TECHNICAL TRANSACTIONS 4/2018
ELECTRICAL ENGINEERING
DOI: 10.4467/2353737XCT.18.060.8372
SUBMISSION OF THE FINAL VERSION: 4/4/2018

Sergii Telenyk (stelenyk@pk.edu.pl)
Department of Automatic Control and Information Technology, Faculty of Electrical and Computer Engineering, Cracow University of Technology

Oleksandr Rolik
Eduard Zharikov
Department of Automation and Control in Technical Systems

Yevhenii Serdiuk
Department of Computer-Aided Management and Data Processing Systems, National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute"

Energy efficient data center resources management using beam search algorithm

Energooszczędne zarządzanie zasobami centrów danych za pomocą algorytmu wyszukiwania promieniowego

Abstract
Modern data centres consume large amounts of power resulting in high levels of carbon dioxide emission. The data centre is a virtual environment in which the workload is performed by virtual machines. A widely used technique to decrease data centre power consumption is to consolidate the virtual machines using a minimal number of physical servers. The authors propose a two-stage method to solve the virtual machine consolidation problem in cloud data centres. The proposed method is programmed in C# to evaluate it and to perform modelling using Google cluster-usage traces. The proposed method enables powering off nearly fifty percent of the previously selected physical servers by using an acceptable number of migrations of virtual machines.

Keywords: cloud computing, virtualisation, virtual machine consolidation, local beam search

Streszczenie
Nowoczesne centra danych zużywają duże ilości energii, co powoduje emisję dwutlenku węgla. Centrum danych reprezentuje zwirtualizowane środowisko, w którym obciążenie jest obsługiwane przez maszyny wirtualne. Powszechną techniką zmniejszania zużycia energii w centrach danych jest konsolidacja maszyn wirtualnych przy użyciu minimalnej liczby serwerów fizycznych. Autorzy proponują dwustopniową metodę rozwiązania problemu konsolidacji maszyn wirtualnych w chmurowych centrach danych. Proponowana metoda jest zaprogramowana w języku C# w celu jej oceny i przeprowadzenia modelowania za pomocą śladów użycia klastrów Google. Proponowana metoda pozwala wyłączyć prawie 50% serwerów fizycznych wcześniej wybranych do wyłączenia przy użyciu dopuszczalnej liczby migracji maszyn wirtualnych.

Słowa kluczowe: chmura obliczeniowa, wirtualizacja, konsolidacja maszyn wirtualnych, lokalne wyszukiwanie promieniowe

1. Introduction

Providing a wide range of cloud information services is based on three main cloud service models: infrastructure as a service (IaaS); platform as a service (PaaS); software as a service (SaaS). The virtualisation of compute units, storage devices, networks, data warehouses, workplaces widely being implemented to more efficient use of data centre hardware.

The infrastructure as a service cloud service model allows to effectively use the hardware of the physical machine (PM) by virtualizing its local resources. In this case, the client is provided by part of the PM resources in the form of a virtual machine (VM). To provide modern information services, the client utilises one or more VMs of a specific configuration defined by the cloud service provider. Thus, there arises a set of tasks related to the management of virtual machines, physical servers, network interactions, storage devices, applications and other systems.

A separate VM or a container inside the VM is created for each instance of cloud application. In this paper, a VM with specific configuration is utilised for each application instance. Nowadays, the cloud service providers propose different configurations for VM instances. Moreover, based on certain business needs, the client can dynamically change the configuration of the VM or can customise load balancing, fault tolerance and backup capabilities. While the cloud service is working, the set of VMs is continuously changing by creating and deleting VMs to serve the changing workload. The number of VMs deployed on a separate PM can be changed in accordance with a client needs; therefore, some PMs become under loaded and, as a result, the power consumption of the data centre increases.

In order to use PM resources more effectively, the virtualisation tools provide the ability to migrate a VM from one PM to another. The impact of migration on the work of the VM is minimal, and VM migration is almost invisible to the client. It is known that the load on the physical servers that the VM is migrating between increases by 5–10 percent. One of the main tasks in managing the resources of the cloud data centre is to place VMs in such a way as to use fewer physical servers and reduce the number of virtual machine migrations.

The process of the redistribution of virtual machines among physical servers is referred to as the consolidation of the VMs. The solutions to the problem of VM consolidation have been presented in many publications [1], however, the virtualisation and cloud technologies available at that time are now outdated. In modern data centres, new hardware and system tools have been introduced and coexist with the previous generation of technologies. Thus, it is important to develop new algorithms and methods for managing the data centre resources in general, and the VM consolidation problem in particular.

In order to solve the VM consolidation problem, the authors propose a two-stage method based on the use of local beam search. The heuristics of the first and second stages of optimisation, the algorithm of the beam search, the evaluation function and the conditions for the completion of the algorithm are developed. Data relating to the Google cluster-usage traces was used for evaluation of the proposed method.

2. Related work

In recent years, several approaches and algorithms have been proposed to solve the VM consolidation problem with different objectives [1–3]. The problem of virtual machine consolidation is considered to be an optimisation problem with various objective functions. It can also considered to be a multi-objective optimisation problem [4, 5]. The peculiarity of this problem is the presence of a large number of environmental conditions and constraints. In addition, the complexity is increased during the formulation of the objective function, which includes several indicators that need to be optimised. The simple management algorithms such as first-fit, best-fit and their modifications (Eucalyptus [6], Microsoft [7], Google [8]) are used in industrial data centres with cloud-based infrastructures for simplicity and robustness. This is due to the demands of the stable running of client applications and services. Moreover, the administrators are involved in the automated process of managing data centre resources through the continuous monitoring of virtual machines and services.

Researchers have proposed solutions that use objective functions such as minimising power consumption, minimising service quality violations, minimising network traffic by eliminating bottlenecks, and maximising productivity. One of the main conditions that need to be satisfied for modern clusters is the ability of the management algorithm to operate online [3]. Furthermore, it is possible to use optimisation methods that work when certain cluster conditions arise. In this case, new tasks are scheduled for the PMs that are not involved in the consolidation process.

In addition to traditional heuristics and deterministic algorithms, local search algorithms such as evolutionary algorithms [9], ant colony optimisation algorithms [10], tabu search [11], and simulated annealing [12] are used to solve the problem of VM consolidation. In our opinion, it is more effective to use the local search technique at the second stage of optimisation after preparing the appropriate set of states through deterministic algorithms in order to improve the solution found at the first stage.

In this paper, it is proposed to apply a local search on a data set obtained at the previous stage of the optimisation in order to reduce the number of migrations of the VMs and to increase the number of PMs being switched to 'sleep' mode.

3. Description of the system model

Nowadays, most services for clients are provided by cloud data centres. Cloud data centres are complex systems that consist of server subsystems, storage subsystems, network subsystems, and engineering subsystems.

In this paper, the authors consider the problem of managing a cluster of PMs as a subsystem of a data centre using virtualisation to provide utilizations of VMs requested by clients. To compose the cluster, two approaches are used – the use of heterogeneous configuration and use of the homogeneous configuration of PMs. Both approaches have their advantages and disadvantages. It is impossible to avoid heterogeneous configurations on the scale of a data centre due to the evolution of the server core element and the new requirements of the users of IT infrastructure. This is why the configuration of PMs in each cluster may differ from the configuration of PMs in other clusters; however, the worst-case scenario is when the configuration of each PM in the cluster may differ.

The beam search algorithm is realized for a cluster, which consists of physical machines with different configurations. Each PM provides several resources for VMs deployed on it, such as: processor (CPU); memory capacity (RAM); access to the storage subsystem (IOPS); access to the network subsystem (NET). The number of VMs deployed on the PM depends on the available capacity of the PM resources, on the VM requirements for resources, on the time spent to run a task within the VM, and on the intensity of the flow of tasks. The number of VMs running on the PM is constantly changing. The change in the number of VMs occurs in the following states: the creation of a new VM, the removal of a VM, the migration of a VM.

The authors consider that the CPU and RAM resources of PMs may be allocated for VMs. However, the algorithm can be extended by taking into account other resources required by a VM. These two resources are chosen due to the use of input data from the Google cluster-usage traces (GCT) [13]. The machine events table and task events table from GCT have been used to study the operation of the beam search algorithm. Six thousand PMs were selected from the first table and seventy thousand tasks were randomly selected from the second table. For each PM from the machine events table, the following attributes are used: machine ID; CPU capacity; memory capacity. For each PM from the task events table the following attributes are used: task index within the job; machine ID; re-source request for CPU cores; resource request for RAM. The data in the tables are normalised with respect to the PM with the largest value of the capacity of each resource.

4. Description of the problem

The cluster is composed by a set P of M PMs and a set V of N VMs, $N, M \in \mathbb{N}$. During the management cycle of applying the beam search algorithm, the number of PMs and VMs does not change. Each task from the task events table is performed in a separate VM. In most cases the number of PMs and VMs between the management cycles may vary.

The required capacity of the j-th VM of resource k is denoted by $c_j^k \in (0,1]$, $k \in \{CPU, RAM\}$. The c_j^k is determined by the requirements of the task and is normalised with respect to the PM with the largest capacity of the resource k. The capacity of physical server i for resource k is denoted by $C_i^k \in (0,1]$. The C_i^k is determined by the type of PM and is normalised with respect to the PM with the largest capacity of the resource k.

In most cases the proposed model presumes the changes to the variables c_j^k and C_i^k during the lifetime of the VMs and the PMs. However, the model does not assume the changes to these variables during VM migration or during the management cycle. In the performed analysis, the constraints on the variables c_j^k and C_i^k are imposed by input data from the Google cluster-usage traces (GCT) [13]. Thus, the variables c_j^k and C_i^k are fixed during modelling.

The set P consists of a set A of physical machines that are defined to be switched to 'sleep' mode and a set B of physical servers that provide resources for VMs that will migrate from set A, $A \cup B = P$, from the PMs.

The migration of the virtual machine j to the physical machine i is denoted by $U_{ij} \in \{0,1\}$. The migration occurs if $U_{ij}=1$. Each VM from V has its own ID associated with the number j during the management cycle of the algorithm. Each PM also has its own ID associated with the number i during the management cycle of the algorithm.

5. The method of VM consolidation based on the beam search algorithm

Description of the main algorithm

The input data of the beam search algorithm is set A of PMs and set B of PMs that have available resources. Thus, it is possible to allocate additional tasks on PMs from set B in the form of VMs. The number of PMs and VMs does not change during modelling.

The idea of the algorithm is to take the i-th PM from set A and search for such a PM or PMs from set B, that is ready to accept the j-th VM migrating from the i-th PM. If it is possible to switch all or part of the PMs from set A to sleep mode, then the matrix of migrations U_{ij} is obtained as a result of the management cycle.

Description of the beam search algorithm

The first stage is to prepare the input data for the second stage. The first stage involves obtaining set A of PMs to be released from the VM and set B of PMs for determining the migration plan.

The second stage is performed for each PM from set A:

1) At each step, any VM assigned to the i-th PM is selected and the following variants of migrations are considered:
   a) migrating the given VM to another PM which has enough CPU and RAM resources
   b) checking the possibility of exchanging the given VM with another VM from another PM under the condition that another VM requires fewer resources (in such a way, states with a better rate are considered and it is possible to avoid an endless loop) and, as a result, the other PM will not be overloaded after this exchange
2) selecting n (which is a width of the beam) exchanges with the highest rating from all possible exchanges
3) completing the search if the i-th PM has been released from the virtual machines or if it is impossible to construct new states because there are no options left to implement the admissible exchange a) or b)

The comparison of the states is performed using criterion (1):

$$J = \sum_{i=1}^{m} u_i^2 + \sum_{i=1}^{n} f_i^2 \tag{1}$$

where:
u_i – the number of resources used on the *i*-th PM,
f_i – the number of available resources on the *i*-th PM,
m – the number of PMs in set *B*,
n – the number of PMs in set *A*.

Thus, the algorithm gradually reduces the number of resources used on each PM belonging to set *A*, and utilises the resources of PMs from set *B*.

The terms of completion of the algorithm

To complete the algorithm, the following conditions must be met:

1) the exhaustion of set *A*
2) the inability to migrate all VMs from a specified number of PMs, denoted as Th_A

If the second condition is not applied, the search cycles may run too long in comparison to the time spent on VM creation or compared to the time spent on VM migration when the VM has average requirements for memory resources. To determine Th_A, it is suggested to apply a heuristic that takes into account a certain percentage of physical servers from set *A* but not less than α physical servers. In the proposed algorithm, the values are accepted as follows, Th_A=0,05, α=10. Using small values of α, the algorithm skips a significant number of PMs that could have been switched to sleep mode. Thus, as a result of the completion of the algorithm, those PMs were not released from the VMs running on them. This termination criterion is introduced to reduce the execution time of the algorithm.

Definition of input data (description of the first stage of the method)

The main task on the first stage of the method is to obtain a set of PMs as expectants to be switched to 'sleep' mode. It is proposed to obtain set *A* of PMs using two techniques – the lower boundary technique (LB) and the threshold of available resources technique (TAR). Let us consider each of the techniques in more detail.

The LB technique is used to determine the number of physical servers that cannot be switched to sleep mode after migration of all the VMs that run on them. Such an idea arises from the hypotheses used in the algorithm for determining the lower boundary.

The search of the lower boundary is performed as follows:

1) calculating the average resource volume for all PMs which host VMs,

   $\frac{1}{Q}\sum_{i=1}^{Q} C_i^k, 0 < Q \le M$ – the values for each resource *k* are calculated separately

2) calculating the sum of necessary resources for run of all the existing VMs for each resource separately, $\sum_{j=1}^{R} c_j^k, 0 < R \le N$

3) calculating the ratio of the required resources to the average resource volume and rounding off the values to a larger integer – the values for each resource *k* are calculated separately. The highest number obtained in this step will be the lower boundary
4) sorting the PMs in one of the following ways:
   a) by the number of resources of the PM, then by the ratio of the used resources to the number of working VMs
   b) by the number of resources of the PM, then by the number of assigned tasks, then by the ratio of used resources to the number of working VMs

As a result of detailed study of the algorithm, it has been revealed that option 'b' is significantly more effective. By applying option 'b' it was possible to switch 82 PMs to sleep mode; while applying option 'a' only 33 PMs could be switched to sleep mode using the same data set. As a result, the difference between the number of used PMs and the resulting lower boundary is calculated. The obtained number is the number of PMs of set *A* that may potentially be switched to the sleep mode. These PMs are the first in the sorted list of set *A*. The rest of the PMs go to set *B*.

The idea behind the TAR technique is the formation of set *A* in such a way that set *A* only contains PMs in which the total number of unused resources exceeds the amount of resources of one of the PMs in the cluster.

The threshold of available resources is denoted by β. Building set *A* using the threshold of available resources is performed as follows:

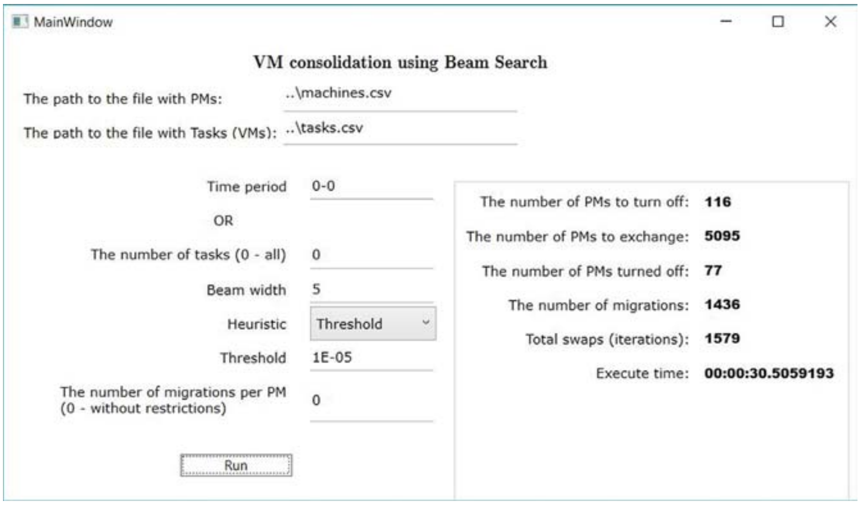
1) choosing the value β
2) selecting PMs which have more available resources than value β on each resource *k*
3) calculating the amount of free resources of selected PMs for each resource separately
4) sorting selected PMs similar to option 'b' from the lower boundary method
5) passing through sorted list of set *A*.

Until the sum of resources is greater than the volume of resources of the current PM, the algorithm deducts this volume from the sum, adds the current PM to the set *A* and proceeds to the next PM. Otherwise the algorithm stops passing through the sorted list of set *A*. Thus, set *A* is built. Set *A* contains PMs which must be switched to sleep mode. On the other hand, set *B* contains PMs for the exchange of virtual machines.

6. Evaluation of simulation results

The trace-driven experiments were conducted on fragments of the GCT [13] input data set. Data for testing and studying the proposed algorithm was selected from the GCT for a specific time range for which data was collected on a real cluster. This was necessary to take into account all data for the same period of time in the GCT tables. Seventy thousand tasks with specific requirements for resources *k* were randomly selected from the data set. Each created VM corresponds to one task from the data set. Six thousand physical servers were selected from the data set where 5832 servers host VMs, and 168 servers do not host VMs, so they are available for hosting VMs. The width of the beam for the algorithm was chosen to be 5. The study of the influence of the beam width on the performance of the algorithm was performed in [14]. According to study [14], it is recommended to choose a beam width in the range 5-8, depending on the conditions of the method applications.

The proposed model has been implemented in C#. The simulations were conducted on the computer with an Intel i7-3632QM processor and 8 GB of RAM running Windows 10 Pro 64bit. The interface of the simulation program is shown in Fig. 1.



The path to the file with PMs: ..\machines.csv
The path to the file with Tasks (VMs): ..\tasks.csv

Time period	0-0
OR	
The number of tasks (0 - all)	0
Beam width	5
Heuristic	Threshold
Threshold	1E-05
The number of migrations per PM (0 - without restrictions)	0

The number of PMs to turn off:	**116**
The number of PMs to exchange:	**5095**
The number of PMs turned off:	**77**
The number of migrations:	**1436**
Total swaps (iterations):	**1579**
Execute time:	**00:00:30.5059193**

Fig. 1. The program interface

The first series of experiments were conducted without any limitation on the maximum number of simultaneous migrations per PM. Table 1 presents the simulation results of VM consolidation problem solving using the beam search algorithm with the threshold of available resources technique for different values of β. Table 2 presents the simulation results of VM consolidation problem solving using the beam search algorithm with the lower boundary technique.

The number of PMs switched to the sleep mode is denoted by PM_{sleep}. The number of VM migrations is denoted by VM_{mig}. The duration of solving the VM consolidation problem by the proposed algorithm is denoted by T. The number of PMs switched to sleep mode and the number of VM migrations are chosen as qualitative indicators of the proposed algorithm.

The effect of the threshold value β on the qualitative indicators of the algorithm performance is significant, as shown in Fig. 2 and Fig. 3. The dependence between the threshold value and the qualitative indicators of the algorithm performance was almost linear. The smaller the threshold, the more PMs are processed by the algorithm.

For β=0, some of the PMs (namely, those in which at least one resource is fully loaded) are not processed by the algorithm because when checking the availability of resources, strict inequality is used. For β <0,0002, the algorithm with the TAR technique begins to work more efficiently and at the limit values, it works more effectively than the LB technique.

Table 1. Simulation results of solving the VM consolidation problem using the TAR technique

β	B	A	PM_{sleep}	VM_{mig}	T, s
0.005	199	3	0	0	0.034
0.004	299	3	0	0	0.052
0.003	1098	9	0	0	0.279
0.002	1393	10	0	0	0.38
0.001	1868	28	13	206	1.421
0.0009	1943	31	15	238	1.817
0.0008	2002	39	22	352	2.465
0.0007	2081	43	25	403	3.067
0.0006	2204	48	29	470	3.321
0.0005	2302	55	32	502	4.446
0.0004	2407	63	37	566	5.884
0.0003	2565	73	43	714	6.784
0.0002	2764	82	49	836	8.68
0.0001	3706	95	57	956	12.8
0.00005	4323	106	66	1128	18.422
0.00001	5095	116	77	1442	29.995
0	5328	125	82	1447	33.15

Table 2. Simulation results of solving the VM consolidation problem using the LB technique

B	A	PM_{sleep}	VM_{mig}	T, s
5707	125	81	1435	32.13

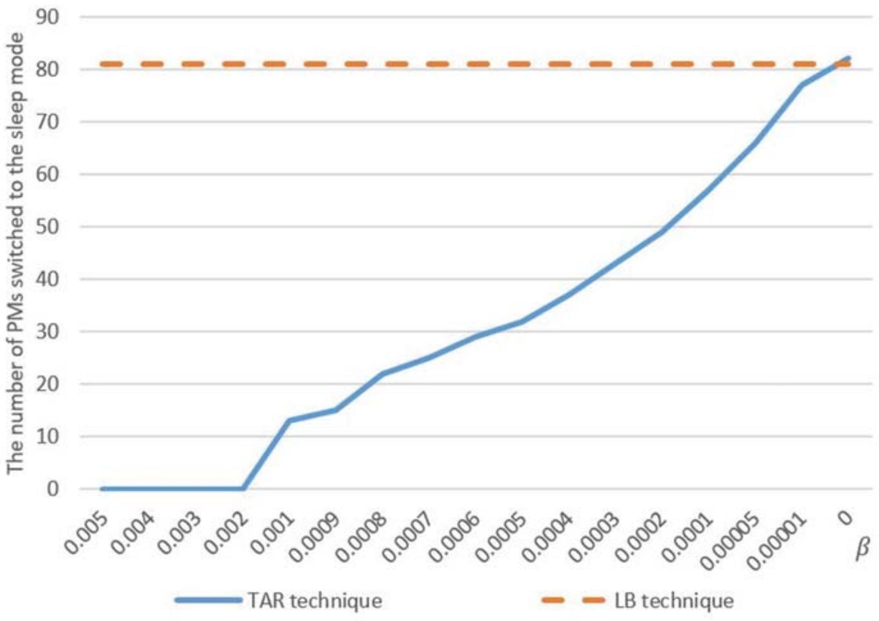


Fig. 2. The number of PMs switched to sleep mode

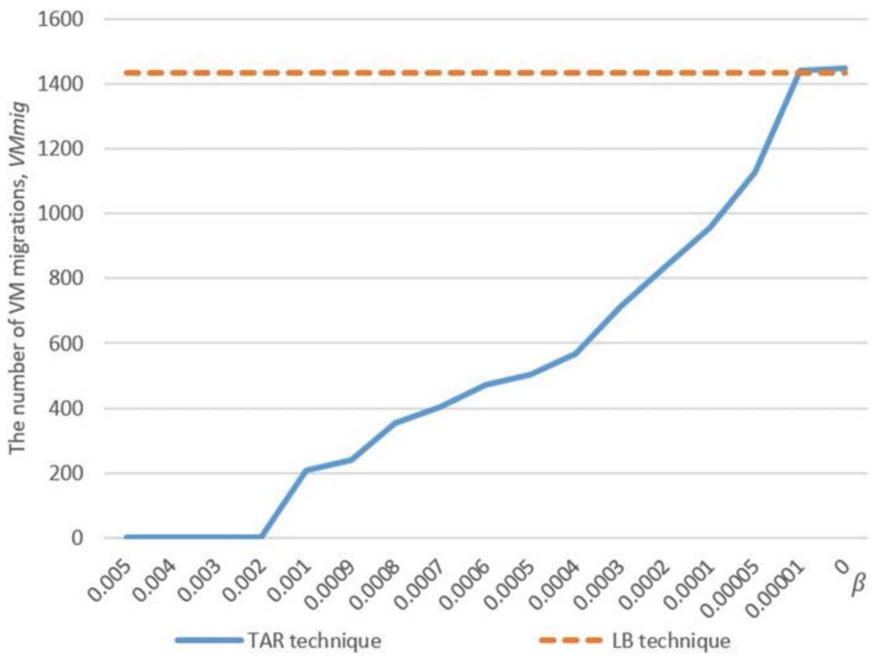


Fig. 3. The number of VM migrations

The advantage of using the TAR technique is the ability to adapt to the state of the cluster, taking into account such indicators as the state of other resources, the time of migration of the VM, and the intensity of incoming requests for the creation of new VMs.

An acceptable solution to the VM consolidation problem is possible when restrictions on the number of migrations as well as on the optimisation time are applied. The results show that the LB technique may take around 32 seconds to perform. Such performance may be impossible in production environments with dynamic VM management because the management system must take into account the dynamics of the new virtual machine deployment processes and the migration of virtual machines simultaneously.

To reduce the number of VM migrations, it is recommended to limit the maximum number of simultaneous migrations per PM. In production, hypervisors limit the maximum number of simultaneous migrations per PM, per network resource, and per datastore to provide a near 100 percent guarantee of no SLA violation during VM migrations [15]. On some workloads that are RAM intensive or on older hardware, a high number of simultaneous migrations (more than the recommended number) could saturate a system resource and result in downtime [16].

The second series of experiments were conducted with a limitation on the maximum number of simultaneous migrations per PM. This limitation starts to have an impact upon simulation results when the limit of simultaneous migrations per PM is 2. In production, the hypervisors are usually configured to a value in the range of 4-16, depending on the hardware [15]. The results show that the impact of this limitation appears when the number of PMs in set *B* is too small to accept VMs from the PMs in set *A*. In such cases, the PM_{sleep} decreases by around 40 percent.

Six data sets with different numbers of PMs and VMs were created using Google cluster-usage traces. The PMs and tasks were taken from different time ranges of the traces for an all-round check of the algorithm's operation. The obtained results show the same performance indicators as were obtained with the first configuration of the simulation environment.

7. Conclusion

In this paper, a two-stage method based on the beam search algorithm has been proposed in order to manage the process of virtual machine consolidation. The primary results of this paper are: the algorithm of the beam search for solving the VM consolidation problem under certain conditions; the heuristics of the first and the second stages of the algorithm; the evaluation function; the conditions for the completion of the algorithm; the consideration of constrains on the maximum number of simultaneous migrations per PM.

To evaluate the proposed method, the experiments were conducted on six fragments of the Google cluster-usage traces as an input data set. An analysis of the proposed techniques of the lower boundary and the threshold of the available resources was performed. As a result, it has been revealed that the technique with the threshold of available resources showed more efficient results while solving the VM consolidation problem. Moreover, the technique with the threshold of available resources allows adaptation to the state of the cluster, taking into account the state of other resources, the time of migration of the VM, the intensity of incoming requests for the creation of new VMs, and the limitation on the maximum number of simultaneous migrations per PM.

The benefit of the work is in the development of a two-stage method based on the algorithm of local beam search for solving the VM consolidation problem of the cloud-based data centre. The proposed method allows the turning off of almost 50 percent of the physical servers that are potentially configured to switch to sleep mode due to the use of a valid number of virtual machine migrations.

References

[1] Lopez Pires F., Baran B., *A virtual machine placement taxonomy,* [in:] Proc. of the 15th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid), 2015, 159–168.

[2] Ahmad R.W., Gani A., Hamid S.H.A., Shiraz M., Yousafzai A., Xia F., *A survey on virtual machine migration and server consolidation frameworks for cloud data centers,* Journal of Network and Computer Applications, Vol. 52, 2015, 11–25.

[3] Telenyk S., Zharikov E., Rolik O., *An approach to virtual machine placement in cloud data centers,* [in:] Proc. of the 2016 International Conference Radio Electronics & Info Communications (UkrMiCo) 11–16 September, Kyiv, Ukraine, 2016, 1–6.

[4] Lopez Pires F., Baran B., *Multi-objective virtual machine placement with service level agreement: A memetic algorithm approach,* [in:] Proceedings of the 2013 IEEE/ACM 6th International Conference on Utility and Cloud Computing, IEEE Computer Society, 2013, 203–210.

[5] Saber T., Ventresque A., Brandic I., Thorburn J., Murphy L., *Towards a Multi-objective VM Reassignment for Large Decentralised Data Centres,* 2015 IEEE/ACM 8th International Conference on Utility and Cloud Computing (UCC), Limassol, 2015, 65–74.

[6] *Eucalyptus community,* http://open.eucalyptus.com (access: 10.12.2017).

[7] Lee S., Panigrahy R., Prabhakaran V., Ramasubrahmanian V., Talwar K., Uyeda L., Wieder U., *Validating heuristics for virtual machines consolidation,* Microsoft Research, MSR-TR-2011-9, 2011.

[8] Sharma B., Chudnovsky V., Hellerstein J.L., Rifaat R., Das C.R., *Modeling and synthesizing task placement constraints in google compute clusters,* In Proceedings of the 2nd ACM Symposium on Cloud Computing (SOCC), 2011.

[9] Mark C.C., Niyato D., Chen-Khong T., *Evolutionary optimal virtual machine placement and demand forecaster for cloud computing,* [in:] IEEE International Conference on Advanced Information Networking and Applications (AINA), 2011, 348–355.

[10] Gao, Y., Guan, H., Qi, Z., Hou, Y., Liu, L., *A multi-objective ant colony system algorithm for virtual machine placement in cloud computing,* Journal of Computer and System Sciences, Vol. 79, No. 8, 2013, 1230–1242.

[11] Ferreto, T., De Rose C., Heiss, H.U., *Maximum migration time guarantees in dynamic server consolidation for virtualized data centers,* [in:] Euro-Par 2011 Parallel Processing, Springer 2011, 443–454.

[12] Wu Y., Tang M., Fraser W., *A simulated annealing algorithm for energy efficient virtual machine placement,* [in:] IEEE International Conference on Systems, Man, and Cybernetics (SMC), 2012, 1245–1250.

[13] Reiss C., Wilkes J., Hellerstein J.L., *Google cluster-usage traces: format+ schema,* Google Inc., Mountain View, CA, USA, Technical Report, 2011.

[14] Zharikov E., *Managing data center resources using heuristic search,* Problems in programming, Vol. 4, 2017, 16–27.

[15] *Limits on Simultaneous Migrations,* https://docs.vmware.com/en/VMware-vSphere/6.0/com.vmware.vsphere.vcenterhost.doc/GUID-25EA5833-03B5-4EDD-A167-87578B8009B3.html, last accessed 2018/01/10 (access: 24.12.2017).

[16] Telenyk S., Zharikov E., Rolik O., *Consolidation of Virtual Machines Using Stochastic Local Search,* Advances in Intelligent Systems and Computing, Springer, 2017, 523–537.