УДК 004.75
DOI: 10.26102/2310-6018/2022.37.2.030

Methods and models of resource allocation service in load balancing clusters for data centers

V.P. Mochalov, N.Y. Bratchenko, G.I. Linets, I.S. Palkanov

North-Caucasus Federal University, Stavropol, Russian Federation
ilya0693@yandex.ru

Abstract. The object of the research is computing clusters of cloud data centers, containing many servers, data storage systems, an input-output system interconnected by a communication network. The goal of this research is to develop methods and models for improving the performance of a data center cluster by reducing the processing time of service requests as well as reducing equipment costs due to the efficient allocation of its resources. Therefore, it is necessary to implement optimization algorithms for placing virtual machines (VMs) on physical servers in real time based on load balancing. The proposed method of resource allocation is based on an iterative greedy algorithm and a limited search procedure. Reduction in the computation time is achieved by introducing restrictions on the permissible search depth. The paper puts forward a mathematical model of resource allocation, built using the Erlang model in the form of a multi-line m-node queuing system (QS) of the $M|M|m|n$ type with an n-seat buffer, which makes it possible to determine the main indicators of service request quality in the form of QS parameters. The efficiency of this approach was tested on a simulation model built on the basis of the system functioning statistical analysis. Its experimental study was also carried out.

Keywords: computing clusters, virtual machines, physical servers, resource allocation model, heuristic algorithms, model experiment.

For citation: Mochalov V.P., Bratchenko N.Yu., Linets G.I., Palkanov I.S. Methods and models of resource allocation in load balancing clusters. *Modeling, Optimization and Information Technology.* 2022;10(2). Available from: https://moitvivt.ru/ru/journal/pdf?id=1188 DOI: 10.26102/2310-6018/2022.37.2.030

Методы и модели сервиса распределения ресурсов в кластерах с балансировкой нагрузки центров обработки данных

В.П. Мочалов, Н.Ю. Братченко, Г.И. Линец, И.С. Палканов

Северо-Кавказский федеральный университет,
Ставрополь, Российская Федерация
ilya0693@yandex.ru

Резюме. Объектом исследования являются вычислительные кластеры облачных центров обработки данных (ЦОД), содержащие множество серверов, систем хранения данных, систему ввода-вывода связанных между собой коммуникационной сетью. Целью работы является разработка методов и моделей повышения производительности кластера ЦОД путем уменьшения времени обработки запросов на обслуживание, а также уменьшения затрат на оборудование за счет эффективного распределения его ресурсов. Это вызывает необходимость реализации оптимизационных алгоритмов размещения виртуальных машин (ВМ) на физических серверах в реальном времени на основе балансировки нагрузки. В основу предложенного метода распределения ресурсов положен итерационный жадный алгоритм и процедура ограниченного перебора. Сокращение времени вычислений достигается при этом путем введения ограничений на допустимую глубину перебора. В работе предложена математическая модель распределения ресурсов, построенная на основе модели Эрланга в виде многолинейной m-узловой системы массового обслуживания (СМО) типа $M|M|m|n$ с n-местным буфером, позволяющая определять основные показатели качества обслуживания запросов в виде параметров СМО. Работоспособность предложенного подхода проверена на имитационной модели, построенной на основе статистического анализа функционирования системы, проведено ее экспериментальное исследование.

Ключевые слова: вычислительные кластеры, виртуальные машины, физические серверы, модель распределения ресурсов, эвристические алгоритмы, модельный эксперимент.

Для цитирования: Мочалов В.П., Братченко Н.Ю., Линец Г.И., Палканов И.С. Методы и модели сервиса распределения ресурсов в кластерах с балансировкой нагрузки центров обработки данных. *Моделирование, оптимизация и информационные технологии.* 2022;10(2). Доступно по: https://moitvivt.ru/ru/journal/pdf?id=1188 DOI: 10.26102/2310-6018/2022.37.2.030 (На англ.).

Introduction

Cloud data centers are the most commonly used form of providing computing resources to corporate users. Research results have shown that in corporate data centers, as a rule, from 10% to 35% of the computing power of servers is realized [1,2]. One of the ways to increase the efficiency of using the computing resources of the data center and to reduce the cost of equipment is to improve the quality of its management. To do this, based on the given parameters of the data center computing cluster and input request streams, it is necessary to solve the problems of optimal distribution of software applications among virtual machines; distribution of virtual machines on the hardware and software platform of the data center; determining the number of servers in the cluster; the amount of its internal and external memory, provided achieving the maximum value of its performance as well as the minimum cost of resources of all servers. At the same time, as an optimality criterion, we will employ the average processing time of incoming requests to the data center cluster and its total resource costs. A fairly realistic model of a data center cluster, taking into account the properties of incoming requests for processing, can be built on the basis of a QS with arbitrary distributions of intervals between requests and the duration of their service. However, the application of models such as $G|G|n|m$, $G|G|n|1$, $G|G|n$ helps to obtain only approximate analytical results in the form of upper and lower estimates even for average values of the system load [3,4]. It was shown by L. Kleinrock [5] that under certain assumptions for the study of such systems, the apparatus of the QS type $M\mid M\mid n\mid m$ can be utilized. The article describes the development and investigation of a model for studying the resource allocation processes of cloud data center clusters.

Related works

The load distribution systems of widespread modern hypervisors (VMware Infrastructure, VMware ESXi Server, Microsoft Hyper-V) have direct application distribution methods as well as experimental settings selected empirically, which are often based on statistics and resource utilization forecasting [6–9]. For example, a resource scheduler based on the OpenStack platform algorithms neither properly matches the dynamically changing and unpredictable load rate nor fully reflects the real-time dynamics of all ongoing processes. These algorithms do not provide a complete solution to balance the load of data centers and to allocate physical server resources among virtual machines and network applications in real time [10–13]. Therefore, the quality of service is not properly guaranteed. For a higher efficiency of the hardware-software platform of data centers, it is necessary to develop rational methods and models implementing, first, resource allocation service for load balancing clusters and, second, optimization algorithms for placing requests and software applications on physical servers in real time.

Implementing a static approach to allocating data center resources

The static approach of data center resource allocation is used at a stable demand flow rate and is implemented by the data center hardware and software platform management system, shown in Figure 1. The resource allocation algorithms implemented here perform the placement of VMs and implement the procedures for their migration.

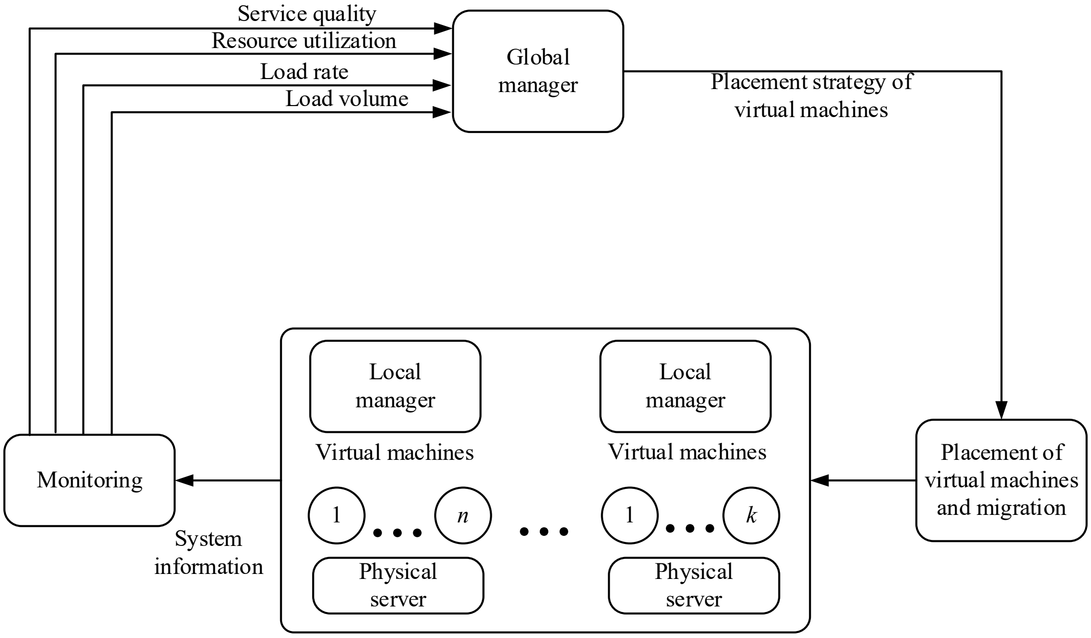

Figure 1 – Load distribution and balancing system
Рисунок 1 – Система распределения и балансировки нагрузки

The monitoring system, local and global managers manage the hardware and software platform of the data center, distribute physical servers, and migrate VMs. The non-polynomial algorithms used in this case, shown in Table 1, provide a solution to the problems of placing VMs on the hardware platform of the data center with acceptable accuracy (the FFD and BED algorithms provide distribution accuracy no more than 22% different from the optimal one) [14,15,16].

Table 1 – Non-polynomial data center resource allocation algorithms
Таблица 1 – Неполиномиальные алгоритмы распределения ресурсов ЦОД

Algorithm	NF	FF	BF	NFD	FFD	BFD
Accuracy	2	1.7	1.7	1.691	1.222	1.222

Studies [6,17,18,19] have shown that in the case of a data center cluster equipment load of more than 40%, these algorithms lead to unacceptable results. This necessitates the development of more rational approaches to the distribution of software applications across VMs as well as the distribution of the hardware and software platform of the data center between VMs. In this case, it is most advisable to use the following constraints [6,14,20]:

– An application can be executed on a single VM only, i.e.,

$$\sum_{n} a_{jn} \leq 1, \tag{1}$$

where $a_{jn} = \begin{cases} 1 \text{ if application is executed on } n \text{ VMs,} \\ 0 \text{ otherwise,} \end{cases}$

$$j = \overline{1,n},\ \ n = \overline{1,k}\,.$$

– The applications distributed to VMs request the total volumes of resources not exceeding the available ones, i.e.,

$$\sum_n a_{jn} C_j \le C_n, \quad \sum_n a_{jn} m_j \le M_n, \tag{2}$$

where C_n and M_n are the performance and memory capacity of the j th VM.

– The applications are indivisible for all time intervals t_n of their execution, i.e.,

$$\sum_{t=0}^{t_n} \sum_n a_{jn} = t_j \sum_n a_{jn}. \tag{3}$$

To solve the problem of distributing software applications over VMs, one can use the greedy heuristic algorithm presented below [21,22], which provides a distribution close to optimal.

1. Rearrange all software applications $j \in r$ in the descending order of their requests for the resources of VMs.

2. Rank all VMs $k \in K$ by their performance, top to bottom.

3. For each application $j(k)$ on each time interval t_n, select a VM that is able to execute this application.

4. If such VM is not selected, the application will not be executed. Otherwise, the required volumes of resources are reserved on the VM.

When solving the problem of distributing a set BM_k, $k = \left(\overline{1,N}\right)$ to a set of application servers S_i, $i = \left(\overline{1,L}\right)$ and file servers H_j, $j = \left(\overline{1,M}\right)$, we introduce the following restrictions:

– Each VM can be placed on a single server only, i.e.,

$$\sum_{i=1}^{L} \sum_{j=1}^{M} \sum_{k=1}^{N} x_{ijk} = 1, \tag{4}$$

where $x_{ijk} = \begin{cases} 1 \text{ if } VM_k \text{ is placed on } S_i \text{ or } H_j, \\ 0 \text{ otherwise.} \end{cases}$

– The memory resources (RAM) requested by VMs from physical servers satisfy the inequality.

$$\sum_{i=1}^{L} \sum_{j=1}^{M} \sum_{k=1}^{N} RAM_k x_{ijk} < RAM_{ij}. \tag{5}$$

– The server performance constraint, written as

$$\sum_{i=1}^{L} \sum_{j=1}^{M} \sum_{k=1}^{N} CPU_k x_{ijk} < CPU_j. \tag{6}$$

– The disk storage capacity constraint, written as

$$\sum_{i=1}^{L}\sum_{j=1}^{M}\sum_{k=1}^{N} S_k x_{ijk} < S_{ij}. \tag{7}$$

– The performance constraint on the data input-output system, written as

$$\sum_{i=1}^{L}\sum_{j=1}^{M}\sum_{k=1}^{N} IOPS_k x_{ijk} < IOPS_{ij}. \tag{8}$$

The throughput of the switching system must meet the following conditions

$$\sum_{i=1}^{n}\sum_{l=1}^{L} r_i l < \Pi, \qquad \sum_{i=1}^{n}\sum_{l=1}^{L} r_i l < K \tag{9}$$

where $r_i l$ is the required throughput of VM_i under the utilization of l virtual channels; Π and *K* are the total throughputs of physical channels and the switching system.

Choose the following criteria of optimality:

– the maximum load of data center equipment,

$$K_1 = \sum_{i=1}^{L}\sum_{j=1}^{M}\sum_{k=1}^{N} CPU_k x_{ijk} \cdot RAM_k x_{ijk} \tag{10}$$

– the maximum performance of the input-output system,

$$K_2 = \sum_{i=1}^{L}\sum_{j=1}^{M}\sum_{k=1}^{N} IOPS_k x_{ijk} \tag{11}$$

– the maximum performance of the switching system and transfer system,

$$K_3 = \sum_{i=1}^{n}\sum_{l=1}^{L} r_i l \tag{12}$$

This problem can be solved by sequential processing of requests for VM placement. The external data center scheduler allocates to each VM the maximum value of the resources of CPU, RAM, disk, IOPS, etc. A physical resource with a minimum residual sum of its parameters is determined, and a VM is placed VM$_i$ on it using the following scheme:

1. All VMs are ranked by the value of the characteristic $V_i = \sum_{j=1}^{K} C_j V_j^i$, where V_j^i is the request of VM_i for resource j, $j = \overline{1,K}$, and C_j denotes the significance of this resource.

2. In accordance with the initial scheme, a VM with the highest resource requirements is selected and then placed on a physical element with a minimum resource. A most reasonable method is to use here an iterative algorithm based on dynamic programming.

For example, the iterative greedy algorithm runs as follows [23,24]. As the input processes a_i of the set $S = \{a_1, a_2, \ldots, a_n\}$ the final times f_i of their implementation are used. The sequentially selected processes are combined by the Greedy Activity Selector procedure (s, f) into a set A in which f_i is the maximum final time of all processes

3. The external data center scheduler estimates the performance of the VM placed to the server.

4. If the VM's performance is below the specified level, the external scheduler moves it to a physical element with a higher level of resources and goes back to Step 3.

5. If the VM's performance is higher than the specified level, the external scheduler moves it to a physical element with a lower level of resources and goes back to Step 3.

If it is impossible to distribute VM, the transition to the limited enumeration procedure is performed. The enumeration depth, which determines the maximum number of servers for which the distribution is assigned, must guarantee the required balance between the quality of service and the VM distribution time (find an acceptable solution in an allowed time). The scheme of the data center resource allocation algorithm is shown in Figure 2.

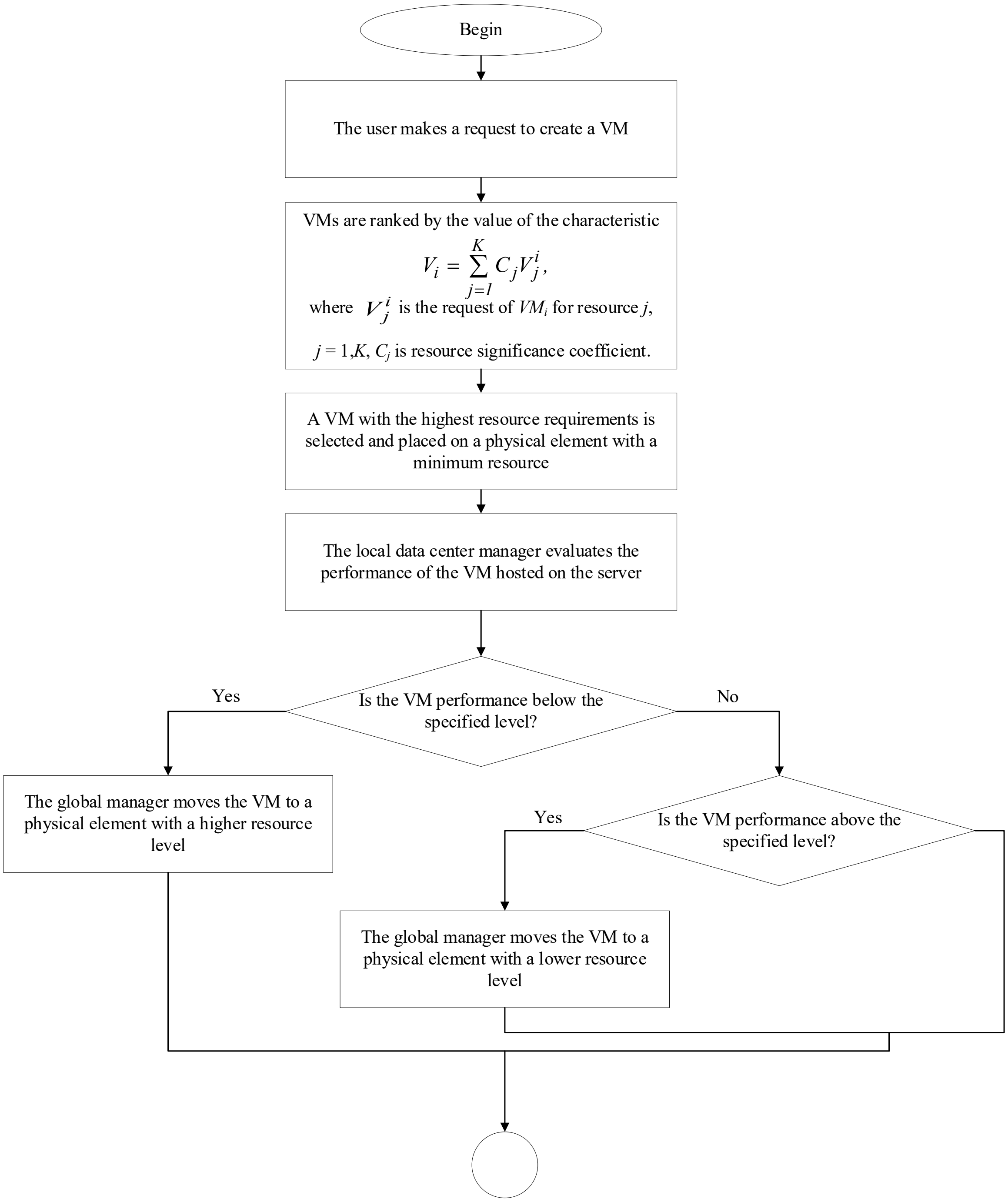
Begin
The user makes a request to create a VM
VMs are ranked by the value of the characteristic
$V_i = \sum_{j=1}^{K} C_j V_j^i$,
where V_j^i is the request of VM_i for resource j,
$j = 1,K$, C_j is resource significance coefficient.
A VM with the highest resource requirements is selected and placed on a physical element with a minimum resource
The local data center manager evaluates the performance of the VM hosted on the server
Is the VM performance below the specified level?
Yes
No
The global manager moves the VM to a physical element with a higher resource level
Is the VM performance above the specified level?
Yes
The global manager moves the VM to a physical element with a lower resource level

Figure 2 – Diagram of the data center resource allocation algorithm
Рисунок 2 – Схема алгоритма распределения ресурсов ЦОД

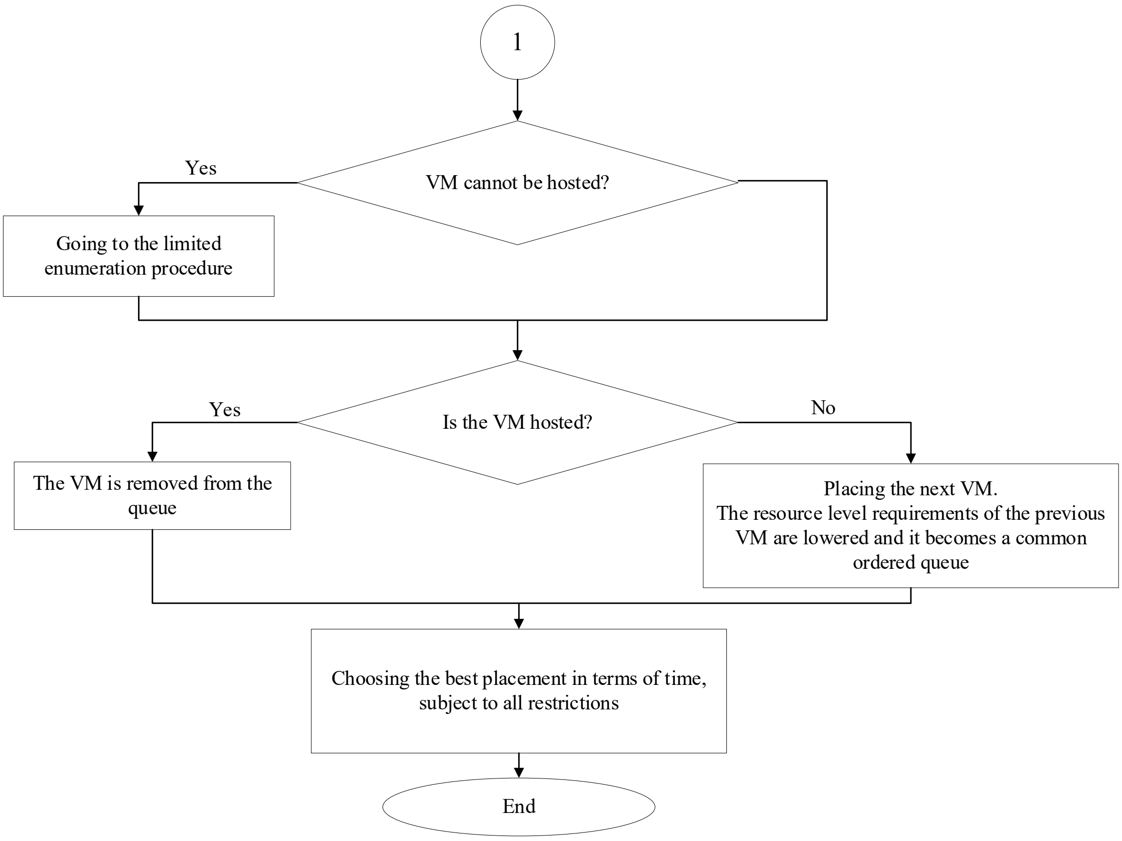
1
VM cannot be hosted?
Yes
Going to the limited enumeration procedure
Is the VM hosted?
Yes
No
The VM is removed from the queue
Placing the next VM.
The resource level requirements of the previous VM are lowered and it becomes a common ordered queue
Choosing the best placement in terms of time, subject to all restrictions
End

Figure 2 (continued)
Рисунок 2 (продолжение)

The quality of service can be estimated by the probability of blocking for P_δ service requests; otherwise, by the probability of no admissible servers per unit time. This can be done using the Erlang formula

$$P_\delta = \frac{a^n / n!}{\sum\limits_{k=0}^{n} a^k / k!}, \tag{13}$$

where P_δ denotes the probability of blocking;
n is the number of servers;
a is the rate of requests.

The value of this probability can be determined in a recurrent way [20,25,26]

$$P_\delta = B(n, a) = \frac{B(n-1, a)}{B(n-a) + n/a}, \tag{14}$$

where $n = 1, 2, ...,$ and $B(0, a) = 1$.

In the case of large values of n and a, the computing time of P_δ can be reduced by choosing a required deviation $P_\delta(i) - P_\delta(i-1) < \delta$ and terminating the enumeration procedure as soon as the value δ is achieved. Here δ specifies the accuracy of calculations.

6. If the resource is not found, then the next VM is placed, the resource requested by the previous VM is decreased, and it is included in the common ordered queue.

7. If the resource is found, then the VM is eliminated from the queue.

8. The time-optimal placement is selected under the existing constraints.

Analytical model of resource allocation for a data center cluster

Figure 3 shows a formalized model of a data center in the form of a multichannel QS, which includes the following elements:

- input stream of requests for processing;
- a set of clusters, each of which is formed by many parallel servers;
- input buffer (cluster memory);
- central buffer common to all clusters.

Figure 3 shows a formalized model of a data center in the form of a multichannel QS, which includes the following elements:

If the incoming flows of requests are ordinary and stationary, then the total input flow is Markov and the main indicators of the quality of its processing can be defined as indicators of the QS. A request that arrives in the system and makes at least one of the cluster servers free occupies any free VM without waiting. The service characteristics of the request can be described by the indicators of the M/M/N system with the service probability P1. When all servers and cluster memory are busy, the request is sent to an unlimited external memory-central buffer. The service characteristics can be described by the indicators of the M/M/N/m system with the service probability P3. If all VMs in the cluster are busy, the request goes to the cluster memory queue. The probability of this event is P2=1-P1-P3.

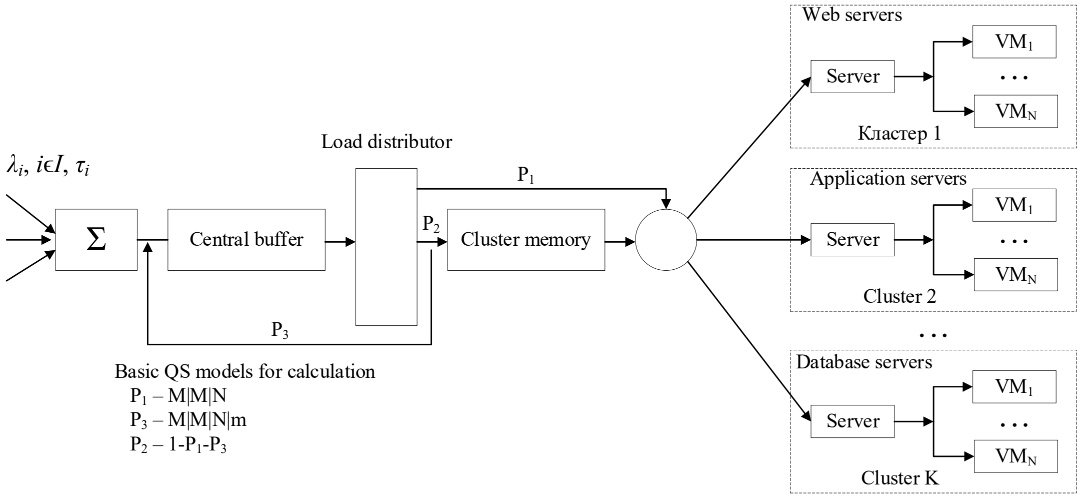

Figure 3 – Formalized data center model
Рисунок 3 – Формализованная модель ЦОД

For the QS $M \mid M \mid m \mid n$, $1 \le n < \infty$, $0 \le m < \infty$ the state graph is shown in Figure 4.

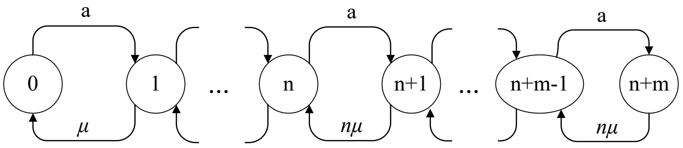

Figure 4 – State graph of the QS $M|M|n|m$
Рисунок 4 – Граф состояний СМО $M|M|n|m$

In the stationary mode, the probability distribution of states of a given QS has the form [15]

$$P_k = \frac{\dfrac{\rho^k}{k!}}{\sum\limits_{k=0}^{n} \dfrac{\rho^k}{k!} + \dfrac{\rho^{n+1}}{n \cdot n!} - \dfrac{1-\left(\dfrac{\rho}{n}\right)^m}{1-\dfrac{\rho}{n}}}, \ 0 \le k \le n, \tag{15}$$

$$P_{m+s} = \frac{\dfrac{\rho^m}{m!} \cdot \left(\dfrac{\rho}{m}\right)^s}{\sum\limits_{k=0}^{n} \dfrac{\rho^k}{k!} + \dfrac{\rho^{n+1}}{n \cdot n!} - \dfrac{1-\left(\dfrac{\rho}{n}\right)^m}{1-\dfrac{\rho}{n}}}, \ 1 \le s \le m, \tag{16}$$

where $\rho = \dfrac{a}{n\mu}$ is the system load (in case $n < \infty$);

P_k, P_{m+s} are the probabilities of finding the system in the *k*-th and *(m + s)*-th states.

System loading:

$$q = 1 - P_0 = \sum_{j=1}^{m+n} P_j \tag{17}$$

Average number of requests in the system:

$$N_c = \sum_{k=0}^{m+n} k \cdot P_k \tag{18}$$

The average number of requests in the queue:

$$N_o = \sum_{k=n+1}^{m+n} (k-n) \cdot P_k \tag{19}$$

Average time spent by requests in the system:

$$T_c = \frac{N_c}{n\mu(1-P_0)} = \frac{\sum\limits_{k=0}^{m+n} k \cdot P_k}{n\mu(1-P_0)} \tag{20}$$

Average waiting time for requests in the system:

$$T_o = \frac{N_o}{n\mu(1-P_0)} = \frac{\sum\limits_{k=m+1}^{m+n} (k-m) \cdot P_k}{n\mu(1-P_0)} \tag{21}$$

Average service time:

$$T_{обсл} = T_c - T_o = \frac{1}{m\mu} + \frac{(m-1)\sum_{k=m}^{m+n} P_k + \sum_{k=2}^{m-1} (k-1)\cdot P_k}{m\mu(1-P_0)} \tag{22}$$

The probability that all servers in the cluster are free:

$$P_0 = \left[\sum_{k=0}^{n-1} \frac{a^k}{k!} + \frac{a^n}{n!(1-\rho)} \right]^{-1} \tag{23}$$

where $a = \frac{\lambda}{n\mu}$ and $\rho = \frac{\Lambda}{n\mu}$ are traffic intensity and cluster utilization rate.

Probability of request processing delay:

$$P_D = P_0 \frac{a^n}{n!(1-\rho)} \tag{24}$$

The probability of an out-of-order request:

$$P_1 = 1 - P_D \tag{25}$$

The probability of a request arriving when the input queue is overflowed is equal to the probability of blocking the system $M \mid M \mid n \mid m$:

$$P_3 = \frac{(1-\rho)\rho^s}{1-\rho^{s+1}} \tag{26}$$

where *s* is the size of the input buffer.

The probability of receiving a request when all the servers in the cluster are busy:

$$P_2 = 1 - P_1 - P_3 \tag{27}$$

For an Erlang system with failures, which contains *n* parallel servers and an unrestricted central buffer, the average time a query spends in a cluster will be [5,27]:

$$\tau_{\text{cp}} = \left[\tau_s P_1 + (\tau_\omega + \tau_s) P_2 + \tau_r P_3 \right] \cdot \frac{1}{1-P_s} \tag{28}$$

where $\tau_\omega = \frac{\rho}{\mu - \Lambda} - \frac{n\rho^{s+1}}{\mu - \Lambda\rho^{s+1}}$.

It can be seen from the obtained expressions that the average time for processing requests by a data center cluster depends on its load by the input stream of requests and on the amount of internal shared memory. By setting the required time for processing an application by the cluster, you can define the necessary parameters of the servers and VM of the cluster.

Numerical experiment

In Python the programming language, a simulation model has been developed that implements the presented algorithm (certificate of state registration of a computer program No. 2020617795 dated July 15, 2020). The simulation was carried out for a data center based on the Huawei 2488 platform, Intel (R) Xeon (R) Gold 6154 processor, 512 Gb memory containing 10 hosts and 40 VMs. The boundaries of the required VM resources were randomly generated in the range of 35–70% of the server resources. The throughput of the switching system is defined in the range of 0.7–1 Gbps. The random VM migration time was set by the gamma distribution. The distribution of the number of requested resources for each VM is described by normal law. The research results are shown in Tables 2-5.

Table 2 – Utilization percentage of the number of servers in a data centre cluster
Таблица 2 – Процент использования количества серверов кластера ЦОД

Number of cluster servers	1	2	3	4	5	6	7	8	9	10
Used cluster servers before placement, %	85	63	100	81	42	37	34	45	30	40
Used cluster servers after placement, %	73	75	70	75	–	–	–	–	–	–

Table 3 – Percentage of buffer memory usage in data center cluster servers
Таблица 3 – Процент использования буферной памяти серверов кластера ЦОД

Numbers of hosts	1	2	3	4	5	6	7	8	9	10
memory usage before placement, %	60	65	70	45	27	25	20	32	40	43
memory usage after placement, %	80	65	40	70	–	–	–	–	–	–

Table 4 – Time to host virtual machines on data center cluster servers
Таблица 4 – Время размещения виртуальных машин на серверах кластера ЦОД

Number of VMs	5	10	15	20	25	30	35	40
Placement time, sec.	12	19	25	31	50	75	85	90

Table 5 – Time of processing requests by a data center cluster
Таблица 5 – Время обработки запросов кластером ЦОД

Load of servers	0.35	0.40	0.45	0.50	0.55	0.60
Request processing time, sec.	0.103	0.147	0.183	0.216	0.342	0.451

It follows from the simulation results that the proposed model for the distribution of data center cluster resources provides acceptable distribution rates for the number of VMs not exceeding 20. With an increase in the number of VMs, the time for distribution and processing of requests by a data center cluster grows exponentially. The implementation of the presented algorithm reduces the number of used data center servers by 60%, with the number of VMs being hosted not exceeding 40.

Conclusion

An approach based on a heuristic greedy algorithm with a bounded enumeration procedure and restrictions on the resources of the hardware and software complex of the data center is proposed to solve and study the problem of efficient allocation of resources for a computing cluster in a cloud data center. The search depth, which determines the maximum possible number of servers and virtual machines to assign allocation, provides the necessary balance between quality of service and allocation time. The algorithm for solving the problem includes the following stages: optimal distribution of software applications among virtual machines, distribution of virtual machines among servers of a data center cluster, construction of an algorithm diagram, determination of the main indicators of request processing quality by means of analytical and simulation modeling. Taking into account that such stochastic systems satisfy the ergodicity condition, the parameters of the data center cluster, which determine the main indicators of request processing quality, are defined as indicators of the QS operating in a stationary mode. The use of the proposed approach will make it possible to reasonably carry out the placement of software applications on the VM of a corporate data center, choose the composition of virtual machines and solve the problems of their rational placement on the physical servers of the data center clusters.

REFERENCES

1. Gnedenko B.V., Kovalenko I.N. Introduction to queuing theory. LCI Publisher; 2007. 400 p.
2. Aliev T.I. Fundamentals of modeling of discrete systems. St. Petersburg, ITMO; 2009. 363 p.
3. Feller E., Rilling L., Morin C. A scalable and autonomic virtual machine management framework for private Clouds. Proceedings of the 12th IEEE/ACMInternational Symposium on Cluster, Cloud and Grid Computing (CCGrid). 2021:482–489.
4. Ward J.S., Barker A. Cloud cover: monitoring large-scale clouds with Varanus. Journal of Cloud Computing: Advances, Systems and Applications, 2015;4:127–135.
5. Kleinrock L.. Queueing Theory. Mashinostroenie; 1979. 432 p.
6. Mochalov V.P., Linets G.I., Bratchenko N.Y., Govorova S.V. An analytical model of a corporate software-controlled network switch. Scalable Computing. 2020;21(2):337–346.
7. Boev V. *Kompjuternoe modelirovanie: Posobie dlja prakticheskih zanjatij, kursovogo i diplomnogo proektirovanija v AnyLogic7*. St. Petersburg, VAS Publ.; 2014. 432 p. (in Russ.)
8. Taihoon K., Soksoo K. *Analysis of Security Session Reusing in Distribution Server System*. Computational Science and Its Applications. ICCSA 2006; 2006. 1045 p.
9. Holland J.H. *Adaptation in Natural and Artificial Systems:An Introductory Analysis with Applications to Biology,Control, and Artificial Intelligence*. The MITPress, Cambridge; 1992. 211 p.
10. Khritankov A. Modeli i algoritmy raspredelenija nagruzki. Algoritmy na osnove setej SMO. *Informacionnye tehnologii i vychislitel'nye seti = Information technologies and computer networks.* 2009;(3):257 p. (in Russ.).
11. Ivanisenko I., Kirichenko L., Radivilova T. Balancer multifractal methods considering load characteristics. *International Journal «Information Content and Processing»*. 2015;2(4):345–368.
12. Panchenko T.V. *Genetic Algorithms*. Astrakhan, *Astrakhanskiy Universitet*; 2007. 87 p.
13. Tsoy Yu.R., Spitsyn V.G. *Genetic Algorithm*. Tomsk, Knowledge Representation in Information Systems; 2006. 146 p.
14. Mochalov V.P., Bratchenko N.Y., Yakovlev S.V. Analytical model of object request broker based on Corba standard. *Journal of Physics: Conference Series*. 2018;1015(2). DOI: 10.1088/1742-6596/1015/2/022012.
15. McNab A., Stagni F., and Luzzi C. LHCb experience with running jobs in virtual machines. *Journal of Physics: Conference Series*. 2016;664:1–7.

16. Ward J.S., Barker A Observing the clouds: a survey and taxonomy of cloud monitoring. *Journal of Cloud Computing: Advances. Systems and Applications*. 2014;3:25–33.
17. Mochalov V.P., Bratchenko N.Y., Yakovlev S.V. Analytical model of integration system for program components of distributed object applications. *International Russian Automation Conference, RusAutoCon 2018*. 2018;8501806. DOI: 10.1109/ RUSAUTOCON.2018.8501806.
18. Computing Center of the Institute of High Energy Physics (IHEP-CC). «VCondor – virtual computing resource pool manager based on HTCondor». 2016. Available by: //github.com/hep-gnu/VCondor.
19. Anne-C´ecile Orgerie. When Clouds become Green: the Green Open Cloud Architecture. *International Conference on Parallel Computing (ParCo)*. 2009;228–237.
20. Mochalov V.P., Bratchenko N.Yu., Yakovlev S.V., Gosteva D.V. Distributed management system for infocommunication networks based on TM Forum Framework. *CEUR Workshop Proceedings*. 2016;2254:81–93.
21. Mochalov V., Bratchenko N., Linets G., Yakovlev S. Distributed management systems for infocommunication networks: A model based on tm forum frameworx. *Computers*. 2019;8(2). DOI: 10.3390/computers8020045.
22. Mochalov V.P., Bratchenko N.Y., Yakovlev S.V. Process-Oriented Management System for Infocommunication Networks and Services Based on TM Forum Frameworx. *Proceedings - 2019 International Russian Automation Conference, RusAutoCon 2019*. 2019;8867619. DOI: 10.1109/RUSAUTOCON.2019.8867619.
23. McNab A., Love P., MacMahon E. Managing virtual machines with Vac and Vcycle. *Journal of Physycs: Conference Series.* 2015;664b:115–122.
24. Beloglazov R. OpenStack Neat: A Framework for Dynamic and Energy-Efficient Consolidation of Virtual Machines in OpenStack Clouds. *Concurrency and Computation: Practice and Experience (CCPE)*. 2015;27(5):1310–1333.
25. Kuzin L.T. *Fundamentals of cybernetic models*. M.: Energia; 1979. 584 p.
26. Open Grid Forum. «Open Cloud Computing Interface». 2016. Available by: http://occiwg.org/.
27. Balashov N., Baranov A., Korenkov V. Optimization of over-provisioned clouds. *Physics of Particles and Nuclei Letters*. 2016;13(5):609–612.

СПИСОК ИСТОЧНИКОВ

1. Gnedenko B.V., Kovalenko I.N. *Introduction to queuing theory.* LCI Publisher; 2007. 400 p.
2. Aliev T.I. *Fundamentals of modeling of discrete systems*. St. Petersburg, ITMO; 2009. 363 p.
3. Feller E., Rilling L., Morin C. A scalable and autonomic virtual machine management framework for private Clouds. *Proceedings of the 12th IEEE/ACMInternational Symposium on Cluster*, *Cloud and Grid Computing (CCGrid)*. 2021:482–489.
4. Ward J.S., Barker A. Cloud cover: monitoring large-scale clouds with Varanus. *Journal of Cloud Computing: Advances, Systems and Applications*, 2015;4:127–135.
5. Kleinrock L.. *Queueing Theory*. Mashinostroenie; 1979. 432 p.
6. Mochalov V.P., Linets G.I., Bratchenko N.Y., Govorova S.V. An analytical model of a corporate software-controlled network switch. *Scalable Computing*. 2020;21(2):337–346.
7. Боев В.Д. *Компьютерное моделирование. Пособие для практических занятий, курсового и дипломного проектирования в AnyLogic7.* Санкт-Петербург; 2014. 432 с.
8. Taihoon K., Soksoo K. *Analysis of Security Session Reusing in Distribution Server System*. Computational Science and Its Applications. ICCSA 2006; 2006. 1045 p.

9. Holland J.H. *Adaptation in Natural and Artificial Systems:An Introductory Analysis with Applications to Biology,Control, and Artificial Intelligence*. The MITPress, Cambridge; 1992. 211 p.
10. Хританков А.С. Модели и алгоритмы распределения нагрузки. Алгоритмы на основе сетей СМО. *Информационные технологии и вычислительные системы*. 2009;(3):33-48.
11. Ivanisenko I., Kirichenko L., Radivilova T. Balancer multifractal methods considering load characteristics. *International Journal «Information Content and Processing»*. 2015;2(4):345–368.
12. Panchenko T.V. *Genetic Algorithms*. Astrakhan, *Astrakhanskiy Universitet*; 2007. 87 p.
13. Tsoy Yu.R., Spitsyn V.G. *Genetic Algorithm*. Tomsk, Knowledge Representation in Information Systems; 2006. 146 p.
14. Mochalov V.P., Bratchenko N.Y., Yakovlev S.V. Analytical model of object request broker based on Corba standard. *Journal of Physics: Conference Series*. 2018;1015(2). DOI: 10.1088/1742-6596/1015/2/022012.
15. McNab A., Stagni F., and Luzzi C. LHCb experience with running jobs in virtual machines. *Journal of Physics: Conference Series*. 2016;664:1–7.
16. Ward J.S., Barker A Observing the clouds: a survey and taxonomy of cloud monitoring. *Journal of Cloud Computing: Advances. Systems and Applications*. 2014;3:25–33.
17. Mochalov V.P., Bratchenko N.Y., Yakovlev S.V. Analytical model of integration system for program components of distributed object applications. *International Russian Automation Conference, RusAutoCon 2018*. 2018;8501806. DOI: 10.1109/ RUSAUTOCON.2018.8501806.
18. Computing Center of the Institute of High Energy Physics (IHEP-CC). «VCondor – virtual computing resource pool manager based on HTCondor». 2016. Доступно по: //github.com/hep-gnu/VCondor.
19. Anne-C´ecile Orgerie. When Clouds become Green: the Green Open Cloud Architecture. *International Conference on Parallel Computing (ParCo)*. 2009;228–237.
20. Mochalov V.P., Bratchenko N.Yu., Yakovlev S.V., Gosteva D.V. Distributed management system for infocommunication networks based on TM Forum Framework. *CEUR Workshop Proceedings*. 2016;2254:81–93.
21. Mochalov V., Bratchenko N., Linets G., Yakovlev S. Distributed management systems for infocommunication networks: A model based on tm forum frameworx. *Computers*. 2019;8(2). DOI: 10.3390/computers8020045.
22. Mochalov V.P., Bratchenko N.Y., Yakovlev S.V. Process-Oriented Management System for Infocommunication Networks and Services Based on TM Forum Frameworx. *Proceedings - 2019 International Russian Automation Conference, RusAutoCon 2019*. 2019;8867619. DOI: 10.1109/RUSAUTOCON.2019.8867619.
23. McNab A., Love P., MacMahon E. Managing virtual machines with Vac and Vcycle. *Journal of Physycs: Conference Series*. 2015;664b:115–122.
24. Beloglazov R. OpenStack Neat: A Framework for Dynamic and Energy-Efficient Consolidation of Virtual Machines in OpenStack Clouds. *Concurrency and Computation: Practice and Experience (CCPE)*. 2015;27(5):1310–1333.
25. Kuzin L.T. *Fundamentals of cybernetic models*. M.: Energia; 1979. 584 p.
26. Open Grid Forum. «Open Cloud Computing Interface». 2016. Доступно по: http://occiwg.org/.
27. Balashov N., Baranov A., Korenkov V. Optimization of over-provisioned clouds. *Physics of Particles and Nuclei Letters*. 2016;13(5):609–612.

ИНФОРМАЦИЯ ОБ АВТОРАХ / INFORMATION ABOUT THE AUTHORS

Мочалов Валерий Петрович, доктор технических наук, профессор кафедры инфокоммуникаций Северо-Кавказского федерального университета, Ставрополь, Российская Федерация.
e-mail: mochalov.valery2015@yandex.ru

Mochalov Valery Petrovich, Doctor of Technical Sciences, Professor of the Chair of Infocommunications of North-Caucasus Federal University, Stavropol, Russian Federation.

Братченко Наталья Юрьевна, кандидат физико-математических наук, доцент; кафедра инфокоммуникаций; доцент Северо-Кавказского федерального университета, Ставрополь, Российская Федерация.
e-mail: nb20062@rambler.ru

Bratchenko Natalya Yurievna, Candidate of Physical and Mathematical Sciences, Assistant Professor of the Chair of Infocommunications of North-Caucasus Federal University, Stavropol, Russian Federation.

Линец Геннадий Иванович, доктор технических наук, доцент; кафедра инфокоммуникаций; заведующий кафедрой инфокоммуникаций Северо-Кавказского федерального университета, Ставрополь, Российская Федерация.
e-mail: kbytw@mail.ru

Linets Gennady Ivanovich, Doctor of Technical Sciences, Assistant Professor; Head of the Chair of Infocommunications of North Caucasian Federal University, Stavropol, Russian Federation.

Палканов Илья Сергеевич, аспирант Северо-Кавказского федерального университета, Ставрополь, Российская Федерация.
e-mail: ilya0693@yandex.ru

Palkanov Ilya Sergeevich, Postgraduate Student of North-Caucasus Federal University, Stavropol, Russian Federation

Статья поступила в редакцию 28.05.2022; одобрена после рецензирования 14.06.2022; принята к публикации 30.06.2022.

The article was submitted 28.05.2022; approved after reviewing 14.06.2022; accepted for publication 30.06.2022.