УДК 004.021; 519.713.2
DOI: 10.25559/SITITO.019.202303.531-542

Оригинальная статья

# Программное исследование объединения полурешеток на множестве подмножеств гридов языка Ватерлоо

**М. Э. Абрамян[1], Б. Ф. Мельников[2*]**
[1] ФГАОУ ВО «Южный федеральный университет», г. Ростов-на-Дону, Российская Федерация
Адрес: 344006, Российская Федерация, г. Ростов-на-Дону, ул. Б. Садовая, д. 105/42
[2] Совместный университет МГУ – ППИ, г. Шэньчжэнь, Китайская Народная Республика
Адрес: 517182, Китайская Народная Республика, провинция Гуандун, г. Шэньчжэнь, р-н Лунган, Даюньсиньчэн, ул. Гоцзидасюеюань, д. 1
[*] bormel@mail.ru

## Аннотация

**Актуальность.** Актуальность рассматриваемой предметной области обусловлена необходимостью исследования множества регулярных языков и, в частности, описания различных их подклассов. Также актуальны задачи, которые могут возникать в некоторых подклассах. Это даст, среди прочего, возможность описания новых алгоритмов эквивалентного преобразования недетерминированных конечных автоматов.

**Цель исследования.** Целью является исследование множества подмножеств гридов языка Ватерлоо с точки зрения абстрактной алгебры.

**Материалы и методы.** Исследование проводилось с применением библиотеки для работы с недетерминированными конечными автоматами NFALib, реализованной одним из авторов на языке C#, а также статистических методов анализа алгоритмов.

**Результаты.** Результатами являются закономерности, полученные при рассмотрении полурешеток на множестве подмножеств гридов языка Ватерлоо.

**Выводы.** Из полученных результатов следует, что минимальный покрывающий автомат, эквивалентный автомату Ватерлоо, можно получить, добавив к минимальному покрывающему множеству гридов один дополнительный. Проведенные расчеты также показывают, что кроме минимального покрывающего автомата можно получить еще 4 минимальных покрывающих автомата, эквивалентных исходному автомату Ватерлоо, однако для получения каждого из них необходимо заменить 1 или 2 грида, входящих в минимальное покрывающее множество.

**Ключевые слова:** недетерминированные конечные автоматы, универсальный автомат, базисный автомат, грид (блок), покрывающий автомат, алгоритмы эквивалентного преобразования, автомат Ватерлоо

**Конфликт интересов:** авторы заявляют об отсутствии конфликта интересов.

**Для цитирования:** Абрамян М. Э., Мельников Б. Ф. Программное исследование объединения полурешеток на множестве подмножеств гридов языка Ватерлоо // Современные информационные технологии и ИТ-образование. 2023. Т. 19, № 3. С. 531-542. https://doi.org/10.25559/SITITO.019.202303.531-542

Original article

# A Program Study of the Union of Semilattices on a Set of Subsets of Grids of the Waterloo Language

**M. E. Abramyan[a], B. F. Melnikov[b*]**
[a] Southern Federal University, Rostov-on-Don, Russian Federation
Address: 105/42 Bolshaya Sadovaya St., Rostov-on-Don 344006, Russian Federation
[b] Shenzhen MSU – BIT University, Shenzhen, People's Republic of China
Address: 1 Guoji-daxueyuan St., Dayunxincheng, Longgang District, Shenzhen 517182, Guangdong Province, People's Republic of China
[*] bormel@mail.ru

## Abstract
**Relevance.** The relevance of the subject area under consideration is due to the need to study the set of regular languages, in particular to describe their various sub-classes. Also relevant are the tasks that may arise in some such subclasses. This will give, among other things, the possibility of describing some new algorithms for equivalent transformation of nondeterministic finite automata.
**Purpose of the study.** The aim is to study the set of subsets of grids of the Waterloo language from the point of view of abstract algebra.
**Materials and methods.** The study was conducted using the library for working with nondeterministic finite automata NFALib implemented by one of the authors in C#, as well as statistical methods for analyzing algorithms.
**Results.** The results are regularities obtained when considering semilattices on a set of subsets of grids of the Waterloo language.
**Conclusions.** It follows from the results obtained that the minimum covering automaton equivalent to the Waterloo automaton can be obtained by adding one additional to the minimum covering set of grids. The calculations also show that in addition to the minimum covering automaton, it is possible to obtain 4 more minimum covering automata equivalent to the original Waterloo automaton, however, to obtain each of them, it is necessary to replace 1 or 2 grids included in the minimum covering set.

**Keywords:** nondeterministic finite automata, universal automaton, basic automaton, grid, covering automaton, equivalent transformation algorithms, automaton Waterloo

**Conflict of interests:** The authors declare no conflict of interest.

**For citation:** Abramyan M.E., Melnikov B.F. A Program Study of the Union of Semilattices on a Set of Subsets of Grids of the Waterloo Language. *Modern Information Technologies and IT-Education.* 2023;19(3):531-542. https://doi.org/10.25559/SITITO.019.202303.531-542

## Введение

Общеизвестно, что для описания регулярного языка существуют разные полные инварианты: не только хорошо известные канонические автоматы[1], но и базисные автоматы [1, 2], а также универсальные автоматы [3-6]. При построении базисных и универсальных автоматов необходимо построить канонические автоматы как для заданного регулярного языка, так и для его зеркального отражения. (Существуют и альтернативные алгоритмы построения, однако они имеют примерно такую же сложность и рассматриваются редко.) В процессе такого построения мы можем получить, среди прочих объектов, специальное бинарное отношение #, определенное на парах состояний этих двух канонических автоматов. Это отношение также является инвариантом (однако неполным) для рассматриваемого языка.

Конечно же, в настоящее время наиболее интересным для исследования является язык Ватерлоо. При этом в большинстве статей — как на английском языке, так и на русском — используется термин «автомат Ватерлоо». Однако из последующего будет понятно, что термин «язык Ватерлоо» лучше. Мы будем употреблять оба варианта.

Построенный для этого языка универсальный автомат обладает следующим свойством: среди соответствующих ему покрывающих автоматов [6-8] существует ему неэквивалентный. До сих пор неизвестно, является ли такой язык Ватерлоо минимальным, но, скорее всего, является. В одной из предыдущих статей [9] мы начали решать задачу доказательства этой минимальности — и эту работу желательно продолжить.

И, конечно, упомянутую в предыдущем абзаце «минимальность» желательно определить строго; она может рассматриваться по разным нормам[2]. Наиболее естественной нормой является число состояний эквивалентного канонического автомата – однако нам интереснее другие варианты:

- либо количество состояний эквивалентного базисного автомата — для языка Ватерлоо это значение равно 20;
- либо произведение количеств состояний двух эквивалентных канонических автоматов (для заданного регулярного языка и для языка, зеркального к заданному) — для Ватерлоо это значение равно 80.

Добавим, что в «смежных» задачах нас часто интересует также специальным образом вводимая «норма» конечного языка[3] (см. [10-12] и др.).

Выше уже были упомянуты покрывающие автоматы. Конечно же, они для любого автомата образуют полурешетку по объединению[4], однако можно показать, что они, вообще говоря, не образуют полурешетки по пересечению[5]. Конкретнее, вместо одной полурешетки по пересечению образуется объединение нескольких таких полурешеток. Рассмотрению подобной конструкции для языка Ватерлоо и посвящена настоящая статья.

Приведем содержание статьи по разделам. В разделе 1 приведена мотивация к исследованию подобных вопросов (иными словами, актуальность рассматриваемой темы). Раздел 2 — предварительные сведения; приведена связанная с конечными автоматами терминология, причем в основном те термины, которые связаны с элементами универсального автомата, прежде всего с гридами, которые можно рассматривать как состояния последнего.

В разделе 3 мы описываем общий подход к программному исследованию автоматов для Ватерлоо-подобных языков. Полученные при проведении вычислений результаты описаны в разделе 4; вследствие этого его можно рассматривать как «большое заключение». Однако, по-видимому, нельзя сказать, что настоящей статьей заканчивается работа на эту тематику: как уже было отмечено, остается нерешенной задача о минимальном Ватерлоо-подобном автомате.

## 1. Актуальность темы (мотивация)

Рассматриваемая тема примыкает к исследованию множества всех возможных дуг недетерминированного конечного автомата, определяющего заданный регулярный язык; такое исследование было начато одним из авторов настоящей статьи еще в конце 1990-х [14, 15] и впоследствии было продолжено в нескольких процитированных выше статьях: [5, 6, 8, 13] и др.

Итак, как уже было отмечено, при исследовании множества возможных дуг автомата (иными словами, дуг автомата COM, дуг универсального автомата) мы в качестве вспомогательной конструкции получаем специальное бинарное отношение #, определенное на парах состояний этих двух канонических автоматов. Это отношение также является инвариантом (однако неполным) для рассматриваемого языка. При этом для каждого такого бинарного отношения существует целый подкласс класса регулярных языков, который им обладает. Следовательно, на множестве всех обычных языков возможно опре-

[1] Гинзбург С. Математическая теория контекстно-свободных языков / Пер. с англ. А. Я. Диковского и Л. С. Модиной ; Под ред. А. В. Гладкого. М. : Мир, 1970. 326 с.; Ахо А., Ульман Дж. Теория синтаксического анализа, перевода и компиляции. М. : Мир, 1978. Т. 1. 613 с.; Брауэр В. Введение в теорию конечных автоматов. М. : Радио и связь, 1987. 392 с.; Мельников Б. Ф. Регулярные языки и недетерминированные конечные автоматы : монография. М. : Изд-во РГСУ, 2018. 179 с.

[2] Формально норма определяется для векторов, обладающих операцией умножения на число, поэтому в нашем случае терминология «неканоническая». Мы такими «векторами» считаем конечные автоматы (например, только канонические), а нормой – некоторую функцию, отображающую множество автоматов в множество вещественных чисел. Для двух изоморфных автоматов (т. е. полученных путем перестановки вершин и соответствующим изменением множества дуг) такая функция обязательно должна давать одинаковые значения, а для двух эквивалентных автоматов, но, вообще говоря, неизоморфных – необязательно равные. Норму регулярного языка при этом следует определять как норму выбранного особым образом автомата, задающего этот язык, – например, соответствующего канонического автомата.

[3] Мельников Б. Ф. Подклассы класса контекстно-свободных языков. М. : Изд-во Моск. ун-та, 1995. 174 с.

[4] Это утверждение верное, но при этом оно требует дополнительных вспомогательных определений (в частности, определения объединения автоматов); эти определения несложны, однако они выходят за рамки настоящей статьи. Важнее то, что полурешетку по объединению образуют соответствующие покрывающие подмножества множества гридов; при этом дополнительных определений не требуется: строгое определение грида будет приведено далее, а мы при этом просто рассматриваем объединения подмножеств.

[5] Этот факт можно рассматривать как одно из следствий сформулированного выше свойства автомата Ватерлоо. Мы называем такие языки и автоматы Ватерлоо-подобными [13], однако нам фактически известен только один такой автомат – не считая тех, которые могут быть получены из него переобозначениями, добавлениями вершин и т. п.

делить (еще одно) бинарное отношение; оно справедливо для некоторых двух языков тогда и только тогда, когда они имеют одинаковое бинарное отношение #. Очевидно, что определенное таким образом бинарное отношение (в некоторых наших предыдущих работах, [11, 12, 16] и др., оно обозначалось R) является на множестве всех регулярных языков отношением эквивалентности. Поэтому возникает вопрос о «наиболее типичном» языке, который является элементом рассматриваемого класса эквивалентности по рассматриваемому отношению.

В нескольких предыдущих статьях мы подробно исследовали регулярные языки и соответствующие им канонические автоматы, которые можно считать такими «типичными элементами» своего класса, рассматривали некоторые их свойства [16-18]. По-видимому, самым интересным из этих свойств является следующее: используя специально описанные преобразования, мы из подобного «типичного» автомата можем получить любой канонический автомат, язык которого соответствует рассматриваемому бинарному отношению #.

Повторим, что к настоящему моменту мы знаем только один пример Ватерлоо-подобного языка (т.е. языка, для универсального автомата которого существует покрывающий автомат, ему неэквивалентный) — статью [9] можно рассматривать как формулировку задачи поиска других подобных языков. Также отметим возможную связь Ватерлоо-подобных языков с бесконечными итерациями конечных языков — [11, 12] и др.

В заключение раздела отметим, что многие из перечисленных здесь задач можно рассматривать как задачи упаковки контейнера — [19] и мн. др., — однако, конечно же, решать их именно таким образом нецелесообразно, для них нужны другие модели, схожие с рассматриваемыми в настоящей статье.

## 2. Предварительные сведения

Основная часть связанной с недетерминированными конечными автоматами терминологии согласована с работой [11] (см. также ссылки из данной статьи). Здесь мы добавим ссылки на некоторые термины, непосредственно связанные с минимизацией автоматов.

Рассмотренный в работе[6] алгоритм вершинной минимизации недетерминированных конечных автоматов основан на анализе бинарного отношения #, связывающего множества состояний $X$ и $Y$ двух канонических автоматов, которые построены на основе анализируемого недетерминированного конечного автомата $K$ (определяющего некоторый регулярный язык $L$) и зеркального к нему[7].

С отношением # связывается набор гридов[8]. Каждый грид определяется парой подмножеств $X_0 \subset X$ и $Y_0 \subset Y$, удовлетворяющих следующему условию: для любых состояний $x \in X_0$ и $y \in Y_0$ выполняется $x \# y$, причем подмножества $X_0$ и $Y_0$ нельзя расширить с сохранением указанного условия. Такой грид будем обозначать $X_0 \times Y_0$. Отметим, что гриды можно рассматривать как состояния универсального автомата.

Множество $M$ гридов называется покрывающим, если для любых элементов $x \in X$ и $y \in Y$, таких, что $x \# y$, найдется грид $X_0 \times Y_0$ из $M$, для которого $x \in X_0$ и $y \in Y_0$. Очевидно, что полный набор гридов, построенных по отображению #, является покрывающим множеством.

В работе [6] описан алгоритм построения по полному набору гридов автомата COM($L$) (фактически изоморфного универсальному автомату), который определяет тот же регулярный язык $L$, что и исходный автомат $K$, и при этом каждый грид соответствует некоторому состоянию автомата COM($L$).

На основе автомата COM($L$) можно определить семейство покрывающих автоматов, каждый из которых получается путем удаления некоторых состояний автомата COM($L$), причем оставшиеся состояния соответствуют гридам, образующим покрывающее множество.

Алгоритм минимизации исходного автомата $K$ состоит в том, чтобы выбрать покрывающее множество гридов $M_0$ минимального размера, для которого построенный на его основе покрывающий автомат является эквивалентным автомату К, т. е. определяет тот же регулярный язык.

Примеры показывают, что не всякое покрывающее множество гридов позволяет получить покрывающий автомат, эквивалентный исходному. Известным примером является автомат Ватерлоо, впервые приведенный в [20], подробно проанализированный в работе[9].

В настоящей работе приводятся результаты дополнительного исследования покрывающих автоматов, созданных на основе автомата Ватерлоо. Исследование проводилось с применением библиотеки для работы с недетерминированными конечными автоматами NFALib, реализованной одним из авторов на языке C#, см. [21-25] и др.

## 3. О подходе к программному исследованию автоматов для Ватерлоо-подобных языков

Перед описанием полученных результатов опишем использованные объекты и приведем фрагменты программы, позволяющие их получить.

Исходный автомат (канонический автомат для регулярного языка Ватерлоо) задается с помощью его текстового описания, содержащегося в файле Waterloo.txt (это описание соответствует[10]):

[6] Мельников Б. Ф. Регулярные языки и недетерминированные конечные автоматы : монография. М. : Изд-во РГСУ, 2018. (разд. 6).
[7] Мельников Б. Ф. Регулярные языки и недетерминированные конечные автоматы : монография. М. : Изд-во РГСУ, 2018. (разд. 3.3).
[8] Мельников Б. Ф. Регулярные языки и недетерминированные конечные автоматы : монография. М. : Изд-во РГСУ, 2018. (разд. 3.4).
[9] Мельников Б. Ф. Регулярные языки и недетерминированные конечные автоматы : монография. М. : Изд-во РГСУ, 2018. (разд. 6).
[10] Мельников Б. Ф. Регулярные языки и недетерминированные конечные автоматы : монография. М. : Изд-во РГСУ, 2018. (табл. 13).

```
NFA w = NFA.FromFile("Waterloo.txt");
  %% Waterloo
            a   b
  ==> A     E   -
      B     F   -
      C     G   B
      D     C   H
  <== E     C   H
  <== F     B   D
      G     A   D
      H     A   -
```

Это детерминированный автомат, канонизация которого приводит к автомату такого же вида. Канонический автомат для зеркального автомата после переобозначения состояний, принимает вид, приведенный в работе[11]:

```
string[] names = new string[] { "X", "Y", "Z", "U", "V",
  "W", "P", "Q", "R", "S" };
NFA w2 = w.CurrentMirrorCanonicalDFA
  .GetRenamedStates(i => names[i]);
            a   b   % States
  ==> ×     Y   -   % E-F
  <== Y     Z   U   % A-B
      Z     V   W   % F-G-H
      U     W   -   % C
      V     P   U   % B-C
      W     Q   R   % D-E
      P     Y   R   % D-E-F
  <== Q     S   -   % A
      R     V   -   % F-G
      S     U   W   % G-H
```

В столбце States приводятся множества состояний до их переобозначения; каждое также состояние соответствует множеству состояний исходного зеркального автомата, полученному в результате его детерминизации.

На основе полученных канонических автоматов строится матрица отношения #, строки которой соответствуют состояниям канонического автомата w, а столбцы — состояниям автомата w2, канонического к зеркальному. Эта матрица соответствует[12]:

```
SharpRelation sharpRelation = w.CurrentSharpRelation;
  %% NFA: Waterloo
     ×  Y  Z  U  V  W  P  Q  R  S
  A  -  #  -  -  -  -  -  #  -  -
  B  -  #  -  -  #  -  -  -  -  -
  C  -  -  -  #  #  -  -  -  -  -
  D  -  -  -  -  -  #  #  -  -  -
  E  #  -  -  -  -  #  #  -  -  -
  F  #  -  #  -  -  -  #  -  #  -
  G  -  -  #  -  -  -  -  -  #  #
  H  -  -  #  -  -  -  -  -  -  #
```

Далее для найденного отношения # определяется полный набор из 14 гридов. После небольшого изменения порядка их следования мы получаем набор, полностью совпадающий с описанным в работе[13].

[11] Мельников Б. Ф. Регулярные языки и недетерминированные конечные автоматы : монография. М. : Изд-во РГСУ, 2018. (табл. 16).
[12] Мельников Б. Ф. Регулярные языки и недетерминированные конечные автоматы : монография. М. : Изд-во РГСУ, 2018. (табл. 17).
[13] Мельников Б. Ф. Регулярные языки и недетерминированные конечные автоматы : монография. М. : Изд-во РГСУ, 2018. С. 107.

```
List<Grid> completeGrids =
  sharpRelation.GetCompleteGrids().ToList();
completeGrids.Add(completeGrids[11]);
completeGrids.RemoveAt(11);
completeGrids.Add(completeGrids[6]);
completeGrids.RemoveAt(6);
  Complete Grids
  { A } × { Y, Q }  % 1
  { A, B } × { Y }  % 2
  { B } × { Y, V }  % 3
  { B, C } × { V }  % 4
  { C } × { U, V }  % 5
  { D, E } × { W, P }  % 6
  { E } × { X, W, P }  % 7
  { E, F } × { X, P }  % 8
  { F } × { X, Z, P, R }  % 9
  { F, G } × { Z, R }  % 10
  { G } × { Z, R, S }  % 11
  { G, H } × { Z, S }  % 12
  { F, G, H } × { Z }  % 13
  { D, E, F } × { P }  % 14
```

На основе полученного полного набора гридов строится полный автомат, представление которого совпадает с приведенным в работе[14].

```
NFA com = w.GetCOM(completeGrids, "COM");
%% COM
         a               b        % Grids
==> 1    6,7,8,14        -        % { A } × { Y, Q }
==> 2    8,14            -        % { A, B } × { Y }
    3    8,9,10,13,14    -        % { B } × { Y, V }
    4    10,13           -        % { B, C } × { V }
    5    10,11,12,13     2,3,4    % { C } × { U, V }
    6    4,5             12,13    % { D, E } × { W, P }
<== 7    4,5             12,13    % { E } × { X, W, P }
<== 8    4               -        % { E, F } × { X, P }
<== 9    2,3,4           6,14     % { F } × { X, Z, P, R }
    10   2               6,14     % { F, G } × { Z, R }
    11   1,2             6,14     % { G } × { Z, R, S }
    12   1,2             -        % { G, H } × { Z, S }
    13   2               -        % { F, G, H } × { Z }
    14   4               -        % { D, E, F } × { P }
```

Построенный полный автомат является эквивалентным исходному автомату Ватерлоо, что можно показать, выполнив его детерминизацию:

```
NFA w3 = com.GetCanonicalDFA();
                       a              b
  ==> 1-2              6-7-8-14       -
  <== 6-7-8-14         4-5            12-13
      4-5              10-11-12-13    2-3-4
      12-13            1-2            -
      10-11-12-13      1-2            6-14
      2-3-4            8-9-10-13-14   -
      6-14             4-5            12-13
  <== 8-9-10-13-14     2-3-4          6-14
```

[14] Мельников Б. Ф. Регулярные языки и недетерминированные конечные автоматы : монография. М. : Изд-во РГСУ, 2018. (табл. 20).

После переименования состояний получаем автомат, представление которого совпадает с представлением исходного автомата Ватерлоо:

```
string[] names3 = new string[] { "A", "E", "C", "H",
  "G", "B", "D", "F" };
NFA w3a = w3.GetRenamedStates(i => names3[i])
  .GetOrderedStates();
          a   b   % States
  ==> A   E   -   % 1-2
      B   F   -   % 2-3-4
      C   G   B   % 4-5
      D   C   H   % 6-14
  <== E   C   H   % 6-7-8-14
  <== F   B   D   % 8-9-10-13-14
      G   A   D   % 10-11-12-13
      H   A   -   % 12-13
```

Далее определяется минимальное покрывающее множество гридов. Это множество содержит гриды 1, 3, 5, 6, 8, 10, 12 из полного набора:

```
List<Grid> minGrids =
  sharpRelation.GetMinGridCovers().First();
  Min Grids
  { A } × { Y, Q }  % 1
  { B } × { Y, V }  % 3
  { C } × { U, V }  % 5
  { D, E } × { W, P }  % 6
  { E, F } × { X, P }  % 8
  { F, G } × { Z, R }  % 10
  { G, H } × { Z, S }  % 12
```

На основе минимального покрывающего множества гридов строится покрывающий автомат:

```
NFA w4 = com.GetCovering(minGrids);
           a       b    % Grids
  ==> 1    6,8     -    % { A } × { Y, Q }
      3    8,10    -    % { B } × { Y, V }
      5    10,12   3    % { C } × { U, V }
      6    5       12   % { D, E } × { W, P }
  <== 8    -       -    % { E, F } × { X, P }
      10   -       6    % { F, G } × { Z, R }
      12   1       -    % { G, H } × { Z, S }
```

После процедуры канонизации полученный покрывающий автомат принимает следующий вид:

```
NFA w4a = w4.GetCanonicalDFA();
              a       b
  ==> 1       6-8     -
  <== 6-8     5       12
      5       10-12   3
      12      1       -
      10-12   1       6
      3       8-10    -
      6       5       12
  <== 8-10    -       6
```

Выполнив переименование состояний, аналогичное тому, которое было выполнено для детерминированного представления полного автомата, получаем автомат, который не является эквивалентным исходному автомату Ватерлоо (отличающаяся строка выделена в представлении автомата полужирным шрифтом):

```
NFA w4b = w4a.GetRenamedStates(i => names1[i])
   .GetOrderedStates();
          a    b    % States
==> A     E    -    % 1
    B     F    -    % 3
    C     G    B    % 5
    D     C    H    % 6
<== E     C    H    % 6-8
<== F     -    D    % 8-10
    G     A    D    % 10-12
    H     A    -    % 12
```

Итак, покрывающий автомат, построенный на основе минимального покрывающего множества гридов, не является эквивалентным исходному.

## 4. Результаты вычислений для покрывающих автоматов

В оставшейся части статьи мы описываем результаты, полученные при анализе других покрывающих автоматов, которые могут быть построены на основе минимального покрывающего множества гридов, дополненного какими-либо другими гридами из исходного полного набора гридов.

Отметим, что все подмножества множества гридов, согласно приведенным в предыдущем разделе определениям, образуют полурешетку по объединению. Однако достаточно простые примеры показывают, что полурешетки по пересечению они не образуют. Для доказательства этого факта достаточно, например, выполнить такие построения.

- Рассмотреть бинарное отношение #, заданное следующей таблицей:

```
# # # .
# # . #
# . # #
. # # #
```

- Далее согласно [16, 17] построить полный автомат, заданный этим отношением (этой таблицей).
- В этом автомате выделить два множества гридов: первое множество соответствует строкам таблицы (это фактически состояния канонического автомата для языка полного автомата, $K^{\#}$ в обозначениях [16]), а второе — столбцам таблицы (это состояния канонического автомата для зеркального языка, $K_{\#}$ в обозначениях [16]). Отметим, что в обоих случаях каждая из «строк» и каждый из «столбцов» действительно образуют грид — это очевидно.

Поскольку мы рассматриваем канонические автоматы, то:

- во-первых, эти множества гридов подходящие (т. е. каждое их них действительно определяет требуемый язык);
- во-вторых, каждое из этих множеств невозможно уменьшить (т. е. удалить из него какой-либо элемент без изменения языка соответствующего ему автомата);
- однако к каждому из них можно добавлять не входящие в него гриды (которые, как мы уже знаем, имеются) без изменения языка получаемого автомата.

Таким образом, получается вынесенное в заглавие объединение полурешеток над подмножествами гридов.

Как будет видно из дальнейшего, целесообразно немного изменить порядок дополнительных гридов, переместив грид 9 полного набора в начало набора дополнительных гридов:

```
Add Grids
{ F } × { X, Z, P, R }  % 9
{ A, B } × { Y }  % 2
{ B, C } × { V }  % 4
{ E } × { X, W, P }  % 7
{ G } × { Z, R, S }  % 11
{ F, G, H } × { Z }  % 13
{ D, E, F } × { P }  % 14
```

Каждый покрывающий автомат будем обозначать 7-значным двоичным числом, в котором цифрой 1 помечаются дополнительные гриды, включенные в покрывающее множество гридов, на основе которого он построен. В частности, автомат 1111111 соответствует полному автомату, построенному на основе полного набора гридов, а автомат 0000000 соответствует рассмотренному выше автомату, построенному на основе минимального покрывающего множества. Порядок разрядов соответствует приведенному выше порядку дополнительных гридов.

На основе описанных выше средств библиотеки NFALib был построен набор из 128 покрывающих автоматов и исследована их эквивалентность. Для проверки эквивалентности двух автоматов использовался алгоритм, в котором:

- вначале автоматы приводятся к каноническому виду,
- после чего для каждого состояния автоматов строится некоторая хеш-характеристика,
- и в случае совпадения хеш-характеристик выполняется сравнение функций переходов для различных комбинаций новых имен второго автомата с применением алгоритма перебора с возвратом.

В результате были получены четыре набора попарно эквивалентных автоматов, свойства которых приведены в таблице 1.

Таблица 1.

Table 1.

| Обозначение набора | Номера автоматов, входящих в набор | Количество автоматов в наборе | Эквивалентны исходному автомату Waterloo? |
|---|---|---|---|
| A | 0000000–0001111 | 16 | нет |
| B | 0010000–0011111 | 16 | нет |
| C | 0100000–0101111 | 16 | нет |
| D | 0110000–1111111 | 80 | да |

*Источник*: составлено авторами.

*Source*: Compiled by the authors.

Кроме того, были исследованы ситуации, при которых в результате добавления некоторого дополнительного грида происходит переход от автомата, неэквивалентного исходному автомату Ватерлоо, к эквивалентному автомату. Всего имеется 80 переходов, причем для каждого из 48 неэквивалентных автоматов имеется один или два подобных перехода:

- один для автоматов из набора A,
- два для автоматов из наборов B и C.

Полученные результаты можно представить в графическом виде: изобразить 7-мерный гиперкуб, выделив на нем светло-красным цветом вершины, которые соответствуют покрывающим автоматам, неэквивалентным исходному автомату Ватерлоо, а также ребра, которые соответствуют переходу от неэквивалентных к эквивалентным автоматам.

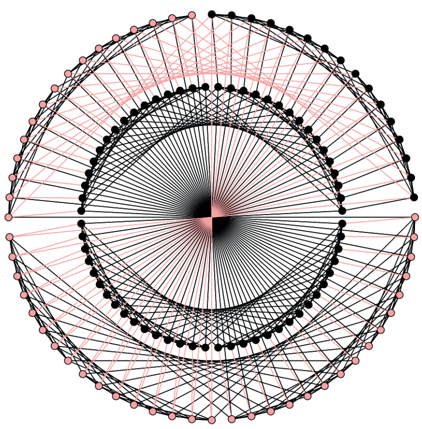

Рис. 1. Внешняя «окружность» содержит первую половину вершин (от 0000000 до 0111111), перебираемых по часовой стрелке от крайней правой вершины, а внутренняя «окружность» — вторую половину вершин (от 1000000 до 1111111), перебираемых в том же порядке

Fig. 1. The outer "circle" contains the first half of vertices (from 0000000 to 0111111), iterated clockwise from the rightmost vertex, and the inner "circle" contains the second half of vertices (from 1000000 to 1111111), iterated in the same order

*Источник*: здесь и далее в статье все рисунки составлены авторами.

*Source*: Hereinafter in this article all figures were drawn up by the authors.

Еще более наглядное представление наборов автоматов и переходов между ними можно получить, если представить каждый из наборов A, B, C в виде 4-мерного гиперкуба, а набор D — в виде пяти 4-мерных гиперкубов, после чего естественным образом объединить полученные гиперкубы в один 7-мерный гиперкуб (рис. 2).

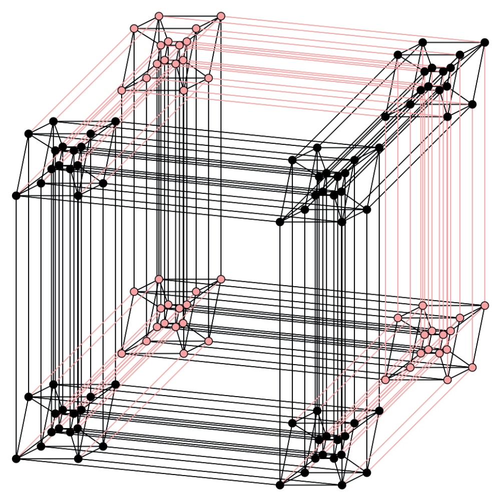

Рис. 2. Один 7-мерный гиперкуб

Fig. 2. One 7-dimensional hypercube

Рисунок 2 показывает, что вершины 7-мерного гиперкуба распадаются на классы эквивалентности по 16 элементов, у которых совпадают три первых двоичных разряда. Интересующие нас свойства автоматов из каждого такого класса эквивалентности одинаковы, поэтому, чтобы полностью проиллюстрировать полученные результаты, достаточно использовать трехмерный «фактор-куб», в котором вершины определяются первыми тремя двоичными разрядами или, что то же самое, наличием или отсутствием в соответствующем покрывающем автомате гридов 9, 2 и 4 (рис. 3).

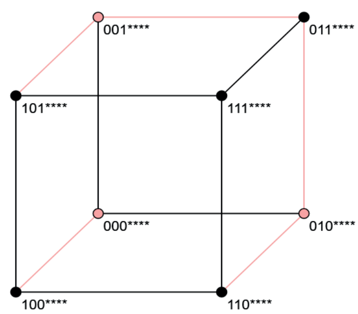


Рис. 3.

Fig. 3.

Из полученных результатов следует, что минимальный покрывающий автомат, эквивалентный автомату Waterloo, можно получить, добавив к минимальному покрывающему множеству гридов дополнительный грид 9, т. е. выполнив переход от автомата 0000000 к автомату 1000000.

Дополнительные расчеты показывают, что кроме минимального покрывающего автомата 1000000 можно получить еще 4 минимальных покрывающих автомата, эквивалентных исходному автомату Waterloo (из 11 различных покрывающих автоматов с 8 состояниями), однако для получения каждого из них необходимо заменить 1 или 2 грида, входящих в минимальное покрывающее множество, на 2 или, соответственно, 3 грида из дополнительного множества:

- грид 3 — на гриды 2 и 4,
- грид 8 — на гриды 7 и 9,
- грид 10 — на гриды 9 и 11,
- гриды 8 и 10 — на гриды 7, 9 и 11.

## Список использованных источников


[1] Melnikov B. F., Melnikova A. A. Some properties of the basis finite automaton // Korean Journal of Computational & Applied Mathematics. 2002. Vol. 9, issue 1. P. 135-150. https://doi.org/10.1007/BF03012345

[2] Долгов В. Н., Мельников Б. Ф., Мельникова А. А. Циклы графа переходов базисного конечного автомата и связанные вопросы // Вестник Воронежского государственного университета. Серия: Физика. Математика. 2016. № 4. С. 95-111. EDN: WYMNXF

[3] Polák L. Minimalizations of NFA using the universal automaton // International Journal of Foundation of Computer Science. 2005. Vol. 16, no. 05. P. 999-1010. https://doi.org/10.1142/S0129054105003431

[4] Lombardy S., Sakarovitch J. The Universal Automaton / Logic and Automata: History and Perspectives. Texts in Logic and Games. Vol. 2 ; ed. by J. Flum, E. Grädel, T. Wilke. Amsterdam University Press, 2007. P. 467-514. URL: https://perso.telecom-paristech.fr/jsaka/PUB/Files/TUA.pdf (дата обращения: 08.09.2023).

[5] Зубова М. А., Мельников Б. Ф. Об одном алгоритме построения универсального автомата Конвея // Вестник Воронежского государственного университета. Серия: Физика. Математика. 2012. № 1. С. 135-137. EDN: NMFEYH

[6] Долгов В. Н., Мельников Б. Ф. Построение универсального конечного автомата. II. Примеры работы алгоритмов // Вестник Воронежского государственного университета. Серия: Физика. Математика. 2014. № 1. С. 78-85. EDN: SBHVZH

[7] Melnikov B. F., Melnikova A. A. Edge-minimization of non-deterministic finite automata // Korean Journal of Computational & Applied Mathematics. 2001. Vol. 8, issue 3. P. 469-479. https://doi.org/10.1007/BF02941980

[8] Мельников Б., Мельникова А. Многоаспектная минимизация недетерминированных конечных автоматов. Ч. I. Вспомогательные факты и алгоритмы // Известия высших учебных заведений. Поволжский регион. Физико-математические науки. 2011. № 4(20). С. 59-69. EDN: NDWBHK

[9] Абрамян М. Э., Мельников Б. Ф., Мельникова Е. А. Таблица состояний конечного автомата: научный проект для старшеклассников // Компьютерные инструменты в образовании. 2019. № 2. С. 87-107. https://doi.org/10.32603/2071-2340-2019-2-86-107

[10] Brosalina A., Melnikov B. Commutation in Global Supermonoid of Free Monoid // Informatica. 2000. Vol. 11, issue 4. P. 353-370. https://doi.org/10.3233/INF-2000-11401

[11] Мельников Б. Лепестковые конечные автоматы: основные определения, примеры и их связь с полными автоматами. Часть I // International Journal of Open Information Technologies. 2022. Т. 10, № 9. С. 1-11. EDN: MARYFR

[12] Мельников Б. Лепестковые конечные автоматы: основные определения, примеры и их связь с полными автоматами. Часть II // International Journal of Open Information Technologies. 2022. Т. 10, № 10. С. 1-10. EDN: CPJNGM

[13] Долгов В. Н., Мельников Б. Ф. Об алгоритмах автоматического построения Ватерлоо-подобных конечных автоматов на основе полных автоматов // Эвристические алгоритмы и распределенные вычисления. 2014. Т. 1, № 4. С. 24-45. EDN: SYEHCT

[14] Melnikov B. F., Vakhitova A. A. Some more on the finite automata // Korean Journal of Computational & Applied Mathematics. 1998. Vol. 5, issue 3. P. 495-505. https://doi.org/10.1007/BF03008877

[15] Melnikov B. F., Sciarini-Guryanova N. V. Possible edges of a finite automaton defining a given regular language // Korean Journal of Computational & Applied Mathematics. 2002. Vol. 9, issue 2. P. 475-485. https://doi.org/10.1007/BF03021555

[16] Melnikov B. F. The complete finite automaton // International Journal of Open Information Technologies. 2017. Vol. 5, no. 10. P. 9-17. EDN: ZISOBN

[17] Melnikov B. F., Melnikova E. A. Waterloo-Like Finite Automata and Algorithms for their Automatic Construction // International Journal of Open Information Technologies. 2017. Vol. 5, no. 12. P. 8-15. EDN: ZWRIED

[18] Мельников Б. Ф., Долгов В. Н. Упрощенные регулярные языки и специальное отношение эквивалентности на классе регулярных языков. Часть I // International Journal of Open Information Technologies. 2022. Т. 10, № 9. С. 12-20. EDN: WJFXEA

[19] Hromkovič J. Algorithmics for Hard Problems: Introduction to Combinatorial Optimization, Randomization, Approximation, and Heuristics. 2nd ed. Berlin : Springer, 2004. 538 p. https://doi.org/10.1007/978-3-662-05269-3

[20] Kameda T., Weiner P. On the State Minimization of Nondeterministic Finite Automata // IEEE Transactions on Computers. 1970. Vol. C-19, no. 7. P. 617-627. https://doi.org/10.1109/T-C.1970.222994

[21] Li L., Qiu D. On the State Minimization of Fuzzy Automata // IEEE Transactions on Fuzzy Systems. 2015. Vol. 23, issue 2. P. 434-443. https://doi.org/10.1109/TFUZZ.2014.2315620

[22] Maarand H., Tamm H. Yet Another Canonical Nondeterministic Automaton // Descriptional Complexity of Formal Systems. DCFS 2022. Lecture Notes in Computer Science ; ed. by Y. S. Han, G. Vaszil. Vol. 13439. Cham : Springer, 2022. P. 184-196. https://doi.org/10.1007/978-3-031-13257-5_14

[23] Melnikov B., Melnikova A. A Polynomial Algorithm for Construction an Automaton for Checking the Equality of Infinite Iterations of Two Finite Languages // Artificial Intelligence Trends in Systems. CSOC 2022. Lecture Notes in Networks and Systems ; ed. by R. Silhavy. Vol. 502. Cham : Springer, 2022. P. 521-530. https://doi.org/10.1007/978-3-031-09076-9_47

[24] Абрамян М. Э. О вычислении веса подзадач при вершинной минимизации недетерминированных конечных автоматов методом ветвей и границ // Известия высших учебных заведений. Поволжский регион. Физико-математические науки. 2021. № 2(58). С. 46-52. https://doi.org/10.21685/2072-3040-2021-2-4

[25] Abramyan M., Melnikov B. A Program Study of the Union of Semilattices on the Set of Subsets of Grids of Waterloo Language // Journal of Applied Mathematics and Physics. 2023. Vol. 11. P. 1459-1470. https://doi.org/10.4236/jamp.2023.115095

*Поступила 08.09.2023; одобрена после рецензирования 01.10.2023; принята к публикации 05.10.2023.*

Об авторах:

**Абрамян Михаил Эдуардович**, доцент Института математики, механики и компьютерных наук им. И. И. Воровича, ФГАОУ ВО «Южный федеральный университет» (344006, Российская Федерация, г. Ростов-на-Дону, ул. Б. Садовая, д. 105/42), кандидат физико-математических наук, доцент, **ORCID: https://orcid.org/0000-0002-2802-6144,** mabr@sfedu.ru

**Мельников Борис Феликсович**, профессор факультета вычислительной математики и кибернетики, Совместный университет МГУ – ППИ (517182, Китайская Народная Республика, провинция Гуандун, г. Шэньчжэнь, р-н Лунган, Даюньсиньчэн, ул. Гоцзидасюеюань, д. 1), доктор физико-математических наук, профессор, **ORCID: https://orcid.org/0000-0002-6765-6800,** bormel@mail.ru

*Все авторы прочитали и одобрили окончательный вариант рукописи.*

## References

[1] Melnikov B.F., Melnikova A.A. Some properties of the basis finite automaton. *Korean Journal of Computational & Applied Mathematics*. 2002;9(1):135-150. https://doi.org/10.1007/BF03012345

[2] Dolgov V.N., Melnikov B.F., Melnikova A.A. The loops of the basis finite automaton and the connected questions. *Proceedings of Voronezh State University. Series: Physics. Mathematics*. 2016;(4):95-111. (In Russ., abstract in Eng.) EDN: WYMNXF

[3] Polák L. Minimalizations of NFA using the universal automaton. *International Journal of Foundation of Computer Science.* 2005;16(05):999-1010. https://doi.org/10.1142/S0129054105003431

[4] Lombardy S., Sakarovitch J. The Universal Automaton. In: Flum J., Grädel E., Wilke T. (eds.) Logic and Automata: History and Perspectives. Texts in Logic and Games. Vol. 2. Amsterdam University Press; 2007. p. 467-514. Available at: https://perso.telecom-paristech.fr/jsaka/PUB/Files/TUA.pdf (accessed 08.09.2023).

[5] Zubova M.A., Melnikov B.F. On algorithm of constructing Conway's universal automaton. *Proceedings of Voronezh State University. Series: Physics. Mathematics*. 2012;(1):135-137. (In Russ., abstract in Eng.) EDN: NMFEYH

[6] Dolgov V.N., Melnikov B.F. Construction of universal finite automaton. II. Examples of algorithms functioning. *Proceedings of Voronezh State University. Series: Physics. Mathematics*. 2014;(1):78-85. (In Russ., abstract in Eng.) EDN: SBHVZH

[7] Melnikov B.F., Melnikova A.A. Edge-minimization of non-deterministic finite automata. *Korean Journal of Computational & Applied Mathematics*. 2001;8(3):469-479. https://doi.org/10.1007/BF02941980

[8] Melnikov B.F., Melnikova A.A. Multi-aspect minimization non-deterministic finite automata (Part I. Supporting facts and algorithms). *University proceedings. Volga region. Physical and mathematical sciences*. 2011:(4):59-69. (In Russ., abstract in Eng.) EDN: NDWBHK

[9] Abramyan M.E., Melnikov B.F., Melnikova E.A. Finite Automata Table: A Science Project for High School Students. *Kompjuternye instrumenty v obrazovanii* = Computer Tools in Education journal. 2019;(2):87-107. (In Russ., abstract in Eng.) https://doi.org/10.32603/2071-2340-2019-2-86-107

[10] Brosalina A., Melnikov B. Commutation in Global Supermonoid of Free Monoids. *Informatica*. 2000;11(4):353-370. https://doi.org/10.3233/INF-2000-11401

[11] Melnikov B.F. Petal (Semi-Flower) Finite Automata: Basic Definitions, Examples and their Relation to Complete Automata. Part I. *International Journal of Open Information Technologies*. 2022;10(9):1-11. (In Russ., abstract in Eng.) EDN: MARYFR

[12] Melnikov B.F. Petal (Semi-Flower) Finite Automata: Basic Definitions, Examples and their Relation to Complete Automata. Part II. *International Journal of Open Information Technologies*. 2022;10(9):1-11. (In Russ., abstract in Eng.) EDN: CPJNGM

[13] Dolgov V.N., Melnikov B.F. On Algorithms of Automatic Construction of Waterloo-Like Finite Automata on the Basis of Full Automata. *Heuristic Algorithms and Distributed Computing*. 2014;1(4):24-45. (In Russ., abstract in Eng.) EDN: SYEHCT

[14] Melnikov B.F., Vakhitova A.A. Some more on the finite automata. *Korean Journal of Computational & Applied Mathematics*. 1998;5(3):495-505. https://doi.org/10.1007/BF03008877

[15] Melnikov B.F., Sciarini-Guryanova N.V. Possible edges of a finite automaton defining a given regular language. *Korean Journal of Computational & Applied Mathematics*. 2002;9(2):475-485. https://doi.org/10.1007/BF03021555

[16] Melnikov B.F. The complete finite automaton. *International Journal of Open Information Technologies*. 2017;5(10):9-17. EDN: ZISOBN

[17] Melnikov B.F., Melnikova E.A. Waterloo-Like Finite Automata and Algorithms for their Automatic Construction. *International Journal of Open Information Technologies*. 2017;5(12):8-15. EDN: ZWRIED

[18] Melnikov B.F., Dolgov V.N. Simplified Regular Languages and a Special Equivalence Relation on the Class of Regular Languages. Part I. *International Journal of Open Information Technologies*. 2022;10(9):12-20. (In Russ., abstract in Eng.) EDN: WJFXEA

[19] Hromkovič J. Algorithmics for Hard Problems: Introduction to Combinatorial Optimization, Randomization, Approximation, and Heuristics. 2nd ed. Berlin: Springer; 2004. 538 p. https://doi.org/10.1007/978-3-662-05269-3

[20] Kameda T., Weiner P. On the State Minimization of Nondeterministic Finite Automata. *IEEE Transactions on Computers*. 1970;C-19(7):617-627. https://doi.org/10.1109/T-C.1970.222994

[21] Li L., Qiu D. On the State Minimization of Fuzzy Automata. *IEEE Transactions on Fuzzy Systems.* 2015;23(2):434-443. https://doi.org/10.1109/TFUZZ.2014.2315620

[22] Maarand H., Tamm H. Yet Another Canonical Nondeterministic Automaton. In: Han Y.S., Vaszil G. (eds.) Descriptional Complexity of Formal Systems. DCFS 2022. *Lecture Notes in Computer Science*. Vol. 13439. Cham: Springer; 2022. p. 184-196. https://doi.org/10.1007/978-3-031-13257-5_14

[23] Melnikov B., Melnikova A. A Polynomial Algorithm for Construction an Automaton for Checking the Equality of Infinite Iterations of Two Finite Languages. In: Silhavy R. (eds.) Artificial Intelligence Trends in Systems. CSOC 2022. *Lecture Notes in Networks and Systems*. Vol. 502. Cham: Springer; 2022. p. 521-530. https://doi.org/10.1007/978-3-031-09076-9_47

[24] Abramyan M.E. Computing the weight of subtasks in state minimization of nondeterministic finite automata by the branch and bound method. *University proceedings. Volga region. Physical and mathematical sciences.* 2021;(2):46-52. (In Russ., abstract in Eng.) https://doi.org/10.21685/2072-3040-2021-2-4

[25] Abramyan M., Melnikov B. A Program Study of the Union of Semilattices on the Set of Subsets of Grids of Waterloo Language. *Journal of Applied Mathematics and Physics*. 2023;11:1459-1470. https://doi.org/10.4236/jamp.2023.115095

*Submitted 08.09.2023; approved after reviewing 01.10.2023; accepted for publication 05.10.2023.*

About the authors:

**Mikhail E. Abramyan,** Associate Professor of the Institute of Mathematics, Mechanics, and Computer Science named after of I.I. Vorovich, Southern Federal University (105/42 Bolshaya Sadovaya St., Rostov-on-Don 344006, Russian Federation), Cand. Sci. (Phys.-Math.), Associate Professor, **ORCID: https://orcid.org/0000-0002-2802-6144,** mabr@sfedu.ru

**Boris F. Melnikov,** Professor of the Faculty of Computational Mathematics and Cybernetics, Shenzhen MSU – BIT University (1 Guojidaxueyuan St., Dayunxincheng, Longgang District, Shenzhen 517182, Guangdong Province, People's Republic of China), Dr. Sci. (Phys.-Math.), Professor, **ORCID: https://orcid.org/0000-0002-6765-680**0, bormel@mail.ru

*All authors have read and approved the final manuscript.*