J. Cent. South Univ. (2017) 24: 2331−2341
DOI: https://doi.org/10.1007/s11771-017-3645-z

Springer

Energy-efficient virtual machine consolidation algorithm in cloud data centers

ZHOU Zhou(周舟)[1], HU Zhi-gang(胡志刚)[1], YU Jun-yang(于俊洋)[1], Jemal Abawajy[2], Morshed Chowdhury[2]

1. School of Software, Central South University, Changsha 410083, China;
2. School of Information Technology, Deakin University, Melbourne, Australia

Abstract: Cloud data centers consume a multitude of power leading to the problem of high energy consumption. In order to solve this problem, an energy-efficient virtual machine (VM) consolidation algorithm named PVDE (prediction-based VM deployment algorithm for energy efficiency) is presented. The proposed algorithm uses linear weighted method to predict the load of a host and classifies the hosts in the data center, based on the predicted host load, into four classes for the purpose of VMs migration. We also propose four types of VM selection algorithms for the purpose of determining potential VMs to be migrated. We performed extensive performance analysis of the proposed algorithms. Experimental results show that, in contrast to other energy-saving algorithms, the algorithm proposed in this work significantly reduces the energy consumption and maintains low service level agreement (SLA) violations.

Key words: cloud computing; energy consumption; linear weighted method; virtual machine consolidation; virtual machine selection algorithm

1 Introduction

With the fast growth of the cloud computing technology [1, 2] and the construction of a large number of data centers, the problem of high energy consumption is becoming more and more serious. It was reported [3] that data centers consume huge energy equal to the electrical production of 26 nuclear power plants. Research shows that most of the hosts in data centers operate at lower than 50% CPU utilization [4, 5]. This suggests that both high performance and energy efficiency of the data centers are important problems. Energy efficiency includes both energy consumption and quality of service (QoS). Service level agreement (SLA) is commonly used to formalize QoS. To address the data center energy consumption problem a wide variety of organizations such as Green Grid have recently been instituted with the aim to decrease the energy consumption and maintain low SLA violation.

In data centers, virtual machine (VM) migration [6] and host consolidation technology [7] offer a new approach for managing the energy consumption. With the aid of VM migration technology, the load scattered on different hosts can be consolidated on specific hosts. In order to reduce the energy consumption, the idle hosts are shut down or switched to low-power state such as sleep mode. However, during the migration and consolidation, some problems should be considered: 1) when a host is considered as being lightly-loaded or lightly-loaded? when a host is supposed to be optimally-loaded or heavily-loaded? 2) which VMs should be migrated? 3) where should VMs be placed? In order to deal with the above problems, this work proposes a new algorithm based on the combination of linear weighted method and VMs selection algorithm to reduce the energy consumption. The main contributions of this work are summarized as follows.

1) Using linear weighted method to predict the load of host.

2) Proposing a new VM placement algorithm named PVDE to reduce energy consumption.

3) Putting forward four kinds of VMs selection policies and making comparison.

4) Evaluating the proposed algorithm by extensive simulation utilizing the CloudSim toolkit.

2 Related work

Improving the energy efficiency of the data centers

Foundation item: Projects(61572525,61272148) supported by the National Natural Science Foundation of China; Project(20120162110061) supported by the PhD Programs Foundation of Ministry of Education of China; Project(CX2014B066) supported by the Hunan Provincial Innovation Foundation for Postgraduate, China; Project(2014zzts044) supported by the Fundamental Research Funds for the Central Universities, China
Received date: 2016−03−23; **Accepted date:** 2016−07−01
Corresponding author: ZHOU Zhou, PhD candidate; Tel: +86−15274975378; E-mail: zhouzhou03201@126.com

is extremely important due to both economic and environmental factors. At present, scholars are doing a large number of researches including power measurement, power modelling and optimization algorithms. In order to decrease the energy consumption and SLA violation, BELOGLAZOV and BUYYA [8] put forward an adaptive threshold method for consolidating VM. Although the approach is promising, they did not consider the load fluctuations leading to massive VM migration.

HANSON et al [9] proposed a DVFS (dynamic voltage and frequency scaling) policy. This policy can save energy by dynamically adjusting the host's CPU voltage frequency. For example, when a small number of tasks need to be addressed, DVFS policy decreases CPU voltage frequency of the hosts for the purpose of saving power; when a great deal of task should be processed, DVFS policy raises the CPU voltage frequency of the host with the intention of meeting the QoS requirements. However, the drawback of the approach is that the energy-saving needs to be further improved.

To solve the high energy consumption problem in data center, BELOGLAZOA and BUYYA [10] came up with a single threshold (ST) algorithm. ST algorithm sets a unified value for all hosts' CPU utilization to make sure all hosts below this value. Experimental results from the CloudSim simulator [11] illustrate that ST algorithm owns better performance than DVFS policy. However, ST algorithm causes too much SLA violations during VM consolidation.

Minimization of migrations algorithm (MM) has been proposed by BELOGLAZOA et al [12], which can improve the energy efficiency as well as minimize operational costs. MM sets two thresholds to keep CPU utilization for all hosts between the two thresholds. In addition, a couple of challenges concerning about high energy consumption problem are discussed at the end of the paper. Although the approach is promising, the energy saving effect of the approach needs to be further improved.

VAN et al [13] came up with a utility-based VM placement manager. They regarded both power consumption and SLA violation as constraint satisfaction problems. Although the method is promising, they did not present total power consumption of the test bed. As an effective scheduling algorithm, it is vital in precisely estimating a task's execution time. In return, overestimated or underestimated of the task execution time could lead to high energy consumption or massive SLA violations. KANG and RANKA [14] came up with an algorithm to reallocate the slack to future tasks, for the purpose of decreasing the energy consumption or meeting the QoS. However, it is difficult for this algorithm to precisely estimate a task's execution time.

BUYYA et al [15] explored the VM placement problem, and put forward some challenges and opportunities for energy management. In our previous study [16], we put forward a three-threshold energy-aware algorithm named MIMT to deal with the energy consumption and SLA violation. However, the three thresholds for controlling host's CPU utilization are fixed. Therefore, MIMT is not suitable for varying workload.

3 Power model, cost of VM migration, and SLA violations metric

In this section, we would present the background information with emphases on the power model, cost of VM migration and SLA violations metrics.

3.1 Power model

KUSIC et al [17] investigated the relationship between the host energy consumption and the CPU resource utilization, and found that the energy consumption of a host ($P(u)$) can be described in Eq. (1):

$$P(u)=k\cdot P_{\max}+(1-k)\cdot P_{\max}\cdot u \tag{1}$$

where k means the fraction of power consumption by an idle server; $P_{\max}$ corresponds to a usual value that could be gained by statistic methods; u corresponds to the CPU utilization of the host.

3.2 Cost of VM migration

Proper VM migration can improve the energy efficiency of data centers. However, excessive VM migration can bring negative impact such as performance degradation. VOORSLUYS et al [18] investigated the problem of performance degradation, as indicated in Eqs. (2) and (3).

$$U_{d_j}=c\cdot\int_{t_0}^{t_0+T_{m_j}}u_j(t)\mathrm{d}t \tag{2}$$

$$T_{m_j}=\frac{M_j}{B_j} \tag{3}$$

where U_{d_j} corresponds to the total performance degradation caused by VM j; c is the coefficient of the average performance degradation caused by VMs (in terms of web-applications, the value of c could be considered 10% of the CPU utilization [18]); t_0 is the start time of migration; T_{m_j} corresponds to the time of migration completed; $u_j(t)$ represents the CPU utilization of VM j; M_j corresponds to the total memory used by VM j; B_j represents the available bandwidth.

3.3 SLA violations metric

The value of SLA violations is vital for the VM placement algorithm, and SLA violations metric [8] could be defined as

$$SLA = \frac{U_{\text{req}} - U_{\text{allo}}}{U_{\text{req}}} \tag{4}$$

where U_{req} corresponds to the requested MIPS by all VMs, and U_{allo} represents the actually allocated MIPS.

4 Prediction-based VM deployment algorithm for energy efficiency

In this section, we present the proposed PVDE algorithm. The prediction function of the host's CPU utilization, and four kinds of VM selection algorithms will be presented. In terms of any VM placement algorithm, some problems must be considered: 1) when a host is considered as being little-loaded or lightly-loaded? when a host is supposed to be optimally-loaded or heavily-loaded? 2) which VMs should be migrated? 3) where should VMs be placed? In order to handle these problems, we firstly utilize the linear weighted method to predict host's CPU utilization, and then, PVDE is proposed.

4.1 Prediction function of host's CPU utilization

It is extremely necessary to attain host's CPU utilization at a certain time. To deal with this problem, a linear weighted method to predict hosts' CPU utilization is proposed. The main idea is as follows: In order to obtain hosts' CPU utilization at t_{i+1} time, several previous hosts' CPU utilization at t_1, t_2, t_3, ⋯, t_{i-1}, t_i time (t_1, t_2, t_3, ⋯, t_{i-1} and t_i represent a time series) are used to make predictions, as shown in Eq. (5):

$$U_{t_{i+1}} = \sum_{i=1}^{t} \partial_i U_{t_i} = \partial_1 U_{t_1} + \partial_2 U_{t_2} + \cdots + \partial_i U_{t_i} \tag{5}$$

where ∂_i represents weight coefficient of host at t_i time; U_{t_i} means the actual values of hosts' CPU utilization at t_i time; $U_{t_{i+1}}$ represents the predictive value of hosts' CPU utilization. If the time interval between the actual value and predicted value is shorter, then the value of correspond weight coefficient ∂_i is greater. How is the prediction accuracy of Eq. (5)? This problem will be discussed in Section 5.2.

4.2 PVDE

The main idea of PVDE can be concluded as follows: PVDE firstly sets three thresholds, namely T_a, T_b and T_c ($0 \le T_a \le T_b \le T_c \le 1$), resulting in the data center hosts separated into four classes, hosts with little load, hosts with light load, hosts with optimal load, and hosts with heavy load. When predictive value of a host's CPU utilization is less than or equal to T_a, the host is often deemed to be with little load. With the intention to save energy consumption, all VMs on little-loaded host (predictive value of CPU utilization is less than or equal to T_a) are migrated to lightly-loaded hosts (CPU utilization at T_a–T_b interval). When predictive value of a host's CPU utilization is between T_a and T_b, the host is often believed to be with light load. With the aim to avoid excessive VM migration leading to high SLA violations, all VMs on lightly-loaded host (predictive value of CPU utilization at T_a–T_b interval) are kept unchanged. When predictive value of a host's CPU utilization is between T_b and T_c, the host is often supposed to be with optimal load. For the purpose of avoiding excessive VM migration leading to high SLA violations, all VMs on optimally-loaded host (predictive value of CPU utilization at T_b–T_c interval) are kept unchanged. When predictive value of a host's CPU utilization is greater than T_c, the host is often deemed to be with heavy load. With the intention to decrease the SLA violations, some VMs on heavily-loaded host (predictive value of CPU utilization is greater than T_c) are migrated to lightly-loaded hosts (CPU utilization at T_a–T_b interval). Figure 1 depicts the flow chart of PVDE. Parameter n in Fig. 1 represents the number of hosts in data center.

In practical cloud-based system, as the workload is dynamic and unpredictable, it is difficult to determine the optimal values of T_a, T_b, and T_c considering energy efficiency. But it is possible to attain optimal interval between these values by using statistic method. What is the optimal interval between T_a and T_b? And what about T_b and T_c? These problems will be discussed in Section 5.3. As we discussed earlier, some VMs on heavily loaded host (predictive value of CPU utilization is greater than T_c) are migrated to another host with light load. However, which VM should be migrated?

4.3 VM selection

As we discussed earlier, VM migration is invoked from heavy loaded host. The problem, however, is which VM should be migrated from the heavily-loaded host? To address this problem, based on the PVDE, four different VM selection algorithms, namely, the minimization and maximization of migrations policy based on PVDE (MMMP), the minimization of migrations policy based on PVDE (MINP), the maximization of migrations policy based on PVDE (MAXP) and the random choice policy based on PVDE (RCP). In the following subsections, we will present each of MMMP, MINP, MAXP, and RCP in detail. For ease of presentation, some notations are presented in Table 1.

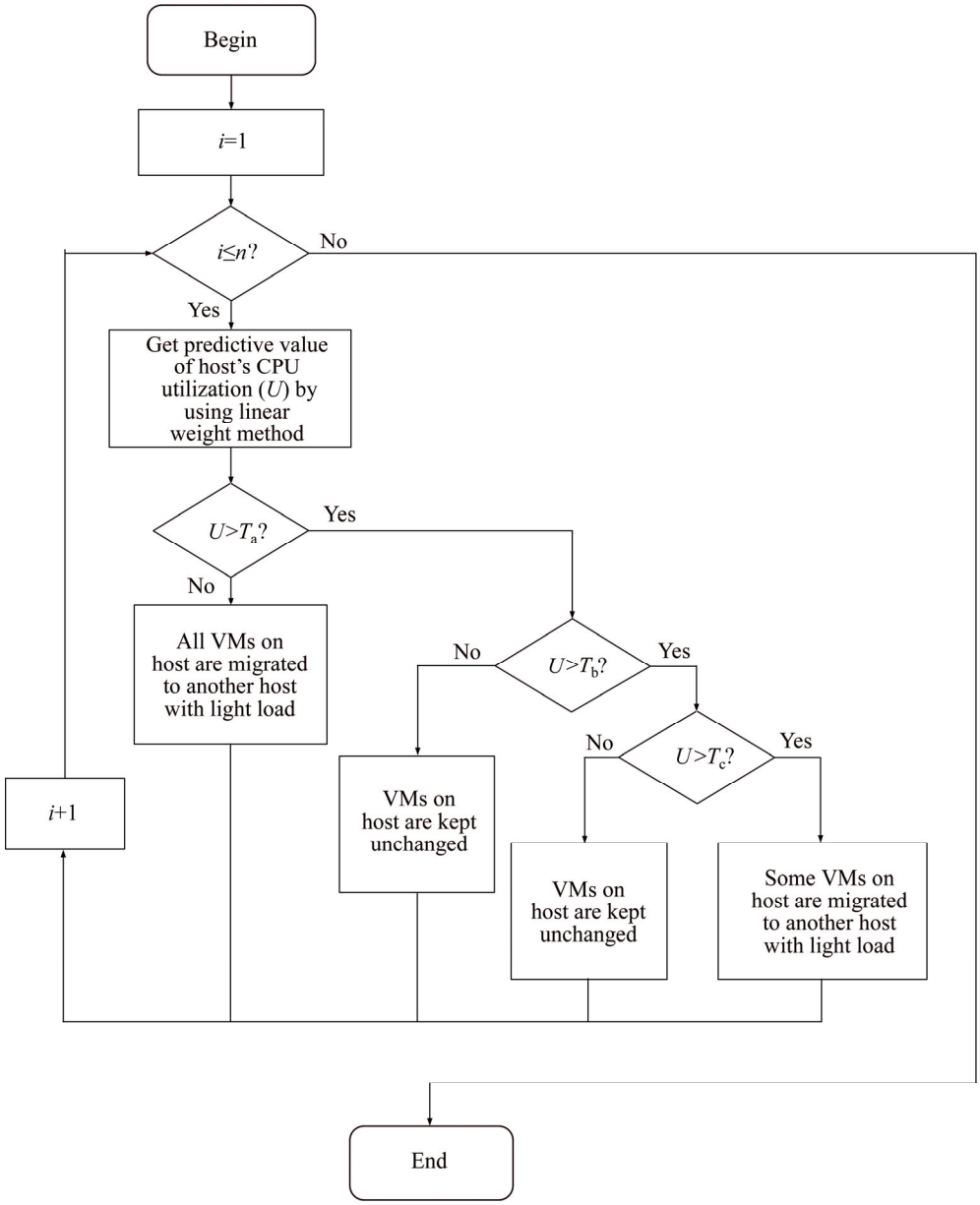

Fig. 1 Flow chart of PVDE

Table 1 Notations and their meanings

Notation	Meaning
VM^i	The set of VMs which allocates to the host i
$S(VM^i)$	Power set of VM^i
U_i	Predictive value of CPU utilization of host i
V_c^i	The fraction of the CPU utilization allocated to VM V
$A(P)$	The number of P
T_a, T_b, T_c	Three thresholds of PVDE (see Section 4.2)

4.3.1 Minimization and maximization of migrations policy based on PVDE (MMMP)

The aim of MMMP is to lower the CPU utilization below T_c threshold. For a host with heavy load (predictive value of CPU utilization is greater than T_c), the MMMP algorithm selects a VM with the highest CPU utilization. It then selects a VM with the lowest CPU utilization to migrate, with the aim to lower the CPU utilization below T_c threshold each time. The MMMP policy chooses a set $T \in VM^i$, formalized as follows.

$$T=\begin{cases}\{P \mid P \in S(VM^i), U_i - \sum_{V \in P} V_c^i < T_c, \\ \quad A(P) \to \min\}, \text{if } U_i > T_c \text{ and } j \bmod 2 \neq 0 \\ \{P \mid P \in S(VM^i), U_i - \sum_{V \in P} V_c^i < T_c, \\ \quad A(P) \to \max\}, \text{if } U_i > T_c \text{ and } j \bmod 2 = 0 \\ \phi, \text{ if } T_b < U_i \le T_c \\ \phi, \text{ if } T_a < U_i \le T_b \\ VM^i, \text{ if } U_i \le T_a \end{cases} \quad (6)$$

where "$A(P)\rightarrow$min" is to minimize the value of $A(P)$, while "$A(P)\rightarrow$max" is to maximize the value of $A(P)$. The meanings of other notations are shown in Table 1. The pseudo-code of MMMP is presented in Algorithm 1.

Algorithm 1: MMMP

Input: T_a, T_b, T_c, hostlist // hostlist means the set of host in data center
Output: migration VmList// "migrationVmList" means the set of VMs which should be migrated
1: for each host in hostlist do
2: vmlist←host. get VmList ( ); //get VMs on the host
3: vmlist.sortByCpuUtilization ( ); // sort by decreasing order
4: hUtil ← host.getPredictiveUtilization ( ); // get the predictive value of the host's CPU utilization
5: if (hUtil ≤ T_a) then // less than or equal to T_a, all VMs on the host are need to migrate
6: migrationVmList.add (host.getVmList( )); //add all VMs to migration list
7: vmlist.remove (host.getVmList( )); //delete all VMs on the host
8: else if (hUtil ≤ T_c) then // Both T_a–T_b and T_b–T_c interval, the VMs on the host remain unchanged
9: continue;
10: else if (hUtil>T_c) then // greater than T_c, some VMs should be chosen to migrate
11: for (int j=0; j<(vmlist.size ( )+1)/2; j++) do
12: hUtil← hUtil – vmlist.get (j).getUtil( ); // chooses a VM with the highest CPU utilization
13: migrationVmList.add (vmlist.get (j));
14: vmlist.remove (vmlist.get (j));
15: if (j=vmlist.size( )–j–1) then // this is the last VM
16: break;
17: else // chooses a VM with the lowest CPU utilization from the host to migrate
18: hUtil ← hUtil – vmlist.get (vmlist.size( ) – j – 1).getUtil( );
19: migrationVmList.add (vmlist.get (vmlist.size( ) – j – 1));
20: vmlist.remove (vmlist.get (vmlist.size() – j – 1));
21: end if
22: if (hUtil < T_c) then // meet the requirement
23: break;
24: end if
25: end for
26: end if
27: end for
28: return migrationVmList;

where Line 10–Line 14 means that MMMP chooses a VM with the highest CPU utilization from the host, Line 17–Line 21 represents that MMMP chooses a VM with the lowest CPU utilization from the host.

4.3.2 Minimization of migrations policy based on PVDE (MINP)

For a host with heavy load, MINP aims to lowering the CPU utilization below T_c threshold. MINP selects the minimum number of VMs to migrate from the host in order to lower the CPU utilization below T_c threshold. This is achieved by choosing a VM with the highest CPU utilization to migrate each time. The MINP policy chooses a set $T \in VM^i$, which is formalized as follows:

$$T=\begin{cases}\{P \mid P \in S(VM^i),\ U_i-\sum_{V\in P} V_c^i < T_c,\\ \quad A(P)\rightarrow \min\},\ \text{if } U_i > T_c\\ \phi,\ \text{if } T_b < U_i \le T_c\\ \phi,\ \text{if } T_a < U_i \le T_b\\ VM^i,\ \text{if } U_i \le T_a\end{cases} \tag{7}$$

where "$A(P)\rightarrow$min" is to minimize the value of $A(P)$. The meanings of other notations are shown in Table 1.

4.3.3 Maximization of migrations policy based on PVDE (MAXP)

For a host with heavy load, MAXP selects the maximum number of VMs to migrate from the host, with the intention to lower the CPU utilization below T_c threshold. In other words, MAXP chooses a VM with the lowest CPU utilization to migrate each time. The MAXP policy chooses a set $T \in VM^i$, which is formalized as follows:

$$T=\begin{cases}\{P \mid P \in S(VM^i),\ U_i-\sum_{V\in P} V_c^i < T_c,\\ \quad A(P)\rightarrow \max\},\ \text{if } U_i > T_c\\ \phi,\ \text{if } T_b < U_i \le T_c\\ \phi,\ \text{if } T_a < U_i \le T_b\\ VM^i,\ \text{if } U_i \le T_a\end{cases} \tag{8}$$

where "$A(P)\rightarrow$max" is to maximize the value of $A(P)$. The meanings of other notations are shown in Table 1.

4.3.4 Random choice policy based on PVDE (RCP)

When a host with heavy load, RCP random selection of a VM from the VMs list, with the intention to reduce the host's CPU utilization, the RCP policy chooses a set $T \in VM^i$, formalized as follows:

$$T=\begin{cases}\{P \mid P \in S(VM^i),\ U_i-\sum_{V\in P}V_c^i<T_c,\\ \quad Y\to M(0,A(P)-1)\},\ \text{if } U_i>T_c\\ \phi,\ \text{if } T_b<U_i\le T_c\\ \phi,\ \text{if } T_a<U_i\le T_b\\ VM^i,\ \text{if } U_i\le T_a\end{cases} \tag{9}$$

where "$M(0, A(P)–1)$" is a uniformly distributed between 0 and $(A(P)–1)$; Y corresponds to a uniformly distributed discrete random variable which is used to select a subset of VM^i.

Performance evaluation for the four kinds of VM selection algorithms (that is MMMP, MINP, MAXP, and RCP) will be presented in Section 5.4.

4.4 VM placement

In general, VM placement could be modelled as a bin packing problem with variable bin sizes and prices. In order to deal with the VM placement problem, a modification of the best fit decreasing algorithm (denoted by MBFDA) could be used to solve it. The pseudo-code of MBFDA is presented in Algorithm 2.

Algorithm 2 MBFDA

Input: T_a, T_b, T_c, vmlist, hostlist // vmlist corresponds to the set of VM
Output: allocation of VMs
1: vmlist. sortByCpuUtilization ( ); // sort by decreasing order by their CPU utilization
2: for each VM in vmlist do
3: minimumPower ← MaxValue; // minimumPower is assigned a max value
4: allocatedHost ← NullValue; // allocatedHost is assigned a null value
5: for each host in hostlist do
6: if (host is fit for Vm (vm)) then //host's CPU capacity, memory, and bandwidth are suitable for VM
7: uA ← getCpuUtilizationAfter-Allocation(host, vm); //get CPU utilization after allocating a VM
8: if (uA < T_a || uA > T_b) then //make sure host with light load (CPU utilization at T_a–T_b interval)
9: continue;
10: end if
11: Power ← getPowerIncreaseAfter-Allocation (host, vm); //get power increase after allocating a VM
12: if (Power < minimumPower) then // line 12-16 is to look for a host which owns the least increasing of power caused by VM allocation
13: minimumPower ← Power;
14: allocatedHost ← host;
15: end if
16: end if
17: end for
18: end for
19: return allocation;

The complexity of the MBFDA is $O(h{\cdot}v)$; h corresponds to the number of hosts; v represents the number of VMs which should be allocated.

5 Experiments and performance evaluation

In this section, we present the performance analysis of the proposed algorithm and compare it with the baseline algorithm.

5.1 Experimental environment

We used CloudSim toolkit [11] to implement the proposed algorithm and investigate its performance. We choose CloudSim as it is commonly used in the same research area for its capability of supporting the modeling of data centers and resource provisioning policies. Table 2 shows the parameter name and its value.

Table 2 Parameter name and its value

Parameter name	Related value
Number of host	300
Hosts' CPU capacity (MIPS)	{3000, 2000, 1000}
Hosts' memory size (MB)	1000
Hosts' bandwidth	100000
Number of VM	900
VMs' CPU capacity (MIPS)	{1000, 750, 500, 250}
VMs' memory size (MB)	128
VMs' bandwidth	2500
Number of task	900

We have simulated a data center which has 300 heterogeneous hosts. Each host possesses the performance equal to 3000, 2000 or 1000 MIPS, 1000 MB of RAM. Each VM possesses the performance equal to 1000, 750, 500, 250 MIPS, 128 MB of RAM. Users submit tasks to 900 heterogeneous VMs, and each VM can run a web-application or any kind of application with variable workload, and the application runs for 150000 MIPS.

5.2 Evaluating prediction accuracy

As for Eq. (5) (discussed in Section 4.1), it is crucial for determining the value of ∂_i (i=1, 2, 3, ⋯, n). In this

work, we choose ∂_i (i=1, 2, 3, 4, 5) to do related testing, and the value of ∂_i (i=1, 2, 3, 4, 5) is acquired by empirical method. By using empirical method, ∂_1=0.7, ∂_2=0.15, ∂_3=0.08, ∂_4=0.05 and ∂_5=0.02. Therefore, Eq. (5) can be modified into Eq. (10).

$$U_{t_{i+1}}=\sum_{i=1}^{5}\partial_i U_{t_i}=0.7\cdot U_{t_1}+0.15\cdot U_{t_2}+0.08\cdot U_{t_3}+0.05\cdot U_{t_4}+0.02\cdot U_{t_5} \quad (10)$$

How is the prediction accuracy of Eq. (10)? In running CloudSim program [11], we record experimental results every 300 s. As for there are 300 hosts (see Table 2) in the data center, the predicted CPU utilization of each host is compared with its actual value, and the correlation coefficient of each host is also calculated. The calculated results show that the minimum value of correlation coefficient is 0.91 and the maximum value of correlation coefficient is 0.99. In general, if the value of correlation coefficient is closer to 1, the prediction accuracy is better. In addition, the average of the 300 hosts' CPU utilization is also used to make comparison with the average of the predictive value of the 300 hosts' CPU utilization, as shown in Fig. 2.

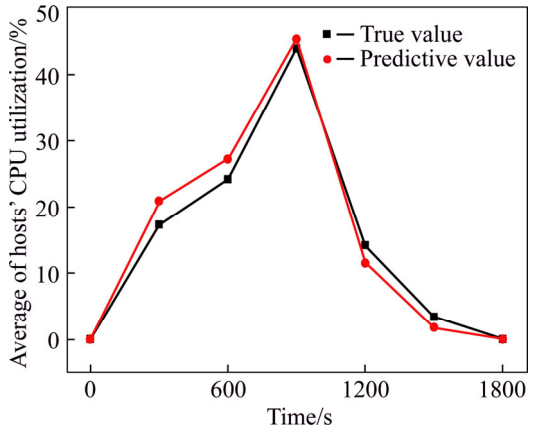

Fig. 2 Comparison between true value and predictive value

In Fig. 2, the black line represents the average of the 300 hosts' CPU utilization in data center, while the red line corresponds to the average of the predictive value of the 300 hosts' CPU utilization in data center. Similarly, we calculate the correlation coefficient P, and P is 0.98. Therefore, Eq. (10) is an effective method for predicting future CPU utilization of a host.

5.3 Optimal interval among three thresholds

When considering the energy efficiency, it is crucial to determine the optimal interval among the three thresholds of T_a, T_b and T_c. Taking MMMP as an example, what is the optimal interval between T_a and T_b? And what about T_b and T_c? To solve these questions, a method of exhaustion is used to determine the optimal interval among the three thresholds. Supposing the difference between T_a and T_b (T_b–T_a) is integer multiple of 0.1 (10%), and the difference between T_b and T_c (T_c–T_b) is also integer multiple of 0.1 (10%). As $T_a\in\{0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0\}$, and $0\le T_a<T_b<T_c\le 1$, the extreme points T_a=0.9 and T_a=1.0 should be removed. The threshold for T_b belongs to {0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9} and $T_c\in\{0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0\}$. Therefore, by using the exhaustion method, the combination of the three thresholds (T_a, T_b and T_c) is 45. For example, when T_a=0 (0%), the combinations of T_a, T_b and T_c can be (T_a, T_b, T_c)={(0, 0.1, 0.2), (0, 0.2, 0.3), (0, 0.3, 0.4), (0, 0.4, 0.5), (0, 0.5, 0.6), (0, 0.6, 0.7), (0, 0.7, 0.8), (0, 0.8, 0.9), (0, 0.9, 1.0)}. Figure 3 indicates the power under different intervals. Furthermore, Fig. 3 indicates the 40% (T_c–T_b=0.4) interval leading to the least power in most cases. As the SLA violation is crucial for VM placement algorithm, the SLA violation should also be considered. When T_a=0 (0%), Fig. 4 shows the SLA violations under different intervals, we can induce that 90% (T_c=1.0, T_b=0.1) interval leading to the least SLA violations. Considering both energy consumption and SLA violations, when T_a=0 (0%), the optimal interval for between T_b and T_c (T_c–T_b=0.4) is 40%.

Taken the same method, when T_a=0.1 (10%), T_a=0.2 (20%), T_a=0.3 (30%), T_a=0.4 (40%), T_a=0.5 (50%),

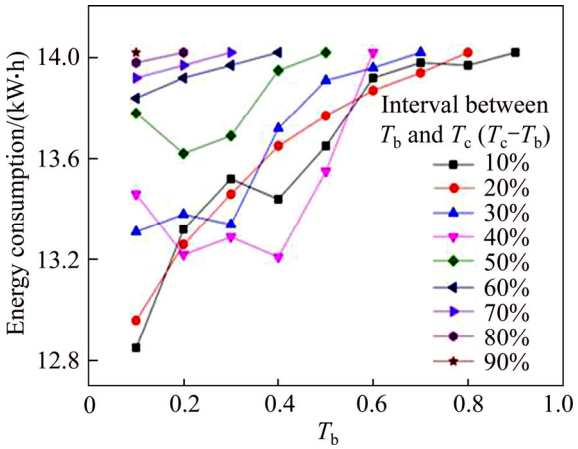

Fig. 3 Energy consumption by MMMP policy when T_a=0 (0%)

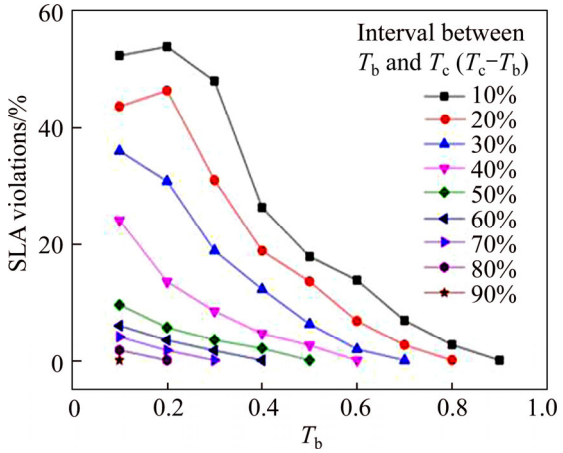

Fig. 4 SLA violations by the MMMP policy when T_a=0 (0%)

T_a=0.6 (60%), T_a=0.7 (70%), T_a=0.8 (80%), the related experiments were done and found that, considering the energy consumption and SLA violations, the optimal interval between T_a and T_b is 40% (T_b–T_a=0.4), and T_b to T_c is also 40% (T_c–T_b=0.4).

By the same method, we have chosen the other three algorithms (MINP, MAXP and RCP) for experiments. The same conclusion can be drawn that optimal interval between T_a and T_b is 40% (T_b–T_a=0.4), and T_b to T_c is also 40% (T_c–T_b=0.4).

5.4 Comparison among four VM selection policies

As discussed earlier, considering energy consumption and SLA violations, the optimal interval between T_a and T_b is 40% (T_b–T_a=0.4), and T_b to T_c is also 40% (T_c–T_b=0.4). In this part, this conclusion is used to make comparison among the four kinds of VM selection algorithms (MMMP, MINP, MAXP, and RCP). Experimental results for both energy consumption and SLA violations are shown in Fig. 5 and Table 3, respectively.

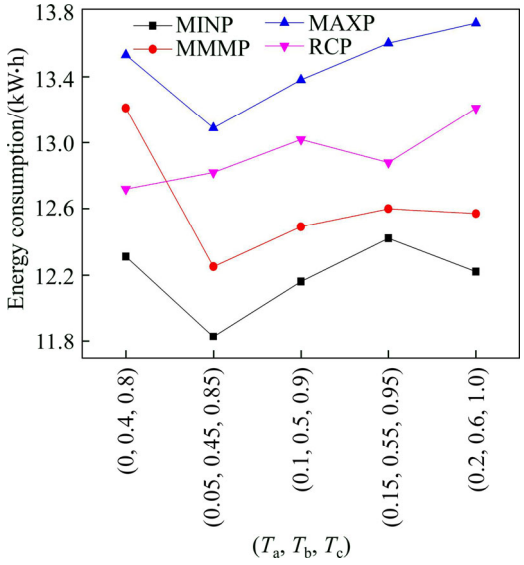

Fig. 5 Energy consumption

Table 3 SLA violations

(T_a, T_b, T_c)	MINP/%	MMMP/%	MAXP/%	RCP/%
(0, 0.4, 0.8)	8.86	4.60	3.76	6.52
(0.05, 0.45, 0.85)	27.80	25.47	24.48	25.91
(0.1, 0.5, 0.9)	39.72	38.20	36.13	37.10
(0.15, 0.55, 0.95)	50.46	48.73	47.36	49.02
(0.2, 0.6, 1.0)	63.27	61.70	59.12	60.42

Figure 5 illustrates the energy consumption for the four VM selection policies. Figure 5 indicates that MINP contributes to the least energy consumption, and MAXP leads to the most energy consumption, and MMMP results in the middle energy consumption. However, in terms of SLA violations from Table 3, the opposite is true. The reason is as follows: for a host with heavy load, MINP selects the minimum number of VMs to migrate, for the purpose of avoiding excessive VMs migration to bring more energy consumption. However, MINP causes many more SLA violations than MAXP and the reason is that MINP chooses a VM with the highest CPU utilization to migrate each time, while a VM with the highest CPU utilization is chosen to migrate leading to more SLA violations compared with other VMs. Contrary to MINP, for a host with heavy load, MAXP selects the maximum number of VMs to migrate. Therefore, MAXP leads to much more energy consumption and less SLA violations. MMMP leads to middle energy consumption and SLA violations compared with MINP and MAXP, the reason is that MMMP firstly chooses a VM with the highest CPU utilization, and then chooses a VM with the lowest CPU utilization to migrate each time. Different from MINP, MAXP, and MMMP, for a host with heavy load, RCP randomly selects VMs to migrate leading to average energy consumption and SLA violations.

From the view of energy efficiency (include energy consumption and SLA violations), in contrast to the other three algorithms (MINP, MAXP, and RCP), MMMP owns the better performance. Therefore, in the next Section (Section 5.5), the MMMP is selected to make comparison with other energy-aware algorithms.

5.5 Comparative analysis

We now compare the proposed algorithm with the baseline algorithm. In order to evaluate the performance of the proposed algorithm, the non power-aware (NPA), DVFS [9], ST [10], MM [12], and MIMA [16] algorithm are selected for making comparison. As for ST [10] algorithm, the main idea of ST algorithm is setting a threshold and all host's CPU utilization should be below this threshold. Figures 6 and 7 illustrate the energy consumption and SLA violations by ST, respectively.

Figure 6 induces that the energy consumption of ST

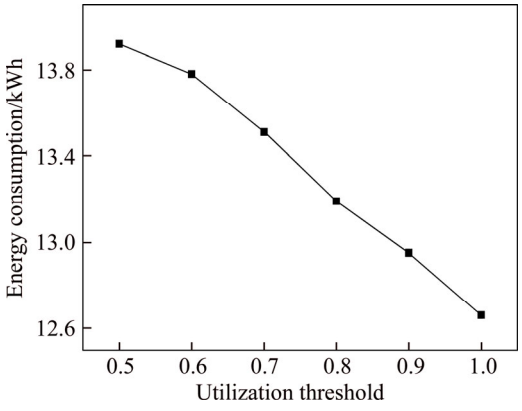

Fig. 6 Energy consumption by ST algorithm at different thresholds

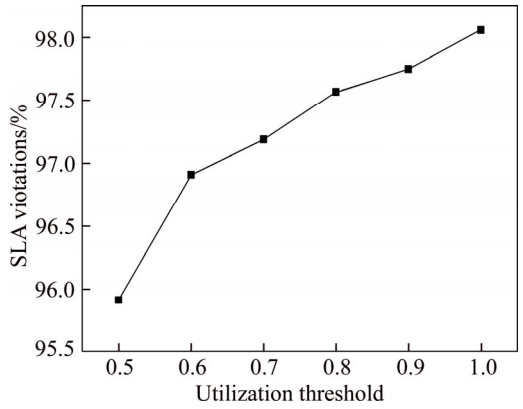

Fig. 7 SLA violations by ST algorithm at different thresholds

decreases with the increasing of utilization threshold from 0.5 to 1.0. Contrary to energy consumption, Fig. 7 displays that the SLA violations increase with the increasing of utilization threshold from 0.5 to 1.0. The reason could be explained by the fact that a higher threshold allows more VMs consolidation by the cost of raising SLA violations.

Different from ST algorithm, MM algorithm [12] sets two thresholds and all host's CPU utilization should be kept between the two thresholds. Considering energy efficiency, BELOGLAZOV et al [12] have proved thus the optimal interval is 40% for the two thresholds. Figures 8 and 9 illustrate the energy consumption and SLA violations by MM, respectively.

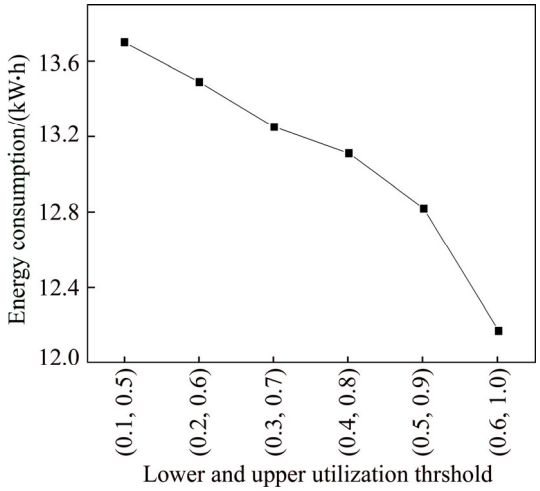

Fig. 8 Energy consumption of MM algorithm

Figure 8 induces that the energy consumption of MM decreases with the increasing of thresholds. Contrary to energy consumption, Fig. 9 displays that the SLA violations increases with the increasing of thresholds. The reason could be also explained by the fact that a higher threshold allows more VMs consolidation by the cost of raising SLA violations.

Table 4 displays the results of several energy-aware algorithm. NPA does not take any energy-aware

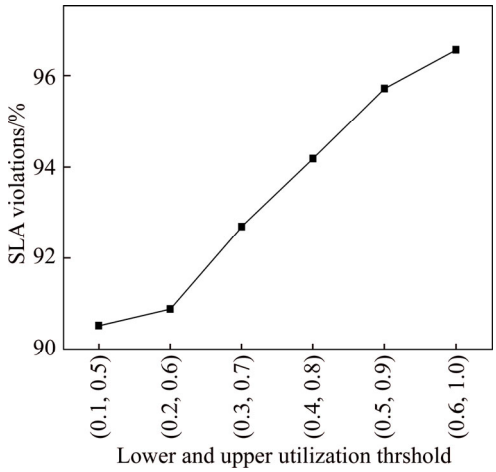

Fig. 9 SLA violations of MM algorithm

measures to 29.72 kW·h. DVFS [9] algorithm adopts approach to adjust the CPU voltage to 14.02 kW·h. Because NPA and DVFS does not consider the VM migration, the SLA violations, the number of VM migrations, and average SLA violations is used symbol "–" to label.

In Table 4, ST (70%) indicates that it sets a threshold 70% and all host's CPU utilization should be below this threshold. If a host's CPU utilization is greater than 0.7 (70%), a VM with the highest CPU utilization on the host must be migrated to another host. ST (70%) to save more energy consumption than both DVFS and NPA algorithm. The reason could be explained that ST (70%) takes good use of VM migration technology.

Different from ST (70%), MM (30%–70%) sets two thresholds: the one is 30% and the other is 70%. All host's CPU utilization should be kept between 0.3 (30%) and 0.7 (70%). Compared with ST (70%), MM (30%–70%) owns better performance due to the improvement of efficiency of VM migration.

Unlike MM (30%–70%), MIMT (5–45%–85%) sets three thresholds to improve the energy efficiency. As for MIMT (5%–45%–85%), the first threshold is 5%, the second one is 45%, and the last one is 85%. Table 4 illustrates that MIMT (5%–45%–85%) is more efficienct than MM (30%–70%). The reason is that MIMT (5%–45%–85%) reduces a lot of unnecessary VM migration, as excessive VM migration leads to high SLA violations.

Compared with MIMT (5%–45%–85%), under the same energy consumption, MMMP (5%–45%–85%) decreases by 8.3% SLA violation. The reason can be explained that MMMP (5%–45%–85%) takes good use of linear weighted method to improve the efficiency of VM migration. In addition, under the same energy consumption, MMMP (10%–50%–90%) decreases by 8.5% SLA violation compared with MIMT (10%–50%–90%).

Table 4 Compared with other energy-aware algorithms

Algorithm	Energy/(kW·h)	SLA violation/%	Number of VM migrations	Average SLA violation/%
NPA	29.72	—	—	—
DVFS	14.02	—	—	—
ST (50%)	13.92	95.91	1922	10.06
ST (60%)	13.78	96.92	1947	11.91
ST (70%)	13.51	97.19	1963	12.18
MM (30%–70%)	13.25	92.68	2315	10.12
MM (40%–80%)	13.11	94.18	2375	10.18
MM (50%–90%)	12.82	95.72	2396	10.24
MIMT(5%–45%–85%)	12.29	33.83	699	10.00
MIMT (10%–50%–90%)	12.42	46.7	979	10.00
MMMP (5%–45%–85%)	12.25	25.47	522	10.00
MMMP (10%–50%–90%)	12.49	38.20	832	10.00

6 Conclusions

A prediction-based VMs placement algorithm was proposed based on the combination of three-threshold algorithm, linear weighted method and VMs selection algorithm. Prediction-based VMs placement algorithm owns better energy efficiency than the traditional energy-saving algorithm. MMMP can be applied in cloud-based systems, for the purpose of improving the energy efficiency and saving the cost of energy consumption.

References

[1] PUTHAL D, SAHOO B P S, MISHRA S, SWAIN S. Cloud computing features, issues, and challenges: A big picture [C]// 2015 International Conference on Computational Intelligence and Networks (CINE). Bhubaneshwar: IEEE, 2015: 116–123.

[2] MAGALHÃES D, CALHEIROS R N, BUYYA R, DANIELO G G. Workload modeling for resource usage analysis and simulation in cloud computing [J]. Computers & Electrical Engineering, 2015, 47(1): 69–81.

[3] RICCIARDI S, CAREGLIO D, SANTOS-BOADA G, SOLÉ-PARETA J, FIORE U, PALMIERI F. Saving energy in data center infrastructures [C]// 2011 First International Conference on Data Compression, Communications and Processing (CCP). Palinuro: IEEE, 2011: 265–270.

[4] BARROSO L A, HOLZLE U. The case for energy-proportional computing [J]. Computer, 2007, 40(12): 33–37.

[5] BOHRER P, ELNOZAHY E N, KELLER T, KISTLER M, LEFURGY C, MCDOWELL C, RAJAMONY R. The case for power management in web servers [M]. Netherlands: Springer, 2002: 261–289.

[6] CLARK C, FRASER K, HAND S, HANSEN J G, JUL E, LIMPACH C, PRATT I, WARFIELD A. Live migration of virtual machines [C]// Proceedings of the 2nd Conference on Symposium on Networked Systems Design & Implementation. CA:USENIX Association, 2005: 273–286.

[7] HERMENIER F, LORCA X, MENAUD J M, MULLER G, LAWALL J. Entropy: A consolidation manager for clusters [C]// Proceedings of the 2009 ACM SIGPLAN/SIGOPS International Conference on Virtual Execution Environments. New York: ACM, 2009: 41–50.

[8] BELOGLAZOV A, BUYYA R. Adaptive threshold-based approach for energy-efficient consolidation of virtual machines in cloud data centers [C]// Proceedings of the 8th International Workshop on Middleware for Grids, Clouds and e-Science. Melbourne: ACM, 2010: 1–6.

[9] HANSON H, KECKLER S W, GHIASI S, RAJAMANI K, RAWSON F, RUBIO J. Thermal response to DVFS: Analysis with an Intel Pentium M [C]// Proceedings of the 2007 International Symposium on Low Power Electronics and Design. New York: ACM, 2007: 219–224.

[10] BELOGLAZOV A, BUYYA R. Energy efficient resource management in virtualized cloud data centers [C]// Proceedings of the 2010 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing. Washington: IEEE Computer Society, 2010: 826–831.

[11] CALHEIROS R N, RANJAN R, BELOGLAZOV A, DEROSE C A, BUYYA R. CloudSim: A toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms [J]. Software: Practice and Experience, 2011, 41(1): 23–50.

[12] BELOGLAZOV A, ABAWAJY J, BUYYA R. Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing [J]. Future Generation Computer Systems, 2012, 28(5): 755–768.

[13] VAN H N, TRAN F D, MENAUD J M. Performance and power management for cloud infrastructures [C]// 2010 IEEE 3rd International Conference on Cloud Computing. Miami: IEEE, 2010: 329–336.

[14] KANG J, RANKA S. Dynamic slack allocation algorithms for energy minimization on parallel machines [J]. Journal of Parallel and Distributed Computing, 2010, 70(5): 417–430.

[15] BUYYA R, RANJAN R, CALHEIROS R N. Modeling and

simulation of scalable cloud computing environments and the CloudSim toolkit: Challenges and opportunities [C]// International Conference on High Performance Computing & Simulation. Leipzig: IEEE, 2009: 1–11.

[16] ZHOU Z, HU Z, SONG T, YU J. A novel virtual machine deployment algorithm with energy efficiency in cloud computing [J]. Journal of Central South University, 2015, 22(5): 974–983.

[17] KUSIC D, KEPHART J O, HANSON J E, KANDASAMY N, JIANG G. Power and performance management of virtualized computing environments via look ahead control [J]. Cluster Computing, 2009, 12(1): 1–15.

[18] VOORSLUYS W, BROBERG J, VENUGOPAL S, BUYYA R. Cost of virtual machine live migration in clouds: A performance evaluation [M]. Berlin, Heidelberg Beijing: Springer, 2009: 254–265.

(Edited by HE Yun-bin)

Cite this article as: ZHOU Zhou, HU Zhi-gang, YU Jun-yang, Jemal Abawajy, Morshed Chowdhury. Energy-efficient virtual machine consolidation algorithm in cloud data centers [J]. Journal of Central South University, 2017, 24(10): 2331–2341. DOI: https://doi.org/10.1007/s11771-017-3645-z.