распределению с нулевым математическим ожиданием и дисперсией α, то для градиента получится формула $\nabla G_\alpha(\mathbf{u})=\int \frac{E(\mathbf{w})}{(2\pi\alpha)^{n/2}\alpha}e^{\frac{-(\mathbf{u}-\mathbf{w})^2}{2\alpha}}(\mathbf{w}-\mathbf{u})d\mathbf{w}$. Метод Монте-Карло позволяет получить оценку для градиента $\nabla G_\alpha(\mathbf{u})=\frac{1}{N\alpha}\sum_{k=1}^{N}E(\mathbf{w}_k)(\mathbf{w}_k-\mathbf{u})$, где вектор $\mathbf{w}_k$ распределён нормально с математическим ожиданием **u** и дисперсией α.

Алгоритм 4

Шаг 1. Выбираем начальные значения весов **u** и другие параметры алгоритма, например закон изменения α. Эти значения должны быть общими для всех узлов, т.е. они должны быть согласованы.

Шаг 2. На каждом компьютере производим независимую генерацию точек $\mathbf{w}_k$ в соответствии с распределением $p_\alpha(\mathbf{u}-\mathbf{w})$.

Шаг 3. На каждом компьютере проводим вычисления $G_\alpha(\mathbf{u})$ и $\nabla G_\alpha(\mathbf{u})$ по приведённым выше формулам. При этом отдельные части сумм вычисляются на разных компьютерах.

Шаг 4. Результаты вычислений шага 2 пересылаются узлами друг другу. Для этого узлы объединяются в ориентированный граф, по рёбрам которого осуществляется прямая пересылка с усреднением пришедших результатов вычислений $G_\alpha(\mathbf{u})$ и $\nabla G_\alpha(\mathbf{u})$ с соответствующими локальными результатами.

Шаг 5. В конечном компьютере осуществляем шаг некоторого алгоритма минимизации $G_\alpha(\mathbf{u})$, т.е. вычисляются новые веса обучаемой сети.

Шаг 6. Итоговые результаты пересылаются в обратном порядке.

Шаг 7. Повторяем шаги 2 – 6 до выполнения условий останова. При этом временно выпавшие из вычислений узлы должны перед возобновлением вычислений получить текущие значения параметров **u** и α. Другие изменения алгоритма в случае выпадения отдельных компьютеров, в частности изменения графа шага 4, очевидны.

Для эффективной работы алгоритма при большом числе узлов следует разумно отбирать пересылаемую информацию, для этого можно использовать кластеризацию и другие описанные выше подходы, применяя данный метод по группам компьютеров.

Заключение.

Поиск глобального экстремума сложной многоэкстремальной функции, например, процесс обучения нейронных сетей, как правило, является весьма ресурсоёмкой задачей, поэтому актуальным является поиск возможностей ускорения соответствующей процедуры. Одной из таких возможностей является вычисления в распределённом режиме. Традиционный подход состоит в попытках распараллелить традиционные алгоритмы нелинейной оптимизации, что не всегда приводит к успеху. Мы предлагаем примеры алгоритмов, которые сконструированы удобными для распределённой реализации. Подобный подход представляется более эффективным для параллельных и распределённых вычислений.

Работа поддержана грантом РФФИ №14-01-00660А.

Литература

1. Нгуен Занг Зуи Чыонг, Краснощёков А.А. Параллельное обучение искусственных нейронных сетей на основе системы Грид-вычислений ANTHILL. // Вестник ДГТУ. с. 451-456.
2. Пильгун В. М. Глубинное обучение нейронных сетей и достижения в их применении. // European research. -2015.- № 3.
3. Нгуен Занг Зуи Чыонг. Решений обратных задач теории упругости с помощью искусственных нейронных сетей. //

Диссертация на соискание ученой степени канд. техн. наук. Ростов- на- Дону. -2014.- с. 62.
4. Короткий С. Нейронные сети: алгоритм обратного распространения // http://www.gotai.net/documents/doc-nn-003.aspx. -1996.
5. Geoffrey E. Hinton. How neural networks learn from experience. // Scientific American. September 1992. pp. 146-151.
6. Тархов Д.А. Нейросетевые модели и алгоритмы. – М.: Радиотехника, 2014. – 352 с.
7. A.N. Vasilyev, D.A. Tarkhov,. Mathematical Models of Complex Systems on the Basis of Artificial Neural Networks// Nonlinear Phenomena in Complex Systems. 2014 vol. 17, no.3, pp. 327-335
8. Spendley, W., Hext, G.R., and Himsworth, F.R. (1962), "Sequential Application of Simplex Designs in Optimisation and Evolutionary Operation", Technometrics, Vol. 4, pp. 441–461.
9. Nelder, J.A. and Mead, R. (1965), "A simplex method for function minimization", Comput. J., 7, pp. 308–313.
10. Жиглявский А.А., Жилинскас А.Г. Методы поиска глобального экстремума. - М.: Наука, 1991. 246с.

Артамонов Ю.С.[1], Востокин С.В.[2]

[1] Самарский государственный аэрокосмический университет им. С.П. Королева (национальный исследовательский университет), г. Самара, аспирант кафедры информационных систем и технологий, artamonov@about.me

[2] Самарский государственный аэрокосмический университет им. С.П. Королева (национальный исследовательский университет), г. Самара, д.т.н, профессор кафедры информационных систем и технологий, easts@mail.ru

ПРИМЕНЕНИЕ ОБЛАЧНОГО СЕРВИСА TEMPLET WEB ПРИ ПРОВЕДЕНИИ ЛАБОРАТОРНЫХ ПРАКТИКУМОВ НА СУПЕРКОМПЬЮТЕРЕ «СЕРГЕЙ КОРОЛЕВ»

КЛЮЧЕВЫЕ СЛОВА

Облачный сервис, библиотека, параллельное программирование, окружение, запуск, мониторинг, вычисления, кластер.

АННОТАЦИЯ

В статье описана специфика приложений научной направленности, сделан акцент на характерных проблемах разработки данного вида программного обеспечения. Приводится классификация систем по способу организации вычислений и уровню абстракции вычислений от физического оборудования. Рассматривается инструментарий для разработки приложений, включающий в себя библиотеки параллельных вычислений, сервис запуска и отслеживания задач, подсистему мониторинга состояния кластера. Описано использование инструментария при проведении лабораторных практикумов на суперкомпьютере «Сергей Королев». Сделан вывод о возможностях применения веб-сервиса Templet в учебном процессе.

Для разработки программ существует множество инструментов – среды разработки, компиляторы и трансляторы, библиотеки, парсеры, отладчики и многое другое. Практически все они ориентированы на создание конечного продукта и лишь немногие – на итеративную разработку и поддержку. Отличительными особенностями приложений для научных вычислений являются: с одной стороны короткий жизненный цикл приложения, при котором часто не требуется отдельная документация, тестирование в различных ОС, оформление графического интерфейса и другие финальные этапы разработки программного обеспечения (ПО), с другой – большое количество вносимых изменений, требующее запуска приложения и получения результатов [1]. Программы научных вычислений очень специализированы и зависят от целевого окружения (кластера, сети, систем хранения данных) довольно сильно, что делает разработку таких приложений непохожей на разработку системного и прикладного ПО.

Помимо специфики самих приложений стоит отметить ряд характерных проблем, возникающих в процессе работы над проектом:

- растущая сложность алгоритмов и архитектуры приложений;
- окружения, в которых производится запуск, зачастую имеют нестандартный набор ПО и периферийных устройств;
- сложный и длительный процесс развёртывания ПО в целевом окружении;
- для тестирования может потребоваться производительное окружение;
- оборудование и инфраструктура могут быть уникальны и их нельзя воссоздать для тестирования;
- необходимо соблюдать регламент работы с целевым окружением, определённое время работы на кластере и лимит нагрузки;
- доступ к окружению может быть ограничен локальной сетью организации.

Под окружением здесь понимается некоторый программно-аппаратный комплекс: совокупность физических или виртуальных ЭВМ, каналов связи, периферийных устройств и ПО, необходимого для работы системы. Например, это может быть кластер серверов, единичный сервер или же Desktop Grid система.

Процесс организации работы над приложением также может быть сложным, поскольку одно приложение может разрабатываться целым коллективом. Причём командная разработка всё чаще требуется при работе над приложениями научных вычислений. Во многом это обусловлено тем, что команда может разрабатывать многие модули и подсистемы параллельно, а в команде часто требуются как специалисты по предметной области, так и системные программисты, специалисты по параллельным вычислениям.

Отметим наиболее важные аспекты командной работы над приложениями:

1. В крупных проектах требуется использовать системы контроля версий (VCS), ведь в процессе разработки код проекта претерпевает значительные изменения, которые нужно контролировать. Обязательно должна быть возможность вернуться к более ранней версии [2].
2. В проекте должен быть простой и эффективный механизм сборки приложения без использования графических утилит, поддерживающий управление зависимостями и различные целевые архитектуры [3]. Следует отметить, что для вычислений всё чаще используются языки, которым не требуется компиляция в машинно-зависимый код: Java, Python, R, Lisp, Scheme. Использование этих языков существенно упрощает работу над проектом, но не стоит забывать о накладных расходах при исполнении в виртуальной машине (на работу интерпретатора).
3. Поскольку тестовый запуск может требовать значительных ресурсов, большое значение приобретают модульные и интеграционные тесты [4]. Такие тесты должны запускаться автоматически при поступлении новых изменений от участников проекта.
4. Следует организовать тестовое окружение, если это возможно, и тестовые запуски проводить в нём. Запуск приложения на суперкомпьютере для целей тестирования и отладки требует часть ценных разделяемых ресурсов, а результаты тестирования могут быть получены с существенной задержкой в случае загруженности основных ресурсов.

Для решения этих проблем разработано множество инструментов, позволяющих организовать работу над проектом, упростить запуск приложений и обработку результатов. В работе идёт речь о системах управления вычислительными задачами, как основном инструменте ресурсоемких научных вычислений.

Системы управления вычислительными задачами можно разделить на несколько специализированных классов:

- низкоуровневые системы управления распределёнными ресурсами (пакетные системы);
- Desktop Grid системы;
- системы виртуализации вычислительных ресурсов;
- высокоуровневые платформы распределённых вычислений.

Низкоуровневые системы управления распределёнными ресурсами призваны утилизировать доступные вычислительные ресурсы и максимизировать пропускную способность, управляя процессом назначения физических ресурсов пользовательским задачам. Пакетные системы управляют такими физическими ресурсами как: процессорное время, оперативная память, дисковое пространство и пропускная способность сети. К таким системам относятся [5]: Platform LSF, Portable Batch System (PBS), IBM Load Leveler и Condor. Пакетные системы обычно имеют следующие подсистемы:

- подсистема управления задачами;
- подсистема управления физическими ресурсами;
- подсистема планирования и управления очередями.

В пакетных системах назначение ресурсов для задач зависит от политики планирования и использования ресурсов. Эта политика определяется администратором системы и является конфигурацией для подсистемы планирования и управления очередями.

Основные достоинства пакетных систем:

- пригодны для организации вычислений на различном оборудовании и операционных системах;
- хорошо подходят для большого числа перезапусков задач;
- позволяют организовать работу множества пользователей;
- документированы, широко известны и повсеместно применяются.

Недостатки:

- требуется обучение пользователей, поскольку им необходимо знать особенности

физического оборудования кластера для запуска задач, такие как: число ядер процессоров, оперативная память, скорость чтения и записи данных, топология сети;
- пакетные задания трудно отлаживать;
- зависшие задачи могут держать занятыми ресурсы всё отведённое им время.

Desktop Grid системы, как и пакетные системы приложения, нацелены на утилизацию вычислительных ресурсов, однако они существенно отличаются от них. Пакетные системы задействуют оборудование организаций: суперкомпьютеры, кластеры и другое дорогостоящее оборудование. В отличие от пакетных систем Desktop Grid задействуют обычные компьютеры [6], а связь между ними организуется через сеть Интернет, компьютеры могут быть выключены или отключены от сети. Участники проектов Desktop Grid не являются экспертами и разработчиками ПО, они участвуют в вычислениях, поскольку им это интересно. Проекты не контролируют своих участников и не могут предотвратить злонамеренное поведение, например, предоставление ложных результатов. К приложениям Desktop Grid относятся: Charlotte, Javelin и Bayanihan, BOINC, XTremweb и Entropia [7].

Плюсы использования Desktop Grid:
- возможность задействовать значительные вычислительные ресурсы длительное время;
- бесплатный доступ к вычислениям;
- хорошо подходит для постоянных исследований и переборных задач;
- прост для участников.

Недостатки:
- нестабильная вычислительная мощность сети компьютеров;
- различные по оборудованию компьютеры;
- ограниченный размер одного вычислительного задания;
- необходимость запуска дубликатов заданий, поскольку они исполняются в недоверенном окружении;
- трудно прогнозируемый срок исполнения заданий.

Системы виртуализации вычислительных ресурсов нацелены на использование виртуальных машин и контейнеров изоляции ресурсов в качестве единицы исполнения. Виртуализация — механизм абстракции физического оборудования и системных ресурсов операционной системы. В настоящее время технологии виртуализации тесно связаны с концепцией облачных вычислений. Облачные вычисления по своей природе требуют задействовать технологии, позволяющие отделить приложения от операционной системы и реального оборудования. К системам виртуализации относятся гипервизоры: Xen, VMware и KVM.

Приложения высокопроизводительных вычислений могут получить выигрыш от виртуализации несколькими путями [8]:
- гипервизор может гарантировать фиксированное выделение ресурсов для гостевых ОС, таких как процессорное время и оперативная память;
- узлы виртуального кластера могут быть запущены на различных ядрах одного или нескольких реальных узлов, причём конфигурацию узлов и их физическое расположение можно менять во время работы;
- изоляция пользовательских процессов делает системы на базе виртуализации более устойчивыми к сбою пользовательского ПО;
- гипервизор берёт на себя обеспечение безопасности процессов и данных.

Недостатки:
- сложность ПО виртуализации;
- виртуализация требует дополнительных ресурсов процессора и памяти на размещение гостевой ОС;
- долгий (по сравнению с запуском приложения) холодный старт новых виртуальных машин.

Стоит отдельно выделить технику контейнеризации приложений, как способ изоляции приложений и частичной виртуализации ресурсов. В отличие от гипервизоров контейнерная виртуализация не требует виртуализации гостевой ОС. Она задействует механизмы ядра операционной системы, которые позволяют изолировать процессы и выделить для них физические ресурсы системы [9]. Такой подход реализован в системах: OpenVZ, LXC и Docker.

И хотя контейнерная виртуализация, по сравнению с виртуальными машинами, требует меньших ресурсов, она также имеет недостатки:

- меньший уровень изоляции пользовательского ПО на уровне разделяемых ресурсов;
- меньший уровень безопасности приложений и данных.

Платформы распределённых вычислений представлены как отдельный класс систем, поскольку задействуют другие подходы к организации распределённых вычислений. Например, используют другие системы для реального исполнения задач и, самое главное, предоставляют высокий уровень абстракции от физических ресурсов.

Этому классу систем принадлежат:

- Globus Toolkit — позволяет запускать задачи с использованием планировщиков PBS, Condor, Platform LSF [10];
- Hadoop — набор библиотек и служебных программ для организации вычислений в парадигме MapReduce [11];
- CLAVIRE - многопрофильная инструментально-технологическая среда, позволяющая строить распределённые приложения из готовых вычислительных узлов, определяя потоки данных [12].

К достоинствам платформ распределённых вычислений стоит отнести развитые средства разработки приложений, многообразие способов запуска и мониторинг исполнения приложений.

Недостатки:

- сложность отладки;
- специализированный процесс разработки приложений, требующий задействования средств платформы;
- сокрытие деталей взаимодействия приложений с оборудованием.

На базе Самарского государственного аэрокосмического университета авторами разработан веб-сервис Templet Web, позволяющий автоматизировать разработку параллельных приложений для кластера «Сергей Королев».

Веб-сервис состоит из 3 компонентов:

- библиотеки параллельного программирования Templet SDK;
- веб-сервис для запуска и мониторинга задач на суперкомпьютере Templet Web;
- подсистема мониторинга состояния кластера.

Templet SDK – комплект инструментов для разработки параллельных приложений. Он включает в себя препроцессор исходных текстов и библиотеку времени исполнения. Препроцессор позволяет разработать и отладить последовательную программу на локальной машине, а затем запустить код на исполнение на суперкомпьютере в параллельном режиме (код будет получен автоматически из последовательной версии, для параллельной работы будет использоваться API POSIX Threads). Применение инструментов Templet SDK не требует от пользователя знаний методов параллельного программирования в UNIX или Windows и позволяет сосредоточиться на решении прикладной задачи.

Веб-приложение для управления проектами и задачами Templet Web, развёрнутое по адресу http://templet.ssau.ru/templet, позволяет:

- начать разработку проекта параллельного приложения при помощи шаблонов вычислительных методов пакета Templet SDK: портфель задач, конвейер и другие;
- организовать частное облако окружений для запуска проектов;
- организовать работу команды над проектом с применением VCS;
- получать доступ к общим окружениям и запущенным задачам;
- разворачивать приложение в тестовом и целевом окружениях;
- отслеживать работу приложения во время продолжительных вычислений;
- получать уведомления о статусе задач в целевых окружениях;
- управлять бинарными зависимостями шаблонов и проектов для различных архитектур и платформ.

Шаблоны проектов позволяют предоставлять готовые структуры проектов и стандартизировать пакеты поставки приложений в целевые окружения. Проекты строятся на основе шаблонов и могут хранить код в системах контроля версий (VCS), в проект подключаются общие окружения для запуска задач. Задачи и окружения проекта доступны всей команде проекта, что позволяет эффективно организовать коллективную работу. Результаты вычислений сохраняются в архиве, к ним можно получать доступ не только по завершению задач, но и позднее. Сервис ведёт учёт использования ресурсов суперкомпьютера и предоставляет пользователям статистику по запущенным задачам и загрузке ресурсов. Это помогает оценить максимальный

объём ресурсов, который сможет использовать приложение, с учётом допустимого времени ожидания вычислительной задачи в очереди пакетной системы.

Templet Web относится к классу высокоуровневых систем и задействует в качестве исполняющего механизма планировщик Torque или отдельно запускаемый процесс. В системе поддерживаются окружения под управлением ОС Linux (в том числе виртуальные) и суперкомпьютер «Сергей Королев». Система имеет расширяемую архитектуру, что позволяет добавлять поддерживаемые окружения по мере её развития. В Templet Web основной акцент сделан на простом процессе разработки приложений и интеграции средств разработки и платформы для исполнения. Сервис работает на двух языках – русском и английском, он доступен для использования не только студентам и преподавателям СГАУ, но и любому стороннему пользователю.

В основе механизма взаимодействия системы и целевых окружений лежит прямое подключение к окружению по безопасному протоколу SSH (Secure Shell). Большим преимуществом такого подхода является возможность использования неподготовленных окружений на базе ОС Linux, поскольку для работы веб-сервиса в этом случае не требуется установки дополнительного ПО и произведения настройки окружения для взаимодействия с системой.

Недостатком такого решения является необходимость иметь прямое подключение от веб-сервиса к серверу окружения (или к узлу управления в случае суперкомпьютера). Зачастую высокопроизводительные кластеры недоступны для использования за пределами сети организации, выходом в таком случае может стать организация защищённой виртуальной сети (VPN). Стоит отметить, что установка дополнительного ПО (например, агента на стороне кластера) сопряжена с большими сложностями, ведь политика безопасности окружения может ограничивать возможность запуска сторонних приложений в режиме демона.

Подсистема мониторинга суперкомпьютера «Сергей Королев» собирает детальную информацию по использованию суперкомпьютера:

- состояние очереди пакетной системы;
- запрашиваемые задачами ресурсы;
- состояние узлов суперкомпьютера.

Информация, собираемая подсистемой мониторинга, используется для решения задач прогнозирования доступных вычислительных ресурсов кластера[13]:

- прогноз количества доступных вычислительных ресурсов;
- прогноз момента запуска вычислительной задачи в очереди пакетной системы.

Данные подсистемы мониторинга доступны в виде графиков загрузки кластера в веб-сервисе Templet Web. Открытые данные мониторинга доступны на сайте проекта по адресу http://templet.ssau.ru/wiki/открытые_данные.

Сервис Templet [14] используется в СГАУ при работе над приложениями научных вычислений и для проведения лабораторных практикумов на базе суперкомпьютера «Сергей Королев».

Для проведения лабораторных занятий веб-сервис позволяет:

- организовать работу группы студентов на кластере в заданное время, с разрешения преподавателя и без предоставления студентам прямого доступа к управляющему узлу кластера.
- вести архив примеров учебных программ с готовыми настройками запуска на кластере.
- сохранять исходный код программ и результаты их исполнения, что помогает исключить плагиат и недобросовестное использование кластера.

При проведении прикладных исследований сервис позволяет:

- предоставить оперативный доступ к кластеру для однократного или тестового расчёта;
- управлять длительными вычислениями и получать промежуточные результаты по мере исполнения задачи;
- организовать типовые расчёты на основе готовых скелетов параллельных алгоритмов.

Студенты создают проекты в системе, указывая в качестве репозитория любой из бесплатных хостингов открытых проектов (Github, SourceForge), поддерживающий доступ по протоколу SVN. Система автоматически выгружает последнюю версию исходного кода программы перед запуском задачи на кластере, сохраняет версию в архив задач и передаёт её для запуска в целевое окружение. Исходный код примеров хранится в служебном SVN репозитории, встроенном в систему, что позволяет дополнять и изменять их по мере развития курса лабораторных работ.

Лабораторный практикум с использованием возможностей Templet Web включает в себя изучение: потоков POSIX, потоков OpenMP, технологий скелетного программирования, библиотек MPI.

За время проведения лабораторных работ в течении весеннего семестра 2015 года системой воспользовались 5 учебных групп (более 70 студентов). При этом было создано 150 проектов, запущено 1500 задач и потрачено 20 часов машинного времени суперкомпьютера.

Инструментарий имеет большое значение при разработке приложений научной направленности, поскольку крайне важно грамотно выбрать платформу для организации разработки, средства распараллеливания и организации высокопроизводительных вычислений. Набор инструментов Templet предоставляет не только сервис для запуска приложений, но и библиотеки шаблонов параллельных приложений, средства для разработки и отладки прикладных решений. Он позволяет прикладным программистам сосредоточить усилия на работе вычислительных алгоритмов, снизить трудоёмкость разработки и сэкономить ресурсы на стадии тестирования и отладки. При использовании сервиса в обучающем процессе он может упростить административную работу преподавателя по настройке доступа студентов к кластеру, помогает исключить плагиат в лабораторных работах и предотвратить недобросовестное использования ресурсов.

Литература

1. Keyes D. E. et al. Multiphysics simulations: Challenges and opportunities // International Journal of High Performance Computing Applications. – 2013. – Vol. 27, no. 1.
2. Sullivan B. Making sense of revision-control systems // Communication of the ACM. – 2009. – Vol. 52, no. 9. – P. 56-62.
3. Neitsch A., Wong K., Godfrey M. W. Build system issues in multilanguage software // 28th IEEE International Conference on Software Maintenance (ICSM). – Riva del Garda, Trento, Italy, 2012. – P. 140-149.
4. Continuous software integration and quality control during software development / M. Ettl, A. Neidhardt, W. Brisken, R. Dassing // Launching the Next-Generation IVS Network, IVS 2012 General Meeting Proceedings, NASA/CP-2012-217504 / Ed. by K.D. Behrend, D.; Baver. – National Aeronautics and Space Administration, 2012. – 12. – P. 227-230.
5. Comparative study of distributed resource management systems SGE, LSF, PBS Pro, and LoadLeveler: Rep. / Technical Report-Citeseerx; Executor: Yonghong Yan, Barbara Chapman: 2008.
6. Anderson D. P. Boinc: A system for public-resource computing and storage // Grid Computing, 2004. Proceedings. Fifth IEEE/ACM International Workshop on / IEEE. – 2004. – P. 4-10.
7. SungJin C., HongSoo K., EunJoung B. Characterizing and classifying Desktop Grid // Seventh IEEE International Symposium on Cluster Computing and the Grid(CCGrid'07). – Rio de Janeiro, Brazil, 2007. – P. 743-748.
8. Virtualization for high-performance computing / Mark F Mergen, Volkmar Uhlig, Orran Krieger, Jimi Xenidis // ACM SIGOPS Operating Systems Review. – 2006. – Vol. 40, no. 2. – P. 8-11.
9. Performance evaluation of container-based virtualization for high performance computing environments / M. G. Xavier, M. V. Neves, F. D. Rossi et al. // Parallel, Distributed and Network-Based Processing (PDP), 2013 21st Euromicro International Conference on / IEEE. – 2013. – P. 233-240.
10. Foster I. Globus toolkit version 4: Software for service-oriented systems // Network and parallel computing. – Springer, 2005. – P. 2-13.
11. Taylor R. C. An overview of the Hadoop, MapReduce, HBase framework and its current applications in bioinformatics // BMC bioinformatics. – 2010. – Vol. 11, no. Suppl 12. – P. S1.
12. Clavire: e-science infrastructure for data-driven computing / K. V. Knyazkov, S. V. Kovalchuk, T. N. Tchurov et al. // Journal of Computational Science. – 2012. – Vol. 3, no. 6. – P. 504-510.
13. Артамонов Ю. С. Основные подходы прогнозирования доступных вычислительных ресурсов в кластерных системах // Перспективные информационные технологии (ПИТ 2014): труды Международной научно-технической конференции / Под. ред. С. А. Прохоров; Самарский государственный аэрокосмический университет. – Самара: Изд-во СНЦ РАН, 2014. – 2014. – С. 305-310.
14. Востокин С. В., Дорошин А. В., Артамонов Ю. С. Программный комплекс Templet. Организация прикладных вычислений на базе суперкомпьютера "Сергей Королев" // Управление движением и навигация летательных аппаратов: Сборник трудов XVII Всероссийского семинара по управлению движением и навигации летательных аппаратов. – Т. 1. – 2014.

Ефимов А.И.

ОАО Конструкторское бюро системного программирования, г.Гомель, Беларусь, **К.Т.Н.,** главный программист, eai_andr_kb@mail.ru

ВИРТУАЛТРЕДИНГ: НОВАЯ АРХИТЕКТУРА ДЛЯ ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМ С ПРЯМЫМ ТОНКО ГРАНУЛИРОВАННЫМ АППАРАТНЫМ МУЛЬТИПРОГРАММИРОВАНИЕМ

КЛЮЧЕВЫЕ СЛОВА

Параллельные вычисления, виртуалтредингговая архитектура, транзакционная микроархитектура, КПД, Силиконовый Занавес, прерывания, аппаратные семафоры.

АННОТАЦИЯ

В работе предложена новая архитектура вычислительных систем, названная виртуалтредингговой (ВТА). Ее особенностью, существенно упрощающей системное программирование и скорость исполнения параллельных приложений, является прямая аппаратная реализация управления нитями и вводом-выводом без прерываний, включая полнофункциональную синхронизацию. В ВТА использованы специальные архитектурные команды, заменяющие интерфейсные функции операционной системы. Ее высокая эффективность поддерживается оригинальной транзакционной микроархитектурой, заменяющей общепринятое грубо гранулированное программно-аппаратное мультипрограммирования нитей как единого целого тонко гранулированным аппаратным мультипрограммированием транзакций, порождаемых из архитектурных команд. Для количественного сравнения эффективности ВТА с существующими архитектурами предложено определение общетехнического показателя качества КПД. Представляется, что включение в учебный процесс подготовки инженеров и системных программистов исследования и реализации ВТА на свободных для академического использования средствах моделирования и макетирования должно значительно усилить конкурентоспособность специалистов в области информационных технологий.

Эффективность вычислительных систем

По утверждению эксперта в области высокопроизводительных вычислительных систем Бартона Смита [1] в настоящее время требуются радикальные решения, способные вывести индустрию IT из спирали специализации (spiral of specialization), в которой “высокопроизводительными считаются те вычисления, которые хорошо выполняются высокопроизводительными системами”. Такие решения должны обеспечить высокую скорость параллельных вычислений общего назначения [2], основным свойством которых является обработка в реальном времени интенсивного потока транзакций большим количеством нитей, взаимодействующих через разделяемую память. Такие вычисления характерны для больших СУБД, платежных систем и систем жесткого реального времени, требующих 5-10 микросекундного темпа обработки и управления [3].

Рассмотрим существующие подходы оценки эффективности вычислительных систем и определим для них общетехнический показатель качества – коэффициент полезного действия (КПД). Стерлинг [4] выделил четыре основные причины снижения скорости систем, “четыре всадника Апокалипсиса":

- накладные расходы - дополнительные работы на управление параллельностью;
- латентность - простои при доступе к памяти или другим частям системы;
- конкуренция - работа при соперничестве за разделяемые ресурсы;
- голодание - простои из-за слабого параллелизма и дисбаланса загрузки.

В совокупности эти причины вызывают состояния обобщенной латентности аппаратуры – состояния, в котором не выполняется полезная работа по продвижению вычислений. Соответственно, КПД аппаратуры можно определить простой формулой:

$$КПД = 1 - Сол / С$$

где С является общей стоимостью аппаратуры при выполнении некоторой работы, а Сол - стоимостью оборудования, находившегося в состоянии обобщенной латентности во время ее выполнения. По аналогии с тепловыми двигателями, улучшив архитектуру вычислительных систем (двигатель Дизеля вместо паровой машины), можно лучше использовать физические ресурсы – площадь и энергопотребление кристалла. Для практической оценки КПД архитектур необходимы соответствующие средства аппаратного мониторинга обобщенной латентности.

В последующих двух разделах понятие обобщенной латентности используется как ключ для сопоставления эффективности существующих архитектур с предлагаемой виртуалтрединговой. Следует отметить, что введенный таким образом КПД более точно отражает именно качество архитектуры, чем общепринятый показатель быстродействие/стоимость, отражающий сочетание качества архитектуры технологии элементной базы.

Программно-аппаратное мультипрограммирование

Как показано работах [5,6], мультипрограммирование естественно считать ортогональным по отношению к виртуальной памяти направлением виртуализации ВС, радикально упрощающим параллельное программирование. Например, последовательная программа сортировки массива методом пузырька является очень простой, если массив полностью умещается в памяти. Но ее сложность возрастает на порядки, если массив не помещается в физической памяти, а виртуальная память не поддерживается. Мультипрограммирование обеспечивает сходное упрощение сильно распараллеленных алгоритмов в соответствии с концепцией взаимодействия последовательных процессов Дейкстры [7].

Важнейшим практическим вопросом в вычислительных системах является эффективность, с которой подержаны эти направления виртуализации. Первая аппаратная реализация виртуальной памяти, заменившая ее чисто программную реализацию на основе оверлеев [8], появилась в суперкомпьютере Atlas, разработанном в 1961 году [9]. В ней использовалась одноуровневая организации с грубо гранулированным свопингом страниц (512 6-байтовых слов) между оперативной паять и барабаном. Последующее развитие привело к многоуровневой организации виртуальной памяти (кэши 1, 2 и 3 уровней с малой, до 32 байт, длиной свопируемых элементов) и когерентной работе в широко масштабируемых многоядерных системах [10,11].

В то же время, эффективность мультипрограммирования осталась практически на уровне, аналогичном уровню развития виртуальной памяти в системе Atlas [9]. В поддержку этого утверждения рассмотрим аппаратно-программное управления нитями и процессами – основными объектами мультипрограммирования в современных операционных системах [12]. Первичное управление любой вычислительной системой обеспечивает размещаемая в ПЗУ программа начальной загрузки 5, называемая также базовой системой ввода-вывода. Как и любая программа, она представляет собой результат трансляции алгоритма на языке (языках) программирования в исполнимый файл кода, содержащий архитектурные команды. На рисунке 1 для удобства дальнейшего изложения они показаны в 3х адреснгом формате архитектуры MIPS, для которой имеется свободный для академического использования проект системы на кристалле (СнК) MIPSfpga [13]. Поле OpC содержит код команды, поля AR[j] и AR[k] содержат адреса архитектурных регистров исходных операндов (либо литералы), а AR[i] – адрес регистра результата.

Исполнения кода BIOS идет в начальном состоянии процессора (или каждого ядра) на всех его аппаратных нитях. В этом состоянии количества программных процессов и нитей совпадает с числом аппаратных. На рисунке 1 аппаратные нити и процессы представлены дескрипторами нитей и контекстов. На основе этих дескрипторов устройства выборки команд и устройства управления памятью реализуют покомандное продвижение вычислений для nt нитей в контексте их процессов-хозяев.

Для поддержки мультипрограммирования BIOS может запускать на исполнение файл кода операционной системы (ОС), либо работать самостоятельно, как это принято во некоторых встроенных системах. В обоих случаях первичные нити порождают вторичные процессы и нити, являющиеся грубо гранулированными единицами мультипрограммирования. Порождение процесса заключается в создании в физической памяти вычислительной системы дерева трансляции, обеспечивающего отображение всех виртуальных адресов данного процесса на физические адреса оперативной памяти или файла свопинга. Как показано на рисунке 1, дескриптор процесса содержит свой идентификатор PId и корень дерева трансляции.

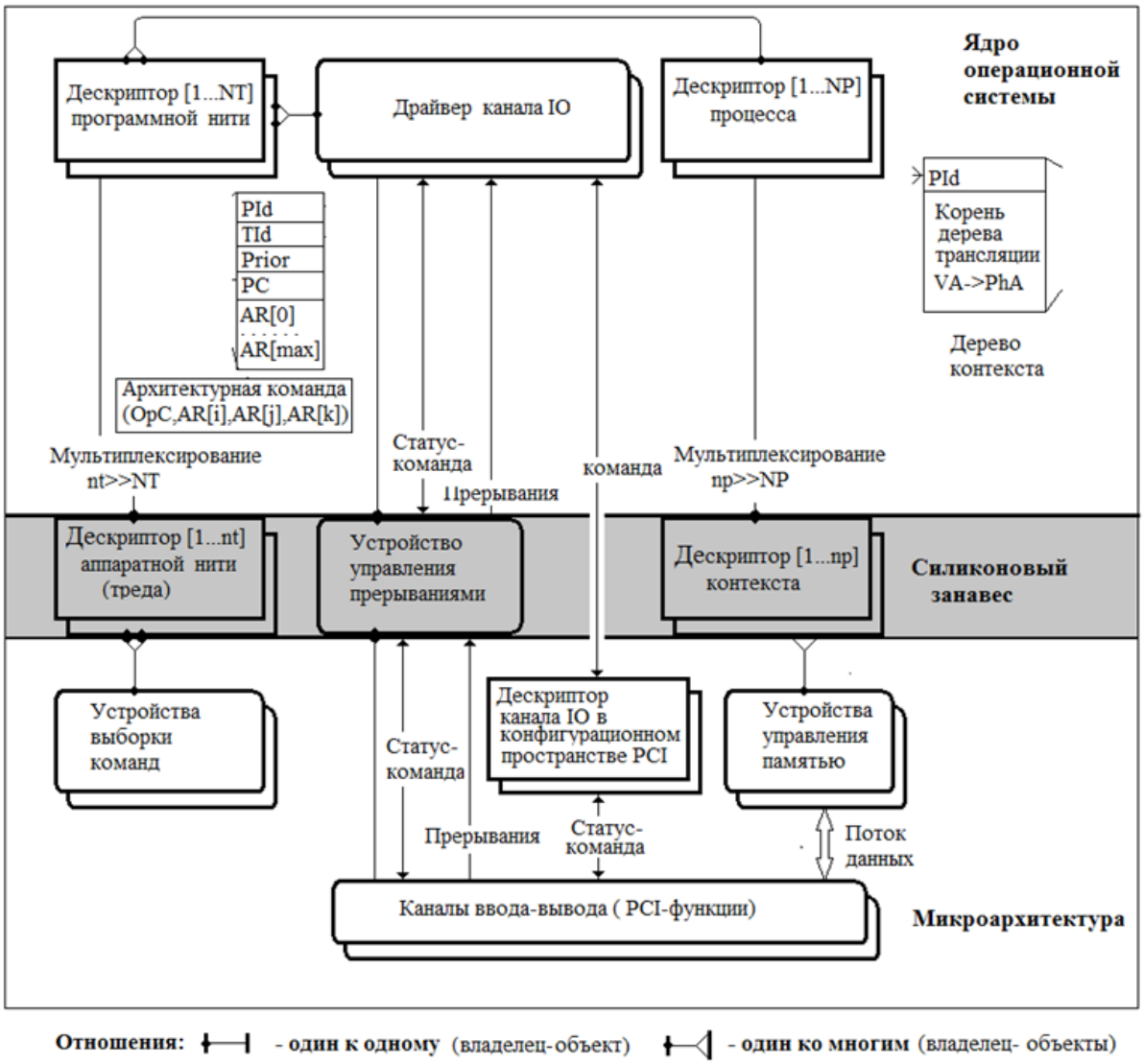


Рис.1. Организации аппаратно-программного мультипрограммирования

Вторичные нити создаются и исполняются в контексте своего процесса-хозяина. Каждая нить представляется дескриптором, содержащим свой идентификатор TId и идентификатор PId своего процесса-хозяина, к пространству которого относятся вырабатываемые архитектурными командами исполнительные адреса. В зависимости от режима работы устройства управления памятью – со включенными средствами трансляции адресов или без - исполнительные адреса являются либо виртуальными либо физическими.

В общем случае в динамике работы ВС возникает показанная на рисунке 1 ситуация исполнения NT программных нитей на nt аппаратных. Программное мультиплексирование, аппаратно поддерживаемое устройством управления прерываниями (от таймера и устройств ввода-вывода), обеспечивают перезарузку (свопинг) дескрипторов программных нитей на аппаратных с накладными расходами, зависящими от архитектуры вычислительной системы. Если загружаемая нить принадлежит процессу, дескриптор которой не загружен в один из np дескрипторов, возникают дополнительные накладные расходы. Например, в широко распространенной архитектуре SPARC V9 [14] используется большой, около 4 килобайт, регистровый файл, обеспечивающий высокую скорость исполнения процедур - при одинаковой тактовой частоте почти втрое высшую, чем в архитектуре Intel. Платой за высокую скорость являются увеличенные в десятки раз накладные расходы, становящиеся неприемлемыми при высоком темпе мультиплексирования.

Суммирую сказанное в данном разделе, можно утверждать, что по существу между инженерами-схемотехниками и системными программистами сформировался показанный на

рисунке 1 Силиконовый Занавес со шлюзами в виде дескрипторов аппаратных нитей и контекстов. Работа по встречному перемещению (свопингу) программных нитей и процессов через эти шлюзами является основной причиной обобщенной латентности, снижающей КПД существующих вычислительных систем при исполнение параллельных вычислений общего назначения.

По-видимому, длительное существование Силиконового Занавеса объясняется двумя основными факторами:

– языковым барьером между инженерами-схемотехниками, до начала 90х годов использовавшими только схемное описание аппаратуры, и системными программистами, использующими текстовое представление;

- высокой стоимостью средств макетирования вычислительных систем, имеющих полностью открытый исходный код на языках описания аппаратуры.

Языковый барьер в настоящее время практически исчез - автор имеет большой опыт использования низкоуровневых языков для системного программирования - по собственному выбору ассемблер для БЭСМ-6, младших моделей ЕС-ЭВМ и PC-AT и по заказу forth для доработки BIOS SPARC V8. Этот опыт и опыт моделирования и макетирования систем на кристалле позволяет утверждать, что программирование на языке verilog гораздо комфортнее, чем на названных выше языках (особенно forth). Чтобы стать специалистом по логическому программированию систем на кристалле, студенту, имеющему опыт разработки и отладки параллельных программ на языке си, потребуется немного усилий, чтобы привыкнуть к разнице между передачей информации через программные переменные и аппаратные объекты – провода и триггеры. Второй фактор в настоящее время стал неактуальным в связи с появлением недорогих свободных проектов систем на кристалле СнК, например упомянутого MIPSfpga [13], который может быть реализован на недорогой макетных платах.

Представляется, что включение в учебный процесс подготовки инженеров и системных программистов совместных проектов реализации BTA на свободных для академического использования средствах моделирования и макетирования позволит создать новую междисциплинарную специальность с высокой конкурентоспособностью на современном рынке. В следующем разделе представлено концептуальное описание реализации BTA на основе базе оригинальной микроархитектуры, названной транзакционной (TMA). Ее исследование и реализация позволит новым специалистам перенести Силиконовый Занавес в самое подходящее место - перед “Всадниками Апокалипсиса”.

Архитектура виртуалтредингoвой системы на кристалле

Прямое аппаратное мультипрограммирование

Как показано на рисунке 1, в конвейерах существующих микроархитектур исполняются только команды, принимаемые на вход устройств выборки из представленного дескрипторами фиксированного множества аппаратных нитей (для сокращения объема и сложности настоящей ной статье управление контекстами не рассматривается). В TMA общепринятое грубо гранулированное одноуровневое представление нитей аппаратными дескрипторами заменено показанным на рисунке 2 двухуровневым представлением. Все программные нити, создаваемые архитектурными командам BTA, постоянно представлены в директории дескрипторов нитей NT вхождениями-дескрипторами, состав полей которых описан в предыдущем разделе. В отличие от сосредоточенного регистрового представления в существующих микроархитектурах, дескриптор нити распределен по уровням системной виртуальной памяти, начиная с регистров, содержащий его корень (на рисунке выделен рамкой) до кэшей и оперативной памяти, в которые могут перемещаться отдельные архитектурные регистры.

Функцию устройств выборки команд реализует монитор нитей, число входных каналов выборки которого выбирается в зависимости от максимального числа нитей NT. Архитектурные команды, как в известных архитектурах, распределены по нескольким уровням памяти от кэша уровня 1 до оперативной памяти. Исполнение нитей реализуется по двухуровневой схеме. На первом уровне монитор нитей в каждом процессорном такте анализирует готовность к исполнению и приоритет полного множества нитей. По результатам анализа для каждой готовой к исполнению нити он выбирает очередную архитектурную команду и формирует на ее основе первичную транзакцию, которую выдает через перекрестный коммутатор в пул исполнительных устройств. Как показано на рисунке 2, транзакция наследует все атрибуты нити-хозяина – ее идентификатор, принадлежность процессу-хозяину и приоритет, а также образ архитектурной команды.

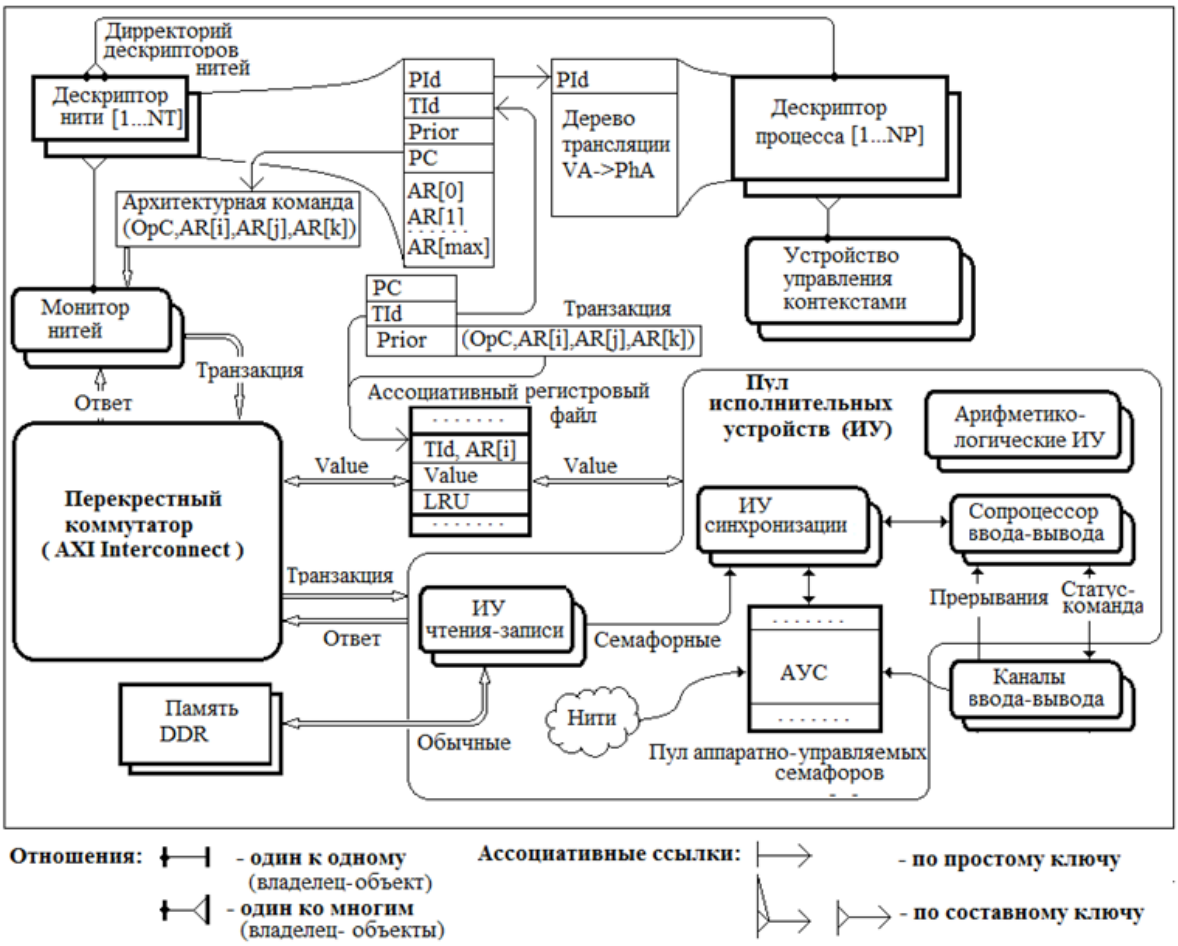


Рис.2. Организации СнК с прямым аппаратным мультипрограммированием

Соответствующее коду операции исполнительное устройство отрабатывает транзакцию и при этом может привлекать другие исполнительные устройства, передавая им вторичные транзакций. Устройство, отработавшее первичную транзакцию, возвращает результат завершения в выдавший ее монитор нитей. В ТМА используется тонко-гранулированное распределенное представление ассоциативного регистрового файла, прототип которого, ориентированный на работу с фиксированным множеством аппаратных нитей, описан в работе [15]. Первичные и вторичные транзакции в качестве операндов и результатов используют его структурированные элементы-регистры как показано на рисунке 2. Тэг элемента состоит из поля идентификатора нити-хозяина представляемого архитектурного регистра TId и поля iAR, содержащего его индекс в архитектурном регистровом файле. Поле Value содержит текущее значение. Поле LRU содержит счетчик обращений за заданный интервал времени и используется для перемещения элемента по уровням системной виртуальной памяти.

По существу монитор, каждое исполнительное устройство и элемент ассоциативного регистрового файла реализуют распределенное тонко гранулированное приоритетное мультипрограммирование на уровне транзакций, которое на архитектурном уровне оно приводит к покомандному продвижению вычислений. В зависимости от желаемой скорости исполнения отдельной нити монитор может использовать так и простое поочередное исполнение команд, так и ускоренное используя суперскалярную технику.

В следующем разделе приведено описание команд аппаратной синхронизации ВТА и их реализации в транзакционной микроархитектуре.

Синхронизация на основе аппаратно-управляемых семафоров

Средства синхронизации в ВТА основаны на использовании аппаратно-управляемых семафоров (АУС), объединяющих функции мутекса и условной переменной современных операционных систем [12]. Дополнительно они реализуют функцию локального сторожевого таймера для контроля времен работы в критической секции и ожидания входа в нее. АУС

организован как активный компонент микроархитектуры, содержащий показанные на рисунке 3 локальные регистры:

TIdCS - идентификатор нити, находящейся в критической секции (КС);
TOutCS - таймаут времени пребывания нити в КС;
PriorCS- временный (повышенный) приоритет при работе нити в КС;
MxQ - очередь запросов нитей к мутексу данного АУС;
CvQ - очередь запросов нитей к условной переменной данного АУС;
TLat - таймаут ожидания входа в КС по описываемым ниже командам SLOck и Swait.

Синхронизация нитей поддержана пятью архитектурными командами SGet, SFree, SLock, SWait и SPas. Следует подчеркнуть, что реализуемые АУС локальный, не использующий прерываний системного таймера, контроль связанных с критическими секциями длительностями и повышение приоритета нити при работе в ней позволяет полностью снять проблему инверсии приоритетов [16].

Команд SGet(t,pr), параметры которой задают поля TOutCS и PriorCS, выделяет выдавшей нити адрес-указатель АУС, обозначаемый далее идентификатором ps, подаваемый в как параметр в остальные четыре команды. Возврат освободившегося АУС выполняет команда Sfree(ps).

Команда SLock(ps,ts) реализует функцию системного вызова pthread_mutex_lock и обеспечивает вход нити в критическую секцию для работы с приоритетом PriorCS в течении времени TOutCS либо отказ входа, если время его ожидания превысило заданное операндом ts. Идентификации этих случаев обеспечивает возвращаемый результат.

Команда SWait(ps,ts) является аналогом вызова pthread_cond_timedwait и реализует неактивное ожидание входа в КС до выдачи команды SPas(ps), позволяя устранить ресурсоемкий опрос разделяемых переменных в критической секции.

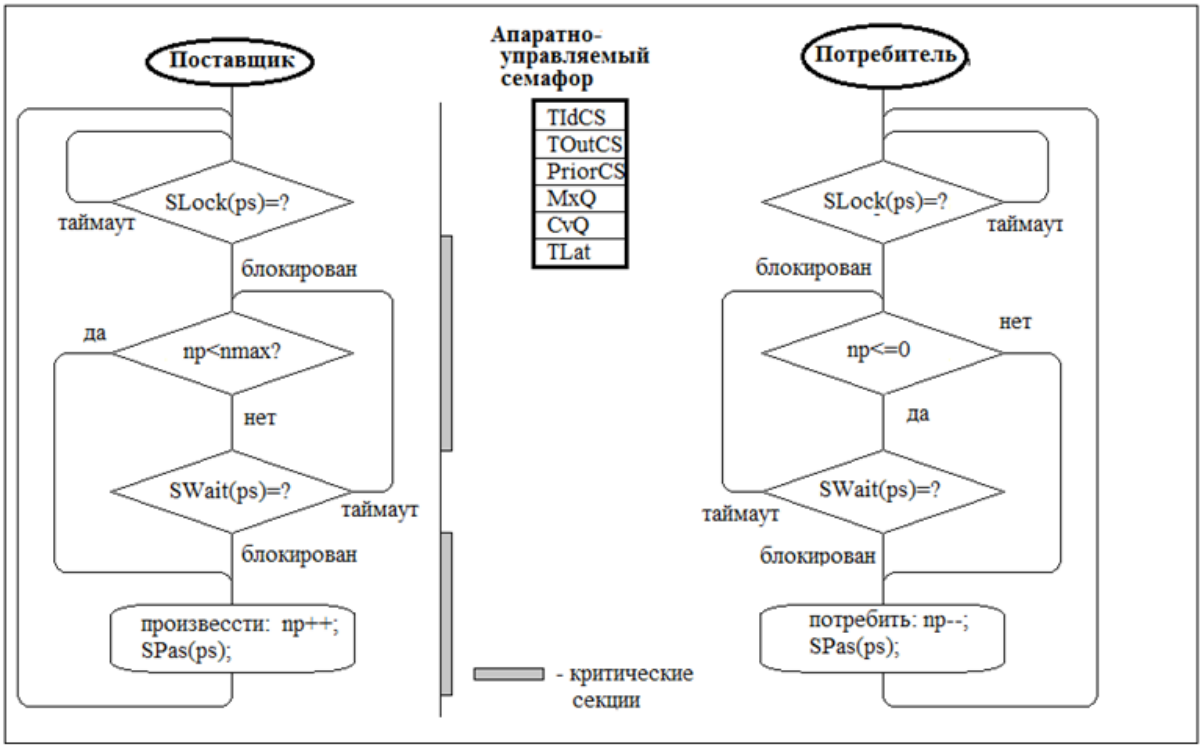


Рис 3. Организации синхронизации с использованием аппаратно управляемых семафоров

Команда SPas(ps) объединяет действие системных вызовов pthread_mutex_unlock и pthread_cond_broadcast, выводит нить из критической секций, охраняемой семафором ps и освобождает ее для одной из других конкурирующих нитей.

В следующем разделе приведен пример, иллюстрирующий применение рассмотренных средств для организации ввода-вывода.

Ввод-вывод: Interruption considered harmful

Как нетрудно убедиться, транзакционная микроархитектура для реализации тонко гранулированного мультипрограммирования, включая аппаратную синхронизацию с

таймированием связанных с критическими секциями длительностей, не использует асинхронных прерываний. Рассмотренные средства синхронизации ВТА в сочетании с эффективной поддержкой большого количества нитей позволяют выделить каждому каналу ввода-вывода, поддерживающему независимую активность внешних устройств, собственную программную нить. Такая нить, называемая нитью ввода-вывода, создается обычной нитью, называемой программной нитью и работает с максимальным приоритетом. Нить ввода-вывода реализует с одной стороны жесткое временное взаимодействие с аппаратурой ввода-вывода, а с другой мягкое взаимодействие с программной нитью по схеме поставщик-потребитель, используя описанные в предыдущем подразделе аппаратные семафоры.

Рисунок 3 на примере реализации алгоритма поставщик-потребитель иллюстрирует семантику средств синхронизации с использованием АУС, которая подходит как для взаимодействия вычислительных нитей между собой так и для взаимодействия с нитями ввода-вывода. Программная нить запрашивает АУС командой SGet, заносит его в доступную нити ввода-вывода переменную, инициализирует общие переменные и запускает нить ввода-вывода. Глобальная переменная np с начальным значением 0 показывает текущее количество заполненных блоков, которое не должно быть больше nmax. Нить-поставщик (при выводе информации это программная нить, а при вводе нить ввода вывода), находясь в критической секции, в зависимости от значения разделяемой переменной np, либо поставляет информацию, увеличивая np, и выходит из КС, выдав инструкцию SPas(ps), либо ожидает освобождения свободного места, перейдя по инструкции SWait(ps) в состояние неактивного ожидания в очереди к условной переменной с освобождением КС. Нить-потребитель либо использует поставленную информации (уменьшает np на рисунке), либо переходит в состояние ожидания, если ее нет.

Традиционной ресурсоемкая техника реализации ввода-вывода, применяемая даже в ОС мягкого реального времени 16, использует программные обработчики прерываний и нити-посредники, которые являются объектами грубо гранулированного мультипрограммирования используют все ресурсы аппаратной нити и, в худшем случае однонитевой архитектуры, ядра в целом. В результате большой объем аппаратуры во время обработки поддерживающих ввод-вывод прерываний будет находиться в состоянии обобщенной латентности, снижающей КПД вычислительной системы. В отличие от нее в ВТА заменяющая всех посредников нить ввода-вывода исполняется как поток транзакций, использующих только необходимый минимум тонко гранулированных элементов аппаратуры оставляя их основной объем для исполнения вычислительных нитей.

В зависимости от открытости исходного кода устройств, реализованных на платах разработки СнК, нить ввода вывода может исполняться либо на процессорном ядре либо на показанном на рисунке 2 сопроцессоре ввода-вывода, имеющем простейшую систему команд, достаточную для взаимодействия с аппаратурой ввода-вывода и плюс три команды синхронизации SLock, SWait и SPas, отрабатываемые исполнительным устройством синхронизации. Поступающие от внешних устройств сигналы по существу являются только запросами внимания, а делает их прерываниями только некоторые архитектуры вычислительных систем.

Достаточно давно, в суперкомпьютерах использовались сервисные процессоры (CDC 6600 [17], 1960) и процессоры ввода вывода (Эльбрус-2 [18], 1980 год). Современные многонитевые суперкомпьютеры для высокопроизводительных вычислений общего назначения являются экстремально многонитевыми (Tera MTA [19], Cray XMT [20]) и вместо прерываний используют опрос выделенной нитью. Соответственно, в рамках учебных проектов реализации ВТА целесообразно отработать макро и микроархитектуру архитектуру сопроцессора ввода-вывода, который бы мог работать с основными современными устройствами USB, Ethernet, SATA и видеоадаптерами.

Завершая раздел, следует подчеркнуть что понятие “прерывание” не используется ни в ВТА, ни в транзакционной микроархитектуре – соответсвенно, можно утверждать, что в новых архитектуры мощных компьютеров interruption considered harmful.

Заключение

Параллельные вычислений общего назначения становятся доминирующими по объему средств, вкладываемых вычислительных системы для их исполнения. Для таких систем очень актуальным остается старый вызов Сеймура Крэя [21] “Каждый может построить быстрый процессор. Фокус в том, чтобы построить быструю систему". Важна также оценка кластерных систем Бартона Смита [22] в отношении таких вычислений “Современные микропроцессоры также

плохи для параллелизма общего назначения, как и динозавры, которые на них охотятся. Если новые капитаны компьютерной индустрии не будут готовы к архитектурным усовершенствованиям, они вскоре осознают тяжесть пути перестающих плыть против течения".

Представляется, что достаточно важные усовершенствования вносит предложенная в настоящей статье виртуалтредингова архитектура. Современные средства автоматизации проектирования, открытые в исходных текстах проекты систем на кристалле и недорогие платы для их макетирования позволяют создать новую междисциплинарную специальность, включив в учебный процесс разработку совместных проектов усовершенствования доступных архитектур средствами виртуалтрединга. Это позволит подготовленным таким образом специалистам, как и пользователям их разработок, избежать судьбы динозавров и обеспечить длительную эволюции в параллельном мире.

Литература

1. Burton Smith, Reinventing Computing Microsoft Research Faculty Summit 2007 http://www.cct.lsu.edu/~estrabd/LACSI2006/Smith.pdf.
2. Burton Smith, The Quest for General Purpose Parallel Computing 1994 Developing a computer science agenda for high-performance computing ACM New York, NY, USA ©1994
3. Операционные системы реального времени http://asutp.ru/?p=600591.
4. T. Sterling. Critical Factors and Directions for Petaflops-scale Supercomputers. California Institute of Technology, NASA Jet Propulsion Laboratory, Presentation to IFIP WG10.3 e-Seminar Series. 4 January 2005.
5. Ефимов А.И. Машина с виртуальным мультипроцессированием: новая архитектура параллельных вычислений общего назначения., Вопросы радиоэлектроники, серия ЭВТ, выпуск 2, 2008 Электронная копия: http://www.mcst.ru/publikacii-mcst-s-2004goda.
6. Ефимов А.И. Двумерная виртуализация процессора: новый подход к повышению реальной эффективности суперкомпьютеров http://www.ospcon.ru/files/media/Efimov.pdf.
7. E.-W. Dijkstra, Cooperating sequential process, In F.Genuys, editor, Programming Languages. Academic Press, New York , 1968.
8. Оверлеи http://en.wikipedia.org/wiki/Overlay_(programming)
9. T. Kilburn, D. Edwards, M. Lanigan, and F. Sumner. One-level storage system. IRE Trans. Elect. Computers, 37:223--235, 1962. http://www.eecs.harvard.edu/cs261/papers/kilburn61.pdf
10. Cray MTA-2 https://ru.wikipedia.org/wiki/Cray_MTA-2
11. SPARC T5-2 SERVER http://www.oracle.com/us/products/servers-storage/ servers/sparc/oracle-sparc/t5-2/sparc-t5-2-ds-1922871.pdf
12. Love, Robert. Linux kernel development / Robert Love. — 3rd ed.
13. MIPSfpga programme opens up the MIPS architecture to universities worldwide http://blog.imgtec.com/mips-processors/mipsfpga-opens-up-the-mips-architecture-to-universities-worldwide.
14. https://www.fujitsu.com/hk/Images/JPS1-R1.0.4-Common-pub.pdf SPARC® Joint Programming Specification (JPS1):Commonality Sun Microsystems and Fujitsu Limited.
15. How to Fake 1000 Registers, Proceedings of the 38th annual IEEE/ACM International Symposium on Microarchitecture http://aggregate.ee.engr.uky.edu/LAR/micro05.pdf.
16. Eleventh real-time Linux workshop, Dresden, 2009 http://lwn.net/images/conf/rtlws11/papers/proceedings_2009.pdf.
17. CDC 6600 https://en.wikipedia.org/wiki/CDC_6600.
18. МВК "Эльбрус-2" http://www2.icmm.ru/~masich/win/lexion/elbrus2/elbrus2.htm
19. The Tera Computer System. Robert Alverson, David Callahan, Daniel Cummings, Brian Koblenz, Allan Porterfield, and Burton Smith ACM International Conference on Supercomputing, pp. 1-6, June 1990 https://en.wikipedia.org/wiki/Cray_XMT.
20. Seymour Cray – A Man Whose Vision Changed the World www.cray.com/Assets/PDF/about/SeymourCray.pdf
21. Burton Smith, The end of architecture, ACM SIGARCH Computer Architecture News Homepage archive Volume 18 Issue 4, Dec. 1990, Pages 10-17.

Голубева Я.В.

Московский государственный университет им. М.В.Ломоносова, г.Москва, бакалавр факультета Вычислительной математики и кибернетики, golubeva.yana.v@gmail.com

РАЗРАБОТКА И ИССЛЕДОВАНИЕ АЛГОРИТМОВ БАЛАНСИРОВКИ НАГРУЗКИ В ПАРАЛЛЕЛЬНОЙ РЕАЛИЗАЦИИ МЕТОДА ВЕТВЕЙ И ГРАНИЦ

КЛЮЧЕВЫЕ СЛОВА

Метод ветвей и границ, параллельная реализация, алгоритмы балансировки нагрузки, анализ производительности алгоритмов.

АННОТАЦИЯ

Статья посвящена проблеме балансировки нагрузки процессоров на примере параллельной реализации метода ветвей и границ. Рассмотрены стратегии балансировки нагрузки, разработан адаптивный алгоритм балансировки нагрузки с фиксированными и изменяемыми параметрами, проведено экспериментальное исследование двух алгоритмов балансировки, проанализированы параллельные ускорение, эффективность и масштабируемость исследованных алгоритмов. Выявлены причины потерь производительности.

Одним из основных методов решения задач глобальной оптимизации является метод ветвей и границ. Идея этого метода заключается в том [1], что множество допустимых решений разбивается на подмножества, и те подмножества (подзадачи), в которых оптимальных решений не существует, отбрасываются. Метод ветвей и границ имеет древовидную структуру, где вершины – это полученные в результате разбиений подзадачи (см. рис. 1). Вершина, отмеченная буквой Т – это отбрасываемая вершина, а вершины, отмеченные буквой К – это задачи-кандидаты, которые могут привести к оптимальному решению.

Такая структура метода создает благоприятные условия для его параллельной реализации, так как поиск оптимальных решений в отдельных ветвях этого дерева может производиться независимо на разных вычислительных узлах. В то же время при параллельной реализации возникают определенные сложности. В частности, возникает проблема балансировки нагрузки. Она заключается в том, что невозможно статически равномерно распределить нагрузку между процессорами, так как дерево является неоднородным и его структура не известна заранее. Это может привести к тому, что в определенный момент времени одни процессоры уже закончат выполнять разветвления и будут простаивать, а другие всё еще будет продолжать решать выделенные им подзадачи.

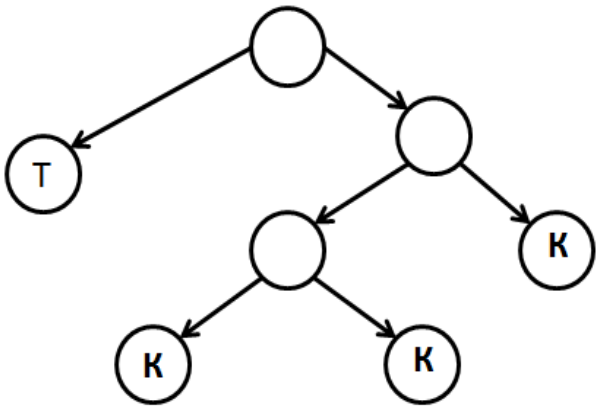


Рис. 1. Древовидная структура Метода ветвей и границ

Таким образом, возникает задача балансировки нагрузки – перераспределения

вычислительной нагрузки между параллельно работающими процессорами. Актуальность данной задачи подтверждается тем, что в современных вычислительных устройствах число процессоров растет, а чем больше процессоров – тем сложнее балансировать их нагрузку.

Краткий обзор методов балансировки нагрузки

В работах [2,3] рассмотрен достаточно простой алгоритм балансировки нагрузки. Балансировка нагрузки в этом алгоритме централизована, так как среди всех рабочих процессов (РП) выделен управляющий процесс (УП). На рис. 2 изображена упрощенная схема работы алгоритма.

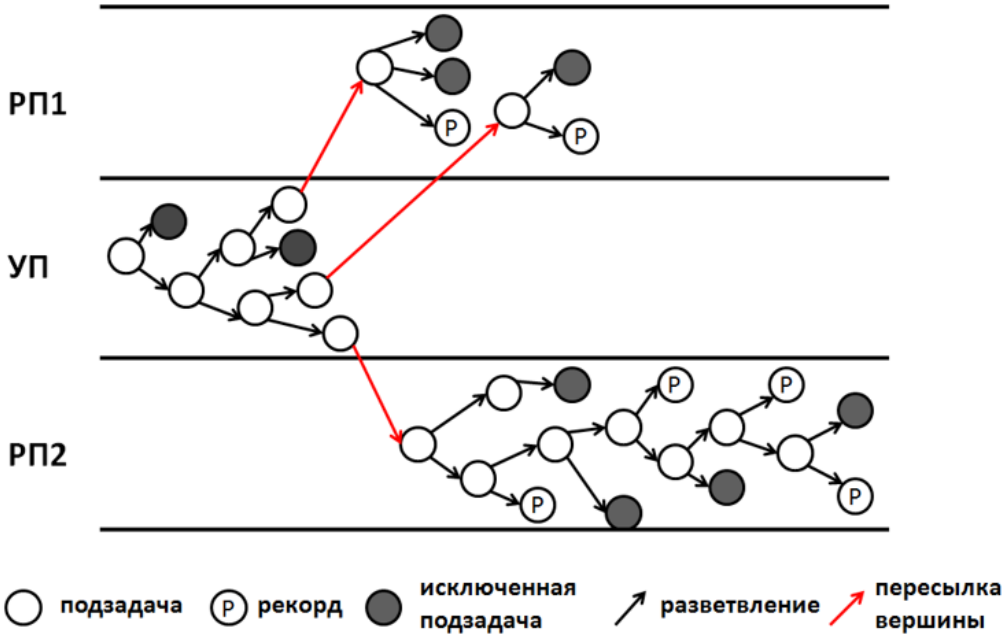


Рис. 2. Схема стандартного алгоритма параллельной балансировки нагрузки

На начальном этапе УП выполняет заранее определенное количество разветвлений, генерируя некоторое количество подзадач, заведомо большее, чем количество рабочих процессов. Далее сгенерированные подзадачи распределяются между рабочими процессами, которые решают их до конца. Если рабочий процесс решил задачу, а на управляющем процессе еще имеются другие подзадачи – то освободившийся рабочий процесс получает новую подзадачу для решения. Полученные в процессе решений рекорды – наилучшие найденные решения – РП отправляет УП, который, сопоставляя полученные результаты, выбирает оптимальный.

В работе [4] рассматриваются 2 параллельных метода обхода дерева, которые можно использовать в алгоритме централизованной балансировки нагрузки:

- *Метод назначаемых поддеревьев* – заключается в разбиении дерева на большое число «маленьких» поддеревьев и назначении их для обхода в порядке освобождения процессов. На первом этапе управляющий процесс осуществляет обход дерева в ширину до тех пор, пока число вершин в очереди не достигнет заранее заданного числа, заведомо намного большего, чем число процессоров в системе. На втором этапе управляющий процесс рассылает рабочим процессам вершины из очереди, и они осуществляют их обход в глубину;
- *Метод выделяемых поддеревьев* – отличается от предыдущего тем, что разбиение на подзадачи происходит не сразу, а по мере выполнения обхода дерева. На первом этапе управляющий процесс осуществляет обход дерева в ширину до тех пор, пока число вершин в очереди не достигнет заранее заданного числа, равного или немного большего, чем число процессоров в системе. На втором этапе управляющий процесс рассылает рабочим процессам вершины из очереди, и они осуществляют их обход в глубину. По окончании обхода процесс освобождается. Для его загрузки выбирается поддерево, обход которого в этот момент ведется одним из процессов, и в нем выделяется поддерево, которое передается для обхода освободившемуся процессу.

В работе [5] рассматривается статическая и динамическая стратегии балансировки нагрузки. В обеих стратегиях сначала генерируется некоторое число подзадач. Статическая

стратегия подразумевает хранение всех сгенерированных и нерешенных подзадач в общем рабочем пуле. Каждый процессор последовательно решает свою подзадачу до конца, используя свой локальный рабочий пул. Динамическая стратегия подразумевает обмен подзадачами между локальными рабочими пулами во время выполнения программы.

Авторы выделяют 3 класса динамических стратегий:

- *Стратегия балансировки по запросу (Strategy on request)* – процессор с пустым (или с почти пустым) рабочим пулом отправляет запрос другому процессору на получение новых подзадач. Запрос может быть одобрен – в этом случае подзадачи пересылаются из одного рабочего пула в другой – или отклонен;
- *Балансировка без запроса (Strategy without request)* – процессоры рассылают часть своих подзадач другим процессорам без запроса с их стороны;
- *Комбинированная стратегия (Combined strategy)* – комбинация двух предыдущих стратегий.

В работе [6] описана общая модель динамической балансировки нагрузки, которая состоит из четырех этапов:

- *Оценка загрузки процессора (Processor Load Evaluation)* – на этом этапе производится оценка нагрузки на каждом процессоре в системе и выявление разбалансированности;
- *Определение выгоды балансировки нагрузки (Load Balancing Profitability Determination)* – на этом этапе определяются накладные расходы, которые повлечет балансировка нагрузки, и принимается решение, будет ли выгодна балансировка нагрузки в данный момент времени;
- *Стратегия перемещения подзадач (Task Migration Strategy)* – здесь определяются источники и получатели подзадач для решения;
- *Стратегия выбора подзадач (Task Selection Strategy)* – на заключительном этапе происходит отбор и отправка подзадач от процессора с высокой нагрузкой процессору с низкой нагрузкой.

В данной модели авторы выделяют следующие 5 стратегий балансировки нагрузки:

1. *Градиентная модель (The Gradient Model)*. Стратегия заключается в том, что процессоры с низкой нагрузкой оповещают другие процессоры о своем статусе, и перегруженные процессоры отвечают отправкой некоторого числа подзадач соседним, менее нагруженным процессорам в системе. Подзадачи перемещаются до тех пор, пока нагрузка соседних процессоров меньше нагрузки процессора-отправителя;
2. *Метод распределения, инициированного отправителем (Sender Initiated Diffusion)*. Стратегия заключается в том, что процессор, при получении информации о том, что среди соседних процессоров присутствуют такие, нагрузка которых опустилась ниже некоторого порога, инициирует отправку излишков малонагруженным процессорам;
3. *Метод распределения, инициированного получателем (Receiver Initiated Diffusion)*. Стратегия отличается от предыдущей тем, что отправку излишков инициируют малонагруженные процессоры;
4. *Метод иерархической балансировки (Hierarchical Balancing Method)*. В этой стратегии процессоры выстраиваются в дерево, которое подвергается балансировке нагрузки. Балансировка вызывается на разных уровнях получением сообщения о разбалансированности между процессорами, находящимися одним уровнем ниже. Если степень разбалансированности превышает некоторый порог, то на этом уровне происходит передача излишка подзадач от более нагруженного процессора к менее нагруженному;
5. *Метод измерений (The Dimension Exchange Method)*. Стратегия заключается в том, что N процессоров представляется в виде logN измерений. Сначала балансируются между собой пары процессоров, находящиеся в первом измерении (это такие процессоры, у которых адреса отличаются только последним значащим битом). Далее балансируются между собой пары процессоров, находящиеся во втором измерении и так далее, пока каждый процессор не сбалансирует свою нагрузку со всеми своими соседями.

Исследование алгоритмов балансировки нагрузки

Для исследования был выбран алгоритм, предложенный в работах [2,3], приведенный в начале обзора. Эксперименты проводились с помощью BNB-Simulator [7] и BNB-Visualizer [8]. Эти приложения позволяют проводить исследования на эмуляторе параллельной системы.

На рис. 3 представлен график загруженности процессоров, полученный с помощью BNB-Simulator и BNB-Visualizer. На горизонтальной оси отложены единицы времени, на вертикальной – число процессоров. В каждую единицу времени показано, сколько процессоров занято (отмечены синим) и сколько простаивает (отмечены зеленым), ниже рассчитаны параллельное ускорение и эффективность.

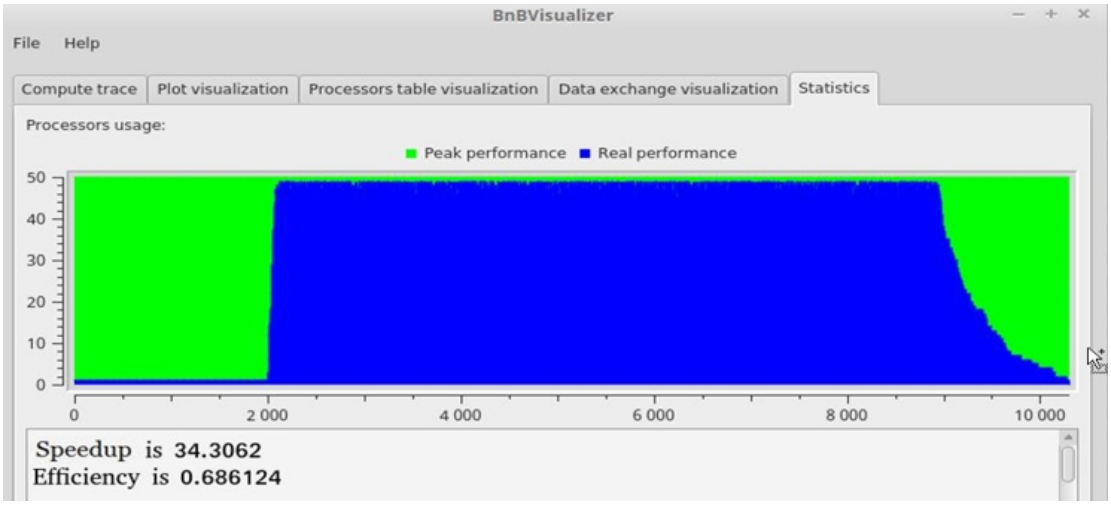


Рис. 3. График загруженности процессоров при выполнении стандартного алгоритма параллельной балансировки нагрузки

На данном графике заметен существенный недостаток исследуемого алгоритма: одни рабочие процессы уже завершили расчеты и простаивают, ожидая тех, кому достались более сложные подзадачи для решения. Также на графике отчетливо виден еще один минус данного алгоритма – достаточно много времени занимают последовательные действия на первоначальном этапе.

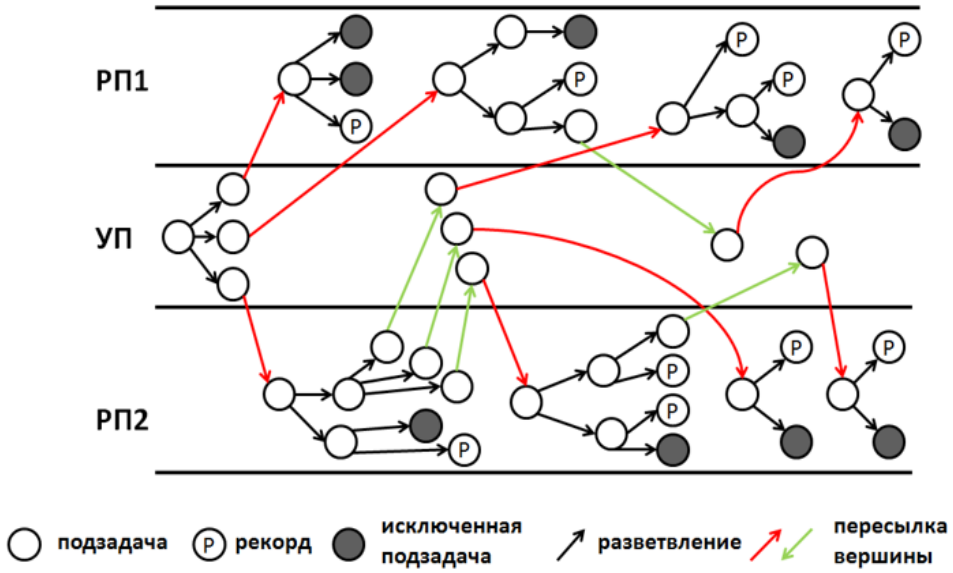


Рис. 4. Схема адаптивного алгоритма параллельной балансировки нагрузки

Чтобы преодолеть эти недостатки, реализован улучшенный вариант рассмотренного алгоритма – адаптивный алгоритм балансировки нагрузки в двух вариантах: с фиксированными и с изменяемыми параметрами.

Основное отличие адаптивного алгоритма заключается в том, что рабочий процесс не всегда решает полученную подзадачу до конца, а выполняет заданное количество ветвлений (определяемое параметром T) и отправляет полученные рекорды и подзадачи управляющему процессу.

На рис. 4 изображена упрощенная схема его работы.

Таким образом, у управляющего процесса отпала необходимость делать большое

количество ветвлений на этапе первоначальных итераций. Во избежание переполнения памяти УП число вершин на нем регулируется пороговыми значениями – M и m и управляющим параметром S, определяющим количество подзадач, отправляемых рабочими процессами управляющему процессу. Если на управляющем процессе число подзадач становится больше или равно M – то параметр S устанавливается равным нулю, и рабочие процессы отправляют УП только рекорды, если они образовались в процессе вычислений. Когда на управляющем процессе остается меньше m вершин – устанавливается значение параметра S равное изначально заданной константе S_0, и отправка подзадач управляющему процессу возобновляется.

Однако, проблема несбалансированности на завершающем этапе расчета всё еще остается, как видно из графика загрузки процессоров, изображенного на рис. 5. Тем не менее, эффективность алгоритма возрастает в связи сокращением времени выполнения последовательных действий (1000 единиц времени против 2000 в предыдущем алгоритме).

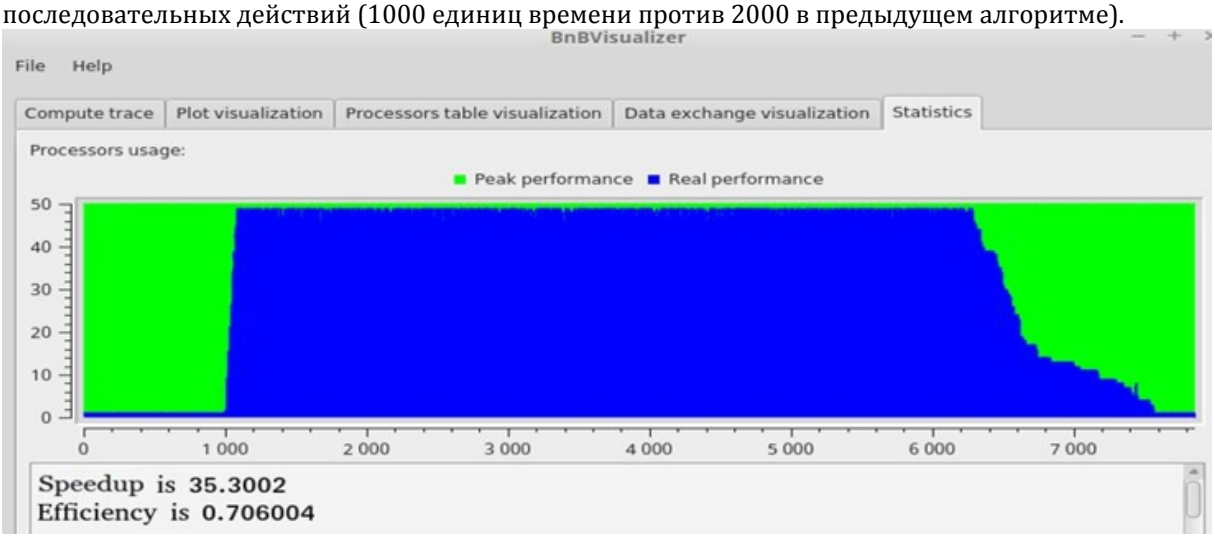


Рис. 5. График загруженности процессоров при выполнении адаптивного алгоритма параллельной балансировки нагрузки с фиксированными параметрами

Для решения обозначенной проблемы адаптивный алгоритм был модернизирован – параметры T и S стали изменяемыми. Параметр S стал зависеть от числа уже имеющихся вершин на УП и обеспечил постепенное заполнение его памяти. Параметр T также стал зависеть от числа вершин на УП. Если количество последних становится меньше числа рабочих процессов – то параметр T сокращается в 10 раз и обеспечивает более частый обмен вершинами между управляющим и рабочими процессами.

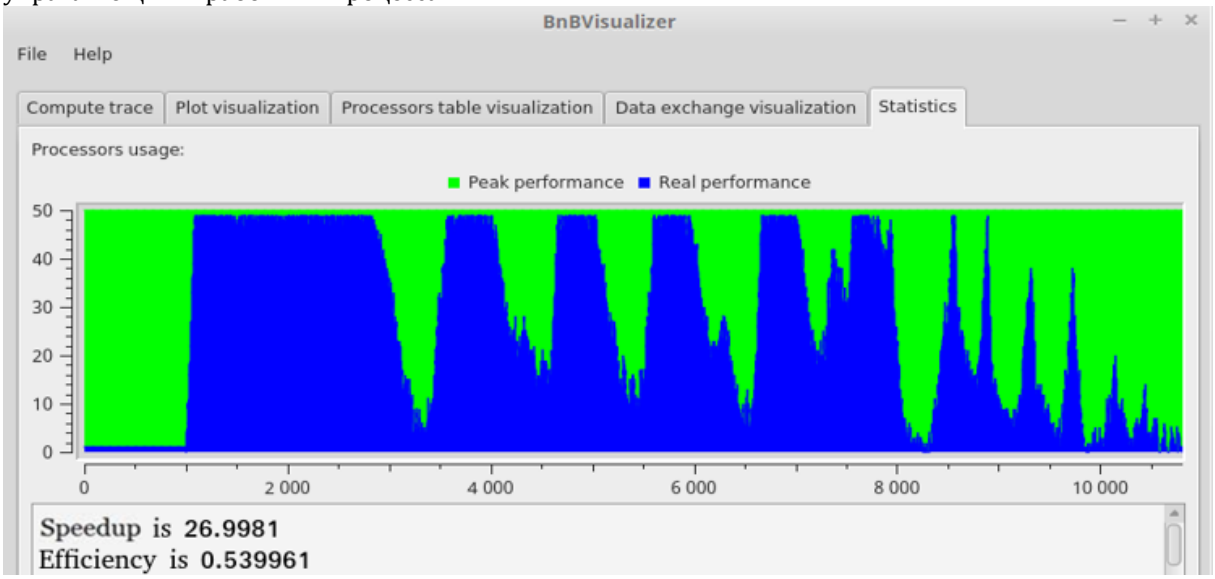


Рис. 6. График загруженности процессоров при выполнении адаптивного алгоритма параллельной балансировки нагрузки с изменяемыми параметрами. Пример падения производительности, вызванного увеличением накладных расходов на коммуникации

Интересно отметить, что на начальных этапах исследования была предпринята попытка сохранять малые значения числа ветвлений РП на протяжении всего времени работы программы. Это привело к увеличению накладных расходов на осуществление коммуникаций между УП и РП, что можно увидеть на графике загрузки процессоров, изображенном на рис. 6.

«Провалы» демонстрируют перегрузку УП – он не успевает обеспечить подзадачами все РП.

В ходе проведения дальнейших экспериментов было установлено, что пока на управляющем процессе количество задач достаточно для обеспечения нагрузки всех рабочих процессов – нет смысла часто их прерывать. Необходимость уменьшения параметра T возникает при дефиците подзадач на управляющем процессе.

График, изображенный на рис. 7, подтверждает справедливость данного замечания: загруженность процессов на заключительном этапе увеличилась, увеличилась и эффективность данного алгоритма по сравнению с предыдущими.

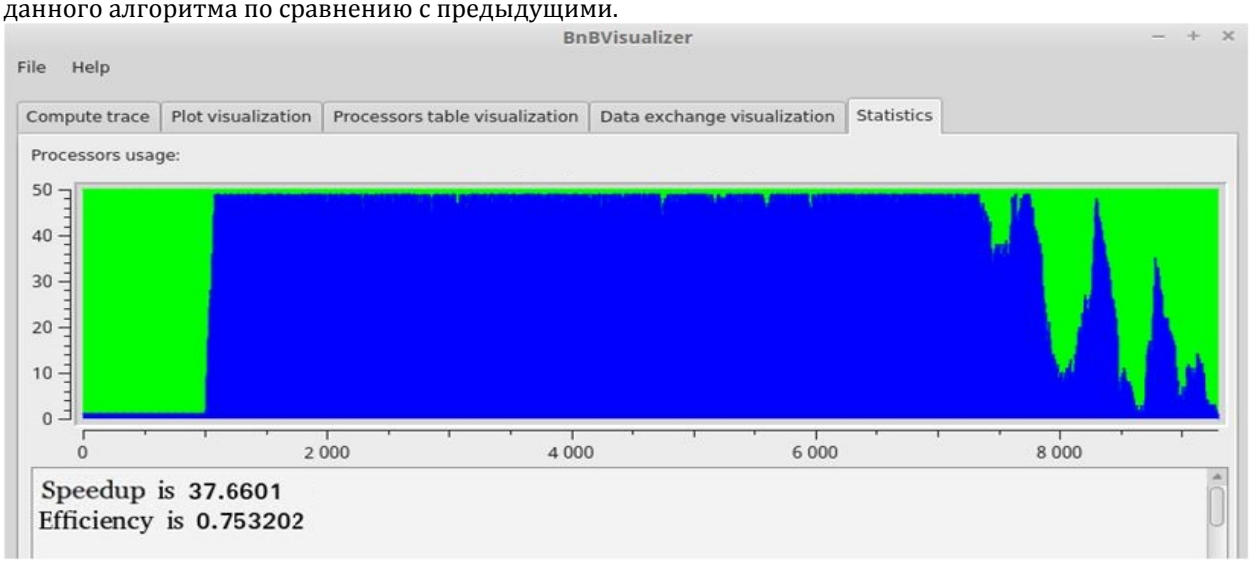


Рис. 7. График загруженности процессоров при выполнении адаптивного алгоритма параллельной балансировки нагрузки с изменяемыми параметрами. Пример правильно подобранных параметров

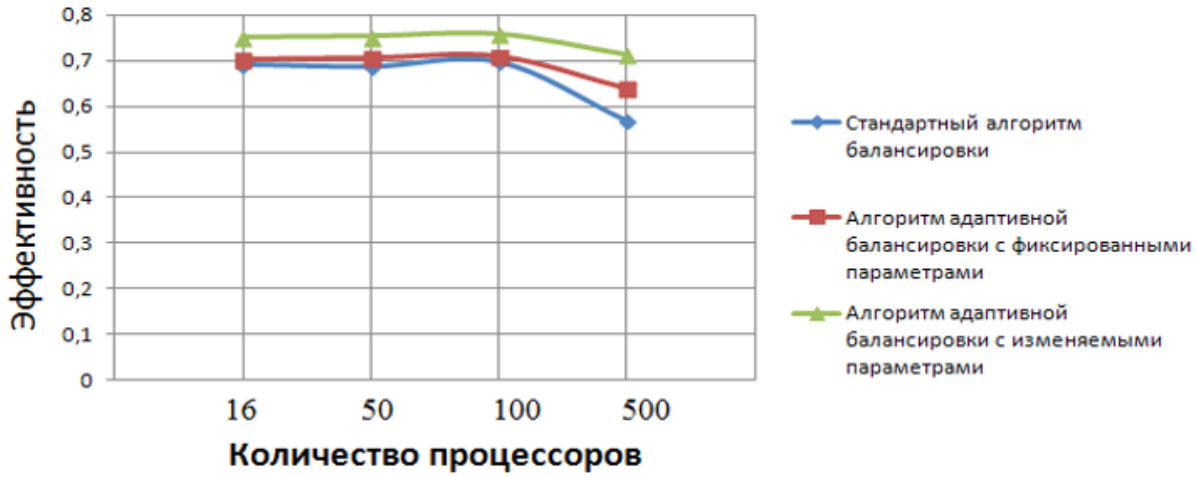


Рис. 8. График масштабируемости

Сравним рассмотренные алгоритмы на разных количествах процессоров и разных размерностях задачи. Было произведено исследование алгоритмов на 16, 50, 100 и 500 процессорах. Результат показан на графике масштабируемости (см. рис. 8).

Алгоритм считается масштабируемым, если существует такое соотношение между числом процессоров и размерностью решаемой задачи, при котором эффективность остается постоянной. При относительно небольшом числе процессоров все алгоритмы масштабируемы. Однако, на 500 процессорах эффективность всех алгоритмов заметно снижается. Меньше всего падение эффективности на 500 процессорах отмечено у алгоритма адаптивной балансировки нагрузки с изменяющимися параметрами.

Заключение

В данной статье рассмотрены некоторые алгоритмы балансировки нагрузки в параллельной реализации метода ветвей и границ, проведено экспериментальное исследование одного из предложенных алгоритмов и его улучшение, которое позволило повысить производительность.

На дальнейших этапах работы будет продолжено изучение алгоритмов балансировки нагрузки. Планируется реализация других алгоритмов балансировки нагрузки (в частности, перечисленных выше), их исследование (как на эмуляторе параллельной системы, так и на реальной системе), сравнение эффективности и анализ масштабируемости.

Литература

1. Сигал И.Х., Иванова А.П. Введение в прикладное дискретное программирование. М.: Физматлит, 2002.
2. М.А.Посыпкин, Архитектура и программная организация библиотеки для решения задач дискретной оптимизации методом ветвей и границ на многопроцессорных вычислительных комплексах// Труды ИСА РАН, Т. 25, с. 18-25.
3. Посыпкин М.А., Сигал И.Х. Исследование алгоритмов параллельных вычислений в задачах дискретной оптимизации ранцевого типа // Ж. вычисл. матем. и матем. физ. 2005. Т. 45. № 10. С. 1801-1809.
4. Н. Е. Тимошевская, Параллельные методы обхода дерева, Матем. моделирование, 2004, том 16, номер 1, 105-114.
5. Gendron B., Crainic T.G., Parallel Branch-and-Bound Algorithms: Survey and Synthesis// Operations Research, Vol. 42, No. 6 (Nov. - Dec., 1994), 1042-1066.
6. Willebeek-LeMair, M.H., Reeves, A.P., Strategies for dynamic load balancing on highly parallel computers// Parallel and Distributed Systems, IEEE Transactions on, Vol. 4, No. 9, September 1993, 979-993.
7. Andrey Fomin, Yuri Orlov, Mikhail Posypkin, Izrael Sigal, Using Simulation to Study Performance of Parallel Tree Search Schedulers // Proceedings of optimization and applications (Optima-2015) Petrovac, Montenegro, September 2015, p. 67.
8. Ю. В. Орлов, Среда комплексного анализа производительности алгоритмов балансировки в параллельном методе ветвей и границ// International Journal of Open Information Technologies ISSN: 2307-8162 vol. 3, no. 9, 2015, http://injoit.org/index.php/j1/article/view/228.

ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ В СИСТЕМАХ УПРАВЛЕНИЯ

Веремей Е.И.

Санкт-Петербургский государственный университет, г. Санкт-Петербург, д.ф.-м.н., профессор, заведующий кафедрой компьютерных технологий и систем, e_veremey@mail.ru

МАТЕМАТИЧЕСКИЕ МОДЕЛИ И МЕТОДЫ В ЦИФРОВЫХ ТЕХНОЛОГИЯХ УПРАВЛЕНИЯ ДВИЖЕНИЕМ МОРСКИХ СУДОВ

КЛЮЧЕВЫЕ СЛОВА

Морское судно, дискретное время, объект управления, обратная связь, устойчивость, стабилизация, цифровая фильтрация, оптимизация.

АННОТАЦИЯ

Рассматриваются вопросы математической формализации содержательных задач моделирования, анализа и синтеза систем управления движением морских судов с применением современных цифровых технологий. Актуальность темы определяется повсеместным применением современной компьютерной техники и программных средств при выполнении научно-исследовательских и проектно-конструкторских работ по созданию и практической реализации таких систем. В основу предлагаемого подхода положена концепция формирования цифровой обратной связи с многоцелевой структурой. Приводится числовой пример синтеза закона управления морским судном.

1. Введение

Для управления морскими подвижными объектами (МПО) в настоящее время применяются компьютерные системы, представляющие собой многоцелевые аппаратно-программные комплексы со сложной структурой, наделенные богатой функциональностью при работе в разнообразных режимах и обеспечивающие необходимые характеристики протекания соответствующих динамических процессов [1–6]. Требования к качеству управления, определяемые безопасностью и комфортом плавания, постоянно возрастают. С другой стороны, лавинообразно развиваются информационные технологии, предоставляя все новые возможности для построения управляющих комплексов с широчайшим привлечением современных математических методов на всех стадиях моделирования, исследования, разработки и практической реализации.

Характеризуя современное состояние цифровых технологий и их применения в процессах управления МПО, имеет определенный смысл выделение трех следующие стратегические линий, находящихся в стадии бурного развития:

– параллельные вычисления и суперкомпьютеры;

– облачные вычисления и технологии;

– встраиваемые вычисления и системы.

Очевидно, что все указанные линии имеют самый непосредственный выход на системы и процессы управления движением судов, хотя первые две из них принципиально отличаются от третьей. Это связано с тем, что они ориентированы на исключительно большие вычислительные ресурсы, предоставляемые пользователю, а третья линия, напротив, существенно ограничена предельно скромными вычислительными средствами. Это определяет значительные отличия информационной поддержки, которая используется в соответствующих системах специалистами по управлению.

Тем не менее, следует отметить, что применение современных компьютерных технологий в любом варианте при создании систем управления, в первую очередь, определяется необходимостью обеспечения требуемой функциональности, что достигается на базе формализованных (математических) подходов на этапе исследовательского проектирования. Главная цель всего комплекса работ, которые выполняются на этом этапе, состоит в формировании математических моделей управляющих устройств или законов управления, обеспечивающих желаемую динамику для систем, работающих во всех возможных режимах плавания.

Вовлечение современных информационных и компьютерных технологий в сферу

управления морскими судами осуществляется в рамках следующих шести обобщенных взаимосвязанных направлений:

– математическое моделирование существующих или разрабатываемых элементов и системы управления в целом, включая морскую среду;

– компьютерное моделирование в соответствии с построенными математическими моделями и критериями качества функционирования;

– анализ структурных, динамических, функциональных и других свойств системы в целом и ее отдельных элементов;

– синтез алгоритмов функционирования системы и ее отдельных варьируемых составляющих в форме их математических моделей;

– поддержка алгоритмов функционирования и информационных потоков в системе в режиме реального времени;

– алгоритмическое и программное обеспечение испытательных стендов и тренажерных комплексов для подготовки экипажей и ремонтного персонала, работающего с системой управления.

Для каждого из указанных направлений существует вполне определенная идеология действий, направленных на достижении наилучших результатов при использовании информационной и компьютерной поддержки.

Для рассматриваемых вопросов особую роль играют цифровые системы управления и обработки сигналов, базирующиеся на современных компьютерных элементах. Очевидно, что для таких систем привлечение современных информационных и компьютерных технологий ощутимо подавляет все прочие подходы в рамках указанных выше направлений. Это связано с очевидной спецификой моделирования, анализа, синтеза и реализации цифровых систем. Заметим, что принципиальная особенность применения компьютеров в системах управления состоит в том, что они одновременно служат как объектом, так и инструментом исследования и проектирования, а также базовым элементом для реализации синтезированных цифровых алгоритмов управления.

В рамках формализованных подходов *искомые элементы проектируемой системы*, а точнее – их математические модели, чаще всего, *формируются как результаты решения различных оптимизационных задач* [7]. Это существенно отличает современную идеологию проектирования от классических подходов, где оптимизационные методы применялись обычно только как вспомогательное средство, что позволяет широко применять компьютерные технологии с существенным повышением качества проектных решений.

В частности, в настоящее время широко используются различные подходы, базирующиеся на *H-теории*, позволяющие минимизировать матричные «коэффициенты усиления» элементов проектируемых систем, представленные нормами в соответствующих пространствах Харди. Весьма популярна теория H_2 и H_∞ -оптимизации [8 – 10], связанная с решением широкого спектра задач в сфере синтеза обратных связей, упрощения математических моделей объектов, обеспечения робастной устойчивости и качества и др. Соответствующие методы и алгоритмы применяются специалистами как рабочий аппарат анализа и синтеза, который всесторонне поддерживается современными компьютерными технологиями – в частности, разнообразными инструментами среды MATLAB.

В данной статье рассматриваются некоторые наиболее существенные детали, характеризующие указанные выше направления в плане их поддержки современным наукоемким программным обеспечением и технологиями его применения на базе передовой компьютерной техники.

2. Дискретные модели динамики систем управления движением

Динамика морских подвижных объектов обычно представляется нелинейными системами обыкновенных дифференциальных уравнений

$$\dot{x} = F_{in}(t, V, \omega) + F_{hd}(t, V, \omega, x_p, \delta) + f_w(t), \quad (1)$$

где векторы $x = \{V, \omega, x_p\} \in E^{12}$, $\mathbf{V} = \{V_x, V_y, V_z\} \in E^3$, и $\omega = \{\omega_x, \omega_y, \omega_z\} \in E^3$ соответственно определяют состояние, линейные и угловые скорости МПО, а вектор $\mathbf{x_p} \in E^6$ – его перемещения и углы поворота. Вектор $\delta \in E^m$ представляет управляющие воздействия на МПО, движение которого происходит под влиянием сил и моментов $\mathbf{F}_{in}$, $\mathbf{F}_{hd}$ инерционной и гидродинамической природы, а также особо выделенных внешних сил и моментов $\mathbf{f}_w(t)$.

Математическая модель (1) дополняется уравнениями динамики приводов

$$\dot{\delta}=\mathbf{F}_{\delta}(t,\delta,\mathbf{u}), \tag{2}$$

где $\mathbf{u}\in E^{m}$ – вектор управляющих сигналов, а также уравнениями измерителей

$$\mathbf{y}=\mathbf{F}_{y}(t,\mathbf{x},\delta), \tag{3}$$

где $\mathbf{y}\in E^{p}$ – вектор измеряемых динамических переменных.

Если рассматривается режим стабилизации МПО при его продольном движении с постоянной скоростью, обычно выполняют линеаризацию уравнений (1 – 3), сводя ее к виду

$$\begin{aligned}&\dot{\mathbf{x}}=\mathbf{A}\mathbf{x}+\mathbf{B}\delta+\mathbf{H}\mathbf{f}_{w}(t),\ \dot{\delta}=\mathbf{u},\\&\mathbf{y}=\mathbf{C}\mathbf{x},\end{aligned} \tag{4}$$

где $\mathbf{x}\in E^{n}$ – вектор состояния МПО, определяющий отклонения от положения равновесия, $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$ и $\mathbf{H}$ – матрицы с постоянными компонентами.

Как было отмечено выше, все современные системы управления движением МПО практически реализуются на базе цифровых устройств, работающих дискретно как по времени, так и по уровню. При этом дискретность по времени носит принципиальный характер в силу конечной длительности вычислительных операций. Несмотря на высокое быстродействие современных процессоров, квантование времени может играть весьма значимую роль при большом объеме вычислений на каждом такте для решения сложных задач.

В отличие от времени, дискретность сигналов по уровню постепенно теряет свою значимость в связи с увеличением количества разрядов в сетке. Сейчас оно достаточно велико для любых уровней сигналов, находящихся в рабочих диапазонах, при разумной для практики точности.

В связи с этим обстоятельством, далее рассматриваются только системы дискретного времени, которые и именуются *цифровыми*, причем для простоты изложение ограничивается классом линейных стационарных (DLTI) моделей.

В частности, при рассмотрении движения морских судов в режиме стабилизации заданного курса используется результат линеаризации уравнений динамики в окрестности нулевого положения равновесия по всем координатам при постоянной скорости хода, который в соответствии с (4) в дискретном времени принимает следующий вид:

$$\begin{aligned}&\mathbf{x}[k+1]=\mathbf{A}\mathbf{x}[k]+\mathbf{b}\,\delta[k]+\mathbf{h}\,d[k],\\&\delta[k+1]=Tu[k]+\delta[k],\ y[k]=\mathbf{c}\mathbf{x}[k].\end{aligned} \tag{5}$$

Здесь $\mathbf{x}\in E^{n}$ – вектор состояния, $\delta\in E^{1}$ – отклонение вертикальных рулей, $d\in E^{1}$ – возмущение от воздействия ветра и водной среды, $y\in E^{1}$ – измеряемая и регулируемая переменная (угол рыскания), $u\in E^{1}$ – управление, $k\in N^{1}$ – текущий момент дискретного времени, $\mathbf{A}$, $\mathbf{b}$, $\mathbf{h}$ и $\mathbf{c}$ – заданные матрицы с постоянными компонентами, T – период дискретности в реальном времени.

Уравнения (5) объекта управления подлежат *замыканию* с использованием математической модели обратной связи. Эта модель, по существу, определяет алгоритм цифровой обработки измеряемой информации, т.е. числовых последовательностей $y=\{y[s]\}$, $\delta=\{\delta[s]\}$, $s\in(-\infty,k]$ и $u=\{u[s]\}$, $s\in(-\infty,k-1]$, для выработки текущего отсчета $u[k]$ управляющего сигнала. Для линейных объектов обратные связи в самом общем виде представляются уравнением

$$u[k]=\Re\left(\{y[s]\},\{\delta[s]\},\{u[s]\}\right), \tag{6}$$

где $\Re$ – линейный стационарный оператор.

В частности, если для управления движением по заданному курсу используется управления с *многоцелевой структурой*, описанной в работах [11, 12], математическая модель (6) принимает следующий вид:

$$\begin{aligned}&\hat{\mathbf{x}}[k+1]=\mathbf{A}\,\hat{\mathbf{x}}[k]+\mathbf{b}\,\delta[k]+\mathbf{g}\left(y[k]-\mathbf{c}\hat{\mathbf{x}}[k]\right),\\&u[k]=\widetilde{\mathbf{k}}\,\hat{\mathbf{x}}[k]+k_{0}\delta[k]+\nu y[k]+\xi[k],\end{aligned} \tag{7}$$

где $\hat{\mathbf{x}}\in E^{n}$ – вектор состояния наблюдателя (оценка вектора $\mathbf{x}$).

Для настройки алгоритма управления (7) необходимо задать коэффициенты $\widetilde{\mathbf{k}}$, k_{0} и ν

стабилизирующего закона управления $u=(\widetilde{\mathbf{k}}+\nu\,\mathbf{c})\mathbf{x}+k_0\delta$ для объекта (5), обеспечивающие гурвицевость матрицы $\mathbf{A}_c=\left(\begin{array}{c|c}\mathbf{A} & \mathbf{b}\\ \hline T\mathbf{k} & Tk_0+1\end{array}\right)$ базовой замкнутой системы, где $\mathbf{k}=\widetilde{\mathbf{k}}+\nu\,\mathbf{c}$. Необходимо также задать вектор $\mathbf{g}$ коэффициентов наблюдателя так, чтобы его матрица $A-\mathbf{gc}$ являлась гурвицевой.

Дополним систему (7) уравнением цифрового фильтра

$$\xi=F^*(z)(y-\mathbf{c}\,\hat{\mathbf{x}}), \tag{8}$$

где передаточная функция F^* не задана и подлежит поиску в процессе синтеза, z – комплексная переменная Z – преобразования.

Назначение фильтра (8) состоит в подавлении главной части частотного спектра морского волнения в составе управляющего сигнала, подаваемого на рули. В результате фильтрации привод рулей должен как можно слабее реагировать на относительно высокие частоты, представляющие волнение, чем поддерживается экономичный режим движения судна.

Однако эта задача должна решаться при выполнении существенных ограничений, определяемых требованиями к динамике замкнутой системы на низких частотах, где необходима достаточно сильная реакция управления, например – на ступенчатые возмущения, порождаемые ветром.

Указанные требования являются противоречивыми, что приводит к задаче о построении такого фильтра, который обеспечивает определенный компромисс между ними при учете условия устойчивости замкнутой системы и ее астатизма по регулируемой и измеряемой переменной y.

3. Оптимизационный подход к синтезу цифровых алгоритмов управления

Целью всего комплекса работ, выполняемых при создании цифровых информационно-управляющих систем, является формировании дискретных алгоритмов работы их варьируемых элементов, обеспечивающих выполнение желаемых динамических требований. Для представленных выше моделей динамики системы управления курсом судна такими элементами являются настраиваемые параметры $\widetilde{\mathbf{k}}$, k_0, ν и $\mathbf{g}$, а также передаточная функция $F^*(z)$ цифрового фильтра.

В рамках формализованных подходов эти элементы удобно находить как результаты решения соответствующих оптимизационных задач. Это существенно отличает современную идеологию синтеза алгоритмов обработки информации от классических подходов, где математические методы обычно применяются лишь на этапе анализа. При этом искомые алгоритмы в классике чаще всего формируются неформальным путем, зачастую – на основе натурного или вычислительного эксперимента.

Оптимизационный подход позволяет широко применять современные компьютерные технологии на всех этапах проектирования, существенно повышая качество проектных решений, и освобождая проектировщика цифровых систем от тех сложных вопросов, которые на сегодняшний день стали рутинными. Последнее обстоятельство позволяет ему сосредоточиться на проблемах, формализация которых либо совсем невозможна, либо нежелательна по каким-либо причинам.

Основу оптимизационного подхода составляет формализация представления о качестве цифровых систем, предполагающая построение совокупности функционалов, значения которых зависят от принимаемых решений. Эти функционалы задаются в соответствующих метрических пространствах искомых элементов (рис. 1).

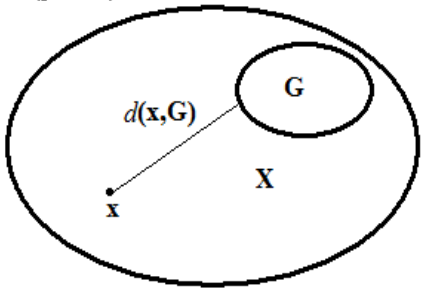


Рис. 1. Пространство возможных и желаемых проектных решений

Для пояснения существа подхода рассмотрим метрическое пространство $\mathbf{X}$, трактуемое,

как множество *возможных проектных решений* по созданию цифровой системы. На этом множестве выделим некоторое непустое подпространство $\mathbf{G}$ (рис. 1) *желаемых проектных решений*. По существу, задача проектировщика состоит в том, чтобы предложить в качестве проектного решения любой элемент из множества $\mathbf{G}$.

Заметим, что в практических ситуациях выбор такого элемента далеко не тривиален: это значит, что взятое наугад решение $\mathbf{x}$ не будет принадлежать множеству $\mathbf{G}$. Иными словами, если ввести расстояние $d(\mathbf{x},\mathbf{G})=\inf\limits_{\mathbf{g}\in\mathbf{G}}\rho(\mathbf{x},\mathbf{g})$ от точки $\mathbf{x}$ до множества $\mathbf{G}$, то может оказаться, что $d(\mathbf{x},\mathbf{G})>0$. Но допустимым является только вариант $d(\mathbf{x},\mathbf{G})=0$, который свидетельствует о принадлежности $\mathbf{x}$ множеству желаемых решений.

Отсюда следует единый подход, позволяющий формализованными путями обеспечить достижения множества $\mathbf{G}$. Он определяется постановкой и решением оптимизационной задачи

$$J(\mathbf{x})=d(\mathbf{x},\mathbf{G})\to\min_{\mathbf{x}\in\mathbf{X}}, \tag{9}$$

о минимизации расстояния от точки $\mathbf{x}$ до множества $\mathbf{G}$. Очевидно, что на любом элементе из множества $\mathbf{G}$ функционал $J(\mathbf{x})$ достигает нулевого глобального минимума и любой численный метод, обеспечивающий его поиск, ведет к желаемому проектному решению.

Рассмотрим один из характерных способов реализации указанного подхода для DLTI системы, представленной *tf*-моделью в *z*-области

$$e=\mathbf{H}(z,\mathbf{h})\mathbf{d}, \tag{10}$$

где $\mathbf{H}(z,\mathbf{h})$ – передаточная матрица, зависящая от параметра $\mathbf{h}\in E^p$. Зададим входной вектор $\mathbf{d}=\{\mathbf{d}[k]\}$ на отрезке $k\in[0,K]$. При заданных начальных условиях это однозначно определит выход $\mathbf{e}=\{\mathbf{e}[k,\mathbf{h}]\}$ системы (10) на указанном отрезке дискретного времени. Поставим задачу о такой настройке параметров, чтобы выполнялись неравенства

$$x_2[k]<x[k,\mathbf{h}]<x_1[k]\ \ \forall k\in[0,K],$$

где, $x[k,\mathbf{h}]=\|e[k,\mathbf{h}]\|$, $x_1=\{x_1[k]\}$ и $x_2=\{x_2[k]\}$ – две заданные последовательности такие, что $x_2[k]<x_1[k]\ \ \forall k\in N^1$. Иными словами, настройка параметров должна обеспечить нахождение последовательности $x=\{x[k,\mathbf{h}]\}$ в пределах заданного динамического «коридора».

Для применения оптимизационного подхода к данной задаче введём меру $\alpha(\mathbf{h})=\sum\limits_{k=0}^{N}\alpha_k(\mathbf{h})$ выхода за указанные пределы, где $\alpha_k(\mathbf{h})$ – это штрафы за нарушения ограничений по *k*-му отсчёту:

$$\alpha_k(\mathbf{h})=\begin{cases}0, & if\ \ x_2[k]\le x[k,\mathbf{h}]\le x_1[k];\\ x[k,\mathbf{h}]-x_1[k], & if\ \ x[k,\mathbf{h}]>x_1[k]_1;\\ x_2[k]-x[k,\mathbf{h}], & if\ \ x[k,\mathbf{h}]<x_2[k].\end{cases}$$

Тогда оптимизационная задача типа (9) по обеспечению нахождения кривой в пределах заданного коридора принимает вид

$$\alpha(\mathbf{h})\to\min_{\mathbf{h}\in E^p}. \tag{11}$$

Описанный подход к параметрическому синтезу цифровых алгоритмов естественным образом реализуется в среде MATLAB в форме инструментального средства Response Optimization Tool, входящего в подсистему Simulink [13]. Этот инструмент позволяет задавать границы допустимого «коридора» в визуальном режиме, обеспечивает вычисление меры выхода за его пределы для заданного вектора $\mathbf{h}$ и осуществляет запуск специального численного метода решения оптимизационной задачи (11).

4. Задача синтеза цифрового фильтра для структуры (7)

Применение оптимизационной идеологии, предложенной выше, позволяет найти настраиваемые параметры $\widetilde{\mathbf{k}}$, k_0, ν и $\mathbf{g}$ для закона управления (7) с многоцелевой структурой [11, 12]. Первые три из них обеспечивают выполнение желаемых динамических требований к поведению замкнутой системы (5) ÷ (8) при автоматической отработке командных поправок по курсу. Вектор $\mathbf{g}$ находится, исходя из аналогичных требований к динамке замкнутой системы, на

которую действуют ступенчатые возмущения $d(t)$, определяемые порывами ветра.

Для известных элементов $\widetilde{\mathbf{k}}, k_0, \nu$ и $\mathbf{g}$ в уравнениях (7) можно поставить формализованную задачу о синтезе цифрового фильтра (8). С этой целью представим уравнения замкнутой системы (5) ÷ (8) в виде

$$\begin{pmatrix}\delta\\ y\end{pmatrix}=\begin{pmatrix}P_1(z) & P_2(z)\\ P_3(z) & P_4(z)\end{pmatrix}\begin{pmatrix}1\\ \widetilde{F}(z)\end{pmatrix}H_\varepsilon(z)d\ , \tag{12}$$

где $H_\varepsilon(z)=\mathbf{c}(\mathbf{E}z-\mathbf{A}+\mathbf{gc})^{-1}\mathbf{h}$, $P_i(z)\ (i=\overline{1,4})$ – элементы матрицы

$$\begin{pmatrix}P_1 & P_2\\ P_3 & P_4\end{pmatrix}=\left(\begin{array}{c|c}\mathbf{0}_{1\times n} & 1\\ \hline \mathbf{c} & 0\end{array}\right)\left[\mathbf{E}_{n+1}z-\left(\begin{array}{c|c}\mathbf{A} & \mathbf{b}\\ \hline T\mathbf{k} & Tk_0+1\end{array}\right)\right]^{-1}\left(\begin{array}{c|c}\mathbf{g} & \mathbf{0}_{n\times 1}\\ \hline T\nu & T\end{array}\right)+\begin{pmatrix}0 & 0\\ 1 & 0\end{pmatrix},$$

причем $P_1(z)=H_a(z)/\Delta(z)$, $P_2(s)=A(z)/\Delta(z)$, где $A(z)=T\det(\mathbf{E}z-\mathbf{A})$,

$$\Delta(z)=\det\left(\begin{array}{c|c}\mathbf{E}z-\mathbf{A} & -\mathbf{b}\\ \hline -T\mathbf{k} & z-Tk_0-1\end{array}\right),\ H_a(z)=\det\left(\begin{array}{c|c}\mathbf{E}z-\mathbf{A} & \mathbf{g}\\ \hline -T\mathbf{k} & T\nu\end{array}\right).$$

Заметим, что параметры $\widetilde{\mathbf{k}}, k_0$ и ν в уравнениях (6) всегда обеспечивают астатизм системы по переменной y при выключенном фильтре (8). Условие сохранения астатизма при включенном фильтре, имеющее вид

$$F^*(z)|_{z=1}=0 \ \text{ или } \ F^*(z)\equiv F(z)\cdot(z-1)\ , \tag{13}$$

будем считать дополнением к системе (12). Здесь $F(z)$ – любая правильная дробь с полюсами внутри единичного круга, т.е. $F(z)\in RH_\infty$.

Введем в рассмотрение два функционала, заданных на движениях системы (12), (13) и характеризующих качество соответствующих процессов управления.

Первый из них представляет собой числовую меру, определяющую интенсивность работы рулей в условиях волнения:

$$F_{d\delta}=F_{d\delta}(z,F)=H_\varepsilon(z)[P_1(z)+P_2(z)F(z)(z-1)]\ . \tag{14}$$

При этом качество фильтрации можно оценить функционалом

$$J_c(F)=\sup_{\omega\in[0,\omega_s]}\left|\frac{P_1(e^{j\omega})+P_2(e^{j\omega})F(e^{j\omega})(e^{j\omega}-1)}{P_1(e^{j\omega})S_\alpha(e^{j\omega})}\right|^2=\left\|\frac{P_1(z)+P_2(z)F(z)(z-1)}{P_1(z)S_\alpha(z)}\right\|_\infty^2, \tag{15}$$

где весовой множитель $S_\alpha(z)=N_a(z)/T_a(z)$ – правильная рациональная дробь с нулями и полюсами внутри единичного круга (N_a и T_a – полиномы), $\omega_s=\pi/T$.

Очевидно, что если $J_c(F)<1$, то $\forall\,\omega\in[0,\omega_s]$ выполняется ограничение

$$|H_\varepsilon(e^{j\omega})P_1(e^{j\omega})+P_2(e^{j\omega})F(e^{j\omega})(e^{j\omega}-1)|^2<|H_\varepsilon(e^{j\omega})P_1(e^{j\omega})S_\alpha(e^{j\omega})|^2\ .$$

Таким образом, минимизация функционала (15) ведет к тому, что амплитудно-частотная характеристика (14) замкнутой системы будет ограничена заданной кривой, определяемой выбором весового множителя $S_\alpha(z)$. Этот выбор делается неформально с учетом заданного спектра возмущения.

Второй функционал, задаваемый соотношениями

$$J_m(F)=\sup_{\omega\in[0,\omega_s]}|F(z)|^2=\|F(z)\|_\infty^2\ , \tag{16}$$

определяет качество динамики замкнутой системы при действии на нее ступенчатого возмущения $d_0[k]=\widetilde{d}\cdot 1[k]$, где $\widetilde{d}=\text{const}$ – заданная постоянная. С учетом тождества (13), гарантирующего астатизм, для переходной характеристики $y_0=\{y_0[k]\}$ по переменной y справедливо неравенство

$$p_m=\max_{k\in[0,\infty)}y_0^2[k]\le p_{mo}+hJ_m(F)\ . \tag{17}$$

Здесь $h=\text{const}$ однозначно определяется исходными данными, p_{m0} – максимум квадрата переходной характеристики при выключенном фильтре.

Естественное стремление к одновременной минимизации функционалов (15) и (16) можно отразить введением единого функционала-свертки

$$J(F)=\sup_{\omega\in[0,\omega_s]}\left[\left|\frac{P_1(e^{j\omega})+P_2(e^{j\omega})F(e^{j\omega})(e^{j\omega}-1)}{P_1(e^{j\omega})S_\alpha(e^{j\omega})}\right|^2+q\left|F(e^{j\omega})\right|^2\right], \qquad (18)$$

где $q\geq 0$ – весовой множитель. Если $q\to 0$, то основное внимание уделяется частотным свойствам, а если $q\to\infty$, то $J_m(F)\to 0$, что соответствует выключению фильтра при стремлении переменной p_m к величине p_{m0}.

Тогда предлагаемая формализация задачи о построении дискретного фильтра морского волнения в составе многоцелевой структуры (6) имеет вид

$$J(F)\to\min_{F\in RH_\infty}, \qquad (19)$$

Заметим, что устойчивость фильтра в данном случае необходима и достаточна для устойчивости системы (4), как показано в [10], [12], а правильность функций $F(z)$ определяет реализуемость дискретного фильтра.

В статье [10] приведен вычислительный алгоритм решения задачи (19), основанный на спектральном подходе к среднеквадратичному оптимальному синтезу, представленному в работах [8, 9].

5. Пример цифровой системы управления курсом

Предложенный подход к цифровой фильтрации в составе многоцелевого закона управления (7), (8) проиллюстрируем на примере системы стабилизации курса транспортного судна с водоизмещением 6000т.

Математической моделью судна, движущегося с постоянной скоростью $V=8$ м/с при наличии морского волнения с интенсивностью 5 баллов по шкале Бофорта, служит следующая система линейных разностных уравнений:

$$\begin{aligned}&\beta[k+1]=a_{11}\beta[k]+a_{12}\omega[k]+b_1\delta[k]+h_1 d[k],\\&\omega[k+1]=a_{21}\beta[k]+a_{22}\omega[k]+b_2\delta[k]+h_2 d[k],\\&\phi[k+1]=T\omega[k]+\phi[k],\ \ \delta[k+1]=Tu[k]+\delta[k],\ \ y[k]=\phi[k].\end{aligned} \qquad (20)$$

Здесь β – угол дрейфа, ω – угловая скорость по курсу, φ – угол рыскания, δ – угол отклонения вертикальных рулей. Для периода дискретности $T=1$ с имеем следующие значения коэффициентов: $a_{11}=0.955$, $a_{12}=0.560$, $a_{21}=0.0267$, $a_{22}=0.592$, $b_1=-0.0132$, $b_2=-0.00742$, $h_1=-0.0648$, $h_2=-0.00456$.

Желаемое качество отработки командных поправок по курсу обеспечивает следующий базовый закон управления:

$$u=\tilde{k}_1\beta+\tilde{k}_2\omega+\tilde{k}_3\phi+k_0\delta+vy, \qquad (21)$$

где $\tilde{k}_1=0.912$, $\tilde{k}=6.11$, $\tilde{k}_3=-2.22$, $k_0=-0.339$, $v=3.44$. Для этого закона имеем собственные значения матрицы $\mathbf{A_c}$: $z_1=0.658$, $z_2=0.757$, $s_3=0.870$, $s_4=0.922$.

Сформируем асимптотический наблюдатель

$$\begin{aligned}&z_1[k+1]=a_{11}z_1[k]+a_{12}z_2[k]+b_1\delta[k]+g_1(y[k]-z_3[k]),\\&z_2[k+1]=a_{21}z_1[k]+a_{22}z_2[k]+b_2\delta[k]+g_2(y[k]-z_3[k]),\\&z_3[k+1]=Tz_2+z_3+g_3(y-z_3),\end{aligned} \qquad (22)$$

где $g_1=0.0335$, $g_2=0.00446$, $g_3=0.0944$. Для этих коэффициентов матрица $\mathbf{A}-\mathbf{gc}$ имеет собственные значения $z_1=0.560$, $s_{2,3}=0.946\pm 0.0183j$.

Уравнения стабилизирующего закона управления по выходу наблюдателя (22) без фильтра имеют вид

$$u=\tilde{k}_1 z_1+\tilde{k}_2 z_2+\tilde{k}_3 z_3+k_0\delta+vy. \qquad (23)$$

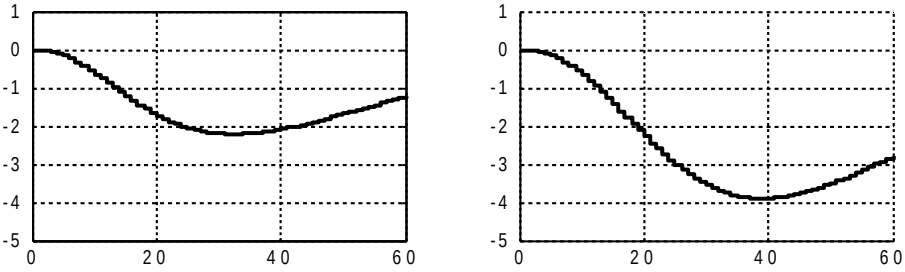
Будем рассматривать движение замкнутой системы (20), (22), (23) в двух режимах: первый определяется ветровым воздействием $d=d_0(t)=3.56\cdot 1(t)$, а второй – морским волнением со спектральной плотностью мощности

$$S_d^*(s)=S_{1d}(s)S_{1d}(-s)\ ,\ S_{1d}(s)=\sqrt{\frac{\alpha D_d}{\pi}}\frac{2s}{s^2+2\alpha+\alpha^2+\beta^2}\cdot\frac{20s^2}{20s^2+13.5s+5}\ ,$$

$D_d=5$, $\beta=0.5$, $\alpha/\beta=0.21$, пересчитанной к дискретному варианту $S_d(z)$.

Динамика первого режима представляется переходной характеристикой по рысканию, изображенной на рис. 2а, а второй режим иллюстрируется на рис. 3 графиком функции $\delta(t)$ для отклонения вертикальных рулей.

Полагая поведение рулей неэкономичным, поставим задачу (19) о синтезе цифрового фильтра, снижающего интенсивность их работы на волнении при учете ограничения $p_m=\max\limits_{k\in[0,\infty)}|y_0[k]|\le 3.8^\circ$ на переходную характеристику.



а) без фильтра *б) с фильтром*

Рис.2. Переходные характеристики замкнутой системы

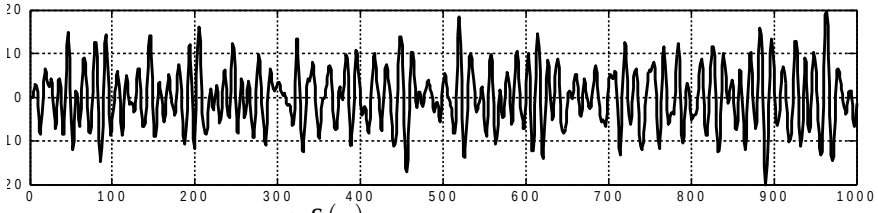


Рис. 3. Отклонения рулей $\delta(t)$ при стабилизации на волнении без фильтра

Введем в рассмотрение весовой множитель S_a вида

$$S_a(z)=\frac{N_a(z)}{T_a(z)}\ ,\ N_a(s)=3.98z^2-6.98z+3.93\ ,\ T_a(s)=6.70z^2-8.00z+1.92$$

для формирования ограничения на амплитудно-частотную характеристику замкнутой системы с учетом принятого спектра волнения.

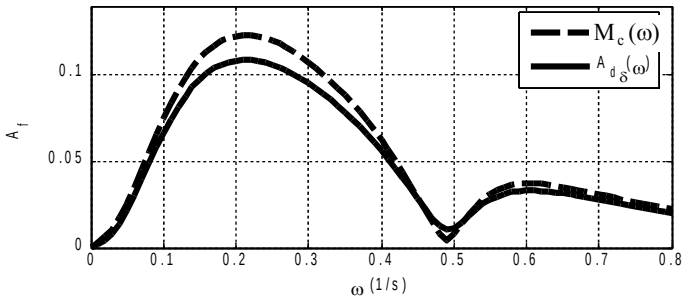


Рис. 4. Заданная граница и частотная характеристика замкнутой системы

Рисунок 4 представляет кривую $M_c(\omega)=|H_\varepsilon(e^{j\omega})P_1(e^{j\omega})S_\alpha(e^{j\omega})|$, которая ограничивает частотную характеристику $A_{d\delta}(\omega)=|F_{d\delta}(e^{j\omega},F)|\le M_c(\omega)$, что должно быть обеспечено выбором передаточной функции F фильтра.

Результат решения задачи (19) имеет вид

$$F_0^*(z)=\frac{-1.52z^2+0.525z+0.996}{z^2-1.15z+0.243}\ . \qquad (24)$$

На рис. 4 представлена амплитудно-частотная характеристика $A_{d\delta}$ замкнутой системы с фильтром (24), на рис. 2б показана ее переходная характеристика по рысканию, которая удовлетворяет принятому ограничению, а на рис. 5 дан график функции $\delta(t)$ для этой системы, функционирующей в условиях принятого волнения. Сравнение с рис. 3 констатирует существенную эффективность использования синтезированного фильтра.

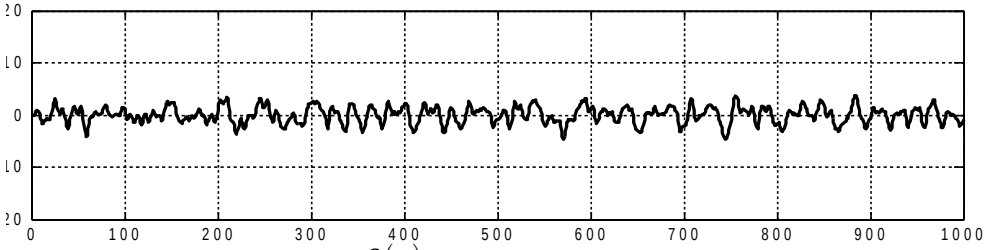


Рис. 5. Отклонения рулей $\delta(t)$ при стабилизации на волнении с фильтром

6. Заключение

В статье рассмотрен комплекс вопросов, связанных с цифровыми системами управления движением судов и цифровой обработкой сигналов, проходящих через эти системы. Очевидно, что для таких систем имеет место определенная специфика в моделировании, анализе, синтезе и бортовой реализации алгоритмов (законов) управления. Математическую формализацию соответствующих содержательных задач предлагается базировать на теории оптимизации, задавая функционалы качества на движениях дискретных моделей рассматриваемых систем.

На примере стабилизации заданного курса в работе предложен новый подход к синтезу цифровых фильтров, включаемых в состав многоцелевой структуры законов управления. В отличие от используемых в настоящее время методов, в его основу положена нестандартная задача минимизации интенсивности управления судном, движущимся в условиях морского волнения.

Разработан специализированный метод решения, отличительной чертой которого служит косвенный учет спектральных свойств волнения, определяемый введением ограничения сверху на амплитудно-частотную характеристику замкнутой системы по отклонениям рулей в заданном диапазоне частот. При этом гарантируется астатизм и априорно заданное ограничение максимального значения переходной характеристики по регулируемой координате.

Предложенный метод реализуется в виде простого алгоритма, работоспособность и эффективность которого проиллюстрирована на содержательном примере стабилизации курса морского судна.

Литература

1. *Fossen T.I.* Guidance and Control of Ocean Vehicles. New York: John Wiley & Sons, 1994.
2. *Fossen T. I.* Handbook of Marine Craft Hydrodynamics and Motion Control. John Wiley & Sons, Ltd., 2011.
3. *Perez T.* Ship Motion Control: Course Keeping and Roll Stabilization using Rudder and Fins. Springer-Verlag, London, 2005.
4. *Grimble M.J. Robust Industrial Control: Optimal Design Approach for Polynomial Systems*. Prentice Hall, 1994.
5. *Лукомский Ю.А., Корчанов В.М.* Управление морскими подвижными объектами. СПб.: Элмор, 1996.
6. *Веремей Е.И., Корчанов В.М., Коровкин М.В., Погожев С.В.* Компьютерное моделирование систем управления движением морских подвижных объектов. СПб.: СПбГУ, 2002.
7. *Веремей Е.И.* Оптимизационный подход к моделированию и разработке информационно-управляющих систем // Прикл. информатика. 2012, № 6 (42), С. 31–41.
8. *Веремей Е.И.* Спектральное представление оптимальных решений задач среднеквадратичного синтеза // Системы управления и информационные технологии, 2012, №3.1 (49), С. 124 – 128.
9. *Веремей Е.И.* Алгоритмы решения одного класса задач H_∞-оптимизации систем управления // Известия РАН. Теория и сист. управл. 2011, № 3, С. 52 – 61.
10. *Веремей Е.И., Сотникова М.В.* Применение метода H_∞-оптимизации для синтеза фильтров морского волнения // Гироскопия и навигация. 2009, № 2, С. 24 –36.
11. *Веремей Е.И., Корчанов В.М.* Многоцелевая стабилизация динамических систем одного класса // Автоматика и телемеханика, 1988, №9, С. 126 – 137.
12. *Веремей Е.И.* Синтез законов многоцелевого управления движением морских объектов // Гироскопия и навигация. 2009, № 4, С. 3 –14.
13. Simulink Design Optimization: User's Guide. – Natick (Mass.): The MathWorks, Inc., 2012.

Веремей Е.И. [1], Сотникова М.В. [2]

[1]Санкт-Петербургский государственный университет, г. Санкт-Петербург, д.ф.-м.н., профессор, заведующий кафедрой компьютерных технологий и систем, e.veremey@spbu.ru

[2] Санкт-Петербургский государственный университет, г. Санкт-Петербург, к.ф.-м.н., доцент кафедры компьютерных технологий и систем, m.sotnikova@spbu.ru

АЛГОРИТМЫ ОПТИМИЗАЦИИ ДВИЖЕНИЯ ПО ЗАДАННОЙ ТРАЕКТОРИИ С УЧЕТОМ ПРОГНОЗА ПОГОДНЫХ УСЛОВИЙ

КЛЮЧЕВЫЕ СЛОВА

Маршрут движения, прогноз погоды, оптимизация, поиск пути на графе.

АННОТАЦИЯ

Цель работы состоит в формировании эффективных вычислительных алгоритмов оптимизации движения подвижного объекта по заданной траектории на длительных дистанциях, где экономия расхода топлива и уменьшение времени в пути имеют существенное значение. Оптимизация выполняется с учетом прогноза погодных условий, в частности, динамически изменяющихся запретных участков. Предлагаемые алгоритмы оптимизации основаны на представлении маршрута движения по фиксированной траектории в двумерном пространстве. Результаты работы алгоритмов продемонстрированы на примерах.

Введение

Данная работа посвящена задаче оптимального распределения скоростей подвижного объекта на заданной траектории, то есть формированию оптимального маршрута движения с учетом прогноза погодных условий. Рассматриваемая задача имеет важное практическое значение, в частности, для контейнеровозов, выполняющих длительные переходы, например, через Атлантический океан. Критериями оптимальности маршрута являются время в пути и расход топлива. Важно подчеркнуть, что в данной работе траектория движения считается заданной, а оптимизации подлежат только скорости движения на отдельных ее участках. Более общая задача формирования маршрута, включающая как поиск траектории, так и распределения скоростей, рассмотрена, например, в работе [1].

Оптимизация движения объекта по заданной траектории выполняется с учетом параметров подвижного объекта, погодных условий и динамических ограничений, обусловленных опасными для движения участками.

Предлагаемое в статье решение основано на представлении маршрута движения по заданной траектории в двумерном пространстве, измерениями которого являются время и длина пройденного пути. На основе такого представления формируется граф и выполняется поиск кратчайшего пути на этом графе. Уточнение полученного решение достигается посредством решения задачи нелинейного программирования. Результаты работы алгоритмов продемонстрированы на примерах.

Постановка задачи

В дальнейшем под маршрутом движения подвижного объекта будем понимать траекторию его движения и распределение линейных скоростей на отдельных участках этой траектории. С целью сведения задачи к конечномерной оптимизации будем представлять траекторию движения объекта в виде конечного числа локсодромических участков, то есть участков с постоянным значением курсового угла, как показано на рис. 1. На рисунке приняты следующие обозначения: $A\left(\psi_0,\lambda_0\right)$ и $B\left(\psi_1,\lambda_1\right)$ – начальная и конечная точки траектории, $M\left(\psi,\lambda\right)$ – текущая позиция подвижного объекта с географическими координатами ψ и λ, M_i, $i\in\overline{1,p+1}$ – точки поворота траектории, $M_1=A$ и $M_{p+1}=B$, φ_i, S_i, V_i, $i\in\overline{1,p}$ – значения курсового угла, длины и заданной скорости на i-м локсодромическом участке, p – общее число участков.

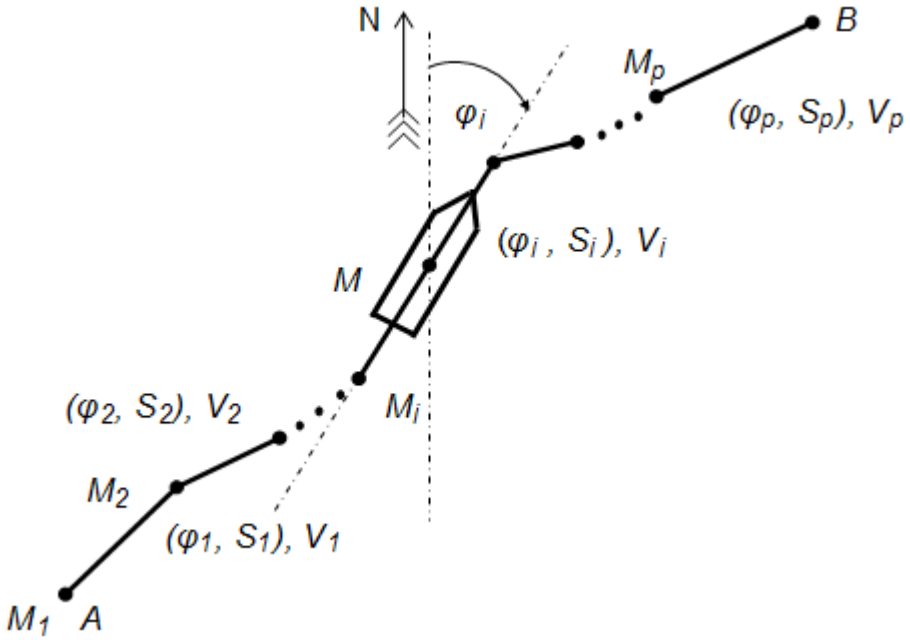


Рис. 1. Траектория движения подвижного объекта

Тогда траектория движения объекта $\gamma(\mathbf{r})$ однозначно определяется вектором

$$\mathbf{r}=\{(\varphi_1, S_1), (\varphi_2, S_2), \ldots, (\varphi_p, S_p)\} \in E^{2p}, \quad (1)$$

а скорости движения по этой траектории – вектором

$$\mathbf{v}=(V_1, V_2, \ldots, V_p) \in E^p. \quad (2)$$

Пара совместно рассматриваемых векторов $\mathbf{r}$ и $\mathbf{v}$ однозначно определяет маршрут движения объекта.

Подчеркнем, что компонентами V_i вектора $\mathbf{v}$ являются заданные (желаемые) скорости движения по участкам траектории. Далее будем различать заданную и фактическую скорости движения объекта. Значение заданной скорости напрямую связано с работой движителей, например, с частотой вращения гребного винта и, следовательно, с расходом топлива. Фактическая скорость движения по участку траектории отличается от заданной и зависит от погодных условий и направления движения объекта. Заданная скорость совпадает с фактической при отсутствии внешних воздействий, например, для морских объектов на тихой воде [2].

Фактическую скорость будем определять приближенно как положение равновесия при заданных силе тяги движителя и внешних воздействиях, определяемых погодными условиями, из следующего уравнения

$$f\left(t,\ w,\ n(v), P_{curr}, P_{wave}, P_{wind}\right)=0. \quad (3)$$

Здесь w и v – фактическая и заданная скорости объекта соответственно, $n(v)$ – частота вращения движителя, соответствующая заданной скорости v, $P_{curr}, P_{wave}, P_{wind}$ – внешние воздействия, определяемые влиянием течения, волнения и ветра на фактическую скорость движения. Уравнение вида (3) служит для определения фактической скорости морских подвижных объектов, однако аналогичная модель может быть сформирована и для других классов подвижных объектов. Дальнейшее изложение будем вести применительно к морским подвижным объектам. Отметим, что для составления уравнения (3) необходимо задать ряд технических параметров подвижного объекта, в частности, длину, ширину, осадку и др.

Прогноз погоды задается для узлов сетки, покрывающей зону плавания Ω_0 на земной поверхности и имеющей фиксированный шаг по широте $\Delta\psi$ и долготе $\Delta\lambda$. Погода в узлах сетки задается с фиксированным временным шагом ΔT, то есть в моменты $T_0, T_0+\Delta T, \ldots, T_1$. В каждом узле сетки доступны значения следующих параметров:

а) ветра (направление и скорость);

б) волнения (высота волны, период, направление);

в) течения (направление и скорость).

Кроме указанных параметров, на район плавания Ω_0 с фиксированным временным шагом ΔT задаются опасные по погодным условиям зоны в виде набора замкнутых контуров. Учитывая, что траектория движения объекта $\gamma(\mathbf{r})$ задана вектором $\mathbf{r}$ и не изменяется, введем запретные участки на этой траектории, определяемые как пересечения контуров опасных зон с траекторией движения. Эти запретные участки, как и прогноз погоды, изменяются динамически с фиксированным временным шагом ΔT. Пусть отрезок $\left[s^i_{j1}, s^i_{j2}\right]$ есть пересечение в момент времени $T_0 + i\Delta T$ контура *j*-ой опасной зоны с траекторией, а s – длина дуги, отсчитываемая от начальной точки. Тогда объединение всех таких отрезков образует запретные участки на траектории в момент $T_0 + i\Delta T$. В результате множество

$$\Omega_i = \bigcup_{j=1}^{k_i} \left[s^i_{j1}, s^i_{j2}\right], \tag{4}$$

где k_i – число опасных зон в момент времени $T_0 + i\Delta T$, представляет собой множество точек на траектории, в которых не может находиться подвижный объект на интервале времени $\left[T_0 + i\Delta T, T_0 + (i+1)\Delta T\right]$.

Введем терминальное ограничение

$$\left(\psi_1 - \psi(t_1, \mathbf{v})\right)^2 + \left(\lambda_1 - \lambda(t_1, \mathbf{v})\right)^2 < \varepsilon, \quad t_1 \le T_1, \tag{5}$$

смысл которого состоит в том, что текущая точка $M(\psi, \lambda)$, представляющая подвижный объект, должна попасть в окрестность конечного положения $B(\psi_1, \lambda_1)$ не позднее конечного момента задания прогноза T_1.

Рассмотрим также ограничения на величину заданной скорости на каждом участке траектории, определяемые ограниченными ресурсами движителей:

$$v_{\min} \le V_i \le v_{\max}, \quad i = \overline{1, p}. \tag{6}$$

Наконец, перейдем к рассмотрению ограничений, определяемых погодными условиями. Для этого рассмотрим функцию $T_{al}(\mathbf{v})$, значение которой равно времени нахождения на опасных участках (5) траектории $(\mathbf{r})$ для выбранного распределения скоростей $\mathbf{V}$. В этом случае подвижный объект не попадает в опасные зоны, если выполняется условие

$$T_{al}(\mathbf{v}) = 0. \tag{7}$$

На основе ограничений (5) – (7) введем допустимое множество распределений заданных скоростей $\mathbf{v}$ на траектории $\gamma(r)$:

$$\Omega = \left\{ \mathbf{v} \in E^p \mid \rho(t_1, \mathbf{v}) < \varepsilon,\ t_1 \le T_1;\ v_{\min} \le V_i \le v_{\max},\ i \in \overline{1, p};\ T_{al}(\mathbf{v}) = 0 \right\}.$$

где функция $\rho(t_1, \mathbf{v})$ определяется в соответствии с ограничением (5). Тогда задачу оптимизации маршрута можно сформулировать как задачу конечномерной оптимизации следующего вида:

$$J_T(\mathbf{v}) \to \min_{v \in \Omega \subset E^p} \quad \text{или} \quad J_F(\mathbf{v}) \to \min_{v \in \Omega \subset E^p} \tag{8}$$

в случае минимизации времени перехода или расхода топлива соответственно.

Алгоритмы оптимизации распределения скоростей

С целью решения задач оптимизации (8) представим процесс движения подвижного объекта по заданной траектории в двумерном пространстве. В этом пространстве введем систему координат Ost с началом в точке O и координатными осями Os и Ot. При этом ось абсцисс Os обозначает пройденный по траектории путь s, а ось ординат Ot – текущее время.

Для получения начального приближения к оптимальному решению задач (8) будем полагать, что фактическая и заданная скорости совпадают. Проведем прямые $s = v_{\min} t$ и $s = v_{\max} t$, отвечающие движению по траектории с заданной минимальной и максимальной

скоростью соответственно. Эти прямые ограничивают все возможные варианты движения по траектории.

Построим прямые $s=S_i,\ i=\overline{1,p-1}$, соответствующие точкам поворота траектории. Так как заданная скорость на участке траектории с прямым значением курсового угла не изменяется, то маршрут движения по траектории в рассматриваемом пространстве изображается ломаной линией. При этом точки сопряжения участков ломаной могут находиться только на прямых $s=S_i$. Отметим, что только в этих точках сопряжения может изменяться угол наклона к оси участков ломаной, если заданные скорости на соседних участках не совпадают.

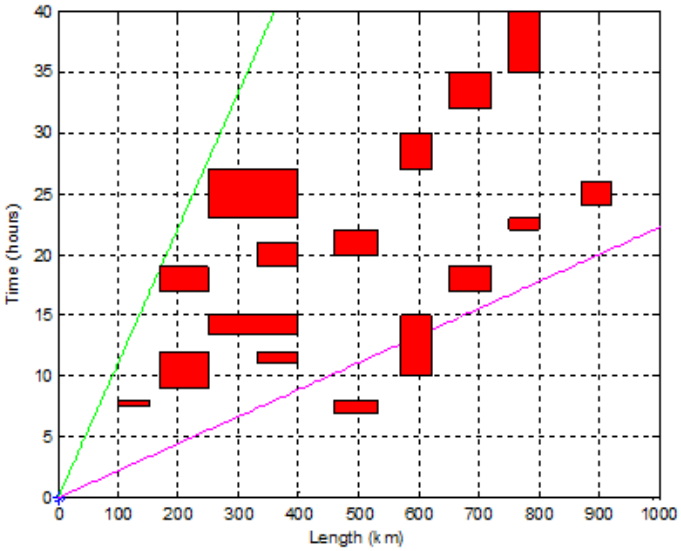


Рис. 2. Опасные зоны и прямые максимальной и минимальной скорости

В каждый момент времени $T_0+i\Delta T$ задания прогноза запретные для движения участки траектории определяются отрезками $\left[s^i_{j1}, s^i_{j2}\right]$. Этим отрезкам на плоскости с введенной системой координат Ost соответствуют прямоугольники, длина стороны которых по оси ординат кратна интервалу ΔT задания прогноза. На рис. 2 приведен пример ограничивающих прямых и опасных зон на траектории.

В итоге задача поиска оптимального по времени в пути распределения скоростей сводится к поиску такой ломаной с началом в точке O, каждое звено которой имеет тангенс угла наклона к оси абсцисс в пределах $\left[\frac{1}{v_{\max}}, \frac{1}{v_{\min}}\right]$ с точками сопряжения на прямых $s=S_i,\ i=\overline{1,p-1}$, причем конечная точка должна лежать на прямой $s=S_p$ с минимальным значением времени t. При этом искомая ломаная не должна пересекать прямоугольники, обозначающие запретные для движения зоны.

Для решения поставленной задачи построим граф в соответствии со следующим алгоритмом.

1) Отметить на прямых $s=S_i,\ i=\overline{1,p-1}$ точки, расположенные между прямыми максимальной и минимальной скорости на равном расстоянии друг от друга Δt. Вычеркнуть все точки, которые попадают внутрь прямоугольников, определяющих опасные зоны.

2) Соединить отрезками пары точек, расположенные на соседних прямых $s=S_i$ и $s=S_{i+1}$, если выполняются следующие условия:

а) угол наклона отрезка к оси абсцисс находится в пределах $\left[\frac{1}{v_{\max}}, \frac{1}{v_{\min}}\right]$;

б) указанный отрезок не пересекает ни один из прямоугольников, определяющих опасные зоны.

3) Принять построенные точки и отрезки в качестве вершин и ребер графа. Каждому ребру

графа присвоить вес, равный времени движения по ребру при условии, что фактическая скорость совпадает с заданной.

4) В качестве конечной точки P принять точку на прямой $s=S_p$ с минимальным значением ординаты $t=t_p$. Найти кратчайший путь на графе, соединяющий начальную точку O и конечную точку P. Если такого пути не существует, то в качестве конечной точки P принять следующую точку на прямой $s=S_p$ со значением ординаты $t=t_p+\Delta t$ и т.д.

5) Использовать найденное распределение скоростей в качестве начального приближения для оптимизации маршрута по времени перехода. Решить задачу оптимизации (8) при помощи методов нелинейного программирования, например, с использованием метода последовательного квадратичного программирования [3].

На рис. 3 показан пример построения графа в соответствии с приведенным алгоритмом. На рис. 4 показана искомая ломаная, соответствующая кратчайшему по времени в пути распределению скоростей на заданной траектории. Для уточнения полученного решения, с учетом влияния погодных условий на скорость движения объекта, необходимо решить задачу оптимизации (8).

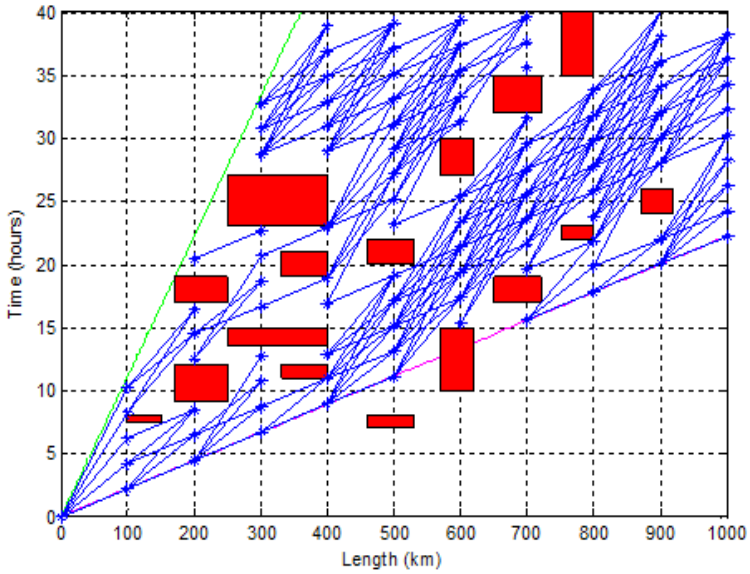


Рис. 3. Граф, представляющий возможные маршруты движения по траектории

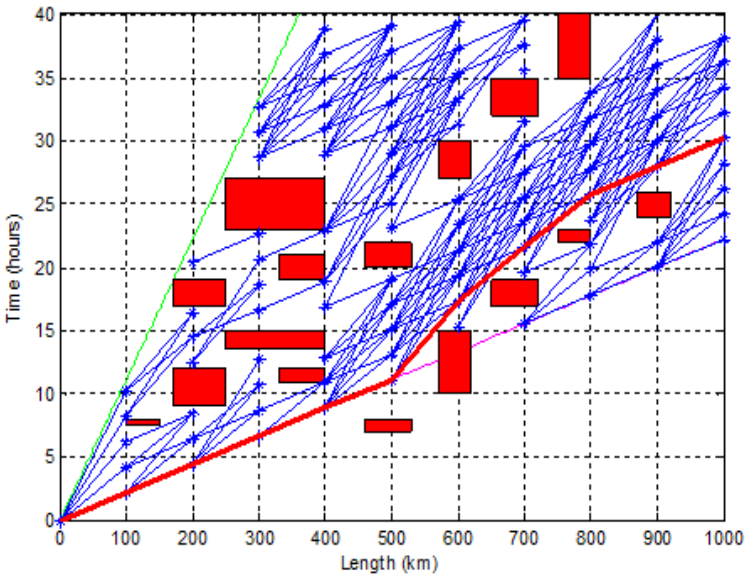


Рис. 4. Оптимальный маршрут движения по заданной траектории

Рассмотрим теперь алгоритм построения оптимального по расходу топлива распределения

скоростей на заданной траектории. Также как и в предыдущем алгоритме, сначала необходимо построить граф и каждому ребру этого графа присвоить вес. Но, в отличие от предыдущего варианта, вес ребра следует принять равным расходу топлива при движении по ребру, который вычисляется по формуле:

$$w = \rho(V)t \ ,$$

где $\rho(V)$ – удельный расход топлива при заданной скорости V , t – время движения по ребру. Далее, в зависимости от постановки задачи, возможны два варианта работы алгоритма. Опишем каждый из них.

1) Требуется минимизировать расход топлива при ограниченном допустимом времени в пути $t_{\max}$. В этом случае для построения оптимального маршрута будем перебирать последовательно точки P на прямой $s = S_p$ со значением ординаты, не превышающим $t_{\max}$. Для каждой из этих точек найдем кратчайший путь на графе с началом в точке O , то есть путь с наименьшим расходом топлива. В качестве результата, среди найденных вариантов, примем ту ломаную, для которой расход топлива минимален.

2) Требуется минимизировать время в пути при заданном ограничении на допустимый расход топлива $f_{\max}$. Для поиска оптимального решения возьмем точку P на прямой $s = S_p$ с наименьшим допустимым значением ординаты. Вычислим кратчайший путь с началом в точке O и с концом в точке P , то есть путь минимизирующий расход топлива. Если при этом расход топлива не превышает заданного максимального значения $f_{\max}$, то принимаем в качестве результата построенную ломаную. В противном случае, переходим к следующей точке на прямой $s = S_p$ с шагом Δt по оси ординат.

В обоих случаях полученное решение необходимо уточнить, учитывая влияние погодных условий на скорость движения подвижного объекта и, соответственно, несовпадение заданной и фактической скорости. Для этого необходимо решить задачу оптимизации (8), принимая найденное распределение скоростей в качестве начального приближения.

Заключение

Предложены алгоритмы поиска оптимального распределения скоростей на заданной траектории, которые сводятся к построению соответствующего графа и поиску кратчайшего пути на графе. Уточнение полученного решения достигается посредством решения вспомогательной задачи нелинейного программирования. Указанные алгоритмы могут быть перенесены на задачу формирования оптимальных маршрутов, представимых в трехмерном пространстве с препятствиями, определяемыми опасными погодными условиями и статическими ограничениями, например, береговыми линиями.

Работа выполнена при финансовой поддержке РФФИ, грант № 14-07-00083.

Литература

1. Сотникова М.В. Алгоритмы формирования маршрутов движения судов с учетом прогноза погодных условий // Вестн. С.-Петерб. ун-та. Сер. 10. 2009. – Вып. 2. – С. 181-196.
2. Веремей Е.И., Корчанов В.М., Коровкин М.В., Погожев С.В. Компьютерное моделирование систем управления движением морских подвижных объектов. СПб.: НИИ Химии СПбГУ, 2002. 370 с.
3. Fletcher R. Practical Methods of Optimization. – 2nd ed. – John Wiley & Sons Ltd., 2000. – 450 p.

Жгун Т.В.[1], Липатов А.В.[2], Лемешова Д.Д.[3]

[1] Новгородский государственный университет им. Ярослава Мудрого, г.Великий Новгород, к.ф.-м.н., доцент кафедры прикладной математики и информатики, zhtv@.mail.ru

[2] Новгородский государственный университет им. Ярослава Мудрого, г.Великий Новгород, аспирант

[3] Новгородский государственный университет им. Ярослава Мудрого, г.Великий Новгород, магистрант

ПРИМЕНЕНИЕ МЕТОДА ГЛАВНЫХ КОМПОНЕНТ ДЛЯ ПОСТРОЕНИЯ ОБЪЕКТИВНЫХ ПОКАЗАТЕЛЕЙ ИЗМЕНЕНИЯ КАЧЕСТВА СИСТЕМЫ

КЛЮЧЕВЫЕ СЛОВА

Качество системы управления, интегральная характеристика качества, изменение характеристики качества, интегральные индикаторы качества жизни, шум в измеряемых данных, отношение сигнал/шум, метод главных компонент, информативность метода главных компонент.

АННОТАЦИЯ

В статье предложен алгоритм определения композитного индекса изменения качества системы на основе метода главных компонент с учетом наличия шума в измеряемых данных. Построение латентной интегральной характеристики рассматривается как задача выделения полезного сигнала из массива данных в условиях априорной неопределенности и решается на основании задаваемого отношения сигнал/шум. Алгоритм был использован для получения интегральных индикаторов качества жизни населения субъектов Российской Федерации.

Введение

Интегральные характеристики качества системы используются при решении задач управления для совершенствования механизмов управления на основании сопоставления интегральных индикаторов рассматриваемых объектов и рейтингов объектов, определяемых их интегральными характеристиками, и являются показателем степени достижения цели управления. Сводные (интегральные) показатели различных синтетических категорий социально-экономического развития (в том числе качества жизни населения) широко используются как в межстрановом и межрегиональном макроэкономическом анализе, так и в теории и практике социально-экономического управления [1]. С 1974 года выходит специальный международный журнал *Social Indicators Research*, который публикует результаты исследований, касающихся измерения качества жизни муниципальных, региональных, национальных и международных систем.

К числу наиболее часто используемых показателей качества жизни относится Индекс развития человеческого потенциала (ИРЧП), который активно используется Организацией объединенных наций начиная с 1990 года. В рамках ежегодных «Докладов о человеческом развитии» даются определенные представления о соотношении качества жизни в разных странах мира на основании этого рейтинга [2]. Среди отечественных исследований, наиболее обоснованным подходом к оценке качества жизни населения регионов России, следует отметить работы С.А. Айвазяна [3-4]. Согласно его методике, вектор интегральных индикаторов есть проекция матрицы предобработанных данных A на первую главную компоненту. Этот метод используется для оценки статических систем, когда первая главная компонента хорошо приближает моделируемую ситуацию, т.е. если максимальное собственное число ковариационной матрицы дает вклад не менее 70% в сумму всех собственных чисел. Такое соотношение выполняется, если рассматривается небольшое количество признаков (не более пяти) и одно из свойств системы доминирует над остальными. При описании социально-экономических систем число переменных значительно более пяти, и структура системы не допускает простого приближения. В качестве выхода из этой ситуации рассматривается согласно [4] понижение порога информативности до 55% и деление исходной системы на подсистемы, описываемые

меньшим количеством переменных. Однако применение этой методики для ряда последовательных наблюдений дает неудовлетворительные результаты.

В статье предлагается алгоритм построения интегральной характеристики изменения качества системы. Построение латентной интегральной характеристики производится на основе метода главных компонент с учетом наличия шума в измеряемых данных. При этом информативность получаемого решения оценивается апостериорно на основании дисперсионного критерия и выбранного параметра отношения сигнал/шум. С помощью предложенного алгоритма построены интегральные индикаторы качества жизни субъектов Российской Федерации за 2007-2012 годы.

1 Постановка задачи вычисления интегральной характеристики изменения качества системы

Рассмотрим построение интегральной оценки системы из m объектов, для которой известны таблицы описаний объектов для ряда наблюдений – матрицы размерностью $m \times n$ $A^t = \left\{ a_{ij} \right\}_{i,j=1}^{n,m}$, $t = 1, 2, \ldots, p$. . Для каждого момента t вектор интегральных показателей имеет вид

$$q^t = A^t \cdot w^t, \quad (1)$$

где $q^t = \langle q^{t_1}, q^{t_2}, \ldots, q^{t_m} \rangle^T$ – вектор интегральных индикаторов момента t, $w^t = \langle w^{t_1}, w^{t_2}, \ldots, w^{t_m} \rangle^T$ – вектор весов показателей для момента t, A^t –матрица предобработанных данных для момента t. Для построения интегрального индикатора качества системы требуется найти веса показателей w^t для каждого момента времени.

Будем рассматривать оцениваемый объект как сложную (не поддающуюся удовлетворительной формализации), большую (число состояний выше современных вычислительных возможностей) систему. Такими системами являются как каждый биологический объект, так и любая социальная система. Система доступна для наблюдения, и известно конечное, достаточно большое число регистрируемых с некоторой точностью численных характеристик этой системы в разные моменты времени. Значимость регистрируемых показателей для функционирования системы в общем случае неизвестна. Для решения задачи управления требуется дать мотивированную оценку каждого наблюдаемого объекта на всем промежутке наблюдений, т.е. вычислить в динамике интегральную характеристику качества системы. Численные характеристики системы предварительно подвергнуты унификации – приведению значений переменных на отрезок [0, 1] по принципу: «чем больше, тем лучше».

Любое измерение, в том числе и статистическое, неизбежно связано с точностью измерительного прибора, поэтому результат измерения неизбежно содержит неустранимую ошибку. Построение интегральной характеристики системы можно рассматривать как задачу выделения полезного сигнала на фоне шумов в условиях отсутствия априорной информации о статистических характеристиках аддитивного шума и полезного сигнала при наличии единственной реализации измеряемого процесса. Эта задача аналогична задаче восстановления цифровых изображений, искаженных белым гауссовским шумом. Метод главных компонент (МГК) позволяет выделить структуру в многомерном массиве данных и с успехом применяется для распознавания изображений и для шумоподавления.

Количественные характеристики конкретной системы, функционально связанные с ее структурными особенностями и условиями ее работы, зависят от отношения сигнала к шуму. Это отношение часто используется для количественной оценки эффективности различения сигналов в оптико-электронных, телевизионных автоматических системах, в средствах контроля и диагностики.

ОСШ – о**тношение сигнал/шум** (англ. *signal-to-noise ratio*, сокр. *SNR*) представляет собой отношение сигнала (а если точнее, суммы сигнала и шума) к шуму. Величина может быть вычислена либо как безразмерное отношение амплитуды сигнала к амплитуде шума: $SNR = A_s / A_n$; либо в децибелах: $SNR\,(dB) = 20 \cdot log_{10}(A_s / A_n)$. Эта величина наиболее полно описывает качество воспроизведения сигнала в телевизионных системах, в системах мобильной связи, в астрофотометрии.

Выбор порогового значения отношения полезного сигнала к шуму фона, позволяющего различить сигнал на фоне шума, обоснован в [5]. Современные технические системы (и человеческий глаз) уверенно выделяют сигнал из шума, если уровень ОСШ системе составляет

около 7 дБ (или в безразмерных единицах 2,2). Такое пороговое значение используется в фотометрии слабых объектов: при регистрации сигнала от тусклых звезд необходимо, чтобы отношение сигнал/шум превышало 2,2. Телевизионные системы могут восстановить неповрежденную картинку, если отношение уровня сигнала к уровню шумов превосходит пороговое значение около 7дБ.

Статистические данные неизбежно содержат погрешность измерения. Любой результат, полученный на основании этих данных, будет содержать неустранимую ошибку. Переход к другому моменту времени означает изменение данных, которое при неизменной структуре системы вызвано как изменением ситуации, так и случайными ошибками. Метод главных компонент на основании различных для разных моментов значений собственных векторов и собственных значений описывает неизменную структуру системы. Следовательно, именно значения собственных чисел и собственных векторов будут тем сигналом, который нужно распознать, т.е. по имеющимся реализациям выделить сигнал из зашумленных данных.

Предположение о том, что при вариации входных данных у собственных чисел имеется общая тенденция, иллюстрирует рис. 1, где представлены значения упорядоченных по убыванию собственных чисел для разных наблюдений. В среднем значении рассматриваемых величин хорошо просматривается тенденция (сигнал) и случайное отклонение от нее.

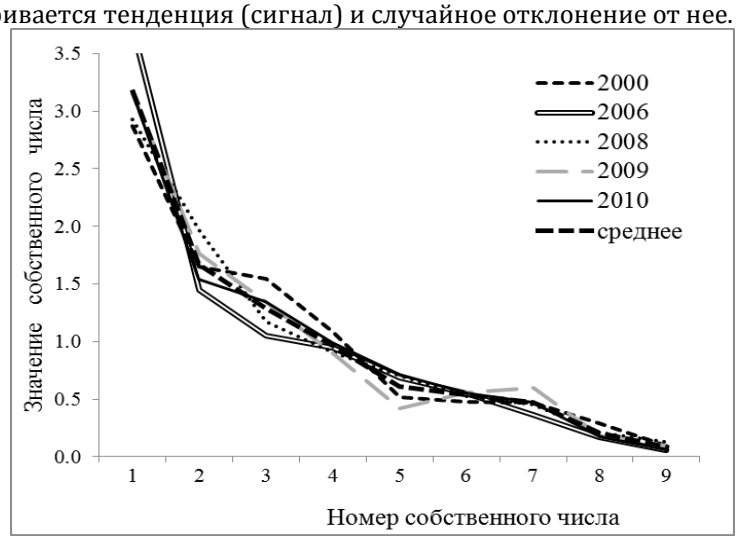


Рис. 1. Собственные значения ковариационной матрицы переменных для различных моментов наблюдения

Собственные вектора определяются с точностью до направления, в отличие от собственных чисел, определяемых однозначно. Среднее значение факторных нагрузок переменных зависит от выбранного направления и не может однозначно характеризовать сигнал. Следовательно, на основе вычисленных для различных наблюдений собственных векторов (упорядоченных в порядке убывания собственных чисел) нужно распознать случайные и неслучайные компоненты этих векторов и определить значения координат неслучайных переменных этих векторов.

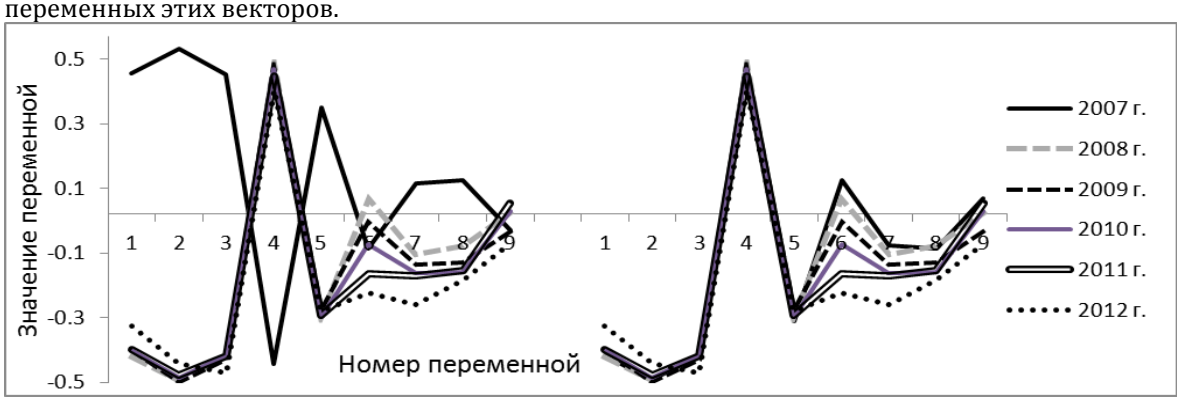


Рис. 2. Согласование направлений собственных векторов для первой главной компоненты. Слева – исходные направления собственных векторов, справа – выбор направления собственных векторов, обеспечивающего максимальную сумму амплитуд действующих сигналов

Наличием неслучайного (т.е. значимого) вклада переменной в структуру главных компонент будем считать не большую величину факторной нагрузки, а **инвариантность** факторной нагрузки при возмущениях, признаком которой будет величина отношения сигнал/шум, определяемая средними значениями переменной и среднеквадратичным отклонением. Если это отношение выше порогового значения, такую переменную считаем неслучайной – действующей. При невыполнении такого соотношения переменная характеризует шумовую составляющую сигнала и не участвует в дальнейшем рассмотрении.

Для проверки этого условия нужно согласовать направления собственных векторов, которые определяются с точностью до направления, в отличие от собственных чисел, определяемых однозначно. На рис. 2 представлен выбор направления собственных векторов для первой главной компоненты.

2 Алгоритм построения интегральной характеристики изменения качества системы на основании регистрируемых измерений как решение задачи выделения полезного сигнала на фоне шумов

Приведем алгоритм построения интегральной характеристики изменения качества системы для ряда наблюдений на основании регистрируемых измерений, который обеспечивает решение задачи выделения сигнала в многомерном массиве данных в условиях априорной неопределенности о свойствах полезного сигнала на основании задаваемого отношения сигнал/шум. Алгоритм включает следующие шаги:

1 шаг. Подготовка данных.

- выбор значения ОСШ для рассматриваемой системы;
- определение набора переменных и деление исходных переменных на блоки, исходя из содержательных соображений;
- обеспечение однородности выборки: устранение пропущенных значений, коллинеарности данных; унификация показателей.

Если исходный показатель связан с анализируемым интегральным свойством качества жизни монотонной зависимостью, то при унифицировании исходных показателей (приведении их на отрезок [0, 1] по принципу «чем больше, тем лучше») переменные x_{ij} для каждого момента наблюдения преобразовываются по правилу:

$$a_{ij} = s_j + (-1)^{s_j} \cdot \frac{x_{ij} - m_j}{M_j - m_j},$$

где $s_j = 0$, если оптимальное значение j –го показателя максимально и $s_j = 1$, если оптимальное значение j –го показателя минимально, m_j – наименьшее значение j-го показателя по всей выборке (глобальный минимум), M_j – наибольшее значение j-го показателя по всей выборке (глобальный максимум).

Если исходный показатель связан с анализируемым интегральным свойством качества жизни немонотонной зависимостью (т.е. внутри диапазона изменения данного показателя существует некоторое оптимальное значение x_j^{opt}, при котором достигается наивысшее качество), то значение соответствующего унифицированного показателя рассчитывается по формуле:

$$a_{ij} = \left(1 - \frac{|x_{ij} - x_{j_{opt}}|}{max\left((M_j - x_{j_{opt}}), (x_{j_{opt}} - m_j)\right)}\right).$$

Дальнейшие шаги проделываем для каждого блока отдельно.

2 шаг. Определение априорных собственных векторов и собственных чисел для каждого наблюдения.

- центрирование унифицированных переменных;
- вычисление по центрированным значениям унифицированных показателей ковариационной матрицы, совпадающей в этом случае с матрицей корреляций;
- нахождение для каждого наблюдения собственных значений ковариационной (корреляционной) матрицы, ранжированных по убыванию;
- нахождение собственных векторов, соответствующих соответствующим собственным числам.

Каждый новый элемент центрированной матрицы данных вычисляется по формуле:

$$z_{ij} = \frac{a_{ij} - \overline{a_j}}{s_j},$$

где $\overline{a_j} = \frac{1}{m}\sum_{i=1}^{m} a_{ij}$ – выборочная средняя, $s_j^2 = \frac{1}{m}\sum_{i=1}^{m}(x_{ij} - x_j)^2$ – выборочная дисперсия, т.е. s_j – выборочное среднеквадратичное отклонение.

3 шаг. Определение эмпирических собственных чисел. Согласование направлению собственных векторов и определение эмпирических главных компонент.

- вычисление эмпирических значений собственных чисел как среднего значения собственных чисел для различных выборок;
- выбор направлению собственных векторов, которое максимизирует сумму ОСШ у действующих переменных (у переменных, для которых вычисленное ОСШ превосходит заданный порог);
- определение эмпирических главных компонент: факторные нагрузки у недействующих переменных обнуляются, для действующих переменных в качестве значения факторной нагрузки принимаем среднее значение по всем наблюдениям (при согласовании направлений собственных векторов);
- выбор числа эмпирических главных компонент (ЭГК) *N* ($N \le n$). Количество выбранных ЭГК *N* должно обеспечивать заданную информативность γ .

4 шаг. Согласование направлений главных компонент.

- нормирование главных компонент, умножением на квадратный корень из соответствующего эмпирического собственного числа;
- согласование направлений нормированных эмпирических главных компонент;
- нахождение весовых коэффициентов целевой функции $W^* = \langle w^{*1}, w^{*2}, \ldots, w^{*n} \rangle$ суммированием нагрузок согласованных эмпирических главных компонент $w_j^* = \sum_{\zeta=1}^{p} w_{\xi j}^*$, *j*=1, ...*n*, ζ = 1, ... , *p*.
- вычисление интегрального показателя как сумму произведений значений вектора весов и матрицы унифицированных показателей A^t

$$q^t = A^t \cdot W^*, \qquad (2)$$

Для фиксированного момента *t* интегральную оценку для объекта с номером *i* записывают в виде аддитивной свертки данных с весами, определяемыми вектором *W**:

$$q_i^t = \sum_{j=1}^{n} w_j^* \cdot a_{ij}^t, \quad i = 1, 2, \ldots m, \qquad w_j^* = \sum_{\zeta=1}^{p} w_{\xi j}^*, \quad j = 1, 2, \ldots, n. \qquad (3)$$

5 шаг. Определение весовых коэффициентов блоков и вычисление интегральной характеристики системы.

1)вес блока *k,* где *k* = 1, ..., К, определяем пропорционально его информативности γ_k :

$$\widetilde{w}_k = \frac{\gamma_k}{\sum_{k=1}^{K} \gamma_k}, \quad \sum_{k=1}^{K} \widetilde{w}_k = 1, \quad \widetilde{w}_k \ge 0;$$

2)вычисляем характеристику всей системы, учитывая веса блоков.

Определение информативности интегральной характеристики изложено ниже.

3 Информативность интегральной характеристики изменения качества системы

В алгоритмах вычисления интегральной характеристики по ОСШ-методу [5-8] используется традиционное для МГК понятие информативности, которое определяет число главных компонент *l,* используемых для вычисления интегральной характеристики.

$$\gamma_\sigma = \frac{\lambda_1 + \lambda_2 + \ldots + \lambda_l}{\lambda_1 + \lambda_2 + \ldots + \lambda_n} \ge \theta \qquad (4)$$

Однако размерность пространства признаков в задачах вычисления интегральной характеристики качества сложной системы не является чересчур большой, и вычислительных

проблем при определении собственных чисел и векторов не возникает. А качественное описание структуры системы требует либо всех главных компонент, либо их достаточно большого количества. Может оказаться, что ценная для конкретной задачи информация содержится как раз в направлениях с меньшей дисперсией. Например, при создании цифровой модели рельефа, которая строится по оцифрованным снимкам, восьмая и девятая главные компоненты дают искомый рельеф, а главные компоненты 12 и 13 в методе «Гусеница» свидетельствует о наличии в анализируемых данных периодики с дробным периодом [9, 10].

Подходы к оценке числа главных компонент по необходимой доле объясненной дисперсии формально применимы всегда, однако неявно они предполагают, что нет разделения на «сигнал» и «шум», и любая заранее заданная точность имеет смысл. При разделении данных на полезный сигнал и шум задаваемая точность теряет смысл и требуется переопределить понятие информативности.

Выбор порогового значения ОСШ для рассматриваемой системы определяет выбор связанного с ОСШ параметра информативности θ, определяющего относительную доля разброса γ_σ , приходящуюся на первые l главных компонент согласно (4). Если информативность $\gamma = \gamma_\sigma$ выражается в долях единицы, величину отношения сигнал/шум можно представить как отношение полезной части используемой информации γ к неиспользуемой информации 1-γ, и тогда $SNR = \frac{\gamma}{1-\gamma}$ Если рассматриваемое значение отношения сигнал/шум не менее порогового значения θ, то справедлива оценка информативности

$$\gamma \geq \frac{\theta}{\theta+1} . \quad (5)$$

Соотношение (5) дает априорную оценку (снизу) информативности выбранной системы признаков в зависимости от используемого значения ОСШ. В таблице 1 представлены некоторые значения, связывающие рассматриваемые показатели.

Увеличивая значения ОСШ, можно надеяться, что информативность будет выше, при этом число используемых эмпирических главных компонент для вычисления интегральной характеристики увеличивается. Однако величина SNR=2,2 является оптимистичной величиной для статистических данных, и при увеличении этого значения хотя бы до трех единиц большая часть переменных эмпирических главных компонент (ЭГК) окажется просто нулевыми, что не увеличит их информативности.

Таблица 1. *Связь дисперсионной информативности с используемым значением ОСШ*

SNR	1.2	1.5	2.0	2.2	3.0	4.0	5.7	9.0	19.0
Информативность, γ_{SNR}	0.55	0.60	0.67	0.69	0.75	0.80	0.85	0.90	0.95

Очевидно, что ОСШ-информативность ЭГК определяется параметрами найденных действующих переменных. Аналогично дисперсионной информативности согласно (4) можно определить ОСШ-информативность для выбранного числа эмпирических главных компонент N:

$$\gamma_{SNR} = \frac{S_{11}+S_{12}+\ldots+S_{1N}}{S_{21}+S_{22}+\ldots+S_{2N}} , \quad (6)$$

где S_{1k} – сумма величин ОСШ у действующих переменных k-ой ЭГК, $S_{2k,}$ – сумма ОСШ всех переменных k-ой ЭГК. В отличие от дисперсионной информативности, ОСШ-информативность не может достигать 100% по логике построения. Информативность выбранной системы признаков определяется дисперсионной и ОСШ-информативностью:

$$\gamma = \gamma_\sigma \cdot \gamma_{SNR} . \quad (7)$$

4 Вычисление интегральных индикаторов качества жизни с помощью предложенного алгоритма

Рассмотрим работу алгоритма для вычисления интегральной характеристики качества жизни. Для построения интегрального индикатора воспользуемся списком переменных из исследования, выполненного под руководством С.А. Айвазяна (табл. 2), являющегося признанным авторитетом в исследовании качества жизни [11]. Деление переменных на блоки произведено авторами исследования [11].

Таблица 2. *Переменные для вычисления интегральных индикаторов качества жизни населения*

	Уровень благосостояния населения
1	Отношение ВРП на душу населения к прожиточному мин, единиц
2	Покупательская способность среднедушевых денежных доходов по отношению к наборам прожиточного минимума (%)
3	Доля численности населения с денежными доходами ниже величины прожиточного минимума
4	Отношение совокупных доходов 20% самых богатых и 20% самых бедных
5	Обеспеченность населения собственными легковыми автомобилями (автомобилей/1 000 чел.)
6	Доля семей, состоящих на учете на получение жилья (%)
7	Приходится общей площади жилищного фонда на одного жителя (кв. м/10 чел.)
8	Доля ветхого и аварийного жилья (%)
9	Плотность автомобильных дорог общего пользования (км/10 000 кв. км)
	Качество населения
10	Ожидаемая продолжительность жизни при рождении (лет)
11	Число умерших детей в возрасте до 1 года на 1000 населения (чел.)
12	Коэффициент естественного прироста (на 1000 населения)
13	Число умерших от инфекционных и паразитарных болезней и туберкулеза на 100 000
14	Число умерших от новообразований на 100 000 населения (чел.)
15	Число умерших от болезней системы кровообращения на 100 000 населения (чел.)
16	Число умерших от болезней органов дыхания на 100 000 населения (чел.)
17	Число умерших от болезней органов пищеварения на 100 000 населения (чел.)
18	Заболеваемость от травм, отравлений и некоторых других внешних причин на 100 000
19	Число инвалидов на 1 000 населения (чел.)
20	Зарегистрировано случаев заболевания врожденными аномалиями на 1 000 населения
21	Доля специалистов с высшим образованием среди занятых в экономике (%)
22	Приведенная производительность труда (ВРП на среднегодовую численность занятых в экономике, тыс. руб./чел)
23	Количество выпускников высших и средних учебных заведений на 1000 населения
	Качество социальной сферы
24	Уровень безработицы (%)
25	Доля работников занятых во вредных и опасных условиях труда в среднегодовой численности занятых в экономике (%)
26	Численность пострадавших на производстве со смертельным исходом или с утратой трудоспособности на 1 раб. день и более на 1 000 работающих (чел.)
27	Коэффициент миграционного прироста на 10 000 населения
28	Число зарегистрированных умышленных убийств и покушений на убийство на 100 000 населения (случаев)
29	Число зарегистрированных фактов умышленного причинения тяжкого вреда здоровью на 100 000 населения (случаев)
30	Число зарегистрированных изнасилований и покушений на изнасилование на 100 000 населения (случаев)
31	Число разбоев, грабежей, краж из квартир граждан в расчете на 100 000 населения (случаев)
32	Число зарегистрированных незаконных присвоений или растрат в расчете на 100 000 населения (случаев)
33	Число больных, состоящих на учете с диагнозом наркомания и токсикомания в расчете на 100 000 населения (чел.)
34	Число больных, состоящих на учете с диагнозом алкоголизм в расчете на 100 000 населения
35	Число больных, инфицированных туберкулезом в расчете на 100 000 населения (чел.)
36	Смертность от внешних причин (на 100 000 человек населения)
37	Число больных психическими расстройствами на 100 000 населения

Отсутствующие в открытом доступе данные заменим на близкие к ним (выделено цветом в табл.2). Все значения переменных взяты из открытых справочников Росстата [12]. Отсутствующие единичные значения дополнены средними значениями по тому федеральному округу, к которому относится рассматриваемый субъект. Отсутствующие значения показателей для ряда лет восстановлены линейной интерполяцией.

Среди перечисленных переменных переменные 1, 2, 5, 7, 9, 10,12, 21, 22, 23 связаны с вычисляемой характеристикой монотонной возрастающей зависимостью, когда оптимальное значение *j*-го показателя максимально.

Для остальных показателей, кроме переменной 27, оптимальное значение показателя минимально. Для переменной 27 «Коэффициент миграционного прироста» примем x_j^{opt} =0, j=27. Значение соответствует ситуации, когда количество выбывших из региона совпадает с количеством приехавших.

Согласование направлений собственных векторов и определение четвертой эмпирической главной компоненты представлено в табл. 3. Направления собственных векторов, соответствующих четвертому по убыванию собственному числу для каждого наблюдения выбраны, чтобы максимизировать сумму значение ОСШ у действующих переменных, для которых выполнено ОСШ-условие: вычисленное отношение сигнал/шум не менее 2,2. Темным цветом в табл. 3 выделены действующие переменные.

Таблица 3. Определение четвертой эмпирической главной компоненты Блока 1

4	Переменные								
ГК	1	2	3	4	5	6	7	8	9
2007	-0.25	0.16	0.40	0.23	-0.32	-0.54	0.41	-0.12	0.36
2008	-0.17	0.03	0.38	0.30	-0.16	-0.41	0.69	-0.23	-0.07
2009	0.07	-0.04	0.27	0.33	-0.15	-0.56	0.66	-0.19	0.05
2010	0.07	-0.05	0.37	0.30	-0.24	-0.41	0.68	-0.24	-0.16
2011	0.18	0.01	0.36	0.38	-0.28	-0.03	0.56	-0.55	-0.06
2012	0.02	0.08	0.32	0.38	-0.22	0.30	0.31	-0.71	-0.03
Среднее, m	-0.01	0.03	0.35	0.32	-0.23	-0.27	0.55	-0.34	0.01
Выб. ср.квадр.откл., s	0.16	0.08	0.05	0.06	0.07	0.34	0.16	0.23	0.18
ОСШ	0.09	0.41	7.57	5.51	3.42	0.80	3.46	1.45	0.08
	Сумма ОСШ ЭГК								22.80
	Сумма действующих ОСШ ЭГК								19.97

В таблице 4 представлены все нормализованные и согласованные по направлению эмпирические главные компоненты (ЭГК) первого блока. Факторные нагрузки ЭГК имеют разные знаки.

Таблица 4. Эмпирические главные компоненты Блока 1

№	Собственные	Переменные								
ГК	числа	1	2	3	4	5	6	7	8	9
1	3.07	0.73	0.88	0.79	-0.77	0.55		0.30	0.27	
2	2.03	-0.22	-0.30		0.38	0.40	0.65	0.64	0.70	
3	1.14	-0.23	0.20			-0.40				0.77
4	0.79			0.31	0.28	-0.20		0.49		
5	0.63		0.08				0.48		-0.36	
6	0.51				0.03	0.19			0.00	
7	0.46	0.44	-0.09	-0.23						0.30

8	0.30	0.17		0.25	0.33			-0.25	0.13	-0.09
9	0.08	0.00	0.22	-0.12	0.13				0.02	

В таблице 5 приведен пример определения информативности интегрального показателя Блока 1: Уровень благосостояния населения согласно (6) и (7).

Таблица 5. Определение информативности интегрального показателя Блока 1

	Номер эмпирической главной компоненты								
	1	2	3	4	5	6	7	8	9
Сумма ОСШ k –ой ЭГК	99.2	67.1	30.3	22.8	12.9	12.9	21.9	46.7	59.7
Сумма действующих ОСШ k –ой ЭГК	98.6	64.5	23.6	20.0	10.1	8.1	17.0	43.2	55.6
Накопленные %%, γ_{SNR}	26.4	43.7	50.0	55.3	58.0	60.2	64.8	76.3	91.2
Эмпирические собственные числа	3.1	2.0	1.1	0.8	0.6	0.5	0.5	0.3	0.1
Накопленные %%, γ_{σ}	34.1	56.6	69.2	78.0	85.0	90.7	95.8	99.1	100.0
Информативность, $\gamma_{\sigma} \cdot \gamma_{SNR}$, %	9.0	24.7	34.6	43.1	49.3	54.6	62.0	75.6	91.2

Суммарная информативность этого блока увеличивается при увеличении числа используемых ЭГК. При рассмотрении всех 9 ЭГК суммарная информативность максимальна и составит около 91%.

Веса всех блоков определяем пропорционально их информативности (табл.6). Эта таблица дает также суммарную оценку информативности решения: действующие переменные объясняют в структуре данных 83,9% информации сигнала при использовании всех ЭГК, когда дисперсионная информативность составляет 100%. Это максимально возможное в этом случае значение информативности.

Таблица 6. Определение весов блоков

Блок	1	2	3	По всем блокам
Сумма ОСШ	373.4	875.1	431.9	1680.4
Сумма действующих ОСШ	340.6	742.4	326.5	1409.5
ОСШ-информативность блока, γ_{SNR} %	91.2	84.8	75.6	83.9
Дисперсионная информативность блока, γ_{σ} %	100	100	100	100
Информативность блока γ , %	91.2	84.8	75.6	83.9
Вес блока	0.242	0.527	0.232	

На рис.4 показаны изменения интегральной характеристики качества жизни для некоторых субъектов России. Следует отметить, что из 37 рассматриваемых показателей 20 отражают физиологическое благополучие человека, что и объясняет (вкупе с особенностями национальной статистики) фантастические показатели национальных республик (Ингушетии, Чечни, Дагестана и пр.). В этих субъектах России значительно меньше шансов заболеть, умереть, подвергнуться криминальному насилию. Ожидаемая продолжительность жизни в Ингушетии наибольшая по России, в частности, превышает этот параметр по Новгородской области на 10 лет. С учетом этого обстоятельства низкий уровень ВРП и высокий уровень безработицы не кажутся столь уж важными в интегральной характеристике качества жизни – меньше работать и дольше жить предпочтительнее, чем наоборот.

Заключение

В работе рассмотрено решение задачи построения латентной интегральной характеристики изменения качества системы на основании регистрируемых измерений для ряда наблюдений. Задача относится к классу задач выделения полезного сигнала из массива данных в условиях априорной неопределенности и решается на основании задаваемого отношения сигнал/шум. Предложен алгоритм построения интегральных характеристик с определением неслучайных составляющих главных компонент, характеризующих структуру рассматриваемой

системы. Также предложен новый подход к выбору числа главных компонент, к определению весов рассматриваемых подсистем и к определению информативности полученной характеристики на основании дисперсионного критерия и выбранного параметра отношения сигнал/шум. Алгоритм был использован для получения интегральных индикаторов качества жизни населения субъектов Российской Федерации. Предлагаемая методика может быть использована для вычисления интегральных оценок изменения качества плохо формализуемых систем.

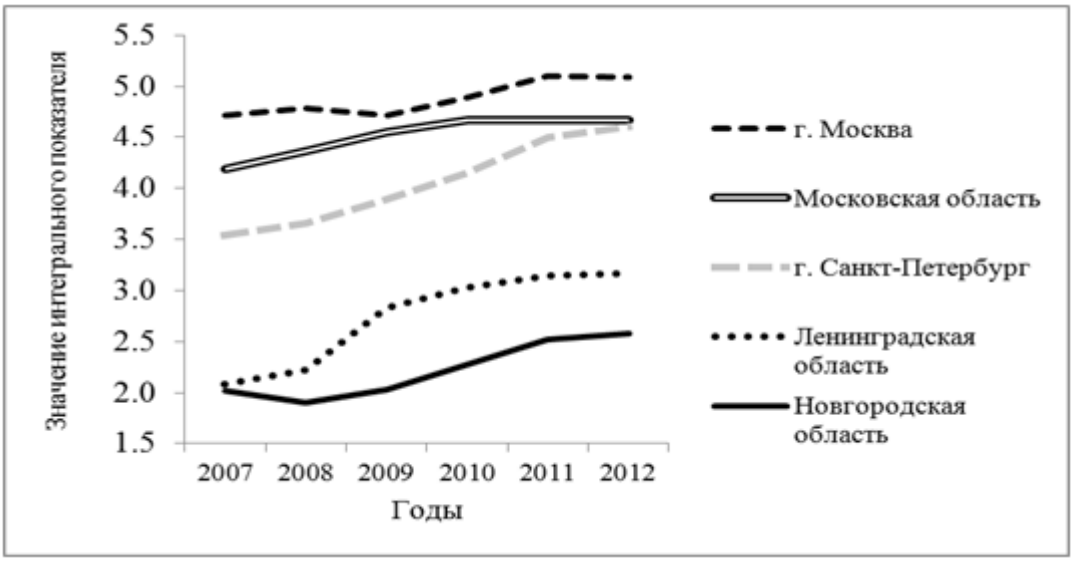


***Рис. 4**. Интегральная характеристика качества жизни некоторых субъектов Российской Федерации за 2007-2012годы*

Работа выполнена при финансовой поддержке проектной части государственного задания в сфере научной активности Министерства образования и науки Российской Федерации, проект № 1.949.2014/К.

Литература

1. Рейтинги стран мира / РВК-рейтинг, 1995-2015.– URL: http://rating.rbc.ru/category.shtml?macro. – (дата обращения: 13.09.2015).
2. Доклады ООН о развитии человека (Human Development Reports). 1990-2013. /United Nations Development Programme, 1990-2014. — URL: http://hdr.undp.org/en/reports/ (дата обращения: 13.09.2015).
3. Айвазян С. А. Интегральные индикаторы качества жизни населения: их построение и использование в социально-экономическом управлении межрегиональных сопоставлениях. — М.: ЦЭМИ РАН, 2000. — 56с.
4. Айвазян С.А. К методологии измерения синтетических категорий качества жизни населения // Экономика и математические методы. — 2003. — Т. 39. — № 2. — С. 33-53.
5. Жгун Т.В. Построение интегральной характеристики изменения качества системы на основании статистических данных как решение задачи выделения сигнала в условиях априорной неопределенности // Вестн. Новг. гос. ун-та. Сер.: Технические науки. — 2014. — № 81. — С.10-16.
6. Жгун Т.В. Построения интегральной характеристики демографического развития территорий на примере муниципальных образований Новгородской области //Региональная экономика: теория и практика. —2013. — № 36(315),сентябрь. — С.2-12.
7. Жгун Т.В. Вычисление интегрального показателя эффективности функционирования динамической системы на примере интегральной оценки демографического развития муниципальных образований Новгородской области // Вестн. Новг. гос. ун-та. Сер.: Физико-математические науки. — 2013. — № 75. — Т.2. —С.11-16.
8. Жгун Т.В. Исследование формальных методов построения латентной характеристики качества систем // Вестн. Новг. гос. ун-та. Сер.: Физико-математические науки. — 2014. — № 80. —.13-19.
9. Голяндина Н.Э., Усевич К.Д., Флоринский И.В. Анализ сингулярного спектра для фильтрации цифровых моделей.//Геодезия и картография 2008. — №5. — С. 21-28. URL://http://iflorinsky.narod.ru/Florinsky-2008c.pdf (дата обращения: 13.01.2014).
10. Голяндина Н.Э. Метод «Гусеница»– SSA: анализ временных рядов.– Спб.: Спб ун-т, 2004.– 74 с.
11. Исакин М.А. Модификация метода k -средних с неизвестным числом классов // Прикладная эконометрика. — 2006. — Выпуск № 4. —С. 62-70.
12. Федеральная служба гос. статистики / Федеральная служба гос. статистики. – М., 2002-2014. - URL: http://www.gks.ru/wps/wcm/connect/rosstat_main/rosstat/ru/statistics/publications/catalog/doc_1137674209312 (дата обращения: 13.09.2015).

Понятский В.М.[1], Грубенко А.Д.[2], Кислинский И.В.[3], Федорищева В.Г.[4]

[1] АО Конструкторское бюро приборостроения им. академика А.Г. Шипунова, г. Тула, начальник бюро, Тульский государственный университет, к.т.н., доцент кафедры Проектирование автоматизированных комплексов, pwmru@rambler.ru

[2] АО Конструкторское бюро приборостроения им. академика А.Г. Шипунова, г. Тула, инженер

[3] АО Конструкторское бюро приборостроения им. академика А.Г. Шипунова, г. Тула, ведущий инженер, Nick-51-Rus@yandex.ru

[4] АО Конструкторское бюро приборостроения им. академика А.Г. Шипунова, г. Тула, ведущий инженер

ИСПОЛЬЗОВАНИЕ ИТ НА ОСНОВЕ ПК MATLAB ДЛЯ МОДЕЛИРОВАНИЯ КАНАЛА ОБМЕНА CAN ДЛЯ СИСТЕМ УПРАВЛЕНИЯ

КЛЮЧЕВЫЕ СЛОВА

Системы управления, блок, канал обмена, прием, передача, шина данных.

АННОТАЦИЯ

В настоящее время в сложных технических комплексах широко используется протокол сетевого обмена CAN (Control Area Network). При достаточно большом количестве сигналов возможна предельная загрузка шины данных. Это зачастую приводит к сбоям, а в некоторых случаях к потерям информации. Для анализа и предотвращения подобных ситуаций целесообразно использовать средства компьютерного моделирования.

Введение

В настоящее время в сложных технических комплексах широко используется протокол сетевого обмена CAN (Control Area Network). При достаточно большом количестве сигналов возможна предельная загрузка шины данных. Это зачастую приводит к сбоям, а в некоторых случаях к потерям информации. Для анализа и предотвращения подобных ситуаций целесообразно использовать средства компьютерного моделирования.

В программном комплексе Matlab имеется библиотека блоков SimEvents, которая позволяет моделировать обмен сигналов через шину данных по заданному протоколу обмена. Из библиотеки блоков SimEvents создаются функциональные узлы, которые имитируют работу передатчиков и приёмников сообщений, задаются атрибуты сигналов (идентификаторы, размер сообщения, содержимое сообщения), создаётся имитация шины обмена данных с такими параметрами, как скорость передачи данных и время обработки сообщения. На этапе моделирования выявляются ситуации, когда по разным причинам могут быть потеряны сигналы, с тем, чтобы дать рекомендации по устранению сбоев.

С использованием библиотеки SimEvents программного комплекса Matlab по заданному протоколу обмена разработана модель, соответствующая логике взаимодействия составных частей технической системы, имеющей порядка десяти блоков и сорока сигналов. Проведенное моделирование обмена сигналов по протоколу CAN позволило получить циклограмму работы канала, оценить количество пропущенных сообщений и загрузку канала при разных скоростях обмена, а также показало эффективность предложенного подхода.

Проведено моделирование протокола взаимодействия блоков реальной системы управления по каналу CAN с использованием ПО Matlab, пакетов Simulink и SimEvents.

Протокол информационного взаимодействия подсистем комплекса по каналу CAN использует стандартный формат кадра CAN, соответствующий спецификации CAN 2.0A (размер идентификатора 11 бит).

Обмен данными предназначен для организации и управления информационным взаимодействием между составными частями системы управления в рамках разработанного протокола. Время обработки кадра данных в канале CAN определяется по формуле:

$$t = \frac{44 + 8 \cdot N}{V},$$

где N - количество байт в сообщении; V - скорость передачи информации по каналу, бит/с.

Моделирование передачи сообщений строится следующим образом: с информационных блоков устройств поступают периодические синусоидальные сигналы (Time-based signal), с помощью блока передатчика преобразуются в события (Time-based Entity Generator) с идентификаторами, которые задаются в маске подсистемы, поступают на шину в соответствии с приоритетом. Прием события осуществляется в блоке приемника, где происходит обратное преобразование события в периодический сигнал (Event to Time Signal) в соответствии с идентификатором. Полученные сигналы для сравнения с исходными передаваемыми сигналами отображаются в блоке Scope. Подсистемы устройств, интерфейсного блока и шины обмена данных представлены на рис. 1 — 3.

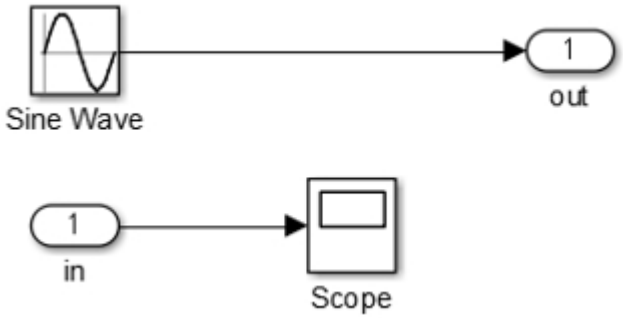
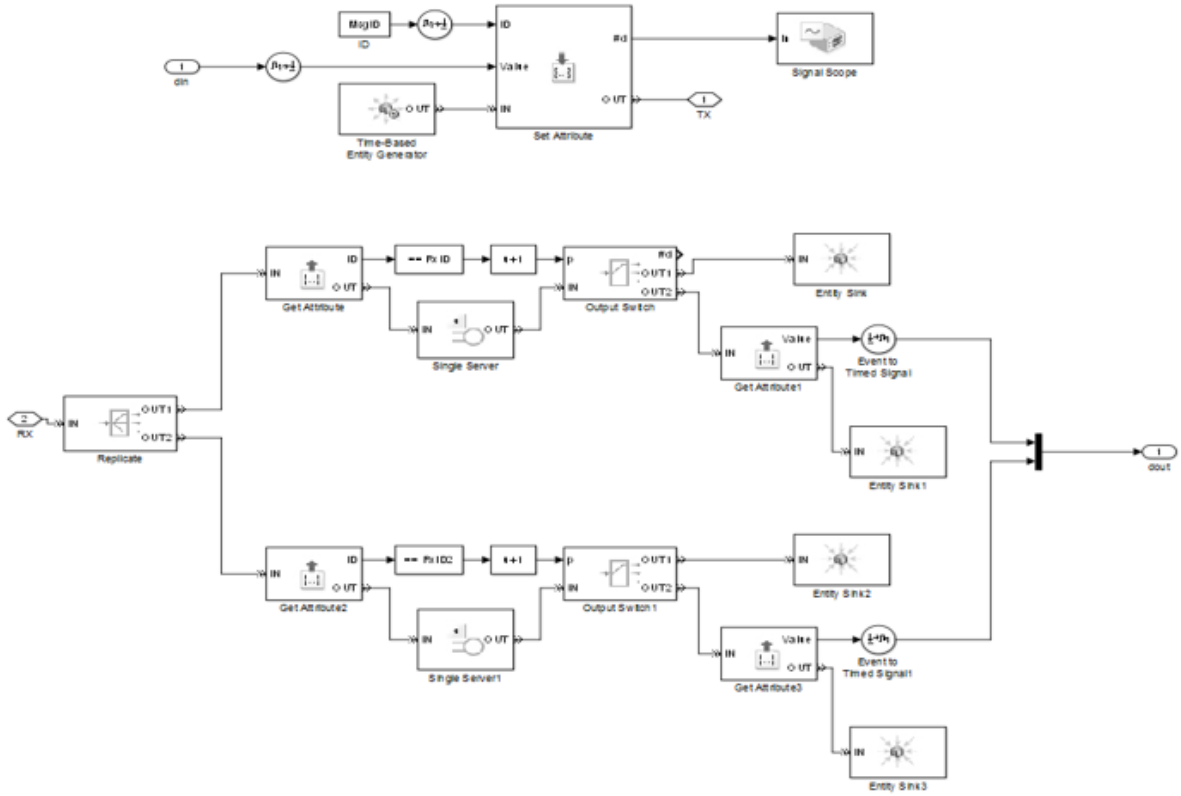


Рис.1 Модель подсистемы устройства

а)



б)

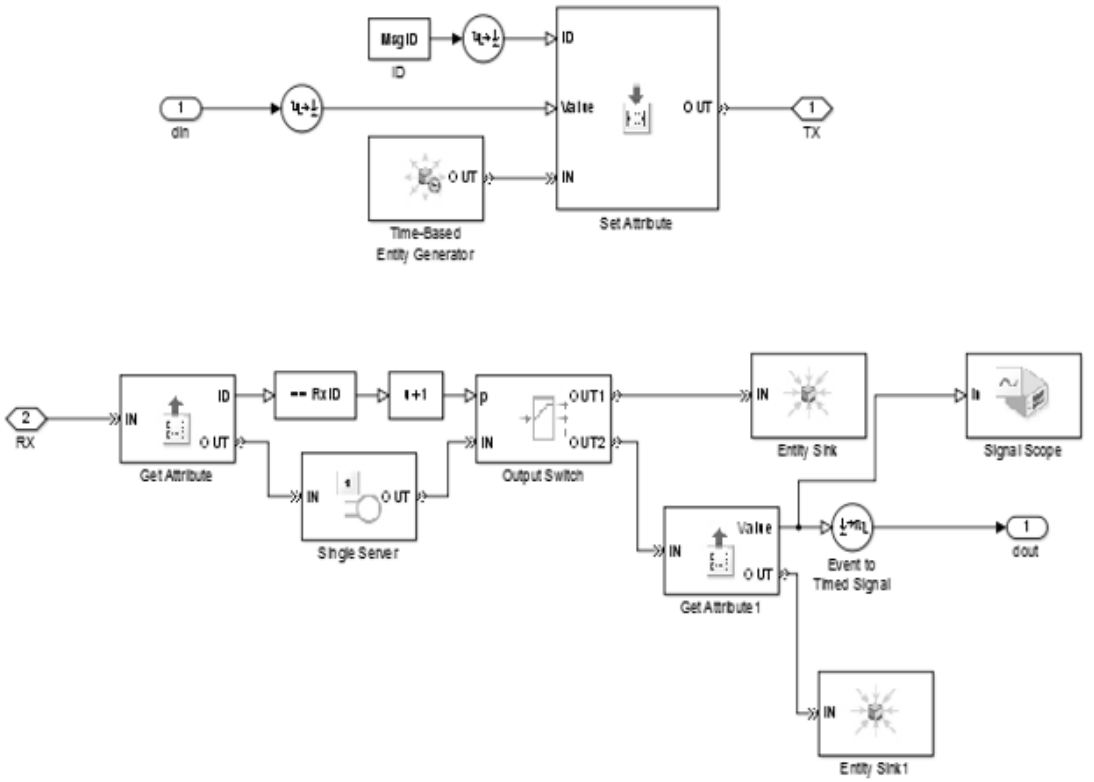
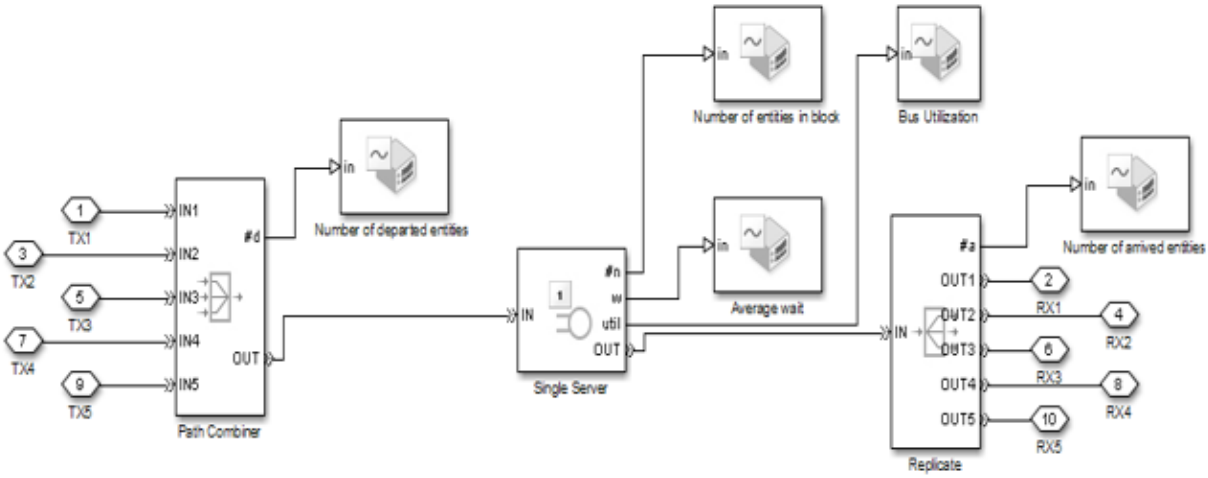


Рис. 2 Модель подсистемы интерфейсного блока приема и передачи сообщений: а – прием одного сообщения (блоки CAN Node1 - CAN Node4); б – прием двух сообщений (блок CAN Node)

a)



б)

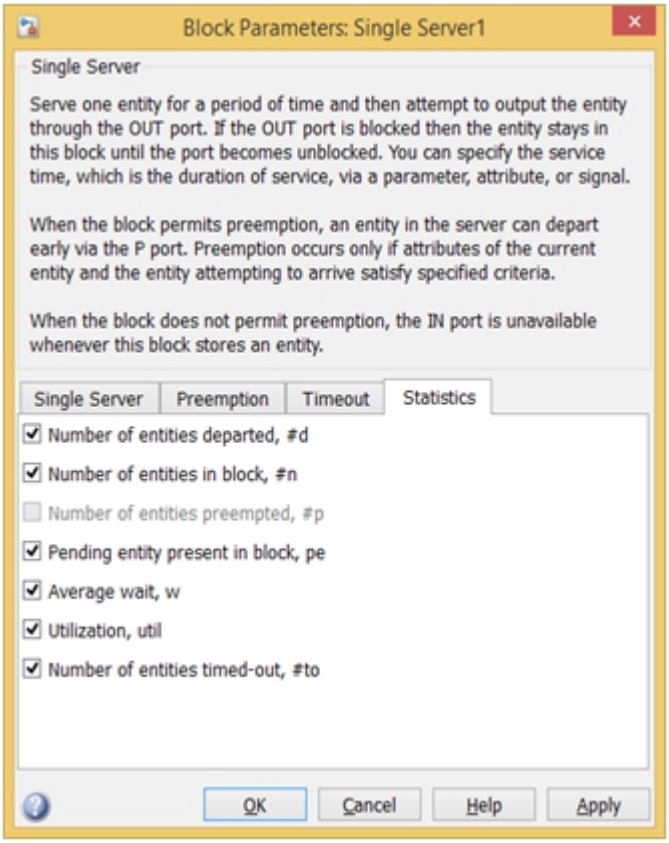


Рис. 3 Модель подсистемы шины CAN: а – Simulink-модель шины CAN с блоками вывода статистики; б – вкладка Statistics окна параметров блока сервера

С помощью блока осциллографа Signal Scope, подключаемого к блокам Simulink-модели, можно выводить статистику прохождения сообщений между устройствами. На рис. 3а приведена Simulink-модель шины CAN с блоками вывода статистики:

- количество сообщений, поступающих в блок (number of arrived entities);
- количество сообщений, уходящих из блока (number of departed entities);
- количество сообщений в блоке (number of entities in block);
- загрузка блока (utilization);
- среднее время ожидания сообщений (average wait).

В соответствии с протоколом передача сообщений между устройствами происходит циклически. Графики переданных сигналов от одного блока другому с ID 102(101) приведены на рис. 4 - 5.

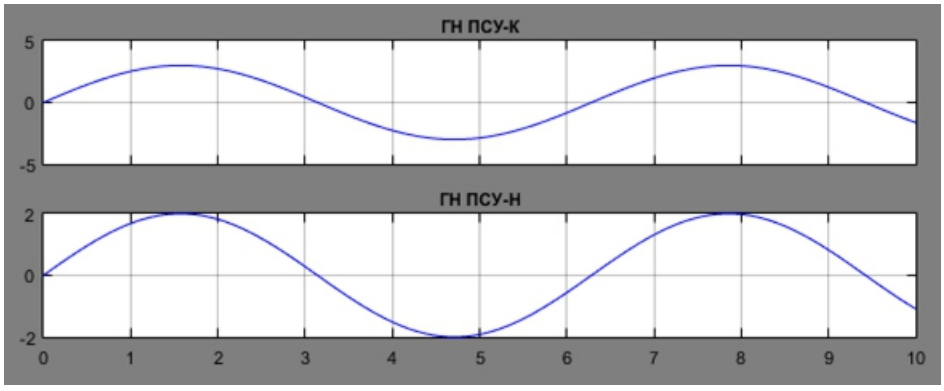


Рис.4 Переданные сигналы с идентификаторами 102 и 101

На выходе приемника сообщений (рис.7) можно увидеть, что полученные сигналы и исходящие сигналы (рис. 6) совпадают.

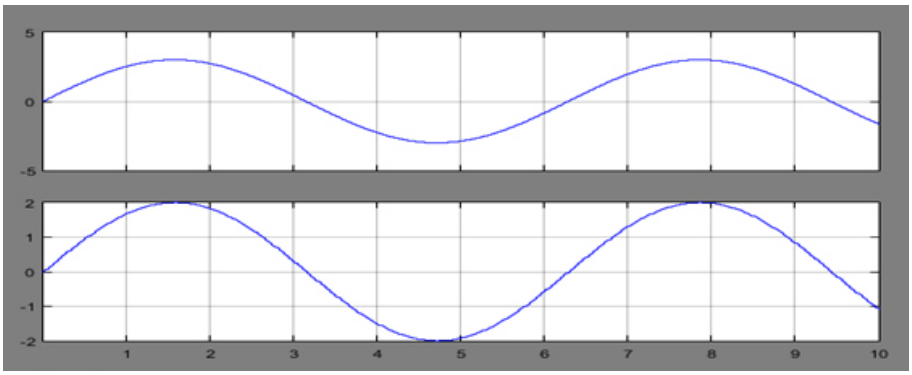


Рис. 5 Полученные сигналы

На рис.6 - 7 приведены спектры передаваемого и принимаемого сигнала с идентификатором 101.

Уменьшение число пропаданий сообщений при асинхронной и неуправляемой передаче данных по шине от блоков СУО можно обеспечить путем минимизации комплексного критерия:

$$\min_{(ID_i,T_i)\in\Pi} \Phi = \min_{(ID_i,T_i)\in\Pi} \sum_{ij} k_i \frac{(s_{вх\,i} - s_{вых\,i})^2}{s_{вх\,i} s_{вых\,i}} (\sum_i k_i)^{-1},$$

где k_i – весовой коэффициент; $s_{вх\,i}$ – исходный сигнал; $s_{вых\,i}$ – принятый сигнал.

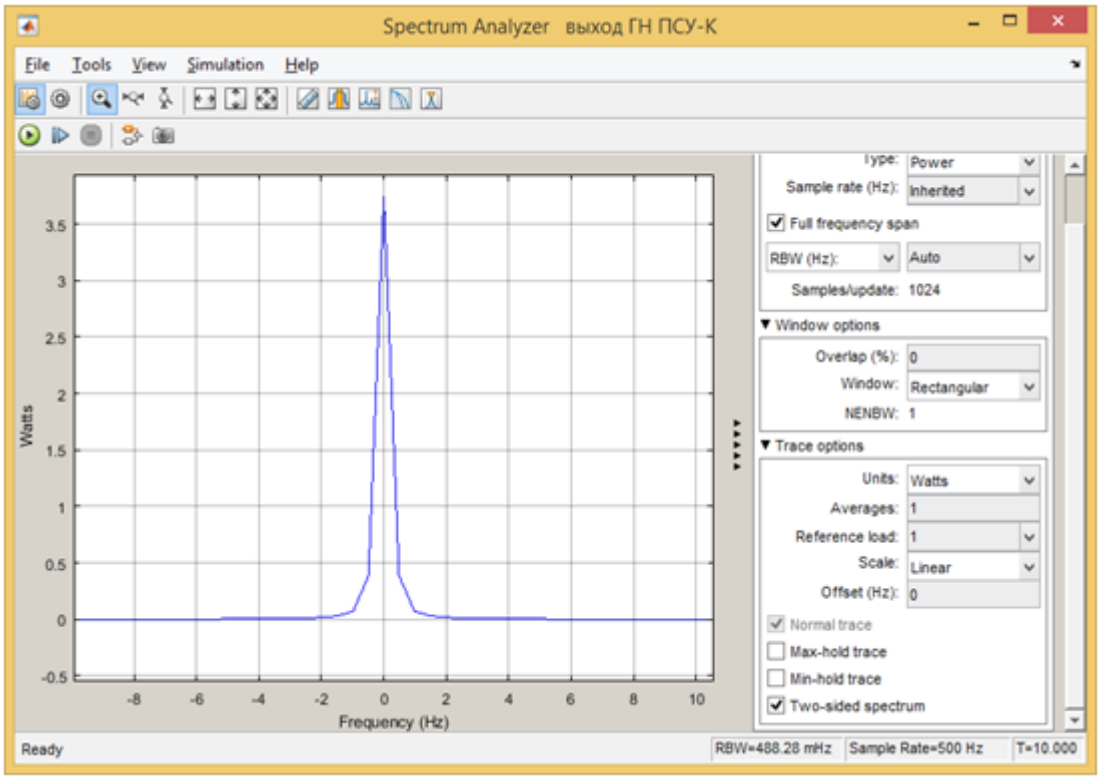


Рис.6 Спектр переданного сигнала с идентификатором 101

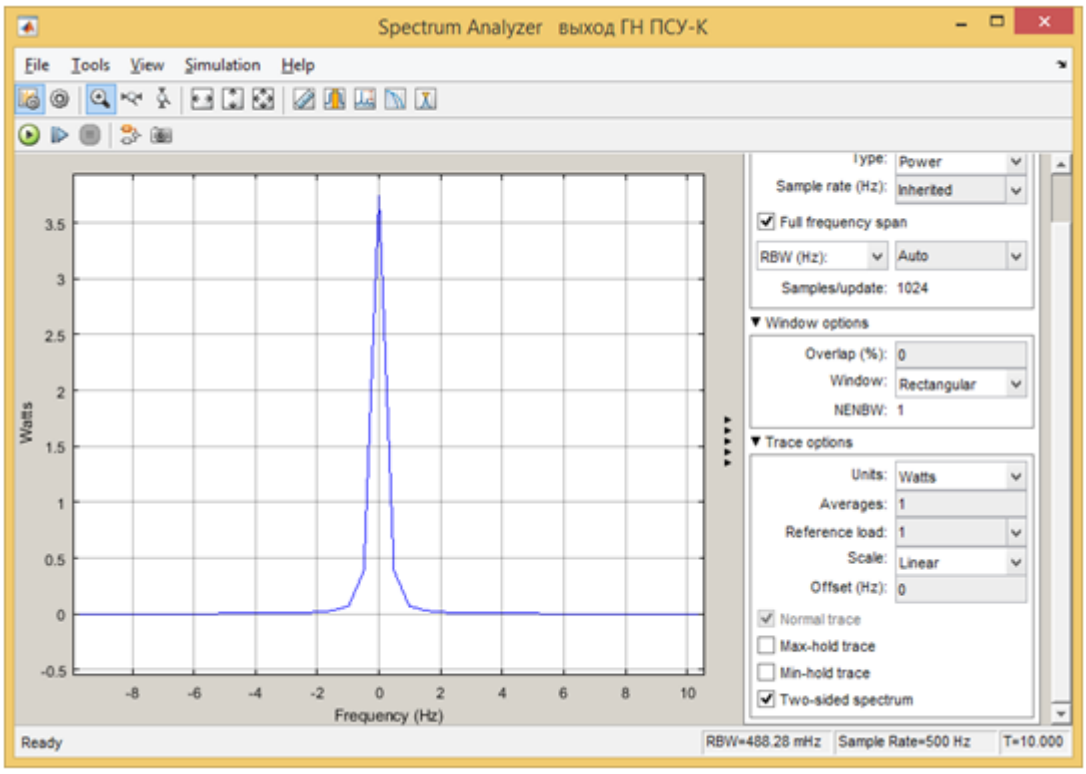


Рис.7 Спектр полученного сигнала с идентификатором 101

На рис. 8 представлен график загрузки канала - процентная зависимость количества сообщений, одновременно попадающих на шину в данный момент времени. В процессе моделирования было установлено, что загрузку канала соответствует стандарту ISO 11898 для протокола CAN2.0A.

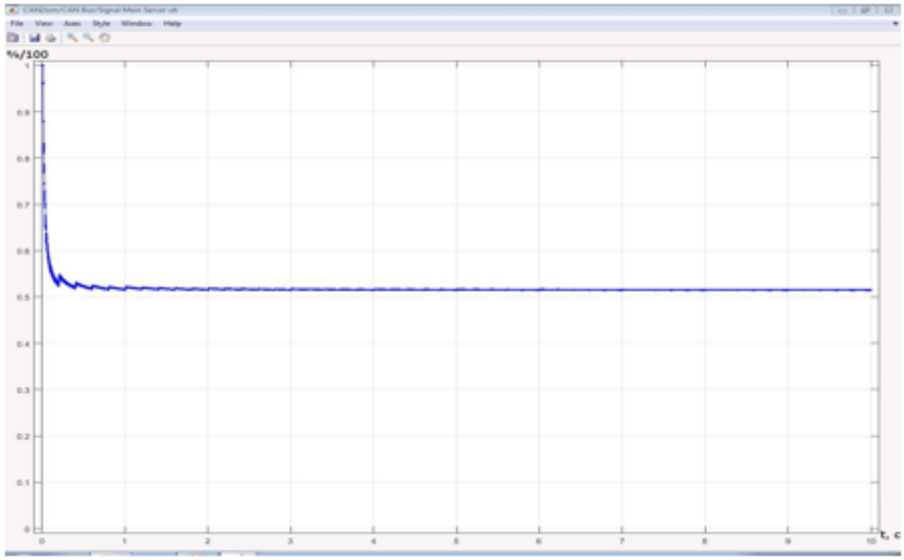


Рис.8 График загрузки канала CAN при скорости 500 Кбит/с

Таким образом, рассмотрены возможности ИТ на основе ПК Matlab по моделированию протокола обмена между блоками системы управления. Разработанная модель протокола взаимодействия блоков устройств реальной системы управления по каналу CAN позволила оценить как загрузку канала, так и задержки и потери сообщений.

Понятский В.М.[1], Федорищева В.Г.[2], Богданова Л.А.[3], Игумнова Т.[4]

[1] АО Конструкторское бюро приборостроения им. академика А.Г. Шипунова, г. Тула, начальник бюро, Тульский государственный университет, к.т.н., доцент кафедры Проектирование автоматизированных комплексов, pwmru@rambler.ru

[2] АО Конструкторское бюро приборостроения им. академика А.Г. Шипунова, г. Тула, ведущий инженер

[3] АО Конструкторское бюро приборостроения им. академика А.Г. Шипунова, г. Тула, начальник сектора, д.т.н.

[4] АО Конструкторское бюро приборостроения им. академика А.Г. Шипунова, г. Тула, ведущий инженер, к.т.н.

ИТ НА ОСНОВЕ POLYSPACE MATLAB ДЛЯ ОТЛАДКИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ВСТРОЕННЫХ МИКРОПРОЦЕССОРОВ СИСТЕМ УПРАВЛЕНИЯ

КЛЮЧЕВЫЕ СЛОВА

Системы управления, программное обеспечение, отладка, ошибки, модель.

АННОТАЦИЯ

В настоящее время актуально использование автоматизированных технологий тестирования и отладки программного обеспечения для встроенных вычислительных систем. Инструменты Polyspace программного комплекса MathWorks Matlab помогают обнаружить ошибки выполнения и подтвердить отсутствие специфических ошибок в исходном коде на языке C/C++ без исполнения кода.

Введение

Методы, традиционно используемые для выявления и отладки ошибок разработанного программного обеспечения, можно разделить на ручное рассмотрение кода и динамическое тестирование. Ручное рассмотрение кода может быть эффективным методом поиска ошибок в относительно небольших приложениях размером в несколько сотен строк кода. Для программ большего размера ручное рассмотрение является трудозатратным процессом.

Динамическое тестирование требует от инженеров написания и выполнения тестов. Для типичного встраиваемого приложения могут потребоваться многие тысячи тестовых векторов. Динамическое тестирование основывается на методе проб и ошибок, что предоставляет только частичное покрытие всех возможных комбинаций значений, которые могут встретиться во время выполнения. Как и ручное рассмотрение кода, этот процесс является трудозатратным. Время, потраченное на создание тестов и ожидание разработки исполняемой системы, часто приводит к тому, что динамическое тестирование смещается в конец цикла разработки программного обеспечения, когда ошибки наиболее дорого исправлять.

Инструменты Polyspace программного комплекса (ПК) MathWorks Matlab помогают обнаружить ошибки выполнения и подтвердить отсутствие специфических ошибок в исходном коде на языке C/C++. Коды при этом могут быть написаны вручную, автоматически сгенерированы или их комбинацией. При помощи методов математического анализа, реализованных в продуктах верификации Polyspace, можно подтвердить отсутствие ошибок переполнения, деления на ноль, обращения к элементам вне границ массивов и других ошибок выполнения в коде. Для этого не требуется исполнения кода, применения специальных средств измерения и тестовых процедур.

Polyspace Code Prover работает совместно с Polyspace Bug Finder для поиска дефектов и проверки соответствия стандартам кодирования. Совместно эти продукты предлагают сквозное решение для статического анализа на ранних стадиях разработки, которое охватывает поиск дефектов, проверку на стандарты кодирования и доказательство наличия или отсутствия ошибок выполнения кода. Такой подход обеспечивает качество встраиваемого программного обеспечения, которое должно функционировать на высочайших уровнях надежности и безопасности.

Получить доступ к Polyspace Code Prover и Polyspace Bug Finder можно из командной строки

Matlab, графического интерфейса пользователя или при помощи плагинов для Visual Studio и Eclipse.

С использованием продуктов Parallel Computing Toolbox и Matlab Distributed Computing Server можно ускорить статический анализ кода, передавая задачи анализа кода на вычислительные кластеры.

Polyspace Bug Finder и Polyspace Code Prover совместно с инструментами Matlab IEC Certification Kit и DO Qualification Kit могут использоваться в процессе сертификации проектов по промышленным стандартам IEC 61508, ISO 26262, DO-178. Отчеты, создаваемые в процессе тестирования, документируют применение правил кодирования и отображают результаты поиска ошибок выполнения программ. Отчеты могут создаваться в форматах PDF, HTML, RTF и других форматах.

ИТ отладки программного обеспечения встроенных вычислительных систем с использованием Polyspace комплекса Matlab

Этап 1. Тестирование и отладка разработанного программного обеспечения с использованием процедуры Polyspace Bug Finder

Основные возможности Polyspace Bug Finder:

- выявление ошибок кодирования, проблем потоков данных и других дефектов в C и C++ коде;
- быстрый анализ больших объемов кода;
- проверка соответствия стандартам кодирования MISRA-C:2004, MISRA-C++:2008, JSF++ и настраиваемым правилам именования;
- расчет цикломатической сложности и других метрик кода;
- интеграция с Eclipse;
- трассируемость результатов верификации кода к моделям Simulink;
- доступ к результатам анализа Polyspace Code Prover;
- низкий уровень ложных срабатываний при выявлении программных дефектов.

Поиск ошибок кодирования и выявление их причин

Polyspace Bug Finder выявляет ошибки и нарушения правил кодирования сразу после написания, изменения или генерации кода. Ошибки кодирования или логики могут порождать сбои и некорректное поведение программы.

Polyspace Bug Finder выявляет самые разные ошибки кодирования – такие, как:

- численные (деление на ноль, переполнение, сдвиг, некорректное использование функций из стандартной библиотеки, переполнение при целочисленной конвертации);
- ошибки программирования (некорректное использование операторов = или ==, несоответствие объявлению);
- ошибки при работе со статической памятью (доступ за пределы массива, null pointer);
- ошибки при работе с динамической памятью (использование освобожденного указателя, незащищенное динамическое выделение памяти);
- ошибки потоков данных (запись без чтения, неинициализированные переменные).

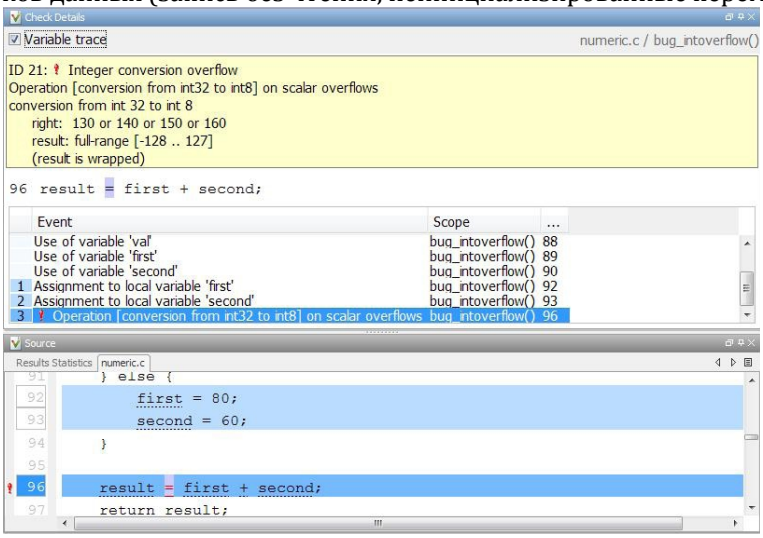


Рис. 1 Переполнение, обнаруженное при помощи Polyspace Bug Finder

Эти ошибки идентифицируются в исходном коде, приводится дополнительная информация, помогающая отследить причину ошибки и выявить ее источник. Разработчики программного обеспечения могут использовать эту информацию для того, чтобы определить, как лучше исправить найденную ошибку.

Верификация соответствия стандартам кодирования (MISRA, JSF, правила именования)

Polyspace Bug Finder поддерживает обнаружение нарушений стандартов кодирования MISRA-C:2004, MISRA-C++:2008, MISRA-AC-AGC, JSF++ и настраиваемых правил именования в коде. Polyspace Bug Finder используется для установления стандартов кодирования, чтобы улучшить читаемость и качество кода. Можно настроить Polyspace Bug Finder для проверки на все правила стандарта, только требуемые правила стандарта или выбрать индивидуальные правила, которые требуется проверять. Также можно задать собственную конфигурацию для использования одинаковых правил кодирования во всей команде разработчиков.

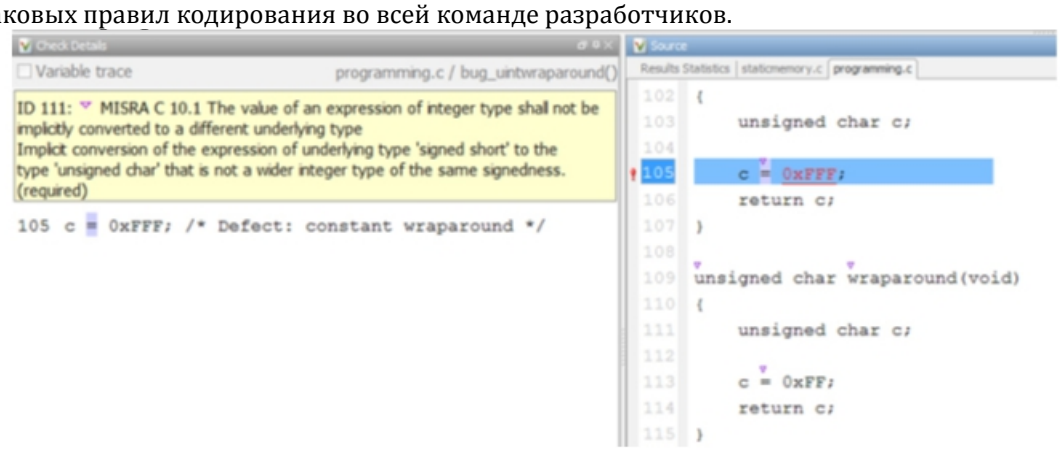


Рис.2 Идентификация нарушения стандарта кодирования MISRA

Создание метрик кода и мониторинг качества программного обеспечения

Метрики кода – это измеренные характеристики исходного кода, позволяющие оценить качество программного обеспечения. Polyspace Bug Finder генерирует метрики, относящиеся ко всему проекту, к одному файлу или к отдельной функции.

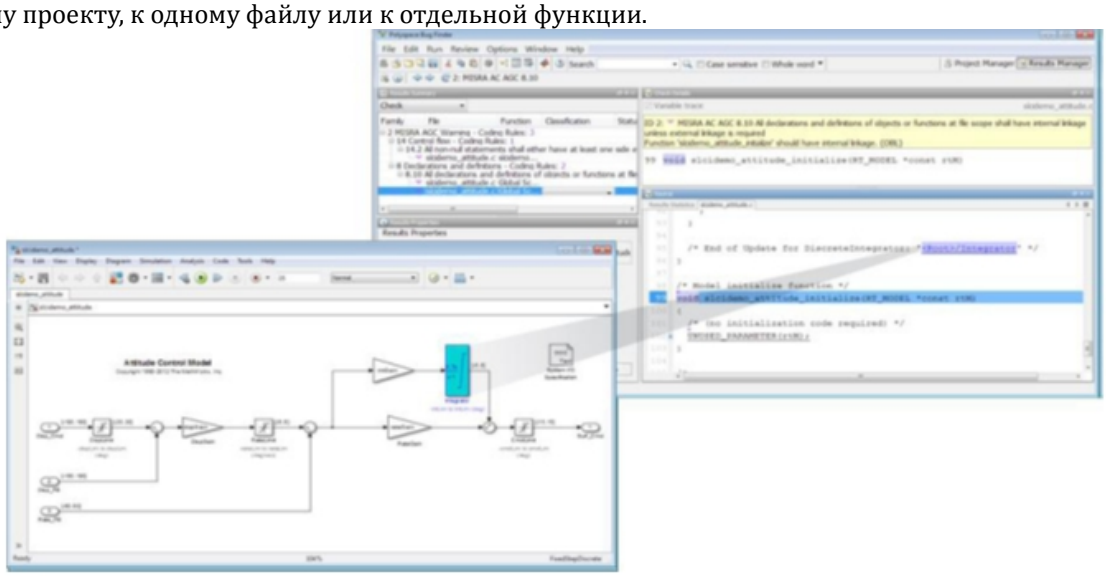


Рис.3 Трассирование нарушений правил кодирования по стандарту MISRA к модели Simulink

Метрики сложности кода включают в себя:

- цикломатическую сложность;
- плотность комментариев;
- уровень вложенности вызовов;
- количество возможных путей;
- число параметров функции;
- количество строк кода.

Используя такие метрики, можно отслеживать прогресс кода относительно заранее определенных целей качества программного обеспечения по мере развития проекта. Измеряя скорость улучшения качества программного обеспечения, Polyspace Bug Finder позволяет разработчикам, тестировщикам и менеджерам проектов ставить цели и создавать код высокого качества.

Polyspace Bug Finder позволяет получать метрики исходного кода и загружать их в отдельный репозиторий, имеющий собственный веб-интерфейс.

Трассирование результатов анализа кода к моделям Simulink

Результаты анализа ошибок в автоматически сгенерированном коде трассируются обратно к модели Simulink. Таким образом, можно определить, какие части модели являются надежными и исправить проблемы в модели, которые приводят к ошибкам в коде.

Этап 2 Тестирование и отладка разработанного программного обеспечения с использованием процедуры Polyspace Code Prover

Основные возможности Polyspace Code Prover:

- доказательство отсутствия определенных ошибок выполнения в C и C++ коде;
- расцветка ошибок выполнения непосредственно в коде;
- расчет диапазонов для переменных и значений, возвращаемых функциями;
- идентификация переменных, нарушающих указанные диапазоны;
- метрики качества для отслеживания соответствия целям качества программного обеспечения;
- веб панель, представляющая метрики кода и состояние качества проекта;
- управляемый процесс рассмотрения кода для классификации результатов и ошибок выполнения;
- графическое представление операций чтения и записи в переменные.

Верификация ошибок выполнения

Ошибки выполнения также называются скрытыми дефектами, так как они редко проявляются непосредственно. Последствия таких ошибок включают отказы и неожиданное поведение ПО во время испытаний или эксплуатации.

По результатам верификации в Polyspace Code Prover, каждая операция в C или C++ коде отмечается цветом, обозначающим статус этой операции:

Зеленый: доказано отсутствие ошибок выполнения;

Красный: доказано наличие ошибки при каждом выполнении этой операции;

Серый: доказана недостижимость операции (может указывать на функциональную проблему);

Оранжевый: недоказанная операция (ошибка может произойти при определенных условиях) (рис.4).

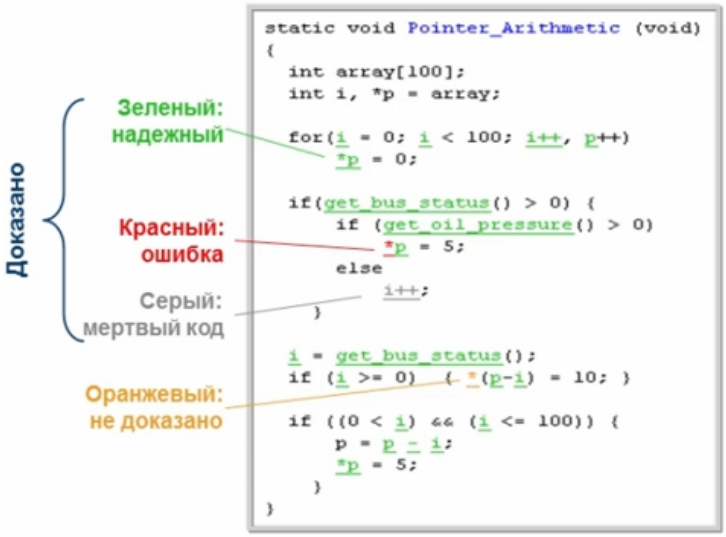


Рис.4 Расцветка ошибок выполнения в коде

Polyspace Code Prover идентифицирует, где могут произойти ошибки выполнения и код, в котором доказано отсутствие ошибок выполнения. Polyspace Code Prover доказывает отсутствие

следующих ошибок выполнения в исходном C и C++ коде:

- переполнение, исчезновение порядка, деление на ноль, извлечение квадратного корня из отрицательного числа и другие арифметические ошибки;
- доступ за пределы массива и некорректное разыменование указателя;
- выражение, всегда возвращающее истину или ложь;
- неинициализированные члены класса (C++);
- чтение неинициализированных данных;
- доступ по нулевому указателю (C++);
- мертвый (недосягаемый) код;
- неправильное приведение типов;
- динамические ошибки, относящиеся к объектно-ориентированному программированию, наследованию и обработке исключений (C++).

Просмотр информации о диапазонах

Polyspace Code Prover отображает информацию о диапазонах переменных и значениях, возвращаемых функциями, и может доказать условия, при которых переменные нарушают указанные диапазоны. При наведении курсора на оператор или переменную, появляется всплывающая подсказка, отображающая информацию о диапазонах. Метод абстрактной интерпретации позволяет точно определять диапазоны с целью доказательства того, что в программном обеспечении не содержатся ошибки выполнения. Информацию о диапазонах можно использовать для отладки программного обеспечения или чтобы убедиться, что определенные переменные или операции не нарушают заданные диапазоны.

В примере ниже, Polyspace Code Prover определил, что в операции деления левый операнд находится в диапазоне от -1701 до 3276, а правый операнд равен 9. Результат деления находится в диапазоне от -189 до 364.

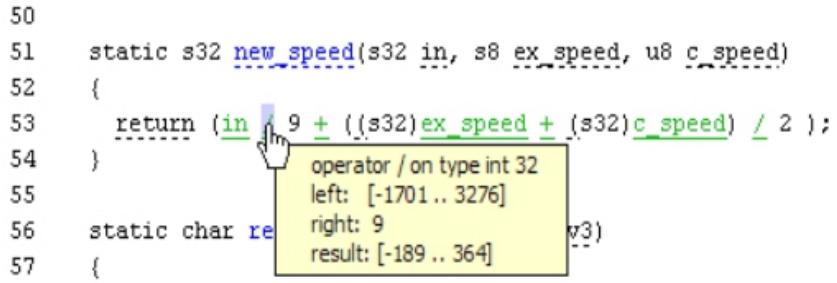


Рис.5 Просмотр информации о диапазоне значений переменной

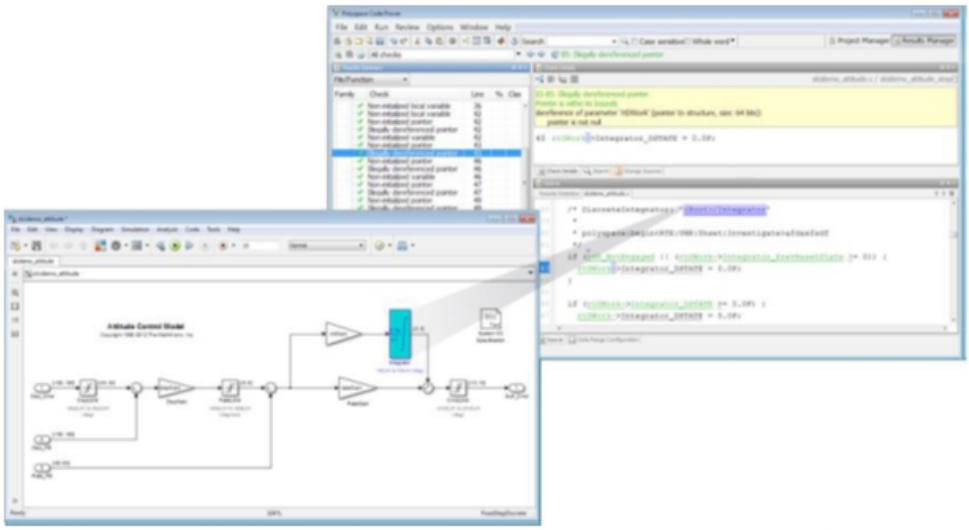


Рис.6 Трассирование результата анализа кода к модели Simulink

Трассирование результатов анализа кода к моделям Simulink

Результаты анализа автоматически сгенерированного кода трассируются обратно к модели Simulink. Таким образом, можно определить, какие части модели являются надежными и

исправить проблемы в модели, которые приводят к дефектам в коде. Также можно идентифицировать потенциальные проблемы между интерфейсами сгенерированного и написанного вручную кода. Например, смешение кода в блоке S-Function, написанного вручную, и сгенерированного кода может потенциально привести к некорректным диапазонам сигналов в интерфейсе и вызвать ошибку выполнения.

Разработка проекта в Polyspace Code Prover и Polyspace Bug Finder

Создание проекта

Основой работы в Polyspace Code Prover и Polyspace Bug Finder является проект. В папках проекта размещаются файлы проверяемого исходного кода, файлы внешних библиотек, записывается конфигурация проекта и результаты проверок. Для создания проекта в меню File нужно выбрать опцию New Project. В открывшемся окне Define project нужно задать имя проекта (Project name), версию проекта (Version), указать язык проекта (Project language). Убрать галочки в настройках Project configuration. Нажать кнопку Next (см. рис. 7).

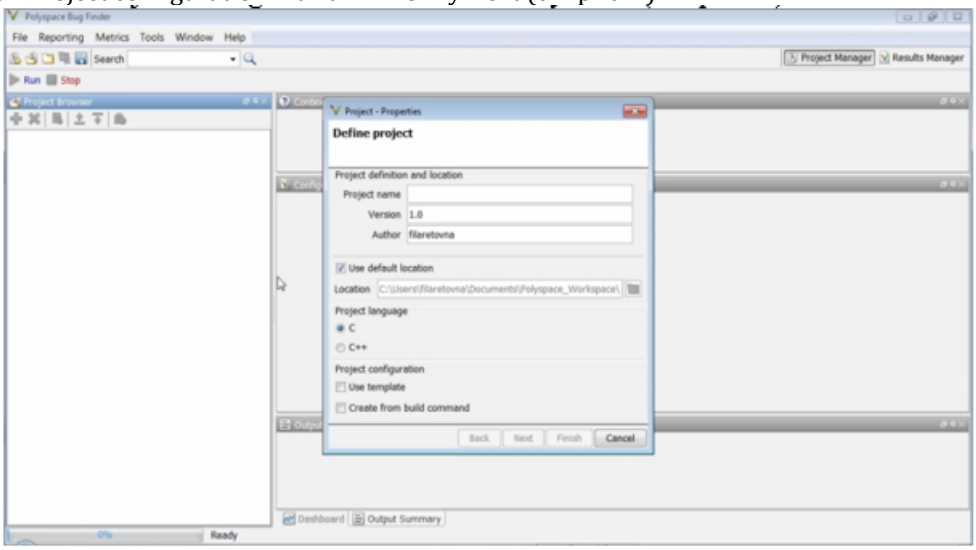


Рис.7 Создание нового проекта в Polyspace Bug Finder

Далее нужно загрузить в папку проекта Source проверяемые файлы C/C++ (кнопка Add Source Files), а в папку Include загрузить подключаемые внешние библиотеки (кнопка Add Include Folder). Нажать кнопку Finish (рис.8). Созданный проект появится в браузере проекта Project Browser.

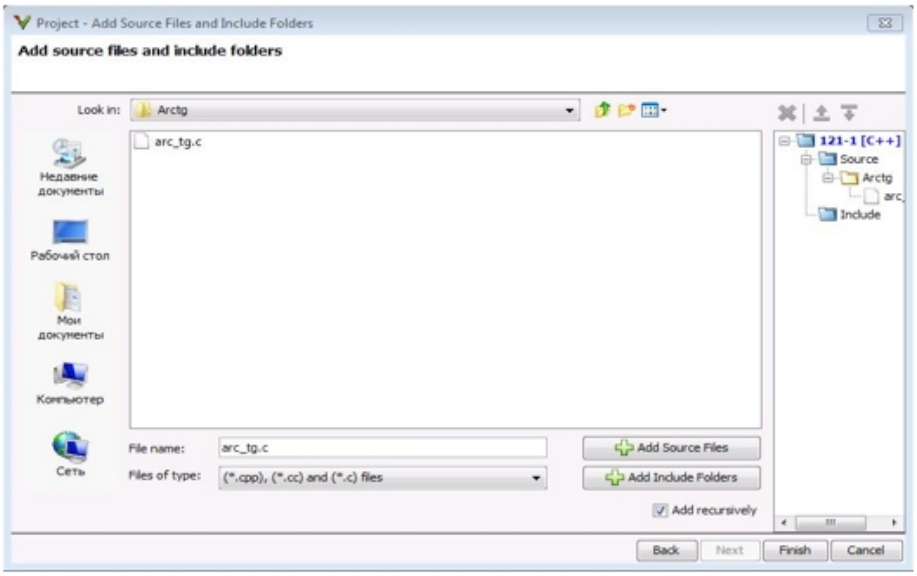


Рис.8 Загрузка проверяемых файлов и папок библиотек в проект

Далее необходимо настроить созданный проект для проверки.

Настройка проекта (конфигурирование проекта)

Для настройки проекта нужно открыть окно Configuration (рис.9).

Во вкладке Target & Compiler можно настроить следующие параметры:

- операционная система (target operating system);
- целевой процессор (target processor type);
- диалект языка (dialect).

Во вкладке Coding rules можно настроить проверку на соответствие стандартам кодирования: выбрать стандарт и настроить подмножество правил кодирования. Например, выбрать стандарт MISRA C: 2012 и проверку на требуемые и рекомендуемые правила кодирования mandatary-recuired.

Во вкладке Reporting можно настроить генерацию отчета в автоматическом режиме по результатам проверки кода.

Во вкладке Advanced Settings можно настроить генерацию метрик кода для контроля характеристик качества кода.

Для запуска анализа кода в созданном проекте нужно нажать в верхнем меню кнопку Run

. Начинается процесс анализа кода, результаты анализа записываются в папку Result проекта.

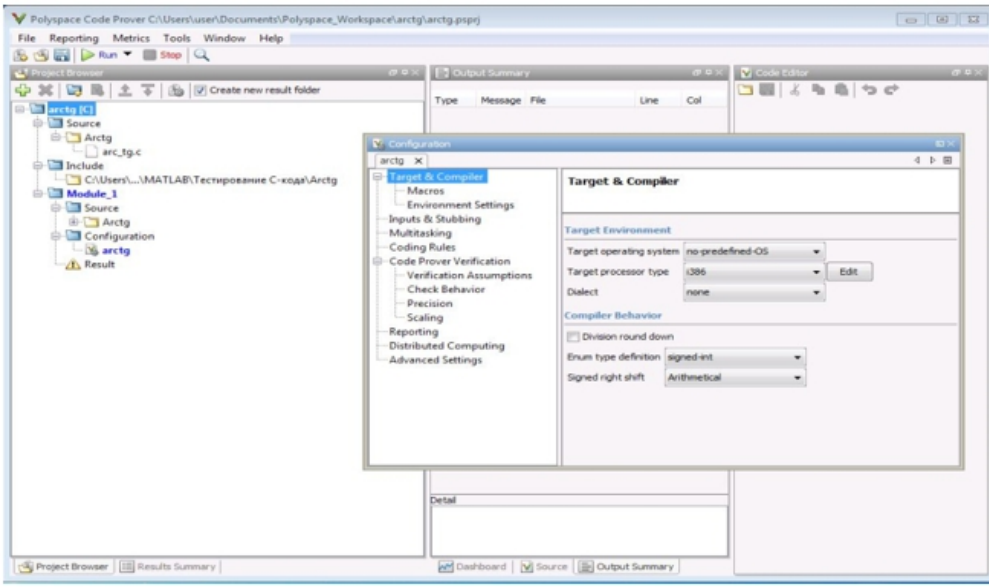
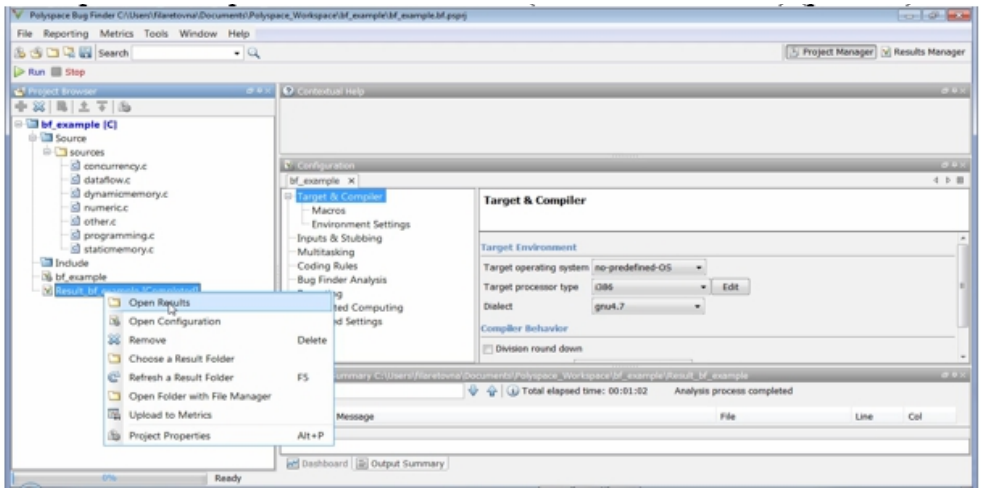


Рис. 9 Настройка проекта

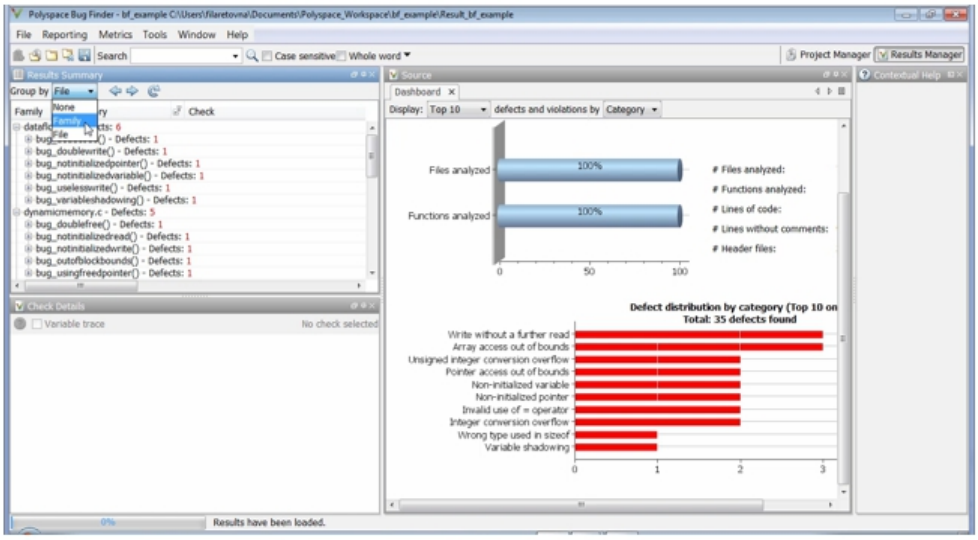
Анализ результатов

Полученные результаты анализа кода можно посмотреть, открыв контекстное меню или дважды кликнув на папку результатов (рис.10а). Результаты отображаются в графическом виде (окно Dashboard) и в виде списка ошибок (окно Results Summary), сгруппированных по типам или по именам файлов, в которых они обнаружены (рис. 10б).

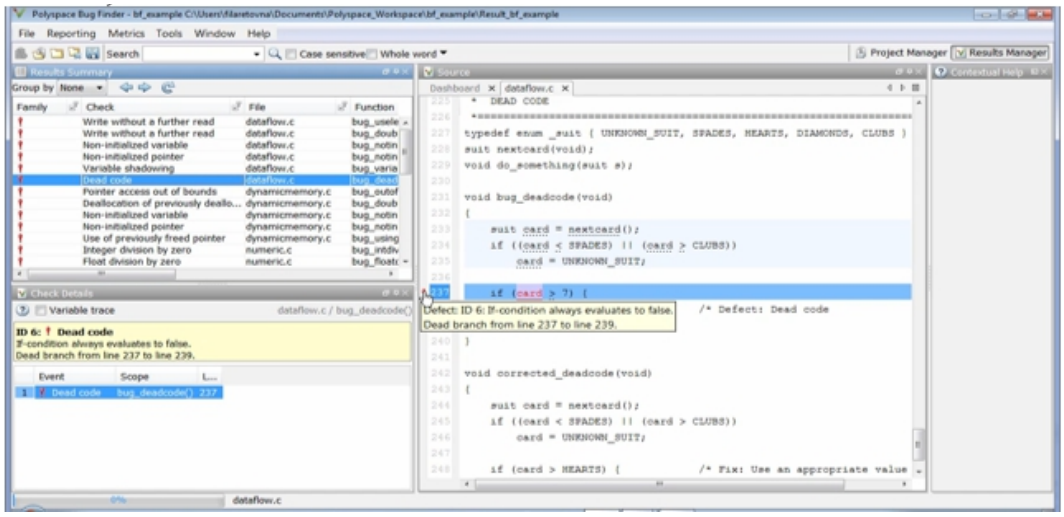
В файле dataflow.c найдена ошибка «мертвый код» (dead code). Кликнув на эту ошибку в окне «итоговый результат» (Results Summary), можно увидеть ее в исходном файле с подробной детализацией (окно Check Details) (рис.10в).



а)



б)



в)

Рис. 10 Просмотр результатов анализа кода в Polyspace Bug Finder

а) открытие файла результатов; б) результаты в виде списка дефектов и в графическом виде; в) ошибка кода «dead code» с отображением в исходном файле (окно Sourses) и детализацией (окно Check Details)

Загрузка и просмотр метрик кода

Измеренные метрики кода, характеризующие его качество, можно размещать в специальном хранилище-репозитории. Такой репозиторий может быть развернут на удаленном сервере и команда разработки ПО может отслеживать на нем изменение качества кода в процессе доработки.

Для загрузки измеренных метрик кода в репозиторий, нужно открыть меню Metrics>Upload to Metrics (рис. 11).

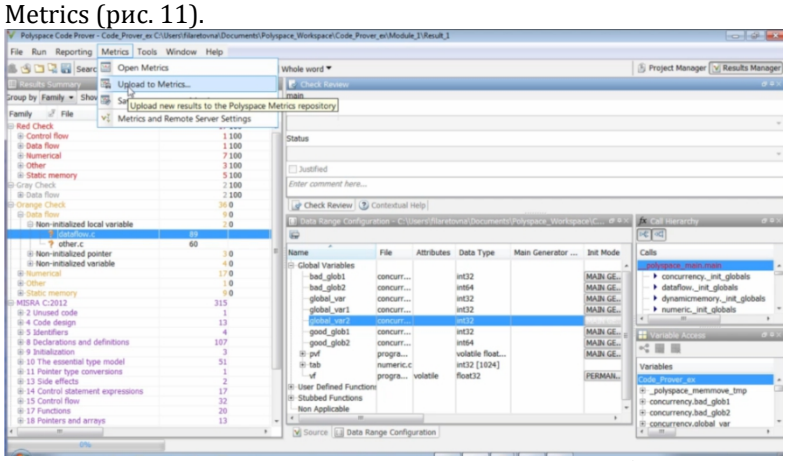


Рис. 11. Загрузка метрик кода в репозиторий

Происходит обнаружение результатов (рис. 12).

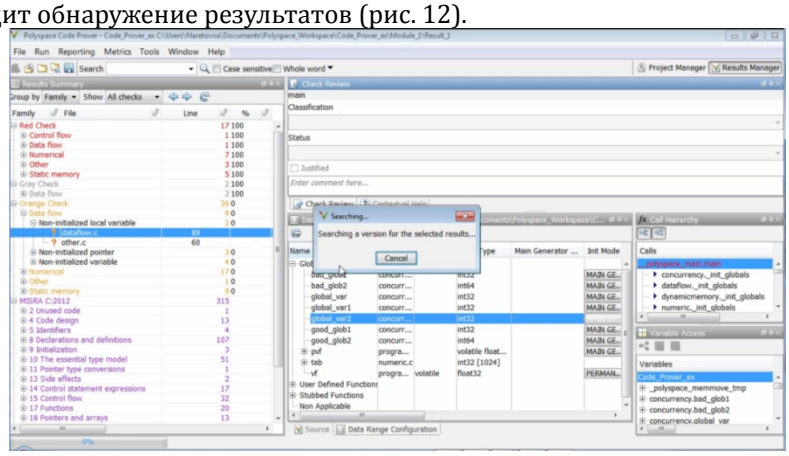


Рис. 12. Обнаружение результатов

В открывшемся окне (рис. 13) нужно задать имя проекта, который будет сохранен в репозитории и его версию. Нажать кнопку Upload.

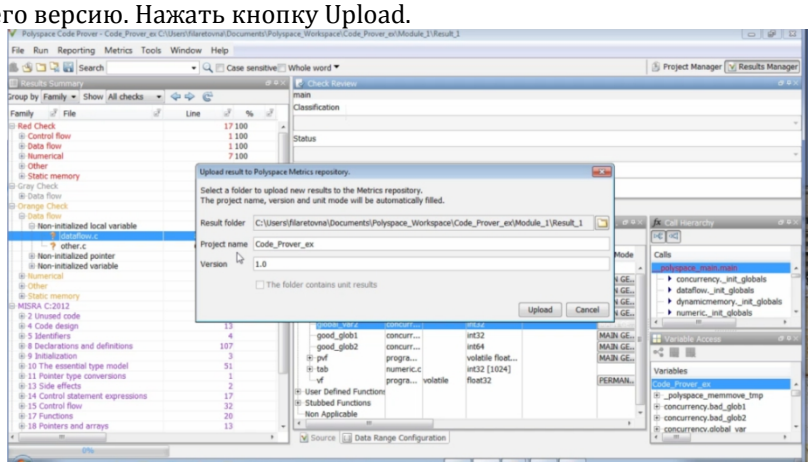


Рис. 13. Создание проекта для загрузки в репозиторий

Доступ к результатам можно защитить с помощью пароля или оставить их открытыми, не вводя пароля (рис. 14).

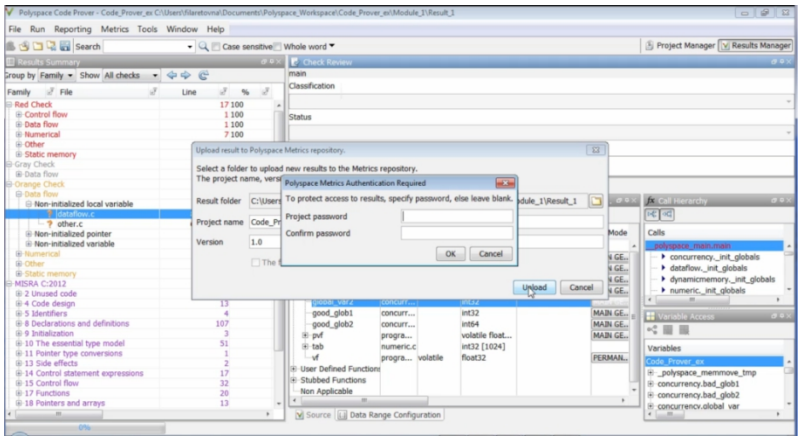


Рис. 14. Защита результатов с помощью пароля

Чтобы открыть результаты, записанные в репозиторий, выбрать в меню Metrics-> Open Polyspace Metrics repository (рис. 15).

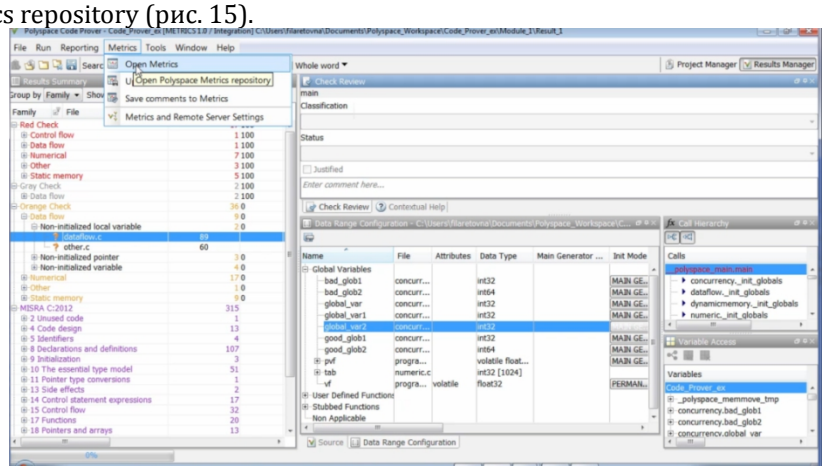


Рис. 15. Открытие репозитория с записанными метриками

Происходит загрузка содержимого репозитория. Появляется проект (рис. 16).

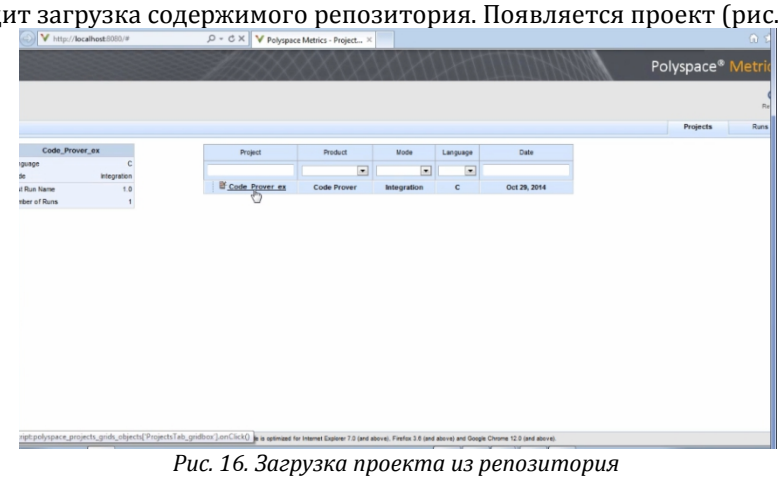


Рис. 16. Загрузка проекта из репозитория

Можно посмотреть различные метрики, например, метрики исходного кода (рис. 17): количество строк кода (Lines of Code), количество исполняемых строк кода (Executable Lines), цикломатическую сложность (Cyclomatic Complexity), плотность комментариев (Comment Density) по каждому файлу кода.

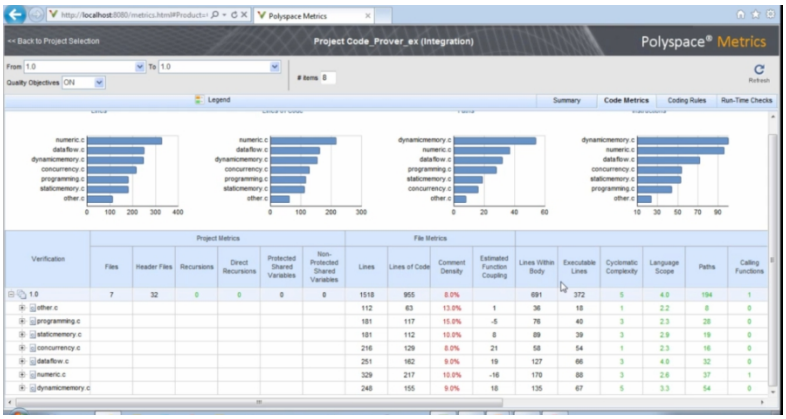


Рис. 17. Просмотр метрик исходного кода

Результаты проверки программного обеспечения для встроенных вычислительных систем

С помощью программ Polyspace Bug Finder и Polyspace Code Prover было проведено тестирование разработанного программного обеспечения, содержащего 2600 строк кода.

Анализ кода на наличие ошибок кодирования (Этап 1)

При проверке Polyspace Bug Finder в коде выявлено наличие «мертвого кода».

Выявлены неинициализированные переменные, «бесполезные» переменные – переменные, не используемые в программе после проведенных модификаций.

Рис. 18. Результаты проверки Polyspace Bug Finder после устранения выявленных дефектов

В результате проведенной работы устранены выявленные дефекты кода (рис. 18):

– доработаны протоколы информационного взаимодействия для баллпрограммы в части учета типа используемого канала (телевизионный или тепловизионный, с широким или узким полем зрения);

– проведены более строгая инициализация и описание переменных;

– устранена избыточная проверка на соответствие диапазону, деления на ноль.

Анализ кода на наличие ошибок выполнения (Этап 2)

На рисунках 19 представлены результаты проверки кода программой Polyspace Code Prover. Проверка показывает отсутствие потенциально опасных ошибок. В коде отмечено наличие условно опасного и «мертвого кода»: возможность неточного преобразования данных из вещественного в целочисленное значение, возможность переполнения (выхода за диапазон, предусмотренный типом переменной).

В соответствии с полученными результатами введено явное описание преобразования

типов, ограничения по диапазону используемых значений, в том числе и за счет изменения типа переменных.

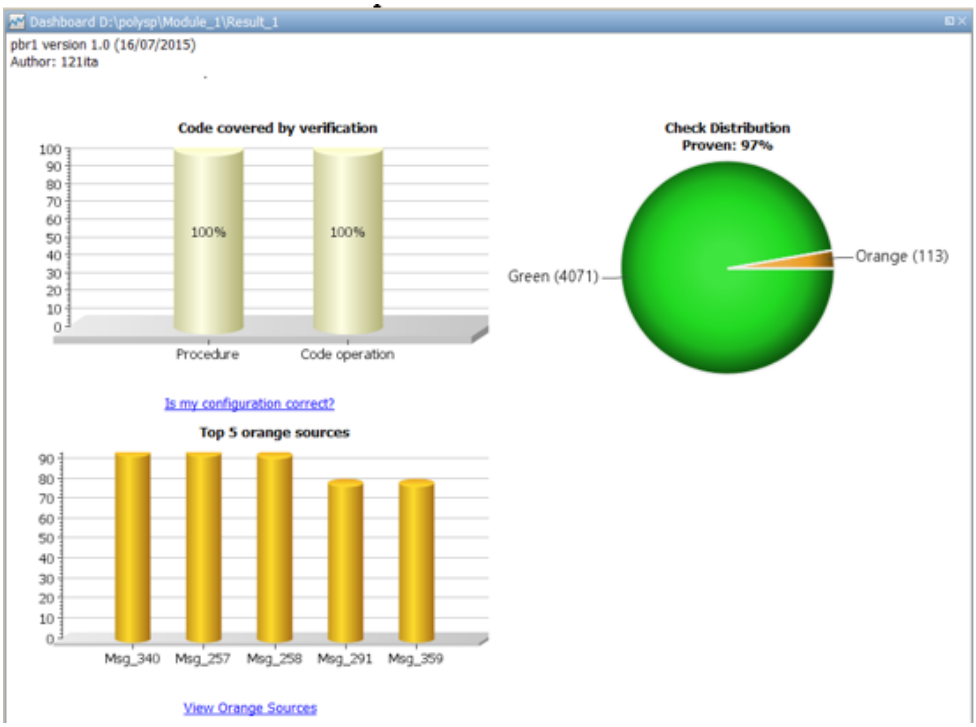


Рис. 19. Результаты проверки исходного кода программой Polyspace Code Prover

Программа Polyspace позволила более корректно сформировать проверки. Например, выявить возможность возникновения ошибок за счет формата представления вещественных чисел в памяти при расчете арифметических выражений с операндами, имеющими существенно различный порядок значений, в частности, избежать получения нуля в знаменателе при ненулевых значениях операндов.

Таким образом, использование ИТ на основе Polyspace Matlab позволяет устранить дефекты и нарушения правил кодирования, сократить время на отладку и проверку при развитии, доработке и модификации программного кода встроенных микропроцессоров для систем управления.

Понятский В.М.[1], Яковлев А.Е.[2], Корзунов О.В. [3], Лужинский А.И. [4], Кислинский И.В.[5]

[1] АО Конструкторское бюро приборостроения им. академика А.Г. Шипунова, г. Тула, начальник бюро, Тульский государственный университет, к.т.н., доцент кафедры Проектирование автоматизированных комплексов, pwmru@rambler.ru

[2] АО Конструкторское бюро приборостроения им. академика А.Г. Шипунова, г. Тула, начальник отдела

[3] АО Конструкторское бюро приборостроения им. академика А.Г. Шипунова, г. Тула, начальник сектора

[4] АО Конструкторское бюро приборостроения им. академика А.Г. Шипунова, г. Тула, инженер, Тульский государственный университет, г. Тула, аспирант кафедры радиоэлектроники, iampvp@yandex.ru

[5] АО Конструкторское бюро приборостроения им. академика А.Г. Шипунова, г. Тула, ведущий инженер, Nick-51-Rus@yandex.ru

ИСПОЛЬЗОВАНИЕ ИТ ПРИ КОМАНДНОЙ ОРГАНИЗАЦИИ РАЗРАБОТКИ МОДЕЛИ СИСТЕМЫ УПРАВЛЕНИЯ

КЛЮЧЕВЫЕ СЛОВА

Системы управления, командная разработка, MATLAB, оптико-электронная система (ОЭС), модель, фильтр Калмана.

АННОТАЦИЯ

Разработка ОЭС является сложным итерационным процессом. Командная работа обеспечивает эффективную разработку сложных многоуровневых систем. К основным принципам командной работы можно отнести организацию работы на верхнем и нижнем уровнях. Для взаимодействия пользователей необходимо использование системы управления версиями, библиотек моделей и данных, технологии «модель-ссылка». Для пользователя обеспечивается возможность организации проектов, векторизации и параметризации моделей, а также комплексный анализ результатов моделирования. Предложенный подход был использован при проектировании режима слежения за малоконтрастными объектами для ОЭС, в котором наблюдаются частые переходы в инерционное сопровождение и раскачки системы сопровождения. Применение методов командной разработки при построении модели ОЭС позволило за короткое время получить эффективные решения за счет возможности проверки вариантов корректировки математического и программного обеспечения на модели ОЭС.

Введение

На сегодняшний день активно развивается направление создания и изучения ОЭС на базе телевизионных пеленгаторов. Разработка ОЭС является сложным итерационным процессом. Актуально использование командной организации работы при разработке ОЭС с применением модельно-ориентированного проектирования.

Командные принципы разработки

При разработке модели ОЭС в среде MATLAB использовались принципы командной разработки модели (рис. 1).

Командная работа обеспечивает эффективность сложных многоуровневых систем. Основные принципы командной разработки разделяются на локальные и сетевые.

Сетевые принципы обеспечивают взаимодействие пользователей и включают в себя создание и повторное использование библиотечных блоков, систему управления версиями, а также технологию создания моделей-ссылок Model Reference.

Рис. 1. Основные принципы командной разработки модели

Создание и повторное использование библиотечных блоков является функциональным узлом, который неоднократно повторяется в рамках одной модели или может быть использован в других моделях. Данный метод позволяет создать библиотечный блок и добавить его в директорию Simulink Library.

Локальные принципы *обеспечивают работу отдельных пользователей и включают в себя методы построения и формирования отдельных подсистем общей модели. Рассмотрим их подробнее на примере использования программного комплекса MATLAB.*

Векторизация используется для упрощения модели, чтобы избежать повтора применения одних и тех же функций для разных источников. Так, сигналы от разных источников объединяются в одну шину данных, которая проходит через один блок функции, вместо того, чтобы этот блок функции был скопирован и вставлен столько раз, сколько есть источников.

Параметризация применяется для упрощения ввода данных при создании сложных подсистем. Различают три вида параметризации:

- маскирование;
- с использованием рабочего пространства модели;
- с использованием скрипта.

Маскирование применяется индивидуально к блоку, в котором напрямую пользователь задаёт или меняет значение параметра. Использование рабочего пространства предоставляет возможность задавать или менять значения параметров в рамках всей модели из одной директории. Использование скрипта позволяет автоматизировать процесс задания или изменения параметров модели до запуска или непосредственно во время моделирования.

Комплексный анализ результатов моделирования подразумевает использование инструмента Simulink Data Inspector. Данный инструмент даёт представление всех сигналов модели в едином интерфейсе и графическом пространстве, позволяет сравнивать сигналы.

Использование Simulink Project предоставляет возможность управлять версиями файлов

проекта (особенно удобно, если разработчики модели работают в разных версиях программы), отслеживать изменения файлов проекта на уровне версионности.

Командная разработка модели ОЭС

Предложенный подход был использован при проектировании режима слежения за малоконтрастными объектами для ОЭС.

Возможность функционирования системы управления в условиях действия помех называется помехоустойчивостью. Для количественной оценки используют отношение сигнал-помеха, при котором погрешность системы не превышает заданного значения с требуемой вероятностью.

Задача повышения помехоустойчивости ОЭС рассматривается как удержание объекта в поле зрения ТА даже в случаях пропадания ошибки слежения от ТА (рис. 2). Для её решения недостаточно только прогнозирования траектории движения объекта, например, с помощью различного вида функций экстраполяции. Необходимо формировать либо сигнал управления *fynp*, либо сигнал ошибки *df*. В случае формирования сигнала управления *fynp* возникают проблемы с изменениями в режимах работы блока управления (БУ). В настоящее время для прогнозирования сигнала ошибки используются усредненные значения на нескольких тактах, предшествующих пропаданию. В этом случае БУ работает постоянно в нормальном режиме, но так как управление ПС осуществляется по его скорости, то при периодических кратковременных пропаданиях сигнала ошибки возникает раскачка ПС, и система теряет устойчивость.

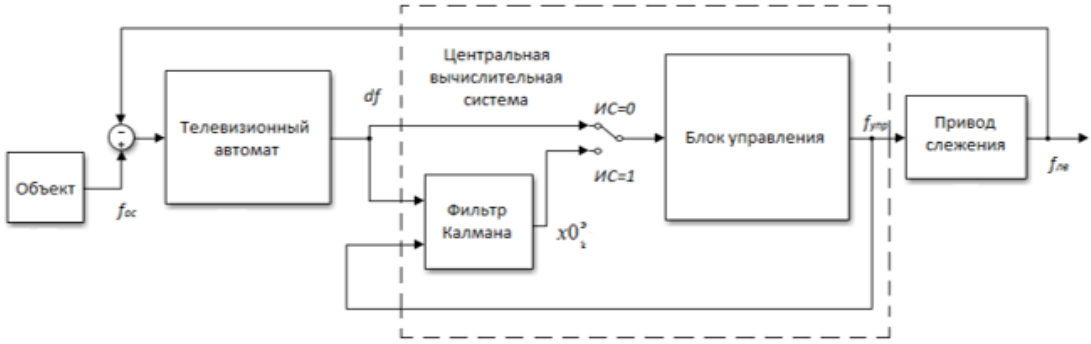


Рис. 2. Функциональная схема системы ОЭС

При сопровождении ОЭС малоконтрастных объектов были выявлены частые переходы в инерционное сопровождение (ИС). Для обеспечения устойчивой работы ОЭС при формировании ошибки слежения за объектом предлагается использовать не только оценки параметров движения объекта (скорость, ускорение и координаты), но и динамическую модель ПС, а также команды управления или сигналы с датчиков углового положения ПС. Для устойчивого управления исполнительным приводом было предложено при пропадании координат объекта слежения использовать фильтрацию Калмана для прогнозирования ошибок слежения [1].

При сопровождении объекта телевизионный автомат (ТА) вырабатывает признак автоматического сопровождения (ИС = 0). Блок управления использует координаты от ТА, при этом фильтр Калмана находится в режиме подслеживания. Как только система переходит в режим инерционного сопровождения (ИС = 1), центральная вычислительная система переключает сигнал с ТА на фильтр Калмана (рис. 2).

Для трехмерной модели движения объекта было проведено проектирование дискретного фильтра Калмана (1) [2 – 7]:

$$
\begin{aligned}
x0_k^э &= x0_{k-1}^о + x1_{k-1}^о \cdot T_o + x2_{k-1}^о \cdot T_o^{\,2}/2 - \mathrm{k_k} \cdot x_k^{c*} \cdot \mathrm{T_0};\\
x1_k^э &= x1_{k-1}^о + x2_{k-1}^о \cdot T_0;\\
x2_k^э &= x2_{k-1}^о;\\
x0_k^о &= x0_k^э + k11_{\mathrm{k}} \cdot (\delta x_k^{c'} - x0_k^э);\\
x1_k^о &= x1_{\text{э}}^э + k12_{\mathrm{k}} \cdot (\delta x_k^{c'} - x0_k^э);\\
x2_k^о &= x2_{\text{э}}^э + k13_{\mathrm{k}} \cdot (\delta x_k^{c'} - x0_k^э),
\end{aligned}
\qquad (1)
$$

где $\delta x_k^{c'}$ – входной сигнал фильтра, $\delta x_k^{c'}$ = *df*; $x0_k^0$ – оценка координат объекта слежения

относительно линии визирования, выходной сигнал фильтра; $x1_k^0$ – оценка скорости движения объекта; $x2_k^0$ – оценка ускорения движения объекта; $\dot{x}_k^{c*}$ – сигнал управления приводом слежения (ПС), $\dot{x}_k^{c*} = f_{упр}$; $k11_k$, $k12_k$, $k13_k$ – коэффициенты фильтра, T_0 – частота работы системы.

Сигналы с надстрочным индексом «*о*» являются оценочными, а сигналы с надстрочным индексом «*э*» – прогнозируемыми на один такт вперед.

При условии отсутствия автоматического сопровождения объекта происходит переход системы в инерционный режим (ИС = 1) и оцениваются координаты, скорости и ускорения объекта слежения:

$$\begin{aligned} x0_k^0 &= x0_{k-1}^0 \cdot k11_k + x1_{k-1}^0 \cdot T_0 + x2_{k-1}^0 \cdot T_0^2/2 - k_k^0 \cdot \dot{x}_{k-1}^{c*} \cdot T_0; \\ x1_k^0 &= x1_{k-1}^0 \cdot k12_k + x2_{k-1}^0 \cdot T_0 \\ x2_k^0 &= x2_{k-1}^0 \cdot k13_k \end{aligned} \qquad (2)$$

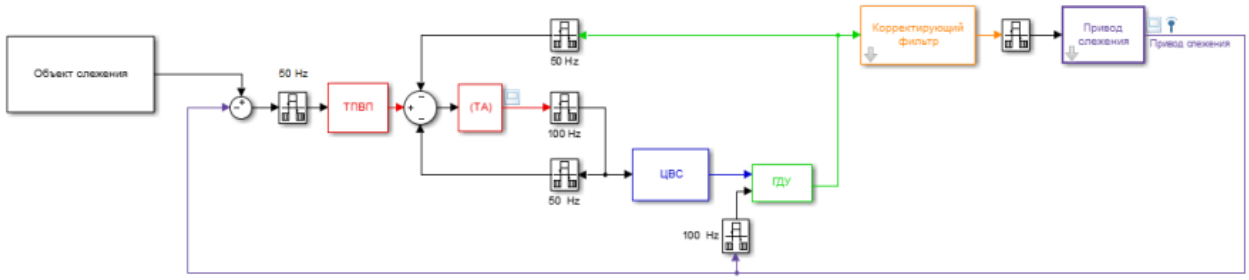
Для корректности использованных алгоритмов фильтрации Калмана и разработки математического и программного обеспечения центральной вычислительной системы (ЦВС) была создана в среде MATLAB модель ОЭС, функциональная схема которой представлена на рис. 2.

Модель ОЭС (рис. 3) включает в себя основные функциональные узлы: подсистему «объект слежения», тепловизионный прибор, ТА, ЦВС, гироскопический датчик угла (ГДУ), корректирующий фильтр (КФ) и привод слежения (ПС).

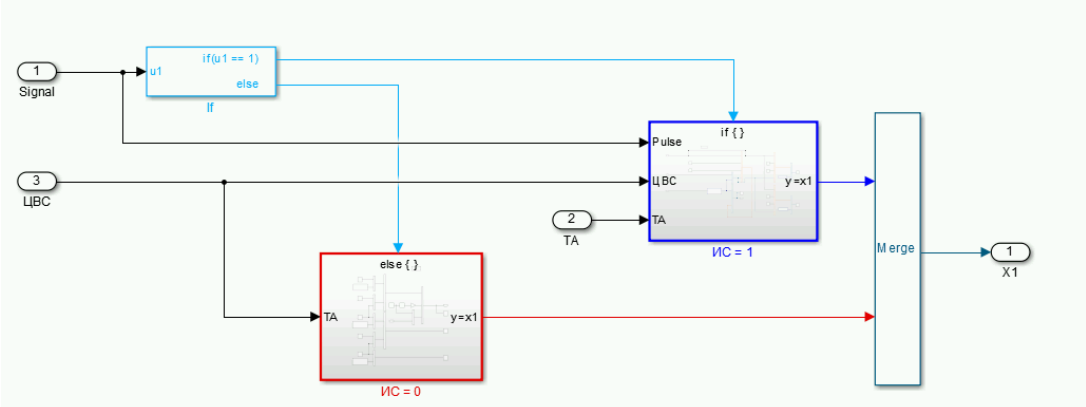
При создании модели ОЭС были использованы локальные и сетевые принципы командной разработки.

Векторизация была использована для объединения двух каналов при моделировании системы ОЭС (горизонтальный X и вертикальный Y) в одну шину данных (X, Y).

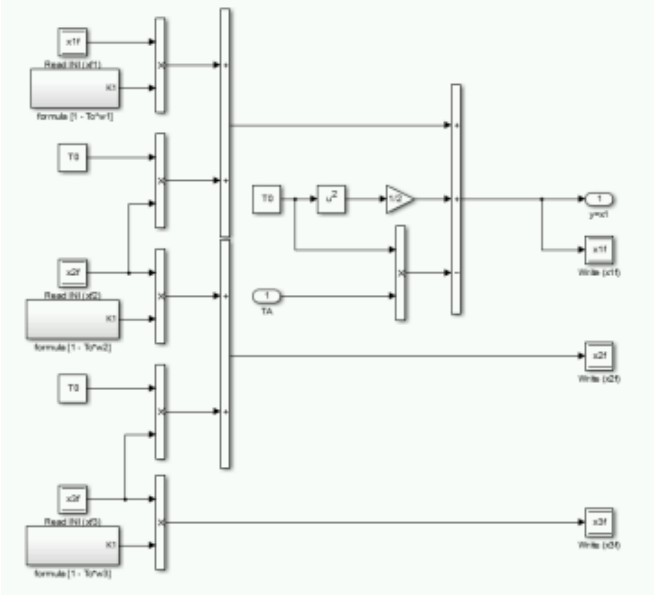
При векторизации в модели ОЭС возникла проблема при формировании передаточных функций с помощью блоков Transfer Fcn: данный блок не может обрабатывать мультиплексный сигнал. Решение этой проблемы возможно при реализации передаточных функций отдельными элементарными блоками, такими как: Gain (коэффициент усиления), Integrator (интегратор), Sum (сумматор) и т.д.



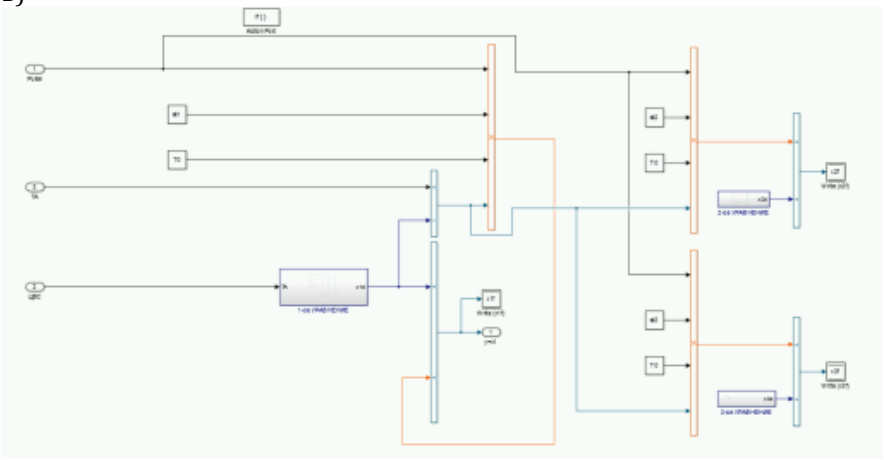
а)



б)



в)



г)

Рис.3. Модель ОЭС в среде MATLAB: а) модель ОЭС; б) модуль Фильтр Калмана в виде модели-ссылки; в) подсистема модуля Фильтр Калмана при наличии АС; г) подсистема модуля Фильтр Калмана при наличии ИС

Параметризация была использована для гибкого изменения параметров подсистем модели: «Корректирующий фильтр» и «Привод слежения».

Учет дискретизации в Simulink-модели начинается с выбора фиксированного шага интегрирования. Также каждую подсистему или отдельный блок можно настроить на моделирование со значением шага интегрирования, который будет отличаться от системного, но не должен быть меньше его.

Технология *Model Reference* была применена при разработке модуля «Фильтр Калмана». Отдельно от основной модели ОЭС была создана модель фильтра Калмана. В модель ОЭС она была добавлена как ссылка при помощи блока Model Reference. Преимущество этой технологии в том, что она позволяет разным группам разработчиков независимо разрабатывать отдельные части системы, которые потом можно собрать в единую модель.

При разработке модели важную роль играет система управления версиями (CVS). CVS обеспечивает удобный для управления процессом разработки модели программный интерфейс.

Функция Simulink Variant Subsystem позволяет в одной модели иметь несколько реализаций отдельных функциональных блоков и выбирать ту реализацию, которая необходима в данный момент, делая её активной.

Разработанная модель «Фильтр Калмана» обеспечивает генерацию С-кода. Процесс генерации универсального С-кода включает: перевод модели для расчёта с фиксированным

шагом; выбор дискретного решателя; выбор конечного языка С или С++; применение команды генерации кода для блока «Build this block» или для всей модели «Build model».

Результатом генерации универсального С-кода является файл с расширением *.c или *.cpp в зависимости от выбранного языка. В данном случае, была выбрана генерация для С-кода и на выходе получен файл Kalman.c. На рис. 4 показан фрагмент кода файла, полученного из модели-ссылки двухканального Фильтра Калмана с векторным представлением сигналов, реализующего уравнения (1) и (2).

Анализ результатов моделирования проводился при помощи инструмента Simulink Data Inspector. Необходимые сигналы фиксируются флагами на соединительных линиях для отображения в едином интерфейсе компонента Simulink Data Inspector.

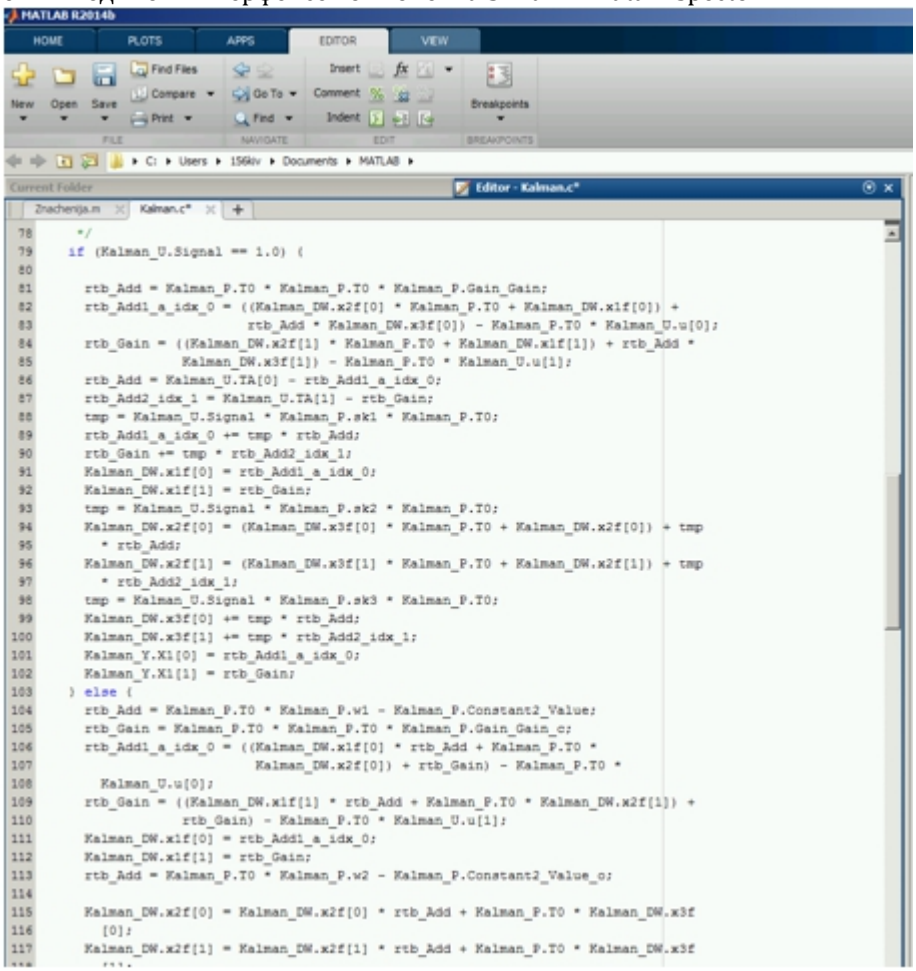


Рис.4. Фрагмент с-кода модуля Фильтр Калмана

Результаты экспериментальных исследований

Были проведены эксперименты с ОЭС по сопровождению малоконтрастного подвижного объекта (рис. 5). В ходе испытаний установлено, что при сопровождении ОЭС с разработанным фильтром Калмана (1) – (2) малоконтрастного объекта, уровень ошибки сохранился, как и в режиме АС объекта, а переходы системы в инерционный режим, характеризуемые значительными отклонениями линии визирования, отсутствуют.

Экспериментальные исследования ОЭС с разработанным фильтром Калмана подтвердили повышение помехоустойчивости сопровождения малоконтрастного объекта [8].

Применение ИТ командной разработки при построении модели ОЭС позволило за короткое время получить эффективные решения за счет возможности проверки вариантов корректировки математического и программного обеспечения на модели ОЭС и уменьшить затраты при использовании техники на натурных экспериментах.

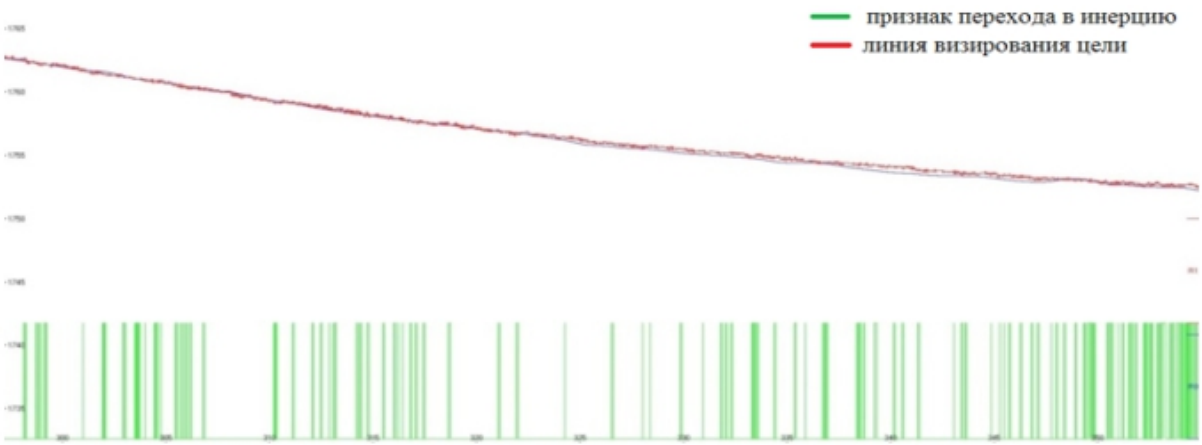


Рис. 5. Экспериментальные результаты сопровождения ОЭС с фильтром Камана малоконтрастного подвижного объекта

Литература

1. Ярлыков М.С. Статистическая теория радионавигации. – М.: Радио и связь, 1985. – 344 с.
2. Сейдж Э.П., Мелса Дж. Л. Теория оценивания и её применение в связи и управлении. – М.: Связь,1976. – 496 с.
3. Понятский В.М., Петрушин В.В. Повышение динамической точности и помехоустойчивости в системах телеуправления // Известия ТулГУ. Сер. «Проблемы проектирования и производства систем и комплексов». – Тула: ТулГУ. Вып. 6, 2003. – С. 333 – 335.
4. Понятский В.М. Проектирование фильтра Калмана для системы управления летательным аппаратом // Юбилейная научно-техническая конференция «Авиационные системы в XXI веке» (11 – 13 апреля 2006 г.) – М.: ГосНИИАС, 2 том, 2006. – С. 172 – 176.
5. Понятский В.М. Исследование способов реализации адаптивной системы управления с фильтром Калмана // Стохастическая оптимизация в информатике. Межвузовский сборник / Под ред. О.Н. Граничина – СПб.: Изд-во С.-Петербургского университета. Вып. 4, 2008. – С. 186 – 200.
6. Понятский В.М. Способ повышения помехоустойчивости робототехнической системы // Труды XII международного семинара «Супервычисления и математическое моделирование» (11 октября – 15 октября 2010 г.) – Саров: ФГУП «РФЯЦ-ВНИИЭФ», 2011. – С. 288 – 300.
7. Понятский В.М. Способы повышения устойчивости оптико-электронной системы при использовании фильтрации Калмана // Стохастическая оптимизация в информатике. Межвузовский сборник. / Под ред. О.Н. Граничина. – СПб.: Издательство С.-Петербургского университета. Вып. 4, 2014. – С. 186 – 200.
8. Кислинский И.В., Корзунов О.В., Лужинский А.И., Понятский В.М., Яковлев А.Е. Исследование оптико-электронной системы с фильтром Калмана при сопровождении малоконтрастных объектов // Сб. докл. X Всерос. юбилейной науч.-техн. конф. «Проблемы совершенствования робототехнических и интеллектуальных систем летательных аппаратов», Москва, Мос. авиац. ин-т, 26 июня 2015 г. / Редкол.: К.М. Тихонов (пред.). – М.: Изд-во МАИ, 2015. – С. 75 – 83.

Смирнова М.А.[1], Смирнов М.Н.[2]

[1] Санкт-Петербургский государственный университет, г.Санкт-Петербург, ассистент кафедры компьютерных технологий и систем, smirnova-ma@bk.ru

[2] Санкт-Петербургский государственный университет, г.Санкт-Петербург, ассистент кафедры компьютерных технологий и систем, smirnov-mn@mail.ru

АСТАТИЧЕСКАЯ КОРРЕКЦИЯ ЦИФРОВЫХ ЗАКОНОВ УПРАВЛЕНИЯ

КЛЮЧЕВЫЕ СЛОВА

Астатизм, цифровое управление, устойчивость.

АННОТАЦИЯ

В статье рассмотрены задачи применения многоцелевых структур для управления линейными объектами в их движении по заданным траекториям с обеспечением нулевой статической ошибки при воздействии постоянных внешних возмущений, а также вопросы астатической коррекции стабилизирующих законов управления. Предложены методы и алгоритмы для решения указанных задач.

Современные системы автоматического управления движением, как правило, функционируют в различных динамических режимах, определяемых конкретным заданием командных сигналов и внешних возмущений, действующих на объект управления. Для каждого из режимов на стадии проектирования системы формируется комплекс требований, условий и ограничений, которые должны неукоснительно выполняться в процессе движения. Следует особо отметить, что указанные комплексы в совокупности чаще всего имеют противоречивый характер в силу существенного различия особенностей динамики режимов движения.

Естественным путем обеспечения всех требуемых динамических свойств является достижение некоторого компромисса по качеству процессов управления в различных режимах. Этот компромисс проще всего обеспечить некоторым единым многоцелевым законом управления для всех режимов, однако потери качества для отдельных режимов здесь очевидны. Этот подход рассмотрен в различных работах, таких как [1 – 9].

В связи с отмеченными обстоятельствами, в работе принят иной подход к формированию многоцелевых законов, базирующийся на частичной фиксации некоторой единой для всех режимов части закона управления с возможным подключением дополнительных адаптивно настраиваемых на отдельные режимы элементов. Математической основой для такой настройки принимается оптимизационный подход, позволяющий трактовать содержательные задачи проектирования как задачи о поиске экстремумов. Идеология предложенного подхода рассматривалась в различных работах [10 – 31], однако в цифровом случае требуются дополнительные уточнения.

В частности, существенное внимание в статье уделяется законам управления движением по заданной траектории, обеспечивающих астатизм замкнутой системы.

Пусть математическая модель динамики подвижного объекта представляется следующей системой дифференциальных уравнений

$$\begin{aligned} \mathbf{x}[n+1] &= \mathbf{F}_x\left(\mathbf{x}[n], \delta[n]\right) + \mathbf{B}\mathbf{d}[n], \\ \delta[n+1] &= \mathbf{F}_\delta\left(\delta[n], \mathbf{u}[n]\right), \\ \mathbf{y}[n] &= \mathbf{C}\mathbf{x}[n], \end{aligned} \tag{1}$$

где функции $\mathbf{F}_x$ и $\mathbf{F}_\delta$ определяют нелинейности объекта и привода соответственно.

Здесь $\mathbf{x} \in E^n$ – вектор состояния, $\delta \in E^m$ – вектор управляющих воздействий, $\mathbf{u} \in E^m$ – вектор управляющих сигналов (управлений), $\mathbf{y} \in E^k$ – вектор измеряемых и регулируемых переменных, $\mathbf{d} \in E^l$ – вектор внешних воздействий, матрицы $\mathbf{B}$ и $\mathbf{C}$ имеют постоянные компоненты.

Также будем рассматривать уравнение обратной связи

$$\begin{aligned}\mathbf{z}[n+1]&=\mathbf{F}_z\left(\mathbf{z}[n],\delta[n],\mathbf{y}[n]\right),\\ \mathbf{u}[n]&=\mathbf{F}_u\left(\mathbf{z}[n],\delta[n],\mathbf{y}[n]\right)\end{aligned} \qquad (2)$$

с вектором состояния $\mathbf{z}\in E^{\nu}$. Функции $\mathbf{F}_z$ и $\mathbf{F}_u$ необходимо определить в результате решения задачи синтеза.

Для замкнутой системы (1), (2), находящейся под воздействием внешнего возмущения вида $\mathbf{d}=\{\mathbf{d}[n]\}=\mathbf{d}_0\cdot\{d_e[n]\}$, где $\mathbf{d}_0\in E^l$, а $d_e=\{d_e[n]\}$ – ступенчатая единичная последовательность, условие астатизма по регулируемой переменной $\mathbf{y}$ запишется в виде $\lim\limits_{n\to\infty}\mathbf{y}[n]=0$.

Для решения задачи стабилизации, уравнения состояния объекта (1) линеаризуются в окрестности нуля при нулевых управляющих и возмущающих воздействиях, а уравнения приводов рассматриваются только в пределах линейного участка. В результате формируется линейная модель динамики подвижного объекта

$$\begin{aligned}\mathbf{x}[n+1]&=\mathbf{A}\mathbf{x}[n]+\mathbf{B}\,\delta[n]+\mathbf{D}\mathbf{d}[n],\\ \delta[n+1]&=\mathbf{u}[n]+\delta[n],\\ \mathbf{y}[n]&=\mathbf{C}\mathbf{x}[n]\end{aligned} \qquad (3)$$

с теми же динамическими переменными.

Задача состоит в том, чтобы спроектировать обратную связь (2) так, чтобы указанное положение равновесия было асимптотически устойчивым, а замкнутая система (3), (2) была астатической по регулируемой переменной $\mathbf{y}$.

Скоростной регулятор в дискретном варианте определяется формулой

$$\mathbf{u}[n]=\mu\left(\mathbf{x}[n+1]-\mathbf{x}[n]\right)+\nu\mathbf{y}[n]. \qquad (4)$$

Он строится однозначно в силу уравнений динамики объекта на основе исходного управления по состоянию

$$\mathbf{u}[n]=\mathbf{K}_x\mathbf{x}[n]+\mathbf{K}_\delta\delta[n]\,. \qquad (5)$$

При этом коэффициенты управлений (5) и (6) определяются с обязательным учетом следующих требований:

– замкнутая линейная система должна быть устойчивой;

– перерегулирование P и длительность переходного процесса T_p не должны превосходить заданных величин, т.е. $P\le P_0$, $T_p\le T_{p0}$.

Суть метода обеспечения астатизма для режима стабилизации в дискретном случае состоит в численном поиске коэффициентов исходного базового закона управления (5), обеспечивающего выполнение указанных требований с переходом к скоростному закону управления (4) в силу уравнений объекта. Поскольку производные вектора состояния недоступны непосредственному измерению, они заменяются оценками, полученными с помощью асимптотического наблюдателя полного порядка.

Для поиска коэффициентов базового закона (5) удобно использовать функционал

$$J=\sum_{n=0}^{\infty}\left(\mathbf{x}'[n]\mathbf{Q}\mathbf{x}[n]+\mathbf{u}'[n]\mathbf{R}\mathbf{u}[n]\right), \qquad (6)$$

заданный на движениях замкнутой системы (3), (5). Минимизация этого функционала позволяет найти матрицы $\mathbf{K}_x,\mathbf{K}_\delta$ базового стабилизирующего регулятора (5). Заметим, что весовые матрицы $\mathbf{Q}$ и $\mathbf{R}$ заранее не задаются и находятся адаптивно при реализации общей схемы синтеза управления (4).

Таким образом, можно сформировать схему синтеза цифрового скоростного закона управления. Эта схема, базирующаяся на конечномерной задаче на условный экстремум, состоит из следующих вычислительных операций:

1. Указывается вектор $\gamma\in E_p$ вещественных числовых параметров, от которых однозначно зависит знакоположительная матрица $\mathbf{Q}=\mathbf{Q}(\gamma)$ и положительно-определенная матрица $\mathbf{R}=\mathbf{R}(\gamma)$ и задаются начальные приближения для его компонентов.

2. Решается задача LQR-оптимального синтеза для замкнутой системы с интегральным квадратичным функционалом

$$J=\sum_{n=0}^{\infty}\left(\mathbf{x}'[n]\mathbf{Q}\mathbf{x}[n]+\mathbf{u}'[n]\mathbf{R}\mathbf{u}[n]\right),,$$

что дает коэффициенты $\mathbf{K}_x=\mathbf{K}_x(\gamma)$, $\mathbf{K}_\delta=\mathbf{K}_\delta(\gamma)$ базового стабилизирующего регулятора.

3. Базовый регулятор преобразуется в силу линейных уравнений объекта к скоростной форме (4) при условии отсутствия внешнего воздействия, что дает коэффициенты $\mu=\mu(\gamma)$, $\nu=\nu(\gamma)$.

4. На соответствующих движениях получившейся замкнутой системы при учете ступенчатых возмущений определяются значения функционалов $P=P(\gamma)$ и $T_p=T_p(\gamma)$, а также значение вспомогательного функционала

$$I=I(\gamma)=P(\gamma)-P_0+|P(\gamma)-P_0|+T_p(\gamma)-T_{p0}+|T_p(\gamma)-T_{p0}|.$$

5. Если для данного вектора γ значение вспомогательного функционала положительно, с помощью любого численного метода спуска задать новое приближение вектора γ и повторить вычисления по пунктам 2 – 5, минимизируя функционал $I=I(\gamma)$ до достижения им нулевого глобального экстремума, соответствующего выполнению желаемых ограничений.

Особенности реализации цифрового траекторного управления с использованием многоцелевой структуры рассмотрим на примере линейного стационарного объекта с математической моделью

$$\begin{aligned}\xi[n+1]&=\mathbf{A}\xi[n]+\mathbf{B}\mathbf{u}[n],\ \ \xi[0]=0,\\ \mathbf{y}[n]&=\mathbf{C}\xi[n]+\mathbf{D}\mathbf{u}[n],\end{aligned}\tag{7}$$

где $\xi\in\mathrm{E}^{\nu}$ – вектор состояния объекта, $\mathbf{u}\in\mathrm{E}^{\mu}$ – вектор управляющих воздействий, $\mathbf{y}\in\mathrm{E}^{k}$ – вектор регулируемых координат, **A,B,C,D** – матрицы соответствующих размерностей с постоянными компонентами.

Будем считать, что заданное движение $\mathbf{y}_d[n]$ реализуемо, если существует обратная связь, обеспечивающая выполнение

$$\mathbf{y}[n]\to\mathbf{y}_d[n]\ \text{ при } n\to\infty$$

в замкнутой системе.

Уравнения (7) определяют линейный стационарный оператор

$$\Re_p: U\to Y\ ,\ \mathbf{y}[n]=\Re_p\mathbf{u}[n]\ ,\tag{8}$$

который при заданных начальных условиях $\xi[0]=\xi_0$ ставит каждому управлению $\mathbf{u}$ из допустимого множества U в однозначное соответствие выход $\mathbf{y}$ из множества Y. Будем считать, что определен и соответствующий обратный оператор $\Re_{\mathrm{p}}^{-1}$.

Пусть при этом задана цифровая стабилизирующая обратная связь с LTI математической моделью

$$\begin{aligned}\zeta[n+1]&=\mathbf{A}_c\zeta[n]+\mathbf{B}_c\mathbf{y}[n],\\ \mathbf{u}[n]&=\mathbf{C}_c\zeta[n]+\mathbf{D}_c\mathbf{y}[n],\end{aligned}\tag{9}$$

где $\zeta\in\mathrm{E}^{\nu_1}$ – вектор состояния регулятора, $\mathbf{A}_c$, $\mathbf{B}_c$, $\mathbf{C}_c$, $\mathbf{D}_c$ – матрицы соответствующих размерностей с постоянными компонентами. Как и в непрерывном времени, начальные условия по вектору ζ принимаются нулевыми.

Уравнениям (9) регулятора соответствует линейный стационарный оператор $\Re_c: Y\to U$ обратной связи

$$\mathbf{u}[n]=\Re_c\mathbf{y}[n]\ ,\tag{10}$$

который ставит каждому измерению $\mathbf{y}$ из множества Y в однозначное соответствие управление $\mathbf{u}$ из множества U.

При замыкании объекта (7) обратной связью (9) в соответствии с соотношениями (8) и (10) имеем

$$\mathbf{y}[n]=\Re_p\Re_c\mathbf{y}[n]\ ,\tag{11}$$

т.е. уравнение, решение которого приводит к линейному стационарному оператору $\Re_3$ замкнутой однородной системы:

$$\Re_3 \mathbf{y}[n]=0\ . \tag{12}$$

Поскольку обратная связь является стабилизирующей, нулевое положение равновесия системы (12) асимптотически устойчиво по Ляпунову, т.е. справедливо условие

$$\mathbf{y}[n]\to\mathbf{0}\ \text{ при } n\to\infty\,. \tag{13}$$

Вместо обратной связи (10) сформируем управляющее воздействие в виде суммы

$$\mathbf{u}[n]=\Re_p^{-1}\mathbf{y}_{\boldsymbol{d}}[n]+\Re_c\left(\mathbf{y}[n]-\mathbf{y}_{\boldsymbol{d}}[n]\right)\,, \tag{14}$$

где первое слагаемое можно трактовать, как задающий командный сигнал $\mathbf{u}^{*}[n]=\Re_p^{-1}\mathbf{y}_d[n]$, подаваемый на замкнутую систему, а второе слагаемое $\widetilde{\mathbf{u}}[n]=\Re_c\left(\mathbf{y}[n]-\mathbf{y}_d[n]\right)$ определяет обратную связь по ошибке $\mathbf{e}[n]=\mathbf{y}[n]-\mathbf{y}_d[n]$ слежения.

Тогда справедливо следующее утверждение:

Управление $\mathbf{u}[n]=\Re_p^{-1}\mathbf{u}_d[n]+\Re_c\left(\mathbf{y}[n]-\mathbf{y}_d[n]\right)$, *где первое слагаемое – задающий командный сигнал, а второе – обратная связь по ошибке слежения* $\mathbf{e}[n]=\mathbf{y}[n]-\mathbf{y}_d[n]$, *обеспечивает реализацию заданного движения* $\mathbf{y}_d[n]$ *для объекта*

$$\xi[n+1]=\mathbf{A}\xi[n]+\mathbf{B}\mathbf{u}[n],$$
$$\mathbf{y}[n]=\mathbf{C}\xi[n]+\mathbf{D}\mathbf{u}[n],$$

т.е. выполнение условия $\mathbf{y}[n]\to\mathbf{y}_d[n]$ *при* $n\to\infty$.

Конкретизируем приведенную схему реализации желаемого движения по заданному направлению с использованием цифрового стабилизирующего регулятора по состоянию объекта.

Рассмотрим линейную цифровую математическую модель подвижного объекта с учетом уравнения привода, работающего в пределах линейного участка:

$$\begin{aligned}\mathbf{x}[n+1]&=\mathbf{A}\mathbf{x}[n]+\mathbf{B}\,\delta[n],\\ \delta[n+1]&=\mathbf{u}[n]+\delta[n],\\ \mathbf{y}[n]&=\mathbf{C}\mathbf{x}[n].\end{aligned} \tag{15}$$

Здесь $x[n]\in E^n$, $\delta[n]\in E^m$, $\mathbf{u}[n]\in E^m$, $\mathbf{y}[n]\in E^k$, $k<n$, причем будем рассматривать часто встречающуюся ситуацию, когда $\mathbf{C}=\left(\mathbf{0}\ \vdots\ \mathbf{C}_2\right)$, где $\mathbf{C_2}$ – не особая квадратная матрица размера $n\times n$, т.е. справедливо равенство

$$\mathbf{y}[n]=\mathbf{C}\mathbf{x}[n]=\left(\mathbf{0}\ \vdots\ \mathbf{C}_2\right)\begin{pmatrix}\mathbf{x}_1[n]\\ \text{- - - -}\\ \mathbf{x}_2[n]\end{pmatrix}=\mathbf{C}_2\mathbf{x}_2[n]\,,\ \mathbf{x}_2[n]\in E^k\ ,\ \det\mathbf{C}_2\neq 0.\,. \tag{16}$$

Введем также в рассмотрение уравнение стабилизирующего регулятора по состоянию

$$\mathbf{u}[n]=\mathbf{K}_x\mathbf{x}[n]+\mathbf{K}_\delta\delta[n], \tag{17}$$

которое с учетом представления $\mathbf{K}_x=\left(\mathbf{K}_{x1}\ \vdots\ \mathbf{K}_{x2}\right)$ можно записать в виде

$$\mathbf{u}[n]=\mathbf{K}_{x1}\mathbf{x}_1[n]+\mathbf{K}_{x2}\mathbf{x}_2[n]+\mathbf{K}_\delta\delta[n]\ .$$

В соответствии с приведенным утверждением, справедливо следующее:

Управляющий сигнал

$$\mathbf{u}[n]=\mathbf{H}^{-1}(q)\,\mathbf{y}_d[n]+\mathbf{K}_{x1}\mathbf{x}_1[n]+\mathbf{K}_{x2}\mathbf{C}_2^{-1}\left[\mathbf{y}[n]-\mathbf{y}_d[n]\right]+\mathbf{K}_\delta\delta[n],$$

обеспечивает реализацию заданного движения $\mathbf{y}_d[n]$ *для объекта с вектором состояния* $\mathbf{x}[n]=(\mathbf{x}'_1[n]\ \vdots\ \mathbf{x}'_2[n])'$

$$\begin{aligned}\mathbf{x}[n+1]&=\mathbf{A}\mathbf{x}[n]+\mathbf{B}\,\delta[n],\\ \delta[n+1]&=\mathbf{u}[n]+\delta[n],\\ \mathbf{y}[n]&=\mathbf{C}\mathbf{x}[n]=\mathbf{C}_2\mathbf{x}_2[n],\end{aligned}$$

т.е. выполнение условия $\mathbf{y}[n]\to\mathbf{y}_d[n]$ *при* $n\to\infty$.

Приведенные утверждения позволяют сформировать правило трансформации заданного стабилизирующего управления для его использования с целью реализации желаемого движения

объекта:

$$\begin{aligned}&\mathbf{u}[n]=\mathbf{K}_x\mathbf{x}[n]+\mathbf{K}_\delta\delta[n]=\mathbf{K}_{x1}\mathbf{x}_1[n]+\mathbf{K}_{x2}\mathbf{x}_2[n]+\mathbf{K}_\delta\delta[n]\Rightarrow\\&\mathbf{u}[n]=\mathbf{H}^{-1}(q)\mathbf{y}_d[n]+\mathbf{K}_{x1}\mathbf{x}_1[n]+\mathbf{K}_{x2}\mathbf{C}_2^{-1}\left[\mathbf{y}[n]-\mathbf{y}_{\boldsymbol{d}}[n]\right]+\mathbf{K}_\delta\delta[n].\end{aligned}\qquad(18)$$

Здесь первое слагаемое $\mathbf{u}^*[n]=\mathbf{H}^{-1}(q)\,\mathbf{y}_d[n]$ можно трактовать, как задающий командный сигнал, который напрямую подается на объект в сумме с обратной связью.

При этом второе слагаемое

$$\widetilde{\mathbf{u}}[n]=\mathbf{K}_{x1}\mathbf{x}_1[n]+\mathbf{K}_{x2}\mathbf{C}_2^{-1}\left[\mathbf{y}[n]-\mathbf{y}_d[n]\right]+\mathbf{K}_\delta\delta[n]$$

определяет обратную связь с учетом ошибки $\mathbf{e}[n]=\mathbf{y}[n]-\mathbf{y}_d[n]$ слежения.

Теперь обратимся к ситуации, когда реализация желаемого движения объекта обеспечивается не с помощью регулятора (17) по состоянию, а с помощью стабилизирующей обратной связи по измеряемому выходу $\mathbf{y}$.

Будем использовать многоцелевую структуру закона управления, представленную выше, обеспечивая ее динамическую коррекцию для достижения астатизма по отработке программного движения при наличии ступенчатых возмущений.

Рассмотрим систему с постоянным внешним воздействием $\mathbf{d}=\left\{\mathbf{d}[n]\right\}=\mathbf{d}_0\cdot\left\{d_e[n]\right\}$

$$\begin{aligned}&\mathbf{x}[n+1]=\mathbf{A}\mathbf{x}[n]+\mathbf{B}\,\delta[n]+\mathbf{D}\mathbf{d}[n],\\&\delta[n+1]=\mathbf{u}[n]+\delta[n],\\&\mathbf{y}[n]=\mathbf{C}\mathbf{x}[n].\end{aligned}\qquad(19)$$

Будем считать, что матрица наблюдателя $\mathbf{G}$ найдена в результате синтеза наблюдателя и выполнен синтез базового алгоритма стабилизации в виде $\mathbf{u}[n]=\mathbf{K}\mathbf{x}[n]+\mathbf{K}_0\delta[n]$. Будем формировать управляющие сигналы по выходу наблюдателя в виде

$$\mathbf{u}[n]=\mathbf{K}\mathbf{z}[n]+\mathbf{K}_0\delta[n],\qquad(20)$$

где $\mathbf{z}$ – оценка вектора состояния системы, полученная с помощью асимптотического наблюдателя

$$\mathbf{z}[n+1]=\mathbf{A}\mathbf{z}[n]+\mathbf{B}\,\delta[n]+\mathbf{G}\left(y[n]-\mathbf{C}\mathbf{z}[n]\right).\qquad(21)$$

Для достижения свойства астатизма введем в (20) аддитивный корректирующий сигнал $\mathbf{K}_\Delta\left(\mathbf{y}[n]-\mathbf{C}\mathbf{z}[n]\right)$, в результате чего выражение (20) преобразуется к виду

$$\mathbf{u}[n]=\mathbf{K}\mathbf{z}[n]+\mathbf{K}_0\delta[n]+\mathbf{K}_\Delta\left(\mathbf{y}[n]-\mathbf{C}\mathbf{z}[n]\right).\qquad(22)$$

Проведя рассуждения аналогичные приведенным выше, приходим к тому, что существует такая матрица $\mathbf{K}_\Delta$, что регулятор с математической моделью

$$\mathbf{u}[n]=\mathbf{H}^{-1}(q)\mathbf{y}_d[n]+\left(\mathbf{K}-\mathbf{K}_\Delta\mathbf{C}\right)\mathbf{z}[n]+\mathbf{K}_0\delta[n]+\mathbf{K}_\Delta\left(\mathbf{y}[n]-\mathbf{y}_d[n]\right),\qquad(23)$$

где $\mathbf{z}$ – оценка вектора состояния системы, полученная с помощью асимптотического наблюдателя (21) для системы (19) с постоянным внешним воздействием $\mathbf{d}=\left\{\mathbf{d}[n]\right\}=\mathbf{d}_0\cdot\left\{d_e[n]\right\}$ обеспечивает астатизм системы по регулируемым координатам и реализацию заданного движения $\mathbf{y}_d[n]$, т.е. выполнение условия $\mathbf{y}[n]\to\mathbf{y}_d[n]$ при $n\to\infty$.

Таким образом, в статье рассмотрены задачи применения многоцелевых структур для управления линейными объектами в их движении по заданным траекториям с обеспечением нулевой статической ошибки при воздействии постоянных внешних возмущений, а также вопросы астатической коррекции стабилизирующих законов управления. Предложены методы и алгоритмы для решения указанных задач.

Литература

1. Веремей Е.И. Синтез законов многоцелевого управления движением морских объектов // Гироскопия и навигация. – 2009. – № 4. С. 3–14.
2. Лукомский Ю. А., Корчанов В. М. Управление морскими подвижными объектами.– СПб.: Элмор, 1996.– 320 с.
3. Лукомский Ю. А., Пешехонов В.Г., Скороходов Д.А. Навигация и управление движением судов. СПб.: «Элмор», 2002. 360с.
4. Мирошник И. В. Теория автоматического управления. Нелинейные и оптимальные системы. – СПб: Питер, 2005. – 271 с.
5. Веремей Е.И. Линейные системы с обратной связью. – СПб.: Изд–во «Лань», 2013. – 448 с.

6. Fossen T.I. Guidance and Control of Ocean Vehicles. John Wiley & Sons. New York, 1999, 480 p.
7. Fossen T.I. Marine control systems. Marine Cybernetics. 2002, 558 p.
8. Holzhuter T. LQG approach for the high–precision track control of ships // IEEE Proceedings on Control Theory and Applications 144(2), 1997. pp. 121–127.
9. Holzhuter T., Schultze R. On the experience with a high–precision track controller for commercial ships // Control Engineering Practise CEP–4(3), 1996. pp. 343–350.
10. Смирнов М.Н. Метод учета ограниченных внешних воздействий при синтезе обратных связей с многоцелевой структурой // Вестн. С.-Петерб. ун-та. Сер. 10: Прикладная математика, информатика, процессы управления. 2014. Вып. 2. С. 130–140.
11. Смирнова М.А. Обеспечение астатизма в системах управления движением морских судов // Вестн. С.-Петерб. ун-та. Сер. 10: Прикладная математика, информатика, процессы управления. 2014. Вып. 2. С. 141–153.
12. Смирнов М.Н. Информационная поддержка процесса обучения при моделировании системы управления шаром на наклонной направляющей // International journal of open information technologies. 2014.Т.2, №3. С. 23-28.
13. Смирнова М.А. Вопросы информатизации обучения на примере программного комплекса цифрового управления роботом // International journal of open information technologies. 2014.Т.2, №3. С. 29-34.
14. Смирнов М.Н. Использование современных информационных технологий для моделирования системы управления шаром на подвижной направляющей // Современные информационные технологии и ИТ-образование. М.: ИНТУИТ.РУ, 2013. №9. С. 728-732.
15. Смирнов М.Н. Оптимизация управления подвижными объектами с ограниченными внешними возмущениями // Современные информационные технологии и ИТ-образование. М.: ИНТУИТ.РУ, 2012. № 8. С. 1018-1024.
16. Смирнова М.А. Программный комплекс цифрового управления роботом, имитирующим башню танка // Современные информационные технологии и ИТ-образование.. М.: ИНТУИТ.РУ, 2013. №9. С. 733-737.
17. Смирнова М.А. Многоцелевое управление подвижными объектами в режиме реального времени // Современные информационные технологии и ИТ-образование. М.: ИНТУИТ.РУ, 2012. №8. С. 1025-1032.
18. Смирнов М.Н. Динамическая компенсация ограниченных внешних возмущений в системе стабилизации курса судна // Материалы XV конференции молодых ученых "Навигация и управление движением". 2013. С. 364-370.
19. Смирнова М.А. Синтез астатических законов управления с неполной обратной связью для морских подвижных объектов // Материалы XV конференции молодых ученых "Навигация и управление движением". 2013. С. 217-223.
20. Смирнов М.Н., Федорова М.А. Компьютерное моделирование системы астатической стабилизации курса морского судна // Процессы управления и устойчивость: Труды 41-й международной научной конференции аспирантов и студентов. СПб, 2010. С. 495-500.
21. Смирнов М.Н. Алгоритм синтеза управлений, подавляющих ограниченные внешние воздействия на морское судно // Процессы управления и устойчивость: Труды 42-й международной научной конференции аспирантов и студентов. СПб, 2011. С. 362-368.
22. Федорова М.А. Синтез и компьютерное моделирование астатической системы управления курсом морского судна // Процессы управления и устойчивость: Труды 42-й международной научной конференции аспирантов и студентов. СПб, 2011. С. 368-374.
23. Смирнов М.Н., Смирнова М.А. Реализация программного комплекса для динамического управления нелинейным объектом // Процессы управления и устойчивость: Труды 44-й международной научной конференции аспирантов и студентов. СПб, 2013. С.297-301.
24. Смирнов М.Н., Смирнова М.А. Современные информационные технологии в процессе обучения технических специалистов // Процессы управления и устойчивость, 2014. Т. 1. С. 397-400.
25. Арзуманян Н.К., Смирнова М.А., Смирнов М.Н. Моделирование системы управления тормозными усилиями автомобиля в Simulink // Процессы управления и устойчивость, 2015. Т. 2. С. 357-363.
26. Какорин Н.С., Смирнов М.Н., Смирнова М.А. Моделирование распространения вирусного заболевания на основе данных о взаимодействии индивидов // Процессы управления и устойчивость, 2015. Т. 2. С. 421-425.
27. Королев Е.А., Петрунин В.Н., Смирнов М.Н., Смирнова М.А. Разработка системы шифрования с применением стегано-криптографических методов // Процессы управления и устойчивость, 2015. Т. 2. С. 437-441.
28. Смирнов А.Н., Смирнов М.Н., Смирнова М.А. Управление аккомпанементом в процессе воспроизведения // Процессы управления и устойчивость, 2015. Т. 2. С. 497-502.
29. Смирнов М.Н., Смирнова М.А. Вопросы синтеза стабилизирующих управлений при наличии неопределенных внешних возмущений // Процессы управления и устойчивость, 2015. Т. 2. С. 503-508.
30. Арзуманян Н.К., Смирнова М.А., Смирнов М.Н. Синтез и моделирование обратных связей антиблокировочной системы автомобиля // Устойчивость и процессы управления: Материалы III международной конференции, посвященной 85-летию со дня рождения профессора, чл.-корр. РАН В.И. Зубова. Спб, 2015. С. 505-506.
31. Веремей Е.И., Смирнов М.Н., Смирнова М.А. Вопросы устойчивости и качества в управлении движением морских судов // Устойчивость и процессы управления: Материалы III международной конференции, посвященной 85-летию со дня рождения профессора, чл.-корр. РАН В.И. Зубова. Спб, 2015. С. 511-512.

Смирнов М.Н.[1], Смирнова М.А.[2]

[1] Санкт-Петербургский государственный университет, г.Санкт-Петербург, ассистент кафедры компьютерных технологий и систем, smirnov-mn@mail.ru

[2] Санкт-Петербургский государственный университет, г.Санкт-Петербург, ассистент кафедры компьютерных технологий и систем, smirnova-ma@bk.ru

ВОПРОСЫ СИНТЕЗА ЦИФРОВЫХ ЗАКОНОВ УПРАВЛЕНИЯ МОРСКИМИ СУДАМИ С УЧЕТОМ НЕОПРЕДЕЛЕННОСТЕЙ

КЛЮЧЕВЫЕ СЛОВА

Морское судно, инвариантный эллипсоид, цифровое управление, неопределенности.

АННОТАЦИЯ

В статье рассматриваются особенности синтеза цифровых законов управления морскими судами с учетом неопределенностей. Особое внимание уделяется ситуации, когда в рамках формализованного задания внешних возмущений имеются неопределенности, и система управления должна не просто их компенсировать в определенном смысле, но ещё обеспечивать выполнение дополнительных требований к динамическим процессам. Предлагается алгоритм решения указанной задачи для цифрового случая.

Уровень развития современных компьютерных технологий, непрерывный рост вычислительных мощностей, появление новых инструментальных программных средств – все это способствует повсеместной автоматизации функционирования механических объектов различного рода с использованием автономных бортовых систем с цифровой реализацией.

В частности, это позволяет устанавливать на современные морские подвижные объекты высокоэффективные системы автоматического управления движением, тем самым облегчая и делая более безопасным для человека выход в открытое море. Такие системы позволяют снизить возможность возникновения аварий из-за человеческого фактора, более экономно расходовать энергетические ресурсы, точно следовать по заданным маршрутам, огибая различные препятствия, компенсировать влияние действующих на него возмущений с учетом особенностей динамики судна.

В связи с этим возникает ряд содержательных и формализованных задач, связанных с проектированием систем автоматического управления движением, а именно: задачи минимизации времени совершения маневра и расхода топлива, задачи построения оптимальных траекторий движения, проблемы подавления внешних воздействий, порождаемых порывами ветра и волнения моря. Чаще всего, все эти задачи эффективно решаются по отдельности, однако на практике зачастую приходится иметь дело с несколькими задачами одновременно.

Особого внимания заслуживает ситуация, когда в рамках формализованного задания внешних возмущений имеются неопределенности, и система управления должна не просто их компенсировать в определенном смысле, но ещё обеспечивать выполнение дополнительных требований к динамическим процессам. Это обстоятельство существенно затрудняет анализ и проектирование системы управления, одной из центральных функций которой является подавление влияния воздействий на судно.

Задача о подавлении внешних возмущений с известными характеристиками относится к основным проблемам теории управления и рассматривается в различных ее разделах и приложениях. В качестве примера можно привести такие работы, как [1 – 13]. Прикладные задачи управления морскими судами рассматриваются в работах [14-29]. В указанных работах известные методы синтеза адаптируются под специфические особенности, присущие динамике морских судов.

Практическая реализация законов управления морскими судами в настоящее время осуществляется с помощью средств цифровой вычислительной техники, что при определенных условиях требует специального учета при синтезе обратных связей. Вопросы синтеза цифровых законов управления морскими судами с учетом неопределенностей являются основной темой,

которой посвящена статья.

Рассмотрим цифровую линейную модель динамики судна

$$\begin{aligned} \boldsymbol{x}[k+1] &= \mathbf{A}\mathbf{x}[k]+\mathbf{B}\boldsymbol{\delta}[k]+\mathbf{D}\mathbf{w}[k], \\ \boldsymbol{\delta}[k+1] &= \boldsymbol{u}[k]+\boldsymbol{\delta}[k], \\ \boldsymbol{y}[k] &= \mathbf{C}\mathbf{x}[k], \end{aligned} \tag{1}$$

где $\boldsymbol{x}\in E^n$ – вектор состояния (здесь он определяет отклонения от положения равновесия), $\boldsymbol{\delta}\in E^m$ – вектор состояния исполнительных органов, $w\in E^l$ – вектор внешних возмущающих воздействий, $\boldsymbol{u}\in E^m$ – вектор управляющих сигналов (управлений), $\mathbf{A}$, $\mathbf{B}$, $\mathbf{D}$, $\mathbf{C}$ и $\mathbf{M}$ – матрицы соответствующих размерностей с постоянными компонентами.

Для простоты будем считать, что измеряемый выход $\boldsymbol{y}$ системы одновременно является и контролируемой переменной.

Будем формировать обратную связь для системы (1) в виде

$$\boldsymbol{u}[k]=\boldsymbol{K}_x\boldsymbol{x}[k]+K_\delta\boldsymbol{\delta}[k]=\boldsymbol{K}\begin{pmatrix}\boldsymbol{x}[k]\\ \hline \boldsymbol{\delta}[k]\end{pmatrix},\ \mathbf{K}=(\mathbf{K}_x \mid \mathbf{K}_\delta), \tag{2}$$

где матрицы $\boldsymbol{K}_x$ и $\boldsymbol{K}_\delta$ соответствующих размерностей имеют постоянные компоненты.

Обозначим через Ω_k множество стабилизирующих регуляторов, для которых корни характеристического полинома

$$\Delta_3(z,\boldsymbol{K})=\det\left(\boldsymbol{E}_{n+m}z-\boldsymbol{A}_0-\boldsymbol{B}_0\boldsymbol{K}\right),\ \mathbf{A}_0=\left(\begin{array}{c|c}\mathbf{A} & \mathbf{B}\\ \hline \mathbf{0} & \mathbf{E}_m\end{array}\right),\ \mathbf{B}_0=\begin{pmatrix}\mathbf{0}\\ \hline \mathbf{E}_m\end{pmatrix} \tag{3}$$

замкнутой системы (1), (2) лежат внутри единичного круга.

Рассмотрим также сужение Ω_{ks} множества стабилизирующих регуляторов, определяя его желаемыми модальными требованиями к замкнутой системе:

$$\Omega_{sk}=\left\{\ \boldsymbol{K}\in\Omega_k:\ \delta_i(\boldsymbol{K})\in C_\Delta,\ i=\overline{1,n+m}\right\}\ , \tag{4}$$

где $\delta_i(\boldsymbol{K})$ – корни характеристического полинома (3) замкнутой системы, C_Δ – заданная область на комплексной плоскости. В частности, в качестве такой области можно принять круг $C_\Delta=\{z\in\boldsymbol{C}^1:\ |z|\le\alpha_d\}$, где $\alpha_d\in(0,1)$ – заданное вещественное число, определяющее степень устойчивости замкнутой системы. Как и для аналоговых регуляторов [6, 11, 18], возможны и другие способы введения допустимой области [30].

Будем считать, что внешнее возмущающее воздействие ограничено по величине, т.е. справедливо соотношение

$$\|\boldsymbol{w}[k]\|_\infty\le 1,\quad k=0,1,2,\ldots\ . \tag{5}$$

Определим функционал $J_d=J_d(\boldsymbol{K})$, характеризующий размер минимального инвариантного эллипсоида, включающего множество $\mathfrak{R}_{\boldsymbol{e}a}$ реакций на внешние воздействия для замкнутой системы (1), (2):

$$J_d=J_d(\boldsymbol{K})=f\left(\boldsymbol{P}(\alpha_0)\right)=\mathrm{tr}\left(\boldsymbol{C_3}\boldsymbol{P}(\alpha_0)\boldsymbol{C}_3'\right)\ , \tag{6}$$

где $\alpha_0=\arg\min\limits_{0<\alpha<1} f\left(\boldsymbol{P}(\alpha)\right)$, $f\left(\boldsymbol{P}(\alpha)\right)=\mathrm{tr}\left(\boldsymbol{C_3}\boldsymbol{P}(\alpha)\boldsymbol{C}_3'\right)$, $\boldsymbol{P}(\alpha)$ – положительно определенное решение линейного матричного уравнения

$$\boldsymbol{A_3}\boldsymbol{P}\boldsymbol{A}_3'-\alpha\boldsymbol{P}+\frac{\alpha}{1-\alpha}\boldsymbol{D_3}\boldsymbol{D}_3'=0\ , \tag{7}$$

$$\boldsymbol{A_3}=\boldsymbol{A_0}+\boldsymbol{B_0}\boldsymbol{K}\ ,\ \mathbf{C}_3=(\mathbf{C}\mid\mathbf{0})\ ,\ \mathbf{D}_3=\begin{pmatrix}\mathbf{D}\\ \hline \mathbf{0}\end{pmatrix}.$$

Определение. Эллипсоид с центром в начале координат

$$\varepsilon_x=\left\{\ \boldsymbol{x}\in E^n:\ \boldsymbol{x}'\boldsymbol{P}^{-1}\boldsymbol{x}\le 1\right\}\ ,\quad \boldsymbol{P}>0,$$

будем называть инвариантным по переменной $\mathbf{x}_c$ (по состоянию) для динамической системы

$$\dot{\boldsymbol{x}}_c = \boldsymbol{A}_c \boldsymbol{x}_c + \boldsymbol{D}_c \boldsymbol{w},$$
$$\boldsymbol{e} = \boldsymbol{C}_c \boldsymbol{x}_c.$$
$$\|\boldsymbol{w}(t)\| \le 1 \quad \text{при } 0 \le t < \infty,$$

если из условия $\boldsymbol{x}(0) \in \varepsilon_x$ следует $\boldsymbol{x}(t) \in \varepsilon_x$ для всех моментов времени $0 \le t < \infty$. Симметрическую матрицу $\boldsymbol{P}$ при этом будем называть матрицей эллипсоида ε_x.

В определении и далее по тексту факт положительной определенности матрицы отражается обозначением $\boldsymbol{P} > 0$.

Иначе говоря, любая траектория системы $\boldsymbol{x}(t)$, начинающаяся внутри эллипсоида ε_x, в каждый последующий момент времени будет оставаться внутри него, что качественно показано на рис. 1.

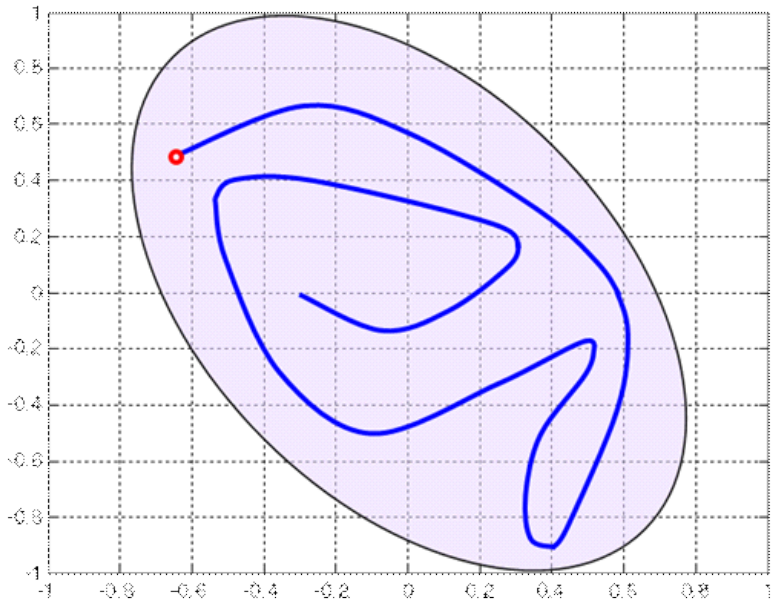


Рис. 1. Инвариантный эллипсоид

Рассмотрим задачу о выборе стабилизирующего регулятора (2), который минимизирует размер J_d инвариантного эллипсоида с учетом желаемых модальных свойств замкнутой системы

$$J_d = J_d(\boldsymbol{K}) \to \min_{K \in \Omega_{sk} \subseteq \Omega_k}, \quad (8)$$

где допустимое множество Ω_{sk} определяется формулой (4).

Если не принимать во внимание дополнительные модальные свойства, то мы приходим к постановке известной задачи

$$J_d = J_d(\boldsymbol{K}) \to \min_{K \in \Omega_k}, \quad (9)$$

которая рассмотрена в работе [31 – 33], где приводится следующее ее решение:

$$J_d = J_d(\boldsymbol{K}) = \min_{K \in \Omega_k} J_d(\boldsymbol{K}) = F\left(\boldsymbol{P}(\alpha, \beta)\right) = \mathrm{tr}\left(\boldsymbol{C_3} \boldsymbol{P}(\alpha, \beta) \boldsymbol{C}_3^{'}\right), \quad (10)$$

где $(\alpha, \beta) = \arg \min_{\alpha>0,\ \beta>0} F\left(\boldsymbol{P}(\alpha, \beta)\right)$, $F\left(\boldsymbol{P}(\alpha, \beta)\right) = \mathrm{tr}\left(\boldsymbol{C_3} \boldsymbol{P}(\alpha, \beta) \boldsymbol{C}_3^{'}\right)$, $\boldsymbol{P}(\alpha, \beta)$ – положительно определенное решение линейного матричного уравнения

$$\boldsymbol{A_0} \boldsymbol{P} \boldsymbol{A}_0^{'} - 2\beta \boldsymbol{B_0} \boldsymbol{B}_0^{'} + \beta^2 \boldsymbol{B_0} \boldsymbol{B}_0^{'} \left(\boldsymbol{A}_0^{'}\right)^{-1} \boldsymbol{P^{-1}} \boldsymbol{A_0^{-1}} \boldsymbol{B_0} \boldsymbol{B}_0^{'} - \alpha \boldsymbol{P} + \frac{\alpha}{1-\alpha} \boldsymbol{D_3} \boldsymbol{D}_3^{'} = 0 \quad (11)$$

$$\boldsymbol{K} = \arg \min_{K \in \Omega_k} J_d(\boldsymbol{K}) = -\beta \boldsymbol{B}_0^{'} \left(\boldsymbol{A}_0^{'}\right)^{-1} \boldsymbol{P^{-1}}(\alpha, \beta). \quad (12)$$

Введем обозначение для шуровского характеристического полинома замкнутой системы (1), (2), (12)

$$\Delta_3(z,\alpha,\beta)=\det\left(\boldsymbol{E}_{n+m}z-\boldsymbol{A}_0-\boldsymbol{B}_0\boldsymbol{K}(\alpha,\beta)\right). \tag{13}$$

Заметим, что решение задачи (8) удобно связать с параметризацией вещественными векторами $\gamma\in E^{n+m}$ характеристических полиномов системы, замкнутой регуляторами из допустимого множества Ω_{sk}, введенного соотношением (4).

Указанная параметризация множества характеристических полиномов определяет соответствующую параметризацию множества Ω_{sk} регуляторов (2), которыми замыкается объект (1). Действительно, зададим произвольный вектор $\boldsymbol{\gamma}\in E^{n+m}$ и по указанным формулам построим полином $\Delta^*(z,\boldsymbol{\gamma})$. Для того, чтобы он был характеристическим полиномом для замкнутой системы необходимо и достаточно, чтобы матрица $\boldsymbol{K}$ коэффициентов регулятора (2) удовлетворяла тождеству

$$\Delta_3(z,\boldsymbol{K})=\det\left(\boldsymbol{E}_{n+m}z-\boldsymbol{A}_0-\boldsymbol{B}_0\boldsymbol{K}\right)\equiv\Delta(z,\gamma).$$

Если собрать все компоненты матрицы $\boldsymbol{K}$ в вектор $\boldsymbol{k}\in E^{m\times n+m\times m}$, то этому тождеству будет соответствовать эквивалентная линейная система

$$\boldsymbol{\Gamma}\mathbf{k}=\boldsymbol{m}(\gamma),\ \mathbf{A}_0=\left(\begin{array}{c|c}\mathbf{A}&\mathbf{B}\\\hline\mathbf{0}&\mathbf{0}\end{array}\right),\ \mathbf{B}_0=\left(\begin{array}{c}\mathbf{0}\\\hline\mathbf{E}_m\end{array}\right).$$

где компоненты матрицы Γ определяются только матрицами $\boldsymbol{A}_0$ и $\boldsymbol{B}_0$, а вектор $\mathbf{m}(\boldsymbol{\gamma})$ определяется коэффициентами полинома $\Delta^*(s,\boldsymbol{\gamma})$ и характеристического полинома матрицы $\boldsymbol{A}_0$.

Заметим, что указанная система всегда совместна в силу условия полной управляемости. Она содержит $n+m$ уравнений и имеет $m\times n+m\times m$ неизвестных, т.е. $n_c=m\times n+m\times m-n-m$ компонент вектора $\boldsymbol{k}$, собранные в вектор $\boldsymbol{h}_c\in E^{n_c}$, могут быть выбраны произвольно. Введем в рассмотрение вектор $\varepsilon=\left|\gamma,\boldsymbol{h}_c\right|\in E^{\lambda}$, $\lambda=n+m+n_c$, и, дополнительно задавая произвольный вектор $\boldsymbol{h}_c$, найдем соответствующее решение $\boldsymbol{k}(\varepsilon)=\boldsymbol{k}(\gamma,\boldsymbol{h}_c)$ линейной системы. Таким образом, найдена матрица $\boldsymbol{K}=\boldsymbol{K}(\varepsilon)=\boldsymbol{K}(\gamma,\boldsymbol{h}_c)$ коэффициентов регулятора (2) из множества Ω_{sk}, параметризованного векторами $\varepsilon=\left|\gamma,\boldsymbol{h}_c\right|$.

На базе приведенной параметризации можно сформулировать следующее утверждение:

Теорема. Для функционала $J_d(\boldsymbol{K})$ в задаче (8) существует минимизирующая последовательность $\{\boldsymbol{K}(\varepsilon_i)\}$ регуляторов (2), определяемая последовательностью векторов $\{\boldsymbol{\varepsilon}_i\}$ такой, что

$$\lim_{i\to\infty}\{\boldsymbol{K}(\varepsilon_i)\}=K_0=\arg\min_{K\in\Omega_{sk}}J_d(\boldsymbol{K}),\ \lim_{i\to\infty}\left|J_d\left(\boldsymbol{K}(\varepsilon_i)\right)\right|=J_d(\boldsymbol{K_0})=J_d^0.$$

Доказательство данного утверждения проводится аналогично доказательству теоремы для непрерывного времени, однако необходимо заметить, что для построения полинома с заданной степенью устойчивости используется следующая схема [29]:

Для любого вектора $\boldsymbol{\gamma}\in E^{n_d}$ степень устойчивости вспомогательного полинома, построенного по формулам

$$\Delta^*(z,\boldsymbol{\gamma})=\begin{cases}\widetilde{\Delta}^*(z,\boldsymbol{\gamma}),\ \text{если } n_d\text{ - четное;}\\(z-a_{d+1}(\boldsymbol{\gamma},\alpha_d))\widetilde{\Delta}^*(z,\boldsymbol{\gamma}),\ \text{если } n_d\text{ - нечетное}\end{cases} \tag{14}$$

не меньше наперед заданной величины $\alpha_d\in(0,1)$, и обратно, если степень устойчивости некоторого полинома $\Delta(s)$ не меньше величины $\alpha_d\in(0,1)$, то можно указать такой вектор $\boldsymbol{\gamma}\in E^{n_d}$, что $\Delta(z)\equiv\Delta^*(z,\boldsymbol{\gamma})$, причем

$$\Delta(z,\gamma)=\prod_{i=1}^{d}\left(z^2+a_i^1(\gamma,\alpha_d)z+a_i^0(\gamma,\alpha_d)\right), \text{ где} \tag{15}$$

$$a_i^1(\gamma,\alpha_d)=-\alpha_d\left(\exp\left(-\frac{\gamma_{i1}^2}{2}+\sqrt{\frac{\gamma_{i1}^4}{4}-\gamma_{i2}^2}\right)+\exp\left(-\frac{\gamma_{i1}^2}{2}-\sqrt{\frac{\gamma_{i1}^4}{4}-\gamma_{i2}^2}\right)\right),$$
$$a_i^0(\gamma,\alpha_d)=\alpha_d^2\exp\left(-\gamma_{i1}^2\right),\quad i=\overline{1,d},\quad a_{d+1}\left(\gamma,\alpha_d\right)=\alpha_d\exp\left(-\gamma_{d0}^2\right), \tag{16}$$
$$\gamma=\{\gamma_{11},\gamma_{12},\gamma_{21},\gamma_{22},\ldots,\gamma_{d1},\gamma_{d2},\gamma_{d0}\}.$$

Таким образом, приведенная теорема является основой для построения алгоритма решения задачи (8). Схему решения задачи (8) можно сформулировать в виде следующего алгоритма.

Алгоритм.

1. Взять любую точку $\gamma\in E^{n+m}$ и построить вспомогательный полином $\Delta^*(z,\boldsymbol{\gamma})$ по формулам (15) – (16) либо по аналогичным формулам [30] для более сложного варианта модальных условий.

2. Сформировать систему линейных уравнений $\boldsymbol{\Gamma k}=\boldsymbol{m}(\gamma)$, где $\boldsymbol{k}$ – вектор, собранный из компонент матрицы $\boldsymbol{K}$, обеспечивающую выполнение тождества

$$\Delta_3(z,\boldsymbol{K})\equiv\Delta(z,\gamma),$$

которая всегда совместна и, если ее решение не является единственным, осуществить произвольный выбор вектора $\boldsymbol{h_c}\in E^{n_c}$ свободных переменных (компонент матрицы $\boldsymbol{K}$) по отношению к этой системе.

3. После подстановки в систему $\boldsymbol{\Gamma k}=\boldsymbol{m}(\gamma)$ принятого вектора $\boldsymbol{\varepsilon}=\left\{\gamma,\boldsymbol{h_c}\right\}\in E^{\lambda}$, $\lambda=n+m+n_c$, найти ее решение $\boldsymbol{k}=\widetilde{\boldsymbol{k}}(\varepsilon)$, а, соответственно, и матрицу $\boldsymbol{K}=\widetilde{\boldsymbol{K}}(\boldsymbol{\varepsilon})$.

4. Подставить найденную матрицу $\boldsymbol{K}=\widetilde{\boldsymbol{K}}(\boldsymbol{\varepsilon})$ коэффициентов обратной связи (2) в соотношения (6), (7) и найти для замкнутой системы (1), (2) соответствующее значение $J_d=J_d(\widetilde{\boldsymbol{K}}(\varepsilon))=\widetilde{J}_d(\varepsilon)$ размера минимального инвариантного эллипсоида.

5. С помощью любого допустимого численного метода решения задачи

$$J_d=J_d(\varepsilon)=J_d\left(\boldsymbol{K}(\boldsymbol{\varepsilon})\right)\to\min_{\varepsilon\in E^{\lambda}}$$

на безусловный экстремум, задать новую точку $\boldsymbol{\varepsilon}$ и, повторяя пункты 3,4, минимизировать функцию $J_d(\boldsymbol{\varepsilon})$.

6. Процесс завершить после нахождения точки $\varepsilon_0=\arg\min_{\varepsilon\in E^{\lambda}} J_d(\varepsilon)$, определяя при этом матрицу $\boldsymbol{K_0}=\widetilde{\boldsymbol{K}}(\boldsymbol{\varepsilon}_0)$, которая принимается в качестве решения задачи

$$J_d=J_d(\boldsymbol{K})\to\min_{K\in\Omega_{sk}\subseteq\Omega_k}$$

обеспечивающего минимальное значение $\min_{K\in\Omega_{sk}} J_d(\boldsymbol{K})=J_d(\boldsymbol{K}_0)=J_d^0=J_d(\varepsilon_0)$ размера минимального инвариантного эллипсоида.

Таким образом, в статье рассмотрены особенности синтеза цифровых законов управления морскими судами с учетом неопределенностей и предложен алгоритм решения задачи (8).

Литература

1. Веремей Е.И. Синтез законов многоцелевого управления движением морских объектов // Гироскопия и навигация. – 2009. – № 4. С. 3–14.
2. Лукомский Ю. А., Корчанов В. М. Управление морскими подвижными объектами.– СПб.: Элмор, 1996.– 320 с.
3. Лукомский Ю. А., Пешехонов В.Г., Скороходов Д.А. Навигация и управление движением судов. СПб.: «Элмор», 2002. 360с.
4. Мирошник И. В. Теория автоматического управления. Нелинейные и оптимальные системы. – СПб: Питер, 2005. – 271 с.

5. Назин А.В., Назин С.А., Поляк Б. Т. О сходимости внешних эллипсоидальных аппроксимаций областей достижимости линейных дискретных динамических систем // АиТ. 2004. 38. С. 39–61.
6. Смирнов М.Н. Метод учета ограниченных внешних воздействий при синтезе обратных связей с многоцелевой структурой // Вестн. С.-Петерб. ун-та. Сер. 10: Прикладная математика, информатика, процессы управления. 2014. Вып. 2. С. 130–140.
7. Смирнова М.А. Обеспечение астатизма в системах управления движением морских судов // Вестн. С.-Петерб. ун-та. Сер. 10: Прикладная математика, информатика, процессы управления. 2014. Вып. 2. С. 141–153.
8. Смирнов М.Н. Информационная поддержка процесса обучения при моделировании системы управления шаром на наклонной направляющей // International journal of open information technologies. 2014.Т.2, №3. С. 23-28.
9. Смирнова М.А. Вопросы информатизации обучения на примере программного комплекса цифрового управления роботом // International journal of open information technologies. 2014.Т.2, №3. С. 29-34.
10. Смирнов М.Н. Использование современных информационных технологий для моделирования системы управления шаром на подвижной направляющей // Современные информационные технологии и ИТ-образование. М.: ИНТУИТ.РУ, 2013. №9. С. 728-732.
11. Смирнов М.Н. Оптимизация управления подвижными объектами с ограниченными внешними возмущениями // Современные информационные технологии и ИТ-образование. М.: ИНТУИТ.РУ, 2012. № 8. С. 1018-1024.
12. Смирнова М.А. Программный комплекс цифрового управления роботом, имитирующим башню танка // Современные информационные технологии и ИТ-образование.. М.: ИНТУИТ.РУ, 2013. №9. С. 733-737.
13. Смирнова М.А. Многоцелевое управление подвижными объектами в режиме реального времени // Современные информационные технологии и ИТ-образование. М.: ИНТУИТ.РУ, 2012. №8. С. 1025-1032.
14. Смирнов М.Н. Динамическая компенсация ограниченных внешних возмущений в системе стабилизации курса судна // Материалы XV конференции молодых ученых "Навигация и управление движением". 2013. С. 364-370.
15. Смирнова М.А. Синтез астатических законов управления с неполной обратной связью для морских подвижных объектов // Материалы XV конференции молодых ученых "Навигация и управление движением". 2013. С. 217-223.
16. Поляк Б. Т., Щербаков П. С. Робастная устойчивость и управление. М.: Наука, 2002.
17. Смирнов М.Н., Федорова М.А. Компьютерное моделирование системы астатической стабилизации курса морского судна // Процессы управления и устойчивость: Труды 41-й международной научной конференции аспирантов и студентов. СПб, 2010. С. 495-500.
18. Смирнов М.Н. Алгоритм синтеза управлений, подавляющих ограниченные внешние воздействия на морское судно // Процессы управления и устойчивость: Труды 42-й международной научной конференции аспирантов и студентов. СПб, 2011. С. 362-368.
19. Федорова М.А. Синтез и компьютерное моделирование астатической системы управления курсом морского судна // Процессы управления и устойчивость: Труды 42-й международной научной конференции аспирантов и студентов. СПб, 2011. С. 368-374.
20. Смирнов М.Н., Смирнова М.А. Реализация программного комплекса для динамического управления нелинейным объектом // Процессы управления и устойчивость: Труды 44-й международной научной конференции аспирантов и студентов. СПб, 2013. С.297-301.
21. Смирнов М.Н., Смирнова М.А. Современные информационные технологии в процессе обучения технических специалистов // Процессы управления и устойчивость, 2014. Т. 1. С. 397-400.
22. Арзуманян Н.К., Смирнова М.А., Смирнов М.Н. Моделирование системы управления тормозными усилиями автомобиля в Simulink // Процессы управления и устойчивость, 2015. Т. 2. С. 357-363.
23. Какорин Н.С., Смирнов М.Н., Смирнова М.А. Моделирование распространения вирусного заболевания на основе данных о взаимодействии индивидов // Процессы управления и устойчивость, 2015. Т. 2. С. 421-425.
24. Королев Е.А., Петрунин В.Н., Смирнов М.Н., Смирнова М.А. Разработка системы шифрования с применением стегано-криптографических методов // Процессы управления и устойчивость, 2015. Т. 2. С. 437-441.
25. Смирнов А.Н., Смирнов М.Н., Смирнова М.А. Управление аккомпанементом в процессе воспроизведения // Процессы управления и устойчивость, 2015. Т. 2. С. 497-502.
26. Смирнов М.Н., Смирнова М.А. Вопросы синтеза стабилизирующих управлений при наличии неопределенных внешних возмущений // Процессы управления и устойчивость, 2015. Т. 2. С. 503-508.
27. Арзуманян Н.К., Смирнова М.А., Смирнов М.Н. Синтез и моделирование обратных связей антиблокировочной системы автомобиля // Устойчивость и процессы управления: Материалы III международной конференции, посвященной 85-летию со дня рождения профессора, чл.-корр. РАН В.И. Зубова. Спб, 2015. С. 505-506.
28. Веремей Е.И., Смирнов М.Н., Смирнова М.А. Вопросы устойчивости и качества в управлении движением морских судов // Устойчивость и процессы управления: Материалы III международной конференции, посвященной 85-летию со дня рождения профессора, чл.-корр. РАН В.И. Зубова. Спб, 2015. С. 511-512.
29. Веремей Е.И. Алгоритмы решения одного класса задач H_∞ -оптимизации систем управления // Известия РАН. Теория и системы управления. — 2011. — № 3. — С. 52 – 61.
30. Веремей Е. И. Обеспечение заданной степени устойчивости регуляторами с неполной информацией // Известия АН СССР. Техн. кибернетика. 1986. № 4. С. 123–130.
31. Назин С. А., Поляк Б. Т., Топунов М. В. Подавление ограниченных внешних возмущений с помощью метода инвариантных эллипсоидов. Автомат. и телемех., 2007, № 3, 106–125
32. Поляк Б. Т., Щербаков П.С. Трудные задачи линейной теории управления. Некоторые подходы к решению // АиТ. 2005. №5. С. 7–46.
33. Поляк Б.Т., Щербаков П. С. Техника D–разбиения при решении линейных матричных неравенств // АиТ. 2006. N11. С. 159–174.

Жук А.П.[1], Бурмистров В.А.[2], Гавришев А.А.[3]

[1]ФГАОУ ВПО СКФУ, г. Ставрополь, к.т.н., профессор кафедры «Организация и технология защиты информации», alekszhuk@mail.ru

[2]ФГАОУ ВПО СКФУ, г. Ставрополь, лаборант кафедры «Организация и технология защиты информации», alexxx.2008@inbox.ru

[3]ФГАОУ ВПО СКФУ, г. Ставрополь, аспирант, alexxx.2008@inbox.ru

СИСТЕМА ПЕРЕДАЧИ ИНФОРМАЦИИ С ИСПОЛЬЗОВАНИЕМ СТОХАСТИЧЕСКИХ ОРТОГОНАЛЬНЫХ АНСАМБЛЕЙ ДИСКРЕТНЫХ МНОГОУРОВНЕВЫХ СИГНАЛОВ

КЛЮЧЕВЫЕ СЛОВА

3G, 4G, LTE, CDMA, структурная скрытность системы передачи информации, стохастические последовательности, системы мобильной радиосвязи.

АННОТАЦИЯ

В статье представлен анализ развития сетей 3G и 4G/LTE, а также темпы их развития. Анализируются способы обеспечения защищенности систем мобильной радиосвязи и их недостатки, для устранения авторами предлагается способ обеспечения защищенности систем мобильной радиосвязи на основе стохастических ортогональных кодовых последовательностей.

Стремительное развитие сетей мобильной радиосвязи наблюдается во всём мире. Активно развиваются сотовые, транкинговые, пейджинговые сети, а также сети абонентского радиодоступа.

По итогам третьего квартала 2013 г. число активных SIM-карт мобильной связи в мире составило 6,6 млрд, и по прогнозам Ericsson, к 2017 г. превысит 8,4 млрд. Активно развиваются сети 3G и 4G/LTE [9]. Ожидается, что к концу 2017 г. мировые доходы от мобильной передачи данных достигнут $200 млрд [9].

Тем не менее, существуют следующие основные проблемы перехода к сетям четвёртого поколения [9]:

1. Дефицит доступного спектра частот в большинстве регионов;
2. Неопределённость архитектуры оказания голосовых услуг в сетях LTE и гетерогенных сетях (беспроводные сети, состоящие из классических базовых станций (макросот), малых сот (фемто-, пико- и микросоты) и точек доступа Wi-Fi операторского класса);
3. Ожидание возврата инвестиций, вложенных в развитие предшествующей технологии 3G.

По оценке J'son & Partners Consulting, 3G-покрытие (доля населения, проживающего на территории, охваченной сетями 3G) на конец 2012 г. превысило 75%, в то время как в 2010 г. этот показатель составлял около 65% [10]. По прогнозу Ericsson, в 2018 г. в мире технология WCDMA/HSPA будет доминировать на мировом рынке [9]. В 2015 г. ожидается появление технологий HSPA+Advanced и WCDMA+. Согласно прогнозу Informa Telecoms & Media, количество подключений к HSPA в мире к концу 2017 года вырастет до 4,2 млрд., в то время как число LTE-подключений составит 940 млн [9] (рис. 1).

По прогнозам J'son & Partners Consulting, до 2018 г. технологии третьего поколения будут доминировать в России с долей около 78% по количеству абонентов. На долю доходов от мобильной передачи данных в сетях 2G/3G придётся до 80% всех поступлений от мобильного интернет-доступа в России к концу прогнозного периода [9] (рис. 2).

Со времён появления работ К. Шеннона, В.А. Котельникова и А.А. Харкевича было известно, что широкополосные сигналы, используемые в современных системах радиосвязи, потенциально очень эффективны для передачи информации. С 80-х годов 20-го века в системах радиосвязи стали использовать хаотические сигналы для передачи информации.

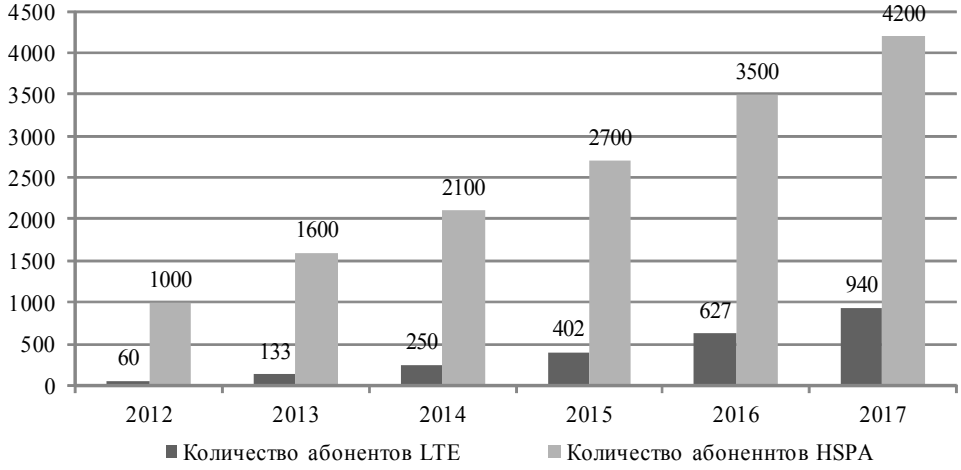


Рис. 1. Динамика абонентской базы HSPA и LTE, 2012 – 2017 гг., млн

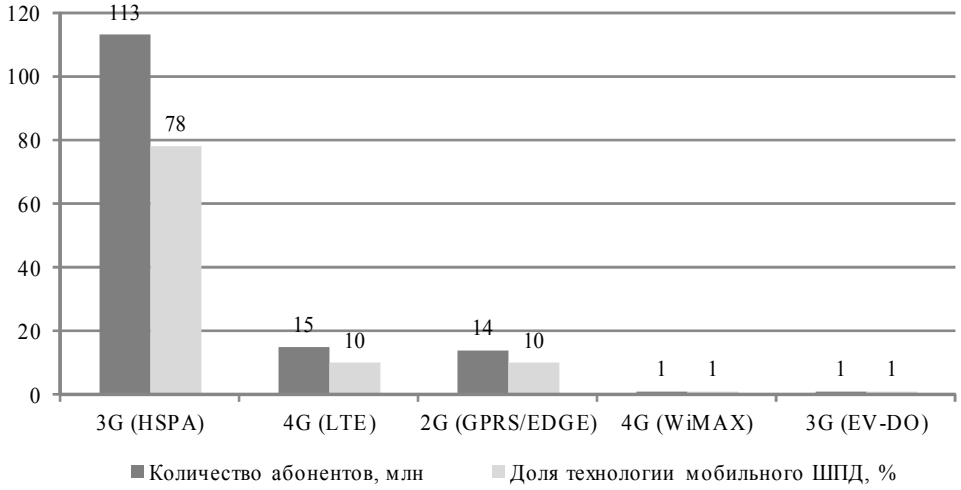


Рис. 2. Прогноз количества абонентов (млн) и долей (%) различных технологий мобильного ШПД в России на 2018 г.

Открытие нерегулярных хаотических колебаний в детерминированных нелинейных динамических системах различной среды (физических, механических, радиотехнических, химических, биологических и т.д.) стало одной из крупнейших научных сенсаций конца 20-го века. Это явление называется детерминированным или динамическим хаосом [3, 4]. Концепция динамического хаоса вышла за рамки породившей её теории нелинейных колебаний и стала новой общенаучной парадигмой, легла в основу нового научного направления, называемого синергетикой. Более того, явление динамического хаоса дало новые важные инженерные идеи, привело к созданию на их основе устройств и теорий, уже используемых на практике, например, технология помехоустойчивой передачи информации квазислучайными (стохастическими) кодами [5]. В частности, имеются предложения по построению коммуникационных систем конфиденциальной передачи информации с использованием в качестве несущих хаотических колебаний, возникающих в электронных устройствах, называемых генераторами хаоса или генераторами хаотических колебаний [3, 5].

Одним из свойств, определяющих привлекательность динамического хаоса для использования в системах радиосвязи, является конфиденциальность передачи информации [5]. Двухпараметрические операции прямого (на передающей стороне) и обратного (на приёмной стороне) стохастического преобразования могут рассматриваться как блочно-потоковый криптоалгоритм, сочетающий сложное преобразование на длине блока, как в блочном алгоритме, с применением источника кодирующей последовательности от независимого источника

(стохастические генераторы псевдослучайных чисел), как в потоковом алгоритме [4, 5, 7]. В этих условиях методы стохастического кодирования, сочетающие введение избыточности при обнаружении и исправлении ошибок со стохастическим криптографическим преобразованием имеют самостоятельное значение как средство обеспечения нескольких задач криптографической защиты информации [5].

Современные системы мобильной связи используют технологии связи третьего и четвёртого поколения (3G и 4G). Технология передачи информации CDMA [6] (Code Division Multiple Access – множественный доступ с кодовым разделением каналов) третьего поколения используется в следующих технологиях 1995 (CdmaOne), 1999 (UMTS), 2000 (CDMA2000, 1xEVDO), 2001(FOMA), 2006 (HSDPA), 2007 (HSUPA). Системы мобильной связи четвёртого поколения используют технологии MIMO, MultiCarrier CDMA (MC-CDMA), OFDMA, UWB и программно – реконфигурируемого радио (Software Defined Radio). Поскольку технология передачи информации CDMA основана на использовании ортогональных сигналов, то от их характеристик и способа использования зависят важнейшие свойства системы в целом. Вариантом обеспечения конфиденциальности передаваемой информации в системах мобильной связи, основанных на использовании технологии передачи информации CDMA, является применение генераторов стохастических ортогональных сигналов [2, 5], теория и практика построения которых в настоящее время глубоко не проработана.

Целью статьи является разработка способа и устройства, позволяющего повысить структурную скрытность системы передачи информации с кодовым разделением каналов.

Для этого необходимо разработать структуру устройства, реализующего способ передачи информации на основе хаотически формируемых ансамблей дискретных многоуровневых ортогональных сигналов (ДМОС).

Существуют устройства и способы, позволяющие находить ортогональные наборы функций для построения системы связи [2, 3, 4, 7].

Существует способ многостанционного доступа с кодовым разделением каналов, суть которого заключается в расширении спектра частот на основе использования 64 видов последовательностей, сформированных по закону функций Уолша.

Недостатком этого способа является то, что сигналы Уолша имеют регулярную структуру, которая заранее известна и поэтому широкополосная система радиосвязи с кодовым разделением каналов, построенная на основе этого способа, будет обладать низкой структурной скрытностью [7].

Существует способ и устройство, его реализующее, - генератор функций Попенко – Турко [2]. Генератор функций Попенко–Турко вычисляет собственные векторы действительной симметрической положительно определённой матрицы, которые описывают систему ортогональных сигналов.

Недостатками генератора функций Попенко–Турко является отсутствие автоматического поступления входных данных для обеспечения фактора случайности в формируемых им системах ортогональных сигналов, а также отсутствие возможности генерирования хаотических ансамблей ортогональных сигналов (собственных векторов) различной структуры [7].

Существует способ, используемый в системе передачи данных с кодовым разделением каналов [8], который включает операцию одновременной передачи сложных широкополосных сигналов на основе нелинейных последовательностей де Брейна со сменой формы последовательности в процессе передачи сообщения от одного информационного символа к другому.

Несмотря на то, что количество ортогональных сигналов, формируемых на основе кодовых словарей де Брейна, больше числа ортогональных сигналов Уолша размерности N, однако их количество является конечным для любой размерности N, что позволяет сделать вывод об их низкой структурной скрытности [7].

Для устранения указанных недостатков вышеперечисленных способов, а также для повышения структурной скрытности системы передачи информации с кодовым разделением каналов предлагается использовать ансамбли ортогональных сигналов, хаотически формируемые на основе собственных векторов диагональной положительно определенной симметрической матрицы размерностью N. Для передачи сообщений, сменяемых от одного информационного символа к другому, применяют ортогональные кодовые комбинации, в качестве которых используют ансамбли дискретных ортогональных сигналов, формируемые путем расчета собственных чисел и собственных векторов диагональной положительно определенной

симметрической матрицы, диагональными коэффициентами которой являются хаотически формируемые числовые последовательности.

В работе предлагается следующая структурная схема системы связи на основе генератора стохастических ортогональных ансамблей дискретных многоуровневых сигналов.

Передающая часть системы связи на основе хаотически формируемых ансамблей дискретных многоуровневых сигналов показан на рис. 3. Приёмная часть системы связи показана на рис. 4.

Рис. 3. Передающая часть системы связи

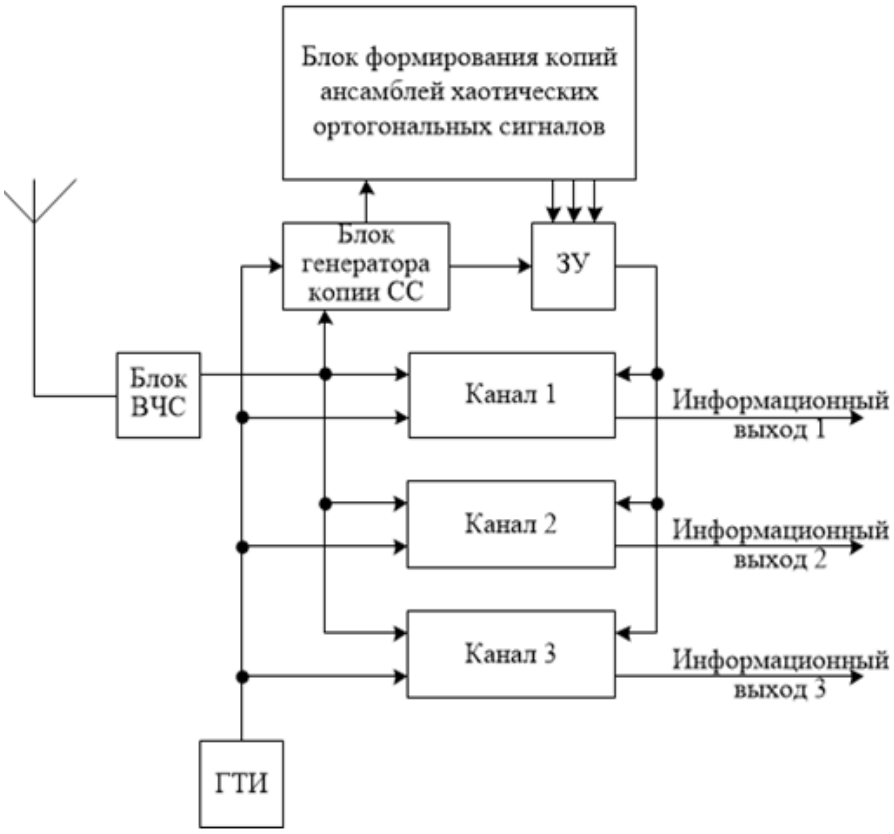


Рис. 4. Приёмная часть системы связи

На рисунке 3 введены следующие обозначения: ГТИ – генератор тактовых импульсов; ГСС – генератор сигналов синхронизации.

На рисунке 4 введены следующие обозначения: СС – сигнал синхронизации; ЗУ – запоминающее устройство; ВЧС – высокочастотная селекция; ГТИ – генератор тактовых импульсов.

Отметим, что каждый из каналов на передающей стороне состоит из запоминающего устройства (ЗУ), блока цифровой информации и модулятора; блок формирования группового сигнала – из сумматора и модулятора; блок модуляции и усиления представляет собой совокупность блока фазовой модуляции и усилителя мощности; каждый из каналов на приёмной стороне состоит из блока корреляционной обработки, блока выделения информации и блока приема информации; блок генератора копии сигналов синхронизации является совокупностью блоков (блок обнаружения сигнала синхронизации, блок поиска и генератора копии сигналов синхронизации).

Пусть информационные сигналы поступают в запоминающее устройство каждого из каналов передачи и хранятся в нем до момента получения управляющего сигнала от блока формирования ансамблей хаотических ортогональных последовательностей, свидетельствующего о завершении этапа формирования стохастическим образом хаотических ансамблей ортогональных сигналов. Затем информационный бит каждого из каналов через блок цифровой информации подается в модулятор, где он модулируется расширяющей последовательностью, поступающей от блока формирования ансамблей хаотических ортогональных последовательностей. Сигналы, снимаемые с выхода модулятора каждого канала, одновременно подаются в сумматор блока формирования группового сигнала, где после их объединения и наложения в модуляторе сигнала синхронизации, поступающего с выхода генератора сигналов синхронизации, происходит формирование группового сигнала, спектр которого после переноса в область несущей частоты в блоке фазовой модуляции и усилителе мощности через антенну излучается в эфир. На приемной стороне поступающий сигнал принимается антенной и подвергается предварительной обработке в блоке высокочастотной селекции. С выхода этого блока сигнал одновременно подается в блок обнаружения сигнала синхронизации и блоки корреляционной обработки каждого канала. При этом блок обнаружения сигнала синхронизации совместно с блоком поиска вводят в синхронизм генератор копии сигналов синхронизации, после чего с генератора копий сигналов синхронизации подается управляющий сигнал на блок формирования копий ансамблей хаотических ортогональных сигналов. По окончании процесса формирования стохастическим образом копий хаотических ансамблей ортогональных сигналов в запоминающее устройство записывается результат работы блока формирования копий ансамблей хаотических ортогональных сигналов, после чего копии хаотических ортогональных сигналов подаются с запоминающего устройства на блоки корреляционной обработки каждого из каналов связи и, пройдя через первый вход блока выделения информации, на второй вход которого подается копия сигнала синхронизации с генератора копий сигналов синхронизации, направляются на вход блока приема информации.

Результаты расчетов [7] показывают, что с увеличением элементов n, используемых для вычисления хаотических ансамблей ортогональных многоуровневых сигналов, увеличивается количество M возможных структур данных сигналов по сравнению с количеством сигналов Q, формируемых на основе кодовых словарей де Брейна.

Предлагаемая система передачи информации обладает повышенной структурной скрытностью по сравнению с прототипом, для этого произведем расчет количества всех возможных ансамблей ортогональных сигналов, формируемых данными системами. Для этого будет рассмотрена диагональная положительно определенная симметрическая матрица вида

$$A=\begin{bmatrix} a_{1,1} & a_{1,2} & \dots & a_{1,N} \\ a_{2,1} & a_{2,2} & \dots & a_{2,N} \\ \dots & \dots & \dots & \dots \\ a_{N,1} & a_{N,2} & \dots & a_{N,N} \end{bmatrix}, \quad (1)$$

используемая в рассматриваемой системе связи для CDMA-модуляции передаваемой информации, где $a_{1,1}, a_{2,2}, \dots, a_{N,N}$ - коэффициенты главной диагонали, $(a_{1,1}, a_{2,1}), (a_{1,3}, a_{3,1}), \dots, (a_{N-1,N}, a_{N,N-1})$ - симметричные элементы верхней и нижней диагонали.

В соответствии с описываемым способом [7] при передаче каждого бита должен использоваться новый хаотический ансамбль ортогональных многоуровневых сигналов, поэтому если обеспечить смену хаотических ансамблей ортогональных многоуровневых сигналов с периодом, равным T_0, где $T_0=1/R$ – временной интервал использования хаотического ансамбля ортогональных многоуровневых сигналов, равный длительности бита информации, а R – скорость передачи бита информации, то период повторения хаотического ансамбля ортогональных многоуровневых сигналов будет определяться соотношением:

$$T_M = \frac{M}{R}, \tag{2}$$

где M – количество ансамблей, определяемое матрицей вида (1) в рассматриваемой системе связи и равное

$$M = C_n^k = \frac{(n+k-1)!}{k!(n-1)!},\ k = \frac{N \cdot N - N}{2},\ N = \dim(A) \tag{3}$$

где k – количество элементов матрицы вида (1) размерности N, находящихся выше или ниже главной диагонали, n – диапазон возможных значений коэффициентов матрицы вида (1), N – размерность матрицы вида (1).

Использование в качестве коэффициентов диагональной положительно определенной симметрической матрицы случайных чисел позволяет повысить количество возможных кодовых последовательностей, используемых для передачи последовательности информационного сообщения.

В соответствии с соотношением при условии, что скорость передачи данных R равна 1МБит/с, dim(A) = 256 и диапазон возможных значений n равен 10, то период повторения одного хаотического ансамбля ортогональных многоуровневых сигналов составит $3{,}54 \times 10^{22}$ лет [7]. Это говорит о том, что при передаче одинаковых последовательностей бит генерируемые расширяющие последовательности не будут повторяться в течение длительного времени.

Выводы:

- система передачи информации с использованием ансамблей ортогональных дискретных многоуровневых сигналов позволяет повысить стркутурную скрытность передаваемой информации;
- в предлагаемой системе передачи информации могут использоваться различные ортогональные коды (Уолша, де Брейна и др.), получаемые путем выбора математического представления коэффициентов матрицы вида (1).

Литература

1. Волковец А.И. Теория вероятности и математическая статистика: Практикум для студ. всех спец. БГУИР дневной формы обучения / А.И.Волковец, А.Б.Гуринович. – Мн.: БГУИР, 2003.
2. Генератор функций Попенко – Турко : пат. 753464 СССР : МПК G 06 F 1/02 / В.С. Попенко, С.А. Турко; заявитель В.С. Попенко, С.А. Турко. - № 4799258/24; заявл. 06.03. 90; опубл. 07.08.1992, Бюл. №29.
3. Генераторы хаотических колебаний / Б.И. Шахтарин [и др.]. – М. : Гелиос АРВ, 2007. – 248 с.
4. Дмитриев А.С. Динамический хаос: новые носители информации для систем связи / А.С. Дмитриев, А.И. Панас. – М. : Издательство Физико – математической литературы, 2002. – 252 с.
5. Осмоловский С.А. Стохастические методы защиты информации / С.А. Осмоловский. – М.: Радио и связь, 2003. – 320 с.
6. Скляр Б. Цифровая связь. Теоретические основы и практическое применение / Б. Скляр; пер. с англ. – М. : Издательский дом «Вильямс», 2003. – 1104 с.
7. Способ передачи информации на основе хаотически формируемых ансамблей дискретных многоуровневых сигналов : пат. 2428795 Рос. Федерация : МПК H 04 B 7/216, H 04 J 13/00, H 04 L 9/22 / Жук А.П., Иванов А.С., Голубь Ю.С., Орёл Д.В.; заявитель и патентообладатель Государственное образовательное учреждение высшего профессионального образования Ставропольский государственный университет. – №2010106663/09; заявл. 24.02.2010; опубл. 10.09.2011, Бюл. №25.
8. Способ передачи информации в системах с кодовым разделением каналов и устройство для его осуществления : пат. 2234191 Рос. Федерация : МПК H 04 B 7/216, H 04 L 9/26 / Косякин С.И., Москвитин И.А., Смирнов А.А.; заявитель и патентообладатель Косякин С.И., Москвитин И.А., Смирнов А.А. - № 2001120727/09; заявл. 24.07.2001; опубл. 10.08.2004.
9. Тенденции и перспективы развития мирового рынка инфраструктуры беспроводной связи (2G, 3G, LTE, WiMAX, Wi-Fi), 2012-2017 гг. [Электронный ресурс]. – Режим доступа: http://web.json.ru/files/news/2014-01-30_Wireless%20Infrastructure_MW_RU.pdf, свободный.
10. Состояние, перспективы развития и рыночный потенциал 3G (HSPA и HSPA+) в России и мире [Электронный ресурс]. – Режим доступа: http://www.json.ru/files/reports/2013-12-10_3G_Russia_world_MW_RU.pdf, свободный.

Фараонов А.В.

ФГБОУ ВПО «Санкт-Петербургский государственный университет гражданской авиации»,
faraonov.a@mail.ru

ПРИНЯТИЕ РЕШЕНИЙ В УСЛОВИЯХ НЕОПРЕДЕЛЕННОСТИ КАК СПОСОБ ПОДГОТОВКИ СПЕЦИАЛИСТОВ ТРАНСПОРТНОЙ ЛОГИСТИКИ

КЛЮЧЕВЫЕ СЛОВА

Подготовка специалистов, транспортная логистика, ситуационная модель, имитационное моделирование, оперативное управление, нечеткий ситуационный подход, нечеткие ситуационные сети.

АННОТАЦИЯ

Предлагается методика подготовки специалиста для реакции на непредвиденные ситуации. Разрабатывается имитационная модель принятия оперативных решений на каждом шаге выбора маршрута. На основе теории нечетких множеств решается задача многокритериального выбора маршрутов доставки в условиях неопределенности. Из множества маршрутов специалист выбирает наилучшую альтернативу нового маршрута. Методом анализа иерархий экспертами оцениваются и анализируются возможные последствия действий специалистов.

Процесс принятия управленческих решений во многом основан на интуиции, опыте, менталитете менеджеров и, безусловно, не может быть полностью формализован [1]. Задача поиска оптимальных путей в направленных графах является весьма актуальной в рамках проблематики управления сложными системами и поддержки принятия решений[2]. При этом специфика предметной области обуславливает дополнительные требования, а именно наличие нечеткости (неопределенности) исходной информации и многокритериальный характер оценки процессов управления и принятия решений. В статье [3] рассматриваются эвристические методы принятия решений в различных задачах дискретной оптимизации. Целью для каждой из этих задач является построение т.н. anytime-алгоритмов – т.е. алгоритмов реального времени, которые в каждый определённый момент работы имеют лучшее (на данный момент) решение, при этом пользователь может просматривать эти псевдо-оптимальные решения в режиме реального времени, а последовательность таких решений в пределе даёт оптимальное решение. Построение подобных алгоритмов является одной из задач, которые могут быть объединены общей тематикой с примерным названием «Обучение нечётких систем». Решение практических задач требует поиска новых, более эффективных моделей и алгоритмов поиска как точных, так и приближённых решений.

Ситуационный подход в принятии решений для транспортно-логистической системы предполагает, не только оценить возникшую ситуацию на маршруте, но и определить управляющие решения. Разработка моделей доставки грузов основана на представлении ситуационной модели в виде узлов графа, переходы которого соответствуют управляющим решениям. Такое представление получило название нечёткой ситуационной сети (НСС) [4,5]. Разработка и решение логистических задач маршрутизации, нахождение оптимальных маршрутов возникают в различных областях транспортной логистики: доставка товаров от поставщика к клиенту, доставка сырья, запасных деталей и узлов на производство, курьерская и почтовая доставка, работа грузовых и экспедиторских операторов и т.д. Сформулирован целый класс задач, с ограничениями по времени (DVRPTW – dynamic vehicle routing problems with time windows), постоянно пополняющийся новыми задачами, учитывающими реальные ограничения, возникающие с развитием логистических процессов. Разработка и исследование транспортно-логистической системы с помощью имитационной модели позволяет оценить компетентность специалиста, при принятии решений без вмешательства в работу реальной системы, растянуть или сжать время функционирования логистической системы, понять сложное взаимодействие элементов внутри системы, оценить степень влияния факторов и выявить «узкие места»[4,5].

Ситуационный шаг управления представляется формулой:

$$S_{NET} : S_j \xrightarrow[u_k]{} S1_j , \quad (1)$$

где S_{NET} – выполнение опорного плана S_{NET} ; S_j – текущая ситуация (узел W_i); $S1_j$ – новая ситуация (узел W_j корректировка опорного плана); $\vec{u}_k$ – выбор маршрута в «непредвиденной ситуации»- выбор модели доставки) (рис. 1).

При «возникновении» непредвиденной ситуации (ситуация формулируется в процессе эксперимента) дальнейший маршрут специалисту предлагается определять следующим образом:

- имеется некоторое начальное множество альтернатив (маршрутов) ***X*** среди которых необходимо произвести выбор наилучшей альтернативы нового маршрута, или же необходимо провести ранжирование альтернатив по предпочтению лица принимающего решение (эксперта);

- задана главная цель ***F*** , исходя из которой, будет производиться выбор или ранжирование множества альтернатив ***X***;

Главная цель (главный критерий) разбивается на подцели (частные критерии); с учетом этого, можно добавить третий пункт;

- задано некоторое множество подцелей $f_1,f_2,...,f_n$, учитываемых при выборе или ранжировании альтернатив множества ***X***.

Специалист с учетом главной цели ***F*** и подцелей f_1, $f_2,...,f_n$ выбирает наилучшею альтернативу нового маршрута из множества ***X*** или/и альтернативы по степени уменьшения их значимости. Например, для иерархии на рис.2 производится оценка ее нижнего уровня (вероятные новые маршруты) через второй уровень (частные критерии: длина, время в пути, марка автомобиля, состояние дороги, время суток), который в свою очередь используется для оценивания главного критерия (новый маршрут).

Первым этапом в задачах принятия решений является декомпозиция проблемы через определение ее компонент и отношений между ними, т.е. построение иерархии задачи принятия решений. Общие рекомендации при построении иерархии могут быть такими: основные цели устанавливаются в вершине иерархии; подцели - непосредственно ниже, силы, влияющие на подцели - еще ниже.

Рисунок 1. Принятие решений в условиях неопределенности

На самом нижнем уровне иерархии следует располагать возможные исходы (альтернативы, сценарии и т.д.). Следующим этапом является осуществление попарного сравнения отдельных компонент иерархии (далее просто сравнения). Попарные сравнения — это процесс, согласно которому эксперт сравнивает все пары объектов из некоторого списка по некоторому критерию, указывая каждый раз, более предпочитаемый объект (по этому критерию).

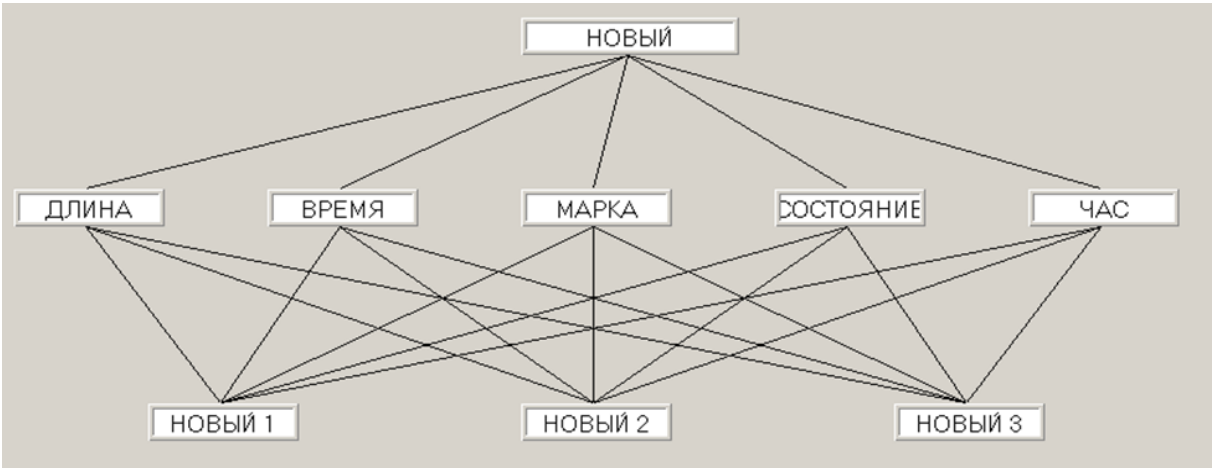


Рисунок 2. Выбор нового маршрута методом анализа иерархий

Для «непредвиденной ситуации» требуется не просто идентифицировать ситуацию и соответствующее ей множество управляющих решений, но и определить рациональные пути достижения целей планирования и оперативного управления выбора маршрута доставки, для чего необходимо определить возможные последствия управляющих решений на несколько шагов вперёд. Задачи оперативного управления выбора маршрута доставки требуют привлечения дополнительных методов, среди которых хорошо себя зарекомендовали методы, основанные на представлении совокупности типовых состояний системы в виде узлов графа, переходы которого соответствуют управляющим решениям. Ситуационная сеть S_{NET} представлена в виде ориентированного графа $S_{NET}=(W,A)$, где W – множество узлов – состояний, а A – множество дуг-переходов между состояниями.

$$S_{NET}=(W,A);\ W=\{\ w_i|i=1,..,N_w\};\ \ A=\{\ a_{i,j}|i=1,..,N_w;\ \ j=1,...,N_w\} \qquad (2)$$

Метод вывода по нечёткой ситуационной сети [2,3] основывается на определении связного подграфа, содержащего некоторое начальное состояние сети w', относительно которого ведётся поиск.

$$S_{NET}'=(W',A'),\ \ W'\subset W, w'\in W';\ A'\{\ a_{i,j}|w_i,w_j\in W\};\qquad (3)$$

Направленным нечетким графом $S_{NET}'=(W',A')$ называется пара множеств, в которой $W'\subset W, w'\in W'$ – множество вершин графа; $A'\{\ a_{i,j}|w_i,w_j\in W'\}$ – нечеткое множество направленных ребер графа, вершина w_i является началом, w_j – концом ребра w_i,w_j; $\mu_A<w_i,w_j>$ – значение функции принадлежности μ_A для ребра w_i,w_j.

Вид подграфа S_{NET}' определяется типом конкретной ситуационной сети:

-для сетей, в дугах переходов которых отсутствует случайная составляющая, S_{NET}' обычно принимает вид цепи (выполнение опорного плана);

-для сетей, учитывающих случайные факторы при переходах, S_{NET}' ищется в виде дерева, соответствующего поливариантному сценарию управления (определяются варианты моделей).

Создание компьютерной модели логистической системы включает такие взаимосвязанные этапы, как содержательная постановка задачи; разработка концептуальной модели; разработка и программная реализация имитационной модели; оценка адекватности модели и точности результатов моделирования; планирование экспериментов; принятие решений. Эти показатели не имеют четко очерченных оптимальных границ. Показатели дают возможность количественно зафиксировать тенденции в подготовке специалиста, определить тип поведения специалиста, и соответственно разработать как коллективную, так и индивидуальную методику подготовки специалистов. Каждое действие специалиста относится к одному из следующих классов:Q_1 — правильно, своевременно выполненное действие; Q_2 — невыполненные действия; Q_3 — неправильные действия; Q_4 — действия, выполненные с опозданием; Q_5 — действия, выполненные ранее необходимого; Q_6 — излишние действия; Q_7 — неоптимальные действия. Таким образом, производится оценка каждого действия (в том числе и невыполненного,

необходимого).Методика определяется следующей последовательностью действий. При возникновении «непредвиденной ситуации» в узле W_j – дальнейший маршрут определяется следующим образом[4-6]. Определяется множество альтернативных (возможных) маршрутов доставки $\mu(j)= S_{NET} = \{ S_{NET\ 1}, S_{NET\ 2}, ... S_{NET\ i}, ... S_{NET\ n}\}$. Каждый маршрут характеризуется параметрами (критериями), - $Y = \{y_1, y_2, ...y_j, ...y_m\}$(например пропускная способность, расстояние, время доставки). Выбор модели доставки грузов μ_{ij} отражает уровень соответствия *i*-го маршрута доставки требованиям по *j*-му параметру ($\mu_{ij} \in [0;1]; i=1,m; j=1,n$).

1. Модель максиминной свертки (ММС). Наилучшим считается маршрут при минимальных недостатках по всем параметрам.

2. Модель абсолютного решения (МАР). Задается минимально допустимое значение μ_{ij}min для каждого параметра *Y*. Выбирается маршрут, с параметрами не хуже заданных.

3. Модель основного параметра (МОП). Решение производится по шагам. На каждом шаге выбирается основной параметр, и поиск наилучшего решения ведется только по нему.

4. Модель компромиссного параметра (МКП). Логист выбирает параметры по уровню их важности и определяет влияние каждого параметра на выбор маршрута.

5. Модель эталонного сравнения (МЭС). Имеется оптимальное решение на основе компромиссной модели, при этом учитываются ограничения на значения параметров. Определяется эталонный вариант маршрута доставки груза X_0. Параметры этого варианта принимаются как минимально допустимые значения параметров μ_{ij}min. Каждый вариант маршрута множества *X* сравнивается с эталонным X_0.

Определение значений функции принадлежности и принятие решений по выбору модели доставки грузов на основе нечетких множеств в среде FuzzyTECH Наиболее перспективными методами принятия решений в слабоструктурированных проблемных областях являются, методы, основанные на теории нечетких множеств. Одним из таких методов является метод анализа альтернатив (принятие решений в условиях неопределенности) [4-6]. Имеется множество альтернатив маршрутов доставки $\mu(j)=X = \{x_1, x_2, ...x_i, ...x_n\}$,тогда для критерия $Y = \{y_1, y_2, ...y_j, ...y_m\}$ может быть рассмотрено нечеткое множество

$$Y = \{\mu_y(x_1) / x_1, \mu_y(x_2) / x_2, ..., \mu_y(x_m) / x_m\}, \quad (4)$$

где $\mu_y(x_i)$ [0, 1]– оценка альтернативы x_iпо критерию *Y*и характеризует степень соответствия альтернативы критерию*Y*.

Если имеется n критериев: Y_1, Y_2, Y_3, то лучшей считается альтернатива, удовлетворяющая и критерию Y_1, и Y_2, и …, и Y_n. Тогда правило для выбора наилучшей альтернативы может быть записано в виде пересечения соответствующих нечетких множеств

$$D = Y_1 \cap Y_2 \cap ... \cap Y_n. \quad (5)$$

Операции пересечения нечетких множеств соответствует операция минимума, выполняемая над их функциями принадлежности

$$\mu_D(a_j) = \min_{i=1,n} \mu_{Y_i}(x_j) \quad j=1,m. \quad (6)$$

В качестве лучшей выбирается альтернатива *x**, имеющая наибольшее значение функции принадлежности [14]

$$\mu_D(x^*) = \max_{j=1,n} \mu_D(x_j) \quad (7)$$

Возможны два варианта определения числовых значений функции принадлежности:

- непосредственное задание функции принадлежности μ_Y экспертом в интервале [0;1];

- построение функции принадлежности μ_Y на основе интервальных оценок [6]. Если имеется интервал $[h^*, h^0]$ значений критерия *h*, который соответствует понятию «хороший» объект, то граничные значения интервала имеют следующую интерпретацию. Когда h^a – результат измерения значения характеристики *h* для объекта *a*, то h^* является границей идеальной области, т.е., если $h^a \geq h^*$, объект следует признать идеально соответствующим понятию «хороший». Возможность такого утверждения $\mu(u) = 1{,}0$ (*u* – субъективное событие, заключающееся в том, что объект, с точки зрения эксперта, находится в состоянии «хороший»). Если $h^a \leq h^0$, ситуация интерпретируется так: возможность того, что объект «хороший», $\mu_u = 0$. Очевидно, что при $h^0 < h^a < h^*$ соответствующие возможности имеют значения $0 < \mu(u) < 1{,}0$. Очевидно, что с приближением значения h^aк границе h^* возможность признания *a* «хорошим» объектом линейно возрастает [4-6].При этом для определения функции принадлежности используется формула

$$\mu(u)=\begin{cases}0, \text{если } h^a \le h^0;\\ \frac{h^a-h^0}{h^*-h^0}, \text{если } h^0 < h^a < h^*;\\ 1, \text{если } h^a \ge h^*.\end{cases} \quad (8)$$

Программа FuzzyTECH [6] позволяет оперировать лингвистическими переменными и создавать для них продукционные правила вывода. В интерактивном режиме программы FuzzyTECH можно не только видеть значение конечного результата $\mu(j)$, но и следить за промежуточными операциями. Данная возможность необходима при внесении новых переменных и правил в процедуру определения альтернативного маршрута. Демонстрация промежуточных результатов контролирует перенос правил нечёткого вывода в программу. Например, для определения альтернативного маршрута от склада на ул. Политехническая, 9 [9] запишем лингвистические переменные и создадим для них продукционные правила вывода (рис. 1). Определяем μ (Marshrut) с тремя термами: «high», «middle» и «low» при трех входных переменных пропускная способность(Psposobnost), расстояние(Rastoynie) и время доставки (Time),получим набор правил. Логист для каждой переменной (рис. 2) вводит вручную степень принадлежности к соответствующим термам. Полученные значения обрабатываются в соответствии с правилами, и в таблице в правой части рис. (2) отображается истинность правила в виде чёрного прямоугольника. Прямоугольник, закрашенный полностью, показывает на истинность, равную 1, прямоугольник не закрашенный – на истинность, равную 0. Промежуточным значениям соответствует прямоугольник, закрашенный частично. Отображение истинности правила позволяет следить за ошибками при переносе правил в программу, а также за влиянием каждого из правил на конечный результат.

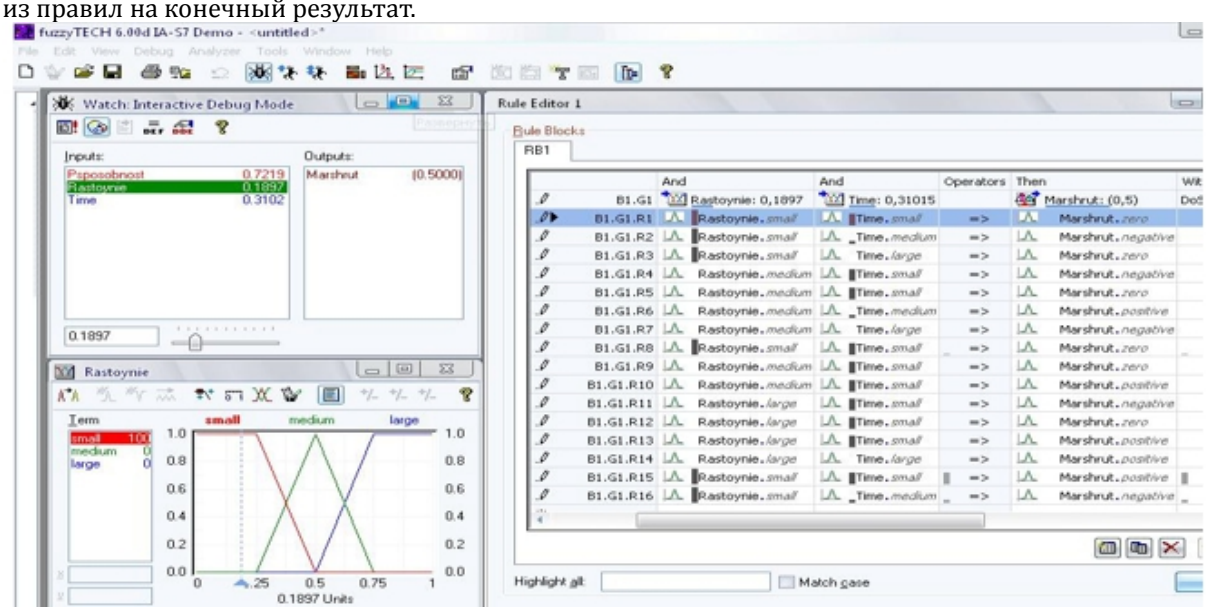


Рисунок.3. Интерактивный режим программы FuzzyTech

Компания «Норма Измерительные Системы»[7] разрабатывает, производит и поставляет приборы коммерческого учета энергоресурсов. Деятельность специалистов компании направлена на решение простых и понятных задач, результат решения которых делает работу с компанией всегда комфортной и коммерчески привлекательной. По этим причинам компании необходимы специалисты оперативно решающие возникающие ситуации (в процессе реализации проекта). Создание компьютерной модели логистической системы включает такие взаимосвязанные этапы, как содержательная постановка задачи; разработка концептуальной модели; разработка и программная реализация имитационной модели; оценка адекватности модели и точности результатов моделирования; планирование экспериментов; принятие решений. Эти показатели не имеют четко очерченных оптимальных границ. Показатели дают возможность количественно зафиксировать тенденции в подготовке специалиста, определить тип поведения специалиста, и соответственно разработать как коллективную, так и индивидуальную методику подготовки специалистов. Специалист проводит серии экспериментов по разработке алгоритмов, составлению имитационных моделей в различных средах:

-в среде «BusinessMap»(«Деловая карта»). Программа Деловая карта предназначена для включения в базы данных технологий пространственных обработок и решения задач транспортной логистики - расчет оптимальных маршрутов для обработки заказов, калькуляция маршрутов, задач пространственных сортировок и пр. Областью применения программы являются базы данных и приложения, включая базы данных на ACCESS, EXCEL, FOXPRO, DBASE, PARADOX, 1C[4-6,8].

-в среде AnyLogic**.** Специалист обучается оценивать возникшую ситуацию на маршруте и разрабатывает управляющие решения. Система AnyLogic [4-6,9] поддерживает три технологии создания имитационных моделей: процессно-ориентированный (дискретно-событийный), системно динамический и агентный, а также любую их комбинацию. Графический интерфейс AnyLogic, инструменты и библиотеки позволяют быстро создавать модели для широко спектра задач от моделирования производства, логистики, бизнес-процессов до стратегических моделей развития компании и рынков. AnyLogic стал корпоративным стандартом на бизнес-моделирование во многих транснациональных компаниях, широко используется в образовании.

-в среде ExtendSim8 [4-6,10]. Специалист обучается созданию и эксплуатации системы безопасности аэропорта. Инструмент имитационного моделирования нового поколения, это расширение продукта Extend® фирмы ImagineThat, основанный на результатах, полученных в теории моделирования и в информационных технологиях за последнее десятилетие, поддерживает на единой платформе существующие подходы дискретно-событийного и непрерывного моделирования (блок-схемы процессов, системную динамику, агентное моделирование, карты состояний, системы уравнений и т.д.). Объектно-ориентированный подход, предлагаемый ExtendSim 8, облегчает итеративное поэтапное построение больших моделей.ExtendSim 8-модели создаются из заранее подготовленных блоков.

При сравнении объектов эксперт использует только качественные характеристики, соответствующие им количественные значения. Для получения результата адекватного ситуации, в которой принимается решение, необходимо, чтобы в матрицах попарных сравнений достигался требуемый уровень согласованности данных. Для практических задач, вполне достаточно, если согласованность суждений эксперта, находится в пределах шкалы 1-9. В [6] рассматриваются различные варианты оценки компетентности экспертов. Для любого упорядочения экспертов строится соответствующая матрица A . Далее, пусть - τ_1 и τ_2 два упорядочения, а A' и A'' две соответствующие им матрицы предпочтений, тогда расстояние между упорядочениями τ_1 и τ_2 определяется по следующей формуле:

$$d(\tau_1,\tau_2)=\frac{1}{2}\sum_{i.j\in 1.n}\left|\alpha'_{i.j}-\alpha''_{i.j}\right|,$$

где $\alpha'_{ij},\alpha''_{ij}$ – элементы матриц упорядочений соответственно A' и A''. Пусть $\tau_1,\dots,\tau_m$ - упорядочения, полученные различными экспертами. Тогда точка

$$\tau_{med}=\min_i\sum_{i\in 1,m}d(\tau_i,\tau_{med})$$

называется медианой множества точек $\tau_1,\dots,\tau_m$, а точка

$$\tau_{mean}=\min_i\sum_{i=1,m}d(\tau_i,\tau_{mean})^2$$

является средним значением множества точек $\tau_1,\dots,\tau_m$.

Таким образом:

- Медиана и среднее значение определяют согласованные упорядочения для множества исходных упорядочений различных экспертов.

- Медиана учитывает мнение большинства экспертов, тогда как среднее значение может посчитать преимущество большинства не вполне убедительным и тем самым провозгласить равноценность.

- Среднее значение всегда единственно, тогда как медиан может от одной до ***m***, где ***m*** - количество экспертов (упорядочений).

Разработана имитационная модель принятия оперативных решений при возникновении непредвиденной ситуации на маршруте, корректировке опорного плана и выборе нового маршрута доставки, обучении необходимым квалификационным навыкам и принятия решений на основе имитационного моделирования транспортно-логистических систем, моделируемых в среде

AnyLogic, ExtendSim8 и «Business Map»[8-10]. Алгоритм состоит из взаимосвязанных этапов, таких как содержательная постановка задачи, разработка концептуальной модели; разработка и программная реализация имитационной модели, оценка адекватности модели и точности результатов моделирования, планирование экспериментов; принятие решений, что позволяет проверить и оценить квалификацию специалистов. Методом анализа иерархий экспертами оцениваются и анализируются возможные последствия действий специалистов, квалификацию, компетентность специалистов без вмешательства в работу реальной системы.

Литература

1. Ерусалимский В. М., Иода Е. В.. Нечеткие соответствия как способ принятия решений в условиях неопределенности // Социально-экономические явления и процессы№ 3 (015), 2009.С. 13-15.
2. Голубев И. В. Поиск оптимальных путей в направленных нечетких графах. СМПО НИЦ ВА ВПВО ВС РФ. 2006.
3. Мельников Б. Ф. Применение мультиэвристического подхода для случайной генерации графа с заданным вектором степеней / Б. Ф. Мельников, Е. Ф. Сайфуллина // Известия высших учебных заведений. Поволжский регион. Физико-математические науки. – 2013. – № 3 (27). – С. 70–83.
4. Фараонов А.В.Ситуационные центры – инструмент подготовки специалистов транспортной логистики//International Journal of Open Information Technologies ISSN: 2307-8162 Vol. 3, No. 4, 2015,стр. 1-7.
5. Фараонов А.В. Логистика выбора маршрута доставки от контейнерной площадки морского порта на основе нечёткой ситуационной сети// Имитационное и комплексное моделирование морской техники и морских транспортных систем. Сборник докладов третьей международной научно – практической конференции ИКМ МТМТС 2015(ISSN 978-5-902241-28-7), стр.148-152.
6. ФараоновА.В. Ситуационная модель выбора маршрута доставки.//Журнал «Прикладная информатика», №2(44), 2013.С.113-126.
7. Сайт: http://www.//normais.ru//.
8. Официальный сайт разработчика ООО «Фирма «ИНГИТ», «Деловая карта»- www.ingit.ru.
9. Сайт: http://www.anylogic.ru/.
10. Сайт: http://www.extendsim.com/index.html, http://imaginethatinc.com/pages/demo.html

Джамбеков А.М.

Астраханский государственный технический университет, г. Астрахань, аспирант кафедры автоматики и управления, azamat-121@mail.ru

РАЗРАБОТКА СИСТЕМЫ УПРАВЛЕНИЯ ПРОЦЕССОМ КАТАЛИТИЧЕСКОГО РИФОРМИНГА НА БАЗЕ НЕЧЕТКОЙ ЛОГИКИ

КЛЮЧЕВЫЕ СЛОВА

Стабилизационная колонна, лингвистическая переменная, нечеткий регулятор, база знаний, продукционное правило.

АННОТАЦИЯ

Рассмотрены особенности применения нечеткой логики при разработке системы управления процессом каталитического риформинга. Приведено обоснование необходимости применения аппарата нечеткой логики при управлении процессом риформинга как инструментария, позволяющего формализовать качественную информацию о процессе. Предложен подход к оценке диапазонов изменения лингвистических переменных и накоплении продукционных правил нечеткого регулятора. Сравнительная оценка показателей качества полученных переходных процессов в системах с нечетким и ПИД-регулятором отражает значительные преимущества нечеткого регулятора при внедрении в автоматизированную систему управления процессом каталитического риформинга.

Введение

В настоящее время одним из важнейших процессов нефтепереработки и нефтехимии является каталитический риформинг. Достаточно сложной является задача управления установкой каталитического риформинга [1]. Главным образом, это связано с затруднениями при получении адекватного математического описания процесса каталитического риформинга [2]. Основной проблемой при математическом моделировании процесса каталитического риформинга является наличие большого количества информации о процессе, которую нельзя формализовать традиционным математическим аппаратом (информация о компонентном составе газосырьевой смеси, состоянии оборудования и пр.) [3].

Математическое описание на основе теории нечетких множеств [4,5] позволяет представить качественную информацию в формализованном виде, а также получить нечеткие модели объектов [6]. Представляется возможным построение нечеткой системы регулирования на основе экспертной информации, сформулированной в виде правил типа: «если давление высоко и температура мала или очень мала, то управляющее воздействие большое положительное» [7,8].

Поэтому, актуальной является проблема получения эффективной модели управления процессом каталитического риформинга, содержащей количественную и качественную информацию о процессе и позволяющей повысить производительность установки каталитического риформинга и улучшить качественные характеристики выпускаемой продукции.

Объектом исследования в данной работе является стабилизационная колонна установки каталитического риформинга. Рассмотрим вопросы разработки системы регулирования температуры стабилизационной колонны на основе экспертной информации.

Целью данной работы является анализ существующих моделей управления процессом каталитического риформинга, проведение возможной их классификации и разработка системы нечеткого управления процессом стабилизации катализата.

Краткое описание технологической схемы

Нестабильный катализат поступает в блок стабилизации катализата из сепаратора предыдущего блока установки риформинга (блока каталитического риформинга), направляется в стабилизационную колонну К-2 через теплообменник Т-6 (трубное пространство), обогревается стабильным катализатом из колонны К-2 (рис.1) [9].

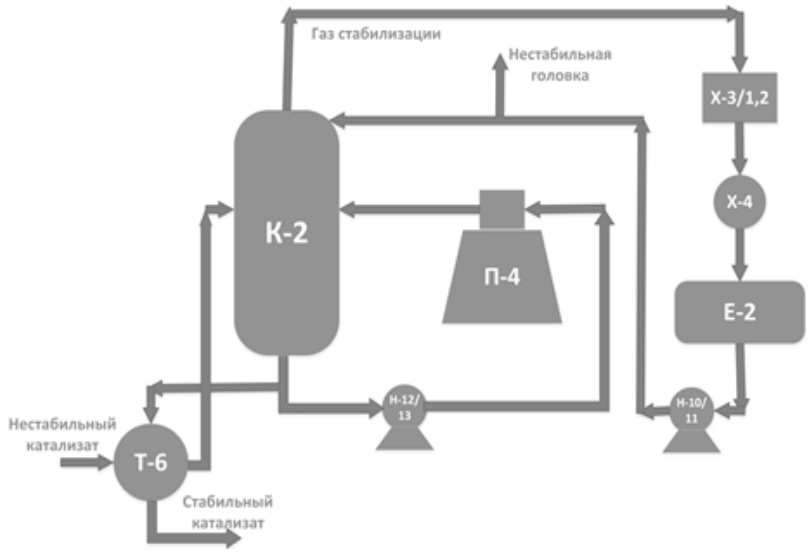


Рис. 1. Упрощенная технологическая схема блока стабилизации катализата

В поток нестабильного катализата перед теплообменником Т-6 сбрасывается газовый конденсат из сепаратора С-7.

В колонне К-2 происходит стабилизация катализата. С верха стабилизационной колонны К-2 выводятся газ стабилизации, нестабильная головка стабилизации, которые после охлаждения и частичной конденсации в аппаратах воздушного охлаждения ХК-3/1, ХК-3/2 и водяном ХК-4 до температуры не выше плюс 45 ^{0}С сепарируются в емкости орошения Е-2. Газ стабилизации из емкости орошения Е-2 сбрасывается в топливную сеть комплекса.

Жидкая фаза из емкости Е-2 возвращается на верхнюю тарелку колонны К-2 насосами Н-10, Н-11 (1 рабочий + 1 резервный) в качестве орошения.

Балансовое количество головки стабилизации выводится насосами Н-10, Н-11 (1 рабочий + 1 резервный) из Е-2, контролируется расходомером и передается на комбинированную установку комплекса на блок очистки и получения сжиженных газов. Температура низа колонны К-2 поддерживается за счет циркуляции стабильного катализата через трубчатую печь П-4 насосами Н-12, Н-13 (1 рабочий + 1 резервный).

Разработка системы нечеткого управления процессом стабилизации катализата риформинга

Структура системы управления температурой низа стабилизационной колонны К-2 с нечетким регулятором (НР) приведена на рис.2.

Рис. 2. Система управления с цифровым НР

Для оценки диапазонов изменения значений ошибки регулирования температуры θ, первой и второй производных ошибки $\dot{\theta}$, $\ddot{\theta}$ и управляющего воздействия на объект m воспользуемся результатами работы одноконтурной АСР с ПИД-регулятором, который позволяет получить удовлетворительные показатели качества переходного процесса (рис.3).

Графики изменения ошибки регулирования температуры θ, первой производной ошибки $\dot{\theta}$, второй производной ошибки $\ddot{\theta}$ и управляющего воздействия на объект m представлены на рис. 4.

Моделирование НР произведем с использованием пакета нечеткой логики (Fuzzy Logic Toolbox) интерактивной системы MATLAB. Выбрана система нечеткого вывода Мамдани,

дефаззификация лингвистических переменных (ЛП) производится методом центра тяжести [10].

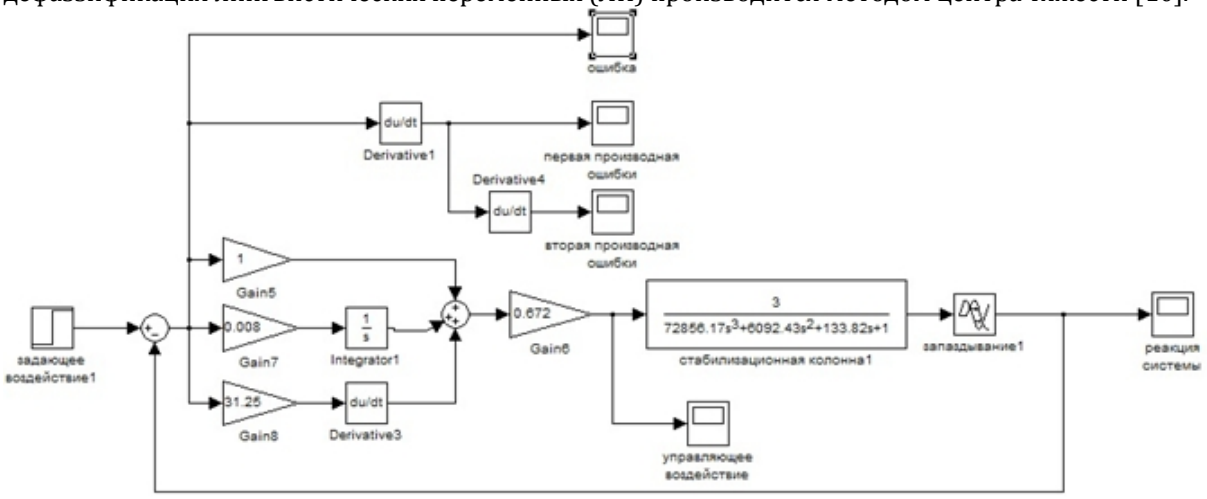


Рис.3. Структурная схема одноконтурной АСР температуры с ПИД-регулятором

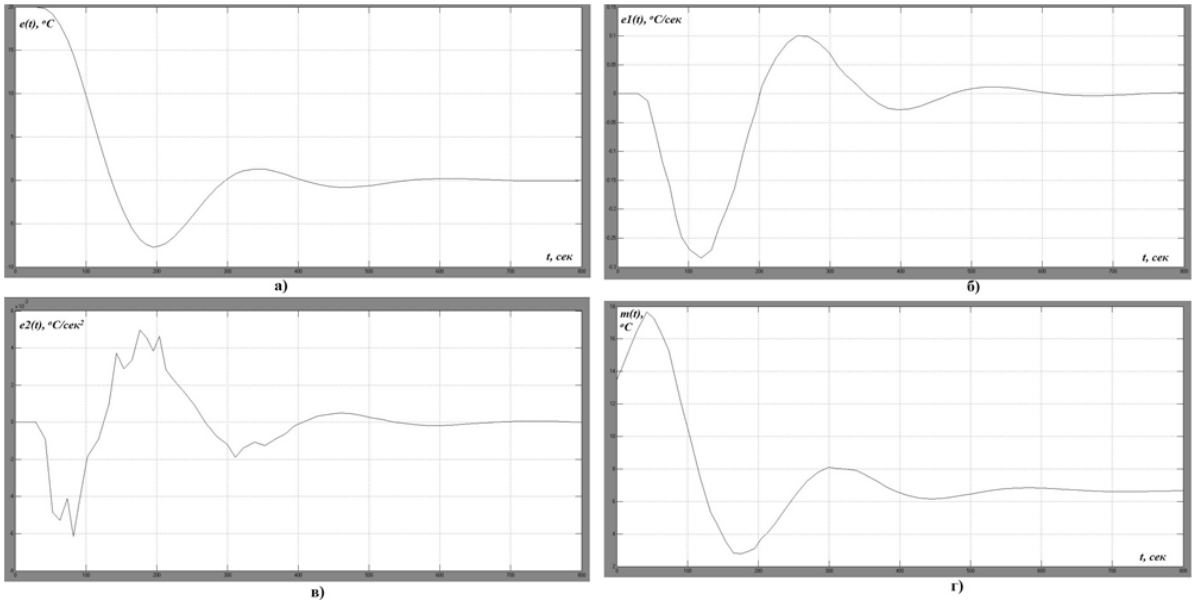


Рис.4. Изменения переменных НР: а) ошибки регулирования; б) первой производной ошибки; в) второй производной ошибки; г) управляющего воздействия на объект

Примем число термов, с помощью которых оцениваются ЛП (входные и выходной параметры НР) ошибка регулирования температуры θ, скорость изменения (первая производная) ошибки $\dot{\theta}$, ускорение (вторая производная) ошибки $\ddot{\theta}$, управляющее воздействие на объект m, равным 3.Для входных и выходной переменных были использованы треугольные ФП.

Отобразим диапазоны $\left[\theta_{\min};\theta_{\max}\right]=[-10;\ 10]$, $\left[\dot{\theta}_{\min};\dot{\theta}_{\max}\right]=[-0,15;\ 0,15]$, $\left[\ddot{\theta}_{\min},\ddot{\theta}_{\max}\right]=[-0,006;0,006]$ и $\left[m_{\min},m_{\max}\right]=[3;\ 11]$ изменения входных и выходного параметров на единое универсальное множество $U=[0,1]$. На множестве $U=[0,1]$ зададим три нечетких подмножества.

Имеем следующие аналитические выражения для каждой ЛП:

$$\begin{cases} \mu_1(u)=1-u,\ u\in[0,\ 1];\ \mu_2(u)=u,\ u\in[0,\ 1]; \\ \mu_3(u)=\begin{cases} 2u,\ u\in[0,1/2] \\ 2(1-u),\ u\in[1/2,1] \end{cases} \end{cases} \quad (1)$$

При поступлении на нечеткий регулятор (НР) значений входных переменных θ^*, $\dot{\theta}^*$ и $\ddot{\theta}^*$ осуществляется расчет величин u_1^*, u_2^* и u_3^* по формулам [11]:

$$\begin{cases} u_1^* = (\theta^* - \theta_{\min})/(\theta_{\max} - \theta_{\min}) = (\theta^* + 10)/20 \\ u_2^* = (\dot{\theta}^* - \dot{\theta}_{\min})/(\dot{\theta}_{\max} - \dot{\theta}_{\min}) = (\dot{\theta}^* + 0,15)/0,3 \\ u_3^* = (\ddot{\theta}^* - \ddot{\theta}_{\min})/(\ddot{\theta}_{\max} - \ddot{\theta}_{\min}) = (\ddot{\theta}^* + 0,006)/0,012 \end{cases} \quad (2)$$

и ФП $\mu^j(u)$, $j=\overline{1,3}$, по формулам (2).

Сформируем лингвистическое правило управления (рабочее правило) НР в виде:

Если $(\theta^* = a_1^j)$ ***и*** $(\dot{\theta}^* = a_2^j)$ ***и*** $(\ddot{\theta}^* = a_3^j)$, ***то*** $(m^* = a_c^j)$, $j=\overline{1,3}$, где a_1^j, a_2^j и a_3^j – лингвистические оценки ошибки, скорости изменения (первой производной) ошибки и второй производной ошибки, рассматриваемые как нечеткие множества, определенные на универсальном множестве, $j=\overline{1,3}$; a_c^j - лингвистические оценки управляющего воздействия на объект, выбираемые из терм-множества переменной *m*. Лингвистические оценки выбираются из терм-множества ЛП θ^*, $\dot{\theta}^*$, $\ddot{\theta}^*$ и m^*:

$$a_i^j \in \{\textit{отрицательная}\,(1), \textit{положительная}\,(2), \textit{близкая к нулю} - \textit{нулевая}\,(3)\}$$

Другими словами, все сигналы (определенные выше ЛП) в системе автоматического управления характеризуются как отрицательные (*j=1*), положительные (*j=2*) или близкие к нулю (*j=3*).

Предложено получение данных правил управления на основе оценки конкретных значений ошибки регулирования температуры *e*, скорости изменения ошибки *e1*, ускорения ошибки *e2* и управляющего воздействия на объект *m* путем фиксации значений данных функций в определенные моменты времени $t_0 \in [0;1000]$ в среде MATLAB (рис. 5). Затем полученным значениям в зависимости от попадания в тот или иной интервал изменения лингвистических переменных НР присваиваются оценки из терм-множества ЛП:

$$\theta^* \in \{\textit{отрицательная}\,(N), \textit{положительная}\,(P), \textit{нулевая}\,(Z)\}$$
$$\dot{\theta}^* \in \{\textit{отрицательная}\,(N), \textit{положительная}\,(P), \textit{нулевая}\,(Z)\}$$
$$\ddot{\theta}^* \in \{\textit{отрицательная}\,(N), \textit{положительная}\,(P), \textit{нулевая}\,(Z)\}$$
$$m^* \in \{\textit{отрицательное}\,(N), \textit{положительное}\,(P), \textit{нулевое}\,(Z)\}$$

ФП для ошибки регулирования температуры θ, первой производной ошибки $\dot{\theta}$, второй производной ошибки $\ddot{\theta}$ и управляющего воздействия на объект m представлены соответственно на рис. 6.

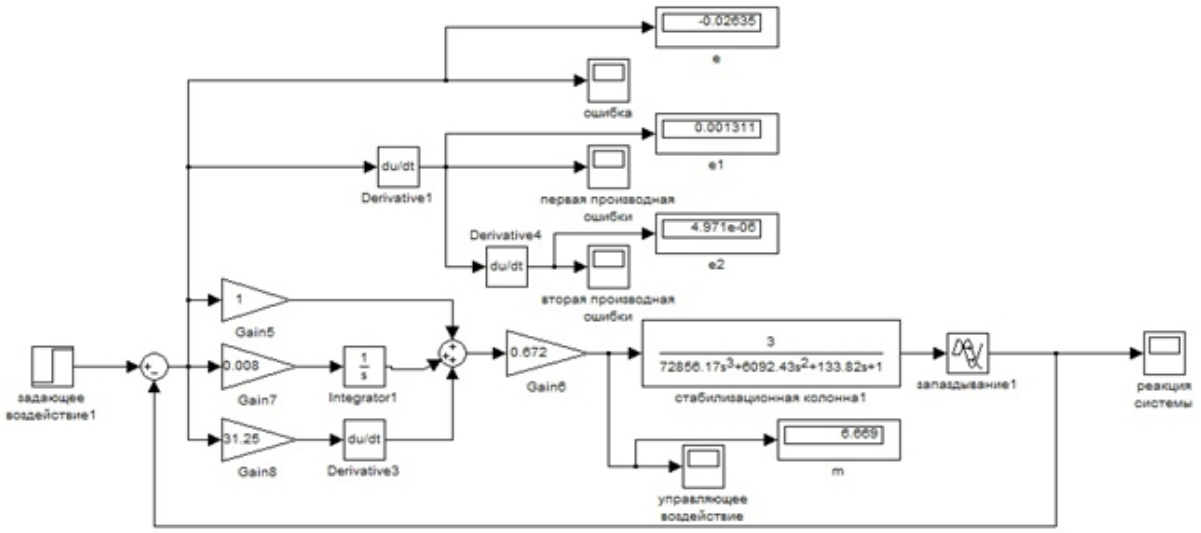


Рис. 5. Оценка значений параметров НР

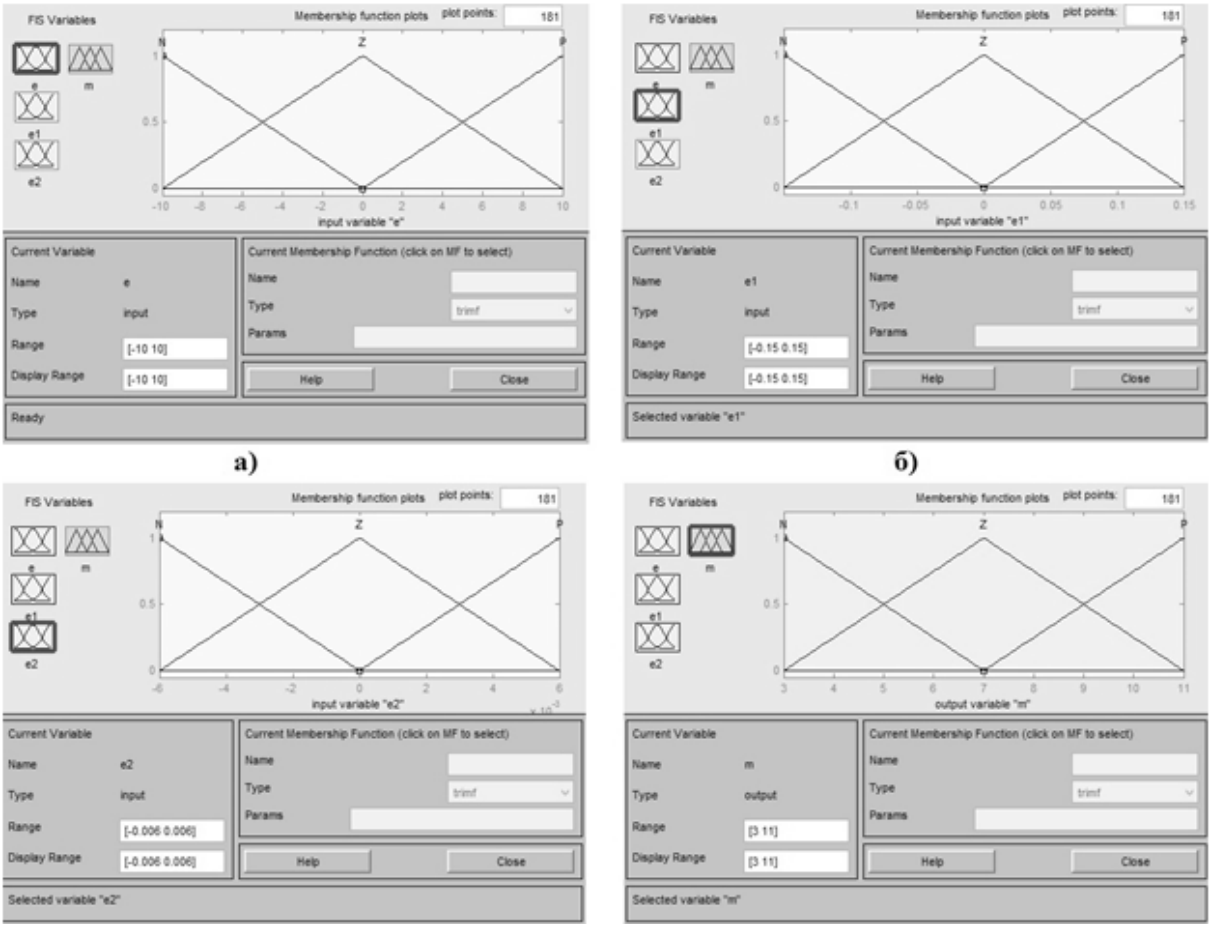


Рис.6. ФП лингвистических переменных НР: а) ошибки регулирования; б) первой производной ошибки; в) второй производной ошибки; г) управляющего воздействия на объект

В результате полученных оценок была сформирована база знаний НР (рис.7).

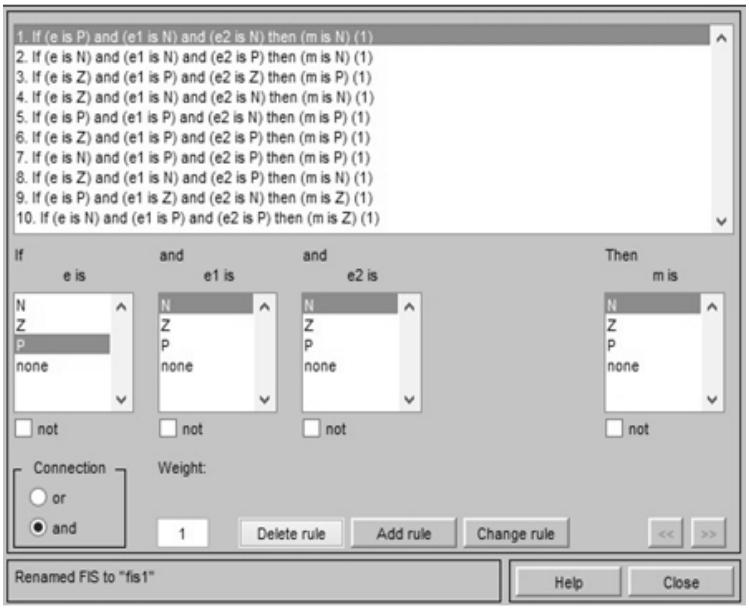


Рис. 7. База знаний НР

В результате проведенных оценок, формирующих вышеприведенную базу знаний, получаем систему регулирования температуры низа колонны К-2 с НР (рис.8). При сравнении переходных процессов в системах регулирования температурой низа колонны К-2 с НР и ПИД-

регулятором (рис.9) получены оценки показателей качества процессов (таб.1).

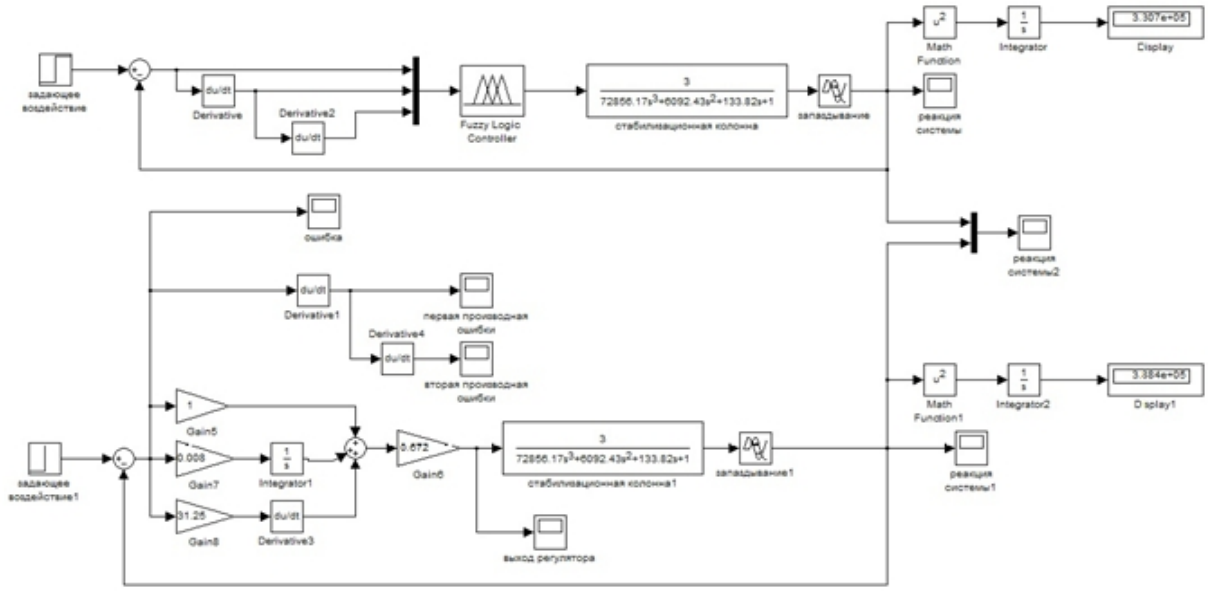


Рис. 8. Структурные схемы систем регулирования температурой низа колонны К-2

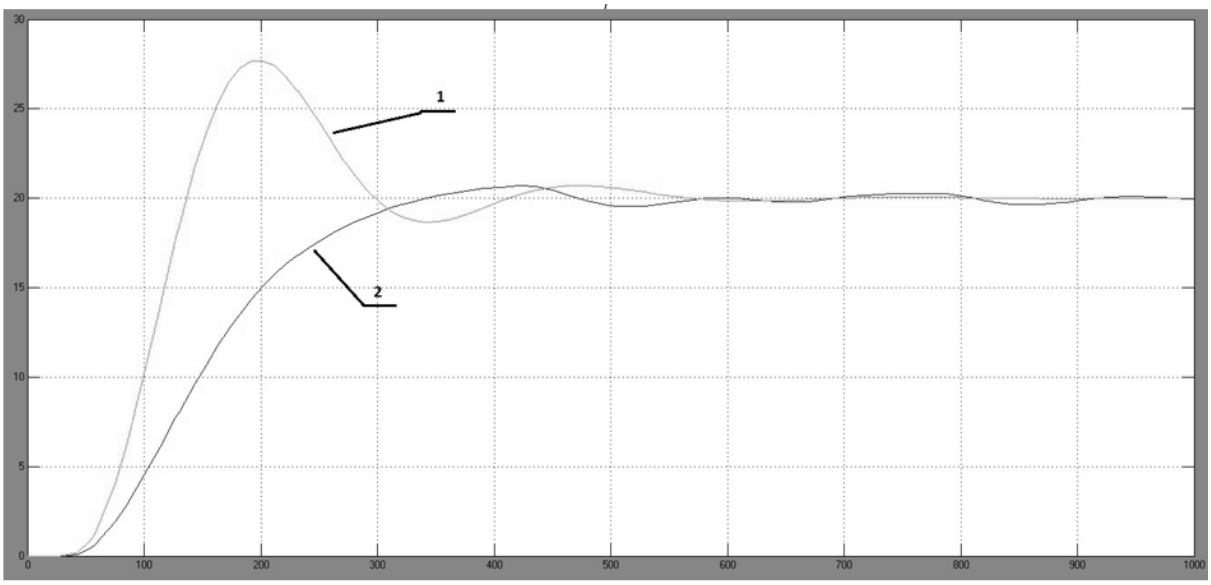


Рис. 9. Переходные процессы в системах: 1 – с ПИД-регулятором; 2 – с НР

Таблица 1. Сравнительная оценка показателей качества процессов

Показатель	Система с ПИД-регулятором	Система с НР
Перерегулирование σ, [°C]	7,5	0,85
Степень затухания ψ	(27,5-20,85)/27,5= 0,2418	(20,85-20,25)/20,85= 0, 0288
Квадратичный интегральный критерий качества I_2	$3{,}884\times10^5$	$3{,}307\times10^5$

Заключение

Таким образом, была получена система автоматического регулирования температуры низа стабилизационной колонны К-2 с нечетким регулятором. По характеристикам переходного процесса данная система превосходит систему регулирования с ПИД-регулятором. В частности, значение квадратичного интегрального критерия качества в системе с НР превосходит в примерно 1,17 раза значение интегрального критерия в системе с ПИД-регулятором. Величина

перерегулирования в системе с НР примерно в 9 раз ниже соответствующей величины в системе с ПИД-регулятором. Таким образом, полученная система управления с нечетким регулятором позволяет получать переходные процессы с высокими показателями качества и может быть применена для разработки комплексной системы управления процессом каталитического риформинга.

Литература

1. Щербатов И.А. Управление сложными слабоформализуемыми многокомпонентными системами: монография. - Ростов н/Д: Изд-во ЮНЦ РАН, 2015. -268 с.
2. Щербатов И.А. Математическое моделирование сложных многокомпонентных систем / И.А. Щербатов, И.О. Проталинский // Вестник Тамбовского государственного технического университета. - 2014. - Т. 20. - № 1. - С. 17-26.
3. Щербатов И.А. Сложные слабоформализуемые многокомпонентные технические системы / И.А. Щербатов, О.М. Проталинский // Управление большими системами: сборник трудов. - 2013. - № 45. - С. 30-46.
4. Щербатов И.А. Система поддержки принятия решений для операторов слабоформализуемых ТП / И.А. Щербатов, О.М. Проталинский // Автоматизация в промышленности. - 2009. - № 7. - С. 41.
5. Проталинский О.М. Гибридная модель каталитического реактора процесса Клауса / О.М. Проталинский, Ю.И. Мичуров, И.А. Щербатов // Известия вузов. Северо-Кавказский. регион. Техн. науки. - 2005. - Прил. № 2. - С. 23-34.
6. Проталинский О.М. Гибридный метод обучения нейронных сетей для классификации каталитической стадии процесса Клауса / О.М. Проталинский, И.А. Щербатов, И.О. Беляев // Вестник Саратовского государственного технического университета. - 2010. - Т. 4. - № 2. - С. 38-43.
7. Щербатов И.А. Снижение объемов промышленных выбросов крупнотоннажных установок с использованием экспертной информации / И.А. Щербатов // Вестник Астраханского государственного технического университета. Сер. Управление, вычислительная техника и информатика. - 2009. - № 1. - С. 121-125.
8. Проталинский О.М. Оптимальное управление технологическим процессом Клауса в условиях неопределенности / О.М. Проталинский, А.Н. Савельев, И.А Щербатов // Известия высших учебных заведений. Северо-Кавказский регион. Сер. Технические науки. - 2006. - № Спец. вып. - С. 19а-25.
9. Кондрашева Н.К. Технологические расчеты и теория каталитического риформинга бензина: Учебное пособие / Н.К. Кондрашева, Д.О. Кондрашев, К.Д. Абдульминев. - Уфа: ООО «Монография», 2008. - 160 с.
10. Штовба С.Д. Проектирование нечетких систем средствами Matlab / С.Д. Штовба. - М.: Горячая линия - Телеком, 2007. - 288 с.
11. Гостев В.И. Проектирование нечетких регуляторов для систем автоматического управления / В.И. Гостев. – СПб.: БХВ-Петербург, 2011. - 416 с.

Коваленченко А.С.

РЭУ им. Плеханова, г. Москва, студентка факультета математической экономики, статистики и информатики, snowall@rambler.ru

К ВОПРОСУ ОБ ОЦЕНКЕ ЭФФЕКТИВНОСТИ ФУНКЦИОНИРОВАНИЯ БЕСПРОВОДНЫХ СЕНСОРНЫХ СЕТЕЙ

КЛЮЧЕВЫЕ СЛОВА

Беспроводные сенсорные сети, методики оценки эффективности функционирования сенсорных сетей, критерии оценки сенсорных сетей, комплексная методика оценки эффективности функционирования.

АННОТАЦИЯ

В статье обсуждаются основные параметры, характеризующие беспроводные сенсорные сети. Приведены недостатки методик оценки эффективности, использующих единичные характеристики. Обоснована необходимость комплексной оценки работы беспроводной сенсорной сети.

Актуальность беспроводных сенсорных сетей обусловлена концепцией интеллектуализации таких объектов как дом, офис и производственные помещения, где человек проводит до 90% своего времени, а также с концепцией создания кибернетических производств (полностью оснащенных роботами), первоочередной задачей которых является внедрение беспроводных технологий на уровне АСУ ТП.

Одним из первых прообразов таких сетей можно считать систему СОСУС, предназначенную для обнаружения и идентификации подводных лодок. В середине 1990-х годов технологии беспроводных сенсорных сетей стали активно развиваться, в начале 2000-х годов развитие микроэлектроники позволило производить для таких устройств достаточно дешёвую элементную базу. Беспроводные сети начала 2010-х годов в основном базируются на стандарте ZigBee.

Рисунок 1. Архитектура сенсорного узла

На сегодняшний момент понятие «сенсорная сеть» (wireless sensor networks) является устоявшимся термином, который характеризует распределенную, самоорганизующуюся систему миниатюрных автономных датчиков - мотов (motes) или сенсоров, которые могут связать воедино глобальные компьютерные сети. Мот представляет собой плату размером обычно не более одного кубического дюйма. Наиболее часто используются датчики температуры, давления, влажности, освещенности, вибрации, реже — магнитоэлектрические, химические, звуковые и некоторые другие. От выбора датчика зависит функциональное назначение беспроводной сенсорной сети.

Основная функциональная обработка данных, собираемых мотами, осуществляется на узле, или шлюзе, который представляет собой достаточно мощный компьютер (рис.1).

Подсистема восприятия (или подсистема мониторинга и восприятия) состоит, как правило, из аналогового устройства, снимающего определенную статистику (например, температуру, влажность, частоту сердечных сокращений и т.п.) и аналого-цифрового преобразователя, преобразующего аналоговые данные в цифровые для последующей передачи и/или обработки. Именно эта подсистема определяет ту область или приложение, в котором сенсор может быть использован.

Подсистема обработки данных включает в себя память и центральный процессор, позволяющие хранить и обрабатывать как генерируемые сенсором данные, так и служебные данные, необходимые для корректного и эффективного функционирования коммуникационной подсистемы.

Все методы построения беспроводных сенсорных сетей можно объединить в две группы:

1. С одним главным узлом (single-hop) – мощность передатчика сенсора достаточна для передачи сигнала к базовой станции.
2. С несколькими главными узлами (multi-hop) – некоторые узлы не только собирают информацию о наблюдаемом процессе, но и собирают информацию от других узлов.

Основными преимуществами беспроводных сенсорных сетей перед другими существующими беспроводными и проводными системами являются:

1) установка датчиков на уже существующее производство без дополнительных монтажных работ, что также упрощает техническое обслуживание системы;
2) невысокая стоимость основной составляющей сенсорной сети – датчика;
3) высокая отказоустойчивость сенсорной сети в целом.

Однако сенсорные сети имеют собственные ограничения и проблемы. Основные проблемы сенсорных сетей приведены на рисунке 2.

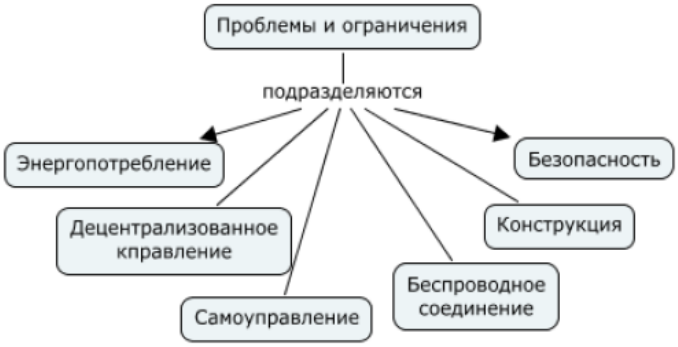


Рисунок 2. Список проблем и ограничений в беспроводных сенсорных сетях

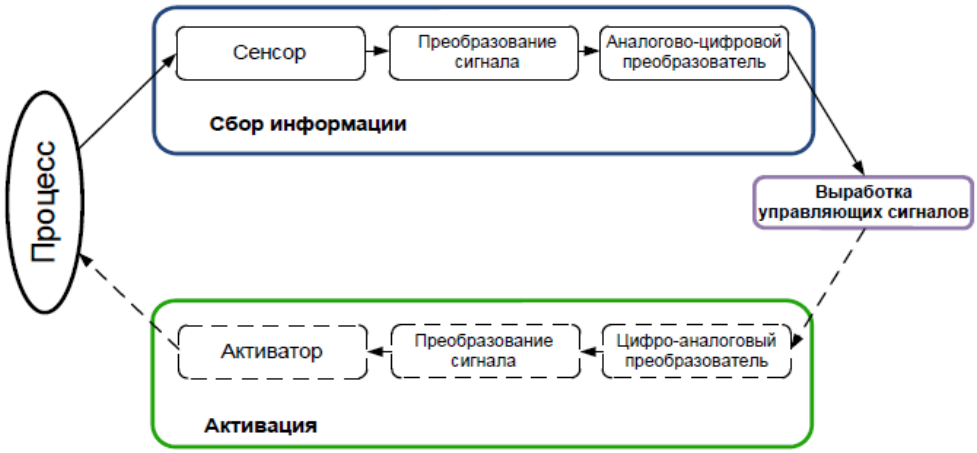


Рисунок 3. Процесс сбора данных информации и активации процесса в сенсорных сетях

Сенсорные сети предназначены для преобразования информации, полученной в результате наблюдения за физическим объектом, в форму, которая может быть использована для хранения информации и ее последующем преобразованием. По результатам обработки

информации может быть выработано управляющее воздействие (рис.3).

Каждая беспроводная сенсорная сеть обладает определенным набором параметров, таким как:

- количество узлов сети;
- скорость передачи данных по каналам связи – параметр определяется битовой скоростью канала. Как правило беспроводная сенсорная сеть состоит их одинаковых устройств, поэтому для всех каналов связи битовая скорость передачи будет одной и той же. Однако, на практике реальная скорость передачи будет ниже и для каждого канала своя, что связано с ошибками при передаче данных по радиоканалу, а также с особенностями использования общей для нескольких устройств среды передачи данных;
- топология сети – определяется наличием или отсутствием радиоканала между любыми двумя узлами;
- плотность размещения узлов – среднее количество расположенных рядом с узлом «соседей»;
- диаметр сети – минимальное количество ретрансляций для передачи данных между двумя самыми удаленными узлами сети;
- задержка в сети – время прохождения от момента возникновения события до момента появления информации о нем на базовой станции;
- время работы сети – работа сети без подзарядки и без замены аккумуляторов;
- пропускная способность – характеризуется объемом информации в единицу времени.

Самый распространённый метод определения эффективности системы основывается на оценке её работы по отдельно взятому критерию. Такой метод оценки эффективности хорош лишь в том случае, когда необходимо выбрать систему, оптимальную по одному из выбранных параметров при заданных условиях функционирования. На сегодняшний день отсутствует методика, позволяющая произвести комплексную оценку эффективности работы беспроводной сенсорной сети. Но что же делать, когда эффективность работы системы нужно оценить по двум и более параметрам?

Эффективность - это свойство системы выполнять поставленную цель в заданных условиях использования и с определенным качеством. Показатели эффективности характеризуют степень приспособленности системы к выполнению поставленных перед нею задач и являются обобщающими показателями оптимальности функционирования ИС, зависящими от локальных показателей, каковыми являются надежность, достоверность, безопасность.

Показатель эффективности сети – это количественная характеристика сети, рассматриваемая применительно к определенным условиям ее функционирования. Он определяется процессом функционирования сети и является функционалом от этого процесса.

Для того чтобы оценить эффективность функционирования беспроводной сенсорной сети необходимо определить назначение сенсорной сети, что позволит выбрать какие параметры необходимо учесть, характеризующие сенсорную сеть. На основе модели сенсорной сети и ее параметров можно произвести комплексную многокритериальную оценку, где каждому параметру будет задан экспертами весовой коэффициент в зависимости от назначения беспроводной сенсорной сети.

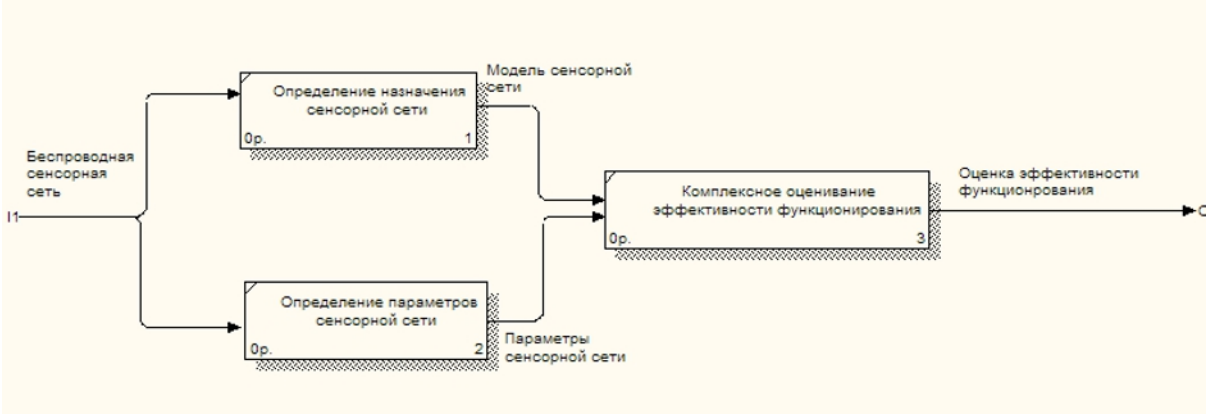


Рисунок 4. Методика оценки эффективности функционирования беспроводной сенсорной сети

Сенсорные сети можно классифицировать по следующим факторам:

1. По требованиям к скорости передачи показаний датчиков:

- «мгновенная» передача;
- с низкой задержкой – незначительная временная задержка допускается;
- с высокой задержкой – допускается передача с большими временными интервалами.

2. По типу организации электропитания устройств сенсорной сети:

- стационарные – питание всех узлов, независимо от функциональной нагрузки, осуществляется от внешней сети электропитания либо от элементов питания высокой емкости;
- полустационарные – электропитание узлов, подвергающихся наибольшей сетевой нагрузке, осуществляется от внешней сети электропитания либо от элементов питания высокой емкости; оконечные узлы имеют автономные элементы питания;
- автономные – ретрансляторы и рядовые узлы сети имеют собственные ограниченные автономные источники питания.

3. По времени «жизни» сенсорной сети:

- кратковременная – от нескольких часов до нескольких дней;
- среднесрочные – до нескольких месяцев;
- долгосрочные – до нескольких лет.

Задача определения эффективности функционирования является многокритериальной задачей, в которой очень важно предварительное ранжирование показателей (сортирование по степени их важности).

Соответственно для оценки будут созданы 3 типа моделей сенсорных сетей, которые в свою очередь включают 3 подтипа. Далее необходимо описать какие параметры критичны для каждой модели и присвоить этим параметрам максимальный вес, то есть произвести ранжирование. Также необходимо не забывать, что сумма всех весов должна быть равна 1. После этого расставить по приоритетности остальные параметры. У каждой модели получится своя таблица весов на основе экспертного мнения. Таких таблиц будет 9. Весовой критерий позволяет учесть значимость частных критериев, полученных на начальном этапе оценки. В итоге комплексная оценка, а именно показатель эффективности функционирования сенсорной сети рассчитывается по следующей формуле:

$$Э = \sum_{i=1}^{n} вес_i Э_i,$$

где, как легко догадаться, Э – показатель эффективности, $Э_i$ – частные параметры сенсорной сети и $вес_i$ – их значимость в рассматриваемой сенсорной сети. Для устранения компенсации одного частного критерия другим х значения используются в приведённом к максимуму форме. Весовые коэффициенты берутся положительными для максимизируемых и отрицательными для минимизируемых частных критериев.

После того как посчитали количественную характеристику эффективности беспроводной сенсорной сети необходимо объективно оценить в процентном соотношении на сколько она близка к «идеалу». Также необходимо составить базу правил на основе которой, мы получим качественную характеристику сети, такую, например, как:

Правило: если «показатель эффективности» не больше 20%, то сеть «неисправна».

Предложенная методика в данной статье позволяет дать комплексную оценку работы системы, в отличие от существующей методики оценки по одному критерию, которая не позволяет в полной мере оценить эффективность функционирования сенсорной сети.

Литература

1. Лекционный материал // учеб. пособие по мероприятию программы повышения квалификации для учителей школ г.Москвы по направлению «Беспроводные динамические сенсорные сети» URL: http://miet.ru/.
2. Методы оценки эффективности работы беспроводной сенсорной сети / О.В. Тужилкин, Н.С. Ульянин // Известия ЮФУ. Технические науки, 05.2012 г. URL: http://izv-tn.tti.sfedu.ru/.
3. Сергиевский М. Беспроводные сенсорные сети // Компьютер Пресс №8. – 2007. URL: http://compress.ru.
4. Скрипов С.А. Имитационное моделирование беспроводных сенсорных сетей со сложной структурой // Молодой ученый, №1. – 2008 г. – С. 59-66.
5. Смурыгин И.М. Концепция организации беспроводных сенсорных сетей и их применение // Молодежный научно-технический вестник. – М.: ФГБОУ ВПО «МГТУ им. Баумана». – 2012 г.

Пусная О.П.[1], Зайцева Т.В.[2], Лихонина А.В.[3]

[1]Белгородский государственный национальный исследовательский университет (НИУ «БелГУ»), г. Белгород, старший преподаватель кафедры прикладной информатики и информационных технологий, pusnaya@bsu.edu.ru

[2]Белгородский государственный национальный исследовательский университет (НИУ «БелГУ»), г. Белгород, к.т.н., доцент кафедры прикладной информатики и информационных технологий, zaitseva@bsu.edu.ru

[3]Специализированная детско-юношеская спортивная школа олимпийского резерва №1 управления физической культуры и спорта Белгородской области, г. Белгород, чемпион мира, тренер отделения кикбоксинга

АВТОМАТИЗИРОВАННАЯ СИСТЕМА УПРАВЛЕНИЯ СПОРТИВНЫМИ ДОСТИЖЕНИЯМИ

КЛЮЧЕВЫЕ СЛОВА

Система управления, спортивные достижения, база данных, принятие решения.

АННОТАЦИЯ

В статье предложена к рассмотрению автоматизированная система управления спортивными достижениями на примере Федерации художественной гимнастики г. Белгород. Приведены примеры оценки и корректировки физических показателей спортсменок.

Ежегодно уровень технической и технологической оснащенности развитых спортивных держав растет. В настоящий момент, в стране, к сожалению, отсутствует эффективная система детско-юношеского спорта, отбора и подготовки спортивного резерва для спортивных сборных команд страны. Для решения данной проблемы была создана автоматизированная система управления спортивными достижениями.

Объектом исследования данной разработки является Федерация художественной гимнастики города Белгорода. Информационной основой при написании данного проекта явились законодательные и нормативно-правовые акты РФ в области развития физической культуры и информационной безопасности, а также совокупность документации о хозяйственной деятельности объекта исследования в сфере аппаратно-программного обеспечения, топологии локальных сетей и специализированных спортивных показателях.

Рис. 1. Внешний вид формы «Главная страница»»

Рассмотрим элементы созданной автоматизированной системы управления спортивными достижениями.

Внешний вид главной формы с размещенными на ней компонентами показан на рисунке 1.

Внешний вид формы, предназначенной для просмотра данных БД, с размещенными на ней компонентами показан на рисунке 2.

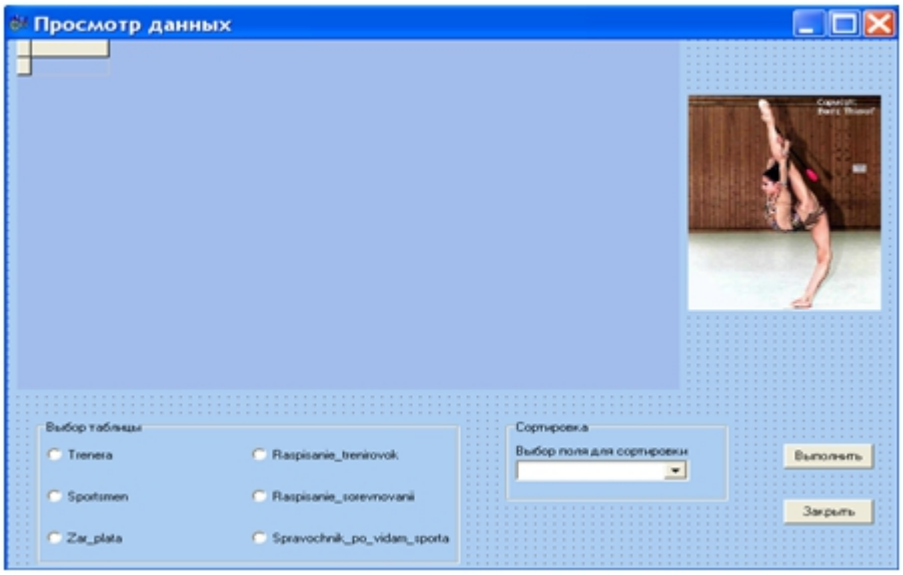


Рис. 2. Внешний вид формы «Просмотр данных»»

Внешний вид формы, предназначенной для поиска данных БД, с размещенными на ней компонентами показан на рисунке 3.

Внешний вид формы, предназначенной для добавления данных в БД, с размещенными на ней компонентами показан на рисунке 4.

Внешний вид формы, предназначенной для удаления данных из БД, с размещенными на ней компонентами показан на рисунке 5.

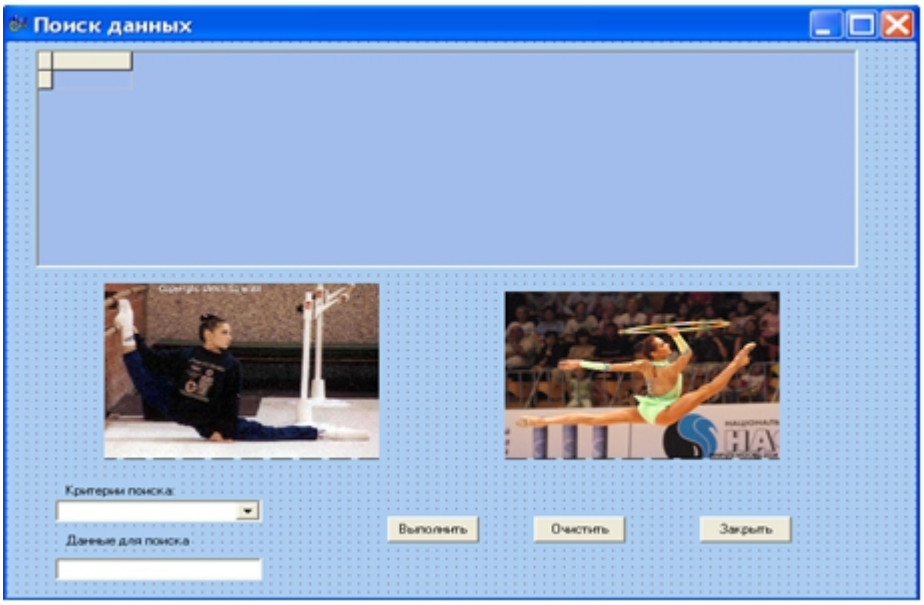


Рис. 3. Внешний вид формы «Поиск данных»»

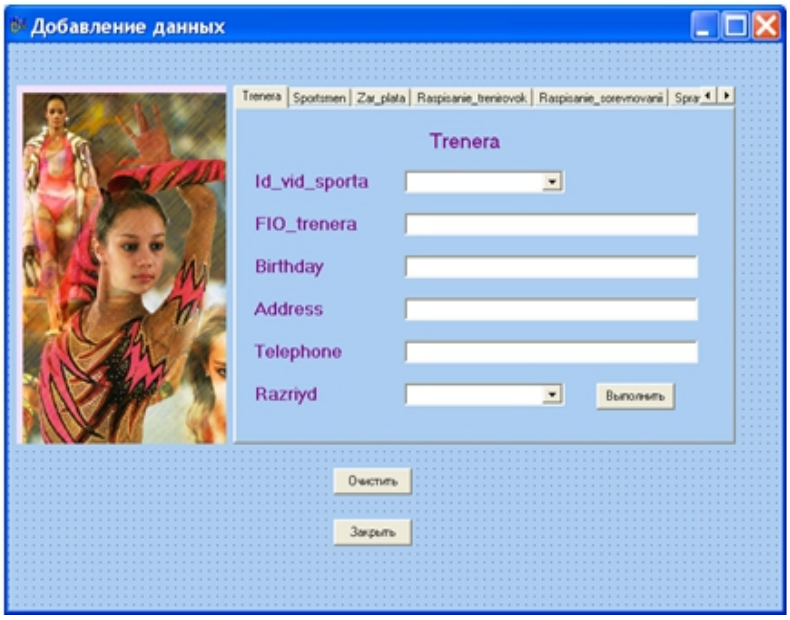


Рис. 4. Внешний вид формы «Добавление данных»

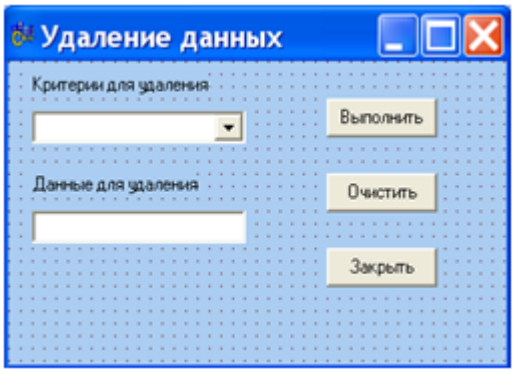


Рис. 5. Внешний вид формы «Удаление данных»

Внешний вид формы, предназначенной для изменения данных в БД, с размещенными на ней компонентами показан на рисунке 6.

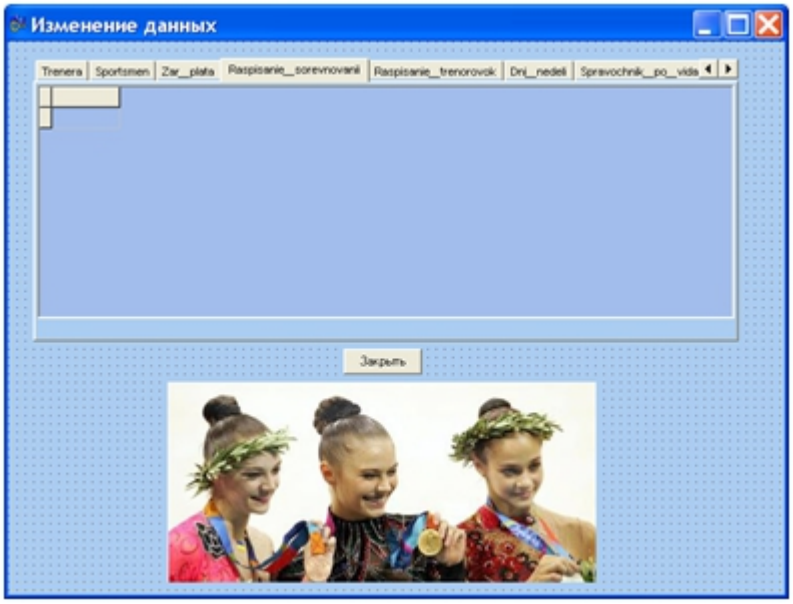


Рис. 6. Внешний вид формы «Изменение данных»

Для подтверждения работоспособности разработанной системы рассмотрим пример ее реализации.

Пример работы данной программы можно посмотреть на рисунке 7, где показано как осуществляется просмотр таблицы «тренеры».

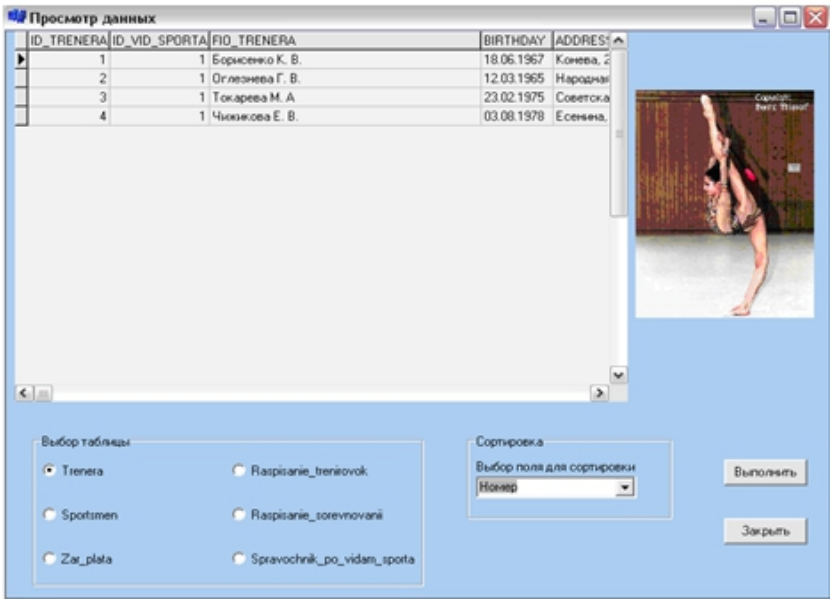


Рис. 7. Просмотр данных таблицы с сортировкой по номеру тренера

Так же можно просматривать информацию о спортсменах, заработной плате тренеров, расписании тренировок и соревнований. Возможно сортировка данных по соответствующим полям, а также поиск данных по заданным критериям и необходимая корректировка данных, в том числе введение новой информации и удаление неактуальной. Необходимая информация корректируется в программе, а затем автоматически сохраняется на сервере.

Пример изменения данных в программе можно посмотреть на рисунке 8, где показано изменение места проведения соревнований.

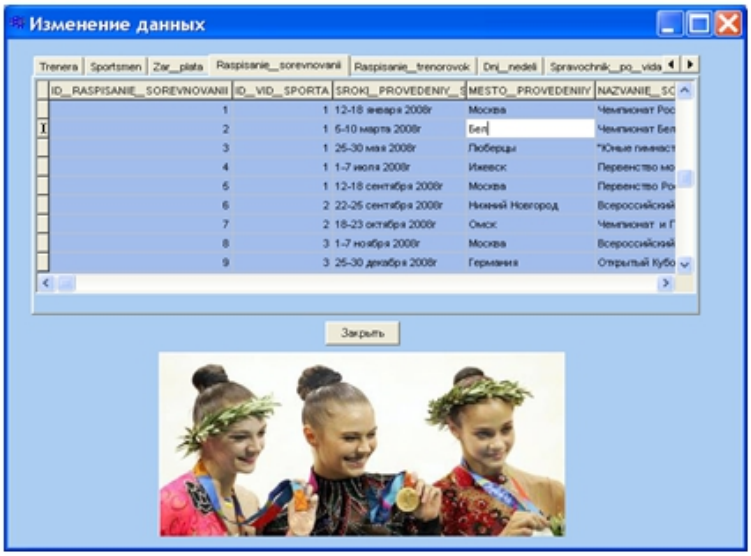


Рис. 8. Изменение данных

Одной из основных функциональных форм является форма, где происходит реализация вычисления результатов контрольных нормативов в БД. Именно на этой форме происходят основные действия по оценке и корректировке физических показателей спортсменок.

Следует отметить, что как с главной страницы, так и со страницы «Контрольные нормативы» и методист, и тренер, и даже спортсмен может просмотреть, и, если потребуется, распечатать развивающие упражнения для всех групп мышц и по всем видам нормативов. Данная

информация представлена в виде справочной информации. Она обязательна для каждого, действенна и актуальна.

Внешний вид формы «Контрольные нормативы», а точнее просмотр справочной информации о развивающих упражнениях представлен на рисунке 9.

Все нормативы в художественной гимнастике разбиты на соответствующие функциональные группы. Была реализована оценка каждого в отдельности взятого вида норматива. Здесь оцениваются результаты, которые показал спортсмен, дается небольшая характеристика возможностей спортсмена и предположение о зачислении спортсмена в резерв сборной команды г. Белгорода по художественной гимнастике.

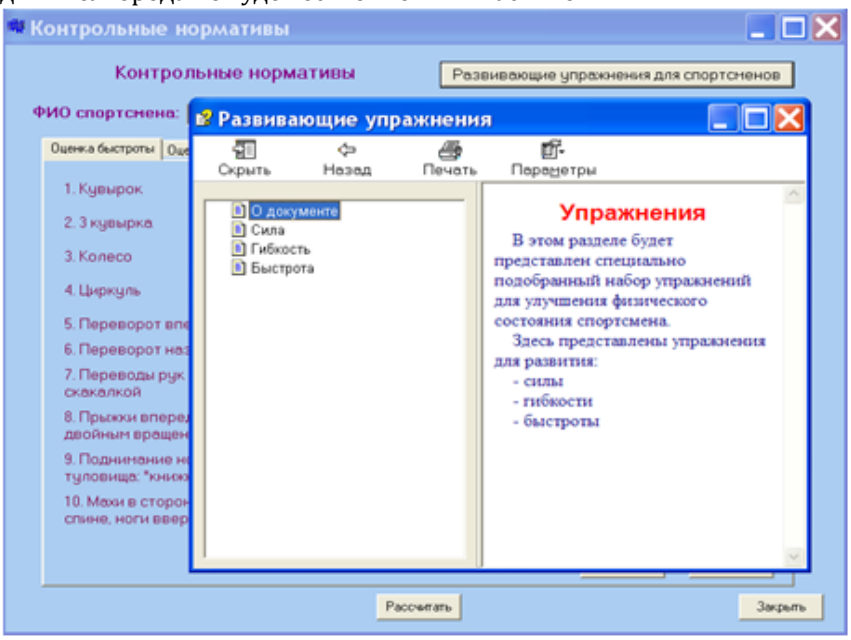


Рис. 9. Развивающие упражнения

Форма с результатами оценки конкретного вида норматива представлена на рисунке 10. Здесь осуществляется оценка прыгучести спортсмена.

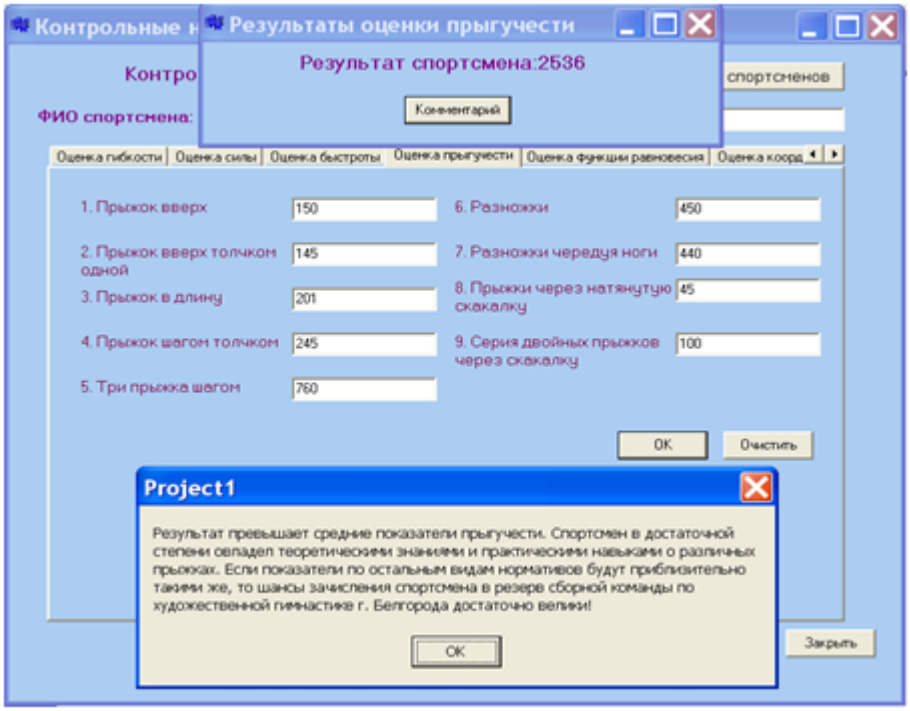


Рис. 10. Результаты оценки прыгучести

После того, как все поля по всем видам нормативов будут заполнены, программа обрабатывает получившиеся результаты.

Далее был реализован подсчет итоговых результатов. По каждому виду нормативов получают конечный результат, и составляется характеристика на спортсмена. Выводится анализ развития соответствующих способностей у спортсмена. Если знания и навыки используются не в полной мере, спортсмену настоятельно рекомендуется ознакомиться с развивающими упражнениями конкретно его интересующих физических показателей. Если же после оценки всех показателей, входящих в состав контрольных нормативов, спортсмен показал отличный результат, его зачисляют в резерв сборной команды г. Белгорода по художественной гимнастике. Результаты являются достоверными, осталось учесть только психологическую готовность спортсмена.

Результат конечной оценки физических возможностей спортсмена по всем видам нормативов представлен на рисунке 11.

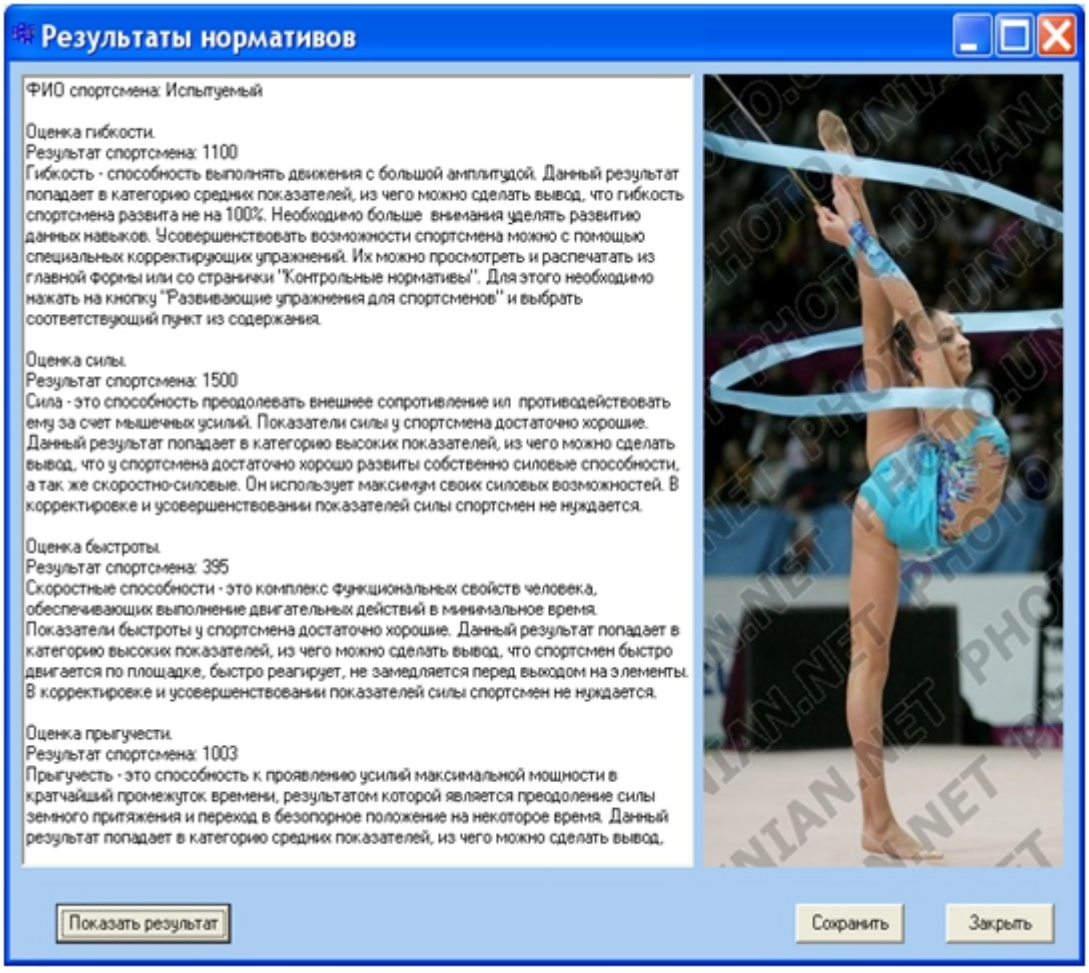


Рис. 11. Результаты нормативов

В Белгородской области художественная гимнастика является массовым видом спорта. Здесь работают квалифицированные тренеры, есть подающие надежду спортсмены, созданы достаточно неплохие условия для тренировок, но нет технической и технологической базы, которая помогла бы тренерам значительно быстрее корректировать тренировочный процесс на основе результатов и достижений своих спортсменов, а также получать рекомендации по усовершенствованию спортивного мастерства.

В статье приведено описание одного из подходов к решению вышеперечисленных задач, которое заключалось в разработке автоматизированной информационной системы.

При формировании требований к системе были выдвинуты условия, которые надо было учесть при разработке данного программного продукта, а именно, система должна обеспечивать более эффективные и продуктивные взаимодействия тренеров и спортсменов, тренеров и методистов, а также тренерско-преподавательского состава.

В процессе разработки практической части были использованы следующие программные средства: СУБД Firebird с утилитой IBExpert, Borland C++ Builder.

Автоматизация системы управления спортивными достижениями основана на внедрении программного обеспечения в деятельность обучающих специалистов на всех этапах подготовки спортсменов.

Разработанная система предоставляет следующие возможности:

- автоматизировать операции, происходящие с документами;
- создавать, вести и хранить документы не только на бумажном носителе, но и в электронном виде;
- усовершенствовать процесс поиска необходимой информации. Отчеты можно генерировать с помощью вычислительной техники по заданным критериям.

В рамках реализации данной системы разработаны следующие подсистемы управления спортивными достижениями:

- подсистема добавления, удаления, изменения информации;
- подсистема поиска по заданным критериям;
- подсистема отчетов по категориям распределения тренировочной и соревновательной нагрузки, а также спортсменам и тренерам;
- подсистема справочной информации о комплексе развивающих упражнений;
- подсистема подсчета промежуточных и итоговых результатов по показателям контрольных нормативов с возможностью формирования в автоматизированном режиме резерва спортивных кадров.

Достижение цели и задач проекта по созданию автоматизированной системы управления спортивными достижениями позволит:

- грамотно скоординировать тренировочный процесс обучающим специалистам по художественной гимнастике;
- выявить сильные и слабые стороны физической подготовленности спортсменам;
- увеличить скорость обмена данными и скоординировать процесс взаимодействия Федерации гимнастики с Управлением по физической культуре, спорту и туризму Белгородской области;
- повысить производительность и эффективность деятельности методиста, за счет внедрения специализированного программного продукта.

Литература

1. Теория и методика физической подготовки в художественной и эстетической гимнастике : учебное пособие. — Электрон. дан. — М. : Советский спорт, 2014. — 264 с. — Режим доступа: http://e.lanbook.com/books/element.php?pl1_id=51909.
2. Зайцева Т.В., Игрунова С.В., Путивцева Н.П. Пусная О.П. Разработка динамической экспертной системы для консультирования клиентов на примере фирмы «ЛИВАМ» // Журнал «В мире научных открытий» Серия Математика. Механика. Информатика. Красноярск. Вып.№8(20), 2011 с.153-163.
3. Зайцева Т.В., Нестерова Е.В., Игрунова С.В., Путивцева Н.П., Пусная О.П. Экспертная система для выбора вида обслуживания ит - инфраструктуры предприятия // Инновационные информационные технологии: Материалы международной научно-практической конференции. / Под ред., С.У. Увайсова; Отв. за вып. И.А. Иванов, Л.М. Агеева, Д.А. Дубоделова, В.Е. Еремина–М.:МИЭМ, 2012, 602 с. С. 532-534 (Innovation Information Technologies: Materials of the International scientific – practical conference. /Ed. Uvaysov S. U., Ivanov I. A., Ageeva L. M., Dubodelova D. A., Eremina V. E. –M.: MIEM, 2012, 602 p. Prague – 2012 April 23-27).

Губарев А.В.

Юго-Западный государственный университет, г. Курск, аспирант кафедры защиты информации и систем связи ЮЗГУ, gybarev-av@yandex.ru

МОДЕЛЬ УГРОЗ СИСТЕМЫ ПЕРЕДАЧИ ДАННЫХ МЕЖДУ УПРАВЛЯЮЩИМ ПРОГРАММНЫМ ОБЕСПЕЧЕНИЕМ И АППАРАТНЫМ СРЕДСТВОМ

КЛЮЧЕВЫЕ СЛОВА

Командное слово, модель угроз, субъектно-объектная модель, имитовставка, определение подлинности.

АННОТАЦИЯ

В статье дается описание модели угроз системы передачи данных между управляющим программным обеспечением и аппаратным средством и алгоритма определения подлинности передаваемой информации, который позволяет противодействовать угрозам в данной системе. При описании модели угроз используется субъектно-объектная модель.

Информационная система, в которой происходит обмен сообщениями, всегда может служить объектом нападения со стороны нарушителя, располагающего тем или иным потенциалом. Одной из распространенных атак является активная атака, которая носит название имитации. Атака имитации состоит в том, что нарушитель «вставляет» в канал связи сфабрикованное им сообщение, которое на самом деле не передавалось от легального отправителя к получателю. При этом нарушитель рассчитывает на то, что получатель воспримет это сфабрикованное сообщение как подлинное.

Одним из важных подходов к противодействию атакам типа имитации в настоящее время является проверка имитовставки каждого сообщения. Имитовставка – в протоколах аутентификации сообщений с доверяющими друг другу участниками – специальный набор символов, добавляемый к сообщению и предназначенный для обеспечения его целостности и аутентификации источника данных.

Задача формирования имитовставки сообщения сводится к применению соответствующих алгоритмов, которые представляют собой совокупность сложных математических операций, требующих определенных, в том числе временных, ресурсов. Кроме этого, известные алгоритмы шифрования и формирования имитовставок сообщений используются при передаче информации внутри информационно-вычислительной сети, а также устанавливают строго определенные требования к длине сообщения или блоков сообщения (64 бита, 128 бит и т.д.), что делает невозможным их использование при информационном обмене между управляющим программным обеспечением и аппаратным средством командными словами, длина которых ограничена шириной интерфейса аппаратного средства. С учетом всего вышеизложенного был разработан метод определения подлинности передаваемых командных слов [1], краткое описание которого приведено ниже.

Метод определения подлинности передаваемых командных слов заключается в передаче объединенных в пул командных слов и формировании зависимостей между ними таким образом, чтобы принимающая сторона однозначно смогла определить те слова, которые были выданы легальным источником. Примем, что $\{A|B|\cdots|N\}$ – слово, образующееся в результате конкатенации слов *A, B,… N*. Задача состоит в том, чтобы передать из источника в приёмник командное слова S_i из пула командных слов и его номер *i*, преобразованные в соответствии с секретным словом S^{sec}, известным и источнику, и приёмнику, таким образом, чтобы в передаваемом слове не встречались в открытом виде ни S_i, ни *i*. Для этого источник выполняет следующую последовательность действий:

- Формирует слово $i' = F_A\left(S^{sec}, i\right)$;
- Формирует слово $S_i^{'} = F_B\left(S_i, i'\right)$;

- Формирует слово $i'' = F_C\left(S_i^{'}, i\right)$;
- Отправляет в приёмник слово $\left\{S_i^{'} \middle| i''\right\}$

Приёмник, в свою очередь, выполняет следующие операции

- Определяет номер $i_{пр} = F_C^{-1}\left(S_i^{'}, i''\right)$;
- Определяет слово $i_{пр}^{'} = F_A\left(S^{sec}, i_{пр}\right)$;
- Определяет содержимое полученного командного слова $S_i = F_B^{-1}\left(S_i^{'}, i_{пр}^{'}\right)$.

В данном случае $F_A^{\square}(A,B)$ – необратимое преобразование, $F_B(A,B)$ и $F_C^{\square}(A,B)$ – обратимые преобразования слова *B* в соответствии с ключом *A*, в результате которых длина получаемое слова равна длине слова *B*, $F_B^{-1}(A,B)$ и $F_C^{-1}(A,B)$ преобразования, обратные $F_B^{\square}(A,B)$ и $F_C^{\square}(A,B)$ соответственно.

После получения приемником последнего командного слова из пула, он начинает разбирать принятые командные слова. Назовём номер $i_{пр}^{\square}$, выделенный приёмником из поступившего пула командных слов, ярусом командного слова. В случае, если полученное командное слово выдано легальным источником, то $i_{пр}^{\square} = i$. В то же время в буфере командных слов может одновременно присутствовать несколько слов с одинаковым ярусом. Такая ситуация возможна при навязывании ложной информации (командных слов) посторонним источником, в качестве которого может выступать программная или аппаратная закладка. Для каждого слова, поступающего в буфер, запоминается его порядковый номер *j* с начала текущего пула. В случае, когда посторонние источники командных слов отсутствуют, выполняется равенство $j = i_{пр}^{\square} = m$, где m – максимальный ярус слов, записанных в буфере. В общем же случае (когда в буфере есть посторонние командные слова) $j \geq i_{пр}^{\square}$, $j \geq m$, $m \geq i_{пр}^{\square}$. Условие записи вновь поступившего командного слова в буфер на ярус $i_{пр}^{\square}$ выглядит следующим образом:

$$i_{пр}^{\square} = F_C^{-1}\left(S_i^{'}, i''\right) \leq m+1$$

В противном же случае слово игнорируется.

После того, как все принятые командные слова распределены по ярусам, необходимо с каждого яруса выбрать по одному слову и составить из него пул командных слов, выданных настоящим источником.

Для выделения из буфера слов легального источника недостаточно обладать информацией о ярусе командного слова, необходимо, чтобы содержимое каждого командного слова источника зависело от содержимого остальных командных слов текущего пула. Наиболее простым средством достижения этого является хеширование сообщений. Каждое командное слово легального источника S_i состоит из информационной части $S_i^{инф}$ – данных, непосредственно обрабатываемых приёмником, и небольшой по объёму имитовставки $S_i^{им}$, сформированной из всех переданных ранее информационных частей командных слов $S_1^{инф} \div S_{i-1}^{инф}$:

$$S_i = \left\{ S_i^{инф} \middle| F_{хеш}\left(S_{i-1}^{инф}, S_{i-2}^{инф}, \ldots, S_1^{инф}\right)\right\},$$

где $F_{хеш}$ – функция криптографического хеширования нескольких слов и формирования из произвольного объёма данных заданной по размеру имитовставки.

Таким образом, формат командного слова, воспринимаемого приёмником, выглядит, как показано на рис. 1:

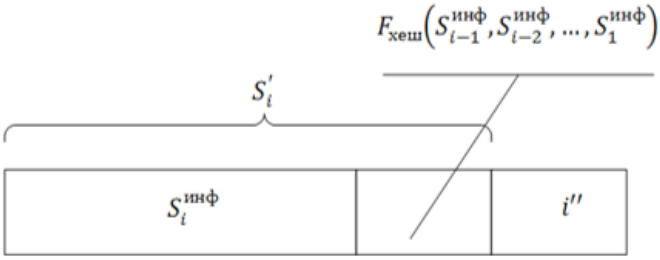


Рис. 1. Формат командного слова

Пусть $S^{\square}_{j,r}$ – слово, поступившее в приёмник *j*–м с начала текущего пула и определённое на *r*–й ярус, $S^{инф}_{j,r}$ – его информационная часть, $S^{им}_{j,r}$ – его имитовставка. Тогда необходимо построить цепочку из *M* слов $S^{инф}_{j(1),1}$, $S^{инф}_{j(2),2}$, ... $S^{инф}_{j(M),M}$ при одновременном соблюдении условий:

$$j(1) \le j(2) \le \ldots \le j(M), \tag{2.3}$$

$$S^{им}_{j(r),r} = F_{хеш}\left(S^{инф}_{j(r-1),r-1}, S^{инф}_{j(r-2),r-2}, \ldots, S^{инф}_{j(1),1}\right), \text{ где } r = 2 \ldots M$$

Из данных условий выводится алгоритм выбора из буфера слов легального источника, который сводится к поочерёдному выбору слов с ярусов буфера и проверки их номеров и имитовставок.

Как уже было сказано выше, источником посторонних командных слов могут являться программные и аппаратные закладки, являющиеся частью информационной системы, в которой происходит передача командных слов от специализированного программного обеспечения к соответствующему аппаратному средству. Стоит отметить, что в процессе моделирования угроз система передачи защищенных командных слов рассматривается как одна из подсистем общей информационной системы, элементами которой являются различные аппаратные средства, а также специализированное программное обеспечение. В рамках данного доклада рассматриваются угрозы, непосредственно связанные с системой передачи защищаемых командных слов и воздействующие на нее, которые возникают в результате активного функционирования программных или аппаратных закладок, направленные на объекты, входящие в данную систему, а также на работу всей системы передачи защищенных командных слов в целом. Угрозы, связанные с действиями пользователей, несанкционированным доступом к информации, перехватом информации, а также угрозы целостности и доступности информации и другие, не относящиеся к системе передачи командных слов не рассматриваются.

Для описания системы передачи командных слов с точки зрения обеспечения безопасности можно использовать субъектно-объектную модель, охватывающую класс угроз, соответствующий описанным программным и аппаратным закладкам [2].

Система передачи командных слов может быть абстрагирована с помощью понятий: объект, субъект, операции для пары «объект-субъект».

Данным абстрактным понятиям поставим в соответствие следующие физические представления в информационной системе.

Объект (O_j) – часть ресурсов информационной системы, с которыми могут взаимодействовать субъекты посредством операций. В рассматриваемой системе передачи данных интересующими нас объектами будут являться передаваемые командные слова, а также аппаратные средства, принимающие и обрабатывающие их.

Субъект (S_i) – элемент, способный осуществить операцию над объектом. Будем рассматривать такой элемент, как пару: ресурсы и операция. В рассматриваемой системе передачи данных в роли субъектов будут выступать специализированное программное обеспечение, осуществляющее передачу командных слов соответствующему аппаратному средству, а также закладки, способные генерировать посторонние командные слова и отправлять их через канал связи соответствующему аппаратному средству.

Множество операций (R) – действия, которые могут совершать субъекты над объектами. В рассматриваемой системе передачи командных слов основными операциями будут являться подготовка и передача пула легальных командных слов специализированным программным обеспечением, а также формирование и отправка посторонних командных слов программной или аппаратной закладкой.

Программная или аппаратная закладка как некоторый тип субъекта, который является источником угроз, имеет следующие особенности:

- закладка, являясь программно-аппаратными ресурсами информационной системы, имеет одну и ту же среду существования, что и объекты, а также в пассивном состоянии является частью некоторого объекта;
- угрозы в рассматриваемой информационной системе могут исходить только от активного субъекта – закладки, который в текущий момент времени владеет функцией управления ресурсами и, в общем, случае способный изменять состояния объектов;
- субъекты в информационной системе могут влиять друг на друга через изменяемые ими объекты. Одним из результатов воздействия субъекта на объект могут быть такие

состояния информационной системы, которые могут представлять угрозу для безопасности информационной системы, а также для способности нормального функционирования элементов информационной системы.

Пары (S_i, O_j) связываются множеством разрешенных операция R_i. Это множество определяется политикой безопасности информационной системы и является подмножеством всего множества R возможных операция для каждой пары. В то же время, пары (S_i, O_j) могут связываться с множеством таких операций R_k, результатом выполнения которых являются деструктивные воздействия на информационную систему. Задачей метода определения подлинности передаваемых командных слов применительно к рассматриваемой информационной системе является противодействие деструктивным операциям из множества R_k. Очевидно, что $R = R_k R_i$.

Также уместно рассмотреть понятие пользователя (злоумышленника). Он всегда связывается с частью информационной системы (объектом), на которую направлены все его действия. Очевидно, что злоумышленник – это внешний по отношению к системе субъект, который может быть источником угроз. Отметим следующие важные свойства злоумышленника:

- злоумышленник воспринимает объекты и получает информацию о состоянии информационной системы через те активные субъекты, доступ к которым он имеет. Таким образом, злоумышленник должен инициализировать субъект – программную или аппаратную закладку и привести его в активное состояние;
- обобщая понятие субъекта, определенное выше, на злоумышленника, отметим, что злоумышленник является внешним субъектом.

В качестве основных уровней знаний нарушителя об информационной системе можно выделить следующие:

- данные об организации работы, структуре и используемых технических, программных и программно-технических средствах информационной системы;
- сведения об информационных ресурсах информационной системы: порядок и правила передачи информации, структура и свойства информационных потоков;
- данные об уязвимостях, включая данные о недокументированных (недекларированных) возможностях технических, программных и программно-технических средств информационной системы [3];
- сведения о возможных каналах реализации угроз;
- информацию о способах реализации угроз.

В то же время нарушитель не обладает следующими знаниями:

- сведения о секретном ключе и применяемых алгоритмах при передаче командных слов;
- сведения о размере командного слова в целом и каждой из его составных частей;
- сведения о деталях реализации метода определения подлинности передаваемых командных слов.

Основным средством реализации угроз рассматриваемой системы передачи командных слов для злоумышленника является программная или аппаратная закладка.

Модель рассматриваемой информационной системы можно представить в виде ориентированного размеченного графа, вершинами которого являются субъекты S_i и объекты O_j. Каждой дуге такого графа поставлена операция $r_i : r_i \subseteq R$. Такая модель представлена на рис. 2.

Для формализации механизма порождения новых объектов воспользуемся следующими определениями.

Определение 1. Объект O_i называется источником для субъекта S_k, если существует субъект S_j, в результате воздействия которого на объект O_i в информационной системе возникает субъект S_k.

Определение 2. Субъект S_j, порождающий новый субъект из объекта O_i, в свою очередь, называется активизирующим субъектом для субъекта S_k.

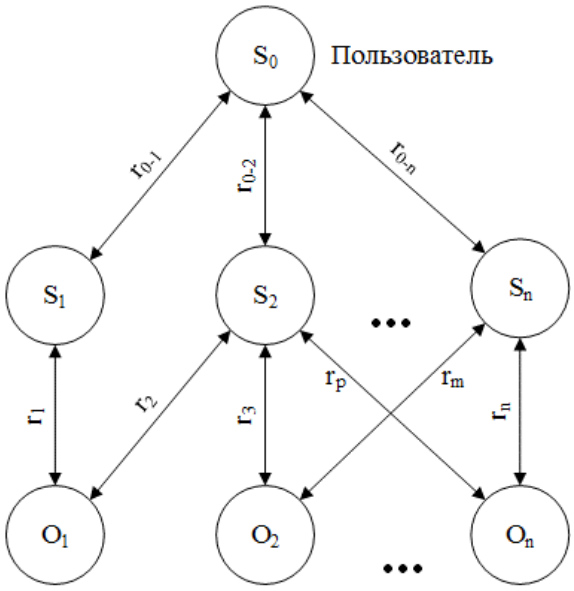


Рис. 2. Субъектно-объектная модель информационной системы

Для описания процессов порождения субъектов доступа вводится следующее обозначение:

$Create\left(S_j, O_i\right) \to S_k$ – из объекта O_i порожден субъект S_k при активизирующем воздействии субъекта S_j.

$Create$ называют операцией порождения субъектов. На физическом уровне операция $Create$ означает, что субъект S_j, обладая управлением и ресурсами, может передать S_k часть ресурсов и управление, тем самым его активизировав. Отметим также, что ввиду того, что в информационной системе действует дискретное время, то под воздействием активизирующего субъекта в момент времени t, новый субъект порождается в момент времени $t+1$. Очевидно, что операция порождения субъектов зависит как от свойств активизирующего субъекта, так и от содержания объекта-источника.

Если рассматривать субъекты и объекты в контексте описанной информационной системы, то свойство активности программных и аппаратных закладок реализуется не только в выполнении действий над объектом с целью создания нового субъекта, но и с целью записи данных в объект. В общем случае необходимо отметить, что поскольку объекты в информационной системе по определению являются пассивными, то для выполнения закладками всех описанных выше действий, необходимо существование потоков информации от субъекта к объекту. Так как данный поток инициируется и реализуется субъектом – закладкой, это означает, что операция порождения потока локализована в субъекте – закладке и отображается состоянием его функционально ассоциированных с ней объектов.

Поток информации между объектом O_m и объектом O_j всегда должен мыть связан с некоторой операцией на объектом O_j, реализуемой субъектом S_j и зависящей от O_m. Поток информации от объекта O_m к объекту O_j обозначим как $Stream\left(S_i, O_m\right) \to O_j$. При этом источником является объект O_m, а приемником потока является O_j. Отсюда следует, что поток всегда инициируется субъектом.

С помощью приведенного определения операции $Stream$ над объектами O_m и O_j формализуется операции субъекта S_i к O_j для передачи ему данных из O_m или наоборот. В частном случае операция $Stream$ может создавать новый объект или уничтожать его.

Следует особо подчеркнуть, что согласно определению $Stream$ потоки информации могут быть только между объектами, а не между субъектом и объектом, в виду того, что субъект это активная сущность, т. е. действия, процессы и т.д., а информация – пассивная сущность, которая может размещаться, извлекаться, порождаться, изменяться и т. д. только в объектах. Активная роль субъекта заключается в самой реализации потока, в его локализации в субъекте (через

субъект), в том числе, через задействование в потоке ассоциированных с субъектом объектов.

Типичными потоками в рассматриваемой системе передачи защищенных командных слов будут являться передача объектов, представленных пулами защищенных командных слов, к объекту – соответствующему аппаратному средству через специализированное программное обеспечение, которое в данном случае будет являться субъектом. В то же время, предполагая наличие программных или аппаратных закладок в рассматриваемой информационной системе, в системе передачи защищенных командных слов в результате выполнения операций субъектами, которыми являются соответствующие закладки, над объектами, то есть принимающими командные слова аппаратными средствами, могут образовываться потоки информации, которые несут деструктивный характер для объекта-приемника, то есть аппаратного средства, и всей информационной системы в целом. Именно подмножество таких потоков представляет собой угрозы информационной безопасности системы передачи защищенных командных слов.

К актуальным угрозам, порождаемым информационными потоками между закладкой и принимающим командные слова аппаратным средствам, относятся следующие:

- угроза передачи запрещенных командных слов соответствующему аппаратному средству;
- угроза имитации;
- угроза перехвата передаваемых командных слов средствами программной или аппаратной закладок;
- угроза несанкционированной модификации передаваемых командных слов;
- угроза потери передаваемых командных слов.

Напомним, что в рамках моделирования угроз не рассматриваются угрозы безопасности информации в сегментах информационной системы, не относящихся к системе передачи защищенных командных слов, поэтому представленный список содержит только угрозы, которые могут деструктивно воздействовать на рассматриваемые нами объекты, то есть передаваемые командные слова, а также аппаратные средства, принимающие указанные командные слова.

Рассмотрим подробнее каждую из описанных выше угроз, а также меры по противодействию данным угрозам в соответствии с методом определения подлинности передаваемых командных слов.

Угроза передачи запрещенных командных слов соответствующему аппаратному средству. Угроза заключается в возможной потере управления над устройством или частичной или полной потере функциональности соответствующего устройства в следствие передачи ему запрещенных командных слов. Реализация данной угрозы возможна при наличии у нарушителя достаточных знаний запрещенных команд, а также соответствующих программных или аппаратных закладок, являющихся источниками запрещенных команд, или возможности изменения команд, передаваемых легальным источником. Мерой противодействия данной угрозе является разработанный метод определения подлинности передаваемых командных слов, который позволяет аутентифицировать только те командные слова, которые были выданы легальным источником, и предотвратить исполнение команд, полученных от постороннего источника.

Угроза имитации. Угроза заключается в навязывании ложных команд аппаратному средству путем передачи ему сформированных по определенному алгоритму команд, что может привести к выполнению устройством неинициированных пользователем действий. Реализация данной угрозы возможна при наличии у нарушителя достаточных знаний алгоритма формирования команд, а также соответствующих программных или аппаратных закладок, являющихся источниками посторонних команд. Мерой противодействия данной угрозе является разработанный метод определения подлинности передаваемых командных слов, который позволяет аутентифицировать только те командные слова, которые были выданы легальным источником, и предотвратить исполнение команд, полученных от постороннего источника.

Угроза перехвата передаваемых командных слов средствами программной или аппаратной закладок. Угроза заключается в возможности осуществления нарушителем несанкционированного доступа к командным словам, передаваемым от управляющего программного обеспечения соответствующему аппаратному средству. Данная угроза обусловлена недостаточностью мер защиты информации от утечки и контроля потоков данных, а также невозможностью осуществления защиты командных слов с помощью криптографических средств (т.к. передаваемые командные слова должны передаваться в доступном для понимания соответствующим аппаратным средством виде, а также в связи с малым размером передаваемых командных слов). Реализация данной угрозы возможна при условии наличия у нарушителя привилегий на установку и запуск программных и аппаратных закладок, осуществляющих

перехват и передачу злоумышленнику командных слов. Мерой противодействия данной угрозе является разработанный метод определения подлинности передаваемых командных слов, который позволяет преобразовывать командные слова по определенному алгоритму в соответствии с секретным ключом с целью исключения передачи командных слов в открытом виде.

Угроза несанкционированной модификации передаваемых командных слов. Угроза заключается в возможности нарушения целостности передаваемых командных слов путем осуществления нарушителем деструктивного воздействия (искажения, замены отдельных бит) на передаваемых командные слова. Реализация данной угрозы возможна при условии наличия у нарушителя привилегий на установку и запуск программных и аппаратных закладок, осуществляющих модификацию передаваемых командных слов. Мерой противодействия данной угрозе является разработанный метод определения подлинности передаваемых командных слов, который проверяет имитовставки принятых командных слов, пытается построить цепочку легальных командных слов и в случае невозможности нахождения такой цепочки (в том числе в результате модификации отдельного командного слова) отправляет запрос на повторную передачу командных слов.

Угроза потери передаваемых командных слов. Угроза заключается в возможности нарушения доступности передаваемых командных слов путем удаления (затирания) передаваемых командных слов. Реализация данной угрозы возможна при условии наличия у нарушителя привилегий на установку и запуск программных и аппаратных закладок, осуществляющих затирание передаваемых командных слов. Мерой противодействия данной угрозе является разработанный метод определения подлинности передаваемых командных слов, который проверяет имитовставки принятых командных слов, каждая их которых зависит от содержимого предыдущего командного слова, пытается построить цепочку легальных командных слов и в случае невозможности нахождения такой цепочки в том числе в результате затирания одного или нескольких командных слов, отправляет запрос на повторную передачу командных слов.

Так как разрабатываемый метод позволяет находить командные слова, переданные легальным источником, среди множества всех принятых командных слов, особый интерес представляет угроза имитации командных слов, в результате которой злоумышленник пытается навязать принимающей стороне посторонние командные слова. В случае, когда в системе передачи командных слов существуют только легальные потоки информации $Stream$ между субъектом и объектом, все M принятых слов выданы легальным источником, и данные командные слова образуют цепочку, удовлетворяющую условию (2.3). Таким образом, мы можем сказать, что все командные слова легального источника будут опознаны и обработаны, то есть невозможна потеря отдельных слов цепочки и замена их словами посторонних источников. В противном же случае, когда в системе передачи командных слов присутствуют деструктивные потоки информации $Stream$, среди общего количества принятых командных слов будут находиться слова, выданные посторонним источником, в результате чего может возникнуть коллизия в процессе построения цепочки легальных командных слов. Коллизией же при выборе будем называть ситуацию, при которой помимо цепочки из таких легальных командных слов будет сформирована альтернативная цепочка, в которой от 1 до M слов будет выдано посторонним источником.

Следует отметить, что слова, принятые приёмником, не рассматриваются по отдельности, а только как неотъемлемая часть цепочки из M слов. Поэтому обе цепочки имеют все основания быть опознанными как выданные легальным источником. Поэтому, в случае, если алгоритм выявил две или более конкурирующие цепочки сообщений, требуется повторная передача источником всего пула командных слов. В случаях же частого появления коллизий в процессе построения цепочки из легальных командных слов отправляющей стороне необходимо рассмотреть ряд факторов, влияющих на появление коллизий, и учесть их при повторной передаче командных слов. Такими факторами могут являться размеры информативной части и имитовставки, которые зависят от количества посторонних слов. При возникновении коллизий в целях исключения угрозы «отказ в обслуживании» необходимо увеличить размер имитовставки, что существенно снизит вероятность возникновения коллизий. Моделирование системы передачи командных слов, нахождение зависимости вероятности возникновения коллизий от характеристик данной системы рассматриваются в работе [4].

Таким образом, проведенное моделирования угроз системы передачи данных между

управляющим программным обеспечением и аппаратным средством позволяет найти и описать возможные угрозы данной системы, а также разработать меры противодействия данным угрозам, в данном случае, систему передачи защищенных командных слов.

Литература

1. Таныгин М.О. Верификация данных, передаваемых между устройством и программным обеспечением [Текст] // Электронные средства и системы управления: Материалы докладов Международной научно-практической конференции (13–16 октября 2010 г.). – Томск: В-Спектр, 2011: В 2 ч. – Ч. 2. С. 49 – 52.
2. Зегжда Д.П. Основы безопасности информационных систем [Текст] / Д. П. Зегжда, А. М. Ивашко – М.: Горячая линия - Телеком, 2000. – 452 с.
3. Руководящий документ. Защита от несанкционированного доступа к информации. Часть 1. Программное обеспечение средств защиты информации. Классификация по уровню контроля отсутствия недекларированных возможностей. – М.: Гостехкомиссия, 1998. – 9 с.
4. Губарев А.В. Моделирование работы системы контроля подлинности командных слов [Текст] / А. В. Губарев // Проблемы информационной безопасности. Компьютерные системы – 2014. – № 4 – С. 169–175.

Марценюк М.А.[1], Селетков И.П.[2]

[1] Пермский государственный национальный исследовательский университет, г.Пермь, доктор ф.-м. наук., профессор, заведующий кафедры Компьютерных систем и телекоммуникаций, mrcn@psu.ru

[2] Пермский государственный национальный исследовательский университет, г.Пермь, ассистент кафедры компьютерных систем и телекоммуникаций, iseletkov@gmail.com

ИСПОЛЬЗОВАНИЕ НЕЧЁТКОГО КЛЕТОЧНОГО АВТОМАТА ДЛЯ МОДЕЛИРОВАНИЯ ДИНАМИЧЕСКИХ ПРОЦЕССОВ В СРЕДАХ С ПАМЯТЬЮ

КЛЮЧЕВЫЕ СЛОВА

Нечеткая логика, нечеткий клеточный автомат, динамические процессы, среды с памятью.

АННОТАЦИЯ

Рассмотрена модель клеточного автомата с памятью, построенная с использованием нечеткой логики. Показано, что использование методов нечеткой логики в моделях клеточных автоматов позволяет повысить гибкость алгоритмов, упростить формализацию знаний эксперта. Для всех вычислений применяется матричный подход нечеткой логики, позволяющий упростить реализацию алгоритма и уменьшить требования к вычислительным ресурсам. В качестве конкретного примера обсуждается задача расчета поля температуры с учётом эффекта температурной памяти.

Введение

Понятие «клеточного автомата» (КА) было впервые введено Дж. Фон Нейманом [1] для реалистичного моделирования пространственно-протяжённых систем [2]. В последние десятилетия предложенные модели получили особенно значительное распространение в таких областях, как моделирование химических и биологических процессов [3,4], решение задач математической физики, обработке изображений [5, 6], моделировании социальных систем [7] и многих других. Обзор современных применений модели клеточных автоматов для решения научных и технических задач представлен в книгах [8, 9] и других источниках.

Как отмечает в своём предисловии к книге [10] известный эксперт в теории КА профессор Т. Тоффоли (T.Toffoli), 50 лет исследований выявили, что КА-модели систем в своей основе служат *концептуальным инструментом познания*. Эти исследования также выявили особое место КА: они дают наибольший вклад в понимании сложных систем на стадии разработки их концептуальных моделей.

Перейдём теперь к более детальному описанию КА. Клеточный автомат состоит из упорядоченного массива данных (клеток) $C=\{C_i, i=1,2,\dots,n\}$, которые описывают поле $C(t)$ (пространственное распределение) исследуемой величины в текущий момент времени *t.* Далее во введении для простоты (но без ограничения общности) считается, что размерность $\dim(C)=1$. Обработка массива $C(t)$ логическим автоматом по единому набору правил f позволяет вычислить массив в последующий момент времени $t+\Delta t$: $C(t+\Delta t)=f(C(t))$. Здесь и далее мы отличаем понятия «клеточный автомат (КА)» и «логический автомат (ЛА)»: КА включает в себя массив $C(t)$ и способ его обработки – ЛА, т.е. КА=$C(t)$+ЛА. Функцию f часто называют правилом работы автомата.

В литературе были предложены несколько алгоритмов работы логического автомата:

- Автомат без памяти или комбинационная схема (КС), действующая по чётко определённым правилам (ЧКС). Эта модель автомата является наиболее распространённой [8].
- В последнее время появились также модели, основанные на использовании чётких автоматов с памятью (ЧАП) [11].

Однако чёткие модели не охватывают всех возможностей применения клеточных автоматов. Поэтому были начаты разработки моделей клеточных автоматов с нечётким управлением, в основе которых лежит нечёткая комбинационная схема (НКС) [12, 13].

Целью данной работы является развитие теории клеточных автоматов, в которых для

обработки массива используются нечёткие комбинационные схемы и нечёткие автоматы с памятью (НАП).

В качестве содержательного примера рассмотрена задача о моделировании нестационарного поля температуры в среде, обладающей тепловой памятью. Как показано в работах [16, 17, 18] уравнение теплопроводности в общем случае содержит слагаемые, зависящие от температуры среды в предшествующие моменты времени. Такое уточнение классической схемы теплопроводности оказывается необходимым для решения задач прецизионного управления температурой высокоточных систем.

Модель нечёткой клеточной комбинационной схемы

Модель НКС применяется в тех случаях, когда предметная область описывается экспертом лингвистическими (словесными) правилами или, когда трудно разработать достаточно простую математическую модель предметной области, и необходима высокая гибкость в настройках системы управления; когда требуется расширить область значений входных параметров «чёткой» комбинационной схемы без введения дополнительных правил и др. Отметим также, что нечёткий клеточный автомат без памяти или нечёткая комбинационная схема, являются разновидностью непрерывного клеточного автомата [13].

Покажем на примере, как может быть построен нечёткий клеточный автомат. Состояние каждой ячейки описывается с помощью лингвистической переменной $\mathcal{Y}$, принимающей ряд значений, например, $\mathcal{Y}_a$ = «*Низкое*», $\mathcal{Y}_b$ = «*Среднее*», ..., $\mathcal{Y}_z$ = «*Высокое*». Эти значения количественно задаются некоторыми нечёткими подмножествами G_a, G_b, G_z универсума *S* с помощью векторных функций $\mu_a(s), \mu_b(s), \mu_z(s)$ (где $s \in S$), определяющих степень принадлежности точек универсума *S* подмножествам G_a, G_b, G_z. Универсальное множество (универсум) состояний *S* ячейки НКС определим как $S = \{0, 1, \ldots, S_{max}\}$.

Вычисления будем производить в матричном виде [19, 20] и для этого вводим компоненты функций принадлежности $\boldsymbol{\mu_a}(s) = (\mu_{a0}, \mu_{a1})$, $\boldsymbol{\mu_b}(s) = (\mu_{b0}, \mu_{b1})$,..., . Истинностные компоненты этих векторов μ_{a1}, μ_{b1} ,... совпадают с функциями принадлежности, введёнными Заде [21], а «ложные» – определяются равенствами $\mu_{a0} = 1 - \mu_{a1}$, $\mu_{b0} = 1 - \mu_{b1}$,... Как показано в [20], переход к векторному описанию нечётких переменных позволяет ввести операции над ними, наиболее естественно обобщающие операции чёткой логики, а также ввести непротиворечивую меру нечёткости, используя известную формулу Шеннона.

Алгоритм работы НКС формулируется на языке лингвистических переменных $\mathcal{Y}$:

$$\mathcal{Y}_i^{t+\Delta t} = \Gamma(\mathcal{Y}_{i-1}^t, \mathcal{Y}_i^t, \mathcal{Y}_{i+1}^t), \tag{1}$$

где аргументы $\mathcal{Y}_{i-1}^t, \mathcal{Y}_i^t, \mathcal{Y}_{i+1}^t$ функции Γ описывают соответственно состояния соседних ячеек с номерами $i-1, i, i+1$ в момент времени *t*, $\mathcal{Y}_i^{t+\Delta t}$ – это состояние ячейки с номером *i* в последующий момент времени $t+\Delta t$ (в формуле (1) для простоты считается, что размер *r* окрестности *0* равен 1). Функция Γ задаётся логическими правилами следующего вида «Если текущие состояния ячеек $\mathcal{Y}_{i-1}^t$ ="*Низкое*", $\mathcal{Y}_i^t$ ="*Низкое*" и ="*Низкое*", то новое состояние ячейки $\mathcal{Y}_i^{t+\Delta t} = \mathcal{Y}_0$ (*"Низкое"*) и т.п.

Правила (1) могут быть записаны в явном виде с помощью логических операций конъюнкции, дизъюнкции и импликации следующим образом:

$$\sum_{k,l,m} \varepsilon_{k,l,m,n} (\mathcal{Y}_{k,i-1}^t \wedge \mathcal{Y}_{l,i}^t \wedge \mathcal{Y}_{m,i+1}^t) \rightarrow \mathcal{Y}_{n,i}^{t+\Delta t}, \tag{2}$$

где индексы k, l, m, n, определяют конкретные значения лингвистических переменных $\mathcal{Y}$, например, $\mathcal{Y}_{i-1}^t$ ="*Низкое*", $\mathcal{Y}_{b,i}^{t+\Delta t}$ ="*Среднее*", а $\mathcal{Y}_{z,i}^{t+\Delta t}$ ="*Высокое*" и т.п. В выражении (2) введена матрица $\varepsilon_{k,l,m,n}$, компоненты которой, в зависимости от значений индексов k, l, m, n, равны 0 или 1. Тем самым задаются правила работы «если ..., то...» рассматриваемой НКС (см. подробнее [19-20]). Символами $\wedge, \rightarrow$ обозначены нечёткие логические операции конъюнкции и импликации соответственно, которые в окончательном результате сводятся к матричным операциям над значениями функций принадлежности $\boldsymbol{\mu_a}(s)$, $\boldsymbol{\mu_b}(s)$,..., $\boldsymbol{\mu_z}(s)$.

Численная реализация работы НКС на основе правил (2) строится следующим образом. В

текущий момент времени t считаются известными параметры s_i^t – текущие состояния каждой ячейки. Цель расчёта – найти параметры $s_i^{t+\Delta t}$ на следующем шаге по времени с помощью нечётких правил (2).

На первом этапе решения этой задачи проведём фаззификацию заданных переменных, вычисляя значения функций $\mu_a(s)$, $\mu_b(s)$,..., $\mu_z(s)$ при $s=s_i^t$. Полученные численные значения подставим в (2) вместо соответствующих значений лингвистических переменных $\mathcal{Y}$.

$$\boldsymbol{F}(s)=\sum_{k,l,m,n}\varepsilon_{k,l,m,n}\left[\mu_k(s_{i-1}^t)\wedge\mu_l(s_i^t)\wedge\mu_m(s_{i+1}^t)\rightarrow\mu_n(s)\right], \quad (3)$$

где введена вспомогательная векторная функция $\boldsymbol{F}(s)$, позволяющая определить новое состояние ячейки $s_i^{t+\Delta t}$.

Для реализации логических операций в формуле (3) используем матричный метод. Логические операции над векторными переменными описаны в [19, 20]: $\boldsymbol{x}\wedge\boldsymbol{y}=C(\boldsymbol{x})\cdot\boldsymbol{y}$ $\boldsymbol{x}\rightarrow\boldsymbol{y}=I(\boldsymbol{x})\cdot\boldsymbol{y}$, где $C(\boldsymbol{x})$ и $I(\boldsymbol{x})$ известные 2×2 матрицы, а точка «·» обозначает матричное умножение квадратной матрицы на матричный столбец, составленный из компонент вектора $\boldsymbol{y}$.

Требуемый для решения задачи расчёт чёткого значения результата $s_i^{t+\Delta t}$ (дефазификация) может быть реализован с помощью найденной функции $\boldsymbol{F}(s)$ «центроидным» методом:

$$\overline{s}=\frac{\int_{all\,S} sF_1(s)\,ds}{\int_{all\,S} F_1(s)\,ds}, \quad (4)$$

где $F_1(s)$ – истинностная компонента векторной функции $\boldsymbol{F}(s)$. Полученный результат вычислений $\overline{s}$ мы отождествляем со значением нового состояния ячейки, в которое она переходит:

$$s_i^{t+\Delta t}=\overline{s}. \quad (5)$$

Далее этот расчёт повторяется для всех ячеек массива.

Моделирование поля температуры

Применим описанный метод для расчёта поля температуры образца при заданных начальных условиях. При разработке алгоритма тепловая задача решалась для теплоизолированного образца (при отсутствии теплового потока на границах области). Подобные требования предъявляются на производстве, где в установке есть несколько точечных источников тепла, а цель управления ими состоит в том, чтобы сформировать с заданный профиль распределения температуры во всём объёме установки [22].

Для обеспечения возможности визуализации результатов ограничимся двумерной задачей, однако все дальнейшие выводы могут быть произведены и для трёхмерной задачи.

Разобьём всю плоскость образца на отдельные ячейки. Количество ячеек выбирается в зависимости от требований к пространственной детализации решения. Для рассмотренного ниже численного примера выбрано поле размером 25 на 25 ячеек.

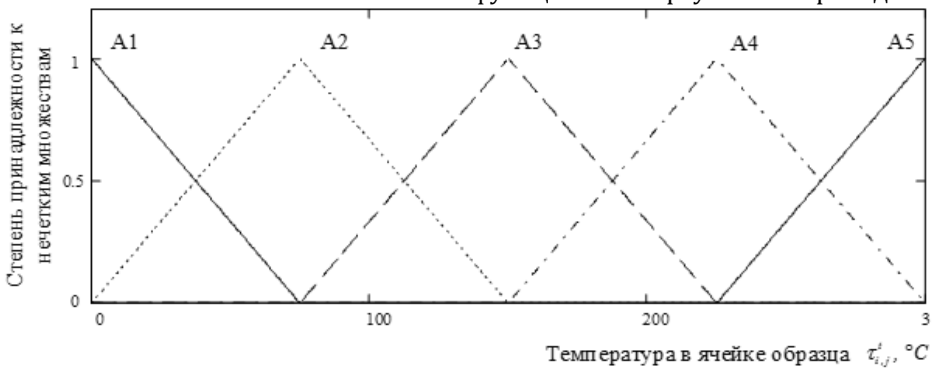
Введём параметры задачи и области их определения (универсальные множества):

- T –температура образца, $T\in[0,300]^0C$.
- $\tau_{i,j}^t$ – температура *i,j*-го элемента образца (ячейки) в момент времени *t*.
- χ – коэффициент теплопроводности образца, $\chi\in[0,400]\frac{Bm}{(м\cdot K)}$ (необходимость введения этого универсума и соответствующих лингвистических переменных обусловлена различием правил изменения температуры образца в зависимости от теплопроводности материала);
- Ψ – относительный коэффициент потерь тепла в третьем измерении (по нормали к плоскости образца) за один шаг по времени Δt в одной ячейке, $\psi\in[0,1]$. Величина Ψ считается равной нулю, если этими потерями можно пренебречь.

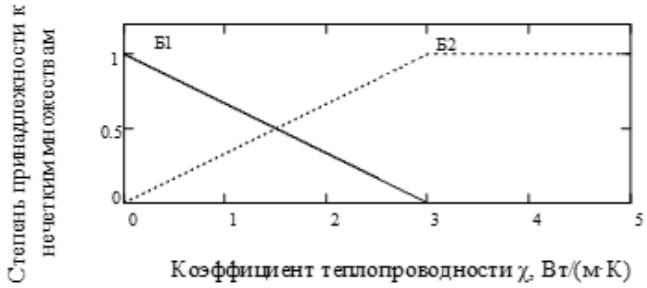
Для алгоритмического описания процессов, происходящих в ячейке, введём следующие лингвистические переменные:

- α – «температура ячейки», принимающая значения: α_a – «очень низкая», α_b – «низкая», α_c – «средняя», α_d – «высокая», α_e – «очень высокая»;
- β – «величина коэффициента теплопроводности» принимающая значения: β_a – «низкий», β_b – «высокий»;
- γ – «величина коэффициента потерь тепла», принимающая значения: γ_a – «низкий», γ_b – «средний», γ_c – «высокий».

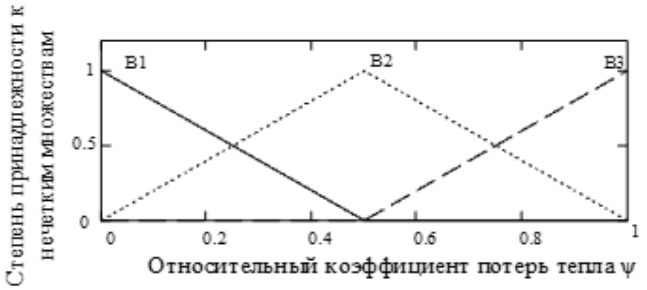
Все термы лингвистических переменных количественно описываются нечёткими подмножествами введённых универсальных множеств. В свою очередь нечёткие подмножества задаются векторными функциями принадлежности $\mu_\alpha(\tau^t_{i,j})$, $\mu_\beta(\chi)$, $\mu_\gamma(\psi)$. Возможный вариант зависимости истинностных частей этих функций от их аргументов приведён на рис. 1.



а)



б)



в)

Рис.1. Графики функций принадлежности элементов универсальных множеств подмножествам, описывающим термы лингвистических переменных: а) «Температура ячейки», б) «Величина коэффициента теплопроводности», в) «Величина коэффициента потерь тепла»,где

А1. «Очень низкая»,
А2. «Низкая»,
А3. «Средняя»,
А4. «Высокая»,
А5. «Очень высокая»,

Б1. «Низкий»,
Б2. «Высокий»,

В1. «Низкий»,
В2. «Средний»,
В3. «Высокий».

Эксперт формулирует правила передачи тепла между ячейками в виде:

«Если в данный момент времени температура ячейки с номерами $i-1, j-1$ $\alpha^t_{i-1,j-1}=\alpha_e$

(«очень высокая»), и температура в ячейке с номерами $i-1, j$ $\alpha^t_{i-1,j}=\alpha_e$ («очень высокая»), и т.д., и коэффициент потерь тепла $\gamma=\gamma_a$ («низкий»), и коэффициент теплопроводности $\beta=\beta_c$ («высокий»), то температура в центральной ячейке (i,j) окружности на следующем шаге будет $\alpha^{t+\Delta t}_{i,j}=\alpha_e$ («очень высокой»). Зависимость между лингвистическими переменными условно выглядит следующим образом:

$$a^{t+\Delta t}_{k,i,j}=\Gamma(a^t_{l1,i-1,j-1}, a^t_{l2,i-1,j}, a^t_{l3,i-1,j+1}, a^t_{l4,i,j-1}, a^t_{l5,i,j}, a^t_{l6,i,j+1}, a^t_{l7,i+1,j-1}, a^t_{l8,i+1,j}, a^t_{l9,i+1,j+1}, \beta_m, \gamma_n). \quad (6)$$

В правилах и формуле (10) анализируется температура в ячейках окрестности Мура первого порядка.

С целью упрощения и унификации дальнейшего алгоритма в данной задаче можно применить дополнительное преобразование. Без внешних воздействий температура во всей системе стремится к равновесному значению. Поэтому, вместо того, чтобы анализировать температуру в каждой ячейке отдельно, можно найти среднюю температуру по окрестности *0* ячейки и анализировать её. Перепишем (6) с учётом того, что $\alpha^t_{l,i,j}$ теперь характеризует среднюю температуру окрестности:

$$a^{t+\Delta t}_{k,i,j}=\Gamma(a^t_{l,i,j}, \beta_m, \gamma_n). \quad (7)$$

В таком виде можно без изменения дальнейшего алгоритма использовать различные формы и размеры окрестностей.

С помощью логических операций правила записываются в следующем виде:

$$\sum_{l,m,n} \varepsilon_{k,l,m,n}(a^t_{l,i,j} \wedge \beta_m \wedge \gamma_n) \rightarrow a^{t+\Delta t}_{k,i,j}, \quad (8)$$

где $\varepsilon_{k,l,m,n}$ – матрица выходов НКС.

Запишем все правила в виде таблицы (табл. 1).

Т а б л и ц а 1. Таблица выходов нечеткой клеточной комбинационной схемы «тепловое поле» на языке лингвистических переменных

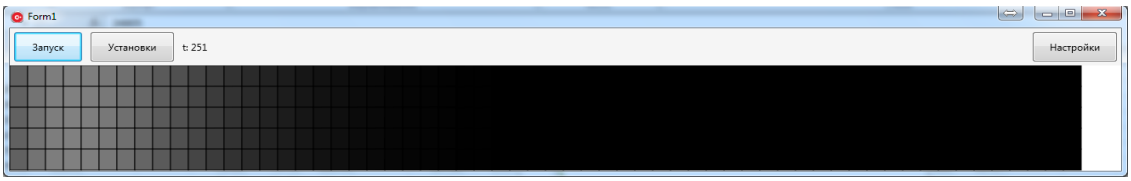
№	l	m	n	k	№	l	m	n
1	a	b	a	a	7	c	b	a
2	b	b	b	a	8	d	b	b
3	c	b	c	a	9	e	b	c
4	b	b	a	b	10	d	b	a
5	c	b	b	b	11	e	b	b
6	d	b	c	b	12	e	b	a

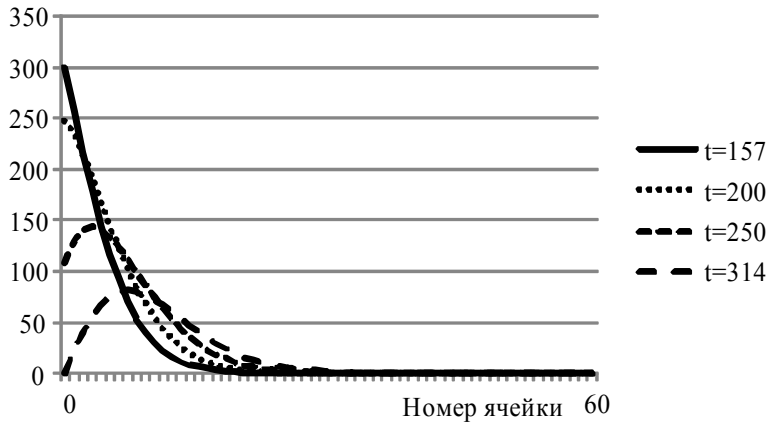
Полученная в соответствии с заданными условиями и алгоритмом (1)-(5) система позволяет проводить анализ различных динамических температурных процессов, в частности, температурных волн и их дисперсии. Произведём расчёт изменения температурного поля в квазилинейном образце при гармоническом граничном условии (9) рис. 2.

$$\tau^t_{i,0}=\tau_{max} \cdot(0.5-0.5\cos(0.02 \cdot t)) \quad (9)$$

а)

б)





в)

Рис.2. Результаты расчётов поля температуры в квазилинейном образце при гармоническом граничном условии (18) в моменты времени а) $t=157\ \cdot\Delta t$ *, б)* $t=251\ \cdot\Delta t$ *, в) поперечные профили температурной волны в некоторые моменты времени.*

Модель нечёткого клеточного автомата с памятью

Классические модели клеточных автоматов (см., например, [2, 8, 9]) подразумевают под собой функционал клеточных комбинационных схем. Фактически хранится и используется только текущее значение исследуемой величины в каждой клетке. При этом история изменения значений в клетках никак не анализируется.

В работе [11] предложена модель «чёткого» клеточного автомата с памятью, в которой для каждой ячейки, кроме значения основной исследуемой величины хранится «состояние», отражающее историю изменения исследуемой величины.

С нашей точки зрения полезной также является модель нечёткого клеточного автомата с памятью (НКАП). По аналогии с «чёткими» клеточными автоматами, НКАП состоит из массива клеток, содержащих информацию о текущем значении исследуемой величины и истории её изменения, а также вычислительное устройство (логический автомат), определяющее переходы клеток между состояниями. Вычислительное устройство представляет собой нечёткий автомат.

Как показано в работе [14], в случае конечной памяти нечёткий автомат можно эквивалентно заменить модифицированной нечёткой комбинационной схемой (МНКС) без потери функциональности. Причём в такой модели не требуется описывать «состояния» ячеек, и достаточно хранить историю изменения исследуемой величины в виде массива-очереди.

Рассмотрим построение такого автомата на конкретном примере. Недавние исследования [16, 17, 18] показывают, что уравнение теплопроводности в общем случае содержит слагаемое, зависящее от температуры среды в предшествующие моменты времени. Иными словами, среда обладает температурной памятью:

$$\boldsymbol{q}(x,t)=-\chi\nabla T(x,t)-\xi\nabla\hat{T}(x,t), \tag{10}$$

Где $\hat{T}(x,t)$ – так называемое температурное смещение

$$\hat{T}(x,t)=\hat{T}(x,0)+\int_0^t T(x,s)ds. \tag{11}$$

Предположим, что в данный момент уже рассчитано предварительное значение температуры в ячейке i,j (часть с коэффициентом χ) по описанному выше (1)-(5) алгоритму. Определим вклад памяти ξ. Для этого найдём среднее за определённое количество шагов по времени значение температуры в ячейке:

$$\bar{\tau}_{i,j}^{t}=\frac{\sum_{s=0}^{L-1}\bar{\tau}_{i,j}^{t-s\Delta t}}{L}, \tag{12}$$

где $\bar{\tau}_{i,j}^{t-s\Delta t}$ – температура в соответствующей ячейке в момент времени $t-s\Delta t$, L – глубина памяти.

Введём лингвистические переменные

- α – «температура в ячейке», принимающая значения: α_a – «очень низкая», α_b – «низкая», α_c – «средняя», α_d – «высокая», α_e – «очень высокая»;
- η – «величина коэффициента влияния памяти вещества», принимающая значения η_a – «низкая», η_b – «высокая».

Графики истинностных частей векторных функций принадлежности $\boldsymbol{\mu}_{\alpha}\left(\bar{\tau}_{i,j}^{t}\right)$ и $\boldsymbol{\mu}_{\eta}\left(\xi\right)$ приведены на рис.1.а, и на рис. 3.

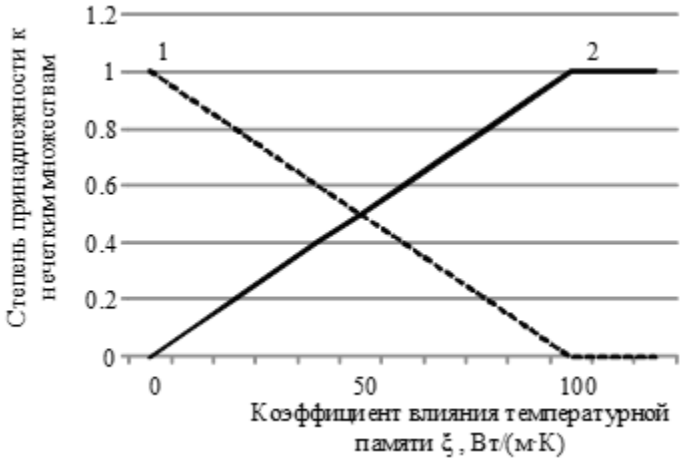


Рис.3. Графики истинностных компонент векторных функций принадлежности значений коэффициента влияния памяти нечётким подмножествам, описывающим термы 1) η_a «низкая» и 2) η_b «высокая» лингвистической переменной «Величина коэффициента влияния памяти вещества»

Задаём лингвистические правила в следующем виде «Если разница средней по времени температуры и текущей температуры α_e «очень высокая», и влияние памяти η_b «высокое», тогда необходимо изменить текущую температуру на величину α_e с учётом знака».

С помощью логических операций правила записываются в следующем виде:

$$\sum_{l,m} e_{k,l,m} a_{l,i,j}^{t} \wedge \eta_m \rightarrow a_{k,i,j}^{t+\Delta t}. \qquad (13)$$

Перечислим все правила в виде таблицы:

Т а б л и ц а 2. Таблица выходов нечеткой модифицированной клеточной комбинационной схемы «тепловое поле» на языке лингвистических переменных

№	l	m	k	№	l	m	k
1	a	a	a	6	c	b	c
2	a	b	a	7	d	a	b
3	b	a	a	8	d	b	d
4	b	b	b	9	e	a	c
5	c	a	b	10	e	b	e

Для численного расчёта строим вспомогательную векторную функцию

$$\boldsymbol{G}^{(i,j)}(\tau) = \underset{k,l,m}{\boldsymbol{D}} e_{k,l,m} C\left[C\left\{\boldsymbol{\mu}_{\alpha l}\left(\bar{\tau}_{i,j}^{t}\right)\right\}\boldsymbol{\mu}_{\eta m}(\xi)\right]\boldsymbol{\mu}_{\alpha k}(\tau). \qquad (14)$$

Чёткое значение добавочного вклада памяти в температуру ячейки с номерами i, j находится центроидным методом:

$$\tilde{\tau}_{i,j}^{t+1} = \frac{\int_{T_{min}}^{T_{max}} \tau \boldsymbol{G}_1^{(i,j)}(\tau)\, d\tau}{\int_{T_{min}}^{T_{max}} \boldsymbol{G}_1^{(i,j)}(\tau)\, d\tau} \qquad (15)$$

Вкладом памяти в модели можно управлять двумя способами:

6. Изменять глубину памяти L. В случае моделирования только по уравнению Фурье $L=1$.
7. Изменять коэффициент влияния памяти ξ.

Построим для сравнения профили волны в квазилинейной среде с гармоническим

граничным условием (9) при $L=1$, $L=2$, $L=5$ и $\xi=100$, рис.4.

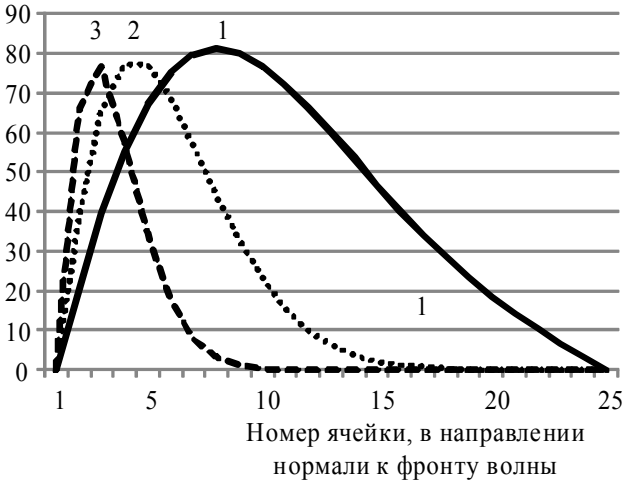


Рис.4. Профили температурной волны в квазилинейном образце с гармоническим граничным условием (17) с параметрами 1) $L=1$ *, 2)* $L=2$ *, 3)* $L=5$ *,* $\xi=100$ *в момент времени* $t=314\ \cdot\Delta t$.

Из рис.4. видно, что наличие памяти делает систему более инерционной.

Выводы

Показано, что задачу расчёта поля температуры в динамических условиях можно решить с помощью нечётких клеточных автоматов без памяти или нечётких комбинационных схем.

Учёт эффекта температурной памяти, предлагаемый как уточнение в современной литературе, можно осуществить с помощью нечётких клеточных автоматов с памятью или нечётких клеточных модифицированных комбинационных схем.

Принципиальное отличие развиваемого подхода от представленных в литературе (см., например, [16, 17, 18]) состоит в том, что здесь не используется какой-либо численной схемы для решения интегро-дифференциального уравнения, к которому сводится эта задача при её описании в указанных источниках. Предлагаемая модель обладает большой гибкостью, так как у разработчика есть возможность увеличивать или уменьшать глубину памяти, изменять как функции принадлежности, которыми описываются значения лингвистических переменных, так и логические формулы, по которым задаётся работа автомата.

Литература

1. Фон Нейман Д. Теория самовоспроизводящихся автоматов. Пер. с англ. М.: Мир, 1971. 384 с.
2. Тоффоли Т., Марголус Н. Машины клеточных автоматов. Пер. с англ. М.: Мир, 1991. 280 с.
3. Беланков А.Б., Столбов В.Ю. Моделирование процесса формирования микроструктуры при кристаллизации металлов с помощью клеточных автоматов. // Математическое моделирование систем и процессов, №10, 2002. С. 6-16.
4. Ahmed E. Fuzzy Cellular Automata Models in Immunology. Journal of Statistal Phisics, 1996, Vol. 85, Pp. 291-294.
5. Zamani S., Javanmard M., Jafarzadeh N., Zamani M. A Novel Image Encryption Scheme Based on Hyper Chaotic Systems and Fuzzy Cellular Automata. The 22nd Iranian Conference on Electrical Engineering (ICEE 2014), May 20-22, 2014.
6. Rosin P., Adamatzky A., Sun X. Cellular Automata in Image Processing and Geometry. Emergence, Complexity and Computation. Springer, Vol. 10, 2014. P. 304.
7. Sridharan R., Pudi V. Design of Arithmetic Circuits in Quantum Dot Cellular Automata Nanotechnology. Springer, 2015. P. 99.
8. Mantelas L., Prastacos P., Hatzichristos T., Koutsopoulos K. Using fuzzy cellular automata to access and simulate urban growth. GeoJournal. Springer, 2012. Pp. 13-28.
9. Mainzer K., Chua L., The Universe as Automaton: From Simplicity and Symmetry to Complexity. Springer. 2012. P. 108.
10. Hoekstra A. G., Kroc J., Sloot P.M.A. Simulating Complex Systems by Cellular Automata. Springer, 2010. P. 393.
11. Salcido A. Cellular Automata - Innovative Modelling for Science and Engineering InTech, 2011. P.440.
12. Alonso-Sanz R., Martın M. Cellular Automata with Memory. Materials of Wolfram Science Conference 2006. // https://www.wolframscience.com/conference/2006/presentations/materials/alonsosanz.pdf (Accessed 09.02.2015).
13. Cattaneo G., Flocchini P., Mauri G., Vogliotti C.Q., Santoro N. Cellular automata in fuzzy backgrounds. Physica D, 1997, Vol. 105, Pp. 105–120.
14. Betel H., Flocchini P. On the Relationship between Fuzzy and Boolean Cellular Automata. Electronic Notes in Theoretical Computer Science, 2009, Vol. 252, Pp. 5–21.
15. Марценюк М.А., Селетков И.П. Приведение конечного нечеткого автомата к нечеткой комбинационной схеме с блоком памяти. // Научно-технические ведомости СПбГПУ. Информатика. Телекоммуникации. Управление. СПб.: Изд-во Политехн. ун-та, 2014. № 6 (210). С. 67-80.

16. Марценюк М. А. Операторно-логические схемы как средство изучения алгоритмов в учебных курсах по математике и информатике. // Прикладная информатика. 2010. №5 (23). С. 43-54.
17. Green A.E., Naghdi P.M. A re-examination of the basic postulates of thermomechanics, Proc. Roy. Soc. London Ser. A 432. 1991. Pp.171–194.
18. Giorgi C, Grandi D., Pata V. On the Green-Naghdi Type III Heat Conduction Model. Mathematics Subject Classification. 2000.
19. Ковалев В.А., Радаев Ю.Н., Семенов Д.А. Связанные динамические задачи гиперболической термоупругости. // Изв. Сарат. ун-та. Сер. Математика. Механика. Информатика, 2009. Т. 9, вып. 4(2). С. 94–127.
20. Марценюк М.А., Поляков В.Б., Селетков И.П. Матричная реализация алгоритмов нечеткого вывода. // Научно-технические ведомости СПбГПУ. Информатика. Телекоммуникации. Управление. СПб.: Изд-во Политехн. ун-та, 2012. № 6 (162). С. 133-141.
21. Марценюк М. А. Матричное представление нечеткой логики. // Нечеткие системы и мягкие вычисления. Научный журнал Российской ассоциации нечетких систем и мягких вычислений. 2007. Т. 2, № 3. С. 7–35.
22. Zadeh L.A. Fuzzy Sets. // Information and Control. 1965, Vol.8, pp. 338-353.
23. Марценюк М. А., А.Ю.Ощепков, А.В.Яценко Синтез адаптивного управления температурными полями распределённых объектов. Теория и эксперимент Вестник Пермского университета Вып, 3. Физика. Пермь.: Из-во Перм. ун-та, 1995. С. 11-43.

Селезнева О.В.

Санкт-Петербургский государственный университет, аспирант факультета прикладной математики-процессов управления кафедры компьютерных технологий и систем

СРАВНЕНИЕ МЕТОДОВ ПРОГНОЗИРОВАНИЯ ТРАЕКТОРИИ МОРСКИХ СУДОВ

КЛЮЧЕВЫЕ СЛОВА

Фильтр Калмана, метод наименьших квадратов, поиск оптимальной траектории, минимизируемый функционал.

АННОТАЦИЯ

Рассматриваются методы прогнозирования траекторий, осуществляется их сравнение и оценка недостатков на примере поиска траектории маневрирующего объекта.

Постановка задачи

Имеется упорядоченный ряд изображений (радиолокационных картин), полученных через фиксированный интервал времени. На одном из них выбирается отметка (сопровождаемый объект). Задача – найти данную отметку на следующих за выбранным изображениях и построить её траекторию движения. Считается, что изображения располагаются в одних координатах (радиолокационная станция неподвижна), количество подвижных отметок ограничено на изображениях (морские суда).

На рисунке ниже приведена траектория, построенная по точкам найденных центров отметки. Как видно по картинке, она получается неровной из-за ошибок позиционирования отметки по причине зашумленности изображений.

Рис.1. – пример траектории, построенной по полученным зашумлённым данным

Также неизвестны параметры ошибки в получаемых геометрических позициях объектов – они могут появляться как из-за неточности аппаратных средств (неточности калибровки РЛС, погрешность аппаратуры, шумы на линии передачи данных), так и из-за погодных условий, общей обстановки и первичных обработок картинок (например, удаления шума или цветокоррекции) с целью нахождения геометрических параметров отметки. Поэтому для дальнейших исследований придётся использовать упрощённые модели движения и ввести ряд допущений.

Дополнительно, в силу ограничений по скорости работы и объёму используемой оперативной памяти, был наложен ряд ограничений при выборе методов.

В качестве возможных решений задачи были рассмотрены два метода – метод наименьших квадратов и упрощённый фильтр Калмана.

Метод наименьших квадратов

Метод наименьших квадратов является простым и эффективным решением для поиска кривой $\tilde{y}(t)$, максимально близко проходящей возле заданных точек y_i. В качестве функции возьмём полином:

$$\tilde{y}(t) = A_0 + A_1 t + A_2 t^2 + \ldots + A_m t^m, \text{где } A_m \neq 0$$

Параметры $A_0, A_1, \ldots, A_m$ будут вычисляться таким образом, чтобы минимизировать функционал:

$$\sum_{i=1}^{m} \sum_{j=1}^{n} \left(\tilde{y}^j(t_i) - y_i^j \right)^2 \to min;$$

где *n* – размерность пространства, в котором происходят вычисления, в нашем случае *n = 2*. *m* – степень полинома, в данной статье рассмотрим случаи с *m = 1* и *m = 2*.

Для линейного случая полином, описывающий искомую функцию, будет выглядеть:

$$Y = At + B;$$

где $Y \in R^2, A = (a_1, a_2), B = (b_1, b_2)$

Параметры a_i и b_i будут вычисляться таким образом, чтобы минимизировать функционал:

$$\sum_{i=1}^{m} \sum_{j=1}^{n} \left(y_j^i - \left(a_j^i t_j + b_j^i \right) \right)^2 \to min;$$

где *m =2*, *n* – количество измерений, участвующих в поиске параметров.

В общем случае будем считать, что цель движется прямолинейно и равномерно, а при резком увеличении значения функционала – что цель произвела маневр, и область поиска следующего положения будет увеличиваться. Минус данного подхода в том, что увеличение области поиска ведёт к резкому увеличению вероятности потерять цель. При поиске параметров функции возьмём последние десять измерений – данное значение было получено эмпирически, с одной стороны оно достаточно большое, чтобы удовлетворять условию на точность поиска коэффициентов, а с другой стороны обеспечивает быстродействие метода при достаточной точности определения маневров.

После нахождения параметров модели, описывающей линию, по которой движется цель, взяв две любые точки на прямой и найдя расстояние между ними и время можно найти среднюю скорость на этом участке. Далее, принимая во внимание постоянство интервалов между изображениями, находится точка, где предположительно окажется цель, а значение функционала и скорость цели определяют размер области с центром в найденной точке, где будет осуществляться поиск нового положения.

Таблица 1. – Значение ошибок для разных целей.

Номер цели	N = 30	N = 40	N = 50	N = 60
1	6864.07	7257.87	8129.73	8438.21
2	6850.12	9109.79	12022.4	14179.8
3	21391.6	23961.2	27354.9	28560.4
4	7684.74	11508.8	14656.9	17017.6

В таблице 1 приведены значения ошибки

$$Err(N) = \sum_{i=1}^{N} \sum_{j=1}^{n} \left(y_i^j - \tilde{y}^j(t_i) \right)^2$$

для четырёх различных объектов. N – количество измерений по которым считается ошибка.

На рисунке 2 показаны две траектории. Фиолетовым цветом изображена траектория, представленная ранее, построенная по центрам найденных отметок. Зелёным цветом – траектория, соединяющая полученные из представленной модели точки, описывающие предсказанное местоположение объекта (предсказанные посредством метода точки). Как видно из рисунка, полученная траектория идёт достаточно близко к той, что постоянна по входным данным. Однако, в момент манёвра траектории расходятся.

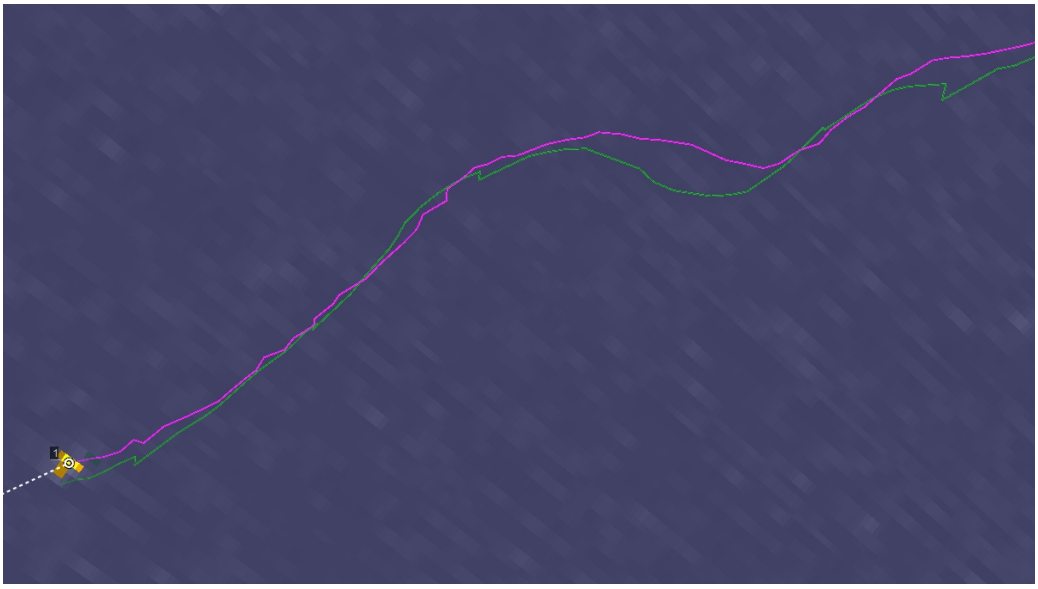


Рис.2 – сравнение траектории, построенной по линейной модели и траектории построенной по центрам отметок

Для квадратичной зависимости полином, описывающий искомую функцию, будет выглядеть так:

$$Y = A\,t^2 + Bt + C\,;$$

Аналогично предыдущему случаю строится минимизируемый функционал:

$$\sum_{i=1}^{m}\sum_{j=1}^{n}\left(y_j^i - \left(a_j^i t_j^2 + b_j^i t_j + c_j^i\right)\right)^2 \to min\,;$$

Таблица 2. – Значение ошибок для разных целей.

Номер цели	N = 30	N = 40	N = 50	N = 60
1	4348.34	4952.51	5751.63	6032.23
2	7600.13	9775.86	12974.8	15011.5
3	22422.0	26318.9	30339.6	32573.6
4	10375.4	15767.8	19907.8	23509.4

В таблице 2 приведены значения ошибки для четырёх различных объектов:

$$Err(N) = \sum_{i=1}^{N}\sum_{j=1}^{n}\left(y_j^i - \tilde{y}^j(t_i)\right)^2.$$

На рисунке 3 представлена траектория, построенная по центрам найденных отметок (фиолетовый цвет) и траектория, соединяющая полученные из квадратичной модели точки, описывающие предсказанное местоположение объекта. В сравнении с предыдущим случаем, новая траектория более точная и гладкая, что свойственно реальной траектории судов.

Фильтр Калмана

Рассмотрим систему:

$$v_{k+1} = v_k + u_k + \xi_k$$
$$z_k = v_k + \eta_k$$

u_k – управляющее воздействие;

v_{k+1} – предсказание состояния системы в *k + 1* момент времени;

z_k – измерение в текущий момент времени;

ξ_k – ошибка модели;

η_k – ошибка сенсора.

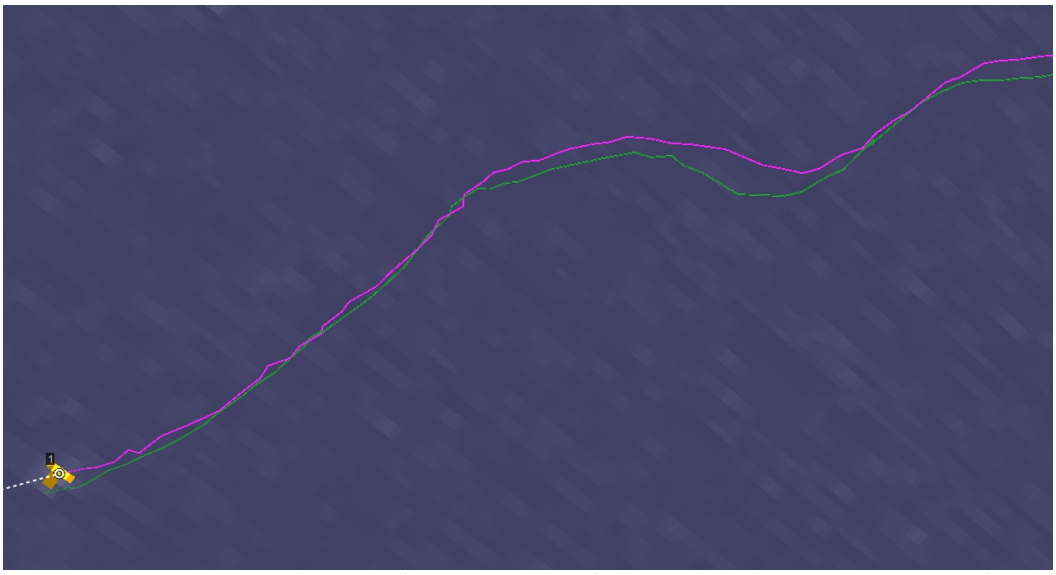


Рис.3 – сравнение траектории, построенной по квадратичной модели и траектории построенной по центрам отметок

В качестве фильтруемой величины возьмём скорость движения объекта. Тогда измерением в текущий момент времени будет значение мгновенной скорости, вычисленное как отношение расстояния между геометрическими центрами объекта и времени, прошедшего между измерениями.

Таким образом, исходная система примет следующий вид:

$$v_{k+1}^{opt} = K_{k+1} \cdot z_{k+1} + (1 - K_{k+1}) \cdot (v_k^{opt} + u_k)$$

K – коэффициент Калмана. После преобразований, имеющих цель найти среднеквадратичное значение от квадрата ошибки e_k (разность оптимального и полученного на выходе фильтра значений) получаем значение коэффициента

$$K_{k+1} = \frac{E\, e_k^2 + \sigma_\xi^2}{E\, e_k^2 + \sigma_\xi^2 + \sigma_\eta^2}$$

Задача сводится к поиску значения *K*. К сожалению, нам ничего не известно о физической модели того, что мы фильтруем, поэтому описанную выше модель упростим до вида

$$v_{k+1}^{opt} = K_{k+1} \cdot z_{k+1} + (1 - K_{k+1}) \cdot v_k^{opt}$$

Эмпирическим путём было установлено что коэффициент K_k с увеличением шага *k*, стабилизируется к некоторому значению K^{opt}. Остаётся найти это значение.

В таблице 3 представлены значения ошибок

$$Err(N) = \sum_{i=1}^{N} \sum_{j=1}^{n} \left(x_{i,j}^{opt} - z_{i,j} \right)^2$$

четырёх различных трасс для разного количества измерений при разном значении параметра *K*. *n = 2* – размерность пространства, в котором происходят вычисления. Как видно в таблице, при фиксированном *K* ошибка монотонна по времени.

Таблица 3. – Значение ошибок для разных целей при разных значениях параметра K. Рис.2 – сравнение траектории, построенной по линейной модели и траектории построенной по центрам отметок

K		N = 30	N = 40	N = 50	N = 60
0.2	1	11537.6	16250.4	39202.3	54205.5
	2	28844.7	39465.1	50868.7	60909.2
	3	60188.4	74967.1	96618.03	112404.0
	4	56437.8	82095.8	106190.03	124862.0
0.3	1	6294.15	8334.53	18028.6	23740.2

	2	20588.2	27563.8	35350.0	41770.1
	3	53636.9	68776.2	97662.1	118610.0
	4	35227.3	50849.1	65077.4	75930.4
0.4	1	5168.35	6432.99	11834.3	14566.2
	2	16731.4	22191.9	28484.7	33402.7
	3	31377.2	35647.8	41229.4	43751.04
	4	27166.7	39247.8	50150.0	60401.6
0.5	1	5085.5	6115.32	9796.2	11525.2
	2	14539.7	19179.5	24656.7	28772.3
	3	26715.7	31620.2	38305.3	41837.0
	4	23353.2	33794.7	43208.7	50924.9
0.6	1	5317.25	6314.12	9207.0	10656.7
	2	13134.1	17259.4	22215.1	25846.2
	3	25549.3	31145.8	38835.4	43408.7
	4	21695.2	32019.2	41392.8	49447.9
0.7	1	5638.02	6681.53	9211.43	10658.0
	2	13199.6	17121.2	21887.8	25267.9
	3	26592.1	32602.4	40733.5	45880.0
	4	19704.6	29086.2	37448.3	44911.6

На рисунке 3, аналогично предыдущим методам, сравнивается значение траекторий, полученных при вычислении геометрических позиций напрямую с картинки и полученных с помощью описанной выше модели. В качестве значения коэффициента Калмана было взято значение *K = 0.5*, которое даёт наименьшее значение ошибки для выбранного объекта.

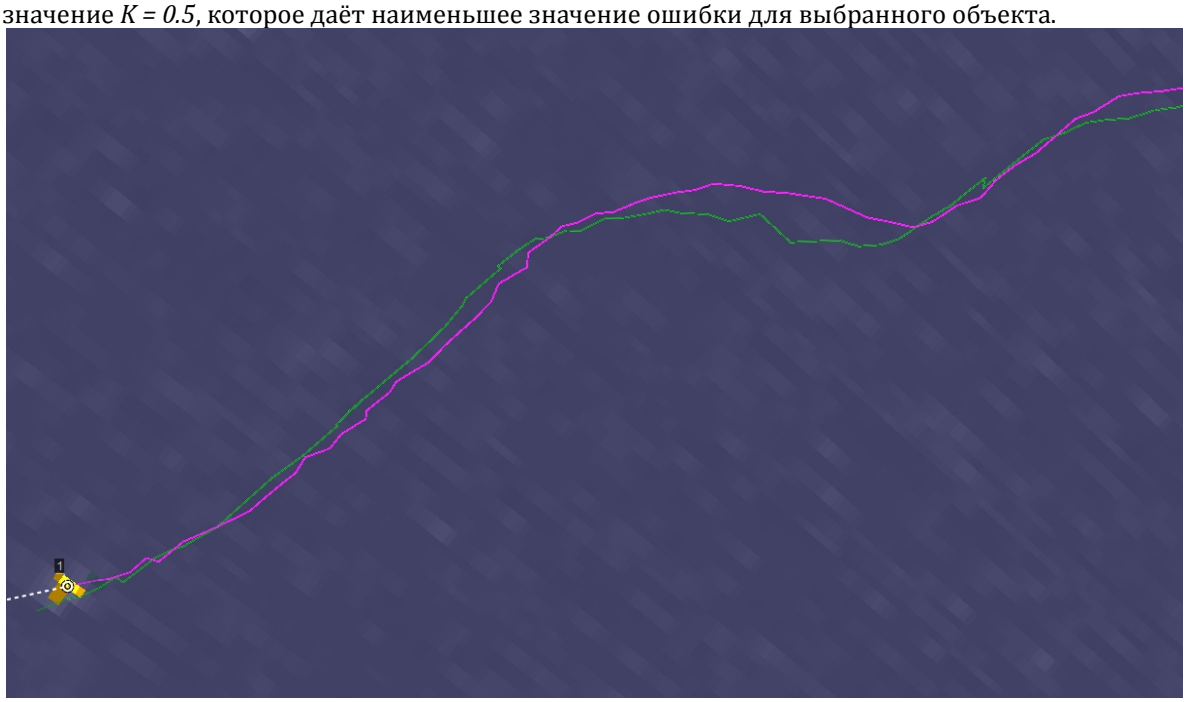


Рис.3 – сравнение траектории, построенной по методу Калмана и траектории построенной по центрам отметок

Минус данного метода заключается в сложности поиска параметра *K*, который также

необходимо пересчитывать при любом изменении параметров системы. Более того, для разных объектов оптимальное значение будет разным (см. Таблицу 3). Минусом также является ломанность построенной траектории. Однако при достаточной вычислительной мощности для определения коэффициентов Калмана, данный метод способен обеспечить большую точность траектории, нежели метод наименьших квадратов.

Итог

Учитывая точность построенной траектории и сложности вычислений был выбран метод наименьших квадратов для квадратичного полинома. Он дал лучшее значение ошибки и обеспечил гладкость траектории по сравнению с методом Калмана.

Литература

1. Линник Ю. В. Метод наименьших квадратов и основы математико-статистической теории обработки наблюдений. — 2-е изд. — М., 1962. (математическая теория)
2. Kalman, R. E. "A New Approach to Linear Filtering and Prediction Problems". Journal of Basic Engineering 82: 35, 1960.
3. Кузьмин С.З. Основы проектирования систем цифровой обработки радиолокационной информации. - М.: "Радио и связь", 1986 - 352 с.
4. Peter, Matisko. Optimality Tests and Adaptive Kalman Filter. 16th IFAC Symposium on System Identification. 16th IFAC Symposium on System Identification. p. 1523, 2012.
5. Fuller, W. A. Measurement Error Models. John Wiley & Sons, 1987.

Плужник Е.В.[1], Никульчев Е.В.[2], Лукьянчиков О.В.[3], Ковшов Е.Е.[4]

[1] Московский технологический институт, первый проректор, e.pluzhnik@gmail.com

[2] Московский технологический институт, д. т. н., профессор, проректор по научной работе, nikulchev@mail.ru

[3] Московский государственный университет информационных технологий, радиотехники и электроники, преподаватель кафедры управления и моделирования систем, luk-it6@yandex.ru

[4] МГТУ «СТАНКИН», д. т. н., профессор, профессор кафедры финансового менеджмента, e.kovshov@stankin.ru

ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ ДЛЯ УПРАВЛЕНИЯ РАСПРЕДЕЛЕННЫМИ ПРИЛОЖЕНИЯМИ В ОБЛАЧНЫХ ВЫЧИСЛИТЕЛЬНЫХ ИНФРАСТРУКТУРАХ

КЛЮЧЕВЫЕ СЛОВА

Облачные технологии, динамическое управление, распределенные архитектуры, программное обеспечение промежуточного слоя, распределенные вычислительные системы.

АННОТАЦИЯ

В статье рассмотрены вопросы реализации динамического управления приложениями, распределенными в облачных инфраструктурах. Выявлены основные особенности облачных приложений и предложены механизмы и технологии реализации системы контроля и управления.

Последние годы облачные вычисления (cloud computing) являются одними из быстро развивающихся информационных сервисов. Это связано, в основном, с востребованностью этой услуги — облачные сервисы берут на себя обслуживание всей ИТ-инфраструктурой, а также ее возможное масштабирование, позволяя обеспечивать надежность приложений при значительном количестве удаленных пользователей. Несмотря на новизну термина, в основе облачных вычислений, особенно на первоначальном этапе, легли давно существующие технологии: системы виртуализации, обеспечение удаленного доступа к данным, элементы GRID-технологий. С точки зрения приложений, облачные системы — обычные клиент-серверные системы, которые ориентированы на обслуживания большого количества клиентов. Особенность заключается в том, что обеспечение требуемого уровня обслуживания пользователей необходимо на программном уровне облачных приложений, более того в программное обеспечение должно быть заложена возможность динамического изменения в зависимости количества и активности пользователей. Отдельной проблемой является обеспечения безопасности.

В настоящее время существует значительные исследования в области поиска эффективных решений для реализации программного управления на основе использования методов теории автоматического управления. В исследовании [1] предложены теоретические, так называемые «брокерские» модели, основанные на динамической раздаче ресурсов через главный, управляющий сервер. В облачных вычислениях предлагается реализовать механизмы цифрового контроллера [2, 3]. Интересно решение по созданию сетевого сервера-контроллера для обработки изображений с использованием облачных вычислений [4]. Разнообразны и формы полученных моделей. В работе [5] предложены нелинейные модели, хотя и для не популярной системы Xen. В целом, общие тенденции исследований таковы, что для облачных систем используются модели автоматизации технологических процесов. Это связано с тем, что с появлением облачных сервисов и увеличения количества удаленных пользователей возникли новые классы задач, ранее не специфичных, для вычислительных систем. Быстродействие программной системы в облаке невозможно определить не только из-за вида запросов, а также из-за используемых пользователем каналов связи, количества одновременного обращения к системе, загруженность ресурсов в конкретный момент времени и проч. В этих случаях необходимо (программно или аппаратно) реализовать систему наблюдения за параметрами и соответственно реализовывать

динамическое управление, а также создание соответствующих исполнительных механизмов, реализующих цель управления.

Для построения качественной облачной системы, необходимо соблюдать ряд основных принципов создания распределенных информационных систем [6]:

- реализация принципов динамического управления;
- использование специальных технологий разработки программного обеспечения;
- механизмы оценки качества управления.

Реализация принципов управления облачной инфраструктурой предложено ввести на основе обратных связей, для которых предлагают различные способы построения систем динамических уравнений. Обеспечение обратными связями возможно средствами гипервизоров, на основе которых строятся современные облачные системы, и от самого программного обеспечения облачных систем, если такие возможности были предусмотрены разработчиками. Введение обратной связи, с одной стороны, позволяет сделать корректирующие действия, с другой — занимает часть вычислительных ресурсов и каналов связи, так что необходима задача оптимального управления, где критерии качества могут свести к минимуму обработку времени или ограниченности ресурсов.

Облачная система строится на виртуальной основе, которая предоставляет удобный способ изменять количество вычислительных ресурсов в зависимости от потребностей. Практически все существующие системы виртуализации обеспечивают средствами, позволяющими следить за основными вычислительными параметрами: загрузкой процессора, оперативной памяти, жесткого диска и сети каждой виртуальной машины.

Проводятся испытания на тестовом стенде, который использует систему виртуализации VMware ESXi [7]. Эта система является гипервизором, который работает напрямую с оборудованием, а не через операционную систему. Для управления предоставляется ряд программного обеспечения администрирования:

1. VMware vSphere;
2. VMware vSphere Web Client;
3. VMware vSphere PowerCLI.

VMware ESXi самостоятельно распределяет нагрузку виртуальных машин, но также содержится ряд надстроек, которые позволяют сделать это вручную.

Для автоматизации управления и разработки собственных методов балансировки нагрузки предназначен функционал библиотек VMware или консоль PowerCLI. PowerCLI — это расширение для Windows Powershell, которое добавляет более 400 новых команд для управления виртуальной инфраструктурой, в том числе и Cloud. Результаты выполнения команд PowerCLI возвращают результаты в формате объектов .NET, что делает удобной разработку средств на языке C# с библиотекой .NET Framework.

Для обеспечения управления системой виртуализации VMware ESXi достаточно следующего набора команд [8]:

1.Connect-VIServer — выполняет подключение к системе виртуализации, после которого можно выполнять все остальные команды;

2.Get-VMHost — получает информацию о хосте, включая его вычислительные характеристики;

3.Get-VM — получает список виртуальных машин и их вычислительные характеристики;

4.Get-Stat — получает журнал загрузки вычислительных ресурсов;

5.Set-VMResourceConfiguration — изменяет параметры пределов и приоритетов распределения вычислительных ресурсов хоста.

Для принятия решения об изменении архитектуры системы и её распределении необходим более широкий спектр собираемых параметров от системы, для чего должны быть предусмотрено реализация механизмов управления на этапе проектирования системы. К параметрам от программного обеспечения можно отнести:

- время выполнения операций (запросов к БД, выполнения функций сервисов);
- частота запроса различных данных;
- географическое размещение пользователей и их активность.

Для имитации гибридной облачной инфраструктуры собран экспериментальный стенд, включающих два сервера и систему коммутации [7]. Сервер, имитирующий публичное облако имеет больше ресурсов, чем сервер, имитирующий приватное облако. Каждый из серверов

представляет собой ферму с виртуальными машинами, управляемыми гипервизорами, дающими возможность оперативного (динамического) управления вычислительными ресурсами сервера (процессорной мощностью). Каждая виртуальная машина развернута на сервере, получает фиксированный объем процессорной мощности в течение некоторой единичной операции, что позволяет избежать нулевых значений ресурсов при значительных нагрузках других виртуальных машин. В соответствии с непредсказуемыми значениями скачков нагрузки ресурсы должны быть разделены между всеми виртуальными машинами в зависимости от требований качества QoS (Quality of Service) для достижения производительности в зависимости от выделенных приоритетов.

Система коммутации представляет собой коммутаторы и маршрутизаторы Cisco, позволяющие динамически управлять каналами связи. В том числе динамически распределять приоритеты по типам трафика.

В качестве базовой модели управления используется система дифференциальных уравнений общего вида:

$$\frac{dy}{dt} = f(y, u, t),$$

где $y = (y_1, \ldots, y_n)^T \in Y \subseteq \mathbf{R}^n$ — вектор контролируемых параметров; t — время; $u = (u_1, \ldots, u_m)^T \in U \subseteq \mathbf{R}^m$ — управление; f — идентифицируемая вектор-функция.

В качестве контролируемого параметрами используются соотношения производительности процессов в зависимости от заданных приоритетов. Подобные параметры были предложены как перспективные подходы достижению QoS для программно-конфигурируемых сетей [9, 10]. Цель управления состоит в распределении ресурсов таким образом, чтобы текущее соотношение производительности соответствовала заданным приоритетным соотношениям в условиях заданных ограничений.

На основании проведенных исследований на экспериментальном стенде, определен круг ситуаций, при которых используемая схема управления не может на данной архитектуре обеспечить гарантированное качество обслуживания. К таким ситуациям можно отнести: добавление нового сервиса в программное обеспечение, ставшего пользоваться большой популярностью, большой рост объема данных, потребовавший увеличения числа виртуальных машин, дополнительных кэш-серверов для гарантированной доставки контента географически удаленным пользователям и т. п. В этих случаях требуется переструктурирование системы, что, в свою очередь, требует переписывание всего программного обеспечения.

Для того что бы систему можно было с легкостью изменить, например для миграции её в гибридную инфраструктуру, необходимо что бы она обладала гибкостью, а именно без особых изменений в программном коде позволяла распределить данные и структуру таблиц по различным СУБД, добавить промежуточные службы, распараллелить расчеты на различных серверах. Для таких изменений необходимо менять технологию промежуточного уровня, что приводит к полному перепрограммированию системы. Для решения данных проблем предлагается технология агентно-реляционного отображения.

Существует несколько основных разновидностей промежуточного слоя:

- технологии распределённых объектов DOT (англ. *distributed object technology*);
- технологии, ориентированные на базы данных (англ. *database access middleware*);
- технологии вызова удалённых процедур RPC (англ. *remote procedure call*);
- технологии, ориентированные на обработку сообщений MOM (англ. *message-oriented middleware*).

Разработчик обычно завязывается на одну или пару технологий промежуточного уровня, строя на них всю систему. Всё удаленное взаимодействие между объектами и обмен данными в системе осуществляется по выбранной концепции. Это ограничивает возможности системы, также зачастую выбирается неэффективное решение реализации функционала не приспособленное технологией, потому что оно менее трудозатратно.

Для решения этих проблем предлагается новая унифицированная технология программирования промежуточного уровня распределенных систем, объединяющая концепции всех технологий промежуточного уровня.

Агентно-реляционное отображение (АРО) — отображение реляционной модели в объектную модель, экземпляры которой имеют возможность удаленного взаимодействия между

собой [11].

В основе технологии, реализующей APO, лежит совместное использование удаленного доступа и концепции ORM (Object-Relational Mapping) [12], а также дополнительные возможности, сформулированные в следующих трех теоретических положениях.

- удаленное взаимодействием между объектами и передача команд осуществляется реляционными командами: выборка (select), изменение (update), добавление (insert), удаление (delete).
- наличие транзакций при выполнении операций.
- хранение информации об удаленных объектах.

В результате технология APO имеет интерфейс технологии ORM, но позволяет выполнять операции не только к БД, но и к удаленным объектам, расположенных на других вычислительных узлах.

В качестве иллюстрации приведем сравнение реализации механизмов взаимодействия с базами данных с использованием технологии APO с сервис-ориентированной архитектурой (см. рис. 1). При построении системы выделен централизованный сервис (на рис. 1 — справа), который будет отвечать за синхронизацию данных, и выступать в качестве инициатора операций.

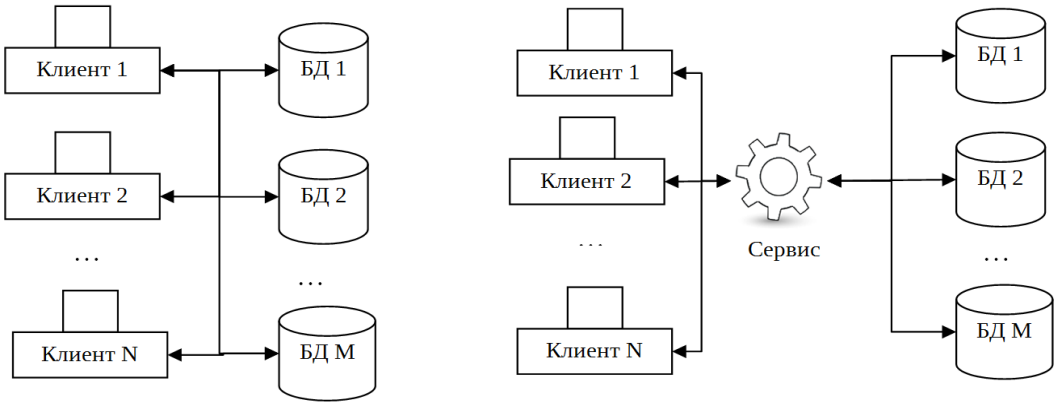
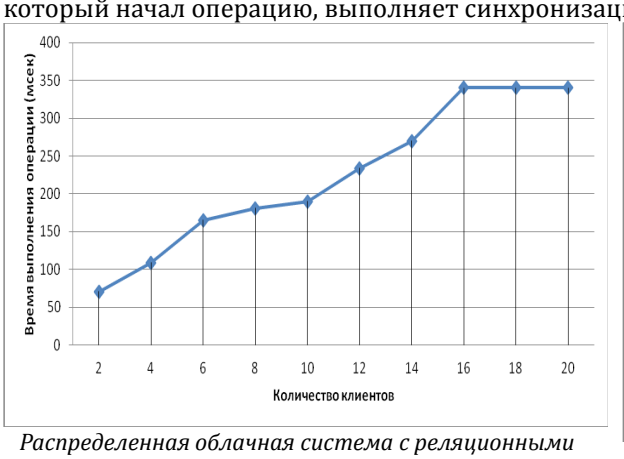


Рис. 1. Распределенная система с реляционными БД и сервис-ориентированная архитектура

В исследуемой системе, присутствует множество клиентских приложений, которые имеют возможность обмениваться удаленно данными между собой, и распределенная БД, как показано на рисунке 4 (где N — количество клиентов в системе, а M – количество БД в системе). В таких системах предпочтительно оперативные данные хранить на клиентах, а долгосрочные данные, которые редко востребованы, хранятся в реляционных БД. При данной архитектуре клиент, который начал операцию, выполняет синхронизацию данных.



Распределенная облачная система с реляционными БД

Сервис-ориентированная система

Рис. 2. Эффективность выполнения операций в зависимости от количества клиентов

В качестве критерия эффективности предложен показатель времени выполнения операций при изменяющемся количестве клиентов.

Экспериментально получены результаты оценки производительности для операции

обновления, показанные на рис. 2.

Важным результатом, является то, что при увеличении числа клиентов в облачной инфраструктуре среднее время выполнения операций остается постоянным.

Проведено сравнение программного кода, приведенной в табл. Данные оценки получены с помощью ПО «SourceMonitor».

Таблица. Оценка эффективности использования технологии APO

Метрики	**С использованием APO**	**Без использования APO**	**Процентное соотношение, %**
Общее количество строк в коде	86	241	35,6
Общее количество операторов в коде	44	147	29,9
Процент ветвящихся выражений	0%	11,6%	0
Общее количество методов в классе	9	18	50
Среднее количество выражений на метод	2	5,3	37,7
Максимальная сложность	1	5	20
Максимальная глубина блоков	1	3	33,3
Средняя глубина блоков	0,68	1,14	59,6
Средняя сложность	1	1,94	51,5

По результатам оценки метриками, оценивающими количество программного кода (общее количество строк в коде, общее количество операторов в коде, среднее количество выражений на метод) эффективность использования технологии около 30%, то есть программисту в среднем требуется писать в 3 раза меньше программного кода. По результатам оценки метриками, оценивающих сложность написанного программного кода (общее количество методов в классе, максимальная глубина блоков, средняя глубина блоков, средняя сложность) эффективность использования технологии около 50%, то есть программный код является более читабельным, проще для понимания, а значит, его проще будет сопровождать и дописывать.

В статье рассмотрены основные принципы контроля и управления облачными системами. Эффективное управление возможно только при наличии механизмов динамического управления, которые можно обеспечить с помощью встроенных возможностей гипервизора или специализированных возможностей программного обеспечения систем, заранее предусмотренных разработчиками. Данные мониторинга параметров, позволяют дать информацию для построения системы правления балансировкой ресурсов и для оценки эффективности работы системы. Также рассмотрены принципы построения программного обеспечения промежуточного слоя, позволяющие реализовать принципы управления.

Литература

1. Lemmon M. D. Towards a passivity framework for power control and responses time management in cloud computing // Proc. 7th Int'l. Workshops on Feedback Computing, 2012.
2. Pluzhnik E., Nikulchev E., Payain S. Optimal control of applications for hybrid cloud services // Proc. 2014 IEEE Tenth World Congress on Services (SERVICES 2014). 2014. P. 458–461.
3. Li Z., Kolmanovsky I., Atkins E., Lu J., Filev D., Michelini J. Cloud aided semi-active suspension control // Proc. 2014 IEEE Symp. on Computational Intelligence in Vehicles and Transportation Systems, 2014. P. 76–83.
4. Wu H., Lou L., Chen C.-C., Hirche A., K¨uhnlenz K. Cloud-based networked visual servo control // IEEE Trans. Industrial Electronics. 2013. V. 60. No. 2. P. 554–566.
5. Patikirikorala, T., Wang, L., Colman, A., Han, J. Differentiated performance management in virtualized environments using nonlinear control // IEEE Transactions on Network and Service Management. 2015. V. 12. No. 1. P. 101–113.
6. Nikulchev E., Pluzhnik E., Biryukov D., Lukyanchikov O.I. Designing applications in a hybrid cloud // Contemporary Engineering Sciences. 2015. T. 8. № 21. С. 963-970.
7. Плужник Е.В., Никульчев Е.В., Паяин С.В. Лабораторный экспериментальный стенд облачных и сетевых технологий // Cloud of Science. 2014. Т. 1. № 1. С. 78-87.
8. https://www.vmware.com/support/developer/PowerCLI/PowerCLI60R1/html/
9. Zhou X., Wei J., Xu C.-Z. Quality-of-service differentiation on the internet: A taxonomy // Journal of Network and Computer Applications. 2007. V. 30. N. 1. P. 354–383.
10. Корячко В.П., Перепелкин Д.А., Иванчикова М.А. Алгоритм адаптивной маршрутизации в корпоративных сетях нескольких провайдеров связи // Вестник Рязанского государственного радиотехнического университета. 2013. № 2 (44). С. 52–56.
11. Лукьянчиков О.И. Технология агентно-реляционного отображения // Современные проблемы науки и образования. 2015. № 1; Режим доступа: URL: http://www.science-education.ru/121-18623.
12. Lukyanchikov O., Payain S., Nikulchev E., Pluzhnik E. Using object-relational mapping to create the distributed databases in a hybrid cloud infrastructure // International Journal of Advanced Computer Science and Applications. 2014. V. 5. N 12. P. 61-64.

Беляков Д.В.

Московский авиационный институт (национальный исследовательский университет), Москва, Оршанская ул., д.3, к.т.н., доцент кафедры Прикладная математика, информационные технологии и электротехника, тел. (495)-141-95-57, pm@mati.ru

ЗАДАЧА ОБ АВТОКОЛЕБАНИЯХ ПЛАСТИНКИ В ПОТОКЕ СРЕДЫ

КЛЮЧЕВЫЕ СЛОВА

Автоколебания, флаттер, пластинка.

АННОТАЦИЯ

Работа посвящена построению и исследованию математической модели автоколебаний аэродинамического профиля в потоке среды.

Постановка задачи

Рассматривается модельная задача о теле, представляющем из себя пластинку, закрепленную с помощью двух упругих элементов и совершающей автоколебания в потоке среды (см. рисунок 1). Введем неподвижную систему координат *XoY*. Будем считать, что в положении покоя пластинка занимает положение равновесия, в котором она ориентирована навстречу потоку. Будем считать, что пластинка может двигаться только по прямой *ВД*, а силы деформации элементов крепления зависят от отклонений линейным образом и сводятся к восстанавливающей силе *F=-kx* и возвращающему моменту $M=-c\vartheta$.

Аэродинамические силы, приложенные к телу приняты в соответствии с эмпирической теорией стационарного обтекания плоской пластины.

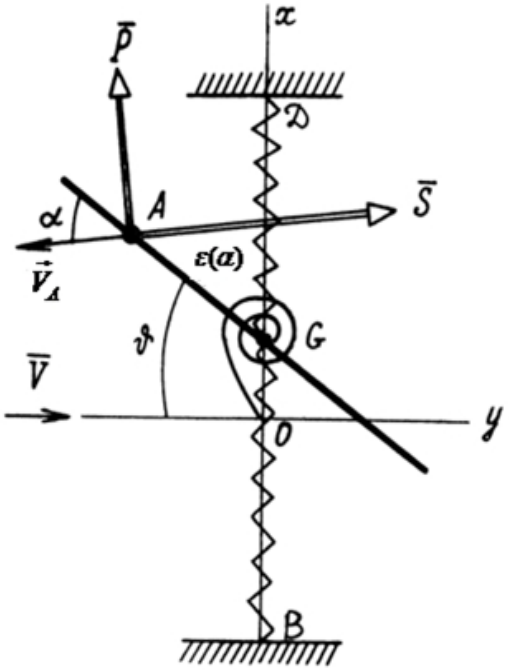


Рисунок 1

В рассматриваемой модели предполагается, что центр давления пластинки точку А можно считать подвижной относительно пластинки. Сдвиг центра давления описывается функцией $\varepsilon(\alpha)$. Зависимость $\varepsilon(\alpha)$ определена из продувок прямоугольных пластинок с заданным удлинением в аэродинамической трубе и является экспериментальными данными [6]. Типичный вид $\varepsilon(\alpha)$ (для удлинения $\lambda=8$) представлен на рисунке 4. Аэродинамические силы, действующие на каждую пластинку, разложим на две составляющие: сила сопротивления $\vec{S}_A$,

направленная против скорости $\vec{V}_A$ точки А относительно потока среды, и подъемная сила $\vec{P}_A$, направленные ей ортогонально. При этом величины аэродинамических сил равны:

$$|\vec{S}_A| = s(\alpha+\delta)V_A^2 = 0.5\rho\sigma c_x(\alpha+\delta)V_A^2,$$

$$|\vec{P}_A| = p(\alpha+\delta)V_A^2 = 0.5\rho\sigma c_y(\alpha+\delta)V_A^2$$

где α - угол атаки между вектором $\vec{V}_A$ и пластинкой p, s - аэродинамические функции углов атаки, c_x, c_y - безразмерные аэродинамические функции (рисунки 3, 4), ρ - плотность воздуха, σ - площадь одной пластинки.

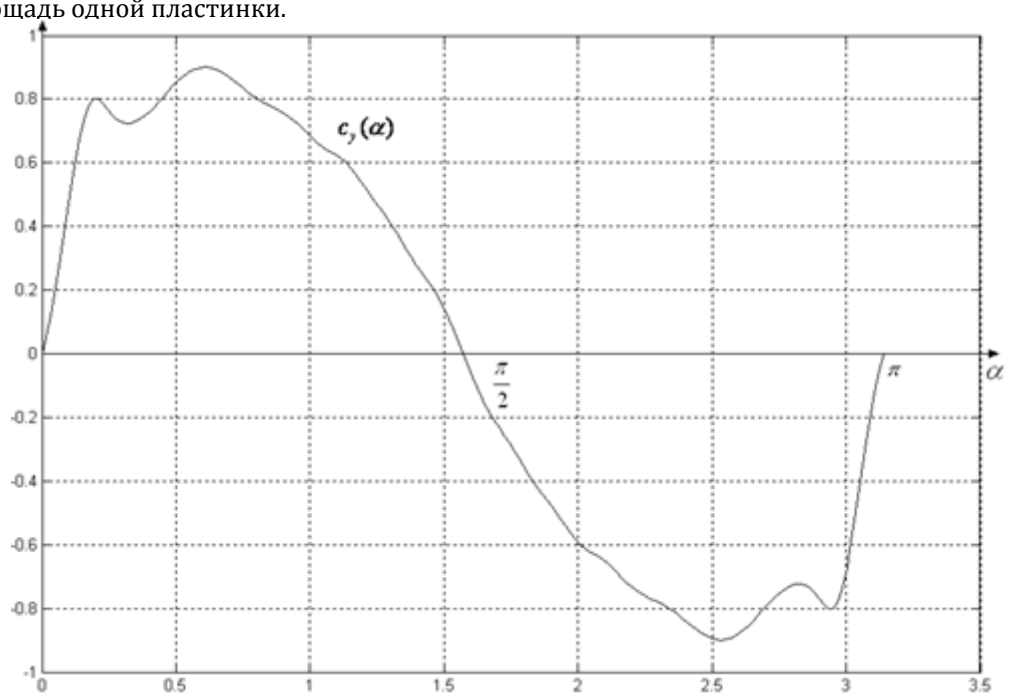


Рисунок 2

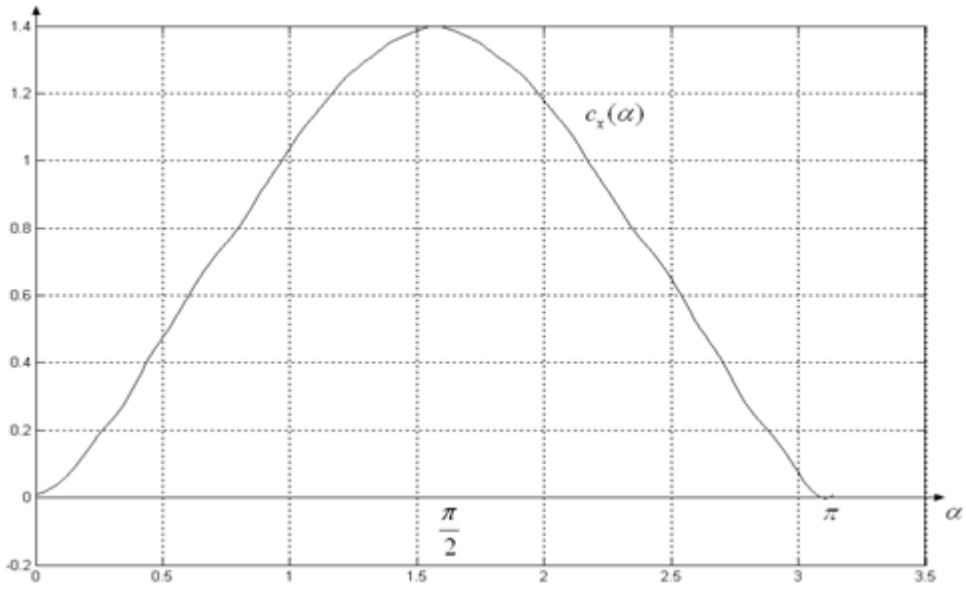


Рисунок 3

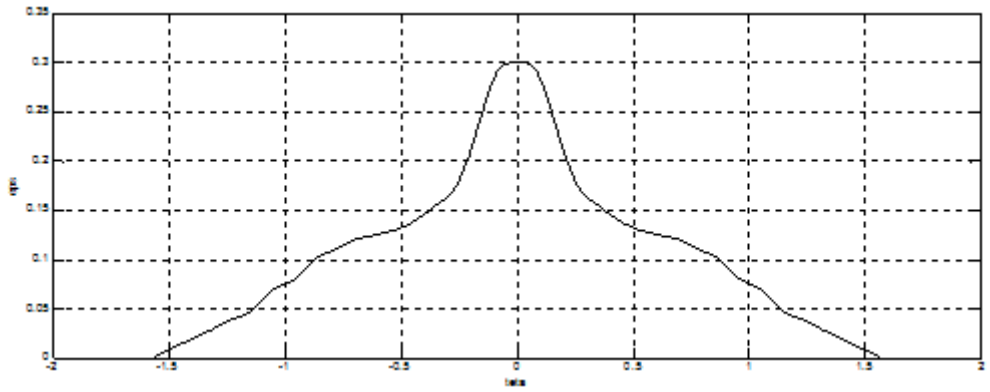


Рисунок 4

Составим уравнения движения рассматриваемого тела. В качестве обобщенных координат, определяющих положение тела, введем координату x центра масс G, совпадающего с серединой пластинки и угол ϑ отклонения пластинки от горизонтали.

Тогда теорема о движении центра масс в проекции на ось *oY* и теорема об изменении кинетического момента будут иметь вид:

$$\begin{aligned} m\ddot{x} &= -s(\alpha)V_A(\dot{x}+\varepsilon(\alpha)\dot{\vartheta}\cos\vartheta)+p(\alpha)V_A(V-\varepsilon(\alpha)\dot{\vartheta}\sin\vartheta)-kx \\ J\ddot{\vartheta} &= s(\alpha)V_A^2\sin\alpha+p(\alpha)V_A^2\cos\alpha-c\vartheta \end{aligned} \quad (1)$$

Кинематические соотношения, связывающие V_A, α с $x, \dot{x}, \vartheta, \dot{\vartheta}$, имеют вид:

$$V_A\cos(\alpha-\vartheta)=V-\varepsilon(\alpha)\dot{\vartheta}\sin\vartheta, V_A\sin(\alpha-\vartheta)=\dot{x}+\varepsilon(\alpha)\dot{\vartheta}cos\vartheta \quad (2)$$

После того, как мы проинтегрируем систему уравнений (1) — (2), мы можем окончательно определить положение тела.

Таким образом, построена математическая модель движения тела, представляющая замкнутую систему уравнений (1) — (2)

Исследование устойчивости положения покоя

Будем считать, что пластинка совершает малые колебания около положения покоя. Исследуем устойчивость тривиального положения равновесия $x=0, \vartheta=0$ по первому приближению. Линеаризуем уравнения движения (1) - (2) при $x\to 0$ и $\vartheta\to 0$.

После несложных преобразований уравнения малых колебаний пластинки примут вид:

$$\begin{cases} m\ddot{x}+kx+(p_0'+s_0)V\dot{x}+p_0'\varepsilon_0V\dot{\vartheta}-p_0'V^2\vartheta=0 \\ J\ddot{\vartheta}+cx+(p_0'+s_0)V\varepsilon_0\dot{x}+(p_0'+s_0)V\varepsilon_0^2\dot{\vartheta}-(p_0'+s_0)V^2\varepsilon_0=0 \end{cases} \quad (3)$$

где $p_0'=p'(0), s_0=s(0), \varepsilon_0=\varepsilon(0)$.

Для исследования устойчивости тривиального положения равновесия $x=0, \vartheta=0$ используем критерий Гурвица. Условия устойчивости имеют вид:

$$\begin{cases} p_0'V(c+k\varepsilon_0^2)>0 \\ p_0'V(\mathrm{J}+m\varepsilon_0^2)>0 \\ c-\varepsilon_0p_0'V^2-s_0\varepsilon_0V^2<0 \end{cases} \quad (4)$$

Видно, что с ростом скорости потока это условие не нарушается.

Сила сопротивления создаёт дополнительный опрокидывающий момент и ухудшает устойчивость пластинки.

Заключение

Таким образом, в работе:

1. Создана математическая модель колебаний пластинки.
2. Получены условия при которых положение покоя пластинки асимптотически устойчиво

Литература

1. Локшин Б.Я. , Привалов В.А., Самсонов В.А. “Введение в задачу о движении точки и тела в сопротивляющейся среде”. Издательство Московского университета. 1992.
2. Беляков Д.В., Самсонов В.А., Филиппов В.В. «Исследование движения несимметричного тела в сопротивляющейся среде». Издательство «МЭИ», журнал «Вестник МЭИ», выпуск № 4 2006 г., стр. 5-10.
3. Беляков Д.В. "Исследование и особенности математической модели движения несимметричного авторотирующего тела в квазистатической среде". Издательство «Новые технологии», журнал "Мехатроника, Автоматизация, Управление". Выпуск № 11. 2007 г., стр. 20-24
4. Самсонов В.А., Беляков Д.В., Чебурахин И.Ф. «Вертикальное снижение тяжелого симметричного авторотирующего тела» в сопротивляющейся среде. Издательство «МАТИ»-РГТУ, сборник «Научные Труды МАТИ», выпуск 9 (81). Москва. 2005. г., стр. 145-150
5. Самсонов В.А., Беляков Д.В. «Математическое моделирование движения симметричного авторотирующего тела, раскрученного до высокой угловой скорости, в воздушной среде». Издательство «МАТИ» -РГТУ, сборник «Научные Труды МАТИ» выпуск 10 (82). Москва. Изд-во МАТИ-РГТУ. 2006 г., стр. 196-200.
6. Табачников В.Г. «Стационарные характеристики крыльев на малых скоростях во всем диапазоне углов атаки.» Труды ЦАГИ 1974 г. выпуск 1621
7. В.Ф. Журавлев Д.М. Климов «Прикладные методы в теории колебаний» Издательство «Наука» 1988 г.
8. В.М. Волосов, Б. М. Моргунов «Метод осреднения в теории нелинейных колебательных систем». Издательство Московского университета. 1971.
9. Н.Н. Боголюбов, Ю.А. Митропольский «Асимптотические методы в теории нелинейных колебаний». Издательство «Наука».1974.
10. И.Г. Малкин «Теория устойчивости движения». Издательство «Наука» 1966 г.
11. А.Н. Тихонов, А.Б. Васильева, А.Г. Свешников «Дифференциальные уравнения». Издательство «Наука» 1985 г.
12. Н.С. Бахвалов Н.П. Жидков Г.М. Кобельков «Численные методы». Издательство Московского университета. 1987.
13. А.А. Самарский, А.В. Гулин «Численные методы». Издательство «Наука» 1989 г.
14. В. Дьяконов «MATLAB 6: учебный курс». Издательский дом «Питер» 2001 г.
15. Paraschivoiu J., “Double Multiple Stremeamtube model with Recent Improvements”, Journal of Energy, vol.7 no.3
16. P. Vittecoq, A. Laneville “The Aerodynamic Forses for a Darrieus Rotor with Straight Blades: Wind Tunnel Measurement.” Journal of Wind Engineering and Industrial Aerodynamics vol.15 Aug-Sept. 1983 pp 381-388.
17. Самсонов В.А. Беляков Д.В. «Математическая модель движения симметричного авторотирующего тела в сопротивляющейся среде.» Юбилейный сборник к 75-летию МАТИ. Изд-во МАТИ-РГТУ. 2007.
18. Д.В. Беляков В.А. Самсонов «Оценка возможностей нового типа ротирующего спускающегося в воздухе объекта». XXVI Академические Чтения по Космонавтике. 2002 г.
19. Беляков Д.В. «Математическое моделирование движения ротирующего спускающегося в воздухе объекта». Труды Пятого Международного Аэрокосмического Конгресса 2006 г.
20. Беляков Д.В. «Математическая модель несимметричного авторотирующего тела в сопротивляющейся среде» XXXШ Международная Молодежная Научная Конференция «Гагаринские Чтения» 2007 г.
21. Беляков Д.В. «Математическое моделирование движения ротирующего спускающегося в воздухе объекта». Пятый Международный Аэрокосмический Конгресс IAC 06. Посвящается 20-летию вывода в космос орбитальнойКонгресс станции “МИР”. Полные доклады. 27-31 августа 2006 г., Москва, Россия. Электронный вид. Рег. номер .
22. Beljakov D.V. Mathematical modelling of movement rotating object going down in air. Fifth International Aerospace Congress IAC'2006. 27-31 August, 2006, Moscow, Russia. SIP RIA, 2006. Full of lecture.Electronic form. Serial number .. . 2007.

Коваленко С.Ю.[1], Кондаков С.А.[2]

[1]Южно-Уральский профессиональный институт, г. Челябинск, старший преподаватель, ЮУПИ, mysife@rambler.ru

[2]Южно-Уральский профессиональный институт, г. Челябинск, к.п.н., доцент, зав. кафедрой МИиВТ, ksa76_2004@mail.ru

КОМПЛЕКСЫ ДЛЯ ПРОВЕДЕНИЯ ЛАБОРАТОРНЫХ РАБОТ

КЛЮЧЕВЫЕ СЛОВА

Лабораторное оборудование, обеспеченность дисциплины, методические пособия, экономическая эффективность, микроконтроллеры.

АННОТАЦИЯ

В статье описаны разработки лабораторного оборудования для проведения лабораторных и практических работ по дисциплинам профессионального цикла.

Из года в год на базе лаборатории «Информатики и вычислительной техники» Южно-Уральского профессионального института разрабатываются программно-аппаратные платформы, которые позволяют в кратчайшие сроки изучить программирование систем, основанные на микроконтроллерах.

Лаборатория служит для формирования практических и исследовательских навыков работы студентов с электронной и вычислительной техникой, автоматизированными и телекоммуникационными системами.

Основные направления деятельности Лаборатории:

- образовательная деятельность, направленная на обеспечение учебного процесса в соответствии с требованиями ФГОС ВПО;
- научно-методическая деятельность, направленная на учебно-методическое обеспечение образовательного процесса и подготовку кадров к использованию лабораторного оборудования;
- научно-исследовательская деятельность, направленная на обеспечение исследовательской работы студентов и преподавателей.

За 2014-2015 год были разработаны 2 платформы, базой у которых стали такие микроконтроллеры как: PIC и STC, но так же в платформу входят и другие блоки:

- жидкокристаллический символьный индикатор 8x2;
- матричную клавиатуру 4x2;
- матричную клавиатуру 4x4;
- часы реального времени PCF8583;
- излучатель звука;
- датчик температуры DS18B20;
- инфракрасный фотоприемник TSOP1736;
- восемь желтых светодиодов;
- микросхему преобразования интерфейсов фирмы FTDI;
- светодиодную матрицу;
- электромеханические реле;
- драйвера двигателя;
- два 32 сегментных индикатора;
- свитчи и переключатели.

Разработанные лабораторные работы позволяют поэтапно изучить все возможности данных микроконтроллеров и блоков индикации и датчиков, а так же язык программирования и среды для разработки программ-прошивок. Не последним вопросов разработки являлась стоимость платформ, этот параметр был ограничен 8 тысячами рублей, то есть вся платформа не должна была превышать бюджет, выделенный на разработку. Для разработки устанавливаются сроки от 3 до 8 месяцев. Платформы не однократно были представлены на всероссийских и

международных выставках, в которых отмечены грамотами, дипломами и медалями. Еще одной особенностью платформ является пункт в техническом задании, в котором сказано, что платформа должна быть мобильной, полностью автономной и с возможностью удаленного контроля процесса выполнения лабораторной работы. Это достигается путем использования независимых источников питания, а так же применения сетевых карт, для обращения к стендам через технологию Ethernet. Все это позволило проводить дистанционные лабораторные работы в центрах повышения квалификации, что делает разработку уникальной в своем сегменте.

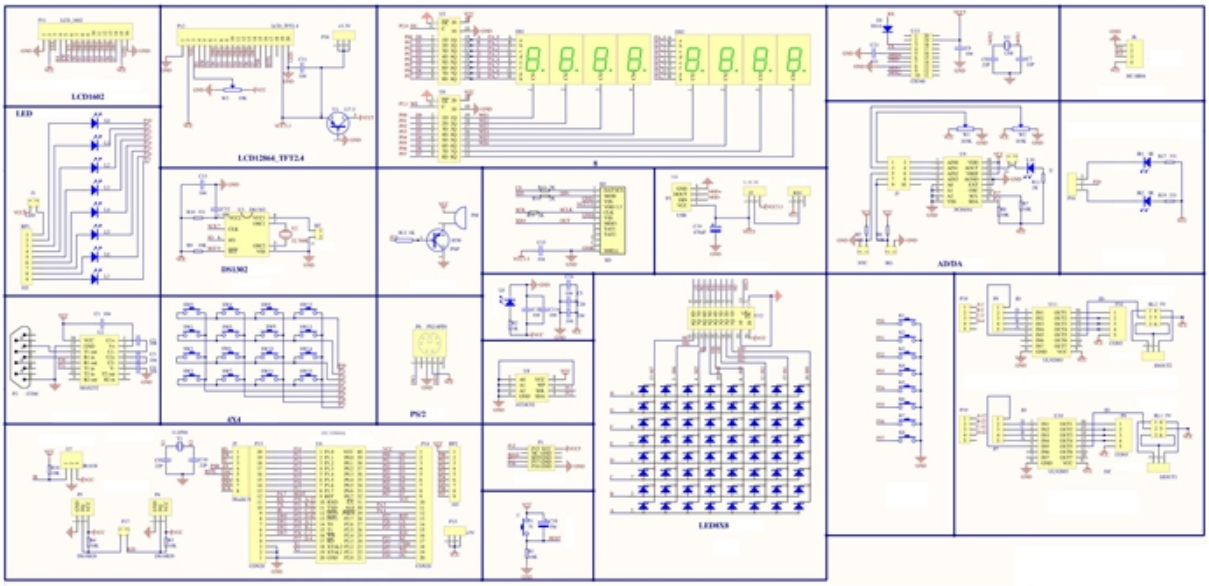


Рис.1 Состав платформы на микроконтроллере STC

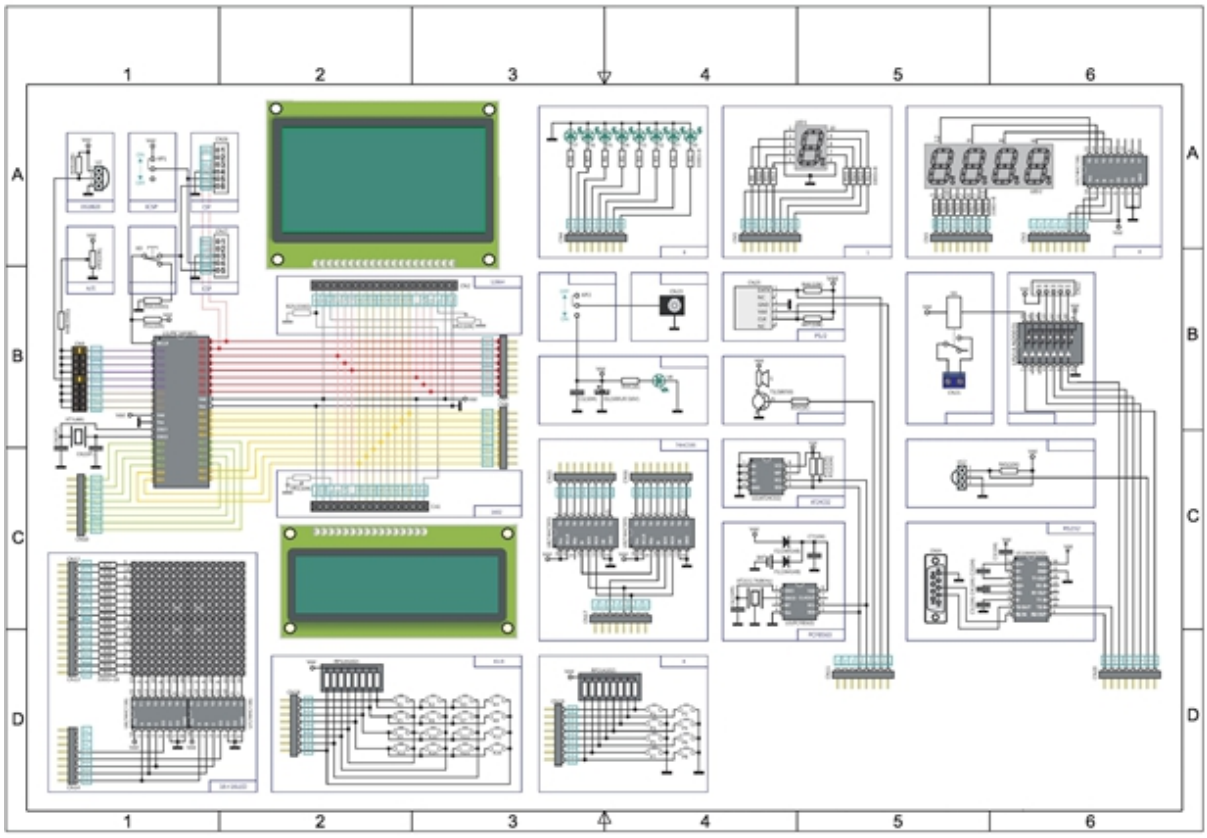


Рис. 2 Состав платформы на микроконтроллере PIC

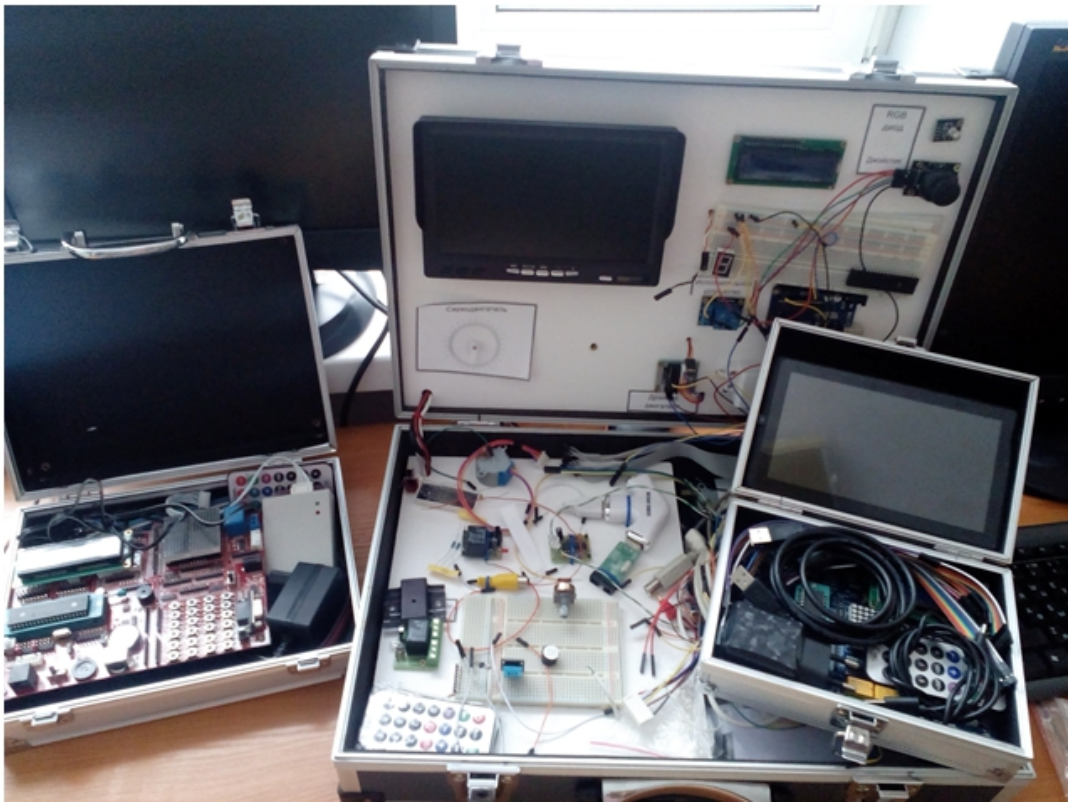


Рис. 3 Внешний вид разработанных платформ

В результате, итоговые платформы не только не превысили допустимый бюджет, но и оказались настолько доступными, порядка 5 тысяч рублей, что позволили оснастить их довольно дорогими датчиками и системами индикации. Программное обеспечение для обеих платформ было взято с официальных сайтов разработчиков микроконтроллеров, оно предоставляется в свободном доступе, и не потребовало вложений и доработок.

Так как разработка и реализация платформ и методических материалов закончилась в феврале месяце, было принято решение внедрить данные платформы в образовательный процесс на дисциплины: микропроцессорная техника в рамках темы: «Микроконтроллеры» и интерфейсы периферийных устройств в рамках тем: «Последовательные и параллельные интерфейсы», что благотворно повлияло на общую успеваемость и итоговые результаты на Государственном итоговом экзамене.

Пока рано делать выводы об использовании данных платформ в образовательной деятельности, но можно заметить повышенный интерес у студентов в освоении тем на реальном оборудовании и, как следствие, повышение успеваемости по дисциплинам с использованием разработанных платформ. Помимо улучшения качественных показателей образовательной деятельности, также при минимальном бюджете улучшается технологическая база кафедры и лаборатории института, что положительно влияет на аккредитационные показатели ВУЗа в целом.

Данные платформы позволяют интерпретировать полученные знания на лекционных занятиях в готовые программно-аппаратные решения для определенных задач, а также наглядно продемонстрировать работу созданного программного обеспечения, в результате написания, которого осуществляется отладка и тестирование. Платформы позволяют обеспечить различную глубину изучения дисциплины (темы) в оптимальных габаритах, не требующих от студентов трудоемкой и опасной коммутации электрических машин, и преобразователей соединительными проводами.

Для получения нужных соединений студенты должны просто произвести включения тех или иных модулей, которые представлены на передних панелях. Модули подобраны таким образом, чтобы максимально охватить технологии, применяемые в современных информационных системах. Разработанные методические материалы в полной мере раскрывают как теоретические аспекты изучаемой темы, так и понятные алгоритмы действий при

лабораторных работах. Тестовые задания позволяют произвести контроль знаний после каждой лабораторной работы, что способствует раннему выявлению сложностей в понятийной тематике и более подробному разбору темы, для достижения понимания в предметной области, а так же производить автоматизированный контроль за успеваемостью и анализировать качество разработанных методических материалов.

Рис. 4 Обеспечение лаборатории оборудованием

Выводы: собственные разработки материально-образовательной оснастки, не уступают рыночным платформам и комплексам в плане раскрытия тем и возможностей применения теоретических знаний, но экономически требуют в десятки раз меньшие затраты, по сравнению с рыночными образцами. Использования разработанных платформ позволило повысить качественную успеваемость и сделать проведение лабораторных работ более ориентированными в прикладном аспекте.

В будущем, планируются разработки платформ под другие типы микроконтроллеров и программируемых реле, а так же интерактивных образовательных стендов по смежным дисциплинам.

Литература

1. Н.А. Дмитриев, М.Н. Ёхин, Б.Н. Ковригин, М.Ю. Павлов. Универсальный лабораторный стенд. Инструментальные средства проектирования и отладки: Учебное пособие / - М.: МИФИ, 2009.

Павлов А.В.

МИРЭА (ТУ), доцент, login11@umail.ru, a_pavlov@mirea.ru)

ФИЛЬТРАЦИЯ И ПРОГНОЗ СМЕЩЕННЫХ НЕСТАЦИОНАРНЫХ ПОСЛЕДОВАТЕЛЬНОСТЕЙ

КЛЮЧЕВЫЕ СЛОВА

Фильтрация, нестационарные последовательности, оптимальное линейное прогнозирование, корреляционная функция.

АННОТАЦИЯ

Для произвольных нестационарных процессов $\xi_k, k\in\dots,-1,0,1,2,\dots$ *приводится разложение аналогичное тождеству Вольда для стационарных процессов. При этом обновляющаяся последовательность ортонормированных случайных величин одна и та же практически для произвольных процессов с ограниченными корреляционными характеристиками и построена ортогонализацией нецентрированных случайных величин с помощью набора нецентрированных корреляционных моментов* $E\xi_i\xi_j$. *Уточняются аналогичные разложения и вид сингулярной компоненты при условии* $E\xi_k^0\xi_m^0=0$ *при всех* $|k-m|>T=const$ *и при условии* $E\xi_{k+r}^0\xi_k^0\to 0$, $r\to\infty$.

Доказано, что при произвольных математических ожиданиях $\xi_k=E_k\neq 0, k=0,\pm 1,\dots,$ *но стационарной корреляционной функции центрированных величин* $E\xi_{k+r}^0\xi_k^0\equiv K(r)=0,1,2,\dots$ *сохраняется в традиционной формулировке тождество Вольда с сингулярной компонентой* $E\xi_m=const$, *коль скоро* $K(r)\to 0, r\to\infty$ *хотя обновляющаяся последовательность построена по набору нецентрированных корреляционных моментов* $E\xi_{k+r}\xi_k$, *зависящих от двух переменных* k, r.

Введение

Работа посвящена оптимальному линейному прогнозированию и фильтрации стационарной или нестационарной случайной последовательности с конечной корреляционной функцией.

Все основные результаты, основанные на теоремах 1-4, базируются на разложениях подобных тождествам Вольда [1,4], доказанным в данной статье для случая стационарных и нестационарных последовательностей (теоремы 1,4), причем в этих теоремах все ортонормированные обновляющиеся последовательности построены с помощью процессов ортогонализации Гильберта-Шмидта в гильбертовых пространствах линейного замыкания [2,3] нецентрированных случайных величин без участия известных или оцененных математических ожиданий, что облегчает применение методов данной статьи в практических задачах прогноза-фильтрации, в том числе, ввиду отсутствия необходимости предварительно оценивать математические ожидания.

Все результаты данной статьи получены без предположения гауссовости рассматриваемых последовательностей.

Обозначим $E\xi_k$ - математическое ожидание ξ_m, $\xi_m^0=\xi_m-E\xi_m$, $E\xi_k^0\xi_m^0=K(m,k), k,m\in\dots,-1,0,1,2,\dots, ||\eta||^2=M\eta^2$. По определению $Pr_{\geq s,\leq r}\xi_k$ - оптимальная в среднеквадратическом смысле линейная оценка ξ_k, полученная проектированием случайной величины ξ_k на $H(s,r)$, $H(s,r)$ - линейное замыкание случайных величин $\xi_i, s\leq i\leq r, r, s\in\dots,-1,0,1,2,\dots, s<r$. По определению $Pr_{\geq -\infty,\leq r}\xi_k=Pr\leq r\xi_k$. $H(r)=H(-\infty,r)$. Соответственно, $H^0(s,r)$ - линейное замыкание

случайных величин $\xi_i^{0,} s \le i \le r$, r, $s \in \ldots, -1, 0, 1, 2, \ldots, s \le r$, и оптимальная в среднеквадратическом смысле линейная оценка центрированной случайной величины ξ_k^0, полученная проектированием на линейное замыкание центрированных случайных величин $H^0(s, r)$, обозначена через $Pr^0_{\ge s, \le r} \xi_k^{0,} Pr^0_{\ge \infty, \le r} \xi_k^0 = Pr^0_{\le r} \xi_k^0$.

В дальнейшем мы будем рассматривать процессы, у которых $K(t) < \infty$, $t \in (-\infty, \infty)$.

Теорема 1.

Для произвольной случайной последовательности $\xi_i, i \in \ldots, -1, 0, 1, 2, \ldots$, такой что $K(t, s) = E\, \xi_t^0\, \xi_s^0 < K_0 < \infty$, $E\, \xi_t < c_2 = const$, при любых целых *t,s,* имеет место:

а) для всех $k = 0, \pm 1, \pm 2, \ldots$,

$$\xi_k = \sum_{m=-\infty}^{k} \delta_m c_{k-m}(k) + \xi_\sigma^k,$$

$$E\, \xi_\sigma^k \delta_i = 0, \quad i = 0, \pm 1, \pm 2, \ldots, \quad E\, \delta_j^2 = 1, \quad E\, \delta_i \delta_j = 0, \text{ для всех } i \ne j$$

$$\xi_\sigma^k \in \bigcap_{j=-\infty}^{k} H(k),$$

и в этой сумме

$$\xi_\sigma^k \equiv 0 \text{ при всех } k \in \ldots, -1, 0, 1, 2, \ldots,$$

если $E_k = E\, \xi_k = 0$, $k = 0, \pm 1, \pm 2, \ldots$;

б) для стационарной в любом смысле последовательности xi_i, $i = \pm 1, \pm 2, \ldots$, если $\lim\limits_{r \to \infty} \sup_{|t-s|>r} K(r) = 0$, $K(r) = K(t + r, t)$, и $K(r_1) \le K(r_2)$ при всех $r_1 < r_2$, то для всех целых *k* имет место :

$$\xi_k = \sum_{m=-\infty}^{k} \delta_m c_{k-m} + E, \quad E = E\, \xi_i,$$

причем во всех этих равенствах ортонормированная последовательность $\{\delta_k\}$ построена специальным процессом ортогонализации нецентрированных векторов ξ_i, $i = 0, \pm 1, \ldots$.

Доказательство проводится методами гильбертовых пространств без применения известных традиционных методов теории вероятности.

Нам понадобится обозначение линейных оценок

$\xi_k(k-R) = Pr_{\le k-R} \xi_k$, $R > 1$, $\xi_k = Pr_{\le k-1} \xi_k$, $\xi_k(k-R, k-S) = Pr_{\ge k-R,\ \le k-S} \xi_k$, $0 \le R \le S \le \infty$.

Теорема 2.

Пусть $\xi_i, i \in \ldots, -1, 0, 1, 2, \ldots$, – произвольный стационарный в каком-либо смысле процесс ($K(t) < \infty$ при всех *t*).

Если $K(t) = 0$ при всех $t > T = const$, $T \in 1, 2, \ldots$, то для произвольного целых *k* и $R > T$,

1.

$$\hat{\xi}_k(k-R) = E = E\xi_k,$$

2.

$$\hat{\xi}_k = E + c_T \delta_{k-T} + \ldots + c_0 \delta_k =,$$

$$\hat{\xi}_k = c_T \delta_{k-T} + \ldots + c_0 \delta_k + \hat{\xi}_k(k-T, k-S) + 0(1),\ S \to \infty, \hat{\xi}_k(k-T, k-S) = E + 0(1), S \to \infty,$$

$$E\, \delta_{k-j} = 0, \quad j = 1, \ldots, T.$$

Доказательство.

Доказательство опирается на методы, развитые в теореме 1.

Замечание.

Для простоты формулировки мы привели теорему 2 в стационарном случае. При доказательстве этой теоремы мы не пользуемся стационарностью последовательности ξ_k. В общем случае имеет место следующее утверждение: в условиях пункта b) теоремы 1 имеют место все утверждения теоремы 2 с теми же обозначениями с заменой E на $E_k = E\xi_k$.

Для доказательства следствия 1 (в нем для простоты рассматривается стационарная последовательность, [4]) нам понадобится лемма 1.

Лемма 1.

Если $Pr_{\geq k-T,\leq k}\xi_k^0$ - проекция вектора ξ_k^0 на $H(k-T,k)$ ($H(k-T,k)$ - подпространство порожденное, вообще говоря, нецентрированными векторами $\xi_{k-T},\dots,\xi_k$), то

$$Pr_{\geq k-T,\leq k}\xi_k^0 \in H(\delta_{k-T},\dots,\delta_k),$$

где $H(\delta_{k-T},\dots,\delta_k)$ - подпространство, порожденное векторами $\delta_{k-T},\dots,\delta_k$.

Следствие 1. $Pr_{\geq k-T,\leq k}\xi_k^0 = c_T\delta_{k-T}+\dots+c_0\delta_k$.

Приведем полезное для сравнения дискретных методов с методами, основанными на интегрировании траекторий следствие 2.

Следствие 2. В условиях пункта б) теоремы 1, если $S(N)$ - сумма коэффициентов оптимальной в средне-квадратическом оценки ξ_k по некоррелированным с ней наблюдениям $\xi_{k-T-1},\dots,\xi_{k-T-R}$, то $S(N)\to 1$.

Доказательство.

По пункту б) теоремы 1 из сходимости оценки в средне-квадратическом к E следует сходимость математических ожиданий [2], что эквивалентно утверждению следствия 2.

Из теоремы 1 непосредственно следует теорема 3, в которой рассмотрена корреляционная функция, стремящаяся к нулю на бесконечности.

Теорема 3.

Если $\xi_i, i\in\dots,-1,0,1,2,\dots,$ - произвольный стационарный в любом смысле процесс, такой что

$$\lim_{r\to\infty} K(r)=0, K(r)=K(t+r,t),$$

то при любом $k=0,\pm1,\dots$ имеет место:

1. $\lim_{R\to+\infty}\hat{\xi}_k(k-R)=E=E\xi_k$,
2. $\hat{\xi}_k=\sum_{m=-\infty}^{k-1}\delta_m c_{k-m}+E$,

где последовательность ортонормированных величин $\{\delta_i\}$ построена ортогонализацией нецентрированных величин $\{\xi_i\}$ без предварительной оценки математического ожидания E.

Доказательство.

В теореме 1 мы доказали, что

$$\xi_k=\sum_{m=R}^{k}\delta_m c_{k-m}+Pt_{\leq k-R}\xi_k, \|\delta_i\|=1, \delta_i\perp\delta_j, \text{ для всех } i\neq j,$$

$$\lim_{R\to+\infty} Pr_{\le k-R}\xi_k = E\ .$$

Данные утверждения эквивалентны утверждениям теоремы 3.

При разложении и прогнозировании последовательностей со стационарной корреляционной функцией и произвольными математическими ожиданиями используется теорема 4.

Теорема 4.

Если $E\xi_m = E_m < \infty$, $m = 0, \pm 1, \pm 2, \ldots$, $E\xi^0_{m+R}\xi^0_m = K(R)$, $R = 0, 1, \ldots$, $\lim_{R\to\infty} K(r) = 0$, то теорема 3 верна в дословной формулировке с заменой E на E_k причем "обновляющая" последовательность δ_i, $i = 0, 1, \ldots$ построена процессом ортогонализации нецентрированных векторов $\{\xi_i\}$ и одна и та же для разных k (хотя $E\xi_{m+R}\xi_m$ зависит, вообще говоря, от двух переменных m и R).

Литература

1. В.С.Королюк, Н.И.Портенко, А.В.Скороход, А.Ф.Турбин. - М.: Наука. Главная ред.физ.-мат.литер.1985. 640 с.
2. Венцель А.Д. Курс теории случайных процессов. - М.: Наука, 1975. 320 с.
3. Колмогоров А.Н., Фомин С.В. Элементы теории функций и функционального анализа. М.:Наука.Гл.ред.физ.-мат.лит., 1976. -544 с.
4. Павлов А.В. Случайные ряды Фурье и их применение к теории фильтрации-прогноза М.:Изд-во МГУ им. Ломоносова, механико-математический ф-т, 2000.-64 с. ISBN 5-93839-002-8.
5. Павлов А.В. Теорема типа больших уклонений для критерия хи-квадрат. М.: Успехи мат.наук. Т.51, 1 (307), 1996.

БОЛЬШИЕ ДАННЫЕ И ИХ ПРИЛОЖЕНИЯ

Королев С.А.[1], Селиверстов А.В.[2], Зверков О.А.[3], Любецкий В.А.[4]

[1]ИППИ РАН, г. Москва, стажер-исследователь, korolev@iitp.ru

[2]ИППИ РАН, г. Москва,к.ф.-м.н., в.н.с., slvstv@iitp.ru

[3]ИППИ РАН, г. Москва, к.ф.-м.н., н.с., zverkov@iitp.ru

[4]ИППИ РАН, г. Москва, д.ф.-м.н., зав.лаб., lyubetsk@iitp.ru

КЛАССИЧЕСКАЯ АТТЕНЮАТОРНАЯ РЕГУЛЯЦИЯ, ЗАВИСИМАЯ ОТ КОНЦЕНТРАЦИИ ТРИПТОФАНА, У АКТИНОБАКТЕРИЙ

КЛЮЧЕВЫЕ СЛОВА

Актинобактерии, классическая аттенюаторная регуляция, лидерный пептид, моделирование транскрипции.

АННОТАЦИЯ

Предсказаны и подтверждены моделированием новые случаи классической аттенюаторной регуляции, зависимой от концентрации триптофанил-тРНК, у актинобактерий. Показано, что экспрессия генов, кодирующих ферменты пути синтеза триптофана, регулируется аттенюаторной регуляцией только у видов двух родов Corynebacterium и Streptomyces. Обсуждается совершенствование методов моделирования аттенюаторной регуляции, вовлекающей псевдоузлы и триплексы на РНК.

Лидерные гены играют ключевую роль в аттенюаторной регуляции, основанной на сопряжении транскрипции и трансляции у прокариот. Впервые такая регуляция была описана для *Escherichia coli* [1], но также известна и для актинобактерий. В частности, экспериментальное подтверждение такой регуляции, зависимой от концентрации триптофана, получено для *Corynebacterium glutamicum* [2] и *Streptomyces venezuelae* [3]. Широкомасштабный биоинформатический поиск проведен в работах [4–6].

Аттенюаторная регуляция определяется структурой, которая включает ген лидерного пептида с регуляторными кодонами в нём и связанные с ним альтернативные вторичные структуры мРНК, одни из которых приводят к преждевременной терминации транскрипции структурного гена, а другие позволяют РНК-полимеразе продолжать транскрипцию структурного гена. Эта альтернатива зависит от скорости, с которой рибосома выполняет трансляцию лидерного пептида. В свою очередь скорость зависит от концентрации аминоацил-тРНК, которая в свою очередь зависит от концентрации соответствующей аминокислоты и аминоацил-тРНК синтетазы.

Нами проведён поиск лидерных генов перед всеми размеченными в аннотациях генами 196 геномов актинобактерий, доступных в базе данных NCBI. При этом лидерные пептиды, кодируемые лидерными генами, содержат три подряд или две близкие пары остатков триптофана. Далее гены, отмеченные в аннотации, называются структурными. Предсказанные нами лидерные гены не перекрывают структурные гены и обычно имеют небольшую длину.

Моделирование классической аттенюаторной регуляции выполнено нашей программой, описанной в [7].

На рисунке 1 показана зависимость числа предсказанных лидерных генов с кодонами триптофана от расстояния между лидерным и структурным генами у актинобактерий.

Классическая аттенюаторная регуляция, зависимая от концентрации триптофана или триптофанил-тРНК, предсказана только у некоторых видов двух родов *Corynebacterium* и *Streptomyces*. При этом расстояние между лидерным и структурным генами колеблется в пределах от 50 до 296 п.н. У многих штаммов коринебактерий *C. pseudotuberculosis* и *C. ulcerans* лидерные гены предсказаны перед обоими генами *trpB* и *trpE*. У штаммов *C. diphtheriae* – только перед опероном, включающим *trpB1* и другие гены пути синтеза триптофана. У *C. aurimucosum* лидерные гены предсказаны перед геном *trpS*, кодирующим триптофанил-тРНК синтетазу, и большим опероном, содержащим ген *trpE* и другие гены пути синтеза триптофана, включая *trpB* и *trpA*. У

Streptomyces гены антранилат синтазы, предположительно регулируемые классической аттенюацией, не входят в единый оперон с другими генами пути синтеза триптофана. Ранее нами исследована потенциальная аттенюаторная регуляция гена *phzE* (называемого также *trpE1*), кодирующего антранилат синтазу [4], перед которым нами предсказан лидерный ген, содержащий три подряд кодона триптофана. У близкого вида *S. cattleya* перед геном *phzB*, кодирующим антранилат синтазу, также предсказан лидерный ген. Аналогичная картина наблюдается у *S. scabiei* и *S. coelicolor* для гена *trpE*.

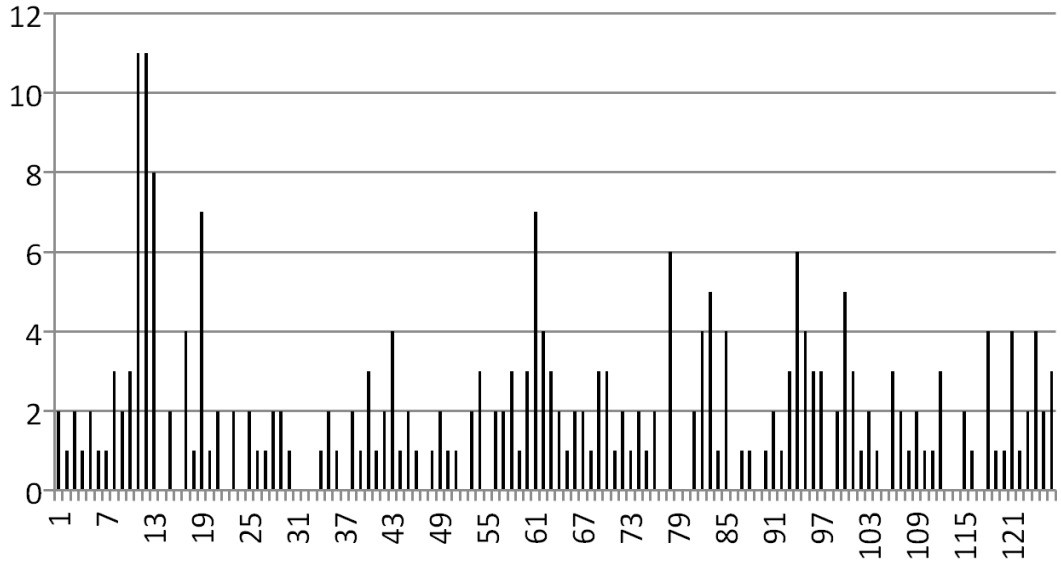


Рис.1. Зависимость числа предсказанных лидерных генов с кодонами триптофана от расстояния между лидерным и структурным генами у актинобактерий

Выполненное нашими методами моделирование подтверждает эффективность классической аттенюации для генов *trpE* и *trpB* у *C. pseudotuberculosis* и гена *trpB1* у *C. diphtheria* 31A. Результат моделирования показан на рисунке 2. Однако такая регуляция гена *trpB1* не подтверждается для штамма *C. diphtheriae* INCA 402.

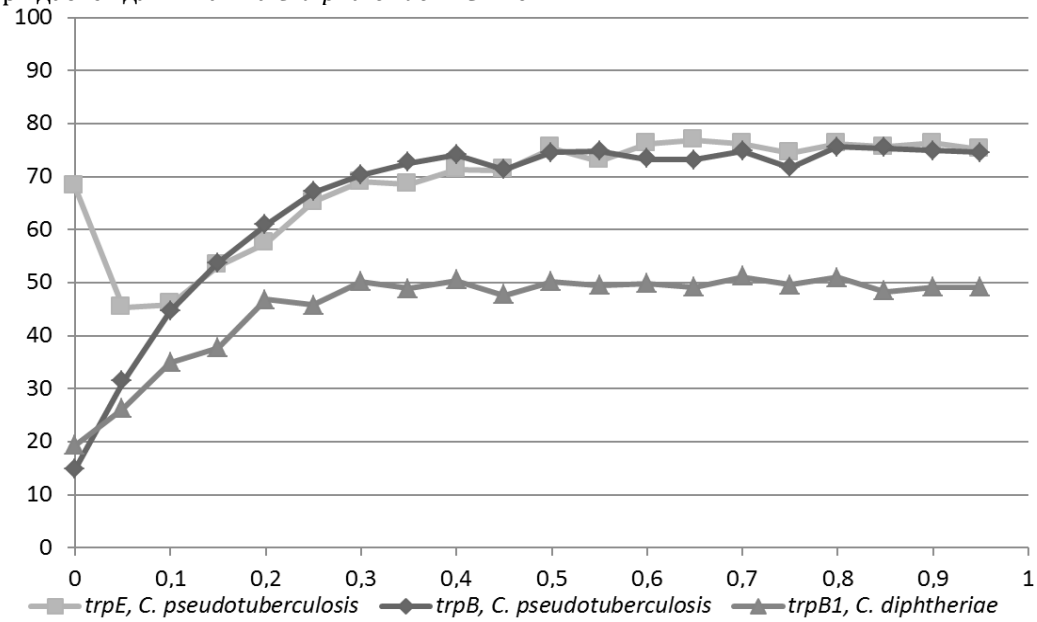


Рис.2. Зависимость частоты терминации транскрипции до начала транскрипции структурного гена от концентрации триптофана в модели для генов trpE и trpB у C. pseudotuberculosis, гена trpB1 у C. diphtheria 31A.

Классическая аттенюаторная регуляция, зависимая от концентрации триптофанил-тРНК предсказана для видов двух родов *Corynebacterium* и *Streptomyces*. Эти предсказания хорошо согласуются с экспериментами [1–2]. Отсутствие такой регуляции у других видов может быть

связано с регуляцией на другом уровне. Некоторые из ферментов пути синтеза триптофана содержат на N-конце домен PF04715, вовлечённый в ингибирование фермента триптофаном [8].

Например, в геноме *Streptomyces avermitilis* MA-4680 (GenBank: NC_003155) есть два гена, предположительно вовлечённых в синтез антранилата из хоризмата, которые имеют низкое сходство между собой. Первый ген *trpE* (с координатами 7417784..7419265) кодирует короткий белок длиной 493, который содержит N-концевой домен PF04715 и домен PF00425, связывающий хоризмат. Второй ген *phzE* (или *trpE1* с координатами complement(7320283..7322268)) кодирует длинный белок, содержащий домен PF00425, связывающий хоризмат, и домен PF00117, характерный для глутаминамидотрансфераз, но не содержит домена, участвующего в ингибировании триптофаном. Третий ген *pabB* (с координатами 8122893..8124014) и четвёртый *pabAB* (с координатами complement(1468399..1470597)). Последний ген также кодирует белок с N-концевым доменом PF04715. Домены определены по базе данных Pfam [9]. Координаты указаны по данным из GenBank, доступным по адресу http://www.ncbi.nlm.nih.gov/nuccore/148878541. Ранее нами исследована потенциальная аттенюаторная регуляция гена *phzE* (называемого также *trpE1*) [4], перед которым нами предсказан лидерный ген, содержащий три подряд кодона триптофана. Почему такая регуляция не найдена перед генами *trpE* и *pabAB*? Одно из возможных объяснений в том, что эти гены кодируют ферменты, которые ингибируются триптофаном. Тогда как фермент, кодируемый геном *phzE*, не ингибируется, поскольку не содержит PF04715 на N-конце. У близкого вида *Streptomyces cattleya* антранилат синтаза, кодируемая геном *phzB*, содержит домены PF00425 и PF00117, но не содержит домен PF04715. Перед этим геном также предсказан лидерный ген. Аналогичная картина наблюдается у *Streptomyces scabiei* (ген *trpE*). У *Streptomyces* гены антранилат синтазы, предположительно регулируемые классической аттенюацией, не входят в единый оперон с другими генами пути синтеза триптофана.

Отметим, что у актинобактерий известны и другие типы аттенюаторной регуляции. Один из них, вовлекающий белок Rho, описан в работе [10]. Но и случаи аттенюации транскрипции, не связанные с дополнительными факторами, могут значительно отличаться от классической аттенюации. Структура РНК может включать псевдоузлы и триплексы [5], существенно затрудняющие моделирование регуляции. Разработка новых алгоритмов, учитывающих эти эффекты, позволяет уточнить список регулируемых генов и эффективность предсказанной регуляции.

Напомним, что триплексы – это три участка РНК, один из которых состоит полностью из пуриновых нуклеотидов, а два других участка имеют относительно пуринового участка параллельную и антипараллельную ориентации. Пара антипараллельных участков образует спираль РНК, в которой комплементарными считаются уотсон-криковские пары и GU пара. Таким образом, два участка триплекса образуют обычную спираль, а еще один, параллельный, участок связан с ней хугстиновскими водородными связями. Последний участок называется третьим плечом триплекса, которое, как и составляющие его нуклеотиды, в записи отделяется знаком * от самой спирали (от ее нуклеотидов). В триплексе возможны триады нуклеотидов: C*GU, G*GC, G*GU, U*AU, A*AU, A*GC, C*GC и некоторые другие. Триады характеризуются рядом особенностей, например, триада C*GC прочная лишь в слабокислой среде, когда происходит протонирование по атому N3 цитозина. Имеется много публикаций о РНК-триплексах, среди которых отметим [10–16]. Если третье плечо триплекса находится на 5'-конце его спирали, то необходимо, чтобы оно располагалось на определенном расстоянии от соседнего к нему плеча спирали, достаточном для сворачивания триплекса. Мы принимали: если третье плечо содержит k нуклеотидов, то оно отделено не менее чем на k+6 нуклеотидами от соседнего плеча. Если третье плечо находится на 3'-конце спирали, то оно может быть и вблизи нее, как показано в [15–16].

Разрешено образование псевдоузлов у всех вторичных структур в каждом текущем окне на РНК между рибосомой, транслирующей лидерный пептид, и РНК-полимеразой. При этом энергия вторичной структуры вычисляется на основе усовершенствования метода, представленного в [17], и учитывается температура типичная для среды обитания бактерии.

При моделировании аттенюаторной регуляции у некоторых видов актинобактерий эффективная зависимость от концентрации триптофана частоты преждевременной терминации транскрипции наблюдается только при учёте РНК-триплексов и псевдоузлов. Однако полученные результаты носят предварительный характер, поскольку точные значения энергий этих структур не известны. В частности, важную роль играет правильное вычисление изменения энтропии при формировании триплексов.

С другой стороны, анализ траекторий моделирования в нашей модели позволяет

приписать некоторым шпилькам роль терминатора, а другим – антитерминатора, и построить адекватное регуляции (если она имеется) локальное множественное выравнивание потенциальных регуляторных областей. Мы ожидаем, что существенные элементы регуляторных структур консервативны, то есть хорошо выравниваются друг с другом. И действительно, такие выравнивания обычно удаётся построить [4–6] Это даёт возможность независимой проверки правильности предсказания, в частности, правильности выбора формул для вычисления энергии и констант для вычисления скоростей образования или распада элементов структуры РНК: спиралей и триплексов.

Описанные методы могут применяться и для предсказания эффективности регуляции инициации трансляции в зависимости от скорости трансляции лидерного пептида. Типичным примером такой регуляции служит LEU-регуляция гена *leuA*, которая обнаружена у большинства актинобактерий из порядка Actinomycetales [4–5]. Здесь предсказание регуляции основано на консервативности структуры РНК, включающей псевдоузел, и позиционную связь этой структуры с участком регуляторных кодонов и сайтом связывания рибосомы перед кодирующей областью *leuA*.

Работа выполнена за счёт гранта Российского научного фонда (проект № 14–50–00150).

Литература

1. Das A., Crawford I.P., Yanofsky C. Regulation of tryptophan operon expression by attenuation in cell-free extracts of *Escherichia coli* // *Journal of Biological Chemistry*, 1982, vol. 257, no. 15, pp. 8795–8798.
2. Heery D.M., Dunican L.K. Cloning of the *trp* gene cluster from a tryptophan hyperproducing strain of *Corynebacterium glutamicum*: Identification of a mutation in the *trp* leader sequence // *Applied and Environmental Microbiology*, 1993, vol. 59, no. 3, pp. 791–799.
3. Lin C., Paradkar A.S., Vining L.C. Regulation of an anthranilate synthase gene in *Streptomyces venezuelae* by a *trp* attenuator // *Microbiology*, 1998. , vol. 144, no. 7, pp. 1971–1980.
4. Seliverstov A.V., Putzer H., Gelfand M.S., Lyubetsky V.A. Comparative analysis of RNA regulatory elements of amino acid metabolism genes in Actinobacteria // *BMC Microbiology*, 2005, vol. 5, no. 54.
5. Лопатовская К.В., Селиверстов А.В., Любецкий В.А. Аттенюаторная регуляция оперонов биосинтеза аминокислот и аминоацил-тРНК у бактерий: сравнительный геномный анализ // *Молекулярная биология*, 2010, том 44, № 1, стр. 140–151.
6. Lyubetsky V.A., Korolev S.A., Seliverstov A.V., Zverkov O.A., Rubanov L.I. Gene expression regulation of the PF00480 or PF14340 domain proteins suggests their involvement in sulfur metabolism // *Computational Biology and Chemistry*, 2014, vol. 49, pp. 7–13.
7. Lyubetsky V.A., Pirogov S.A., Rubanov L.I., Seliverstov A.V. Modeling classic attenuation regulation of gene expression in bacteria, Journal of Bioinformatics and Computational Biology, 2007, vol. 5, no. 1, pp. 155–180.
8. Spraggon G., Kim C., Nguyen-Huu X., Yee M.C., Yanofsky C., Mills S.E. The structures of anthranilate synthase of Serratia marcescens crystallized in the presence of (i) its substrates, chorismate and glutamine, and a product, glutamate, and (ii) its end-product inhibitor, L-tryptophan // *Proc Natl Acad Sci U S A*, 2001, vol. 98, pp. 6021–6026.
9. Finn R.D., Bateman A., Clements J., Coggill P., Eberhardt R.Y., Eddy S.R., Heger A., Hetherington K., Holm L., Mistry J., Sonnhammer E.L.L., Tate J., Punta M. The Pfam protein families database // *Nucleic Acids Research*, 2014, Database Issue 42, pp. D222–D230.
10. Lyubetsky V.A., Korolev S.A., Seliverstov A.V., Zverkov O.A., Rubanov L.I. Gene expression regulation of the PF00480 or PF14340 domain proteins suggests their involvement in sulfur metabolism // *Computational Biology and Chemistry*, 2014, vol. 49, pp. 7–13.
11. Chastain M., Tinoco I.Jr. Poly(rA) binds poly(rG) poly(rC) to form a triple helix // *Nucleic Acids Res*., 1992, vol. 20, no. 2, pp. 315–318.
12. Кнорре Д.Г., Мызина С.Д. *Биологическая химия*. М.: Наука. 2000.
13. Semerad C.L., Maher L.J. Exclusion of RNA strands from a purine motif triple helix // *Nucleic Acids Res*., 1994, vol., 22, no. 24, pp. 5321–5325.
14. Carmona P., Molina M. Binding of oligonucleotides to a viral hairpin forming RNA triplexes with parallel G*GC triplets // *Nucleic Acids Res*., 2002, vol. 30, no. 6, pp. 1333–1337.
15. Klinck R., Guitteta E., Liquier J., Taillandier E., Gouyetteb C., Huynh-Dinhby T. Spectroscopic evidence for an intramolecular RNA triple helix // *FEBS Lett*., 1994, vol. 355, pp. 297–300.
16. Holland J.A., Hoffman D.W. Structural features and stability of an RNA triple helix in solution // *Nucleic Acids Res*., 1996, vol. 24, no. 14, pp. 2841–2848.
17. Isambert H., Siggia E.D. Modeling RNA folding paths with pseudoknots: application to hepatitis delta virus ribozyme // *Proc. Nat. Acad. Sci. USA*., 2000, vol. 97, no. 12, pp. 6515–6520.

Любецкий В.А.[1], Селиверстов А.В.[2]

[1]ИППИ РАН, г. Москва, д.ф.-м.н., зав.лаб., lyubetsk@iitp.ru

[2]ИППИ РАН, г. Москва,к.ф.-м.н., в.н.с., slvstv@iitp.ru

О РЕШЕНИИ ОДНОЙ NP-ПОЛНОЙ ЗАДАЧИ

КЛЮЧЕВЫЕ СЛОВА

Задача о разбиении множества с весами, решение комбинаторной проблемы, недетерминированный полиномиальный алгоритм, NP-полная задача, поиск особых точек на кубике, единственное решение, теория поля комплексных чисел.

АННОТАЦИЯ

Предложен метод решения задачи о разбиении множества с весами как в смысле теории поля комплексных чисел, так и в смысле недетерминированного полиномиального алгоритма. Задача сводится к рекурсивному решению вопроса об особых точках на явно заданной кубике низкого ранга, в предположении, что решение единственное или отсутствует.

Задача состоит в следующем. Дано *n*+1 чисел («весов»), которые могут повторяться, их можно считать целыми и строго положительными. Таким образом, дана последовательность $\alpha_0,\ldots,\alpha_n$. Нужно найти множество индексов, для которых суммарный вес равен точно половине суммы всех весов. Если упростить задачу, спросив, существует ли решение, то она будет NP-полной. Мы предполагаем её решить *при условии*: решений нет или решение единственное, которое при нашем подходе, по сути, повторяет указанное выше условие.

Понятие «решить» разбивается на два случая. Во-первых, можно искать две E-формулы полиномиальной длины в языке поля комплексных чисел: одна – «существует решение» и вторая – «не существует решения». Первая из них очевидна:

$$\exists x_0,..,x_n\left(\left(\alpha_0 x_0+\ldots+\alpha_n x_n\right)\wedge\left(..\,x_i^2=1..\right)\right).$$

Для второй формулы даже её существование совершенно не очевидно. Во-вторых, можно искать недетерминированный полиномиальный алгоритм, который говорит «решений нет» или предъявляет решение (вариант – говорит «решение есть»). Мы рассмотрим оба эти случая на основе, как нам кажется, необычного подхода, состоящего в сведении исходной комбинаторной задачи к исследованию особых точек ранга *n*+1 явно заданной кубической формы в *n*-мерной гиперплоскости *L* в (*n*+1)-мерном комплексном линейном пространстве $\mathbb{C}^{n+1}$. Коэффициенты формы и гиперплоскости тесно связаны.

Приводимое ниже рассуждение, может быть, даже более естественно в комплексном проективном пространстве $P\left(\mathbb{C}^{n+1}\right)$, состоящем из прямых, проходящих через нуль в $\mathbb{C}^{n+1}$ (далее называемых *прямыми*). Но для простоты будем говорить о самом $\mathbb{C}^{n+1}$.

Форма и соответствующая ей гиперповерхность одинаково обозначаются. Нормаль – прямая, заданная градиентом.

Назовём *вершиной* последовательность длины *n*+1, состоящую из чисел ±1. Пара противоположных вершин соответствует разбиению множества на две части в исходной задаче.

Обозначим L гиперплоскость в $\mathbb{C}^{n+1}$, заданную формой

$$L=\alpha_0 x_0+\ldots+\alpha_n x_n,$$

т.е. заданную уравнением $L=0$, её размерность *n*.

Обозначим

$$F=\alpha_0 {x_0}^3+\ldots+\alpha_n {x_n}^3$$

и соответственно F_L сужение этой формы на *L*. Такое сужение F_L – кубическая форма, определённая на *L*, ранга *n*+1, о которой говорилось выше, при постановки задачи. Она может иметь *особые точки*, т.е. точки, в которых градиент равен нулю. Точка нуль $<0,\ldots,0>$ здесь и

далее исключается из рассмотрения. Очевидно, особая точка для F_L лежит на прямой, состоящей сплошь из особых для F_L точек; назовём её *особой прямой*. Таким образом, особые точки и особые прямые (всегда для F_L) можно не различать.

(1) Если вершина x лежит в *L*, то x – особая точка для F_L в *L*. И наоборот, если x – особая прямая в *L*, то на ней имеется пара противоположных вершин.

Доказательство. Пусть *x* – вершина в *L*, т.е. *L*(*x*)=0, то *F*(*x*)=0; градиенты от *L* и от *F* в *x* соответственно равны $<\alpha_0,...,\alpha_n>$ и $<3\alpha_0 x_0^{\,2},...,3\alpha_n x_n^{\,2}>=<3\alpha_0,...,3\alpha_n>$, т.е. коллинеарны между собой. Производную $F'(x)$ можно разложить на компоненты в *L* и в нормали к *L*, поэтому первая из них равна 0 и равна $(F_L)'(x)$.

Пусть *x* – точка из *L* и $(F_L)'(x)=0$. Разложим $F'(x)$ по тем же компонентам. Тогда $F'(x)$ расположена на нормали к *L*, которая определяется вектором $<\alpha_0,...,\alpha_n>$, т.е. $<\alpha_0,...,\alpha_n>$ и $<3\alpha_0 x_0^{\,2},...,3\alpha_n x_n^{\,2}>$ коллинеарны. Последнее означает $x_i^{\,2}=x_j^{\,2}$. На прямой, проходящей через нуль и эту точку *x*, имеется точка *x′* (положим у *x* первую координату равной 1), для которой $x_i^{\,2}=1$; эта *x′* – вершина в *L*.

(2) *Предположим*, что F_L имеет не более одной особой прямой (в проективизации – особой точки). Это – частный случай указанного выше условия, о котором удобнее говорить. Назовём *нетривиальным* невырожденное линейное нескалярное (т.е. не вида $z\cdot E$) преобразование *J* в линейном пространстве L над $\mathbb{C}$, сохраняющее множество нулей формы F_L. Последовательно применяя нетривиальные преобразования к уже полученным подпространствам, получим $L=L_1\oplus...\oplus L_p$, где $p\le n$ и каждое L_i малой размерности, ограниченной сверху константой. Особая прямая, если она есть, лежит в одном из подпространств L_i. Отсутствие особой прямой для F_{L_i} в каждом L_i выражается условием: дискриминант не равен нулю [1]. В одномерном случае это условие можно заменить неравенством нулю всех частных производных в какой-то ненулевой точке. Суммарная длина полученной E-формулы полиномиальная от *n*. В частности, бесквантерная часть этой формулы включает условие нетривиальности каждого преобразования.

(3) Пошевелим базисы в $L_1,...,L_p$, чтобы базисы стали рациональными, получим подпространства $L_1',...,L_p'$. Новые подпространства можно выбрать со следующими свойствами: (*a*) ближайшая пара противоположных вершин единственна, обозначим её $\varepsilon(L_i')$; и (*b*) подпространство L_i содержит особую прямую, если и только если $\varepsilon(L_i')$ – вершины, лежащие в гиперплоскости L. Недетерминированный шаг состоит в указании новых базисов.

Работа выполнена за счёт гранта Российского научного фонда (проект № 14–50–00150).

Литература

1. Морозов А.Ю., Шакиров Ш.Р. Новые и старые результаты в теории результатов // *Теоретическая и математическая физика*, 2010, том 163, № 2, стр. 222–257.

Зверков О.А.[1], Селиверстов А.В.[2], Любецкий В.А.[3]

[1]ИППИ РАН, г. Москва, к.ф.-м.н., н.с., zverkov@iitp.ru

[2]ИППИ РАН, г. Москва,к.ф.-м.н., в.н.с., slvstv@iitp.ru

[3]ИППИ РАН, г. Москва, д.ф.-м.н., зав.лаб., lyubetsk@iitp.ru

О ТРАНСКРИПЦИОННЫХ ФАКТОРАХ, КОДИРУЕМЫХ В ПЛАСТИДАХ РОДОФИТНОЙ ВЕТВИ

КЛЮЧЕВЫЕ СЛОВА

Пластида, белок, транскрипционный фактор, кластеризация.

АННОТАЦИЯ

Расширена ранее полученная кластеризация белков, кодируемых в пластидах родофитной ветви. Результаты представлены в общедоступной базе данных по адресу http://lab6.iitp.ru/ppc/redline67. База данных позволяет проводить быстрый поиск кластера (семейства белков) как по фрагменту аминокислотной последовательности одного из белков, так и по филогенетическому профилю белка. На этой основе нами предсказаны регулоны транскрипционных факторов Ycf28, Ycf29 и Ycf30, кодируемых в пластидах.

Быстрый рост числа секвенированных геномов пластид позволяет развить предположения об эволюции и регуляции этих геномов не только у водорослей, но и у нефотосинтезирующих видов простейших, содержащих пластиды. Среди последних возбудители опасных протозойных инфекций, таких как малярия и токсоплазмоз. Именно пластиды являются одной из эффективных мишеней для терапевтического воздействия и для получения невирулентных штаммов с целью быстрого создания вакцин.

Все известные пластиды происходят от цианобактерий. Выделяют три ветви первичных пластид с независимым происхождением, которые представлены зелёными водорослями и растениями, *Cyanophora paradoxa* и багрянками. Однако многие далёкие от перечисленных виды имеют вторичные или третичные пластиды, происходящие от первичных пластид. В этой работе мы ограничимся исследованием пластид родофитной ветви, имеющих общее происхождение с пластидами багрянок.

Весной 2015 года были опубликованы четыре новых генома пластид из родофитной ветви: *Lepidodinium chlorophorum* (класс Dinophyceae, GenBank: NC_027093, дата 14.05.2015), *Choreocolax polysiphoniae* (отдел Rhodophyta, GenBank: NC_026522, дата 27.03.2015), *Vertebrata lanosa* (отдел Rhodophyta, GenBank: NC_026523, дата 27.03.2015) и *Trachydiscus minutus* (отдел Eustigmatophyceae, GenBank: NC_026851, дата 22.04.2015). В пластиде *Choreocolax polysiphoniae* кодируются, в частности, некоторые гены путей синтеза триптофана и разветвлённых аминокислот, а также гены метаболизма жирных кислот. Однако генов фотосистем нет. Поэтому он существенно отличается от других багрянок.

Для кластеризации белков применён метод, описанный в [1] и успешно апробированный в серии работ [2–4]. Для визуализации кластеров (семейств белков) использована программа *sfdp* из пакета *Graphviz* [5]. Поиск промоторов осуществлён оригинальным алгоритмом, основанном на консервативности известных промоторов [6–7] и экспериментальных данных о влиянии нуклеотидных замен на эффективность инициации транскрипции РНК-полимеразой бактериального типа [8].

Нами выполнена кластеризация белков, кодируемых в пластидах родофитной ветви. Результаты представлены в общедоступной базе данных по адресу http://lab6.iitp.ru/ppc/redline67. База данных позволяет проводить быстрый поиск кластера как по фрагменту аминокислотной последовательности одного из белков, так и по филогенетическому профилю. Четыре добавленных протеома содержат в сумме 462 белка, что увеличивает общее число белков до 8940. Число синглетонов равно 259; число кластеров равно 294.

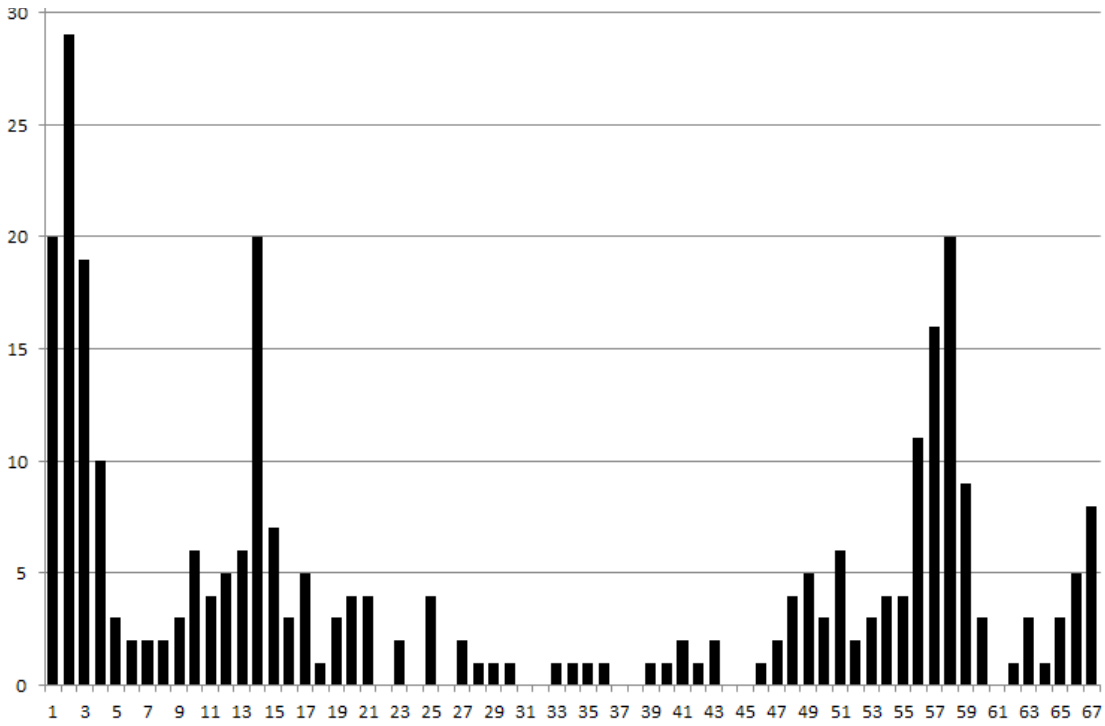


Рис.1. Зависимость числа кластеров белков от числа представленных видов водорослей

Зависимость числа кластеров белков от числа представленных видов показано на рисунке 1. Специфичными для всех пластид рассмотренных видов из отдела Rhodophyta (багрянок) являются гены *odpA/pdhA*, *odpB/pdhB*, *trpA*, *trpG*, *tilS/ycf62* и *infC*.

На рисунке 2 показан граф, вершины которого соответствуют белкам, а рёбра соединяют белки, связанные взаимными хитами (BBH) BLAST'a ($E<10^{-5}$). За редким исключением кластеры отлично разделяются выделением компонент связности. При этом многие из них удивительно плотные. Выделяются следующие пары и одна тройка близких по последовательности белков: PsaA + PsaB, PsbA + PsbD, PsbB + PsbC + синглетон Orf157, AtpA + AtpB, PetJ + PetV, ApcA + ApcD, ApcB + ApcF, CpeA + CpcA, CpcB + CpeB, CpcG + ApcE, TrpG + CarA, TufA + InfB. Здесь каждая сумма означает связный подграф, большинство вершин которого соответствуют одному из указанных типов белков. Параметры кластеризации выбираются так, чтобы указанные подграфы разделились в соответствии с аннотацией белков.

В целом распределение по кластерам белков, кодируемых в пластидах багрянок, показывает значительное отличие *Porphyridium purpureum* от остальных видов, что сопровождается многочисленными перестройками ДНК пластид этих водорослей, а также значительное обособление клады из трёх видов *Galdieria sulphuraria*, *Cyanidium caldarium* и *Cyanidioschyzon merolae*.

По сравнению с нашими предыдущими результатами кластеры белков MoeB и Ycf28 одновременно пополнились белками, кодируемыми в пластидах *Vertebrata lanosa*, у *Choreocolax polysiphoniae* и всех рассмотренных видов вне отдела Rhodophyta ни один из этих белков не кодируется в пластидах. Так подтверждено ранее отмеченное нами совпадение филогенетических профилей этих белков [4].

Белок Ycf28 имеет значительное сходство с транскрипционным фактором NtcA цианобактерий. Поэтому мы предполагаем, что именно Ycf28 регулирует в пластидах транскрипцию гена *moeB*, связывая ДНК вблизи промотора, где нами предсказан консервативный мотив. При этом нет оснований считать, что Ycf28 связан с метаболизмом азота, то есть по сравнению с цианобактериями произошла смена специфичности транскрипционного фактора к субстрату. Отсутствие типичного -35 бокса промотора перед геном *moeB*, говорит о том, что Ycf28 является активатором транскрипции.

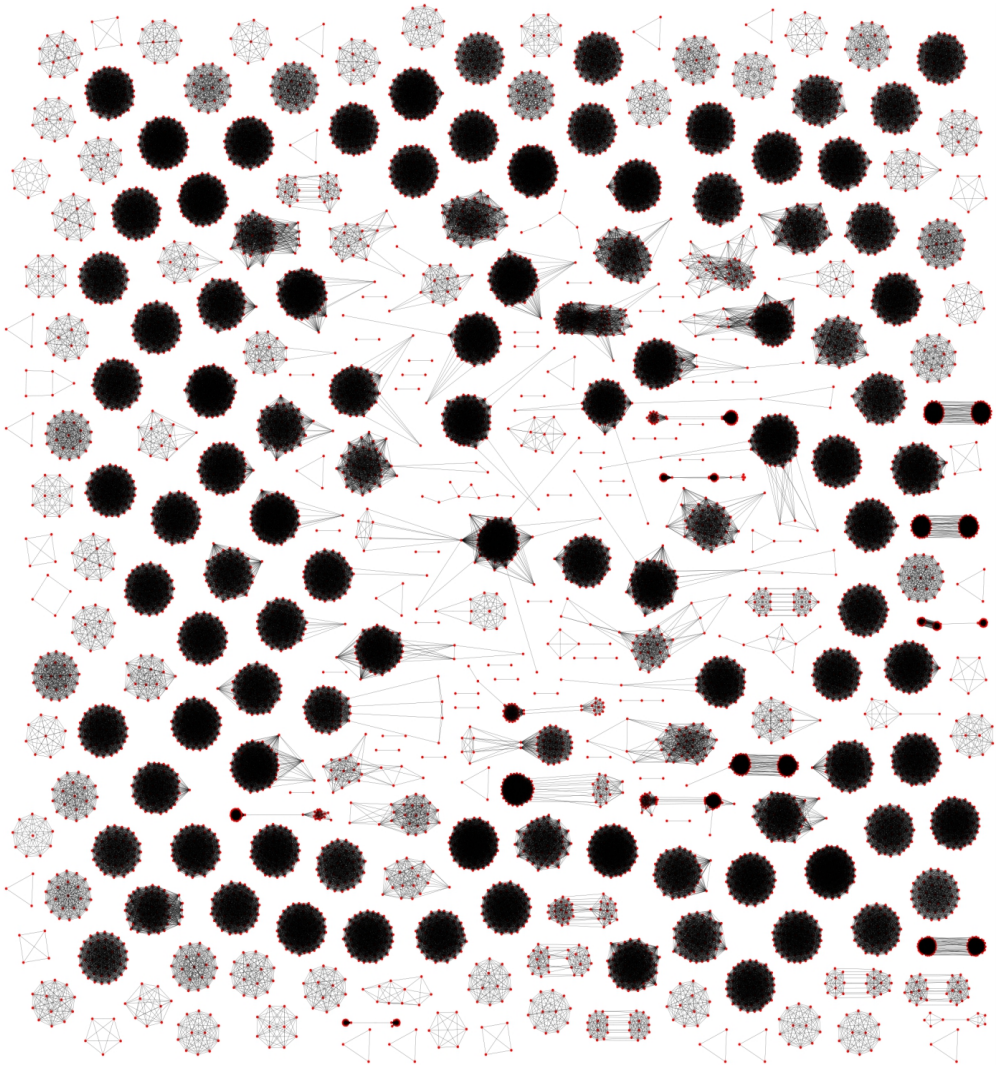


Рис.2. Компоненты связности графа белков в пластидах родофитной ветви

Транскрипционный регулятор Ycf29 кодируется в пластидах криптофитовых водорослей и багрянок кроме *Porphyridium purpureum*. Дерево Ycf29, построенное по выравниванию белков, показано на рисунке 3. Визуализация скобочной структуры выполнена программой TreeView 1.6.6 [9]. Другого белка с таким филогенетическим профилем не найдено. Близкий профиль имеет белок CemA, который есть у *Porphyridium purpureum*, но отсутствует у *Choreocolax polysiphoniae*. Белок CemA содержит домен PF03040 и найден на внутренней стороне наружной мембраны хлоропластов, но не в тилакоидной мембране. Ортологичные CemA белки у цианобактерий вовлечены в транспорт углекислого газа, но сами не являются транспортёрами [10]. Другой белок с близким филогенетическим профилем – это мембранный белок Ycf19. Перед геном *ycf19* предсказан консервативный промотор бактериального типа, близкий к консенсусу. С другой стороны, присутствие Ycf29 у нефотосинтезирующих видов *Cryptomonas paramecium* и *Choreocolax polysiphoniae* свидетельствует о том, что он регулирует процессы, не связанные с фотосинтезом. Поскольку Ycf29 входит в двухкомпонентную систему передачи сигнала, его регулон связан с реакцией на изменения внешних условий, а не внутри пластиды.

Известно, что в пластидах многих видов водорослей кодируется транскрипционный фактор Ycf30, регулирующий экспрессию генов *rbcLS*, кодирующих субъединицы рибулозобисфосфаткарбоксилазы (КФ 4.1.1.39), и гена *cbbX*. Экспериментально показана индуцируемая светом активация транскрипции этих генов в изолированных *Cyanidioschyzon merolae* и предсказан сайт связывания фактора Ycf30 в пластидах багрянок [11]. Построенные нами филогенетические профили этих белков согласуются с этими предсказаниями. Однако низкая консервативность сайта связывания Ycf30 не позволила точно предсказать его положение на ДНК.

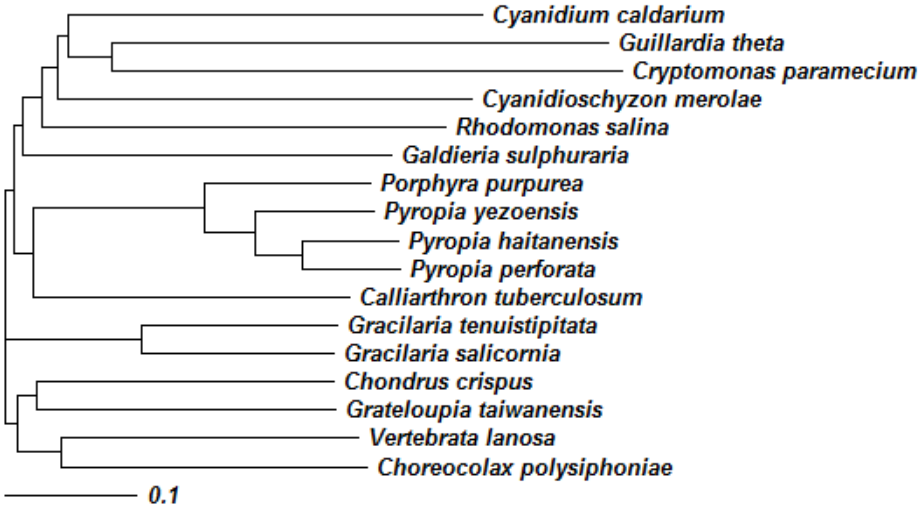


Рис.3. Дерево факторов Ycf29

Ген *ftsH* (синоним *ycf25*) кодирует АТФ-зависимую протеазу, содержащую цинк и предположительно участвующую в клеточном делении. Ортологичные гены широко распространены у прокариот [12]. У бактерии *Oenococcus oeni* белок FtsH важен для защиты от теплового или осмотического стрессов [13].

У видов из группы Stramenopiles: золотисто-бурая водоросль *Heterosigma akashiwo*, жёлто-зелёная *Vaucheria litorea* и бурые водоросли *Ectocarpus siliculosus*, *Fucus vesiculosus* и *Saccharina japonica* нами выполнено выравнивание 5'-лидерных областей генов *ftsH*. Предсказан консервативный промотор, типичный для РНК-полимераз бактериального типа. Это хорошо согласуется с близким филогенетическим положением пластид этих видов.

Дадим подробное описание предсказанных промоторов генов *ftsH*. -35 бокс промотора имеет нуклеотидный состав TTGTAT у *H. akashiwo* и TTGTTT у *V. litorea*, *E. siliculosus*, *F. vesiculosus* и *S. japonica*. -10 бокс во всех случаях имеет 5'-расширение TG. Консенсус этого бокса TGnTAnwwA, где w обозначает любой из двух нуклеотидов T или A. Транскрипция предположительно начинается в позициях -41, -33, -29, -22 и -29 соответственно. Эти промоторы показаны на рисунке 4. Нуклеотидный состав наиболее консервативных позиций соответствует экспериментальным данным о влиянии мутаций на эффективность инициации транскрипции РНК-полимеразой бактериального типа [8].

```
Heterosigma akashiwo   AGTTGTATAAAAAGATTTT-TCATGTTATATATAATTTAA -41
Vaucheria litorea      TATTGTTTTTGTAATTAATTTTATGGTACTTAAAAAAATA -33
Ectocarpus siliculosus GGTTGTTTATTAAGATAAT-AAATGTTAATAATACTTTAT -29
Fucus vesiculosus      GTTTGTTTTTTAATACATT-ATATGTTAATAATAATATTA -22
Saccharina japonica    AGTTGTTTTTTACTCTGAT-ATGTGTTAATAATACTATAT -29
```

Рис.4. Промоторы перед генами ftsH. Выделены -35 и -10 боксы и 5'-расширение последнего. Справа указаны позиции потенциального сайта инициации транскрипции относительно инициирующего кодона

Менее консервативные потенциальные промоторы перед геном *ftsH* предсказаны и у других водорослей. Кроме перечисленных случаев промоторы бактериального типа с 5'-расширением -10 бокса предсказаны у *Porphyra purpurea*, *Pyropia yezoensis*, *Pyropia haitanensis*, *Cyanidium caldarium*, *Gracilaria tenuistipitata*, *Rhodomonas salina*, *Odontella sinensis*, *Thalassiosira pseudonana*, *Thalassiosira oceanica*, *Synedra acus*, *Durinskia baltica* и *Kryptoperidinium foliaceum*. Напомним, что хотя две последние водоросли принадлежат надтипу Alveolata, их пластиды имеют третичное происхождение от пластид диатомовых водорослей [14]. Промоторы без 5'-расширения -10 бокса предсказаны у *Cyanidioschyzon merolae*, *Fistulifera* sp. JPCC DA0580 и *Phaeodactylum tricornutum*. У *Nannochloropsis* spp., *Porphyridium purpureum* и *Guillardia theta* определить промотор не удалось. Выделить консервативный сайт связывания какого-либо транскрипционного фактора вблизи промоторов также не удалось, что позволяет предполагать регуляцию экспрессии гена *ftsH* на каком-то другом уровне. Гены *ftsH* отсутствуют у представителей отделов Apicomplexa и Chromerida, классов Haptophyceae и Pelagophyceae, а также у нефотосинтезирующей

криптофитовой водоросли *Cryptomonas paramecium*.

Работа выполнена за счёт гранта Российского научного фонда (проект № 14–50–00150).

Литература

1. Любецкий В.А., Селиверстов А.В., Зверков О.А. Построение разделяющих паралоги семейств гомологичных белков, кодируемых в пластидах цветковых растений // *Математическая биология и биоинформатика*. 2013. Т. 8, № 1. С. 225–233.
2. Зверков О.А., Селиверстов А.В., Любецкий В.А. Белковые семейства, специфичные для пластомов небольших таксономических групп водорослей и простейших // *Молекулярная биология*. 2012. Т. 46, № 5. С. 799–809.
3. Lyubetsky V., Seliverstov A., Zverkov O. Transcription regulation of plastid genes involved in sulfate transport in Viridiplantae // *BioMed Research International*. 2013. V. 2013. Article ID 413450. 6 p.
4. Zverkov O.A., Seliverstov A.V., Lyubetsky V.A. A database of plastid protein families from red algae and Apicomplexa and expression regulation of the *moeB* gene // *BioMed Research International*. 2015. V. 2015. Article ID 510598, 5 p.
5. Yifan Hu. Efficient high-quality force-directed graph drawing // *The Mathematica Journal*. 2006. V. 10, № 1. P. 37–71.
6. Селиверстов А.В., Лысенко Е.А., Любецкий В.А. Быстрая эволюция промоторов пластомных генов *ndhF* у цветковых растений // *Физиология растений*. 2009. Т. 56, № 6. С. 926–934.
7. Lyubetsky V., Rubanov L., Seliverstov A. Lack of conservation of bacterial type promoters in plastids of Streptophyta // *Biology Direct*. 2010. V. 5, № 34. 11 p.
8. Homann A., Link G. DNA-binding and transcription characteristics of three cloned sigma factors from mustard (*Sinapis alba* L.) suggest overlapping and distinct roles in plastid gene expression // *Eur. J. Biochem*. 2003. V. 270, № 6. P. 1288–1300.
9. Page R.D.M. TREEVIEW: An application to display phylogenetic trees on personal computers // *Computer Applications in the Biosciences*. 1996. V. 12. P. 357–358.
10. Katoh A., Lee K.S., Fukuzawa H., Ohyama K., Ogawa T. *cemA* homologue essential to CO_2 transport in the cyanobacterium *Synechocystis* PCC6803 // *Proc Natl Acad Sci USA*. 1996. V. 93. P. 4006–4010.
11. Minoda A., Weber A.P., Tanaka K., Miyagishima S.Y. Nucleus-independent control of the rubisco operon by the plastid-encoded transcription factor Ycf30 in the red alga *Cyanidioschyzon merolae* // *Plant Physiol*. 2010. V. 154, № 3. P. 1532–1540.
12. Herman C., Prakash S., Lu C.Z., Matouschek A., Gross C.A. Lack of a robust unfoldase activity confers a unique level of substrate specificity to the universal AAA protease FtsH // *Mol Cell*. 2003. V. 11, P. 659–669.
13. Bourdineaud J.P., Nehme B., Tesse S., Lonvaud-Funel A. The ftsH gene of the wine bacterium Oenococcus oeni is involved in protection against environmental stress // *Appl Environ Microbiol*. 2003. V. 69. P. 2512–2520.
14. Imanian B., Pombert J.F., Keeling P.J. The complete plastid genomes of the two 'dinotoms' *Durinskia baltica* and *Kryptoperidinium foliaceum* // *PLoS ONE*. 2010. V. 5, № 5. Article E10711.

Гагарин А.П.

Московский авиационный институт (национальный исследовательский университет), Москва, к.т.н., профессор по кафедре информатики, gagarin_ay@outlook.com

ОЦЕНКА СТРУКТУРНОЙ СЛОЖНОСТИ НЕКОТОРЫХ ТИПОВЫХ ОБЪЕКТНЫХ МОДЕЛЕЙ ДАННЫХ В МОЛЕКУЛЯРНОЙ ГЕНЕТИКИ

КЛЮЧЕВЫЕ СЛОВА

MDA (архитектура, задаваемая моделью), объектная модель, язык UML, домен, доменная спецификация, класс, диаграмма классов, атрибут класса, метод класса, ассоциация, IDE, биочип, экспрессия генов, структурная сложность, дерево, гамак.

АННОТАЦИЯ

Предлагается оценка структурной сложности для модели данных MAGE-OM (MicroArray and Gene Expression Object Model), предложенной Консорциумом OMG -- международной «Рабочей группой по созданию и продвижению объектно-ориентированных технологий и стандартов» с целью совершенствования применения информационных технологий в сфере биологических наук.

Успех научных исследований в современной молекулярной биологии во многом зависит от применения информационных технологий, обеспечивающих оперативное общение исследователей, хранение и коллективный поисковый доступ к большим массивам данных, их аналитическую обработку и адекватную визуализацию. Развитие этих технологий на основе моделей в рамках архитектуры MDA позволяет сочетать стабильность – в части форматов хранения и протоколов передачи данных с адаптивностью – в части программных инструментов аналитики и визуализации. От сложности этих моделей зависит качество и стоимость реализуемых по ним программных продуктов и, в конечном итоге, оснащённость ими конкретной области исследований.

Продолжающееся распространение языка UML как средства документирования проектных решений при разработке компьютерных программ и описания их областей приложения стимулирует практический интерес к прагматике этого языка, уточнению стилей и выработке критериев его успешного применения. Ввиду того, что тексты на языке UML предназначаются, главным образом, для восприятия человеком, важнейшим критерием является сложность, от которой зависит понятность текста. Законченный текст на языке UML выражает некоторую модель, которая изображается на одной или нескольких диаграммах. Сложность конкретной UML-диаграммы зависит от сложности изображаемой модели и языкового стиля, которым пользовался автор диаграммы. Авторитетные, прошедшие успешную апробацию, тексты могут служить образцом использования языка и эталоном стилистической сложности при известной сложности моделируемого объекта.

В качестве таковых рассматриваются доменные спецификации, выпущенные, начиная с 2001 года, «Консорциумом (рабочей группой) по созданию и продвижению объектно-ориентированных технологий и стандартов» (Object Management Group, далее – Консорциум OMG). Они представлены на сайте Консорциума [1]. Спецификации, относящиеся к научным исследованиям в области биохимии и молекулярной генетики, перечислены ниже:

formal/2001-06-08 Biomolecular Sequence Analysis
formal/2002-02-01 Genomics Maps
formal/2002-05-01 Macromolecular Structure
formal/2002-05-03 Bibliographic Query Service
formal/2003-03-19 Laboratory Equipment Control Interface Specification
formal/2003-10-01 MicroArray and Gene Expression
formal/2004-12-01 Life Science Identifiers
formal/2005-11-01 Single Nucleotide Polymorphisms
formal/2005-12-02 Life Science Analysis Engine
formal/2007-01-02 Chemical Structure and Access Representation
formal/2010-07-01 Phenotype and Genotype Object Model

Далее исследуется спецификация [2], в которой определена объектная модель MAGE-OM (MicroArray and Gene Expression Object Model), отображающая сущности и процессы экспериментального исследования экспрессии генов на биочипах. Модель предназначена для обеспечения обмена информацией между исследователями по унифицированным формам и может служить основой для разработки программного обеспечения исследований такого рода. О практическом использовании модели свидетельствует [3].

Модель состоит из набора пакетов, отображающих все основные стороны исследований. Содержание пакетов показано на языке UML на диаграммах классов, перечисленных в Табл. 1.

№/№	Наименование диаграммы	Количество		
		классов	связей-ассоциаций	связей-обобщений
1	Фундаментальные классы (Logical View)	4/7	1/5	2
2	Аудит и права доступа (Audit and Security)	6/7	5/6	2
3	Описание объекта (Description)	5/7	10/11	-
4	Биологическая последовательность (BioSequence)	6/8	4/9	2
5	Структура биочипа (Array Design)	5/12	3/12	1
6	Элемент структуры биочипа (Design Element)	9/16	5/12	6
7	Отображение элементов структуры биочипа (Design Element Maps)	7/11	11/11	3
8	Биочип (Array)	10/24	9/26	1
9	Биособытие (BioEvent)	2/16	1/1	14
10	Биоматериал (BioMaterial)	8/13	7/16	3
11	Биопроба (BioAssay)	9/20	5/18	6
12	Данные биопроб (BioAssayData)	13/20	7/15	9
13	Преобразование данных биопроб (BioAssayData Transformation)	10/13	8/12	2
14	Эксперимент (Experiment)	4/11	4/17	0
15	Высокоуровневый анализ (High Level Analysis)	4/9	4/11	0
16	Измерения (Measurement)	9/9	1/1	7
17	Параметризуемый объект (Parametrizable)	5/9	4/11	3
18	Параметризуемое приложение (Parametrizable Application)	9/10	7/8	3
19	Количественный тип (QuantitationType)	12/15	1/6	11

Таблица 1. Диаграммы модели MAGE-OM

Количество классов и ассоциаций показано парами чисел. Первое число показывает количество собственных элементов данного вида на диаграмме, а второе – полное количество

элементов, в которые, кроме собственных, входят дубли элементов из других диаграмм, служащие для указания связи между диаграммами. Матрица смежности по этим связям показана на Таблице 2. Число в ячейке матрицы показывает количество связей.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
1		2	1																
2			1																
3		1																	
4			2			2													
5		1	1													1	1		
6			3	1	5											1			
7						1													
8	1	4			2	2				1						2	1		
9						3				1	5		4				1		
10	1	1	2													1			
11												4							
12	1					4					1								1
13											2								1
14		1	2								1	1			1	1			
15			1									4							
16																			
17		1	2													1			
18		1																	
19			1									1							

Таблица 2. Матрица смежности диаграмм модели MAGE-OM

Из матрицы смежности видно, что диаграммы 8, 9 и 14 (в нумерации Табл. 1) и соответствующие пакеты другими диаграммами не используются, но используют 5 – 7 других диаграмм. Диаграммы 4, 5, 6, 8 и 9, 11, 13, 19 по связям образуют два кластера, органически связанные с исследованиями на биочипах, остальные диаграммы – с этими исследованиями по существу не связаны, имеют вспомогательный характер, обеспечивают полноту отображения домена и могут быть использованы в исследованиях, основывающихся на иных технологиях и с иными целями. По количеству классов оба кластера примерно равны.

Далее исследован первый из указанных кластеров. Чтобы исключить дублирование элементов, все его 4 диаграммы сведены в одну, показанную на Рис. 1. Сводная диаграмма содержит 35 классов и 51 связь, из которых всего 5 связей-обобщений, 3 неориентированные ассоциации, остальные – композиции. 6 классов – несобственные, представляющие диаграммы спецификации, связанные с данной. Количество атрибутов в собственных классах – 50. Поле стереотипа в обозначении класса используется как локальный идентификатор класса на диаграмме. Связи обозначаются парой номеров классов, соединённых связью.

Каждую диаграмму можно трактовать как ориентированный граф, в котором классы являются узлами, а ассоциации и обобщения – рёбрами. В сводной диаграмме общая характеристика такого графа: максимальная степень узла графа – 7, ветвлений – 14, слияний – 6. При подсчёте ветвлений и слияний связи-обобщения не учитываются.

По конфигурации граф сводной диаграммы имеет в целом древовидную структуру, в которой можно выделить два “дерева” и один “гамак”. Циклы (контуры) отсутствуют.

В графе модели, представленный на сводной диаграмме, могут быть выделены два дерева, одно из которых начинается от класса 1 (ArrayManufacture), а второе – от класса 9 (ArrayDesign). Они связаны вблизи “корней” ориентированной ассоциацией <5, 9>, идущей от класса Array и классу ArrayDesign. Деревья соприкасаются в узлах 19, 21, 29.

“Гамак” располагается между узлами 1 и 31. Длина ветви в обоих деревьях не превышает 7.

Сложности данного графа предлагается оценить по формуле:

$$5)\ C = 10 \times K / P,$$

где K – это количество узлов, в которых происходит слияние ветвей, а P – количество ветвей, образованных в графе. В случае сводной диаграммы K = 6, P = 24, то есть, C = 2,5.

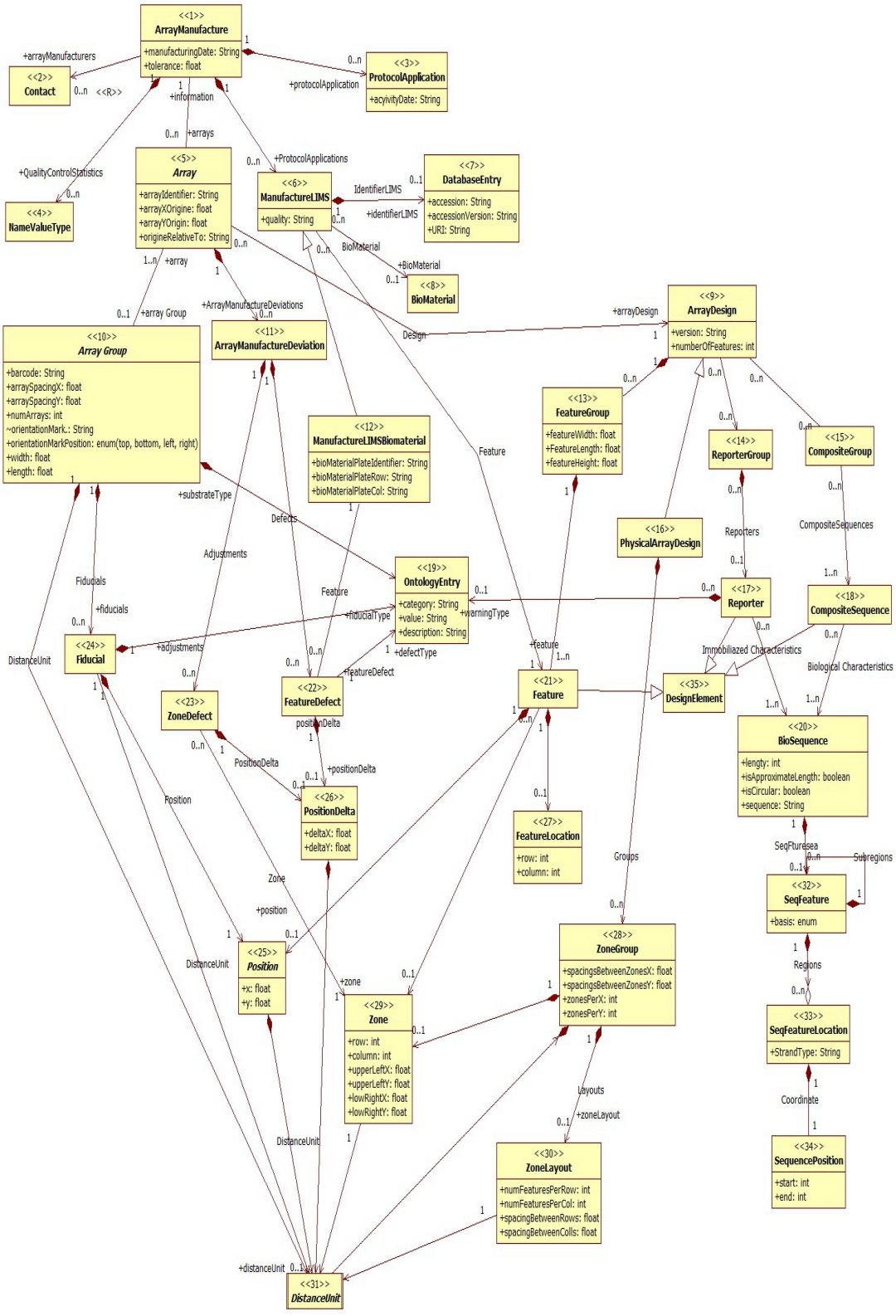


Рис1. Сводная диаграмма классов

Применимость этой формулы обосновывается тем, что трудность понимания диаграммы зависит от отклонения конкретной диаграммы от простейших структур – “дерева” и “гамака”. Чем больше сливающихся ветвей и чем в большем количестве узлов они сливаются, тем менее понятна диаграмма и тем выше должна быть оценена её сложность. Эта оценка непосредственно

выражается произведением D × K, где D – количество ветвей, закончившихся слиянием. Если F – количество финальных ветвей в графе, то D = P - F. Однако такая абсолютная оценка зависит от количества элементов диаграммы, что может мешать корректно сравнивать по сложности диаграммы, существенно отличающиеся количеством элементов. Чтобы ослабить эту зависимость предлагается взвесить D по F, а K – по D. Введение масштабирующего коэффициента 10 приводит к формуле 1.

Безупречное “дерево” имеет по данной формуле сложность 0, а безупречный “гамак” – наибольшую возможную сложность 1. Следовательно, можно заключить, что рассматриваемая сводная диаграмма имеет сложность выше средней.

Используемый подход к оценке сложности диаграмм классов, образующих ациклический граф по ориентированным ассоциациям (преимущественно композициям и агрегациям), соответствует критериям оценки совершенства программ, принятым в структурном программировании.

Для сравнения сложность исследуемой диаграммы также оценивается по методике, предложенной в [4]. Для диаграммы классов в случае преобладания связей одного типа и без учёта атрибутов и операций классов рекомендована формула

$$S = \sum S_{obj} / \left(1 + O_{bj} + \sqrt{T_{obj}}\right),$$

где

S - оценка диаграммы,

S_{obj} - оценки для элемента диаграммы,

O_{bj} - число объектов на диаграмме,

T_{obj} - число типов объектов на диаграмме.

В исследуемой диаграмме O_{bj} = 35, T_{obj} = 1. Авторы методики [4] предлагают для класса оценку S_{obj} = 5. В результате сложность диаграммы оказывается равной 4,7, что превышает указанный нормальный уровень сложности – 3-3,5 для диаграмм классов без учёта атрибутов и операций, указанный авторами методики.

В заключение следует отметить, что в диаграммах OMG-спецификаций язык UML использован ограниченно: только один тип диаграмм (из 13 типов) – диаграммы классов, из отношений преобладают отношения композиции в сочетании с незначительным количеством отношений агрегации. То есть, модель отображает, главным образом, структуры данных, что соответствует первичному её назначению – унифицировать форматы обмена данными об экспериментах. Для программ, реализующих компьютерную поддержку исследовательской деятельности, модель задаёт не более чем общую структуру. Диаграммы последовательностей (sequence diagrams) или деятельности (activity diagrams), на которых могли бы быть выражены алгоритмы такой поддержки, в спецификацию не входят.

Литература

1. OMG Domain Specifications. URL: http://www.omg.org/spec/
2. Gene Expression Specification. URL: http://www.omg.org/spec/GENE/1.1/
3. caArray 1.5 MAGE-OM_1.1 API Installation Guide http://www.docin.com/p-479081277.html
4. SDM metrics http://www.sdmetrics.com/

Рубанов Л.И.[1], Селиверстов А.В.[2], Зверков О.А.[3], Любецкий В.А.[4]

[1]ИППИ РАН, г. Москва, к.т.н., в.н.с., rubanov@iitp.ru

[2]ИППИ РАН, г. Москва, к.ф.-м.н., в.н.с., slvstv@iitp.ru

[3]ИППИ РАН, г. Москва, к.ф.-м.н., н.с., zverkov@iitp.ru

[4]ИППИ РАН, г. Москва, д.ф.-м.н., зав.лаб., lyubetsk@iitp.ru

УЛЬТРАКОНСЕРВАТИВНЫЕ ЭЛЕМЕНТЫ У ПРОСТЕЙШИХ ИЗ НАДТИПА ALVEOLATA

КЛЮЧЕВЫЕ СЛОВА

Ультраконсервативный элемент, простейшие, Alveolata, плотный подграф, кластеризация, параллельные вычисления.

АННОТАЦИЯ

Разработан и реализован алгоритм поиска ультраконсервативных элементов ДНК, основанный на поиске плотных подграфов в многодольном графе. У 22 видов из надтипа Alveolata построены кластеры ультраконсервативных элементов. Подтверждено, что род Cryptosporidium не входит в класс кокцидий, а является близким родственником плазмодиев, пироплазмид и Gregarina niphandrodes. Подтверждено, что фотосинтезирующие простейшие Chromera velia и Vitrella brassicaformis близкие родственники. Показано, что в составе типа Apicomplexa кокцидии сохранили большее число элементов, присутствовавших у общего предка надтипа Alveolata.

Введение. Ультраконсервативные элементы геномов впервые были обнаружены у млекопитающих [1, 2]. Функциональная роль таких элементов до сих пор не определена, но их очень высокая консервативность и сложность нуклеотидного состава позволяют рассматривать их как новый источник филогенетической информации в геномах, поскольку они наследуются от общего предка и находятся под действием стабилизирующего отбора. Такие участки могут служить маркёрами (зондами) для определения филогенетического положения малоизученных видов. За последнее время значительно увеличилось число секвенированных геномов простейших.

Нами проведён поиск ультраконсервативных элементов у 22 видов из надтипа Alveolata. Этот надтип представлен простейшими, многие из которых содержат пластиды, и включает тип Apicomplexa (споровики) и некоторые фотосинтезирующие виды, в том числе *Chromera velia* и *Vitrella brassicaformis*. Хотя виды из типа Apicomplexa не способны к фотосинтезу, их апикопласты происходят от пластид багрянок (Rhodophyta), в них происходит синтез изопреноидов и других соединений [3]. В то же время, криптоспоридии и близкий к ним вид *Gregarina niphandrodes* не содержат пластид [4, 5]. Подтверждение их родства позволяет, в частности, делать выводы о роли апикопластов у других споровиков. Это может иметь практическое значение, поскольку тип Apicomplexa включает большое число возбудителей протозойных инфекций, причём апикопласты служат удобной мишенью для терапевтического воздействия. Пироплазмиды представлены видами двух родов – это тейлерии и бабезии. Поскольку близкое родство тейлерий и бабезий подтверждается независимо от наших исследований, их можно рассматривать вместе. Это позволяет компенсировать погрешности, связанные с малыми размерами их геномов.

Методы. Применяемый метод состоит из двух шагов. В начале по исходным данным строится многодольный граф, доли которого соответствуют видам, вершины – участкам ДНК из этих видов, а рёбра соединяют похожие участки. При попарном выравнивании участков накладывается ограничение сверху на число идущих подряд делеций, что позволяет существенно сократить трудоёмкость вычисления. Также накладывается ограничение на суммарный штраф за замены нуклеотидов и за делеции. Кроме того участки отбирались по сложности (коэффициенту сжатия алгоритмом Лемпеля–Зива).

Пусть даны две нуклеотидные последовательности длиной n. Нужно найти все пары похожих слов длины не менее l (по одному слову из каждой последовательности). Сложность наивного алгоритма $O(n^2l^2)$ неприемлема для поиска пар слов в полных геномах, тем более для нескольких видов. Необходим быстрый алгоритм. Мы предполагаем, что слова тождественно совпадают на ключе, то есть участке длиной не менее некоторого $k < l$. Во-первых, индексируем первую последовательность, составляя хеш-таблицу всех ключей длины k в ней. Во-вторых, последовательно перебираем все ключи той же длины во второй последовательности, проверяя каждый ключ по составленной хеш-таблице. В-третьих, в случае совпадения ключей проверяем наличие в окрестности этих позиций искомой пары слов, это – ребро предварительного двудольного графа. Далее проводилась склейка вершин из одного вида, участвующих в различных ребрах, исходя из величины d перекрытия соответствующих участков генома.

Счет проводился для значений параметров: l=60, k=16, d=40. При выравнивании участков использовался штраф 1 за замену буквы и штраф 2,1 за делецию, причем допускалось не более 2 делеций подряд. Участки достаточной длины признавались похожими, если суммарный штраф за выравнивание не превышал 17,5. В точках совпадения ключей выбирались максимально длинные участки в пределах такого суммарного штрафа.

Сопоставление 22 полных геномов с построением начальных ребер графа выполнялось на суперкомпьютерах МВС-100К и МВС-10П Межведомственного суперкомпьютерного центра РАН (www.jscc.ru) и заняло около 200 часов. Последующая обработка осуществлялась на собственном 64-ядерном сервере, при этом сборка графа потребовала 13 часов на 16 процессорах, а поиск плотных подграфов тоже 13 часов, но на 22 процессорах. Отметим, что значительное время тратится на чтение и запись огромных файлов. Изначально в графе было 45 млн. вершин и 1.5 млрд. дуг (каждое ребро представлено двумя дугами встречных направлений). После объединения нескольких сильно пересекающихся участков в одну вершину их стало без малого 5 млн. (при том же числе ребер; кратные ребра удаляются алгоритмом построения плотного подграфа).

Ультраконсервативные элементы соответствуют максимальным по включению m-плотным подграфам этого графа для значений m, начиная с трёх. Напомним, что подграф многодольного графа называется m-плотным, если каждая его вершина соединена хотя бы одним ребром с вершинами из не менее чем (m–1)й другой доли. Например, m-клика в многодольном графе (с одной вершиной в доле) – это m-плотный подграф. Полный m-дольный граф m-плотный. Поиск m-клики в m-дольном графе, каждая доля которого содержит хотя бы по три вершины, – это NP-трудная задача. Однако высокая вычислительная сложность показана лишь для графов с очень высокой плотностью рёбер.

Значительная трудность состоит в том, что при неудачном выборе параметров или способа построения исходного графа в нём возникает гигантская связная компонента, объединяющая значительную часть вершин графа. Критериями успеха служат отсутствие гигантской компоненты и наличие m-плотных подграфов для значений m, больших половины от числа долей графа, то есть числа видов. Существование гигантской компоненты в исходном графе не обязательно служит препятствием к выделению m-плотных подграфов с малым числом вершин, когда параметр m достаточно большой. Однако при m=2 каждая компонента связности будет 2-плотным подграфом, который не разбивается алгоритмом на меньшие части. Значительные трудности можно ожидать и при других малых значениях m.

В случае двудольного графа, в котором никакие два ребра с общей вершиной не имеют одинакового веса, каждый выдаваемый нашим алгоритмом 2-плотный подграф имеет лишь одно ребро, то есть является 2-кликой. Отметим, что существуют 3-плотные 3-дольные графы, которые не содержат треугольников, то есть 3-клик. Пример показан на рис. 1, где вершины из одной доли лежат на горизонтальных прямых.

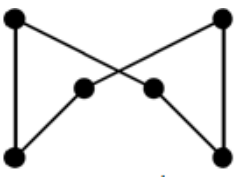


Рис.1. 3-плотный 3-дольный граф, который не содержит треугольников; вершины из одной доли лежат на горизонтали

В биоинформатике часто рассматривают взвешенные многодольные графы, у которых вес каждого ребра отражает сходство последовательностей, приписанных его концам. Важно искать

такие *m*-плотные подграфы взвешенного многодольного графа, которые содержат рёбра большого веса и не содержат много вершин из одной доли. Последние условия предполагают заданными некоторые пороги. Такие подграфы называют *кластерами*. Нами предложен алгоритм поиска кластеров, допускающий эффективное распараллеливание. Для однопроцессорного вычислительного устройства алгоритм описан в [6]. На его входе дан взвешенный многодольный граф. Этот параллельный алгоритм одновременно обрабатывает каждую вершину графа. Поскольку для алгоритма важно только взаимное соотношение между весами ребер, инцидентных одной вершине, в качестве веса каждой дуги использовалось значение длины участка, соответствующего началу этой дуги. При этом две дуги одного ребра могут иметь несколько отличающиеся веса по причине делеций, но помечаются или удаляются всегда вместе.

В каждой вершине графа алгоритм независимо выполняет следующие операции, работа начинается с шага *A*. После каждого шага выполняется синхронизация процессов во всех вершинах графа: очередной шаг начинается после окончания предыдущего шага во всех вершинах.

Шаг A. Если вершина соединена рёбрами менее чем с (*m*–1) долей, то её и все инцидентные ей рёбра удаляются. Если вершина соединена с какой-то долей лишь одним ребром, то помечаем его (в дальнейшем помеченное ребро может быть удалено только вместе с одним из его концов).

Если при выполнении шага *A* в графе произошли изменения, то снова выполняются шаги *A* во всех вершинах. В противном случае в каждой вершине однократно выполняется шаг *B*.

Шаг B. Если инцидентное вершине ребро не помечено и его вес строго меньше весов всех других не помеченных инцидентных рёбер (или оно единственное), то ребро удаляется. При проверке используются веса исходящих дуг.

Если после выполнения шага *B* в графе произошли изменения, то все вершины снова переходят к шагам *A*. В противном случае алгоритм заканчивает работу. Каждая компонента связности полученного в результате графа – искомый кластер.

Результаты. У 22 видов из надтипа Alveolata построено 845 кластеров (без гигантской компоненты). Данные о числе кластеров в зависимости от числа видов, представленных в кластере: 2 с представителями из 13 видов, 1 – из 12 видов, 4 – из 11 видов, 8 – из 10 видов, 17 – из 9 видов, 24 – из 8 видов, 35 – из 7 видов, 75 – из 6 видов, 131 – из 5 видов, 290 – из 4 видов, 258 – из 3 видов. Подавляющая доля кластеров (787) содержит не более одного участка из каждого вида, и только 9 кластеров содержат более двух участков из какого-нибудь вида (максимум 5 участков).

Большинство кластеров составляют два подмножества с маленьким пересечением: первое включает кокцидии, *Chromera velia* и *Vitrella brassicaformis*; второе включает криптоспоридии, плазмодии, пироплазмиды (тейлерии, бабезии) и *Gregarina niphandrodes*.

Вид *Gregarina niphandrodes* представлен в 10 кластерах, из которых 9 также содержат представителя из рода *Cryptosporidium*. Один из кластеров содержит два разных элемента из *Gregarina niphandrodes.*

Фотосинтезирующий вид *Vitrella brassicaformis* представлен в 34 кластерах. Из них 24 кластера одновременно содержат преставителей *Vitrella brassicaformis* и хотя бы одной кокцидии.

Другой фотосинтезирующий вид *Chromera velia* представлен в 20 кластерах, из которых 8 содержат представителей из *Vitrella brassicaformis* и 16 содержат представителей из кокцидий. То есть сходство между фотосинтезирующими видами оказалось меньше, чем сходство каждого их них с классом кокцидий в целом.

Кокцидия *Eimeria falciformis* представлена в 74 кластерах Наиболее изученная кокцидия *Toxoplasma gondii* представлена в 69 кластерах. Близкая к ней кокцидия *Neospora caninum* представлена в 53 кластерах. Кокцидия *Sarcocystis neurona* представлена в 26 кластерах, из которых 20 содержат представителей из других кокцидий и 8 содержат представителей из *Eimeria falciformis*.

Большое число кластеров (304) содержат представителей только из двух родов *Plasmodium* и *Cryptosporidium*.

Дерево, листья которого соответствуют трём видам *Gregarina niphandrodes*, *Chromera velia* и *Vitrella brassicaformis*, двум родам *Plasmodium* и *Cryptosporidium*, порядку Piroplasmida и классу Coccidia, отражающее число общих ультраконсервативных элементов, показано на рис. 2. Число в вершине дерева равно числу кластеров ультраконсервативных элементов, имеющих представителя в каждом листе порождённого поддерева. Расстояние по дереву тем меньше, чем больше общих кластеров.

Выравнивание против транскриптов из базы данных PlasmoDB (http://plasmodb.org) показывает, что наибольшие кластеры состоят из фрагментов генов, кодирующих белки. Кластеры

из 13 элементов кодируют фрагменты ribonucleoside-diphosphate reductase large subunit (PF3D7_1437200) и proline-tRNA ligase (PF3D7_1213800); кластер из 12 элементов – calmodulin (PF3D7_1434200); кластеры из 11 элементов –splicing factor U2AF small subunit (PF3D7_1119300), isoleucine-tRNA ligase (PF3D7_1332900), large subunit rRNA methyltransferase (PF3D7_1354300) и рибосомный белок L10. Здесь в скобках указан соответствующий ген из *Plasmodium falciparum* 3D7. В последнем случае фрагмент гена PF3D7_1414300 не входит в кластер, но присутствуют фрагменты ортологичных генов.

Некоторые из ультраконсервативных элементов, соответствующих вершинам гигантской компоненты графа, кодируют фрагменты рибосомных РНК или консервативных РНК сплайсосом. Однако другие не соответствуют ранее описанным типам РНК, распознаваемых сравнением с базой данных Rfam [7] или базой данных о uRNA [8]. Они требуют дальнейшего исследования. Впрочем, для сбора филогенетической информации о виде можно не выделять РНК отдельно от других участков генома. Новые участки могут представлять собой регуляторные элементы или необычные РНК.

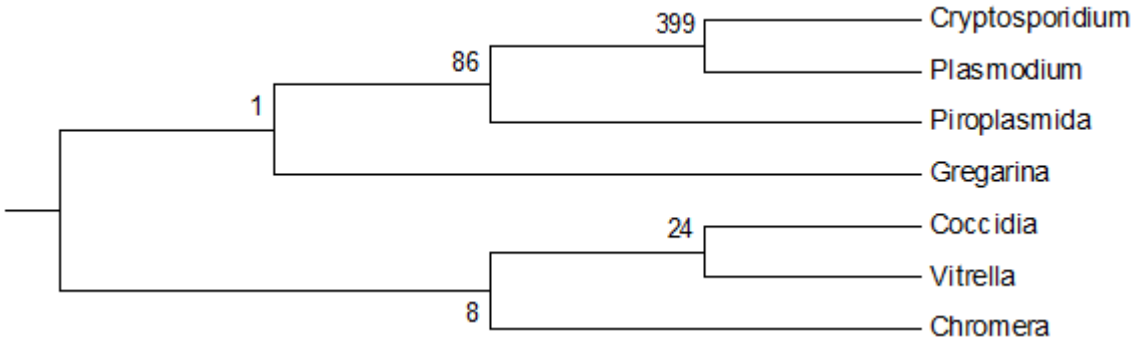


Рис.2. *Дерево видов из надтипа Alveolata, построенное по ультраконсервативным участкам. Расстояние по дереву между видами монотонно убывает при увеличении числа общих ультраконсервативных элементов* (*число общих элементов указано в корне поддерева для всех его листьев*)

Полученные результаты позволяют сделать вывод об эффективности разработанного нами алгоритма для поиска ультраконсервативных элементов генома, а также уточнить филогенетическое положение некоторых простейших из надтипа Alveolata.

Важно, что поиск ультраконсервативных участков не требует предварительной аннотации генома. Он может быть выполнен сразу после сборки достаточно длинных контигов.

Выводы

1. Подтверждено, что род *Cryptosporidium* не входит в класс кокцидий, а является близким родственником плазмодиев, пироплазмид (бабезии и тейлерии) и *Gregarina niphandrodes*. Это согласуется с недавними результатами других авторов [9, 10], хотя в прошлом криптоспоридии объединялись систематиками вместе с кокцидиями.
2. Подтверждён известный факт, что фотосинтезирующие простейшие *Chromera velia* и *Vitrella brassicaformis* близкие родственники [11].
3. В составе типа Apicomplexa кокцидии сохранили большее число элементов, присутствовавших у общего предка надтипа Alveolata. В частности, *Chromera velia* и *Vitrella brassicaformis* ближе к кокцидиям, чем к другим классам в составе типа Apicomplexa.
4. Подтверждено, что вид *Sarcocystis neurona* близок к другим кокцидиям.

Работа выполнена за счёт гранта Российского научного фонда (проект № 14–50–00150).

Литература

1. Bejerano G.., Pheasant M., Makunin I., Stephen S., Kent W.J., Mattick J.S., Haussler D. Ultraconserved elements in the human genome // *Science*, 2004, vol. 304, no. 5675, pp. 1321–13254.
2. Makunin I.V., Shloma V.V., Stephen S.J., Pheasant M., Belyakin S.N. Comparison of Ultra-Conserved Elements in Drosophilids and Vertebrates // *PloS one*, 2013, vol. 8, no. 12, e82362.
3. Садовская Т.А., Селиверстов А.В. Анализ 5'-лидерных областей некоторых генов пластид у простейших типа Apicomplexa и у красных водорослей // *Молекулярная биология*, 2009, том 43, № 4, стр. 599–604.
4. Zhu G., Marchewka M.J., Keithly J.S. *Cryptosporidium parvum* appears to lack a plastid genome // *Microbiol*. 2000, vol. 146, pp. 315–321.
5. Toso M.A., Omoto C.K. *Gregarina niphandrodes* may lack both a plastid genome and organelle // J *Eukaryot Microbiol*. 2007. vol. 54, no. 1, pp. 66–72.

6. Любецкий В.А., Селиверстов А.В. Некоторые алгоритмы, связанные с конечными группами // *Информационные процессы*, 2003, том 3, №1, стр. 39–46.
7. Burge S.W., Daub J., Eberhardt R., Tate J., Barquist L., Nawrocki E.P., Eddy S.R., Gardner P.P., Bateman A. Rfam 11.0: 10 years of RNA families // *Nucleic Acids Research*, 2013, vol. 41, Database issue, D226–232.
8. Zwieb C. The uRNA database // *Nucleic Acids Research*, 1997, vol. 25, no. 1, pp. 102–103.
9. Barta J.R., Thompson R.C.A. What is *Cryptosporidium*? Reappraising its biology and phylogenetic affinities // *Trends in Parasitology*, vol. 22, no. 10, pp. 463–468.
10. Bachvaroff T.R., Handy S.M., Place A.R., Delwiche C.F. Alveolate Phylogeny Inferred using Concatenated Ribosomal Proteins // *J. Eukaryot. Microbiol*., 2011, vol. 58, no. 3, pp. 223–233.
11. Oborník M., Lukeš J. The Organellar Genomes of *Chromera* and *Vitrella*, the Phototrophic Relatives of Apicomplexan Parasites // *Annu. Rev. Microbiol*., 2015, vol. 69, pp. 129–144.

Рубанов Л.И.[1], Селиверстов А.В.[2], Любецкий В.А.[3]

[1]ИППИ РАН, г. Москва, к.т.н., в.н.с., rubanov@iitp.ru

[2]ИППИ РАН, г. Москва, к.ф.-м.н., в.н.с., slvstv@iitp.ru

[3]ИППИ РАН, г. Москва, д.ф.-м.н., зав.лаб., lyubetsk@iitp.ru

ШИРОКОМАСШТАБНЫЙ ПОИСК УЛЬТРАКОНСЕРВАТИВНЫХ ЭЛЕМЕНТОВ В ПОЛНЫХ ГЕНОМАХ

КЛЮЧЕВЫЕ СЛОВА

Большие данные, высокопроизводительные вычисления, полный геном, ультраконсервативный элемент, полулокальное выравнивание, плотный подграф, кластеризация.

АННОТАЦИЯ

Описывается метод совместного анализа большого числа полных геномов с целью нахождения в них ультраконсервативных элементов – похожих друг на друга (не обязательно тождественных) участков, встречающихся сразу во многих геномах. Задача связана с обработкой больших данных, для ее решения разработан ряд оригинальных параллельных алгоритмов с линейной или близкой к ней сложностью.

Введение и постановка задачи. Задача состоит в поиске ультраконсервативных элементов в данном наборе геномов, последовательностей порядка 3 миллионов – 6 миллиардов букв в четырёхбуквенном алфавите. Число геномов может достигать нескольких сотен. Нужно кластеризовать слова, подпоследовательности небольшой длины, взятые из данных последовательностей, таким образом, чтобы один кластер состоял из «почти одинаковых» слов. Последнее означает, что каждый кластер, называемый ещё ультраконсервативным элементом, допускает хорошее выравнивание входящих в него слов, т.е. редакционное расстояние между ними небольшое. Забегая вперёд, отметим, что типичная возникающая трудность состоит в том, чтобы не было слишком больших («гигантских») кластеров. Это зависит от правильного выбора параметров задачи, что само по себе является нетривиальной задачей.

Задача относится к числу типичных трудных задач обработки больших данных, о чем свидетельствуют характерные размеры полных геномов, а также число уже известных геномов и темпы расшифровки новых. На неформальном уровне каждый такой элемент представляет собой участок ДНК, который встречается одновременно во многих геномах, но не обязательно в виде точной копии, а с возможностью замен, вставок и удалений отдельных букв. Таким образом, задача соединяет в себе две важнейшие задачи – кластеризацию слов и построение их множественного выравнивания.

Более строго, рассматривается следующая задача. Даны *M* символьных последовательностей над конечным алфавитом Σ. Любую конечную последовательность $A = \langle a_1 a_2 \ldots a_N \rangle$ будем называть строкой, а произвольный ее отрезок между позициями *i* и *j* включительно – подстрокой (или *словом*), которую обозначим $A[i..j] \subseteq A$. Длины строк могут различаться, но не превосходят заданной величины *N*; без ограничения общности можно считать длину каждой строки равной *N*. Для пары слов определено редакционное расстояние [1, 2] λ, используя фиксированные цены элементарных операций, которые предполагаются симметричными и однородными, т.е. цена вставки равна цене удаления ($\delta_i = \delta_d = \delta_{id}$) и цены всех замен равны между собой. Часто предполагают также, что цена вставки/удаления больше цены замены, $\delta_{id} > \delta_s$, но для простоты изложения будем временно считать $\delta_{id} = \delta_s = 1$; при этом редакционное расстояние между двумя словами равно минимальному числу операций, преобразующих одно слово в другое.

Определение. Будем говорить, что слово $w = \langle w_1 w_2 \ldots w_l \rangle$ *встречается* в данной строке

$A = \langle a_1 a_2 \ldots a_N \rangle$, если в ней существуют две позиции $1 \le i \le j \le N$, такие, что редакционное расстояние $\lambda(w, A[i,j]) \le \varepsilon$, где ε – заданный порог. Для таких случаев будем использовать обозначения $w \in A, w \approx A[i..j]$.

Задача состоит в том, чтобы найти все слова длины не менее $l = N$, которые встречаются, по меньшей мере, в $m \le M$ строках, с указанием их позиций в каждой строке.

Согласно определению, здесь под «встречей» понимается не только точное совпадение слов, т.е. равенство алфавитных символов на соответственных позициях, но и приблизительное совпадение, когда слова неодинаковы, но близки в смысле редакционного расстояния. При этом порог ε предполагается небольшим, например, не более 6 редакционных операций (замен, вставок, удалений) при длине $l = 150$. Основная сложность вытекает из размерности задачи: длина строк *N* очень велика (общая длина полного генома составляет порядка 10^9), а число строк *M* может достигать сотен. Трудоемкость прямого перебора квадратичная и составляет порядка $O(M^2 N^2 l^2)$ операций сравнения, т.е. свыше 10^{25}, а для параллельного перебора необходима общая память размером по меньшей мере $O(MN)$ – терабайтного объема. Это не позволяет решать такую задачу даже с использованием современных суперкомпьютеров. Требуется более эффективная технология, основанная на применении быстрых алгоритмов и лучше приспособленная к распараллеливанию. С теоретико-графовых позиций предлагаемый метод состоит из двух этапов.

На *первом этапе* анализируются две строки и в них выделяются все пары искомых слов-кандидатов (для определенности – слов второй строки, которые встречаются в первой строке). Каждая найденная пара слов задает ребро будущего графа, в котором вершины соответствуют словам, а ребрами соединены вершины, для которых редакционное расстояние между соответствующими словами не превосходит ε. Для поиска разработан быстрый алгоритм с линейной (или почти линейной) сложностью. При наличии в составе вычислительной системы $M(M-1)/2$ процессоров данный этап может выполняться параллельно для каждой неупорядоченной пары строк, независимо от прочих пар. Однако алгоритм наиболее эффективен, когда в каждой параллельной ветви выполняется сравнение одной строки со всеми оставшимися, для чего достаточно $M-1$ процессоров. Возможны также и промежуточные по числу ветвей варианты, позволяющие лучше сбалансировать неоднородную вычислительную нагрузку между процессорами; этот вопрос исследован нами в [3]. Результатом первого этапа являются независимые ребра, которые в совокупности образуют *M*-дольный граф низкой плотности (в котором число ребер ближе по порядку к числу вершин, чем к квадрату числа вершин), обычно состоящий из большого числа связных компонент.

На *втором этапе* проводится уплотнение начального графа с последующим выделением в нем плотных подграфов (в идеале – являющихся кликами) с вершинами, принадлежащими *m* и более различным долям. Каждый такой *m*-плотный подграф в целом описывает одно найденное искомое слово; вершина этого подграфа, относящаяся к *k*-й доле начального графа, определяет позицию этого слова в *k*-й исходной строке. Совокупность всех найденных *m*-плотных подграфов дает решение задачи. Уплотнение графа выполняется путем склейки вершин, которым соответствуют одни и те же или сильно пересекающиеся (не менее чем на длине *d*) слова, с одновременным удалением возникающих кратных ребер. Для поиска *m*-плотных подграфов в большом графе разработан новый параллельный алгоритм, построенный по принципу клеточного автомата, в котором каждой вершине исходного графа соответствует свой многошаговый процесс, однако эти процессы исполняются с единой синхронизацией шагов. Как показали эксперименты на реальных данных, даже для очень больших графов (например, порядка 10^7 вершин и 10^9 ребер) алгоритм завершается после небольшого числа шагов (обычно менее сотни), и гибко масштабируется для любого числа доступных процессоров. В следующих разделах излагаются подробности указанных двух этапов метода и описываются лежащие в их основе алгоритмы.

Поиск слов, встречающихся в двух строках. Вначале уточним ряд моментов, опущенных выше для краткости. Во-первых, из введенного определения встречаемости следует, что для слова $w \in A$ любое его подслово также встречается в этой строке: $u \subseteq w \Rightarrow u \in A$. Поэтому для уменьшения числа решений задачи естественно потребовать, чтобы искомые слова были

непродолжаемыми, т.е. никакое найденное слово не было подстрокой другого найденного слова. Однако такое требование может вступить в конфликт с основной целью задачи. В самом деле, легко представить ситуацию, когда некоторое слово *w* встречается в *m* исходных строках, а его подслово $u \subseteq w$ – в $m' > m$ строках. Оба эти решения могут представлять практический интерес, поэтому ограничим требование непродолжаемости слова пределами каждого выявленного множества из *m* и более строк, в которых встречается это слово.

Во-вторых, на практике часто представляют интерес не любые слова, широко встречающиеся в исходных строках, а только обладающие достаточной *колмогоровской сложностью*. Например, в полном геноме иногда присутствуют длинные подстроки, состоящие из многих тысяч повторений одного-двух алфавитных символов, например, АТ-повторы. Найденные в таких подстроках элементы обычно не представляют интереса и должны быть отброшены. Как показано ниже, это также помогает снизить затраты памяти и ускорить работу алгоритма. Для оценки сложности найденных кандидатов использован алгоритм сжатия данных Зива–Лемпеля [4] в реализации GNU Gzip [5] с установленным порогом для коэффициента компрессии *r*; величина порога определялась эмпирически.

Основная трудность этого этапа связана с квадратичной (от длины строки) сложностью непосредственного сравнения слов двух последовательностей. Известные быстрые алгоритмы поиска подстроки в строке в данном случае неприменимы, поскольку способны отыскивать только точно совпадающие слова. Требовался алгоритм более низкой сложности, который был разработан на базе трех принципов:

1) *Индексация строк.* Наивный квадратичный алгоритм проверяет для каждого слова строки *B*, встречается ли оно в одной и той же строке *A*. Ясно, что такую проверку можно в среднем существенно ускорить с помощью надлежащей индексации строки *A*, которая проводится один раз, а затем многократно используется. Эта идея хорошо известна и широко используется, например, поисковыми машинами Интернет. Подробности разработанных для данной задачи индексов обсуждаются ниже.

2) *Предварительный поиск по ключу*. В рассматриваемой постановке ищутся элементы с высокой консервативностью, так что порог допустимого редакционного расстояния между искомыми словами предполагается низким: различия между любыми двумя словами устраняются не более чем за ε операций вставки, удаления или замены одного символа. Отсюда следует, что искомые слова гарантированно содержат *точно* совпадающие подслова некоторой длины *k*, которые можно использовать в качестве обычных поисковых ключей. Например, для указанных выше значений параметров $\varepsilon = 6,\ l = 150$ длина ключа может составлять $k \approx 30$.

3) *Полулокальное выравнивание*. После поиска по точному ключу каждое подслово-кандидат подлежит расширению в обе стороны с соблюдением порога ε, пока не будет достигнута общая длина не менее *l*, иначе кандидат отвергается. Для нахождения редакционного расстояния между словами обычно применяется алгоритм Нидлмана–Вунша [6] со сложностью $O(l^2)$. Этот алгоритм глобального выравнивания предполагает концы слов зафиксированными, тогда как в нашем случае известно только начало или конец слова, в зависимости от того, в какую сторону выполняется расширение. Использование общих алгоритмов локального выравнивания (Смита–Уотермена, BLAST и др.) в рассматриваемом случае явно избыточно, коль скоро один из концов каждого слова фиксирован (примыкает к ключу). На основе алгоритма [6] был разработан полулокальный алгоритм выравнивания слов с одним закрепленным концом, причем благодаря учету порога ε и ограничению числа подряд идущих делеций *D*, созданный алгоритм имеет пространственно-временную сложность $O(Dl)$, т.е. является линейным по *l*, в отличие от исходного квадратичного алгоритма.

Пусть *k* – длина ключа, определенная в соответствии со вторым принципом. Проиндексируем строку *A* следующим образом. Строится хеш-таблица H_A, в которую поочередно заносятся все присутствующие ключи, т.е. слова длины *k* в составе строки *A,* вместе с позицией их начала в строке:

$$H_A = \left\{ h_A(i) \in \left(K_A, j\right) \mid K_A = A[j..(j+k-1)], i = Hash\left(K_A\right) \right\},$$

где $Hash(K_A)$ – хеш-функция, преобразующая значение очередного ключа в адрес внутри хеш-таблицы. При построении хеш-таблицы неизбежно возникают коллизии, которые приводят к

снижению производительности алгоритма и в данном случае имеют двоякую природу: (а) возникающие из-за несовершенства хеш-функции и (б) вызванные повторениями одного и того же слова длины *k* в исходной строке. Частоту появления коллизий первого рода можно пытаться снизить путем уменьшения коэффициента заполнения таблицы и/или выбора другой хеш-функции, но коллизии второго рода неустранимы в принципе, хотя с увеличением *k* их частота естественным образом снижается. Частоту коллизий второго рода для реальных больших данных (полный геном *Sarcocystis neurona*, длина строки *N* = 124377056, размер алфавита $|\Sigma| = 4$) иллюстрирует табл. 1. В этом примере при длине ключа $k \geq 24$ коллизии возникают в среднем менее чем в 4% случаев, но при уменьшении длины ключа до 16 эта доля превосходит 20%. Многократно повторяющиеся ключи обычно принадлежат участкам генома с низкой колмогоровской сложностью, которые в дальнейшем будут отброшены, однако для ускорения мы используем порог *t* для числа встреч ключа в анализируемой последовательности. Поскольку коллизии неизбежны, хоть и достаточно редки, для построения индекса выбран вариант хеш-таблицы с цепочками. Размер хеш-таблицы определяется исключительно длиной строки *A* и желаемым коэффициентом заполнения. Поскольку строка *B* на этом этапе не участвует, индекс для каждой из *M* исходных строк может быть построен заранее (и параллельно), а затем многократно использоваться со всевозможными *B*.

Таблица 1. Число повторяющихся ключей в строке большой длины

Число вхождений ключа	*k*=16	*k*=24	*k*=32	*k*=48
1	94919216	119034327	121193592	121823700
2	5284353	764077	401812	242260
3, 4	1438861	190900	65945	24494
5–8	486137	60275	15503	4875
9–16	183799	20869	4129	1116
17–32	72581	7429	1202	194
33–64	29861	2996	437	155
65–128	11196	1217	208	85
129–256	4986	498	86	41
257–512	1776	148	9	3
513–1024	739	67	8	6
Более 1024	447	90	3	1
Число различных ключей	102433952	120082826	121682934	122096930
Среднее число вхождений	1.21408	1.03559	1.02191	1.01833

С построенной хеш-таблицей H_A, для каждой строки *B* выполняется следующий алгоритм.

1. Поочередно от каждой позиции *j* строки *B* берем слово $K_j = B[j..(j+k-1)]$, которое будет использоваться в качестве ключа для обращения к хеш-таблице.
2. Проверяем, присутствует ли этот ключ в H_A; в случае отсутствия переходим на шаге 1 к следующей позиции *j*.
3. Если существует такое *s*, что $h_A(s) = (K_j, i)$, то найдена новая пара кандидатов (*i*, *j*): на позиции *i* в строке *A* и на позиции *j* в строке *B*. Если элемент $h_A(s)$ содержит цепочку позиций *j* и имеет место коллизия второго рода, то перебираются все пары (*i*, *j*), пропуская элементы с другими значениями ключа (при коллизии первого рода).
4. Для найденной пары кандидатов проверяем возможность расширения ключей до искомых слов и в случае успеха запоминаем их. Алгоритм расширения подробно описывается ниже, здесь просто предположим, что его сложность равна некоторой константе *C*.
5. Переходим на шаге 1 к следующей позиции *j* строки *B* вплоть до конца строки.

Если не учитывать коллизии, сложность алгоритма $O(N)\cdot C$, т.е. в среднем линейная от длины строки. Используемая память $O(N)$ также линейная, но и это может создавать трудности при очень больших *N*. Однако имеется важный резерв экономии памяти, связанный с представлением ключей в хеш-таблице. В случае короткого алфавита хешируемые ключи можно перекодировать с использованием меньшего числа бит на символ и хранить в упакованном виде в

меньшем числе байтов. Поскольку решаемая задача относится к строкам из 4-буквенного алфавита нуклеотидов (A, C, G, T), то этим способом затраты памяти на хранение ключей в хеш-таблице удается снизить в четыре раза. В этом случае удобно выбирать k кратным 4. Перед проверкой по хеш-таблице слова-кандидаты подвергаются такому же преобразованию.

В п. 4 алгоритма для найденной пары (i, j) позиций ключа в строках A и B проверяется возможность расширения ключей до искомых слов, а именно, в каждой строке ищутся положения начала, $i_1 \le i, j_1 \le j$ и конца, $i_2 \ge i+k-1, j_2 \ge j+k-1$, такие что $i_2 - i_1 \ge l-1, j_2 - j_1 \ge l-1$ и $A[i_1..i_2] \approx B[j_1..j_2]$. Согласно определению выше, последнее условие подразумевает редакционное расстояние не более ε между расширенными словами. Вариантов такого расширения может быть много; мы хотим выбрать тот, который доставляет максимум величины $L = i_2 - i_1 + j_2 - j_1$ (суммарная длин слов), что обеспечивает непродолжаемость. Для этого одновременно с расширением ключа в каждую сторону строится полулокальное выравнивание добавляемых подстрок и запоминаются те позиции, на которых редакционное расстояние между ними изменяется. Расширение выполняется параллельно в обе стороны и продолжается до тех пор, пока не будет превышен порог ε или достигнуто начало (конец) строки. В двух полученных списках позиций выбирается лучший по суммарной длине вариант расширения вправо и влево, удовлетворяющий всем вышеприведенным условиям; отсутствие такого варианта означает неуспех проверки. Учитывая малость ε, перебор всех вариантов не создает трудностей.

Опишем алгоритм расширения ключа с полулокальным выравниванием в одну сторону, для определенности – от концов ключей в сторону конца строк. Как известно, классический алгоритм [6] выравнивания двух строк с длинами m и n опирается на вычисление двух матриц F, P порядка $(m+1)(n+1)$, откуда и проистекает его квадратичная пространственная и временная сложность. Этот алгоритм неэффективен для нашего применения, потому что позиции строк A и B, до которых расширяется ключ, заранее неизвестны, а матрицы требуется вычислить полностью, несмотря на наличие известного порога ε для максимума редакционного расстояния. Для устранения этого недостатка внесены следующие изменения (рис. 2): в каждой матрице выделена область, прилегающая к главной диагонали и содержащая дополнительно по D соседних диагоналей сверху и снизу (рис. 2, а). Смысл параметра D – максимально допустимое число подряд идущих делеций. Назовем такую область D-поясом. Матрица редакционного предписания нужна для составления вышеупомянутого списка позиций расширения, на которых редакционное расстояние возрастает.

Заметим, что траектория обратного хода алгоритма оптимального выравнивания обязана лежать внутри D-пояса независимо от позиции, до которой расширяется ключ в данную сторону, иначе нарушается ограничение на число подряд идущих делеций, т.к. каждый сдвиг в сторону от диагонального направления в матрице соответствует вставке или удалению символа.

Разместим выделенные D-пояса в двух матрицах G и Q порядка $(AD+1)\times(V+1)$, где V – заранее неизвестная граница расширения вправо (рис. 2, б)). Переход от координат u, v в матрицах G, Q к координатам m, n в матрицах F, P и строках A, B осуществляется по формулам $m = v + (u-D)^+$, $n = v + (D-u)^+$ где $(D-u)^+$ означает положительную часть. С учетом этого преобразования алгоритм выравнивания строится как обычно, но очередность вычисления элементов матриц и смысл направлений указателей соответствуют указанным на рис. 2, в). Временная и пространственная сложность этого алгоритма равны $O(VD) = O(l)$, поскольку $D = l$. Таким образом, для проверки и расширения слов-кандидатов построен *линейный* от длины слов алгоритм.

Суммируя изложенное, заключаем, что общая сложность первого этапа нашего метода составляет $O(Nl)$ для одной пары строк. Для всех пар строк эта величина умножается еще на $O(M^2)$, но пары строк можно обрабатывать параллельно – по отдельности или группами, как было указано выше. Перейдем теперь ко второму этапу нашего метода.

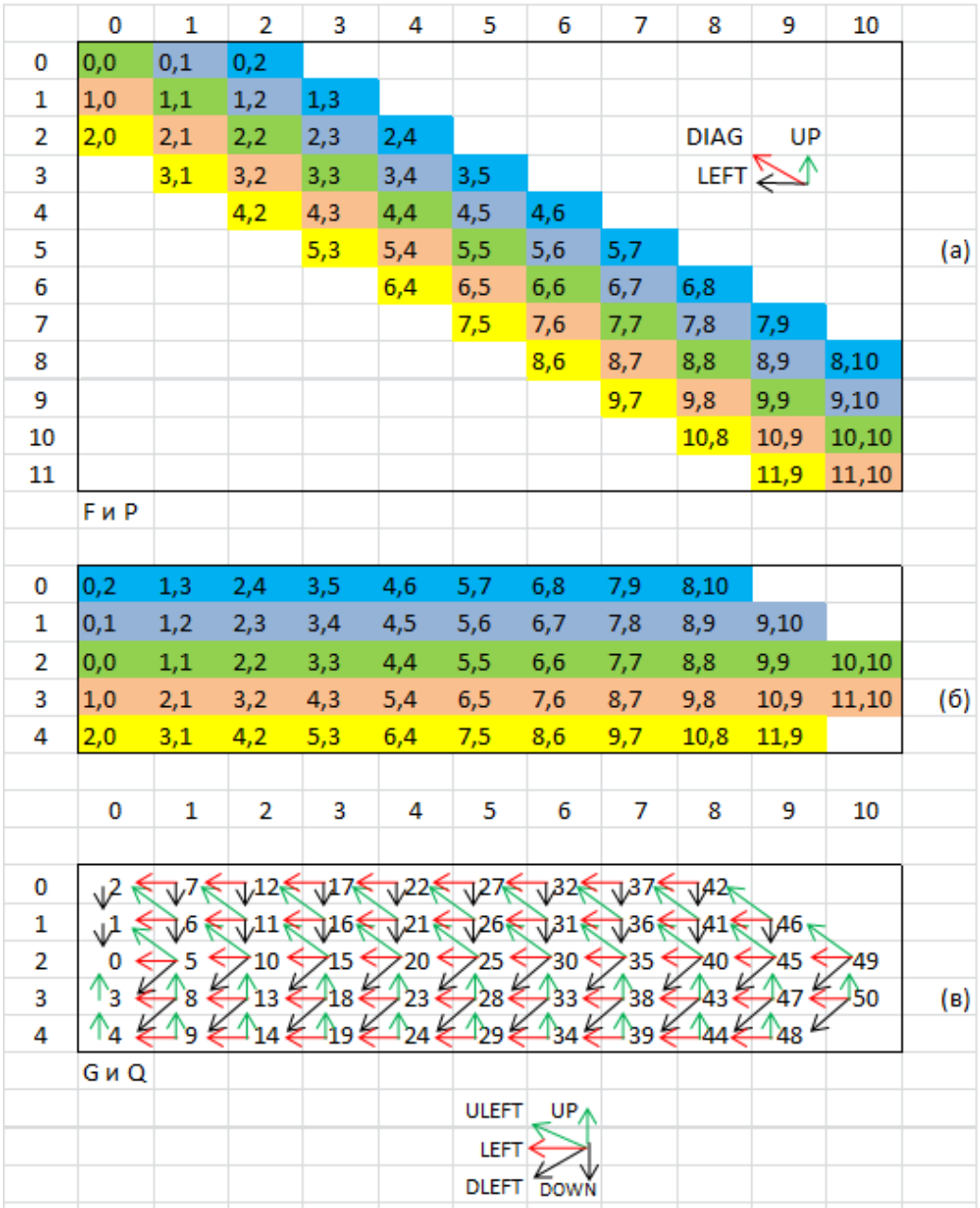


Рис. 2. Преобразование матриц алгоритма Нидлмана–Вунша: (а) вид исходной матрицы с выделенным D-поясом для m=11, n=10, D=2; в клетках записаны координаты элементов; (б) расположение D-пояса в преобразованной матрице; (в) очередность вычисления элементов преобразованной матрицы и направления указателей обратного хода в матрице редакционного предписания.

Нахождение *m*-плотных подграфов в *M*-дольном графе. После первого этапа имеем совокупность пар слов, которые принадлежат двум различным исходным строкам A и B и совпадают с точностью до редакционного расстояния ε: $A[i_1..i_2] \approx B[j_1..j_2]$. Сопоставим каждой такой паре ребро графа; этот граф содержит M долей, вершины которых соединены ребрами (внутри долей ребер нет). Рассматриваемая задача в целом эквивалентна нахождению в этом графе m-клик, т.е. подграфов с вершинами, которые принадлежат m долям, и каждая вершина соединена ребрами со всеми вершинами прочих $m-1$ долей. Это известная NP-трудная задача, для которой не существует быстрого алгоритма. Поэтому в рамках нашего метода решается близкая задача нахождения m-плотных подграфов, прежде всего – с наибольшим суммарным весом ребер. В качестве веса ребра естественно использовать монотонную функцию от длины и/или степени консервативности слов. Предполагается, что среди найденных решений этой

задачи можно по очереди отобрать присутствующие среди них клики, либо уже перебором выделить внутри них подграфы, являющиеся *m*-кликами.

Поскольку расширение точно совпадающих слов, т.е. построение вершин на конце ребра выше проводилось независимо от других ребер, на конце нескольких ребер могут стоять вершины, относящиеся к одному и тому же месту некоторой строки, но с несовпадающими концами слов и потому формально различные. Это снижает плотность всего графа и ведет к ненужному увеличению числа компонент связности, поэтому вначале проводится операция склейки вершин. При этом требуется, чтобы все объединяемые в одну вершину участки последовательности не просто пересекались попарно, а для любой пары содержали один и тот же общий фрагмент длиной не менее *d*.

После того, как выполнены все возможные операции склейки вершин, выполняется собственно алгоритм нахождения *m*-плотных подграфов. Этот алгоритм характеризуется глубоким внутренним параллелизмом и построен аналогично клеточному автомату, у которого в роли клеток выступают вершины графа, а роль соседей каждой клетки играют инцидентные ей вершины графа. В каждой вершине графа запускается независимый процесс, который состоит из последовательности шагов двух видов и начинается с шага 1. После каждого шага выполняется синхронизация процессов во всех вершинах графа, так что очередной шаг каждого процесса начинается только после окончания предыдущего шага во всех вершинах. Содержание шагов:

- Если вершина соединена ребрами менее чем с $(m-1)$ долями графа, то сама эта вершина и все инцидентные ей ребра удаляются.
- Если вершина соединена с некоторой долей только одним ребром, то помечаем его; в дальнейшем помеченное ребро может быть удалено только вместе с одним из его концов.
- Если при выполнении шага 1 в графе произошли какие-либо изменения, то снова выполняется шаг 1 во всех вершинах графа. В противном случае в каждой вершине однократно выполняется шаг 2.
- Если инцидентное вершине ребро не помечено и его вес строго меньше весов всех других не помеченных инцидентных ребер (или оно единственное), то ребро удаляется.
- Если при выполнении шага 2 в графе произошли какие-либо изменения, то все вершины снова переходят к шагу 1, иначе конец алгоритма. Каждая компонента связности получившегося графа есть искомый *m*-плотный подграф.

Данный алгоритм предоставляет гибкие возможности масштабирования вплоть до числа процессоров, равного числу вершин графа, и не требует общей памяти. Если число доступных процессоров меньше числа вершин, они равномерно распределяются между имеющимися процессорами, а затем каждый из них выполняет очередной шаг процесса по очереди для каждой назначенной ему вершины. Для этого алгоритма трудно получить аналитические оценки сложности, но это не имеет большого значения: практические расчеты с реальными большими данными показали, что второй этап предложенной технологии выполняется значительно быстрее первого.

Однако серьезная трудность состоит в том, что при неудачном выборе параметров или способа построения исходного графа в нём возникает гигантская связная компонента, объединяющая значительную часть вершин графа. Критериями успеха служат отсутствие гигантской компоненты и наличие *m*-плотных подграфов для значений *m*, больших половины от числа долей графа, то есть числа геномов. Как показали эксперименты, риск появления и размеры гигантской компоненты уменьшаются, если предварительно удалить из графа все ребра с весом меньше некоторого порога, но вопрос требует дальнейшего изучения. В любом случае, существование гигантской компоненты в исходном графе не обязательно является препятствием к выделению *m*-плотных подграфов с малым числом вершин при достаточно большом *m*.

Пример использования. Анализировались полные геномы 22 видов из надтипа Alveolata (максимальная длина генома составляла $1{,}24*10^8$). Счет проводился для значений параметров $l=60$, $k=16$, $t=200$, $\delta_s=1$, $\delta_{id}=2.1$, $\varepsilon=17.5$, $D=2$, $r=2.2$, $d=40$, $m=3$. Первый этап изложенного метода выполнялся в МСЦ РАН на суперкомьютерах МВС-100К и МВС-10П [7] с использованием 256–512 процессоров и потребовал около 200 часов. В результате было найдено $7.75\cdot10^8$ подходящих пар участков (общее число участков $4.54\cdot10^7$). Второй этап выполнялся в ИППИ РАН на 64-ядерном сервере с 256 Гб оперативной памяти. Сборка графа потребовала 13 час. на 16 процессорах, в результате был построен граф с 4977108 вершинами (и

тем же числом ребер $7.75 \cdot 10^8$, т.к. кратные ребра удалялись в процессе поиска плотных подграфов). Поиск плотных подграфов продолжался 13 час. на 22 процессорах, в результате чего было построено 845 кластеров, в которые вошли 3924 вершины (помимо гигантской компоненты, содержащей около 3 млн. вершин). Данные о числе кластеров в зависимости от числа долей, представленных в кластере: 2 с представителями из 13 долей, 1 – из 12 долей, 4 – из 11 долей, 8 – из 10 долей, 17 – из 9 долей, 24 – из 8 долей, 35 – из 7 долей, 75 – из 6 долей, 131 – из 5 долей, 290 – из 4 долей, 258 – из 3 долей. Подавляющая часть кластеров (787) содержит не более одной вершины из каждой доли, и только 9 кластеров содержат более двух вершин из какой-нибудь доли (максимум 5 вершин). Отметим, что значительная часть машинного времени была потрачена на чтение и запись огромных файлов. Анализ обнаруженных ультраконсервативных элементов и полученные биологические результаты представлены в отдельном докладе.

В заключение укажем, что описанный метод поиска ультраконсервативных элементов может быть применен и в других областях, где используется сопоставление любых текстов, в том числе на естественном языке, с целью определения авторства, стиля, заимствований и т.д. [8].

Работа выполнена за счёт гранта Российского научного фонда (проект № 14–50–00150).

Литература

1. Левенштейн В.И. Двоичные коды с исправлением выпадений, вставок и замещений символов // Доклады Академии Наук СССР. 1965, т. 163, № 4, стр. 845-848.
2. Gusfield D. Algorithms on Strings, Trees, and Sequences: Computer Science and Computational Biology. Cambridge University Press, Cambridge, UK, 1997.
3. Рубанов Л.И. Параллельное моделирование Монте-Карло на системах с распределённой памятью // *International Journal of Open Information Technologies*, 2014, т. 2, № 2, стр. 12-20.
4. Ziv J., Lempel A. A universal algorithm for sequential data compression // *IEEE Transactions on Information Theory*. 1977, vol. 23, no. 3, pp. 337-343.
5. GNU Operating System. GNU Gzip. http://www.gnu.org/software/gzip/
6. Needleman S.B., Wunsch C.D. A general method applicable to the search for similarities in the amino acid sequence of two proteins // *Journal of Molecular Biology*. 1970, vol. 48, no. 3, pp. 443-453.
7. Межведомственный суперкомпьютерный центр РАН. http://www.jscc.ru/scomputers.shtml
8. Herranz J., Nin J., Solé M. Optimal Symbol Alignment Distance: A New Distance for Sequences of Symbols // *IEEE Transactions on Knowledge & Data Engineering*, 2011, vol. 23, no. 10, pp. 1541–1554.

Истомина С.Н.

Московский авиационный институт (национальный исследовательский университет), Москва, Оршанская ул., д. 3, к.х.н., доцент кафедры Прикладная математика, информационные технологии и электротехника (495)-141-95-57, pm@mati.ru

СРАВНИТЕЛЬНЫЙ АНАЛИЗ СВЯЗАННЫХ РЯДОВ: ДЛИН ОРТОЛОГИЧНЫХ БЕЛКОВ И ИХ ПРИРАЩЕНИЙ

КЛЮЧЕВЫЕ СЛОВА

Ортологичные белки, базисный белок, связанные ряды, длина белка, кластерный анализ, метод главных компонент, корреляция, регрессия.

АННОТАЦИЯ

Работа посвящена изучению длин ортологичных белков четырех организмов, один из которых базисный. Параллельно проводился анализ приращений длин белков относительно длин базисного. Представление данных в виде матриц, из рядов длин упорядоченных по базисному белку со строками по четыре, позволило выделить кластеры длин с различным свойствами рядов по кластерам, для обоснования этого использованы также метод главных компонент и корреляционный анализ.

В работе [1] приведены значения длин для большого числа разных ортологичных генов – белков (от 260 до 350 значений каждый ряд) у четырех организмов, один из которых является базисным. Но исследование проводилось отдельно для пар рядов приращений длин белков относительно базисного.

В данной работе все необходимые в ходе исследования расчеты проводились при одновременном изучении четырёх рядов длин ортологичных белков аналогично связанным рядам динамики. В качестве связующего рассматривался ряд из длин белков базисного организма *S*.

Методы статистической обработки анализа многомерных данных применялись для определения взаимосвязей этих рядов для четырёх организмов *S*, *T*, *P* и *N*. Изучаются группы, составленные из четвёрок длин или их приращений, соответствующих белкам из *S*, которые обозначаются соответственно A1 и A2.

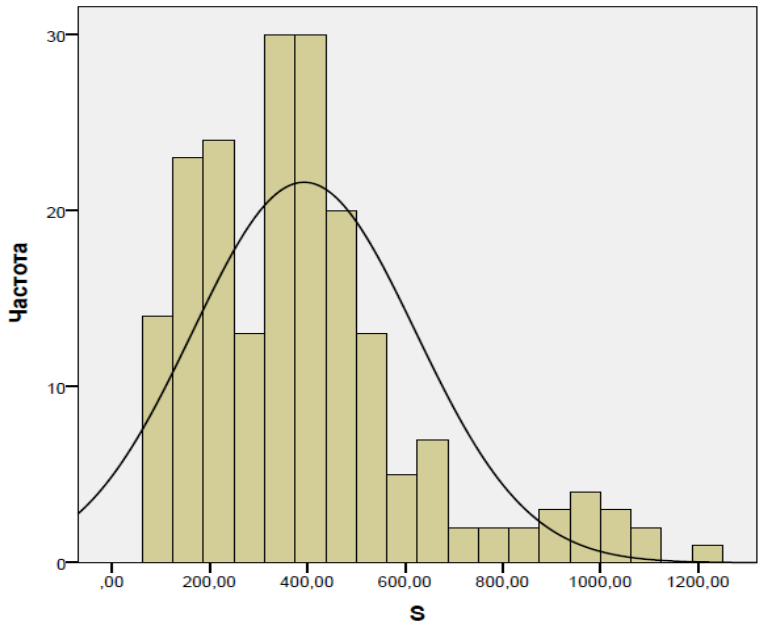


Рисунок 1. Гистограмма частот длин ортологичных белков S

1. Результаты кластерного анализа

Распределение длин ортологичных белков для каждого отдельного ряда с большой

достоверностью можно считать нормальным. Например, на рисунках 1 и 2 приведены гистограммы для длин ряда S и приращений длин ряда T-S.

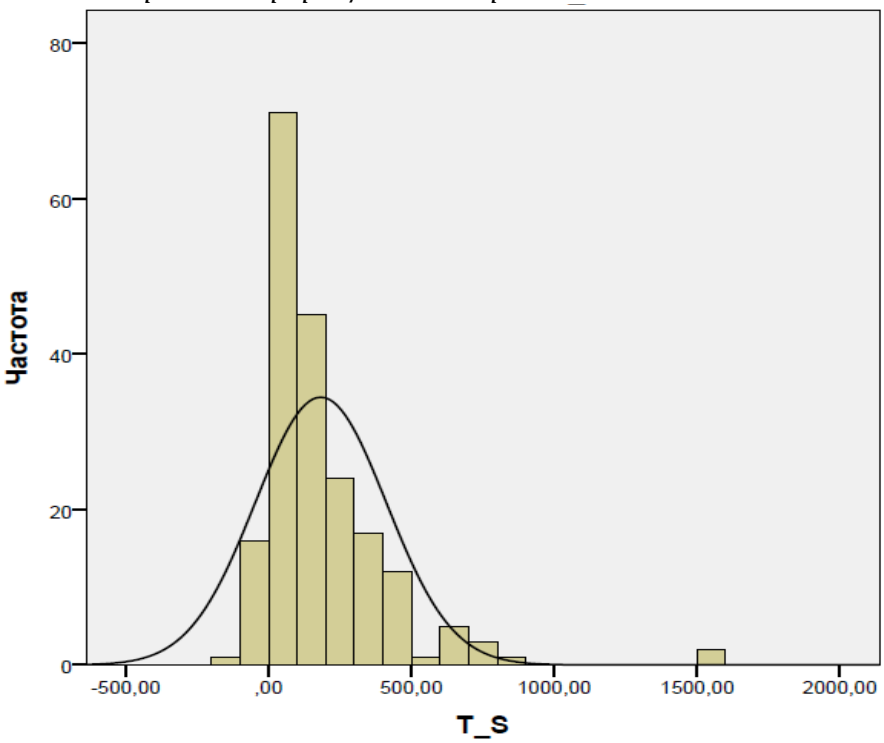


Рисунок 2. Гистограмма частот приращений длин ряда T-S

Статистические характеристики значений длин белков и их приращений для рассматриваемых рядов, представлены в таблице 1.

Таблица 1. Статистические характеристики значений длин белков и их приращений для рассматриваемых рядов

Ряд	Среднее значение	Среднее квадратическое отклонение	Значение дисперсии
S	393,3	228,41	52174,34
T	575,8	372,87	139034,31
N	521,6	358,66	128639,38
P	504,3	348,85	121698,33
T-S	182,5	229,27	52567,33
P-S	128,3	200,04	40016,89
N-S	110,9	175,91	30945,64

Таким образом, рассматриваются две группы с одинаковым числом наблюдений длин и их приращений. Однородность значений каждой группы (4 столбца по 198 значений) проверена по критерию Кочрена, гипотеза об их однородности принимается.

Если рассматривать ряды каждой группы, как связанные через базисный ряд, упорядоченный по убыванию значений, то порядок значений трех рядов зависит от порядка базисного ряда. Отсюда возникает предположение о возможном наличии для четвёрок белков, которые мы будем называть объектами исследования, разбиения группы на кластеры.

Для проверки такого предположения проведён кластерный анализ, чтобы с помощью группировки объектов и расчётов расстояний между объектами, выделить компактные и удалённые кластеры.

В качестве меры расстояния использовали квадратичную евклидову метрику, способствующую увеличению контрастности кластеров, и наиболее распространенный иерархический агломерационный метод.

Чтобы определить оптимальное число кластеров, необходимо определить, в какой момент мера расстояния между двумя кластерами увеличивается скачкообразно. Используя программный

пакет SPSS, мы по шагам получили обзорные таблицы агломерации и дендрограммы, отображающие процесс слияния и образования кластеров.

В результате для первой группы А1 длин белков получены 3 кластера, а для второй группы А2 длин приращений 2 кластера.

Принадлежность объекта к конкретному кластеру выясняли методом k-средних кластерного анализа, основанного на процедуре оптимального разбиения начальных данных на известное число кластеров. Эта процедура перемещает объекты из одного кластера в другой, чтобы минимизировать внутрикластерную и максимизировать межкластерную дисперсию.

Распределение по кластерам объектов группы А.1 показано в таблице 2.

Таким образом, группа А1 образует 3 кластера: первый из 158 строк - белки от самых малых до средних и больших, второй из 37 строк - белки, длины которых заметно больше средних и третий кластер из трех строк значений длин.

Таблица 2. *Распределение по кластерам объектов группы А.1*

Номер кластера	Количество объектов	Номера объектов в группе А1
1.	158	Номера объектов, не вошедшие в кластеры 2 и 3
2.	37	3,4,5,6, 8,9, 11, 14, 16, 19, 21, 23, 24, 27,29, 37, 46, 49, 52, 54, 62, 66, 84, 86, 100, 102, 103, 104, 107,108, 117, 118, 132, 150, 156, 191, 195
3.	37	1, 2, 10

Для группы А.2 тем же методом было получено два кластера: один кластер из 195 строк, а другой из 3 строк, тех же, что и в третьем кластере таблицы 4. Далее этот кластер не анализировался из-за малого числа объектов. ш

Отметим, что три объекта, выделенные в малочисленный кластер – одни и те же в матрицах А.1 и А.2, и выделяются также после ортогонального преобразования методом главных компонент.

2. Результаты метода главных компонент

Для проверки результатов кластерного анализа использован метод главных компонент на основе ортогонального преобразования исходных данных. Его результаты представляют упорядочение объектов, которое задаётся коэффициентами линейных комбинаций – главными компонентами.

Для расчётов использовался тот же пакет IBM SPSS Statistics.

В результате, как для рядов длин ортологичных белков, так и для рядов их приращений получена только одна главная компонента – одна линейная комбинация, характеризующая связь длин исчерпывающим образом. Эта компонента объясняет примерно 91% совокупной дисперсии для длин белков и около 68% совокупной дисперсии для приращений белков.

Эта главная компонента имеют вид для длин белков:

$$Y=0{,}933*S+0{,}954*T+0{,}968*N+0{,}965*P$$

и для приращений длин:

$$\widetilde{Y} =0{,}618*S+0{,}890*(T-S)+0{,}898*(N-S)+0{,}859*(P-S).$$

Отметим, что в первом случае коэффициенты всех четырёх рядов приблизительно равны, а во втором случае коэффициенты приращений также приблизительно равны.

Таблица 3. *Диапазоны значений главных компонент для рядов длин исходных белков с учетом кластеризации*

Кластер	Количество наблюдений	Минимальное значение	Максимальное значение	Среднее значение
1	158	281,16	3106,48	1441,8803
2	37	1958,11	5172,01	3459,8920
3	3	6511,72	7467,00	7105,965

Эти результаты можно интерпретировать, в частности, как оценку специфичности длин ортологичных белков в разных кластерах, что выражается чёткой отделённостью значений главной компоненты каждого кластера, см. Таблицу 3 для группы А1.

Отметим, что эти результаты хорошо согласуются с наличием кластеров, которым в качестве совокупной характеристики входящих в них четверок, можно присвоить указанные средние значения главной компоненты.

3. Результаты корреляционного анализа длин рядов по кластерам

Корреляционный анализ проведен для рядов длин и их приращений, указанных в таблицах А1, А2, как без учёта разделения на кластеры, так и отдельно по двум первым кластерам.

Для таблицы А1 получены весьма высокие значения парных коэффициентов корреляции и определена значимая автокорреляция рядов длин. Авторегрессия этих рядов ожидаемо оказалась значимой, что, как правило, означает наличие ложной корреляции (из-за общей связи с базисным рядом). После удаления авторегрессии определены уточнённые парные коэффициенты корреляции и получены линейные регрессии, адекватно описывающие связи рядов по кластерам, которые приведены на рис. 3-6.

Связь длин белка T с длинами белка S по каждому из двух кластеров (номер в индексе) описывается уравнениями регрессии: $T_1 = 1{,}01S + 130{,}81$ и $T_2 = 0{,}60S + 588{,}14$. Гипотеза об адекватности этих уравнений не отвергается по критерию Фишера: $F_1 = 146$ при $n_1 = 158$ и $F_2 = 10{,}58$ при $n_2 = 37$.

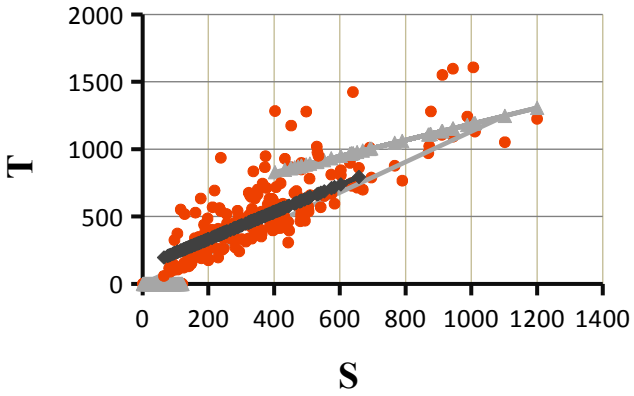


Рисунок 3. *Графики уравнений регрессии по двум кластерам длин белков T и S и точки с координатами, равными значениям длин этих белков*

На рисунке 3 представлены графики этих уравнений и эмпирические точки, координаты которых равны длинам соответствующих белков.

Отметим, что угловой коэффициент прямой по первому кластеру примерно равен 1, то есть в среднем длины белков T и базисного ряда отличаются на одну и ту же величину: величину свободного члена уравнения.

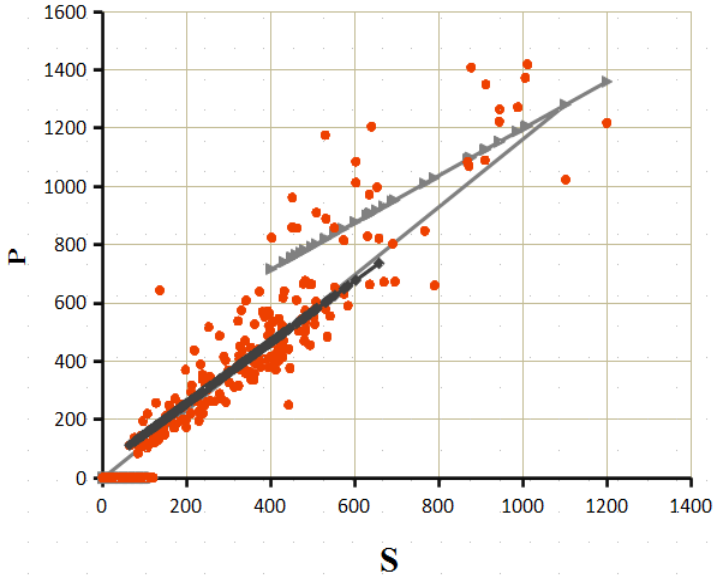


Рисунок 4. *Графики уравнений регрессии по кластерам длин белков P и S и точки с координатами, равными значениям длин этих белков*

По второму кластеру при почти втрое большем свободном члене имеем угловой коэффициент регрессии заметно меньше 1. Это может означать, что во втором кластере белки

более длинные, но имеют меньший относительный прирост.

Для следующего организма *P* связь длин с длинами белка *S* по каждому из двух кластеров описывается уравнениями регрессии: P_1 = 1,05*S* + 44,35 и P_2 = 0,81S + 391,88. Гипотеза об адекватности этих уравнений не отвергается по критерию Фишера: F_1 = 223,75 при n_1 = 158 и F_2 = 32,08 при n_2 =37.

На рисунке 4 представлены графики этих уравнений и эмпирические точки, координаты которых равны длинам соответствующих белков аналогично рисунку 1.

По рисунку 4 видно, что изменение длин белка *P* подобно изменению длин белка *T*.

Изменение длин белков организма N по сравнению с длинами базисного организма имеет примерно ту же картину, что видно на рисунке.

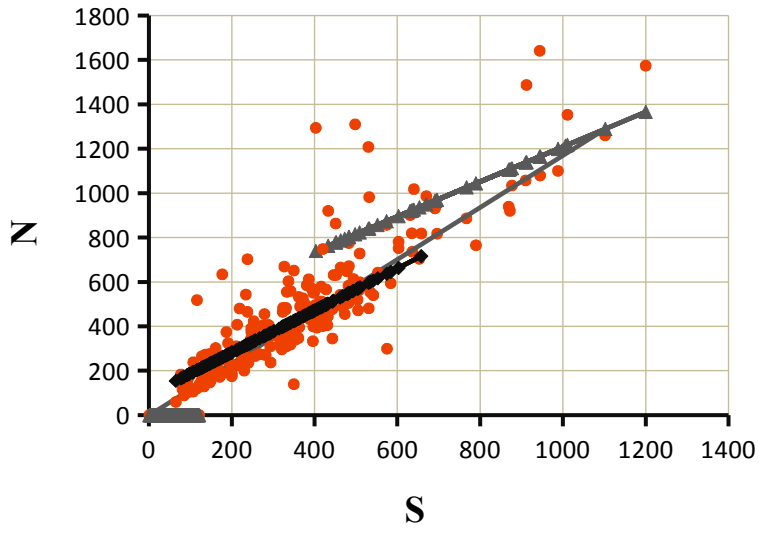


Рисунок 5. *Графики уравнений регрессии по кластерам длин белков N и S и точки с координатами, равными значениям длин этих белков*

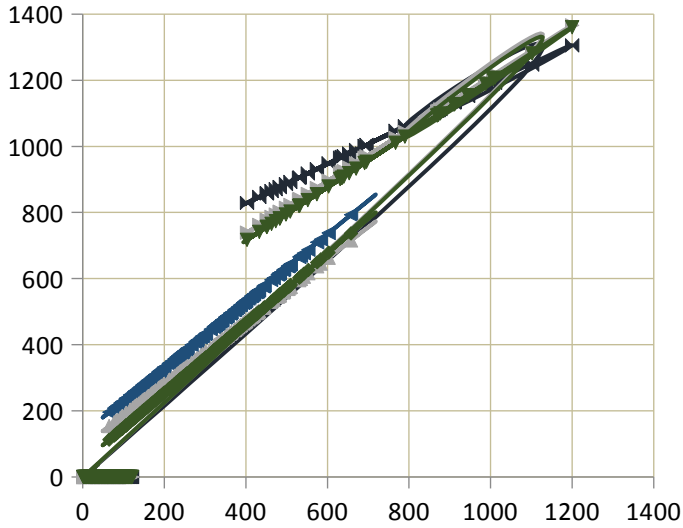


Рисунок 6. Графики регрессий, описывающих зависимости T(S), P(S), N(S) в одной системе координат

Для наглядности на рис. 6 приведены графики проведённых выше регрессий в одной системе координат.

На рисунке 6 показана биссектриса первого координатного угла, на которой находятся

точки с длиной белков базисного организма.

Чтобы показать результаты корреляционного анализа длин для пар ортологичных белков между собой приведем подробно регрессию длин рядов T и N по каждому из двух кластеров: T_1 = 0,98N + 64,45 и T_2 = 0,76N + 262,39. Гипотеза об адекватности этих уравнений не отвергается по критерию Фишера: F_1 = 54,27 при n_1 = 158 и F_2 = 10,16 при n_2 =37.

На рисунке 7 представлены графики этих уравнений и эмпирические точки, координаты которых равны длинам соответствующих белков.

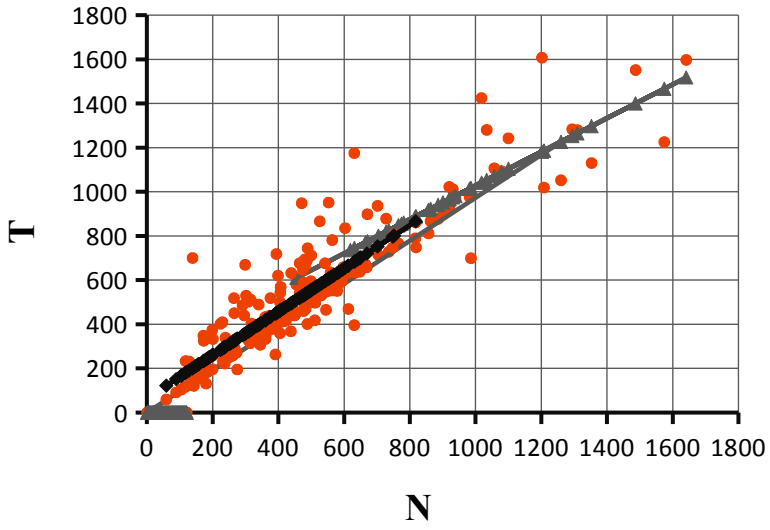


Рисунок 7. Графики уравнений регрессии рядов N и T и точки с координатами, равными значениям длин этих белков

Отметим, что между парами длин ортологичных белков характер связи несколько отличается от их связей с длинами базисного белка, например, отличается от предыдущих меньшим разрывом значений между кластерами.

Для таблицы A2 корреляционный анализ приращений длин ортологичных белков не обнаружил автокорреляции и значения парных коэффициентов корреляции не имеют ложной корреляции. Результаты анализа, как и уравнения регрессии оказались вполне соответствующими результатам, полученным в работе [1] при анализе приращений длин ортологичных белков взятых парами, причем множества таких пар для организмов *T* и *N* было гораздо больше. Поэтому эти результаты мы не приводим в данной работе.

Работа выполнена за счёт гранта Российского научного фонда (проект № 14–50–00150).

Литература

1. A.V. Seliverstov, O.A. Zverkov, S.N. Istomina, S.A. Pirogov, P.S. Kitsis, Comparative Analysis of Apicoplast-Targeted Protein Extension Lengths in Apicomplexan Parasites, BioMed Research International, 2015, Vol. 2015, Article ID 452958.

Горбунов К.Ю.[1], Любецкий В.А.[2]

[1]ИППИ РАН, г. Москва, к.ф.-м.н., в.н.с., gorbunov@iitp.ru

[2]ИППИ РАН, г. Москва, д.ф.-м.н., зав.лаб., lyubetsk@iitp.ru

РЕКОНСТРУКЦИЯ ПРЕДКОВЫХ ХРОМОСОМНЫХ СТРУКТУР

КЛЮЧЕВЫЕ СЛОВА

Хромосомная структура, эволюционное дерево, предковый вид, реконструкция вдоль дерева, эффективный алгоритм, булево линейное программирование.

АННОТАЦИЯ

Рассматривается задача реконструкции хромосомных структур в нелистовых вершинах данного филогенетического дерева на основе заданных структур в листьях. Структура может иметь паралоги – гены с одинаковыми именами. Рассматриваемая задача сводится к задаче булева линейного программирования, решаемой эффективным алгоритмом.

Определения и постановка задачи. Будем рассматривать реконструкцию хромосомных структур на основе модели хромосомной структуры как произвольного набора цепей и циклов, состоящих из ориентированных рёбер: генов, которым приписаны имена – натуральные числа *i*. На рис.1 изображены две структуры: *a* и *b*.

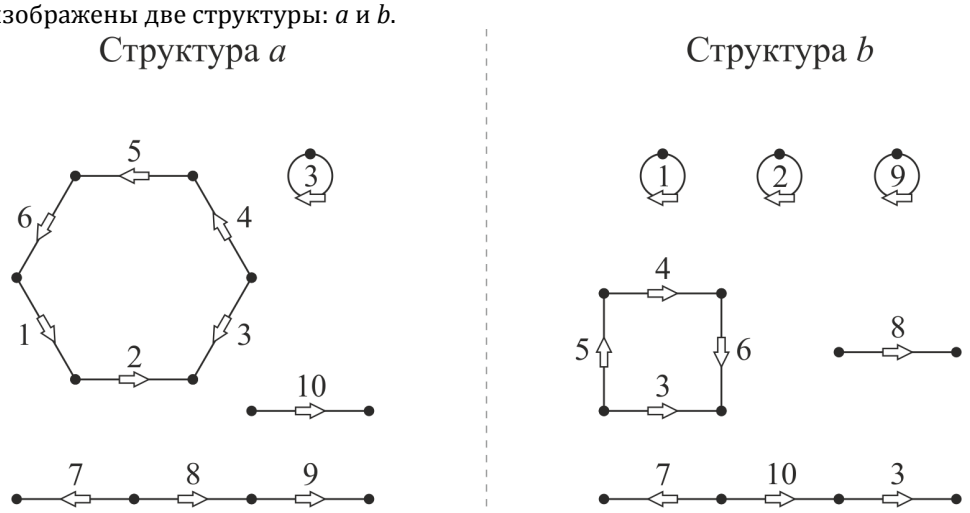


Рис.1. Две структуры a и b, каждая с двумя паралогами – генами, помеченными номером 3

Паралоги – гены из одной структуры, имеющие одинаковые номера *i*: для их идентификации применяется нумерация вида *i.j*, в которой число *i* назовём *первым номером* гена, а *j* – *вторым номером* (само выражение *i.j* будем называть *полным номером*; полные номера внутри одной хромосомной структуры не повторяются). В [1] мы рассматривали задачу реконструкции в предположении весьма существенного ограничения на модель – структуры должны иметь постоянный генный состав и не иметь паралогов. Здесь рассмотрим общий случай. Будем использовать понятие *брейкпоинтового расстояния* между двумя структурами. Для структур без паралогов (где есть лишь первые номера) оно определяется как число пар соответствующих краёв генов, соседних (или, как говорится в [1], *склеенных*) в одной структуре и не склеенных или отсутствующих в другой, сложенное с числом генов, присутствующих в одной структуре и отсутствующих в другой (под генами в разных структурах здесь понимаются гены с одинаковыми номерами).

Для структур с паралогами расстояние определяется как минимум расстояний между структурами, которые получаются установлением частичной биекции между паралогами каждого гена, представленного в обеих исходных структурах. Точнее, генов с первым номером *i* может быть разное число в структурах *a* и *b*, пусть это – соответственно множества $P_a(i)$ и $P_b(i)$; обозначим f_i

биекцию между частью множества $P_a(i)$ и частью множества $P_b(i)$; определяются *нумерации i.j* всех генов в $P_a(i)$ и $P_b(i)$ вторыми номерами, относительно которых f_i – тождественная функция; гены, не попавшие в области определения и значений f_i, имеют все разные вторые номера. Такая нумерация имеет естественную интерпретацию: если $y=f_i(x)$, то y «наследуется» от x; прочие гены из $P_a(i)$ и из $P_b(i)$ – самостоятельные, все различные между собой, поэтому они имеют различные значения j. Гены, не принадлежащие области определения или области значений f_i, можно считать *потерянными* и соответственно *возникшими* (на ребре, соединяющем a с b в филогенетическом дереве). Теперь *расстояние* между a и b определяется как минимум расстояний между a' и b', которые получаются из a и b при всевозможных нумерациях вторыми номерами всех генов в каждом множестве генов с одним и тем же первым номером (такую нумерацию будем называть добавлением вторых номеров). Например, для структур на рис.1 минимальное брейкпоинтовое расстояние достигается при нумерации в каждой структуре одного из паралогов гена 3 как 3.1 (неважно какого), а другого как 3.2. Оно равно 18.

Итак, дано корневое (возможно, небинарное) дерево, и в каждом листе своя хромосомная структура. Требуется расставить хромосомные структуры по нелистовым вершинам, чтобы сумма по всем рёбрам расстояний между структурами на концах ребра была минимальна. Покажем что это эквивалентно более «простому» требованию: расставить хромосомные структуры с нумерацией генов полными номерами по нелистовым вершинам и добавить вторые номера генам в листьях так, чтобы сумма по всем рёбрам расстояний между структурами на концах ребра (уже фактически без паралогов) была минимальна. Пусть найдена расстановка, минимизирующая описанную функцию F при первом требовании. Опишем расстановку, дающую не большее её значение для второго требования. Рассмотрим биекции f_i между генами на концах прикорневых рёбер, соответствующие определению функции F. Добавим генам в корне дерева вторые номера произвольным образом. Добавим вторые номера в некорневом конце каждого прикорневого ребра в соответствии с f_i. Таким образом, идя от корня, мы добавим вторые номера генам во всех вершинах дерева, включая листья. Поскольку сами биекции f_i не меняются, получаем расстановку для второго требования с тем же значением F. Обратное соответствие (от второго требования к первому) получается «стиранием» вторых номеров у всех генов. Таким образом, далее рассматриваем задачу в последней формулировке.

Сведение задачи реконструкции к задаче булевого линейного программирования. Легко видеть, что в искомой расстановке имеются лишь гены, полные номера которых присутствуют хотя бы в одном листе. Кроме того, нам удобно считать, что каждая (тождественная) биекция f_i, определяемая искомой расстановкой, определена на объединении M всех присутствующих в листьях номеров *i.j*, а отсутствующие в структуре гены специальным образом помечены. Очевидно, это определение эквивалентно исходному. «Началом» ребра в дереве считается та вершина, которая ближе к корню; аналогично определяется «конец» ребра.

Добавим произвольным образом генам в данных листовых структурах вторые номера; полученную нумерацию обозначим как I. Для перехода к задаче линейного булевого программирования введём переменные, линейные ограничения и линейную функцию, которую нужно минимизировать.

Введём переменные, указывающие соответствие между нумерацией I и искомой нумерацией генов в листьях. Для каждого листа e дерева и каждой упорядоченной пары *k.i*, *k.j* элементов из M введём переменную z_{kije} (первый индекс будем для краткости опускать); эта переменная равна 1, если ген *k.i* в e имеет в искомой нумерации номер *k.j*, иначе $z_{kije}=0$. Соответствие между генами с первым номером k, определённое переменными z_{ije}, должно быть биективным, что выражается линейными равенствами: для фиксированного i и e сумма по j значений z_{ije} равна 1, и аналогично для фиксированных j и e.

Введём переменные, определяющие хромосомные структуры в вершинах.

Для каждой вершины v и каждой неупорядоченной пары k, l различных краёв генов введём переменную x_{klv}; эта переменная равна 1 если эти края склеены в вершине v, иначе она равна 0. Каждый край может быть склеен не более чем с одним краем, что выражается линейными неравенствами: при фиксированных k и v сумма по l значений x_{klv} не превышает 1, и аналогично для фиксированных l и v. В листьях значения переменных x_{kl} заданы.

Для каждой вершины v и каждого гена k введём переменную y_{kv}; эта переменная равна 1 если ген k отсутствует в v; иначе она равна 0. Края отсутствующих генов ни с чем не склеены, что выражается линейными неравенствами $x_{ijv} \le 1 - y_{kv}$, где i или j – край гена k. Для каждого гена k и

каждого его края i вместо этих неравенств можно использовать одно неравенство $\sum_{j \neq i} x_{ijv} \leq 1 - y_{kv}$, где j пробегает все края генов в v. В листьях значения переменных y_k заданы.

Введём переменные, вычисляющие брейкпоинтовое расстояние между структурами, приписанными началу и концу ребра дерева. Пусть края k и l генов лежат в конце ребра e, а края m, n – в его начале. Назовём четвёрку k, l, m, n краёв генов *брейкпойнтовой*, если пара возможных склеек – склейка краёв k и l и склейка краёв m и n, – учитывается при определении брейкпойнтового расстояния на нелистовом e или может учитываться на листовом e после какого-нибудь (возможно, отличного от I) добавления вторых номеров генам в листьях (эта четвёрка – упорядоченная пара двух неупорядоченных пар: (k,l) и (m,n)).

Для каждого ребра e и каждой брейкпойнтовой четвёрки k, l, m, n краёв введём переменную s_{klmne}; эта переменная равна 1, если эти края принадлежат соответствующим генам (напомним: на листовом e соответствие определяется переменными типа z) и склеены в начале ребра и не склеены (или хотя бы один ген отсутствует) в конце ребра или наоборот. Это условие для s_{klmne} выражается линейными неравенствами; приведём их ниже для случая, когда e листовое, все четыре края – начала генов (или все – концы) и все четыре гена имеют одинаковые первые номера (другие случаи аналогичны): если e=(u,v), гены i_1 и j_1 имеют, соответственно, края k и m, а гены i_2 и j_2 – края l и n, то

$s_{klmne} \geq x_{klv} - x_{mnu} - (1 - z_{i1j1e}) - (1 - z_{i2j2e})$, $s_{klmne} \geq x_{mnu} - x_{klv} - (1 - z_{i1j1e}) - (1 - z_{i2j2e})$,

$s_{klmne} \geq x_{klv} - x_{mnu} - (1 - z_{i1j2e}) - (1 - z_{i2j1e})$, $s_{klmne} \geq x_{mnu} - x_{klv} - (1 - z_{i1j2e}) - (1 - z_{i2j1e})$.

Если правая часть неравенства равна 0, переменная s_{klmne} принимает значение 0 за счёт того, что она входит в качестве слагаемого в минимизируемую функцию.

Для каждого ребра e=(u,v) и каждых двух генов i и j, лежащих, соответственно, в конце и в начале ребра e, введём переменную s_{ije}; эта переменная равна 1, если эти гены соответствуют друг другу и один из них присутствует в структуре, а другой отсутствует. Это условие выражается (для листового e) линейными неравенствами $s_{ije} \geq y_{iv} - y_{ju} - (1 - z_{ije})$ и $s_{ije} \geq y_{ju} - y_{iv} - (1 - z_{ije})$.

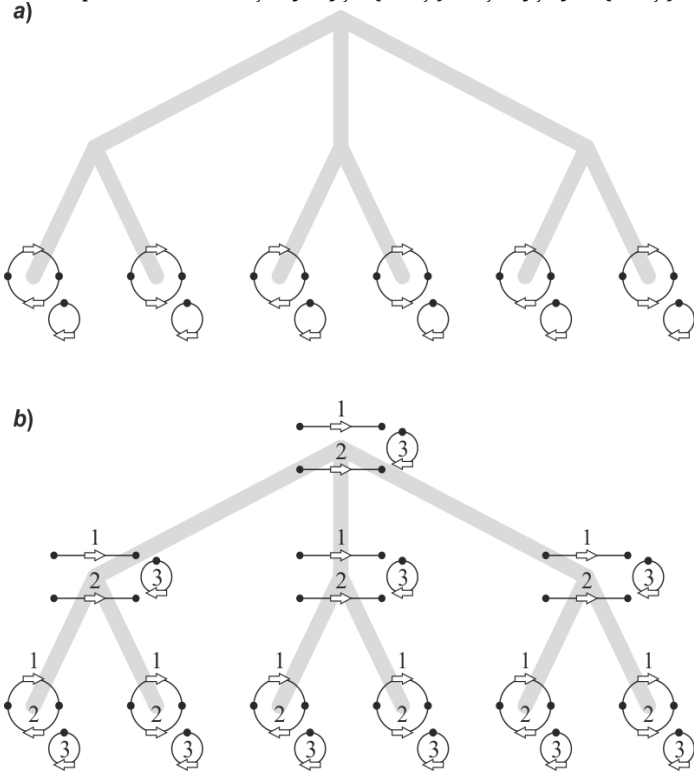


Рис.2. Пример реконструкции для искусственных данных. a) В каждом листе дерева дана хромосомная структура с тремя паралогами одного и того же гена. b) Реконструированные структуры, построенные алгоритмом в нелистовых вершинах, и добавленные вторые номера генов в листьях

Минимизируемая функция – сумма всех переменных s_{klmne} и s_{ije}, линейные ограничения описаны выше. Решение описанной задачи линейного булева программирования даёт искомую расстановку хромосомных структур по нелистовым вершинам дерева и вторые номера генам в

листьях.

Замечание 1. Описанное сведение задачи реконструкции хромосомных структур к задаче булевого линейного программирования легко обобщается на случай, когда стоимости брейкпоинтовых операций (переход от двух склеенных краёв к расклеенным или наоборот, и переход от присутствия гена к отсутствию или наоборот) различны. Для этого каждую переменную s_{klmne} следует заменить на две переменных $s1_{klmne}$ и $s2_{klmne}$. Для первой из них действуют линейные ограничения вида $s1_{klmne} \geq x_{klv} - x_{mnu} - \ldots$; для второй – ограничения вида $s2_{klmne} \geq x_{mnu} - x_{klv} - \ldots$. Аналогично, переменную s_{ije} следует заменить на $s1_{ije}$ и $s2_{ije}$. Теперь можно в минимизируемой функции умножать новые переменные на коэффициенты, отражающие веса событий.

Замечание 2. Задачу вычисления брейкпойнтового расстояния между двумя данными хромосомными структурами с паралогами можно рассматривать как частный случай задачи реконструкции. Здесь "дерево" состоит из одного ребра с данными структурами на концах. Очевидно, остаются лишь переменные типа z и s.

На рис.2*b* приведено решение задачи реконструкции хромосомных структур для искусственных данных, показанных на рис.2*a*. Все гены в данных листовых структурах имеют один и тот же первый номер 1 (не показан), в реконструированных структурах показаны лишь вторые номера.

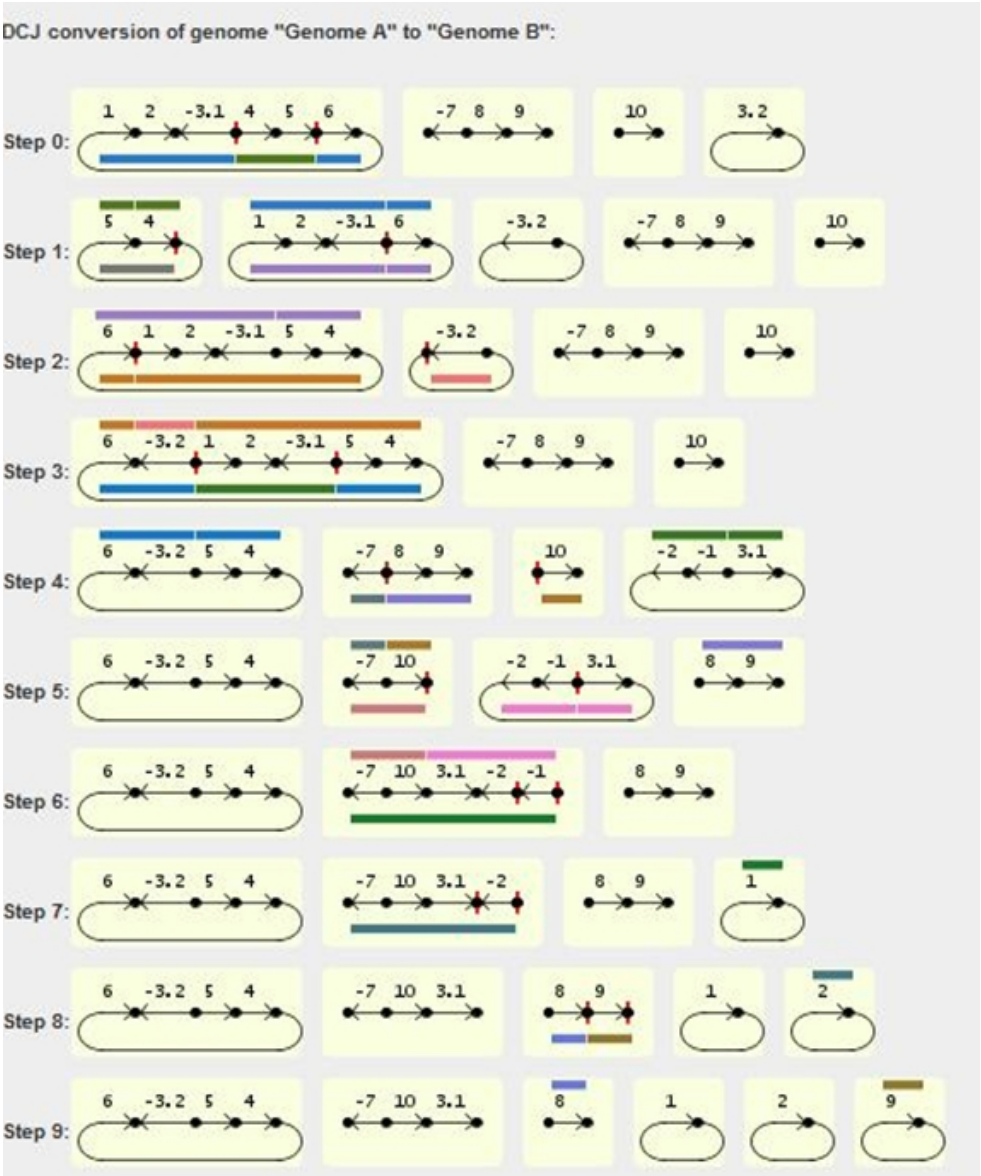


Рис.3. Кратчайшая последовательность операций, переводящая структуру a в структуру b (рис.1) для случая, когда верхний ген 3 в структуре a соответствует нижнему гену 3 в структуре b.

Обобщение понятия расстояния. Пусть, для простоты, даны две структуры с одинаковым набором генов. Брейкпоинтовое расстояние между ними можно рассматривать как минимальное число операций расклейки двух склеенных краёв генов или склейки двух свободных краёв, необходимых для перевода одной структуры в другую. Добавим к этим операциям следующие *операции* над хромосомной структурой: *двойная переклейка* – расклейка двух склеек краёв генов и переклейка четырёх краёв, приводящая к новой структуре и *полуторная переклейка* – расклейка двух склеенных краёв и склеивание одного края с каким-то несклеенным краем, второй край остаётся свободным. Получим обобщение брейкпойнтового расстояния, называемое *биологическим расстоянием*. Для случая двух хромосомных структур с одинаковым набором генов (без паралогов) реализация алгоритма вычисления биологического расстояния описана в [3]. Пример с двумя структурами на рис.1 показывает, что биологическое расстояние "чувствительнее" брейкпоинтового: теперь два возможных соответствия между паралогами гена 3 дают различные значения расстояния. Если верхний ген в структуре *a* соответствует нижнему гену в структуре *b*, перевести *a* в *b* можно самое меньшее за девять операций – рис.3; при альтернативном соответствии достаточно семи операций – рис.4 (рисунки получены программой, описанной в [3]).

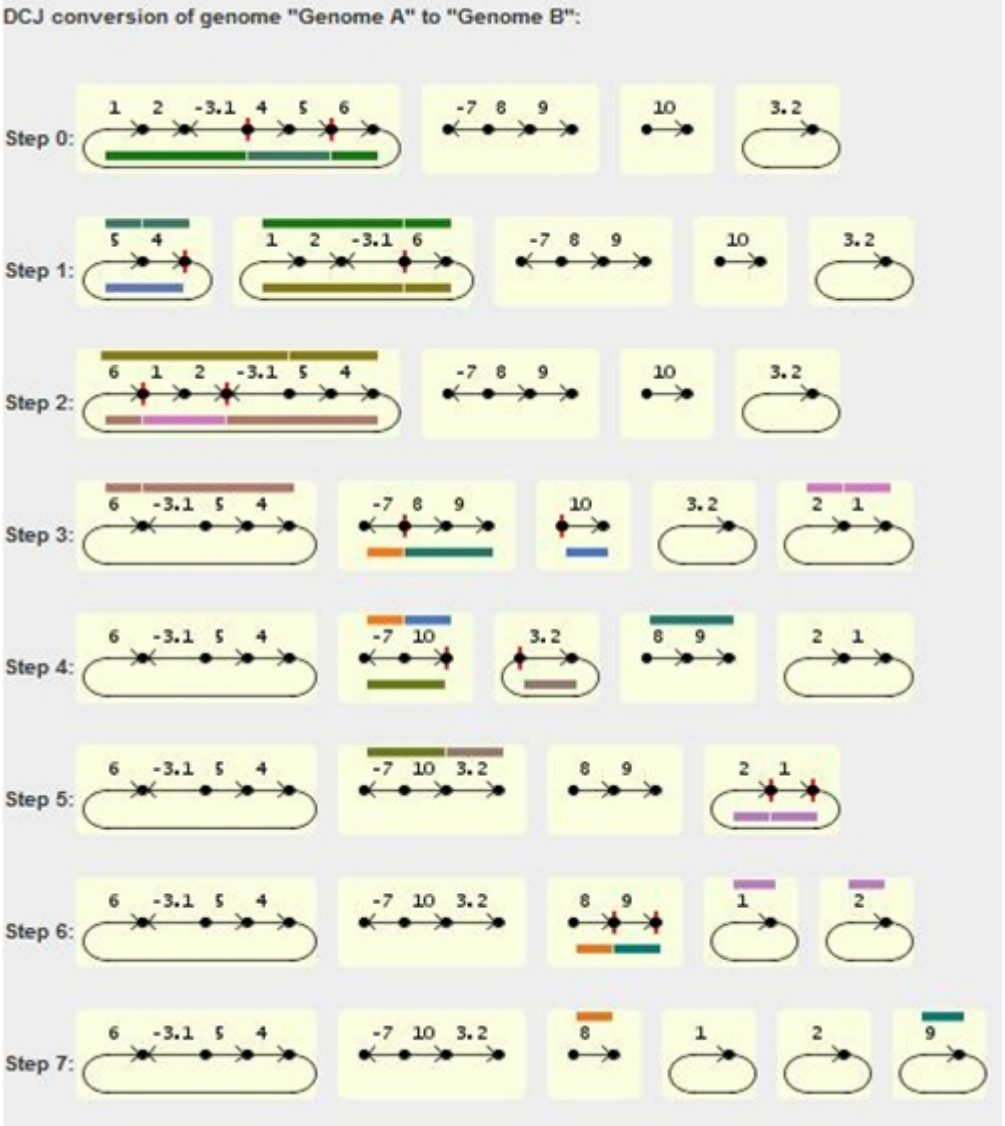


Рис.4. Кратчайшая последовательность операций, переводящая структуру a в структуру b (рис.1) для случая, когда верхний ген 3 в структуре a соответствует верхнему гену 3 в структуре b

Для биологического расстояния задачи его вычисления (в том числе для структур с различным набором генов и с паралогами) и соответствущей реконструкции хромосомных структур рассматриваются в [1, 2], а также в нескольких готовящихся к публикации работах

авторов. Реализации соответствующих алгоритмов доступны на сайте http://lab6.iitp.ru/ru/chromo/.

Работа выполнена за счёт гранта Российского научного фонда (проект № 14–50–00150).

Литература

1. Горбунов К.Ю., Гершгорин Р.А., Любецкий В.А. Перестройка и реконструкция хромосомных структур // *Молекулярная биология*, 2015, том 49, № 3, стр. 372–383.
2. Gershgorin R.A., Gorbunov K.Yu., Seliverstov A.V., Lyubetsky V.A. Evolution of Chromosome Structures // "Information Technology and Systems 2015" An IITP RAS Interdisciplinary Conference & School (ITaS'15), Sochi, Russia, Sep 7–11 2015, pp. 105–120.
3. Hilker R., Sickinger C., Pedersen C., and Stoye J. UniMoG – a unifying framework for genomic distance calculation and sorting based on DCJ // *Bioinformatics*, 2012, vol. 28, pp. 2509–2511.

Королев С.А.[1], Селиверстов А.В.[2], Любецкий В.А.[3]

[1]ИППИ РАН, г. Москва, стажер-исследователь, korolev@iitp.ru

[2]ИППИ РАН, г. Москва,к.ф.-м.н., в.н.с., slvstv@iitp.ru

[3]ИППИ РАН, г. Москва, д.ф.-м.н., зав.лаб., lyubetsk@iitp.ru

О ТРАНСЛЯЦИИ РИБОСОМНОГО БЕЛКА L16 В ПЛАСТИДАХ ЦВЕТКОВЫХ РАСТЕНИЙ

КЛЮЧЕВЫЕ СЛОВА

Цветковые растения, трансляция, рибосомный белок L16, регуляция.

АННОТАЦИЯ

Предсказана регуляция инициации трансляции рибосомного белка L16 в пластидах цветковых растений. Показана консервативная спираль РНК в 5'-нетранслируемой области, перекрывающая сайт связывания рибосомы.

Проведён анализ 5'-лидерных областей перед генами *rpl16* с интронами. Предсказана консервативная спираль РНК в 5'-лидерных областях генов *rpl16* с интронами. Эта спираль перекрывает сайт связывания рибосомы и препятствует инициации трансляции до завершения сплайсинга. Этот ген имеет короткий первый экзон (обычно его длина всего 9 п.н.). Хотя у магнолии интроны в этом гене отсутствуют.

Рассматриваемая спираль РНК обнаружена у многих цветковых растений, включая рано отделившиеся ветви дерева видов: *Amborella trichopoda* (амборелла волосистоножковая), *Chloranthus* spicatus (зеленоцвет или хлорант колосковый), *Illicium oligandrum* (бадьян), *Nuphar advena* (кубышка), *Nymphaea alba* (кувшинка) и *Ceratophyllum demersum* (роголистник). Также она предсказана у *Acorus* spp. и многих видов двудольных растений из группы rosids. (Розиды – это подкласс клады покрытосеменных цветковых растений). Однако она отсутствует или лишь спорадически встречается в больших таксономических группах, включая филогенетическую группу asterids.

В 5'-нетранслируемых областях перед *rpl16* у всех рано отделившихся ветвей цветковых растений (basal Magnoliophyta) найдена шпилька, перекрывающая сайт связывания рибосомы. Наиболее стабильные структуры РНК предсказаны у *Illicium oligandrum* и *Chloranthus spicatus*. У *Nuphar advena* и *Nymphaea alba* эти шпильки отличаются одним нуклеотидом, но это приводит к образованию во втором случае большей петли. У *Ceratophyllum demersum* шпилька относительно короткая. У *Amborella trichopoda* шпилька отделена от кодирующей области большим расстоянием, чем у других видов. Эти структуры показаны на рисунке.

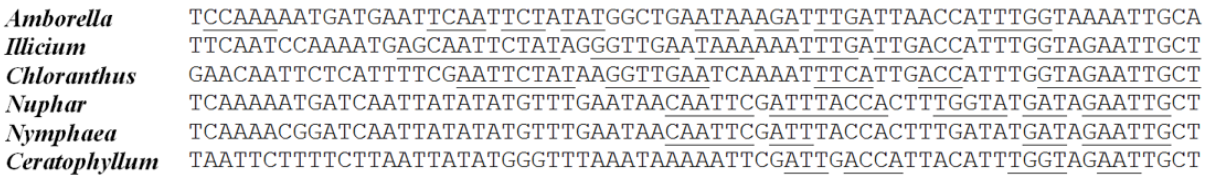


Рис. В 5'-нетранслируемых областях перед rpl16 у рано отделившихся ветвей цветковых растений подчёркнуты спирали РНК

Отметим, что ранее [1] нами предсказана регуляция инициации трансляции у трёх генов с интронами: *atpF*, *clpP* и *petB*. Такая регуляция для других генов также обсуждается в [2]. Рассматриваемый случай гена *rpl16* особенно интересен тем, что рибосома сразу после инициации трансляции уже перекрывает край первого экзона, препятствуя сплайсингу. Поэтому в этом случае задержка начала трансляции второго экзона до завершения сплайсинга не может быть объяснена различием в скорости элонгации РНК-полимеразы фагового типа и скорости элонгации рибосомы, хотя для некоторых других генов это может дать исчерпывающее объяснение [3].

Работа выполнена за счёт гранта Российского научного фонда (проект № 14–50–00150).

Литература

1. Seliverstov A.V., Lyubetsky V.A. Translation regulation of intron-containing genes in chloroplasts // *Journal of Bioinformatics and Computational Biology*, 2006, vol. 4, no. 4, pp. 783–792.
2. Zerges W. Translation in chloroplasts // *Biochimie*, 2000, vol. 82, pp. 583–601.
3. Lyubetsky V.A., Zverkov O.A., Rubanov L.I., Seliverstov A.V. Modeling RNA polymerase competition: the effect of σ-subunit knockout and heat shock on gene transcription level // *Biology Direct*, 2011, vol. 6, no. 3.

Фролов А.С.[1], Семенов А.С.[2]

[1] АО «Научно-исследовательский центр электронной вычислительной техники», начальник отдела, frolov@nicevt.ru

[2] АО «Научно-исследовательский центр электронной вычислительной техники», начальник сектора, semenov@nicevt.ru

ИСПОЛЬЗОВАНИЕ CHARM++ ДЛЯ РЕШЕНИЯ ГРАФОВЫХ ЗАДАЧ В МАСШТАБАХ ЭКЗАФЛОПСНЫХ ВЫЧИСЛЕНИЙ

КЛЮЧЕВЫЕ СЛОВА

Обработка графов, языки параллельного программирования, программные модели, суперкомпьютеры, экзафлопс.

АННОТАЦИЯ

В статье рассматривается проблема создания программной модели для экзафлопсных вычислительных систем, ориентированной на параллельную обработку больших графов. Предлагается в качестве основы программной модели использование модели параллельного языка Charm++ с специальными расширениями, ориентированными с одной стороны на поддержку архитектурных особенностей систем экзафлопсного класса, с другой стороны поддержка нерегулярных приложений с мелко-зернистым параллелизмом, характерным в том числе для задач обработки больших графов. Приводятся результаты экспериментов по введению многоуровневого параллелизма в программную модели Charm++ в виде библиотеки uChareLib. Результаты экспериментов получены на вычислительном кластере МВС10П на базе коммерческой коммуникационной сети Infiniband FDR и на 36-узловом кластере с отечественной коммуникационной сетью «Ангара».

Введение

Одной из ключевых проблем создания суперкомпьютерных комплексов экзафлопсной производительности является вопрос разработки программных моделей и инструментальных средств программирования, позволяющих эффективно решать прикладные задачи в масштабах систем, состоящих из десятков тысяч вычислительных узлов, обладающих гибридной архитектурой, объединяющей разнородные вычислители (процессоры, ускорители на СБИС и ПЛИС) посредством единой высокоскоростной кэш-коггерентной сети, многоуровневой иерархической памятью, объединенных высокоскоростной коммуникационной сетью. [1,2]

Таким образом, достижение экзафлопсной производительности [с точки зрения архитектуры] сопряжено не только с экстенсивным наращиванием количества ядер и узлов вычислительной системы (относительно петафлопсного уровня количество ядер в системе будет x1000), но и с большим разнообразием (гибридностью) внутри вычислительного узла, более глубокой взаимной интеграцией вычислительных ядер друг с другим и с устройствами ввода-вывода (в том числе с сетевым адаптером), увеличением количества уровней иерархии памяти. Все эти изменения должны найти адекватную поддержку со стороны программных моделей и инструментальных средств программирования, используемых для экзафлопсных систем.

Развитие программных моделей (т. е. языков программирования) в значительной степени более инертно, чем аппаратных средств, что объясняется необходимостью поддерживать коды, написанные с использованием традиционных средств параллельного программирования, таких как MPI, OpenMP, при чем развитие самих стандартов зачастую не отражает реальной ситуации с точки зрения их использования: коммуникационные операции MPI 1.0 по прежнему являются наиболее используемой практикой обменов данными между параллельными процессами. Такое явление характерно для прикладных областей, где имеется большой задел по количеству [legacy] прикладных задач и стоимость переписывания их с использованием новых программных моделей превышает возможный выигрыш для конечных пользователей.

Для относительно новых прикладных областей, таких как параллельная обработка больших графов, ситуация принципиально иная. Отсутствие проблемы поддержки legacy-кодов,

позволяют использовать наиболее эффективные программные модели как с точки зрения производительности, так и с точки зрения продуктивности. Поэтому можно отметить, что именно в данных областях наиболее вероятно внедрение новых программных моделей

Кроме того, параллельная обработка графов характеризуется: во-первых, нерегулярной структурой обрабатываемых данных, как следствие интенсивной работой с памятью по непредсказуемым адресам и преимущественно каждое обращение считывает или записывает одно 64-разрядное слово, что не позволяет эффективно использовать пропускную способность памяти и кэши-данных; во-вторых, интенсивными обменами короткими сообщениями по коммуникационной сети с характерным шаблоном "все-всем"; в-третьих, преобладанием целочисленных адресных вычислений и операций обращения к памяти над арифметическими операциями с плавающей точкой; в-четвертых, неравномерной нагрузкой на вычислительные узлы.

Таким образом, для задач обработки больших графов на суперкомпьютерных комплексах экзафлопсной производительности необходимо создание новых альтернативных моделей вычислений отражающих как специфику графовых задач, так и архитектурные особенности экзафлопсных систем.

Данная работа посвящена созданию новой программной модели на основе языка программирования Charm++ [3] с управлением потоком сообщений, ориентированной на параллельную обработку графов в масштабах экзафлопсных систем. Далее приводится краткое описание Charm++ и предлагаемых расширений к модели программирования Charm++; приводятся результаты экспериментов по введению многоуровневого параллелизма в программную модели Charm++ в виде библиотеки *uChareLib*. В заключении приводится обобщение представленного материала и экспериментально полученных результатов, формулируются задачи дальнейшего исследования.

Стандартная и расширенная программные модели CHARM++

Язык параллельного программирования Charm++ является расширением языка C++ и основан на объектно-ориентированной модели с управлением асинхронным потоком сообщений [3].

Базовым объектом в Charm++ является *chare* (или *chare-объект*), обладающий помимо всех свойств объектов в C++ (инкапсулированными данными, множеством публичных и приватных методов и т. п.), интерфейсом из специальных *entry-методов*, вызовы которых соответствуют передаче данных (т. е. сообщений) между chare-объектами.

Приложение в Charm++ состоит из множества chare-объектов, обменивающихся между собой данными посредством вызовов entry-методов. Кроме того, вызов entry-метода порождает выполнение непосредственно самого метода на том узле, где находится chare-объект, entry-метод которого был вызван, то есть таким образом реализуется концепция *управления потоком данных* или *активных сообщений*.

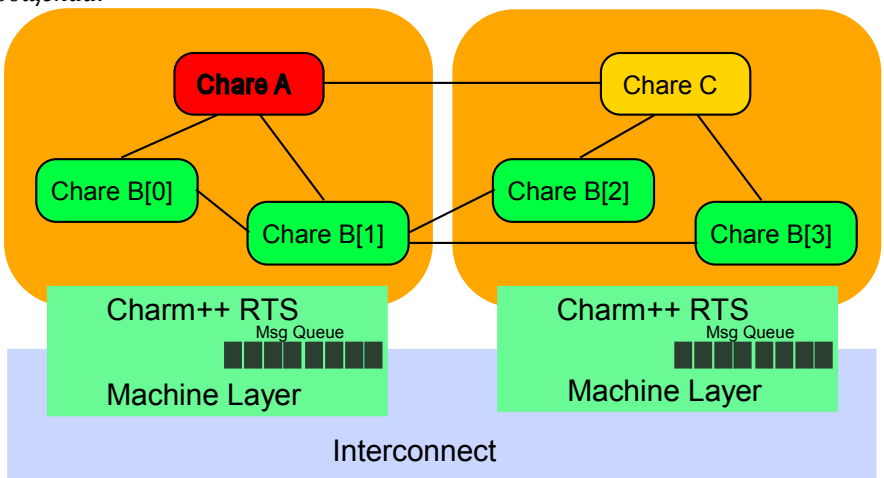


Рис.1. Классическая программная модель Charm++

В процессе выполнения entry-метода могут быть изменены только данные, принадлежащие chare-объекту. Одновременно у chare-объекта может выполняться не более одного entry-метода, что позволяет избежать проблемы обеспечения атомарности изменения

данных, принадлежащих chare-объекту, и значительно упрощает программирование. Схематически программная модель Charm++ представлена на рисунке 1.

Кроме простых chare-объектов Charm++ имеется возможность создавать массивы chare-объектов. При этом массивы бывают как одномерными, так и многомерными. Использование массивов позволяет создавать большое количество chare-объектов, управлять их распределением по вычислительным узлам, выполнять над ними коллективные операции.

Кроме того, в Charm++ поддерживается автоматическая динамическая балансировка нагрузки на вычислительные узлы, что реализуется на уровне runtime-системы за счет перемещения chare-объектов между вычислительными узлами. Для прикладного программиста данная возможность совершенна прозрачна, так как программная модель Charm++ не специфицирует расположение chare-объектов в вычислительной системе, и все объекты адресуются в едином адресном пространстве.

Основными причинами выбора Charm++ как основы для создания новой программной модели, ориентированной на обработку больших графов в масштабах экзафлопсных вычислительных систем, являлись: во-первых, асинхронная вычислительная модель Charm++ с управлением потоком данных; во-вторых, открытый исходный код; в-третьих, развитая инфраструктура Charm++ (наличие средств отладки и профилирования параллельных программ); в-третьих, активное сообщество разработчиков и пользователей.

Основные концепции модели Charm++ были разработаны почти 20 лет назад в то время, когда массово-параллельные системы (вычислительные кластеры) на основе коммерческих микропроцессоров только набирали популярность. Тем не менее, модель Charm++, вобравшая в себя как принципы dataflow-моделей, так и многопоточных (мультитредовых) моделей вычислений, оказалась достаточно успешной и с момента создания почти не претерпевала изменений. Однако, за этот период архитектура вычислительных систем существенна изменилась, а с появлением экзафлосных суперкомпьютеров разница архитектур станет еще значительней.

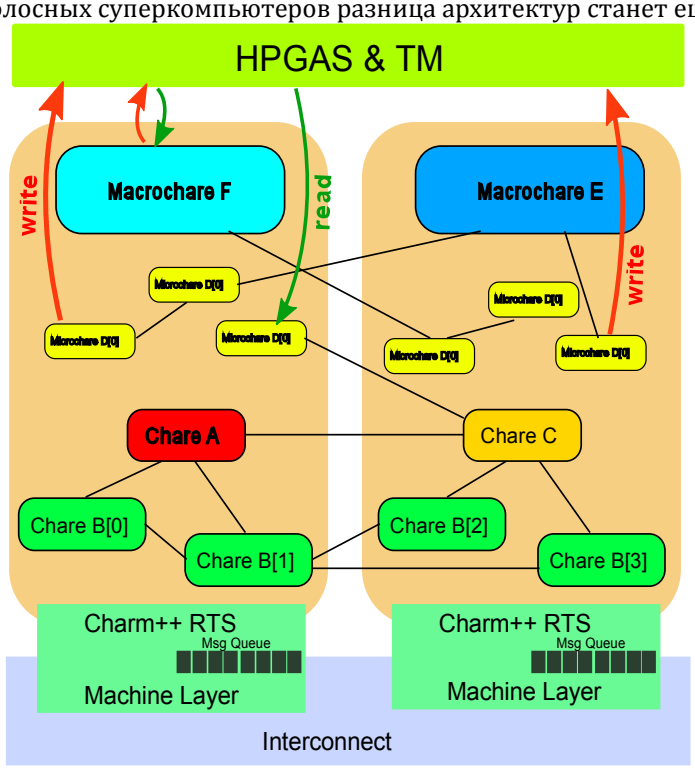


Рис.2. Расширенная программная модель Charm++

Таким образом, сохранение основных принципов программной модели Charm++ (управление потоком сообщений, асинхронность, инкапсулированность данных), введение расширений, отражающих новые архитектурные возможности суперкомпьютеров, а также соответствие классу задач обработки больших графов – суть методологии создания новой программой модели, описываемой в данной статье. Схематически предлагаемая расширенная программная модель Charm++ приведена на рисунке 2.

Ниже приведены описания предлагаемых расширений к Charm++.

Классификация chare-объектов по целевой архитектуре. Для соответствия гибридной архитектуре вычислительных узлов экзафлосных вычислительных систем, включающей в себя вычислительная ядра разного типа (CPU, GPU, FPGA), предлагается расширить тип chare-объекта Charm++ соответствующим спецификатором. Для реализации предлагается ввести сооветствующие ключевые слова в язык интерфейса Charm++, используемый в .ci файлах. Система поддержки выполнения Charm++ программ (runtime-система) будет автоматически распределять chare-объекты по соответсвующим вычислительным ядрам узла суперкомпьютера. Некоторые аспекты совместного использования Charm++ и CUDA для программирования оснащенных GPU кластеров представлены в [4].

Классификация chare-объектов по граннулярности параллелизма. Также предлагается разделять chare-объекты по степени гранулярности параллелизма (т. е. объему данных, соддержащихся внутри объекта, и количеству команд процессора, требуемых для выполнения entry-методов) Предлагается ввести три типа объектов: *микро*, *нормальный* и *макро*. В соответствии с названиями микро-chare-объекты будут соответствовать самым малым вычислительным единицам параллельного алгоритма (таких объектов на одном ядре может выполняться до десятков тысяч), нормальные chare-объекты соответствуют обычным Charm++ объектам, макро-chare-объекты будут соответствовать наиболее "тяжелым" вычислениям, требующим большого количества данных, и включать entry-методы, состоящие из большого числа инструкций (например, содержать вложенные циклы и т.п.). Такая специализация позволит runtime-системе оптимизировать хранение chare-объектов и обеспечить наиболее эффективное планирование выполнения entry-методов. Для реализации предлагается ввести сооветствующие ключевые слова в язык интерфейса Charm++, используемый в .ci файлах.

Динамические домены синхронизации. Использование динамических доменов синхронизации позволит повысить эффективность и масштабируемость приложений, в которых предполагается выполнение какой-либо синхронизации (барьера, определения состояния завершения всех entry-методов и т.п.) по динамически определяемому подмножеству chare-объектов. Данная операция требуется во многих графовых алгоритмах и их асинхронных реализациях (например, в алгоритме Борувки поиска минимального остовного дерева графа).

Многоуровневая глобально адресуемая распределенная общая память (HPGAS). Изначально программная модель Charm++ не предполагает использования общей памяти как таковой. Все chare-объекты работают только с локальными данными, что позволяет перемещать chare-объекты между вычислительными узлами. Однако для некоторых алгоритмов введение общей памяти позволяет упростить реализацию и повысить эффективность. Предлагаемые уровни иерархии должны включать: уровень кэш-памяти (L1-L3 кэши), уровень сверхбыстрой памяти (HBM-памяти), уровень локальной памяти (DDR4), уровень локальной NVRAM, уровень удаленной памяти, доступной через, RDMA операции. Данные должны размещаться в подсистеме памяти в соответствии с указанным уровнем, что позволит обеспечить максимальную гибкость в управлении доступом к памяти. Некоторые аспекты совместимости программных моделей Charm++ и PGAS рассмотрены [5,6].

Поддержка проблемно-ориентированных языков программирования. Расширение возможностей программной модели Charm++ может привести к усложнению программирования. Данная проблема может быть решена за счет использования проблемно-ориентированных языков программирования (DSL). Для работы с графами может быть использован язык Green-Marl [7] или другие языки для работы с графами (в том числе языки запросов к графовым базам данных).

Представленный перечень расширения программной модели Charm++ не является окончательным, и его уточнения будет проводиться в ходе дальнейших исследований.

Экспериментальные результаты

Для демонстрации потенциала расширенной программной модели Charm++ рассмотрим введение микро-chare-объектов (или uchare-объектов) дополнительно к обычным chare-объектам, определенным в Charm++ .

Для реализации uchare-объектов была разработана библиотека *uChareLib* и в язык интерфейса Charm++ (язык описания Charm++ конструкций в .ci файлах) было введено соответствующее ключевое слово – uchare. Пример описания массива из uchare-объектов представлен на рисунке 3.

```
mainmodule test_hello {
 extern module uChareLib;
 readonly CProxy_Main mainProxy;

 mainchare Main {
  entry Main(CkArgMsg *m);
  entry [reductiontarget] void start();
  entry void done();
 };

 uchare array [1D] Hello {
  entry void hello( int callee);
  entry void buybuy( int callee);
 };
 array [1D] uChareSet<Hello, CProxy_Hello,
    CBase_Hello>;
};
```

Рис.3. Пример .ci файла с массивом uchare-объектов

Использование uchare-объектов позволяет: во-первых, агрегировать объекты и хранить их более оптимально, чем обычные chare-объекты; во-вторых, автоматически агрегировать короткие сообщения, которые возникают при вызове entry-методов у uchare-объектов.

Для оценки проивзодительности были выбраны два оценочных теста: HPCC RandomAccess [8] и Asynchronous Breadth-First Search (ABFS). Тест HPCC RandomAccess реализует псевдослучайный доступ к большому массиву данных, единица измерения производительности GUP/s (Giga Updates Per Second). Тест ABFS реализует поиск достижимых вершин в графе от заданной вершины (корня), в отличие от стандартного BFS, используемого в Graph500 [9], в асинхронном BFS отсутствует глобальная синхронизация после обхода каждого уровня (т. е. решается задача только поиска достижимых вершин, без построения дерева предков). Единица измерения производительности GTE/s (Giga Traversed Edges Per Second).

Проводилось сравнение трех реализации: классической Charm++ реализации, реализаций, использующих библиотеку агрегации коротких сообщений TRAM [10] и uChareLib.

В качестве тестовых платформ использовались: суперкомпьютер МВС10П, установленный в МСЦ РАН, и 36-узловой прототипный кластер с отечественной высокоскоростной коммуникационной сетью «Ангара» [11, 12], установленный в АО «НИЦЭВТ».

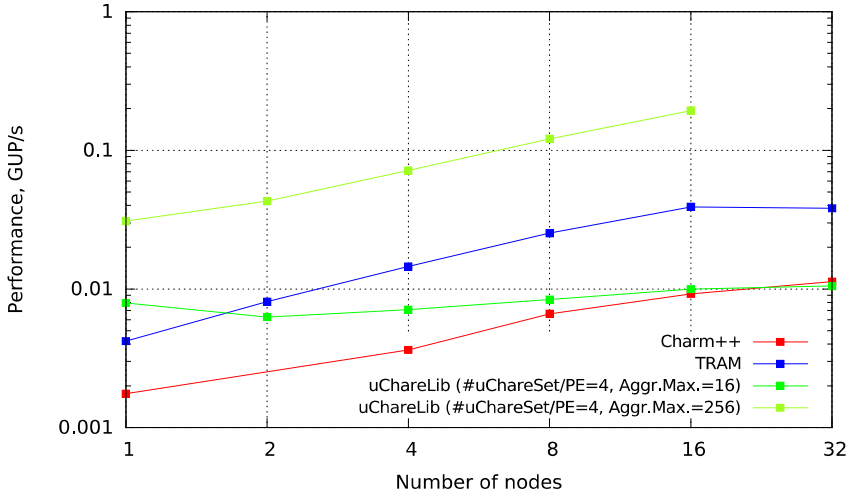


Рис.4. RandomAccess на МВС10П: количество chare, uchare-объктов/PE – 16384

На рисунке 4 приведены результаты запуска теста RandomAccess на вычислительном комплексе МВС10П. На рисунке представлены зависимость производительности (GUP/s) от числа узлов. Количество используемых ядер – 16. Размер в таблицы – 2^{20} элементов (элемент – 64-разрядное слово) на одно ядро, то есть на один узел размер таблицы составляет 16 x 2^{20}. С увеличением числа узлов таблица увеличивается пропорционально (т. е. тест исследуется в режиме weak scaling). Количество используемых chare-объектов (для Charm++ и TRAM) и uchare-объектов (для uChareLib) – 16384, то есть имитируется приложение с мелкозернистым

параллелизмом.

Как видно из рисунка, наименьшую производительность показала реализация теста RandomAccess на классической программной модели Charm++. Использование библиотеки агрегаций сообщений TRAM позволили увеличить производительность в 10 раз относительно Charm++. Вместе с тем результаты uChareLib превосходят TRAM также в 10 раз. При этом, важным параметром является размер агрегируемого сообщения (на рисунке Aggr.Max). Уменьшение этого параметра с 256 коротких сообщений в одном агрегированном до 16 приводит к падению производительности uChareLib-реализации до уровня Charm++. Для библиотеки TRAM размер буфер для агрегируемых сообщений равен 1024.

На рисунке 5 приведены результаты запуска теста ABFS на вычислительном комплексе МВС10П. На рисунке представлены зависимость производительности (MTE/S) от числа узлов. Количество используемых ядер – 16. Размер графа – 2^{16}, средняя степень вершин – 16. Граф генерируется с равномерно случайным распределением дуг между вершинами.

Как видно из рисунка, производительность реализаций ABFS на Charm++ и TRAM одинакова. Использование uChareLib дает резкий прирост производительности в 44 раза. Также как и в случае с RandomAccess производительность uChareLib крайне чувствительна к параметрам: Aggr.Max – максимальный размер агрегируемого сообщения и количеству chare-объеквтов (uChareSet), используемых для агрегации uchare-объектов.

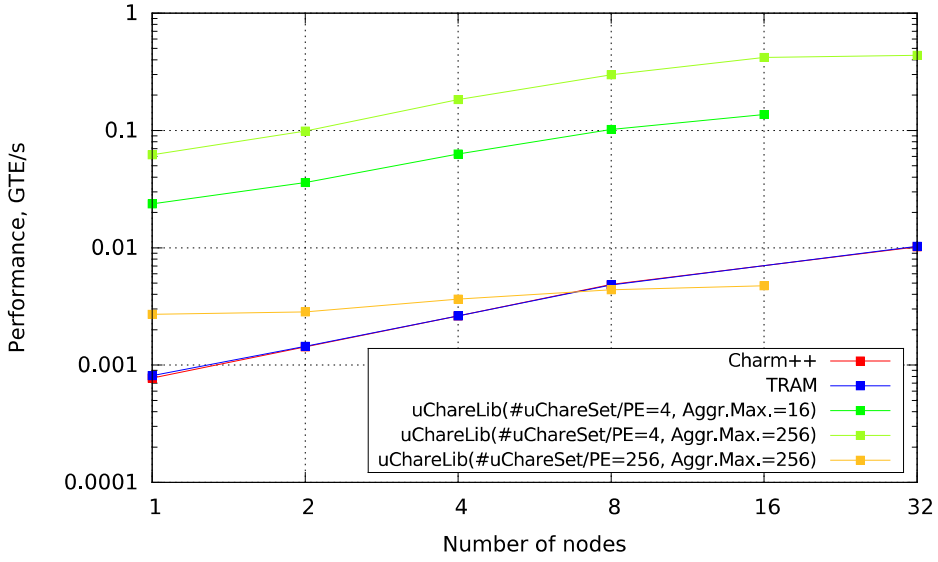


Рис.5. ABFS на МВС10П (равномерно случайный граф, размер графа $N=2^{16}$, K=16)

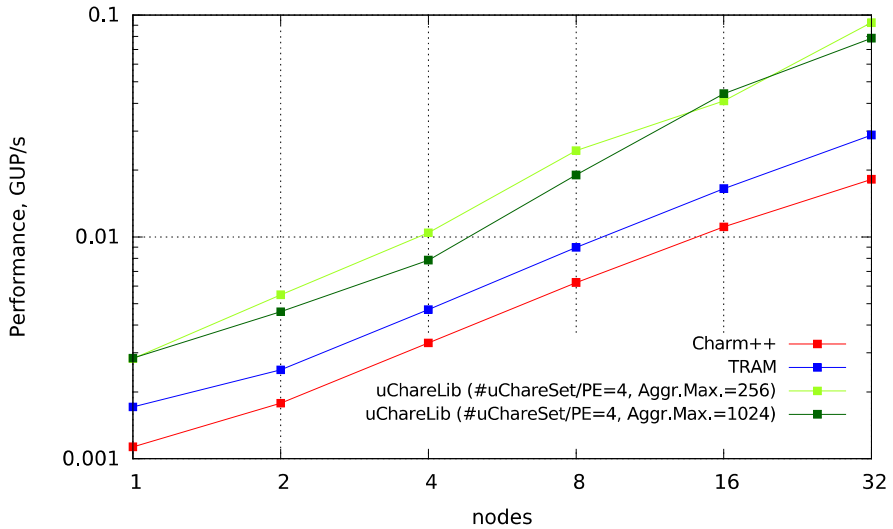


Рис.6. RandomAccess на 36-узловом кластере с сетью «Ангара»

На рисунке 6 приведены результаты запуска теста RandomAccess на 36-узловом вычислительном кластере на базе коммуникационной сети «Ангара». Узлы кластера объедирены в 3-х мерный тор – 3x3x4. Количество используемых ядер – 8.

Как видно из рисунка, соотношение производительности исследуемых реализаций теста RandomAccess аналогично результатам, полученным на МВС10П. Наименьшую производительность показал тест, реализованный на Charm++, далее TRAM и uChareLib.

Заключение

В статье представлена расширенная программная модель Charm++, ориентированная на обработку больших графов на экзафлопсных системах. Предлагаемые расширения включают: специализацию chare-объектов по целевой архитектуре и граннулярности параллелизма, поддержку динамических доменов синхронизации, многоуровневой глобально адресуемой распределенной общей памяти, а также поддержку проблемно-ориентированных языков программирования.

Показан эффект от одного из предлагаемых расширений – поддержки микро-chare-объектов, на двух тестах HPCC RandomAccess и Asynchronous BFS. Ускорение, полученное от использования микро-chare-объектов, составило x5 для RandomAccess и x44 раза для ABFS на МВС10П. Для 36-узлового кластера с сетью «Ангара» на тесте RandomAccess ускорение uChareLib относительно библиотеки TRAM составило 3 раза.

Дальнейшие планы по данному направлению включают исследование uChareLib на разнообразных графовых задачах и большом числе узлов, реализацию остальных предложенных расширений модели Charm++.

Работа выполнена при поддержке гранта РФФИ №15-07-09368.

Литература

1. J. A. Ang, R. F. Barrett, R. E. Benner, D. Burke, C. Chan, J. Cook, D. Donofrio, S. D. Hammond, K. S. Hemmert, S. M. Kelly, H. Le, V. J. Leung, D. R. Resnick, A. F. Rodrigues, J. Shalf, D. Stark, D. Unat, and N. J. Wright. 2014. Abstract machine models and proxy architectures for exascale computing. In Proceedings of the 1st International Workshop on Hardware-Software Co-Design for High Performance Computing (Co-HPC '14). IEEE Press, Piscataway, NJ, USA, 25-32. DOI=http://dx.doi.org/10.1109/Co-HPC.2014.4
2. Anrew Lumsdaine, New Execution Models are Required for Big Data at Exascale
3. Laxmikant V. Kale and Sanjeev Krishnan. Charm++: A portable concurrent object oriented system based on c++. SIGPLAN Not., 28(10):91–108, October 1993.
4. Lukasz Wesolowski, How to Write a Parallel GPU Application Using CUDA and Charm++, Charm Workshop 2010
5. Phil Miller, Aaron Becker, and Laxmikant Kalé. 2012. Using shared arrays in message-driven parallel programs. Parallel Comput. 38, 1-2 (January 2012), 66-74. DOI=http://dx.doi.org/10.1016/j.parco.2011.10.005
6. Aaron Becker, Phil Miller, Laxmikant V Kalé PGAS in the message-driven execution model 1st Workshop on Asynchrony in the PGAS Programming Model APGAS 2009
7. Sungpack Hong, Hassan Chafi, Edic Sedlar, and Kunle Olukotun. 2012. Green-Marl: a DSL for easy and efficient graph analysis. In Proceedings of the seventeenth international conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS XVII). ACM, New York, NY, USA, 349-362. DOI=http://dx.doi.org/10.1145/2150976.2151013
8. Piotr R Luszczek, David H Bailey, Jack J Dongarra, Jeremy Kepner, Robert F Lucas, Rolf Rabenseifner, and Daisuke Takahashi. The hpc challenge (hpcc) benchmark suite. In Proceedings of the 2006 ACM/IEEE Conference on Supercomputing, SC '06, New York, NY, USA, 2006. ACM.
9. Graph500, http://www.graph500.org.
10. Lukasz Wesolowski, Ramprasad Venkataraman, Abhishek Gupta, JaeSeung Yeom, Keith Bisset, Yanhua Sun, Pritish Jetley, Thomas R. Quinn, and Laxmikant V. Kale. TRAM: Optimizing Fine-grained Communication with Topological Routing and Aggregation of Messages. In Proceedings of the International Conference on Parallel Processing, ICPP '14, Minneapolis, MN, September 2014
11. Симонов А.С., Слуцкин А.И., Макагон Д.В., Сыромятников Е.Л., Жабин И.А., Фролов А.С., Щербак А.Н. Опыт разработки отечественной высокоскоростной коммуникационной сети для суперкомпьютеров. – Доклад на Третьем Московском суперкомпьютерном форуме (МСКФ-2012), Россия, г. Москва, ВВЦ, 01 ноября 2012 г. http://www.ospcon.ru/files/media/mscf_2012_tesis.pdf.
12. И.Жабин, Д.Макагон, А.Симонов, Е.Сыромятников, А.Фрлоов, Е.Щербак Кристалл для "Ангары", Суперкомпьютеры №4(16), 2013, 46-49.

Издательство «Лань» — одно из ведущих издательств в Российской Федерации, специализирующееся на выпуске учебной литературы для высшей школы. Мы стремимся осуществлять свою издательскую деятельность на самом высоком профессиональном уровне, формировать издательский портфель в тесном сотрудничестве с научно-преподавательским сообществом, издавать лучшую современную и классическую учебную литературу.

Редакция физико-математической литературы — старейшая редакция издательства «Лань», за годы работы сформирован солидный издательский портфель. Сейчас мы активно развиваем новое направление — учебная литература по информатике и информационным технологиям. По вопросам авторского сотрудничества просьба обращаться к заведующей физико-математической редакции Крамор Н.Р. E-mail: fml@lanbook.ru, тел.: (812) 336-25-09. http://www.lanbook.com/ 196105, Санкт-Петербург, пр. Юрия Гагарина, д.1 лит. А.

Рецензируемый научно-практический журнал **«Прикладная информатика» (ISSN 1993-8313)** является преемником одноименного периодического сборника, выпускавшегося с 1981 года издательством «Финансы и статистика» 6 раз в год.

В настоящее время учредителем и издателем журнала является НОУ ВПО «Московский финансово-промышленный университет «Синергия». Свидетельство о регистрации СМИ: ПИ № ФС77-57869 выдано Роскомнадзором 25.04.2014 г. Журнал «Прикладная информатика» входит в Перечень ведущих рецензируемых научных журналов и изданий, рекомендованных ВАК для публикации основных результатов диссертационных исследований. Тематика публикаций журнала связана с теоретическими и прикладными аспектами применения компьютерного моделирования и информационных технологий в различных областях профессиональной деятельности. Журнал принимает на рассмотрение материалы по следующим группам специальностей: 05.13.00 «Информатика, вычислительная техника и управление»; 08.00.13 «Математические и инструментальные методы экономики. Заместитель главного редактора Власова Е.А. E-mail: Evlasova@synergy.ru, Тел.: (495) 663-93-88 (доб. 18-39). http://www.appliedinformatics.ru/.

International Journal of Open Information Technologies

В Московском государственном университете имени М.В. Ломоносова на факультете вычислительной математики и кибернетики лабораторией открытых информационных технологий с 2013 года выпускается электронное периодическое издание **«International Journal of Open Information Technologies» (ISSN 2307-8162)**. Тематикой и специализацией журнала являются научные и прикладные аспекты теоретической информатики, прикладной математики, программирования, вычислительной техники, информационных и коммуникационных технологий, а также вопросы подготовки научных и высококвалифицированных кадров по указанным направлениям на основе современной системы ИТ-образования. Учредителем INJOIT является Фонд содействия развитию интернет-медиа, ИТ-образования, человеческого потенциала «Лига интернет-медиа». Свидетельство о регистрации СМИ (форма периодического распространения – сетевое издание): Эл № ФС77-61434, выдано Роскомнадзором 10.04.2015 г. Выходит 12 раз в год. Журнал принимает на рассмотрение материалы по следующим группам специальностей: 01.01.00 Математика; 05.12.00 Радиотехника и связь; 05.13.00 Информатика, вычислительная техника и управление. Заместитель главного редактора: Намиот Д.Е. E-mail: dnamiot@gmail.com; Тел./факс: (495) 939-46-26; http://injoit.ru/.

Издательство «Открытые системы», созданное в 1993 году, производящее высокопрофессиональные бизнес-издания для специалистов и руководителей из таких отраслей, как информационные технологии, компьютерные системы, телекоммуникации, сети передачи данных, полиграфия и медицина. Лидирующие позиции издательства обеспечиваются широким спектром изданий: **«Открытые системы. СУБД»**, **«Журнал сетевых решений/LAN»**, **«Computerworld Россия»**, **«Мир ПК»** **«Publish/Дизайн. Верстка. Печать»** и др. Главный редактор Волков Д.В. E-mail: vlk@osp.ru; Тел.: (499) 703-18-54 (доб.331). http://www.osp.ru/.