Vol. 14, no. 1. 2018 ISSN 2411-1473 sitito.cs.msu.ru

Большие данные и приложения

Big Data and applications

УДК 004.021
DOI: 10.25559/SITITO.14.201801.101-110

SIMPLE HEURISTIC ALGORITHM FOR DYNAMIC VM REALLOCATION IN IAAS CLOUDS

Nikita A. Balashov[1], Alexandr V. Baranov[1], Ivan S. Kadochnikov[1], Vladimir V. Korenkov[1,2], Nikolay A. Kutovskiy[1], Igor S. Pelevanyuk[1]

[1] Joint Institute for Nuclear Research, Dubna, Russia

[2] Plekhanov Russian University of Economics, Moscow, Russia

Abstract

The rapid development of cloud technologies and its high prevalence in both commercial and academic areas have stimulated active research in the domain of optimal cloud resource management. One of the most active research directions is dynamic virtual machine (VM) placement optimization in clouds build on Infrastructure-as-a-Service model. This kind of research may pursue different goals with energy-aware optimization being the most common goal as it aims at a urgent problem of green cloud computing - reducing energy consumption by data centers. In this paper we

About the authors:

Nikita A. Balashov, software engineer, Laboratory of Information Technologies, Joint Institute for Nuclear Research (6 Joliot-Curie St., Dubna 141980, Moscow region, Russia); ORCID: http://orcid.org/0000-0002-3646-0522, balashov@jinr.ru

Alexandr V. Baranov, software engineer, Laboratory of Information Technologies, Joint Institute for Nuclear Research (6 Joliot-Curie St., Dubna 141980, Moscow region, Russia); ORCID: http://orcid.org/0000-0002-3568-4196, baranov@jinr.ru

Ivan S. Kadochnikov, software engineer, Laboratory of Information Technologies, Joint Institute for Nuclear Research (6 Joliot-Curie St., Dubna 141980, Moscow region, Russia); ORCID: http://orcid.org/0000-0001-8026-3190, kadivas@jinr.ru

Vladimir V. Korenkov, Doctor of Technical Sciences, Professor, Director of the Laboratory of Information Technologies, Joint Institute for Nuclear Research (6 Joliot-Curie St., Dubna, Moscow region, 141980, Russia); Professor, Plekhanov Russian University of Economics (36 Stremyanny lane, Moscow 117997 Russia); ORCID: http://orcid.org/0000-0002-2342-7862, korenkov@cv.jinr.ru

Nikolay A. Kutovskiy, Candidate of Physical and Mathematical Sciences, researcher, Laboratory of Information Technologies, Joint Institute for Nuclear Research (6 Joliot-Curie St., Dubna 141980, Moscow region, Russia); ORCID: http://orcid.org/0000-0002-2920-8775, kut@jinr.ru

Igor S. Pelevanyuk, software engineer, Laboratory of Information Technologies, Joint Institute for Nuclear Research (6 Joliot-Curie St., Dubna 141980, Moscow region, Russia); ORCID: http://orcid.org/0000-0002-4353-493X, pelevanyuk@jinr.ru

present a new heuristic algorithm of dynamic reallocation of VMs based on an approach presented in one of our previous works. In the algorithm we apply a 2-rank strategy to classify VMs and servers corresponding to the highly and lowly active VMs and solve four tasks: VM classification, host classification, forming a VM migration map and VMs migration. Dividing all of the VMs and servers into two classes we attempt to implement the possibility of risk reduction in case of hardware overloads under overcommitment conditions and to reduce the influence of the occurring overloads on the performance of the cloud VMs. Presented algorithm was developed based on the workload profile of the JINR cloud (a scientific private cloud) with the goal of maximizing its usage, but it can also be applied in both public and private commercial clouds to organize the simultaneous use of different SLA and QoS levels in the same cloud environment by giving each VM rank its own level of overcommitment.

Keywords

Cloud computing; virtualization; optimization; intelligent control; datacenters; VM consolidation.

ПРОСТОЙ ЭВРИСТИЧЕСКИЙ АЛГОРИТМ ДИНАМИЧЕСКОГО ПЕРЕРАСПРЕДЕЛЕНИЯ ВМ В IAAS ОБЛАКАХ

Н.А. Балашов[1], А.В. Баранов[1], И.С. Кадочников[1], В.В. Кореньков[1,2], Н.А. Кутовский[1], И.С. Пелеванюк[1]

[1] Объединенный институт ядерных исследований, г. Дубна, Россия

[2] Российский экономический университет им. Г.В. Плеханова, г. Москва, Россия

Аннотация

Одним из наиболее популярных видов использования облачных технологий как в науке в целом, так и в физике высоких энергий в частности, является построение облачных вычислительных центров, основанных на модели Инфраструктура как услуга (англ. Infrastructure-as-a-Service, IaaS). Условия предоставления пользователям виртуальных машин (ВМ) могут различаться, но в большинстве случаев можно диагностировать проблему неэффективного использования ресурсов. В данной статье авторами представлен разработанный метод повышения эффективности использования облачных ресурсов, в основе которого лежит идея динамического перераспределения ВМ на физическом оборудовании. Предлагаемый метод разработан на основании результатов проведенного авторами статьи анализа потребления облачных ресурсов в научной сфере на примере облачной инфраструктуры Объединенного института ядерных исследований (ОИЯИ). В данной статье авторы рассматривают вариант реализации предложенного метода - эвристический алгоритм динамического перераспределения ВМ на основе двухранговой стратегии. Задачей предлагаемого алгоритма является минимизация количества простаивающих ресурсов облачной инфраструктуры и одновременно минимизация влияния перегрузок. Алгоритмические расчеты основаны на данных о потреблении оперативной памяти и ресурсов процессора как виртуальными машинами, так и каждым сервером, составляющим облачную инфраструктуру. Рассматриваемый алгоритм включает следующие этапы: классификацию ВМ и серверов согласно заданным параметрам, составление карты миграции и, непосредственно, миграция ВМ согласно составленной карте. Представленный алгоритм может найти применение не только в частных облачных инфраструктурах, но также и в коммерческой среде: проранжировав облачные ресурсы и задав каждому рангу свою степень overcommit, можно организовать работу в единой облачной среде с различными Service-Level Agreements (SLA) и Quality of Service (QoS).

Vol. 14, no. 1. 2018 ISSN 2411-1473 sitito.cs.msu.ru

Ключевые слова

Облачные вычисления; виртуализация; оптимизация; интеллектуальное управление; ЦОД; консолидация ВМ.

1. Introduction

The rapid pace of cloud technologies development and their ubiquitous use in commercial area [1] could not but lead to their introduction in the field of science and education. One of the most popular cloud technologies use-cases in the science in general, and in high energy physics in particular [2-6], is building cloud datacenters based on Infrastructure as a Service (IaaS) model [7]. In such systems an end-user gets a pool of virtual machines (VM) with the ability to install any required additional software by himself. In such clouds user VMs are automatically allocated to physical servers (which form a cloud) by a built-in VM scheduler. Thus, cloud users get a universal computing tool with the ability to adapt it to their current personal needs by themselves.

As is known, pursuing universality of a system often leads to various performance issues and one of the most common problems is the falling efficiency in the utilization of physical equipment. In private scientific cloud infrastructures this problem can worsen due to internal organization of the provided service. In particular, unlike the commercial clouds in which a service is provided on the pay-as-you-go model, i.e. service users are charged based on usage, in scientific private cloud infrastructure the service is often provided at no cost. Evidently, in the absence of any restrictions this approach is characterized by abuse of allocated resources by users, inevitably leading to significant reduction of the provider's equipment usage efficiency. This problem is often solved by signing Service Level Agreement (SLA) which, for example, may contain a clause on the automatic removement of user VMs when not used. There are other popular approaches to the problem, e.g. methods of artificial performance reduction by over-allocating virtual resources. This technique is referred to as "overcommitment" and it leads to increased load of hardware, but may also lead to the overloading of physical equipment and, consequently, to the cloud VMs performance degradation.

In one of our previous works [8] we analyzed cloud resources usage in scientific area by the example of the Joint Institute of Nuclear Research (JINR) cloud infrastructure and suggested a new method of the efficiency improvement of the cloud resources utilization based on dynamic reallocation of VMs across the cloud hardware. A software framework, which simplifies development of various schemes of dynamic reallocation of VMs in clouds, was also introduced by the authors [9]. In this paper we review implementation of a version of the earlier suggested method - heuristic algorithm of dynamic VM re-allocation based on 2-rank strategy. The algorithm is intended for minimizing the number of idle resources in cloud infrastructure and at the same time minimizing the consequences of the hardware overload.

The rest of the paper is organized as follows. Section 2 discusses related work, followed by the model of the cloud loads discussed in Section 3. The 2-rank algorithm for dynamic reallocation of VMs is presented in Section 4. Section 5 concludes the paper with a summary and future research direction.

2. Related work

Traditional approach to increase the efficiency of resource usage in cloud computing environments and to reduce the capital expenditure on the cloud infrastructure (built on IaaS model) maintenance is to consolidate VMs using overcommitment technologies, thus forming a more dense VM distribution across the hardware in the cloud. Some platforms (e.g., OpenStack), which cloud computing environments are based on, are able to perform such consolidation of VMs using simple algorithms, while other platforms lack such functionality (e.g., OpenNebula). Nevertheless, overcommitting resources inevitably leads to the reduction of the Quality of Service (QoS) since it is never possible to accurately predict the future load in advance and there may be situations when real workloads will exceed the maximum of physically possible because of the abrupt growth of the load generated by the VMs. A large number of publications are devoted to the research of method for VM scheduling under overcommitment conditions minimizing QoS

degradation, in particular one of the recent research [10], in which authors consider VM scheduling in the IaaS-clouds, taking into account overcommitment, as a bin-packing problem [11] with chance constraints. However, even more number of research works is focused on research of methods for dynamic optimization of virtual resources distribution in cloud environments [12-19], and there is also a number of publications describing software implementations of such methods [20-22]. There are also such methods pursuing more exotic goals, i.e. thermal-aware resource management designed to improve the efficiency of heat extraction [23]. In essence all of these methods are focused on optimizing virtual resources distribution on physical equipment organized in a way to maximize VM density among servers while minimizing degradation of the QoS. Implementation of such a class of methods is possible due to sufficiently advanced VM live-migration technology [24], i.e. transferring a VM from one server to another one without disconnecting the client, with close to zero downtime and restoring the full state of the VM running processes after migration. In all of the abovementioned works methods of dynamic optimization of the VM distribution are considered in context of solving of one of the green computing urgent problems: decreasing energy consumption by data centers [25]. To treat this problem such methods are used to maximize the workloads density to decrease the number of active servers: idle servers containing no VMs are then powered off or put into power-saving mode, thus reducing total power usage.

In this paper we describe a new algorithm of dynamic reallocation of virtual resources, based on the analysis of the workloads profile in the JINR Cloud [22]. Proposed algorithm is an attempt to implement possibility of risk reduction in case of a hardware overload and to reduce the influence of the occurring overloads on the performance of the cloud VMs. The proposed algorithm relies on a custom monitoring and performance data collection system, since not every cloud platform provides such a suitable system [26-27].

3. Cloud Load Model

Four key characteristics of the VM performance are memory size, central processing unit (CPU) resources, number and speed of I/O disk operations and volume of network traffic. If a cloud system has low network traffic and a negligibly small number of disk operations (which is seen in JINR Cloud), then memory size and CPU resources are the only significant performance metrics. In this paper we consider only the memory size and the number and the usage of CPU. The proposed algorithm is aimed at maximizing memory and CPU usage of the cloud servers.

The total usage of CPU resources C on a single server we define as follows:

$$\mathrm{C} = \sum_{i=1}^{k} c_j + c_h$$

where k is the number of VMs placed on server, c_j is the CPU usage of the VM j, and c_h is the CPU usage generated by the server itself, which is actually an overhead spent on the server functioning.

Then the total CPU load of the cloud as a whole L_c can be expressed in the form:

$$L_c = \sum_{i=1}^{n} C_i$$

where $n \in N$ is the total number of servers included in the cloud.

The memory consumption M is defined in the same way:

$$M = \sum_{j=1}^{k} m_j + m_h \; L_m = \sum_{i=1}^{n} M_i$$

where m_j is the memory size used by VM j, and m_h is the memory size - an overhead spent on the server functioning.

4. Algorithm description

To implement the algorithm the following input data are needed:

Memory and CPU usage by VMs;

Memory and CPU usage by servers.

The following parameters also have to be defined by the cloud system administrator based on risks which the cloud provider is ready to take:

P - parameter used to determine the rank R of the VM, $P \in (0,1)$;

t - time for which the average VM load is computed;

Factors O_c and O_m determine valid values of the CPU and memory overcommit levels, defined for each of two ranks. Value of these parameters less than 2 mean that servers of this rank will always be underloaded, values of 1 mean that maximum load may reach 100 %, the greater the value above 1, the higher the risk of the hardware overload.

The proposed algorithm is as follows:

The algorithm assumes that all VMs in the cloud can be divided into two classes: highly active VMs, which are assigned the rank 1, and low-active VMs, which get the rank 0. All the cloud servers also get one of the ranks and the VMs with the assigned rank are placed on the servers with the same corresponding rank. In each iteration of the algorithm ranks of VMs are recomputed in accordance with the pre-defined rules and all the VMs which changed the rank are migrated to the other cluster. When necessary the ranks of the servers that have no VMs left are deleted forming cluster of the unranked servers. In general, the cycle of the algorithm consists of the following stages:

VM classification.
Server classification.
Computing VM migration map
VM migration.
Deletion of the rank of the idle servers.

In the following sections we review in more detail some of these stages of the described algorithm.

4.1 VM classification

To classify the cloud VMs (Algorithm 1) we consider the following function:

$$R = \begin{cases} 1 if V_c^2 + V_m^2 > P^2 \\ 0 if V_c^2 + V_m^2 \leq P^2 \end{cases},$$

where V_c and V_m correspond to the average CPU and memory usage for the time t normalized by 1, and P is the predefined variable used to define the rank as described in the previous section. Time t and value of P are the parameters and are defined by the system administrator. For instance, if $P = 0.5$, then the VM rank distribution in the JINR cloud for the randomly chosen time period t equal to one month will take form as illustrated on Fig. 1.

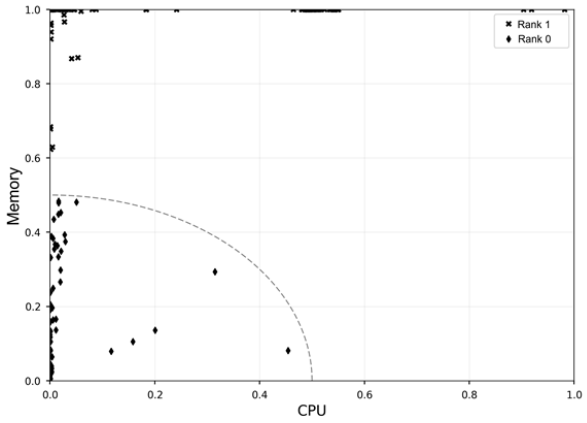

Figure 1. *Example of the VM classification in the JINR cloud. The X-axis shows the real CPU usage of the VM as a fraction of the total allocated CPU for the VM, the Y-axis shows the real memory usage by the VM as a fraction of the total allocated memory for the VM*

Algorithm 1. Classification of VMs

Input: *VMList, P, t*
1: **foreach** *VM* **in** *VMList* **do**
2: | **if** *VM* has history for the period *t* **do**
3: | | **if (***VM*.cpuUsage^2 + *VM*.memUsage^2) < *P*^2 **do**
4: | | | Set rank of the *VM* equal 1
5: | | **else**
6: | | | Set rank of the *VM* equal 0
7: **return** *VMs*

4.2 Server classification

In the described algorithm each rank of the VM corresponds to a server rank, therefore there should exist a minimum of two server ranks. However, new VMs, that has no statistics for the defined time period $[t_c - t; t_c]$, where t_c is the current time, cannot be classified at this stage. To address the issue we use a number of unclassified servers, which will host all new incoming VMs scheduling them in accordance with the built-in algorithms of the used cloud platform.

On the initial stage of the algorithm any server may contain VMs of two classes simultaneously. To reduce the number of VM migrations on the following stage a simple algorithm can be used.

Choose the rank which is assigned to the greater number of VMs in the cloud.

Sort the servers by the hosted number of VMs of the chosen rank in descending order.

The servers are added to the list (cluster) of the chosen rank servers one by one until it gains the capacity suitable to host all of the VMs of the corresponding rank.

Repeat step 3 to form a list of servers of the other rank.

The rest of the servers are left unclassified with no rank. Thus, after the system initialization there will be three groups (clusters): server cluster of rank 1, cluster of rank 0 and the cluster composed of servers with no rank assigned (Algorithm 2). On the current stage some part of the servers, evidently, will host VMs of both classes simultaneously - the system will self-organize on the next iterations of the resources reallocation cycle.

Algorithm 2. Initial classification of Hosts

Input: *HostList*, *ClusterList*
Input: *initialRank*, the rank that the maximum number of VMs has
1: **if** *initialRank* == 0 **do**
2: | Sort *ClusterList* ascending
3: **else**
4: | Sort *ClusterList* descending
5: Sort *HostList* by number of VMs having *initialRank* descending
6: **foreach** *cluster* **in** *ClusterList* **do**
7: | **foreach** *host* **in** *HostList* **do**
8: | | **if** (*host*.memSize***cluster*.Om + *cluster*.memSize) < *cluster*.requiredMemSize **and**
9: | | | (*host*.cpuSize***cluster*.Oc + *cluster*.cpuSize) < *cluster*.requiredCpuSize **do**
10: | | | | Add *host* to the *cluster*
11: | | | | Remove *host* from the *HostList*
12: | | **else**
13: | | | **break**
14: **return** *ClusterList*

4.3 Resources reallocation cycle

The resources reallocation cycle occurs at regular intervals *t* and can be split into two subtasks:

- VM classification;
- VM migration.

The process of VM classification has been described in above section, but the VM migration can be considered as a bin packing problem with elements (VMs) of different sizes and containers (servers) also of different sizes which is known to be NP-hard. To address this problem we consider the following simple algorithm for forming the VM migration map which is a modification of the well-known Best-Fit Decreasing (BFD) algorithm:

Form a list of VMs of the same rank which are to be migrated, i.e. those VMs the rank of which doesn't match the rank of the hosting them servers.

Sort the list of VMs by memory size in decreasing order, then sort groups of VMs of the same memory by CPU resources in decreasing order.

Sort unclassified servers and the servers of the corresponding rank in the same way but in increasing order.

Get the first VM in the list and compare its size to the size of the first host in the list of ranked servers: if it fits the server, this pair is added to the migration map, the VM is removed from the list and the size of the server is reduced. Otherwise, the procedure is repeated for the next server in the list.

If no suitable server was found, the first server from the unclassified list is added to the list of ranked servers and with the current VM it is added to the migration map, the VM then is removed from the list. If the server doesn't fit, the next server in the list considered. If there is no suitable server found, the VM is skipped.

Repeat 3-5 for the rest of the VMs.

As a result the migration map and the list of VMs that don't fit any server at the moment are formed.

It can be seen from this algorithm that packing starts with large VMs first: since the smaller the item (VM) the higher the probability to find a suitable container (server) of enough capacity, and because the free capacity is reduced with every match, the probability to successfully find the next match is also reduced with each iteration. Sorting VMs and servers as described above reduces the probability of not finding the suitable server in the list.

This algorithm is repeated twice - once for each of two classes of objects. The flowchart illustrating the algorithm is presented on Fig. 2. The algorithm results in a VM migration map and a list of VMs which can't be reallocated at the moment. The migration map is a list in which an element is a pair VM-server showing on which server the VM should be placed.

It is also can be seen, that when new servers are added to the system they will automatically be classified during on the resources reallocation cycle according to current needs.

Migration of the VMs is processed one by one using the migration map to minimize the influence on the total system performance. After all migration are finished ranks of all of the servers, which don't host any VMs anymore but still have a rank, are deleted and the servers are moved to the list of unclassified servers.

5. Conclusion

In this paper we gave a review of an algorithm for dynamic reallocation of VMs in a cloud environment based on 2-rank strategy - one of the possible implementations of the proposed earlier method of dynamic resources reallocation [8]. The algorithm is aimed, in the first place, to be used in private cloud infrastructures with workload profile similar of the JINR cloud one. Presented algorithm may be used not only in private cloud infrastructure, but also in commercial area: ranking cloud resources and giving each rank its own level of overcommitment it is possible to organize the simultaneous use of different SLA and QoS levels in the same cloud environment.

Vol. 14, no. 1. 2018 ISSN 2411-1473 sitito.cs.msu.ru

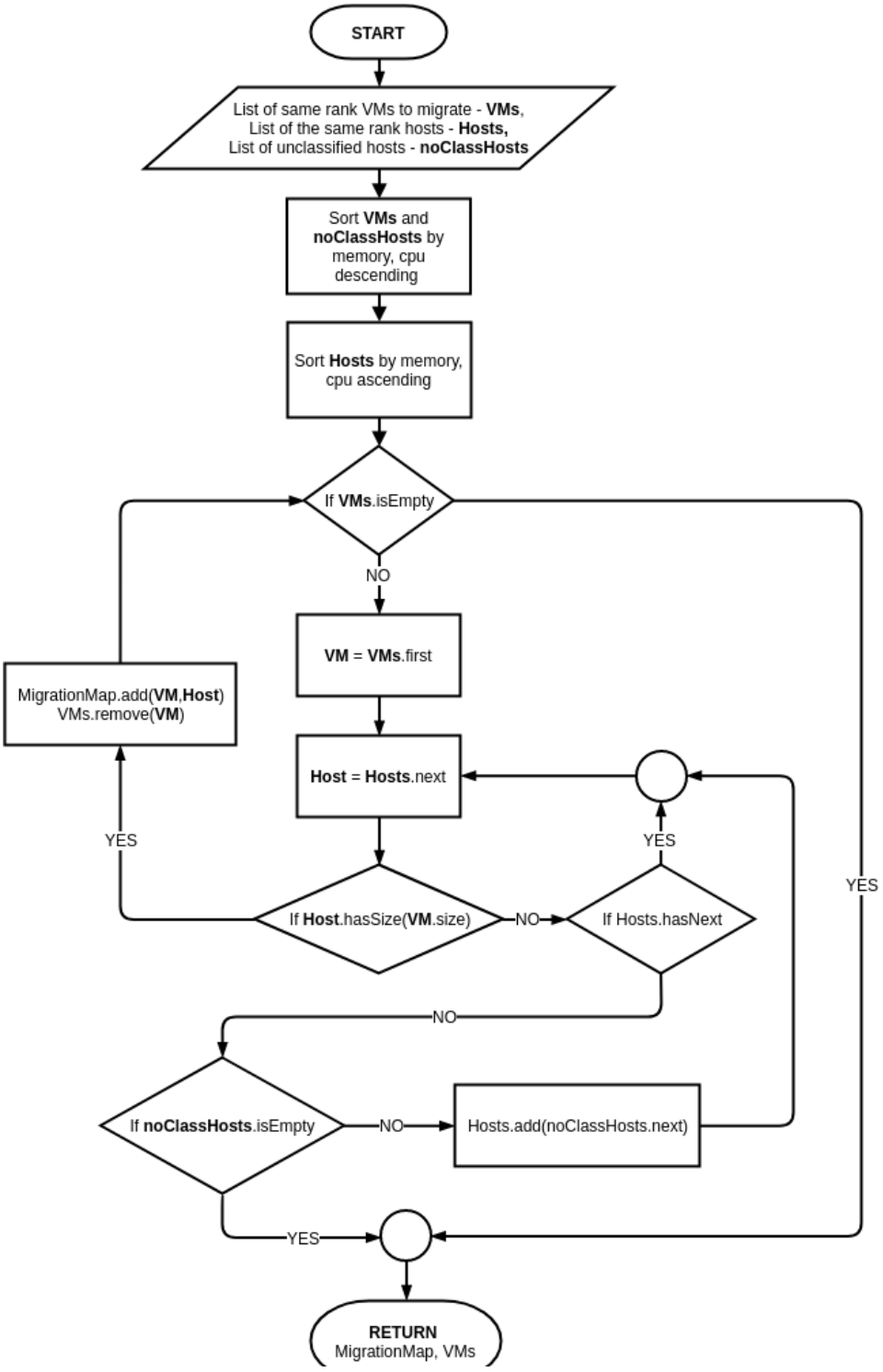

Figure 2. *Flowchart showing the algorithm of forming the migration map for one of the ranks*

To make implementation of this algorithm and others possible the authors developed the software framework [9] which was put into operation in the JINR cloud. In our future research we plan to

analyze the gathered statistical data on the algorithm functioning to get an experimental assessment of its efficiency. We also plan to investigate the ways of integrating the system with batch systems [22] to load the unclassified idle servers with short-living preemptable batch jobs, thus minimizing the time the servers are idle.

Acknowledgements

The work was supported by the RFBR project 15-29-0727.

REFERENCES

[1] Wiens K. Cloud Computing Trends: 2018 State of the Cloud Survey // RightScale Cloud Management Blog, 2018. Available at: https://www.rightscale.com/blog/cloud-industry-insights/cloud-computing-trends-2018-state-cloud-survey (accessed 16.01.18).
[2] Meinhard H. Virtualization, clouds and IaaS at CERN. *VTDC '12 Proceedings of the 6th international workshop on Virtualization Technologies in Distributed Computing*, ACM New York, NY, USA. 2012. p. 27-28.
[3] Timm S. et al. Cloud Services for the Fermilab Scientific Stakeholders. *Journal of Physics: Conference Series*. 2015; 664(2). DOI: https://doi.org/10.1007/s41781-017-0001-9
[4] Timm S. et al. Virtual machine provisioning, code management, and data movement design for the Fermilab HEPCloud Facility. *Journal of Physics: Conference Series*. 2017. Vol. 898, Track 3: Distributed Computing, id. 052041. DOI: https://doi.org/10.1088/1742-6596/898/5/052041
[5] Baranov A.V., Balashov N.A., Kutovskiy N.A., Semenov R.N. JINR cloud infrastructure evolution. *Physics of Particles and Nuclei Letters.* 2016; 13(5):672–675. DOI: https://doi.org/10.1134/S154747711605006X
[6] Amoroso A. et al., A modular (almost) automatic set-up for elastic multi-tenants cloud (micro)infrastructures. *Journal of Physics: Conference Series*. 2017; 898(8). DOI: https://doi.org/10.1088/1742-6596/898/8/082031
[7] Mell P., Grance T. The NIST definition of cloud computing. *Communications of the ACM*. 2010; 53(6):50.
[8] Balashov N., Baranov A., Korenkov V. Optimization of over-provisioned clouds. *Physics of Particles and Nuclei Letters*. 2016; 13(5):609-612. DOI: https://doi.org/10.1134/S154747711605006X
[9] Balashov N.A. Baranov A.V. Kadochnikov I.S. et al. Smart cloud scheduler. *CEUR Workshop Proceedings*. 2016; 1787:114-118. Available at: http://ceur-ws.org/Vol-1787/114-118-paper-18.pdf (accessed 16.01.18).
[10] Cohen M.C., Keller P.W., Mirrokni V., Zadimoghaddam M. Overcommitment in Cloud Services – Bin packing with Chance Constraints// Computing Research Repository, 2017. Vol. abs/1705.09335. Available at: https://arxiv.org/abs/1705.09335 (accessed 16.01.18).
[11] Martello S., Toth P. Bin-packing problem // Knapsack Problems: Algorithms and Computer Implementations. Chichester, UK: John Wiley and Sons, 1990.
[12] Ashraf A., Porres I. Multi-objective dynamic virtual machine consolidation in the cloud using ant colony system. *International Journal of Parallel, Emergent and Distributed Systems*. 2017; 33(1):103-120. DOI: https://doi.org/10.1080/17445760.2017.1278601
[13] Beloglazov A., Buyya R. Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in cloud data centers. *Concurrency and Computation: Practice and Experience (CCPE)*. 2012; 24(13):1397–1420. DOI: https://doi.org/10.1002/cpe.1867
[14] Corradi A., Fanelli M., Foschini L. VM consolidation: A real case based on OpenStack Cloud. *Future Generation Computer Systems*. 2014; 32:118-127. DOI: https://doi.org/10.1016/j.future.2012.05.012
[15] Zahedi Fard S.Y., Ahmadi M.R., Adabi S.J. A dynamic VM consolidation technique for QoS and energy consumption in cloud environment. *The Journal of Supercomputing*. 2017; 73(10):4347–4368. DOI: https://doi.org/10.1007/s11227-017-2016-8
[16] Dabbagh M., Hamdaoui B., Guizani M., Rayes A. An Energy-Efficient VM Prediction and Migration Framework for Overcommitted Clouds. *IEEE Transactions on Cloud Computing*. 2016; 99:1-1. DOI: https://doi.org/10.1109/TCC.2016.2564403
[17] Mosa A., Paton N.W. Optimizing virtual machine placement for energy and SLA in clouds using utility functions. *Journal of Cloud Computing: Advances, Systems and Applications*. 2016; 5:1-17. DOI: https://doi.org/10.1186/s13677-016-0067-7
[18] Hwang I., Pedram M. Hierarchical, Portfolio Theory-Based Virtual Machine Consolidation in a Compute Cloud. *IEEE Transactions on Services Computing*. 2018; 11(1):63-77. DOI: https://doi.org/10.1109/TSC.2016.2531672
[19] Mastroianni M. Meo, Papuzzo G. Probabilistic Consolidation of Virtual Machines in Self-Organizing Cloud Data Centers. *IEEE Transactions on Cloud Computing*. 2013; 1(2):215-228. DOI: https://doi.org/10.1109/TCC.2013.17
[20] Beloglazov A., Buyya R. OpenStack Neat: A Framework for Dynamic and Energy-Efficient Consolidation of Virtual Machines in OpenStack Clouds. *Concurrency and Computation: Practice and Experience (CCPE)*. 2014; 27(5):1310-1333. DOI: https://doi.org/10.1002/cpe.3314
[21] Feller E., Rilling L., Morin C. Snooze: A scalable and autonomic virtual machine management framework for private Clouds. *Proceedings of the 12th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid)*, 2012. p. 482–489. DOI: https://doi.org/10.1109/CCGrid.2012.71
[22] Balashov N.A., Baranov A.V., Kadochnikov I.S., Korenkov V.V., Kutovskiy N.A., Nechaevskiy A.V., Pelevanyuk I.S. Software complex for intelligent scheduling and adaptive self-organization of virtual computing resources based in LIT JINR cloud center. *Izvestiya SFedU. Engineering Sciences*. 2016; 12(185):92-103. (In Russian) DOI: https://doi.org/10.18522/2311-3103-2016-12-92103
[23] Lee E.K., Viswanathan H., Pompili D. Proactive Thermal-Aware Resource Management in Virtualized HPC Cloud Datacenters. *IEEE Transactions on Cloud Computing*. 2017; 5(2):234-248. DOI: https://doi.org/10.1109/TCC.2015.2474368
[24] Cloudhary A., Govil M.C., Singh G., Awastkhi L.K., Pilli E.S., Kapil D. A critical survey of live virtual machine migration techniques. *Journal of Cloud Computing*. 2017; 6:23. DOI: https://doi.org/10.1186/s13677-017-0092-1
[25] Radu L.D. Green Cloud Computing: A Literature Survey. *Symmetry*. 2017; 9(12):295. DOI: https://doi.org/10.3390/sym9120295

[26] Baginyan A. et al. Multi-level monitoring system for multifunctional information and computing complex at JINR. *CEUR Workshop proceedings*. 2017; 2023:226-233.
[27] Kadochnikov I.S., Balashov N.A., Baranov A.V., Pelevanyuk I.S., Kutovskiy N.A., Korenkov V.V., Nechaevskiy A.V. Evaluation of monitoring systems for metric collection in intelligent cloud scheduling. *CEUR Workshop proceedings*. 2016; 1787:279-283.

Submitted 16.01.2018; Revised 10.02.2018; Published 30.03.2018.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

[1] *Wiens K.* Cloud Computing Trends: 2018 State of the Cloud Survey // RightScale Cloud Management Blog, 2018. URL: https://www.rightscale.com/blog/cloud-industry-insights/cloud-computing-trends-2018-state-cloud-survey (дата обращения: 16.01.18).
[2] Virtualization, clouds and IaaS at CERN / H. Meinhard // VTDC '12 Proceedings of the 6th international workshop on Virtualization Technologies in Distributed Computing, ACM New York, NY, USA, 2012. Pp. 27-28.
[3] *Timm S.* et al. Cloud Services for the Fermilab Scientific Stakeholders // Journal of Physics: Conference Series. 2015. Vol. 664, issue 2. DOI: https://doi.org/10.1007/s41781-017-0001-9
[4] *Timm S.* et al. Virtual machine provisioning, code management, and data movement design for the Fermilab HEPCloud Facility // Journal of Physics: Conference Series. 2017. Vol. 898, Track 3: Distributed Computing, id. 052041. DOI: https://doi.org/10.1088/1742-6596/898/5/052041
[5] *Baranov A.V., Balashov N.A., Kutovskiy N.A., Semenov R.N.* JINR cloud infrastructure evolution // Physics of Particles and Nuclei Letters. 2016. Vol. 13, issue 5. Pp. 672–675. DOI: https://doi.org/10.1134/S154747711605006X
[6] *Amoroso A.* et al., A modular (almost) automatic set-up for elastic multi-tenants cloud (micro)infrastructures // Journal of Physics: Conference Series. 2017. Vol. 898, issue 8. DOI: https://doi.org/10.1088/1742-6596/898/8/082031
[7] *Mell P., Grance T.* The NIST definition of cloud computing // Communications of the ACM. 2010. Vol. 53, issue 6. Pp. 50.
[8] *Balashov N., Baranov A., Korenkov V.* Optimization of over-provisioned clouds // Physics of Particles and Nuclei Letters. 2016. Vol. 13, issue 5. Pp. 609-612. DOI: https://doi.org/10.1134/S154747711605006X
[9] *Balashov N.A. Baranov A.V. Kadochnikov I.S.* et al. Smart cloud scheduler // CEUR Workshop Proceedings. 2016. Vol. 1787. Pp. 114-118. URL: http://ceur-ws.org/Vol-1787/114-118-paper-18.pdf (дата обращения: 16.01.18).
[10] *Cohen M.C., Keller P.W., Mirrokni V., Zadimoghaddam M.* Overcommitment in Cloud Services - Bin packing with Chance Constraints // Computing Research Repository, 2017. Vol. abs/1705.09335. URL: https://arxiv.org/abs/1705.09335 (дата обращения: 16.01.18).
[11] *Martello S., Toth P.* Bin-packing problem // Knapsack Problems: Algorithms and Computer Implementations. Chichester, UK: John Wiley and Sons, 1990.
[12] *Ashraf A., Porres I.* Multi-objective dynamic virtual machine consolidation in the cloud using ant colony system // International Journal of Parallel, Emergent and Distributed Systems. 2017. Vol. 33, issue 1. Pp. 103-120. DOI: https://doi.org/10.1080/17445760.2017.1278601
[13] *Beloglazov A., Buyya R.* Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in cloud data centers // Concurrency and Computation: Practice and Experience (CCPE). 2012. Vol. 24, issue 13. Pp. 1397–1420. DOI: https://doi.org/10.1002/cpe.1867
[14] *Corradi A., Fanelli M., Foschini L.* VM consolidation: A real case based on OpenStack Cloud // Future Generation Computer Systems. 2014. Vol. 32. Pp. 118-127. DOI: https://doi.org/10.1016/j.future.2012.05.012
[15] *Zahedi Fard S.Y., Ahmadi M.R., Adabi S.J.* A dynamic VM consolidation technique for QoS and energy consumption in cloud environment // The Journal of Supercomputing. 2017. Vol. 73, issue 10. Pp. 4347–4368. DOI: https://doi.org/10.1007/s11227-017-2016-8
[16] *Dabbagh M., Hamdaoui B., Guizani M., Rayes A.* An Energy-Efficient VM Prediction and Migration Framework for Overcommitted Clouds // IEEE Transactions on Cloud Computing. 2016. Vol. 99. Pp. 1-1. DOI: https://doi.org/10.1109/TCC.2016.2564403
[17] *Mosa A., Paton N.W.* Optimizing virtual machine placement for energy and SLA in clouds using utility functions // Journal of Cloud Computing: Advances, Systems and Applications. 2016. Vol. 5. Pp. 1-17. DOI: https://doi.org/10.1186/s13677-016-0067-7
[18] *Hwang I., Pedram M.* Hierarchical, Portfolio Theory-Based Virtual Machine Consolidation in a Compute Cloud // IEEE Transactions on Services Computing. 2018. Vol. 11, issue 1. Pp. 63-77. DOI: https://doi.org/10.1109/TSC.2016.2531672
[19] *Mastroianni M. Meo, Papuzzo G.* Probabilistic Consolidation of Virtual Machines in Self-Organizing Cloud Data Centers // IEEE Transactions on Cloud Computing. 2013. Vol. 1, issue 2. Pp. 215-228. DOI: https://doi.org/10.1109/TCC.2013.17
[20] *Beloglazov A., Buyya R.* OpenStack Neat: A Framework for Dynamic and Energy-Efficient Consolidation of Virtual Machines in OpenStack Clouds // Concurrency and Computation: Practice and Experience (CCPE). 2014. Vol. 27, issue 5. Pp. 1310-1333. DOI: https://doi.org/10.1002/cpe.3314
[21] *Feller E., Rilling L., Morin C.* Snooze: A scalable and autonomic virtual machine management framework for private Clouds // Proceedings of the 12th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid), 2012. Pp. 482–489. DOI: https://doi.org/10.1109/CCGrid.2012.71
[22] *Балашов Н.А., Баранов А.В., Кадочников И.С., Кореньков В.В., Кутовский Н.А., Нечаевский А.В., Пелеванюк И.С.* Программный комплекс интеллектуального диспетчирования и адаптивной самоорганизации виртуальных вычислительных ресурсов на базе облачного центра ЛИТ ОИЯИ // Известия ЮФУ. Технические науки. 2016. № 12(185). С. 92-103. DOI: https://doi.org/10.18522/2311-3103-2016-12-92103
[23] *Lee E.K., Viswanathan H., Pompili D.* Proactive Thermal-Aware Resource Management in Virtualized HPC Cloud Datacenters // IEEE

Transactions on Cloud Computing. 2017. Vol. 5, issue 2. Pp. 234-248. DOI: https://doi.org/10.1109/TCC.2015.2474368

[24] *Cloudhary A., Govil M.C., Singh G., Awastkhi L.K., Pilli E.S., Kapil D.* A critical survey of live virtual machine migration techniques // Journal of Cloud Computing. 2017. Vol. 6, issue 23. Pp. 1-41. DOI: https://doi.org/10.1186/s13677-017-0092-1

[25] *Radu L.D.* Green Cloud Computing: A Literature Survey // Symmetry. 2017. Vol. 9, issue 12. Pp. 295. DOI: https://doi.org/10.3390/sym9120295

[26] *Baginyan A.* et al. Multi-level monitoring system for multifunctional information and computing complex at JINR // CEUR Workshop proceedings. 2017. Vol. 2023. Pp. 226-233.

[27] *Kadochnikov I.S., Balashov N.A., Baranov A.V., Pelevanyuk I.S., Kutovskiy N.A., Korenkov V.V., Nechaevskiy A.V.* Evaluation of monitoring systems for metric collection in intelligent cloud scheduling // CEUR Workshop proceedings. 2016. Vol. 1787. Pp. 279-283.

Поступила 16.01.2018; принята к публикации 10.02.2018; опубликована онлайн 30.03.2018.

Об авторах:

Балашов Никита Александрович, инженер-программист 1 категории, Лаборатория информационных технологий, Объединенный институт ядерных исследований (141980, Россия, Московская область, г. Дубна, ул. Жолио-Кюри, д. 6); ORCID: http://orcid.org/0000-0002-3646-0522, balashov@jinr.ru

Баранов Александр Владимирович, инженер-программист 1 категории, Лаборатория информационных технологий, Объединенный институт ядерных исследований (141980, Россия, Московская область, г. Дубна, ул. Жолио-Кюри, д. 6); ORCID: http://orcid.org/0000-0002-3568-4196, baranov@jinr.ru

Кадочников Иван Сергеевич, инженер-программист, Лаборатория информационных технологий, Объединенный институт ядерных исследований (141980, Россия, Московская область, г. Дубна, ул. Жолио-Кюри, д. 6); ORCID: http://orcid.org/0000-0001-8026-3190, kadivas@jinr.ru

Кореньков Владимир Васильевич, доктор технических наук, профессор, директор Лаборатории информационных технологий, Объединенный институт ядерных исследований (141980, Россия, Московская область, г. Дубна, ул. Жолио-Кюри, д. 6); профессор, Российский экономический университет им. Г.В. Плеханова (117997, Россия, г. Москва, Стремянный пер., д. 36); ORCID: http://orcid.org/0000-0002-2342-7862, korenkov@cv.jinr.ru

Кутовский Николай Александрович, кандидат физико-математических наук, научный сотрудник, Лаборатория информационных технологий, Объединенный институт ядерных исследований (141980, Россия, Московская область, г. Дубна, ул. Жолио-Кюри, д. 6); ORCID: http://orcid.org/0000-0002-2920-8775, kut@jinr.ru

Пелеванюк Игорь Станиславович, инженер-программист, Лаборатория информационных технологий, Объединенный институт ядерных исследований (141980, Россия, Московская область, г. Дубна, ул. Жолио-Кюри, д. 6); ORCID: http://orcid.org/0000-0002-4353-493X, pelevanyuk@jinr.ru