J Supercomput (2017) 73:2001–2017
DOI 10.1007/s11227-016-1900-y

Energy-aware framework with Markov chain-based parallel simulated annealing algorithm for dynamic management of virtual machines in cloud data centers

Mehdi Rajabzadeh[1] · **Abolfazl Toroghi Haghighat**[2]

Published online: 20 October 2016

Abstract Significant savings in the energy consumption, without sacrificing service level agreement (SLA), are an excellent economic incentive for cloud providers. By applying efficient virtual Machine placement and consolidation algorithms, they are able to achieve these goals. In this paper, we propose a comprehensive technique for optimum energy consumption and SLA violation reduction. In the proposed approach, the issues of allocation and management of virtual machines are divided into smaller parts. In each part, new algorithms are proposed or existing algorithms have been improved. The proposed method performs all steps in distributed mode and acts in centralized mode only in the placement of virtual machines that require a global vision. For this purpose, the population-based or parallel simulated annealing (SA) algorithm is used in the Markov chain model for virtual machines placement policy. Simulation of algorithms in different scenarios in the CloudSim confirms better performance of the proposed comprehensive algorithm.

Keywords Energy consumption · SLA violation · Virtual machine placement · Parallel simulated annealing · Markov chain

1 Introduction

In the past few years, the use of Cloud Computing infrastructure to run business and consumer-based IT applications has increased rapidly [1].

✉ Mehdi Rajabzadeh
mehdi.rajabzadeh@srbiau.ac.ir

1 Science and Reaserch Branch, Islamic Azad University, Tehran, Iran

2 Qazvin Branch, Islamic Azad University, Qazvin, Iran

The ever-increasing cloud computing has been resulting in ever-increasing energy consumption and, therefore, overwhelming electricity bills for data centers [2,3]. Energy consumption in Cloud data centers will continue to grow rapidly unless advanced energy-efficient resource management solutions are developed and applied [3,4].

By applying efficient VM placement and consolidation algorithms, Cloud providers are able to enhance energy efficiency [5]. Dynamic VM consolidation approaches leverage dynamic nature of Cloud model, both servers and their VMs are periodically monitored. To minimize the number of active servers and maximize the quality of delivered services, whenever a server goes in under or overload state, its VMs are reallocated using live VM migration [6,7].

In addition, the problem of low server utilization is exacerbated by narrow dynamic power ranges of servers: even completely idle servers still consume up to 70 % of their peak power [8,9]. Therefore, keeping servers underutilized is highly inefficient from the energy consumption perspective. This problem increases the rate of cost to the efficiency. Therefore, we should take servers to low-power mode to the extent possible with the help of virtualization and migration of virtual machines from low-load servers and integration of them into other servers [10].

In this paper, to improve the process of management of resources and to reduce energy consumption, we propose an energy-aware framework. In the proposed framework, the main problem has been divided into smaller sections. In each section, a new algorithm has been presented, or the available algorithms were developed. One of the points that differentiate our approach from the other methods is that in the previous methods, overloaded servers have been considered as the basis for virtual machine migration decisions. From our perspective, however, overload per se is not regarded as an acute problem, and it should not be addressed unless SLA violation is likely. Therefore, we base migration on criticality of the conditions by defining critical conditions[1] for servers.

The rest of the paper is organized as follows: in Sect. 2, we discuss about related works. Section 3 gives the details of the proposed Comprehensive algorithm, step by step. In Sect. 4, we perform experiments and compare our work with some other popular algorithms. At the end of the paper, besides general conclusion of the paper, solutions have been presented for continuation of the work in future.

2 Related works

VM consolidation techniques have been very attractive to reduce energy consumption and increase resource utilization in virtualized data centers. Therefore, a lot of work has been done in this area and depending on the modeling techniques used, different problem solving techniques are proposed. The goal has been to improve the energy efficiency of system, while avoiding VM performance degradation [4,5]. This

[1] In this paper, critical conditions are conditions defined based on the system status from the aspect of processing power and available memory value. On that basis, an overloaded system with poor processing power or low memory is in critical conditions.

problem has usually been considered as a multi-dimensional Bin-Packing problem. In this regard, several algorithms have been proposed with different objectives such as minimizing the number of active servers [10–13].

When two applications cannot be assigned to the same target server, item–item incompatibility constraints occur. Bin–item incompatibility constraints arise when a given application cannot be moved to a particular server. For consolidating servers with these constraint types, authors [14], propose a placement solution with a two-stage heuristic algorithm.

Chen et al. [7] build up a two-objective optimization model and propose a virtual machine placement algorithm based on matrix transformation. Two-objective optimization model takes the balance of different dimensions in minimizing number of physical machines and resource utilization into consideration. The two-objective optimization model first sets up corresponding virtual machine requests queue matrix, cluster matrix, and the corresponding initialization placement matrix by considering virtual machine placement algorithm based on matrix transformation and then looks for the best result meeting the two goals by the corresponding matrix transformation.

In [3], Lee et al. first attempted to solve the VM placement problem using the simulated annealing to achieve energy saving. On this basis, Dhingra and Paul [15] use optimized simulated annealing for VM placement process, looking for energy consumption by random iteration better physical host servers. It ensures virtual machines SLA server degree, increases the resources utilization, and reduces the energy consumption.

Tang and Pan [16] propose a virtual machine placement algorithm based on genetic algorithm. Such algorithms consider the energy consumption of data centers physical host and the energy consumption of network communication by generating a random initial population and then use multiple gene mutations to look for minimum energy consumption of physical hosts and network communication. However, the algorithm just considers one network structure and has high computational complexity and low allocating efficiency.

In [10], Beloglazov et al. proposed a novel approach that for any known stationary workload and a given state configuration optimally solves the problem of host overload detection by maximizing the mean inter-migration time under the specified QoS goal based on a Markov chain model. They heuristically adapt the algorithm to handle unknown non-stationary workloads using the Multisize Sliding Window workload estimation technique. In this work, they focused only on the problem of host overload detection.

Asyabi et al. [6] proposed an energy-aware heterogeneous VM scheduling algorithm along with performance. They used a set of objective functions in terms of fitness metric and placed a set of VMs on a set of servers aiming to minimize the total energy consumption in the entire data center.

Chen et al. [7] proposed three algorithms to solve the virtual machine placement problem in large scale. First, they propose a physical machine (PM) classification algorithm by analyzing pseudotime complexity and find out an important factor (the number of physical hosts) that affects the efficiency, which improves running efficiency through reduction number of physical hosts; second, they present a VM placement optimization model using multi-target heuristic algorithm and figure out the positive

and negative vectors of three goals using matrix transformation so as to provide the mapping of VMs to hosts by comparing distance with positive and negative vectors such that the energy consumption is saved, resources wastage of occupied PM is lowered, multi-dimensional resource utilization is optimized, and the running time is shortened. Finally, they consider the poor placement efficiency problem of large-scale virtual serial requests and design a concurrent VM classification algorithm using the K-means method.

In [8], an adaptive fuzzy threshold-based algorithm has been proposed to detect overloaded and underloaded hosts. The proposed algorithm generates rules dynamically and updates membership functions to adapt to changes in workload.

Our work is different from the previous works, since we propose a redesigned energy-aware framework with comprehensive algorithm for VM management to achieve a better energy–performance tradeoff. In this work, we will respond to four questions posed in the Sect. 3, step by step.

3 Proposed algorithm

Extremely high energy costs are not only due to the large number of computing resources and the lack of hardware performance. Data obtained from 5000 servers during 6 months indicate that the servers have not been usually vacant; however, their productivity has been rarely high and often work with 10–50 % of their capacity. This problem increases the rate of cost to the efficiency. Therefore, we should take servers to low-power mode to the extent possible with the help of virtualization and migration of virtual machines from low-load servers and integration of them in other servers. In all methods of resource management, according to the objectives that are considered, virtual machines are transferred from a source machine to a destination machine. By migration of a virtual machine, which is stationed in a low- or overloaded host, to another host, the possibilities of improving energy consumption and utilizing resources to provide the quality of desired service are created [17].

In this paper, to improve the process of management of resources and to reduce energy consumption, we provide some answers to the following questions by breaking the main problem into four sections.

1. *Detecting the host in a critical condition* what is time of a migration from a machine with a critical condition?
2. *Virtual machine selection* which virtual machines must migrate from the hosts?
3. *Virtual machine placement* where would be destination of the migrated virtual machines?
4. *Low-load host detection* Which of the low-load hosts are the best selections to be shutdown to save energy?

The proposed comprehensive algorithm performs all the steps in distributed form and functions in centralized form only in the virtual machine placement that should have a global vision.

A high-level view of the proposed framework can be seen in Fig. 1. Local management section has been implemented on the hosts and global information management section holds all the hosts.

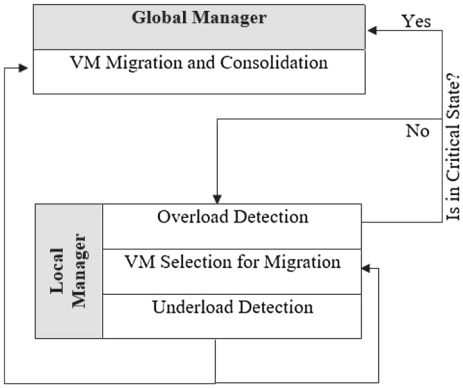

Fig. 1 An overview of the proposed framework

Splitting the problems improves the scalability of the system, as the host underload or overload detection and VM placement algorithms are executed locally by each compute host. It follows that the software layer of the system is tiered comprising local and global managers. The local managers objective is the continuous monitoring of the nodes condition, and detecting host underload, overload or critical conditions. Based on the decisions made by the local managers, the global manager issues VM migration commands to optimize the VM placement.
In what follows, the algorithms used in each section are explained.

3.1 Detection of a server with critical condition

We know that each live migration has additional overhead costs and according to conducted studies, could take up to 10 % of the processing efficiency. In addition, it wastes the bandwidth. Thus, in the cloud data centers with thousands of hosts, performing unnecessary migrations disrupt the balance of the entire system and will have a negative impact on the efficiency of running applications [10]. In other words, in cloud environments, a proper dynamic management approach, based on the performance of hosts, should have the best decision for migration of virtual machines to be able to prevent unnecessary migrations. Due to the heterogeneity of systems in the data center, considering a fixed value as the overload threshold cannot be much appropriate [10]. As an example, a host with less number of CPU cores is more likely to go to the overloaded mode by adding a virtual machine; however, at the same conditions, a host with greater number of cores is less likely to go to the overloaded mode by the addition of the virtual machine. That is why each machine should be considered as overloaded with regard to its specific conditions.

In this paper, the LR[2] algorithm is used to detect whether the CPU is overloaded. Moreover, the extent to which the main memory is occupied is investigated as another important factor alongside the CPU usage level. It is possible that an overloaded machine, regarding the high number of cores, is less likely to be in trouble in terms

[2] Local Regression.

```
If (host is overload and single-core) or (host is overload and low memory) then
        //state of the host is critical
        VmSelectionforMigration ();
Else
        //state of the host is not critical
        Do nothing ();
```

Fig. 2 The pseudo-code for detection of a server with critical state

of the percentage of CPU efficiency; however, a large amount of its main memory is occupied in which case there is the possibility of SLA violation and it should be considered as a critical state.

Therefore, we consider being overloaded a necessary condition but not a sufficient condition for migration. This means that if a host with the mentioned definitions is detected to be overloaded, in case it has the possibility of SLA violation, one must take action to migrate a virtual machine or machines from it. The modes in which the possibility of SLA violation is likely can be stated as pseudo-code as shown in Fig. 2.

3.2 Selection of a virtual machine for migration

The following points should be noted in selecting a virtual machine for migration:

- The higher its rate of usage is from the processor (CPU Usage), with migration, a greater percentage of the processor is released.
- The lower the rate of usage is from the main memory (Allocated Memory), the less it is dependent on the source machine for migration, and the migration is done at a lower cost. At the same time, migration time[3] is reduced, according to the Eq. 1, and the declining efficiency is more tolerable, according to the Eq. 2.

$$T_{mi} = \frac{M_i}{B_i} \tag{1}$$

$$U_{di} = 0.1 \int_0^{t_0+T_{mi}} U_i(t)\mathrm{d}t \tag{2}$$

M_i is the amount of memory used by VM_i, U_{di} is the total performance degradation by VM_i, t_0 is the initial time of migration, T_{mi} is the time taken to complete the migration, $U_i(t)$ is the CPU utilization by VM_i, and B_i is the available network bandwidth. To implement this idea, it is acted as follows:

1. A list of critical servers is created which is shown with CS.[4]

[3] This has been considered with the assumption that the VM images and data are stored in a shared storage in the network. In fact, there is no need for copying the storage space concerning the VM with this assumption. This has been considered for further simplification of the job. On the other hand, half of the bandwidth has been considered in the simulations made for migration and the other half for VM communication.

[4] Critical server.

2. A list of virtual machines of each host is formed and sorted out in descending order according to CU.[5] If this parameter is equal for some virtual machines, they are sorted out based on less use of the main memory.
3. The first virtual machine is removed from the list and if it leads the server to get out of the critical state, it is selected for migration and placed on the migration list. This makes the server to get out of the critical state, but it still keeps it at the upper threshold. However, if the server is still in critical state, we have to select the next virtual machine form the list and transfer it to the migration list. In fact, this process is repeated until the server is not in the critical state any longer.

3.3 Placement policy of virtual machines

The placement problems of VMs on servers have always been a huge challenge at the cloud data center.

VM placement problems in cloud data center are a physical resources mapping process of VMs to servers according to the reasonable allocation rules. This stage is similar to finding a solution for the Bin-Packing problem. In fact, the placement is intended to be performed in a way that the number of active servers is minimal.

In this section, we used Markov chain model with population-based or parallel simulated annealing meta-heuristic algorithm for optimization. In fact, we have turned the PABFD policy presented by Beloglazov to MCSA-PABFD [10]. SA algorithm is an unbound algorithm that is used for difficult designs.

A combination of SA and parallelism is used to help increase the SA power and faster resolution of problems. For this purpose, the population-based SA algorithm is used in the virtual machines placement policy.

In this policy, it has been taken into consideration that migration should not lead to overloading to replace the virtual machines selected for migration since such migration bears two major drawbacks: first, it increases the likelihood of SLA violations at the destination and, second, it increases the probability of another migration at the destination, for which it consumes energy. Thus, interquartile range which is one of the measures of dispersion and covers the distance between the first quarter and the third one is employed to solve the problem (as shown in Fig. 3). The way it is implemented will be described in the algorithm stages.

Proposed algorithm uses the following mechanisms for virtual machines placement:

Step 1 Generating an initial population and evaluation of it: a list of servers should be established for the primary population to be formed. This list should not contain the overloaded, low-loaded and turned-off servers. Interquartile range is used to prepare the lists. Interquartile range of a sample represents a distance containing observations interval. In this way, too large or too small are deleted. The interquartile range, as a measure of dispersion, is preferred over the variance.

To form the list, all turned-on servers will be ascending ordered in terms of overloading states and those servers located within the interquartile range are added to the list. In this way, overloaded and low-loaded servers are out of the selection area.

[5] CPU Usage.

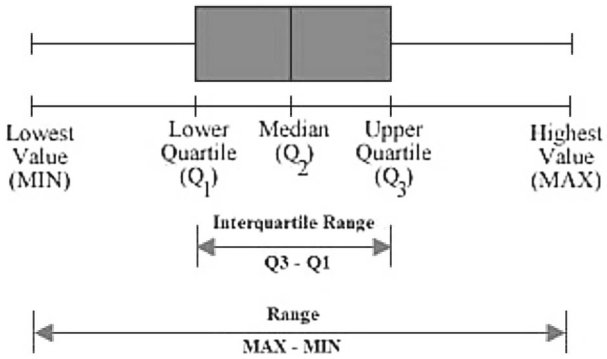

Fig. 3 Interquartile range of a sample

Then, a list of the virtual machines selected to migrate will be formed. (The number of server list members and the list of virtual machines must be equal; otherwise, interquartile primary servers will be repeated due to being low loaded.)

A few random arrangements are selected from the list of servers and considered as the primary population. Then, this population (lists) will be evaluated. The normalized evaluation criterion is total increase in energy expenditure of servers. The less it is, the better the other would be.

$$F = \frac{\sum_{i=1}^{n} \acute{E}_i - \sum_{i=1}^{n} E_i}{\sum_{i=1}^{n} \acute{E}_i} \tag{3}$$

In the Eq. 3, E is the energy before the attribution of virtual machines and $\acute{E}$ is the energy after their attribution.

Step 2 Determining the best found answer: the best answer from the set of lists is selected and recorded.

Step 3 Setting the initial temperature $T = T_0$. One of the most important parameters rarely discussed in the simulated annealing method is the initial value of the temperature parameter. Many researchers have used a large value as the initial value of the temperature-like parameter. But, if the initial value of the parameter temperature is too large, the algorithm is faced with a large number of relative minimums which result in a great number of upward motions (bad answer) to be accepted at the time of leaving them due to the random nature of the algorithm. On the other hand, if the initial value of the temperature parameter is too small, the algorithm will quickly converge to a local minimum and will not have the power to leave it. Therefore, an optimum value must be selected for the initial value of the temperature-like parameter. In this study, we will provide a model for estimating the initial value of the temperature parameter based on the previous conducted studies.

Before presenting the model, we should introduce a parameter called acceptance ratio. The acceptance ratio, which is defined as the ratio of the number of accepted changes to the total number of changes proposed in each Markov chain, can be estimated using the Eq. 4:

$$\Psi_k = \frac{m_1 + m_2 \times \exp\left(\frac{-\triangle f_{m_2}}{T_k}\right)}{m_1 + m_2} \tag{4}$$

In this equation,m_1 is the number of changes that $\Delta f \leq 0$ and m_2 is the number of changes that $\Delta f > 0$.

The average changes in the objective function are shown with $\Delta f m_2$ along the m_2. Moreover, T_K is the current value of the temperature parameter. By taking into account a relatively large value for the acceptance coefficient like $\Psi_K = 0.99$ of the initial value of temperature parameter, T_0 is determined using the Eq. 5 [19,20]:

$$T_0 = \frac{-\triangle f_{m_2}}{\ln\left(\Psi_1 + \frac{m_1}{m_2}(\Psi_1 - 1)\right)} \tag{5}$$

Step 4 Perfuming the steps 5–8 to the specified number (inner algorithm loop)

Step 5 For each of the members of the population, a certain number of neighbors are generated and evaluated. Here, mutation, insertion and reversion are used with the specified probability based on the roulette wheel.

Step 6 Lists neighbors are descending ordered based on evaluation criteria being the normalized energy consumption and their best members will win and be added to the primary population.

Step 7 According to the SA comparison rule, each member of the primary population is compared with a member of the winning neighborhood population. (If it was better, it would be accepted; otherwise, it may be accepted with a probability[6]).

Step 8 The best response so far has been updated.

Step 9 If the termination conditions are not met, the temperature will be reduced and it will be started from step 4. In the simulated annealing method, the temperature parameter value remains constant in each Markov chain until equilibrium is obtained. When equilibrium is established, the Markov chain varies and thus the value of temperature parameter should be reduced using a proper function. There exist many reduction functions of the temperature parameter value that are used in the simulated annealing method. These functions can be classified into two general categories of static and dynamic.

Static function parameters are determined prior to the start of the calculations and keep constant during the calculations.

Dynamic or adaptive functions often contain parameters that will change by computing the development and obtaining new information.

Static functions are easy to use, but the problem is that they are slow to use. It is more difficult to use adaptive functions. However, several studies have shown that the convergence rate of adaptive functions is much more than static functions, and if the calculation speed is concerned, adaptive functions must be utilized.

[6] In our implementation, it is accepted in case it is better and it is accepted conditionally and with the probability of $p = \frac{-\triangle f}{T}$ in case it is worse. However, some other functions can be provided for p and it only needs some certain conditions.

The simplest and most widely used reduction function of the temperature parameter is a geometric function that can be expressed as Eq. 6:

$$T_{K+1} = \lambda \times T_K \tag{6}$$

where T_K and T_{K+1} are the values of temperature parameter in Kth and (K+1)th Markov chains, respectively. The parameter λ is a constant parameter the value of which is desirably chosen from a range of 0.8–0.99.

As mentioned before, constant λ causes the calculations to become slow. Therefore, in this section, a model is presented for making λ adaptive and increasing the speed of calculations.

The conducted studies show that when the value of temperature parameter is high, λ can be small so that computing can speed up, however, with advances in computing and reduction in the temperature parameter value, λ value should be large so that the convergence of the method can be guaranteed. Thus, we use the Eq. 7 for updating λ in each Markov chain:

$$\lambda_K = \lambda_1^{\frac{NT_1}{NT_K}} \tag{7}$$

where λ_K and λ_1 are the values of λ in the Kth Markov chain and the value of λ in the first Markov chain, respectively. Moreover, NT_K and NT_1 are the lengths of the kth and first Markov chains, respectively.

λ_1 and NT_1, although depend on the type of the target problem, must be selected in such a way that the quality of the final answer as well as the computation time is appropriate. The values of 0.94 and 100, respectively, are used for λ_1 and NT_1 [21,22].

Step 10 The end.

3.4 Selection of low-loaded machines

At this stage, a list of all machines that are not overloaded or turned off (for all active non-overloaded machines) is prepared in an ascending order based on the CPU usage. Here, the total load and number of CPU cores are considered.

At this stage, the machines available in the first quarter of the list will be selected and it is attempted to reduce the number of active machines and to take them to low power consumption mode to the extent possible.

4 Performance evaluation

It is extremely difficult to conduct repeatable large-scale experiments on a real infrastructure, which is required to evaluate and compare the proposed algorithms. To ensure the repeatability of the experiments, we choose simulations as a suitable way to evaluate the algorithms.

CloudSim has been chosen in our experiments, which has been developed by the Cloud Computing and Distributed Systems (CLOUDS) Laboratory, University of Melbourne. For experiments, the data provided as a part of the CoMon project, a monitoring infrastructure for PlanetLab [18], were used.

Table 1 The three VM types used in our experiments

VM types	Cores	Capacity (MIPs)	RAM (MB)	Storage (GB)	BW (Mbit/s)
Large	1	2500	1740	2.5	100
Medium	1	2000	870	2.5	100
Small	1	1000	613	2.5	100

4.1 Experiments setup

The simulated environment consisted of 800 heterogeneous servers, half of which were HP ProLiant ML110 G4, and the other half were HP ProLiant ML110 G5. Each server has a bandwidth of 1 Gbps, and half of the bandwidth has been considered in the simulations made for migration and the other half for VM communication.

The simulation period has been considered to be 86,400 s.

To guarantee better performance evaluation of VM consolidation, 600 servers are dual core and the rest are single core.

The CPU frequency of each core of G4 is 1,860 MIPS with a memory of 4096 MB, and the CPU frequency of each core of G5 is 2660 MIPS with a memory of 4096 MB.

As for the power model, we used real data on power consumption provided by the result of the SPECpower[7] benchmark in our work. In simulations, we used the three types of Virtual Machines as shown in Table 1.

4.2 Performance metrics

For performance evaluation of the algorithms, some metrics were used. In this section, we define them briefly.

- *OTF* The fraction of time during which active hosts have experienced the CPU utilization of 100 %.
- *PDM* The overall performance degradation by VMs due to migrations.
- *SLAV* Two metrics for measuring the level of SLA violations in an IaaS environment are OTF and PDM.

$$\text{OTF} = \frac{1}{N}\sum_{i=1}^{N}\frac{T_{si}}{T_{ai}} \tag{8}$$

$$\text{PDM} = \frac{1}{M}\sum_{j=1}^{M}\frac{C_{dj}}{C_{rj}} \tag{9}$$

In Eqs. 8 and 9, N is the number of servers; T_{si} is the total time during which the server i has experienced the utilization of 100 % leading to an SLA violation; T_{ai} is the total of the server i being in the active state (serving VMs); M is the number of VMs;

[7] http://www.spec.org.

Cdj is the estimate of the performance degradation of the VM_j caused by migrations; C_{rj} is the total CPU capacity requested by the VM_j during its lifetime.

In our work, C_{dj} is estimated to be 10 % of the CPU utilization in MIPS during all migrations of the VM_j. Both the OTF and PDM metrics independently characterize the level of SLA violations in the system; therefore, a combined metric that encompasses both performance degradation due to host overloading and VM migrations is proposed, denoted SLA Violation (SLAV). The metric is calculated as shown in Eq. 10.

$$\mathrm{SLAV} = \mathrm{OTF} \times \mathrm{PDM} \tag{10}$$

Energy The total energy consumption in cloud data center.
Migrations The total number migrations happened in the N servers in cloud data center.
ESV This metric is proposed to combine the two parameters: Energy and SLAV, which can be calculated using Eq. 11.

$$\mathrm{ESV} = \mathrm{Energy} \times \mathrm{SLAV} \tag{11}$$

4.3 Workload data

To make the results more convincing, the real workload provided by CloudSim is adopted in our experiments.

The only characteristic recorded in the workload is the CPU utilization of VMs. It is a part of CoMon project [18], which is collected from more than a thousand VMs from the servers located at more than 500 places around the world. It has 10-day records from March to April in 2011 and a time interval of 5 min between two consecutive records. These values are also available in the CloudSim simulator at the path examples/workload/planetlab, where we used item 20110303.

In addition, Table 2 of [17] gives a brief analysis of the workload.

4.4 Simulation results and analysis

Since the proposed comprehensive algorithm uses the LR algorithm for overloading assessment, the combination of LR-MC[8] and LR-MMT[9] has been compared with the proposed comprehensive algorithm [17].

The comparison has been performed in three modes of low load, medium load, and high load, and according to the metrics referred to in Sect. 4.2.

In this comparison, the proposed comprehensive algorithm, which is referred to as CA in the Figures, had a better performance compared to other algorithms. Simulation results are presented in Figs. 4, 5, 6, 7, 8, 9 for a better assessment.

According to Fig. 4, the proposed algorithm has had lower energy use than the other compared algorithms due to the proper solutions considered in it. Based on the results

[8] Local regression-maximum correlation.

[9] Local regression-minimum migration time.

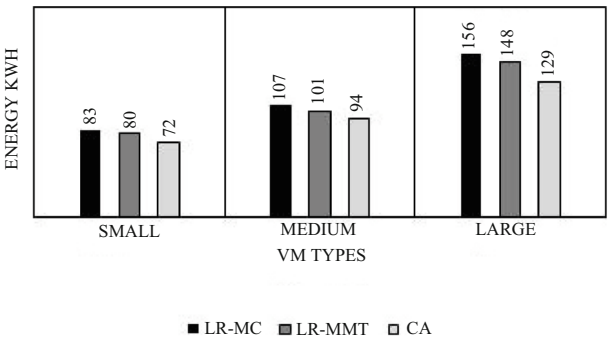

Fig. 4 The energy consumption of algorithms

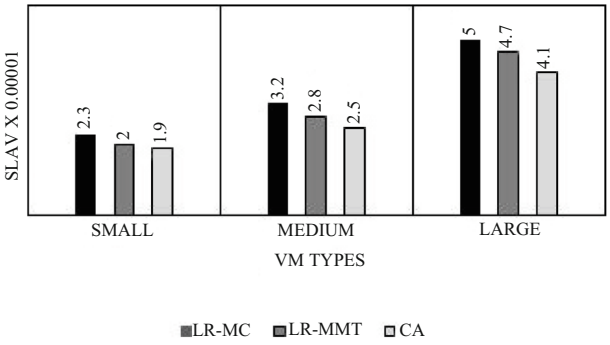

Fig. 5 The SLA violation evaluation of algorithms

of the experiments, energy use has decreased significantly in all the three scenarios. Our algorithm has performed more properly than the best algorithm, i.e., LR-MMT, by 10 % in low load, by 6.9 % in moderate load, and by 12.8 % in high load. As we know, to reduce energy use, we should power off as many servers as possible, and avoid unnecessary migrations as far as possible, since reduction of the number of migrations can play a major role in reducing energy use based on the advantages suggested in the beginning paragraph in Sect. 3.1. We have considered both of these in presenting the proposed solution. In fact, one of the important purposes of introducing the concept of critical states in the proposed solution has been to reduce unnecessary migrations. Furthermore, powering low-load servers off has been taken into account at the end of Sect. 3. It should be mentioned that based on the results obtained from our experiments in high load, the average number of times the servers have moved to the powered-off state has been 4085 times in the proposed algorithm and 3996 and 3958 times in the other two algorithms.

One of the positive points about our work is reduction of SLA violation. Many of the studies previously conducted in the area of energy use reduction in cloud data centers have paid little attention to SLA violation or have not regarded it as very important, in such a way that it would be regarded as acceptable even if SLA violation increased a bit in case of energy use reduction. However, the set of arrangements made in the

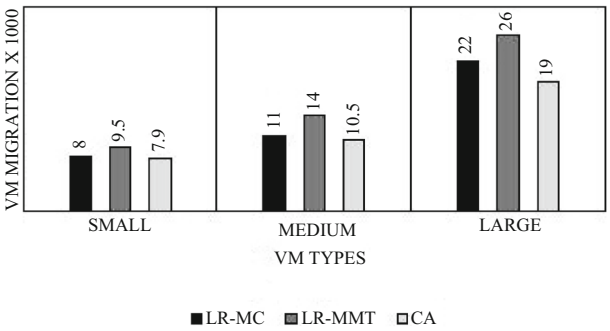

Fig. 6 The migration evaluation of algorithms

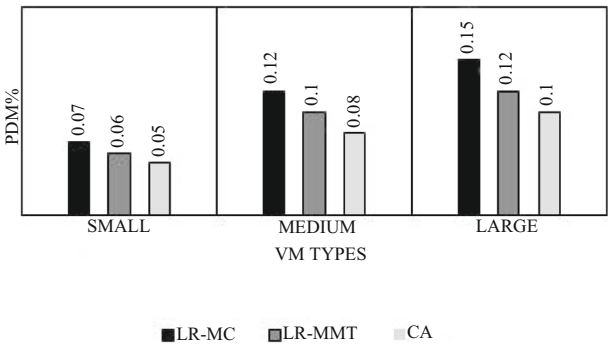

Fig. 7 The PDM evaluation of algorithms

proposed algorithm, avoidance of unnecessary migrations in particular, has resulted in energy use reduction along with reduction of SLA violation. As shown in Fig. 5, SLA violation has decreased as compared to the best algorithm, i.e., LR-MMT, by 5 % in low load, by 10 % in moderate load, and by 12.7 % in high load. Based on the simulation results in Fig. 6, the number of migrations has decreased as compared to the other algorithms; the decrease has been more remarkable, by about 13.6 %, in high load. This is due to making the right decisions and using proper algorithms in the proposed framework. In the other algorithms, migration of virtual machines is taken into account as soon as overload is detected, whereas in our algorithm, blind migration is not performed, and virtual machine migration is done in conditions where the server status is critically based on its processing power and memory value. Clearly, increase in the number of virtual machine migrations leads to a greater fall in performance based on Relation 2. Therefore, the decrease in the number of migrations in the proposed algorithm has resulted in better performance reduction caused by migration than in the other algorithms. Based on the results in Fig. 7, better performance by 17 % in average is observed also in this area. Based on Fig. 8, even though the percentage of occasions where the CPU is in overloaded state has become higher than the other two states, the results are close together. This results from the fact that the proposed algorithm does not consider overload as denoting criticality, and will not reduce load unless

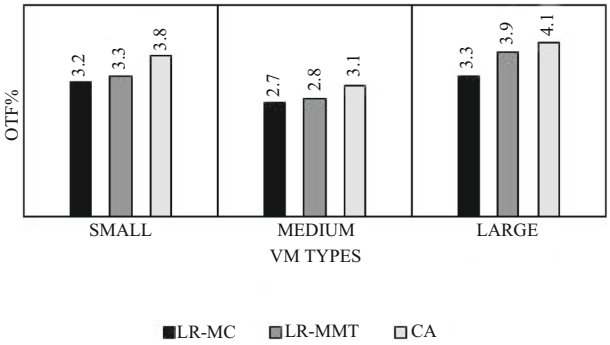

Fig. 8 The OTF evaluation of algorithms

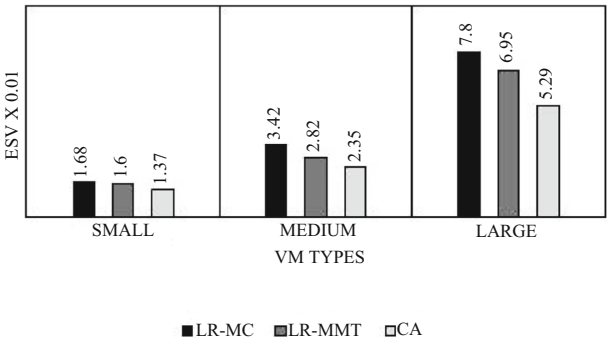

Fig. 9 The ESV evaluation of algorithms

the conditions are detected as critical. Thus, the percentage of occasions where the processor is in overloaded state will be higher. On the other hand, it is SLA violation reduction that matters, which is concerned with the product of parameters PDM and OTF based on Relation 10. The results displayed in Fig. 5, discussed previously, confirm this. Based on Eq. 7, parameter ESV is directly related to SLA violation and energy use values. By reducing SLA violation and saving more energy, the proposed algorithm has performed better than the other two algorithms also in this regard. As observed in Fig. 9, our algorithm has performed better than the best algorithm, i.e., LR-MMT, by 14.3 % in low load, by 16.7 % in moderate load, and by 23.8 % in high load.

5 Conclusions and future works

In this paper, we proposed an energy-aware framework for VM consolidation to make better energy-performance tradeoff.

First, we determine the hosts with critical state. To do this, we classify the host status overload into two types, i.e., with the possibility of SLA violation and without

it. Second, using the virtual machine selection algorithm, virtual machines are selected to migrate from critical hosts.

After forming a list of virtual machines selected for migration, using the host selection algorithm, new hosts are selected as the destination for the migrating VMs.

The population-based or parallel SA algorithm is used in the Markov chain model in this stage.

Finally, the low-load hosts are selected using the low-load host selection algorithm, and go to the off or sleep state by migration of all VMs stationed in them. Turning off the largest number of active machines along with reducing the number of unnecessary migrations are two key objectives in our work.

We have evaluated our comprehensive algorithm and some other popular algorithms, and the simulation results show that our work gets a better energy-performance trade-off (the ESV metric). More work is still underway for the proposed work.

It will be evaluated in real environment. We also want to improve proposed algorithm by applying some of these issues as future works: deeper analysis of operation of the proposed algorithms in different simulated scenarios, consideration of data centers with geographical expansion and consideration of the energy consumed in switching equipment.

Acknowledgements This work is sponsored by Islamic Azad University Science and Research Branch. We thank them for their support.

References

1. Buyya R, Yeo CS, Venugopal S, Broberg J, Brandic I (2009) Cloud computing and emerging IT platforms: vision, hype, and reality for delivering computing as the 5th utility. Future Gener Comput Syst 6:599616
2. Monil M, Rahman RM (2016) VM consolidation approach based on heuristics, fuzzy logic, and migration control. J Cloud Comp 5:8–25
3. Lee YC, Zomaya AY (2012) Energy efficient utilization of resources in cloud computing systems. J Supercomput 60(2):268280
4. Heddeghem WV, Lambert S, Lannoo B, Colle D, Pickavet M (2014) Trends in worldwide ICT electricity consumption from 2007 to 2012. Comput Commun 50:64–76
5. Khosravi A, Kumar SG, Buyya R (2013) Energy and carbon-efficient placement of virtual machines in distributed cloud data centers. In: Euro-Par'13 Proceedings of the 19th International Conference on Parallel Processing. Springer, Berlin, Germany, pp 317–328
6. Asyabi E, Sharifi M (2015) A new approach for dynamic virtual machine consolidation in cloud data centers. Int J Mod Educ Comput Sci 4:61–66
7. Chen L, Zhang J, Cai L, Li R, He T, Meng T (2015) MTAD: a multitarget heuristic algorithm for virtual machine placement. Int J Distrib Sens Netw 11(10):679170. doi:10.1155/2015/679170
8. Salimian L, Esfahani FS, Shahraki MN (2016) An adaptive fuzzy threshold-based approach for energy and performance efficient consolidation of virtual machines. Computing 98(6):641660
9. Barroso LA, Clidaras J, HolzleThe U (2013) Datacenter as a computer: an introduction to the design of warehouse-scale machines, 2nd edn. Morgan & Claypool Publishers, USA
10. Beloglazov A, Buyya R (2013) Managing overloaded PMs for dynamic consolidation of virtual machines in cloud data centers under quality of service constraints. IEEE Trans Parallel Distrib Syst 24:1366–1379
11. Yang J, Liu C, Shang Y (2014) A cost-aware auto-scaling approach using the workload prediction in service clouds. Inf Syst Front 16:7–18
12. Xiao Zh, Song W, Chen Q (2013) Dynamic resource allocation using virtual machines for cloud computing environment. IEEE Trans Parallel Distrib Syst 24:1107–1117

13. ASHRAE (2011) Thermal guidelines for data processing environments. American Society of Heating and Refrigerating and Air-Conditioning Engineers, USA, Tech, Rep
14. Wolke A, Ayush BT, Pfeiffer C, Bichler M (2015) More than bin packing. Inf Syst 52(C):83–95
15. Dhingra A, Paul S (2014) Green cloud: heuristic based BFO technique to optimize resource allocation. Indian J Sci Technol 7(5):685691
16. Tang M, Pan S (2014) A hybrid genetic algorithm for the energy-efficient virtual machine placement problem in data centers. Neural Process Lett 41:211–221
17. Beloglazov A, Buyya R (2012) Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in cloud data centers. Concurr Comput Pract Exp 13:1397–1420 Wiley Press
18. Park KS, Pai SV (2006) CoMon: a mostly-scalable monitoring system for planet-lab. ACM SIGOPS Oper Syst Rev 40:6574
19. Junior HA, Ingber L, Petraglia A, Petraglia MR, Machado MA (2012) Stochastic global optimization and its applications with fuzzy adaptive simulated annealing. Intell Syst Ref Libr 35:33–62
20. Lee DY, Wexler AS (2011) Simulated annealing implementation with shorter Markov chain length to reduce computational burden and its application to the analysis of pulmonary airway architecture. Comput Biol Med 41:707715
21. Scott LR, Harmonosky CM (2005) An improved simulated annealing simulation optimization method for discrete parameter stochastic systems Elsevier. Comput Oper Res 32:343358
22. Vasan A, Raju KS (2012) Comparative analysis of simulated annealing, simulated quenching and genetic, algorithms for optimal reservoir operation. Appl Soft Comput 9:274281