ISSN 0361-7688, Programming and Computer Software, 2019, Vol. 45, No. 5, pp. 251–256. © Pleiades Publishing, Ltd., 2019.
Russian Text © The Author(s), 2019, published in Programmirovanie, 2019, Vol. 45, No. 5.

Approaches to Improving the Efficiency of Data Centers

V. A. Kostenko[a],* and A. A. Chupakhin[a],**

[a] *Moscow State University, Moscow, 119991 Russia*

** e-mail: kost@cs.msu.su*

*** e-mail: andrewchup@lvk.cs.msu.ru*

Received November 18, 2018; revised November 18, 2018; accepted November 26, 2018

Abstract—Methods for improving the efficiency of using data center resources are proposed. They are based on expanding the functional capabilities and improving the accuracy of mapping requests to physical resources in the scheduler used in the cloud platform.

DOI: 10.1134/S0361768819050049

INTRODUCTION

We consider data centers working as IaaS (Infrastructure-as-a-Service). If a data center (DC) operates in the IaaS mode, the performance criteria should be specified in the service-level agreement (SLA) for all types of resources—storage, computing and network resources. The computing resources, data storages, and networking resources must be considered as scheduled types of resources. For the virtual resources allocated to a DC, one must guarantee the compliance with the required SLA.

We will evaluate the efficiency of the DC operation using the following criteria:

1. Utilization of the physical resources (the utilization of computing resources and storage should be maximized and the workload of network resources should be minimized).

2. Percentage of assigned requests for the creation of virtual networks relative to the number of arrived requests.

3. Performance of virtual machines.

The approaches to improving the efficiency of using DCs proposed in this paper are based on expanding the functional capabilities and improving the accuracy of the algorithm of mapping requests to physical resources that is used in the scheduler of the cloud platform.

1. THE PROBLEM OF MAPPING REQUESTS TO PHYSICAL DC RESOURCES

The model of DC physical resources is specified by a labeled graph $H = (P \cup M \cup K, L)$, where P is the set of computer nodes (servers), M is the set of data storages, K is the set of commutation elements of the DC data communication network, and L is the set of physical data transmission channels. On the sets P, M, K, and L, vector functions of a scalar argument are defined that specify, respectively, the characteristics of the computer nodes, data storages, commutation elements, and data transmission channels.

A request for creating a virtual network is specified by the labeled graph $G = (W \cup S, E)$, where W is the set of virtual machines, S is the set of virtual data storages (storage elements), and E is the set of virtual data transmission channels. On the sets W, S, and E, vector functions of a scalar argument that specify the characteristics of the requested virtual element (the required SLA) are defined. For the virtual machines, the following assignment policies may be additionally specified:

1. For each virtual machine, a set of physical servers on one of which it should be located; this is called the VM-to-PM affinity rules;

2. For each virtual machine, a set of physical servers on which it cannot be located; this is called the VM-to-PM anti-affinity rules;

3. The set of virtual machines that must be located on the same physical server; this is called the VM-to-VM affinity rules;

4. The set of virtual machines that must be located on different physical servers; this is called the VM-to-VM anti-affinity rules.

The assignment of a number of virtual machines to the same server reduces delay of their interaction. The placement of virtual machines on different servers improves reliability; for example, two instances of the same Web server can be deployed on two different machines. The VM-to-PM policies have an administrative function—moving virtual machines to a different server or deleting them from a server that should be soon switched off.

The assignment of a request is the mapping

$$A: G \to H = \{W \to P, S \to M, E \to \{K, L\}\}.$$

There are three types of constraints between the characteristics of requests and the corresponding characteristics of physical resources. Denote by x a characteristic of a request element and by y the corresponding characteristic of the physical resource. Then, these constraints can be written as follows.

1. The physical resource cannot be overloaded:

$$\sum_{i \in R_j} x_i \le y_j,$$

where R_j is the set of requests assigned to the physical resource j.

2. The types of the physical and virtual resources must match each other:

$$x_i = y_j.$$

3. The physical resource must have the required characteristics:

$$x_i \le y_j.$$

A mapping A is said to be correct if, for all physical resources and all the characteristics, constraints 1–3 and the given policies of the virtual machines are satisfied.

The residual graph H_{res} of available resources is the graph H for which the values of the functions are redefined according to the corresponding characteristics that must satisfy constraint 1.The value of each characteristic of each physical resource is reduced by the sum of values of the corresponding virtual resources assigned to this physical resource.

The input data for constructing the mapping of virtual resources to physical resources are as follows:

1. The residual graph of available resources $H_{res} = (P \cup M \cup K, L)$.

2. The set of assigned requests $Ten = \{(T_i, flag_i)\}$, where $flag_i = (0|1)$ indicates if the request T_i may or may not migrate.

3. The mapping of assigned requests: $Assigned = \{A: T_i \to H_{res}\}$.

4. The set of requests $Z = \{G_i\}$ received by the scheduler;

5. The set of additionally requested resources (virtual machines, storage, and networking resources) to the earlier assigned requests $V = \{(W, i_w), (S, i_s), (E, i_e)\}$, where i_w, i_s, i_e are the indexes of requests associated with these virtual resources.

Define the set $R = Z \cup V$ of all upcoming requests that want to be assigned to physical resources of the DC. It is required to assign for execution on the DC the maximal number of requests from the set R and, if the mapping of some virtual resources changed, then design a migration plan for these resources. The migration plan of virtual resources must be designed taking into account the guaranteed compliance with the SLA for the virtual resources that were earlier allocated in the DC as this plan is implemented and taking into account the constraints on the plan execution time.

2. REQUIREMENTS FOR THE ALGORITHM

The efficiency of operation of cloud platforms and the efficiency of the DC resource utilization significantly depends on the algorithm of mapping requests to the physical resources of the DC that is used in the cloud platform scheduler. These can be greedy algorithms, algorithms that combine greedy strategies with limited search [1, 2], and heuristic algorithms [3, 4]. In particular, modified bin packing algorithms [5] are used. Other approaches use ant colony algorithms [6], simulation annealing [7], genetic [8], and immune algorithms [9].

To improve the efficiency of utilizing the physical resources of a DC, the algorithm must possess the following properties:

1. The number of assigned requests is maximized.

2. The computing resources and data storage elements are the resources to be scheduled.

3. Networking resources are the resources to be scheduled.

4. The virtual resource assignment policies are taken into account.

5. The NUMA server architecture is taken into account.

6. A migration plan for earlier allocated virtual resources can be designed. This is needed to avoid fragmentation of physical resources occurring during the DC operation.

To our knowledge, none of the available algorithms possesses all these properties.

3. AN ALGORITHM FOR REQUEST MAPPING AND APPROACHES TO IMPROVING THE EFFICIENCY OF DATA CENTER UTILIZATION

We propose an algorithm based on a combination of greedy strategies and limited search [10]. This algorithm consists of two stages:

1. Assigning virtual machines to computer nodes and virtual data storage systems to storage servers.

2. For the mappings of virtual machines and virtual data storage systems to physical resources obtained at the first step, the underlying physical channels are assigned for the virtual channels in the DC exchange network.

Algorithm of Stage 1.

1. Select the next request G_i from the set of resource requests R using a greedy criterion K_G.

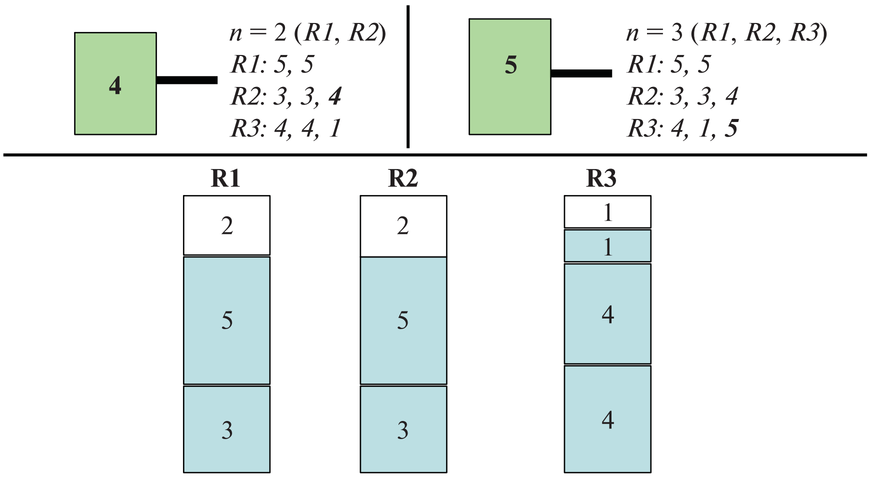

Fig. 1. Example of using the limited search procedure.

2. Select the next element e (a virtual machine $e \in W$ $(W \in G_i)$ or a storage element $e \in S$ $(S \in G_i)$) using a greedy criterion K_e. If G_i is empty, then go to Step 1.

3. Form the set of physical resources Ph ($Ph \subseteq P$ or $Ph \subseteq M$, respectively), to which the selected element e may be assigned, i.e., the set satisfying the mapping correctness constraints if the request e is assigned to the physical resource belonging to this set.

3.1. If the set Ph is empty, then call the limited search procedure.

1. If this procedure returns failure (is unsuccessful), then the request G_i cannot be assigned; in this case, remove the earlier assigned elements G_i of the request and redefine the values of physical resource characteristics that must satisfy constraint 1.

2. If this procedure returns a new mapping, then call the migration plan designing procedure. If such a plan is successfully constructed, then assign the elements according to the new mapping and redefine the values of the physical characteristics so that they satisfy constraint 1.

3. If the set R is not empty, then go to Step 1; otherwise, stop the algorithm.

3.2. If the set Ph is not empty, then select a physical element $ph \subseteq Ph$ for assigning using the greedy criterion K_{ph}, assign the request element e to the physical element ph, and redefine the values of its physical characteristics. If there are unexamined request elements, then go to Step 2. If the set R is not empty, then go to Step 1; otherwise, stop the execution of the algorithm.

The principle of operation of the limited search procedure is illustrated in Fig. 1.

The search depth is restricted by the given number n, which specifies the maximum number of physical elements among which the reassignment may be performed (e.g., if $n = 2$, then the assigned elements can be removed and reassigned not more than from two physical servers simultaneously). The left part of Fig. 1 illustrates the situation in which a virtual machine requiring four cores cannot be assigned for execution on any of the physical servers even though the total number of free cores is sufficient for its assignment. The virtual machines assigned to the servers R1, R2, and R3 are shaded, and the number of cores occupied by each machine is shown; the white box shows the number of free cores on the server. The reassignment of virtual machines on the servers R1 and R2 eliminates fragmentation, and the new virtual machines can be assigned as a result. That is, in this case, the search depth $n = 2$ is sufficient. The right part of Fig. 1 illustrates the situation in which the virtual machine requires five cores. In this case, the search depth two cannot help, and the new virtual machine cannot be assigned. However, if the search depth is $n = 3$, then the virtual machine can be assigned. That is, by increasing the search depth, we can improve the accuracy of the algorithm; however, its computational complexity is also increased in this case.

A description of the greedy criteria used in the proposed algorithm and a description of the algorithm used for the virtual channel assignments in the DC exchange network can be found in [1, 2].

4. INCREASING THE UTILIZATION OF PHYSICAL RESOURCES AND THE PERCENTAGE OF ASSIGNED REQUESTS

During a DC operation, virtual resources are removed from execution and new virtual resources are started. As an attempt to allocate a new virtual resource is made, the following situation can occur. The total amount of free physical resources complying with the SLA is sufficient for allocating a new virtual

Table 1. Efficiency of using the DC resources

	No restrictions on the migration time are imposed			Migration is not allowed		
	CPU utilization	RAM utilization	Percentage of assigned requests	CPU utilization	RAM utilization	Percentage of assigned requests
Test 1	0.97	0.33	100%	0.83	0.29	75%
Test 2	0.97	0.33	100%	0.83	0.29	75%
Test 3	0.89	0.49	85%	0.83	0.47	75%
Test 4	0.86	0.48	79%	0.83	0.47	75%
Test 5	0.85	0.47	77.5%	0.83	0.47	75%

resource, but there is no physical resource to which this virtual resource could be allocated in compliance with the SLA and constraints on the correctness of mapping. That is, in the course of the DC operation, the physical resources are fragmented, which decreases the efficiency of their use. The proposed algorithm can defragment the resources by moving the working virtual resources to other physical resources, which can increase the utilization of physical resources and increase the percentage of assigned requests. We investigated the influence of migration on the efficiency of DC operation in terms of utilization of physical resources and the number of assigned requests.

Table 1 shows the results of execution of the algorithm of mapping requests to physical resources of the DC that uses the migration plan for the working virtual resources in the following two cases:

1. Migration of the working virtual resources is not allowed.

2. There are no restrictions on the migration time of the virtual resources that changed their location.

Table 1 shows the percentage of assigned requests taking into account the earlier assigned and new requests; it also shows the utilization of computing resources.

If migration is allowed, the utilization of resources and the percentage of assigned requests in all the tests is higher than when migration is not allowed. The gain is the greater, the more often the limited search procedure is called and the more often it is successful.

However, if a restriction on the migration time is imposed, then the results produced by the algorithm are the worse, the less the allowed migration time. This is seen from the results produced by the algorithm presented in Table 2. Here the time is measured in seconds.

To demonstrate the influence of migration on the quality of the solution produced by the scheduling algorithm, we selected data for which the limited search procedure is frequently called. The physical resources are interconnected clusters of three servers and a switch. Virtual machines were assigned to these servers such that the assignment of the new virtual machine without calling the limited search procedure were not possible. These clusters are connected into a ring by channels that connect switches.

Table 3 presents the description of parameters used for the generation of input data—the number of servers in the topology, the intervals within which the number of CPUs and the amount of RAM in the virtual machines and physical servers may vary, and the capacity of the physical communication channels (measured in Mb/s). The percentage of assigned requests at the time when the algorithm begins executing is interpreted as follows. First, N requests are generated. Then M requests from these N requests are chosen and assigned. The percentage of assigned requests is M/N. The algorithm receives $N - M$ not assigned requests at its input.

5. IMPROVING THE PERFORMANCE OF VIRTUAL MACHINES

Modern cloud platforms do not guarantee the stable performance of virtual machines. The authors of paper [11] studied the performance of virtual machines in the Amazon cloud platform during a month. The results of investigating fluctuations of the basic characteristics affecting the performance of virtual machines are presented below.

3. CPU performance—24%;

4. RAM speed—10%;

5. Sequential disk read—17%; sequential disk write—19%;

6. Random disk read/write operations—9% to 20%.

The CPU and RAM performance were evaluated using the utility Ubench [12]. The rate of sequential read and write operations was measured in Kb/s. For the random read/write operations, the time was measured in seconds.

The cause of such large values of the coefficients of variation is as follows.

1. Heterogeneous infrastructure of DC physical resources [13]. Virtual machines can be assigned to servers with different types of processors because one cannot specify the type of processor in the request for a virtual machine.

Таблица 2. Efficiency of using the DC resources under restriction imposed on the migration time

	Test 1	Test 2	Test 3	Test 4	Test 5
Migration time = 500 arbitrary units					
CPU utilization	0.97	0.97	0.89	0.86	0.85
RAM utilization	0.33	0.33	0.49	0.48	0.47
Percentage of assigned requests	100%	100%	85%	79%	77.5%
Migration time = 400 arbitrary units					
CPU utilization	0.94	0.95	0.89	0.86	0.85
RAM utilization	0.325	0.328	0.49	0.48	0.47
Percentage of assigned requests	95%	97%	85%	79%	77.5%
Migration time и = 300 arbitrary units					
CPU utilization	0.9	0.91	0.89	0.86	0.85
RAM utilization	0.31	0.32	0.49	0.48	0.47
Percentage of assigned requests	87.5%	90%	85%	79%	77.5%
Migration time = 200 arbitrary units					
CPU utilization	0.87	0.87	0.89	0.86	0.85
RAM utilization	0.3	0.3	0.49	0.48	0.47
Percentage of assigned requests	82.5%	82%	85%	79%	77.5%
Migration time = 100 усл. ед.					
CPU utilization	0.85	0.85	0.86	0.86	0.85
RAM utilization	0.3	0.29	0.48	0.48	0.47
Percentage of assigned requests	77.5%	78%	80%	79%	77.5%
Migration time = 50 arbitrary units					
CPU utilization	0.85	0.84	0.84	0.85	0.85
RAM utilization	0.3	0.29	0.47	0.47	0.47
Percentage of assigned requests	77.5%	76%	76.2%	77.5%	77.5%

Таблица 3. Description of parameters for the generation of input data

	Physical servers				Virtual machines		Fraction of assigned requests
Test no.	Number	CPU	RAM	Channels	CPU	RAM	0.75
1	30	10	8192	50	1-5	1024	0.75
2	45	10	8192	80	1-5	1024	0.75
3	60	10	5120	150	1-5	1024	0.75
4	90	10	5120	200	1-5	1024	0.75
5	120	10	5120	300	1-5	1024	0.75

2. Resource overcommit. The total amount of resources needed for the virtual machines can be greater than the amount of resources available on the server. This situation is called overcommit [14]. In this case, the server resources are shared between the virtual machines, and these machines cannot permanently use a certain set of resources because the utilization of resources is controlled by the hypervisor [15];

3. NUMA architecture. Virtual machines may be assigned to servers with the NUMA architecture.

In the proposed algorithm, the resource overcommit is eliminated by introducing the constraint on the correctness of mapping that requires that the physical resource cannot be overloaded.

The performance of virtual machines placed in servers with the NUMA architecture can be improved by taking into account the server architecture. A virtual machine has the highest performance if all its cores have direct access to the same block of memory. In the proposed algorithm, this can be taken into account by introducing the matrix of distances between the cores.

Experimental studies showed that taking into account the NUMA architecture reduces the amount of assigned requests, but the performance of the virtual machines in this case increases. If the NUMA architecture is ignored, the performance of as many as 20% of virtual machines can decreases.

The heterogeneity of the infrastructure of DC physical resources can be taken into account by introducing the type of the requested processor into the set of SLA criteria. In the proposed algorithm, this will be taken into account when the correctness of mapping is checked; more specifically, it is taken into account when the matching of the types of the physical and virtual resources is checked.

CONCLUSIONS

The proposed algorithm for mapping the requests to DC resources increases the utilization of physical resources and the percentage of successfully assigned requests due to the elimination of resource fragmentation occurring in the course of the DC operation. The performance of virtual machines increases due to taking into account the architecture of servers in the construction of the mapping.

The typification of constraints between the characteristics of resources and SLA criteria and the capability of specifying the assignment policies for virtual machines allow us to configure the algorithm depending on the features of the cloud platform. For example, you can define the SLA criteria that must be specified in the request.

The algorithm combining greedy strategies with limited search is considered that makes it possible to choose a desired balance between the algorithm accuracy and its computational complexity.

FUNDING

This work was supported by the Russian Foundation for Basic Research, project no. 19-07-00614.

REFERENCES

1. Zotov, I.A. and Kostenko, V.A., Resource allocation algorithm in data centers with a unified scheduler for different types of resources, *J. Comput. Syst. Sci. Int.,* 2015, vol. **54**, no. 1, pp. 59–68.
2. Vdovin, P.M. and Kostenko, V.A., Algorithm for resource allocation in data centers with independent schedulers for different types of resources, *J. Comput. Syst. Sci. Int.,* 2014, vol. **53**, no. 6, pp. 854–866.
3. Rabbani, Md., Resource management in virtualized data center, Waterloo, Ontario, Canada, 2014. https://uwspace.uwaterloo.ca/bitstream/handle/10012/8280/Rabbani_Md.pdf?sequence=3
4. Benson, T., Akella, A., Shaikh, A., and Sahu, S., CloudNaaS: A cloud networking platform for enterprise applications, Madison, WI, USA. 2011. http://pages.cs.wisc.edu/~tbenson/papers/CloudNaaS.pdf
5. Ngenzi, A., Selvarani, R., and Nair, S., Dynamic resource management in cloud data centers for server consolidation, Bangalore-India, 2015. https://arxiv.org/ftp/arxiv/papers/1505/1505.00577.pdf
6. Ashraf, A. and Porres, I., Multi-objective dynamic virtual machine consolidation in the cloud using ant colony system, *Int. J. Parallel, Emergent Distrib. Syst.,* 2018, vol. 33, no. 1.
7. Ricci, R., Alfeld, C., and Lepreau, J., A solver for the network testbed mapping problem, *ACM Special Interest Group Data Commun., Comput. Commun. Review,* 2003, vol. **33**, pp. 65–81.
8. Mi, X., Chang, X., Liu, J., Sun, L., and Xing, B., Embedding virtual infrastructure based on genetic algorithm, *Proc. of the 13th Int. Conf. on Parallel and Distributed Computing, Applications and Technologies,* 2012, pp. 239–244.
9. Zhang, Z., Su, S., Lin, Y., Cheng, X., Shuang, K., and Xu, P., Adaptive multi-objective artificial immune system based virtual network embedding, J. Network Comput. Appl., 2015, vol. **53**, no. C, pp. 140–155.
10. Kostenko, V.A., Combinatorial optimization algorithms combining greedy strategies with a limited search procedure, *J. Comput. Syst. Sci. Int.,* 2017, vol. **56**, no. 2, pp. 218–226.
11. Schad, J., Dittrich, J., and Jorge-Arnulfo Quiané-Ruiz, J-A., Runtime measurements in the cloud: Observing, analyzing, and reducing variance, *J. Proc. VLDB Endowment,* 2010, vol. **3**, no. 1–2, pp. 460–471.
12. Unix benchmark Utility. http://phystech.com/download/ubench.html
13. Ou, Z. et al. Is the same instance type created equal? Exploiting heterogeneity of public clouds, *IEEE Trans. Cloud Comput.,* 2013, vol. 2, no 2.
14. Simao, J. and Veiga, L., Partial utility-driven scheduling for flexible SLA and pricing arbitration in clouds, *IEEE Trans. Cloud Comput.,* 2016, vol. **4**, no. 4, pp. 467–480.
15. Cherkasova, L., Gupta, D., and Vahdat, A., Comparison of the three CPU schedulers in Xen, *Newsletter ACM SIGMETRICS Performance Evaluation Review,* 2007, vol. **35**, no. 2, pp. 42–51.

Translated by A. Klimontovich

SPELL; OK