Hindawi
Scientific Programming
Volume 2019, Article ID 3204346, 16 pages
https://doi.org/10.1155/2019/3204346

Research Article

An Energy and SLA-Aware Resource Management Strategy in Cloud Data Centers

Chi Zhang,[1] **Yuxin Wang**,[2] **Yuanchen Lv**,[2] **Hao Wu**,[1] **and He Guo**[1]

[1]*School of Software Technology, Dalian University of Technology, Dalian 116024, China*
[2]*School of Computer Science and Technology, Dalian University of Technology, Dalian 116024, China*

Correspondence should be addressed to Yuxin Wang; wyx@dlut.edu.cn

Received 2 September 2019; Revised 18 October 2019; Accepted 24 October 2019; Published 18 November 2019

Academic Editor: Basilio B. Fraguela

Reducing energy consumption of data centers is an important way for cloud providers to improve their investment yield, but they must also ensure that the services delivered meet the various requirements of consumers. In this paper, we propose a resource management strategy to reduce both energy consumption and Service Level Agreement (SLA) violations in cloud data centers. It contains three improved methods for subproblems in dynamic virtual machine (VM) consolidation. For making hosts detection more effective and improving the VM selection results, first, the overloaded hosts detecting method sets a dynamic independent saturation threshold for each host, respectively, which takes the CPU utilization trend into consideration; second, the underutilized hosts detecting method uses multiple factors besides CPU utilization and the Naive Bayesian classifier to calculate the combined weights of hosts in prioritization step; and third, the VM selection method considers both current CPU usage and future growth space of CPU demand of VMs. To evaluate the performance of the proposed strategy, it is simulated in CloudSim and compared with five existing energy–saving strategies using real-world workload traces. The experimental results show that our strategy outperforms others with minimum energy consumption and SLA violation.

1. Introduction

Cloud computing [1] has revolutionized the ownership model of IT infrastructure by offering on-demand provisioning of elastic resources [2]. Due to its flexibility, low-latency, and parallel processing capability, it has become a suitable and popular platform in many areas. Many industry magnates, such as Google, IBM, Microsoft, and Amazon, have begun to put massive manpower and financial resources to promote the commercialization of cloud computing and related services [3]. A number of large-scale data centers have been built all around the world. Since the average energy consumption of a data center is almost as much as 25000 households', the rapid expansion of the number of data centers must be accompanied by the fast increasing in energy demand. Such high energy consumption can directly lead to the increasing of carbon dioxide (CO_2) emissions and operational costs of data centers [4]. In view of global warming and the return on investment reducing, the issue of high energy consumption of data centers has aroused great concern from both governments and cloud providers. Consequently, improving the energy efficiency and eliminating unnecessary energy costs have become hot spots in the industry and the main difficulty and challenge of the next-generation data centers.

Many infrastructure-based solutions have been made to deal with the problem [5], but their implementations are expensive, and the reduction of energy consumption is limited [6]. In addition, apart from the huge quantity and low power efficiency of infrastructure, inefficient usage of computing resources is another reason for the high energy consumption in cloud data centers. By collecting data from more than five thousand hosts over six months, a fact was found that the hosts in the data centers are rarely idle or fully utilized, and for most of the time, only 10% to 50% of their full capacity are operated [7]. Moreover, it is important to realize that the idle host uses about 70% of its peak power consumption. All the above data indicate that inefficient resource usage leads to huge amount of energy wastage. Therefore, although many remarkable improvements on infrastructure have been made, designing effective resource management strategies to improve resource utilization is still necessary and meaningful in further decreasing energy consumption of a data center.

To address this problem, the capabilities of virtualization technology [8] should be well utilized. First, it allows multiple virtual machines (VMs) to be created on a single host and mapped to different consumers, which increases the throughput and scalability of a data center. Second, it provides a function named live migration [9]; in this way, a VM can be transformed between hosts with a close to zero downtime. With the support of dynamic migration, dynamic VM consolidation has emerged as the most popular strategy in this area recently. VMs are reallocated periodically in the dynamic VM consolidation method: some VMs are migrated from overloaded hosts to avoid performance degradation; all VMs on underutilized hosts are moved out to shut these hosts down to minimize the number of active hosts. But it should be stressed that excessive resource utilization may affect the performance of cloud services. For instance, resources requirements of some VMs may increase a lot abruptly, and during VM live migration process, resources are occupied on both source and target hosts. Maintaining a reliable Quality of Service (QoS) is essential for cloud providers as consumers pay for the services they get. SLA is the concrete form of QoS, which describes various details of service level provided to consumers [10]. Improper migrations and unconstrained VM consolidations can cause performance degradation of VMs and then lead to SLA violation. Then, a penalty must be paid to the customer, which will increase the total costs of cloud providers. Therefore, the trade-off between energy consumption and SLA violation should be found in the VM dynamic consolidation strategy.

In this paper, we propose an energy and SLA-aware resource management strategy based on dynamic VM consolidation. It intends to improve the resource utilization and the status of VM allocation in cloud data center, and then the energy consumption can be reduced while meeting the QoS delivered by cloud providers. Generally, for the dynamic VM consolidation, four subproblems need to be seriously considered: (1) overloaded hosts detection; (2) VM selection from overloaded hosts; (3) underutilized hosts detection; and (4) VM placement [11]. The proposed strategy contains methods to deal with the subproblems mentioned above. Finally, we run it on the CloudSim toolkit with real-world workload traces. Furthermore, the superiority of this strategy is demonstrated by comparing with several existing strategies. The proposing of some new and effective parameters in the strategy makes it more reasonable in the detection of overloaded and underutilized hosts and the selection of VMs from overloaded hosts than the existing strategies. Specifically, the differences from previous works along with the main contributions we made are listed as follows

(1) For overloaded hosts detection, previous methods either set a common upper threshold for all hosts or take the host as the basic investigation unit to obtain its upper threshold, which makes them naïve and unreasonable. In our method, we introduce a dynamic independent saturation threshold for each host. When calculating the saturation threshold of a host, each VM in it is considered as the basic investigation unit; that is, parameters such as the type and CPU usage of each VM are considered, as well as the number of VMs on it. Accordingly, there adds a new host state type, saturated state. Meanwhile, this method takes the CPU utilization trend of host into account by introducing the saturation degree.

(2) Instead of just considering CPU utilization as most of the previous overloaded hosts detecting methods do, a new indicator for candidate hosts is introduced in priority calculation process in our method. This indicator considers both the CPU usage of each VM and the number of VMs to improve the performance of the detection. In addition, the Naive Bayesian classifier is applied for predicting the variation trend of the indicator.

(3) In order to accommodate these changes above, we also present a new VM selection method. For the purpose of reducing energy consumption and SLA violation, the basic idea of our method is reducing the number and cost of migrations. So, it takes both the current CPU usage and the future growth space of CPU demand of VMs into consideration, which makes it more comprehensive than the previous works.

The rest of the paper is organized as follows. The previous works related to energy-aware resource management are presented in Section 2. Section 3 is the main part of this paper, which introduces our strategy and the correlative methods in detail. Experimentation setup is depicted in Section 4. Experimental results are given and analyzed in Section 5. Finally, Section 6 provides the conclusion of our research.

2. Related Work

Many works have done to provide high-quality serveries with minimal energy consumption in cloud data centers except infrastructure-based optimizations. In general, depending on whether they are implemented at hardware or software level, the mainstream energy-aware resource management strategies can be divided into two types.

2.1. Hardware Strategies. Hardware strategies employ parallel architectures, multicore architectures, voltage and frequency scaling, and dynamic component consolidation and deactivation to reduce energy consumption of hardware in cloud data centers. The DVFS introduced above is the most popular one among them [12]. By employing this technique, the CPU can adjust its performance dynamically. Specifically, in order to save energy consumption, the voltage and frequency of CPU will be reduced when it is not fully utilized. The DVFS has improved energy consumption issue to some extent, but it has some limitations. The methods based on DVFS are static and offline, which means the workload traces should be notified in advance, or the future CPU utilization should be predicted by leveraging the knowledge of past periods. So, they may not be suitable for using when the workload trace is unknown and irregular.

2.2. Software Strategies. Most of the software strategies introduce significant VM dynamic consolidation methods to optimize resource utilization and reduce energy consumption along cloud data center. Zhu et al. [13] studied dynamic VM consolidation problem of automated resource allocation and capacity planning. They set a static CPU utilization upper threshold of 85% and introduce a heuristic method for detecting overloaded hosts. The value of 85% was proposed by Gmach et al. [14] for the first time, based on their study of real workload. Beloglazov and Buyya [15] divides the VM allocation into two parts, allocation of new requested VMs and optimization of current placements of existing VMs. The first part is considered as a bin-packing problem, and this paper solves it by applying Modified Best Fit Decreasing (MBFD) method. For the second part, they propose four heuristics methods for choosing VMs to migrate. The four methods are Single Threshold (ST), Minimization of Migrations (MMs), Highest Potential Growth (HPG), and Random Choice (RC). Meanwhile, the authors present a decentralized architecture of the energy-aware resource management system and three stages of VM placement optimization in [16]. The stages are VM reallocation considering current resource utilization, virtual network topologies optimization, and VM reallocation considering thermal state of hosts. They prove that their heuristics perform better than DVFS.

In order to adapt to variable and unknown workload, several strategies are focusing on adopting statistical analysis of historical data. Beloglazov and Buyya [17] give a competitive analysis and prove competitive ratios of the single VM migration and dynamic VM consolidation problems. Furthermore, they propose an adaptive double CPU utilization thresholds method. In [11], they summarize and extend their previous work. The problem of dynamic VM consolidation is split into four parts, and they put forward several heuristic methods for each part. To find overloaded hosts, there are four statistical methods: Median Absolute Deviation (MAD), Interquartile Range (IQR), Local Regression (LR), and Local Regression Robust (LRR). Minimum Migration Time (MMT), Maximum Correlation (MC), and Random Choice (RC) are proposed to deal with the subproblem of VM selection. They also propose a simple method for underutilized hosts detecting and use Power-Aware Best Fit Decreasing (PABFD) for VM placement. Arianyan et al. [18] introduce a holistic method for resource management procedure in cloud data centers, which is called Enhanced Optimization (EO). Besides, Technique for Order of Preference by Similarity to Ideal Solution (TOPSIS) power and SLA-aware allocation (TPSA) method are proposed as the resource allocation methods. Moreover, for underutilized hosts detecting, methods including Available Capacity (AC), Migration Delay (MDL), and TOPSIS-Available Capacity-Number of VMs-Migration Delay (TACND) are proposed. Yadav et al. [19] introduce Maximum Utilization Minimum Size (MuMs) based on statistical analysis of hosts CPU utilization history as the VM selection method. Then, Yadav and Zhang [20] propose an adaptive heuristic M estimation Regression (MeReg) method to estimate upper CPU utilization threshold using recent CPU utilization history. Yadav et al. [21] also propose a novel overloaded host detection method called Least Medial Square Regression (LmsReg) and a VM selection method called Minimum Utilization Prediction (MuP). LmsReg is more robust than other regression techniques. MuP considers the types of application running and CPU utilization at different time periods. A multiresource double-threshold method is proposed by Yadav et al. [22] who propose two regression-based methods named Gradient Descent-based Regression (Gdr) and Maximize Correlation Percentage (MCP) to set a dynamic CPU utilization upper threshold and a dynamic Bandwidth-aware (Bw) VM selection method. Based on the first-order Markov chain model, a load detection method named Median Absolute Deviation Markov Chain Host Detection method (MadMCHD) is proposed by Melhem et al. [23] to find the future overloaded and underutilized hosts. They also add the Markov prediction model into the PABFD and propose a Markov Power Aware Best Fit Decreasing (MPABFD) method for VM placement. Ranjbari and Torkestani [24] use the Learning Automata Overload Detection (LAOD) to predict the CPU utilization of a host upon its historical usage data and determine whether it is overloaded dynamically.

With the popularity of artificial intelligence techniques, some artificial intelligence strategies are proposed to give the most optimal VM allocation, which take advantage of various genetic methods, such as neural networks, machine learning, and fuzzy method. For example, Abd et al. [25] propose a DNA-based fuzzy genetic method (DFGA) that deals with real-time tasks of dynamic users to reduce power consumption in cloud data centers. An energy-aware VM scheduling approach named PreAntPolicy is introduced by Duan et al. [26], which consists of a fractal mathematics-based prediction model and a scheduler using an improved ant colony method. Li et al. [27] first develop a multiresource double-threshold method. Then, they introduce Modified Particle Swarm Optimization (MPSO) method into VM reallocation. Ghobaei-Arani et al. [28] propose a VM placement optimization method combining learning automata theory, correlation coefficient, and ensemble prediction model. However, these methods acquire long learning periods to give good solutions. Zhou et al. [29] introduce an adaptive three-threshold framework and use a method named K-Means clustering algorithm Midrange-Interquartile range (KMI) to get the three thresholds, then the hosts are divided into four classes: less loaded hosts, little loaded hosts, normally loaded hosts, and overloaded hosts. Based on this framework, they also put forward two VM selection methods named Maximum ratio of CPU utilization to memory utilization (MRCU) and Minimum the product of a CPU utilization (MPCU) for CPU intensive and I/O intensive workload, respectively, and a VM placement method named VM Placement with Maximizing energy Efficiency (VPME).

3. Energy and SLA-Efficient Resource Management Strategy

In this section, we give the detailed introduction of the proposed energy and SLA-aware resource management strategy. For subproblems in dynamic VM consolidation, it contains three improved methods to complete overloaded hosts detecting, underutilized hosts detecting, and VM selection. Meanwhile, it uses the existing PABFD method for VM placement. In order to explain the working process of the whole resource management strategy and the relations between the four methods, we give a flow chart in Figure 1. The acronyms in the figure are the name of the methods that are detailed in the rest of this section.

When a host is judged as overloaded, some VMs selected on it are put into the migration list. However, the VMs on the migration list will not be reallocated until all hosts have been detected. In contrast, each time an underutilized host is identified, it is necessary to reallocate all its VMs immediately, and then the underutilized hosts detecting process for the next candidate host can proceed.

For ease of reference and understanding, Table 1 summarizes the acronyms of the terms defined in this section.

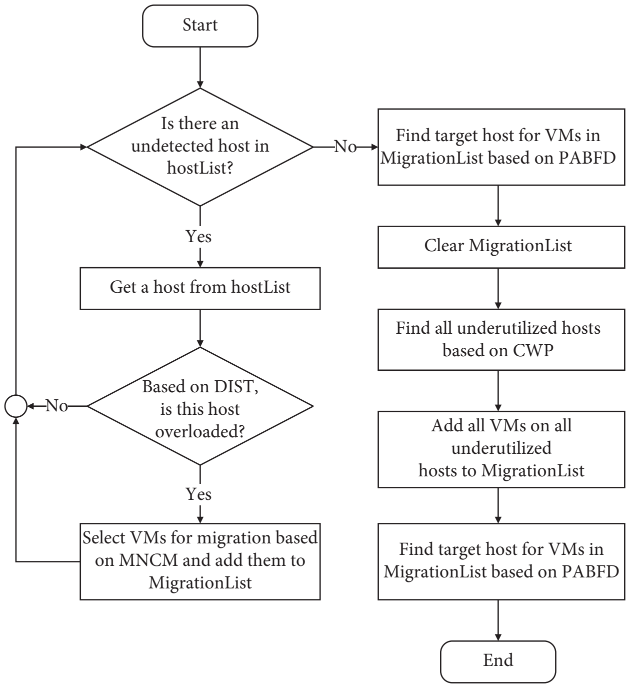

Start
Is there an undetected host in hostList?
No
Yes
Get a host from hostList
Based on DIST, is this host overloaded?
No
Yes
Select VMs for migration based on MNCM and add them to MigrationList
Find target host for VMs in MigrationList based on PABFD
Clear MigrationList
Find all underutilized hosts based on CWP
Add all VMs on all underutilized hosts to MigrationList
Find target host for VMs in MigrationList based on PABFD
End

Figure 1: The flowchart of resource management strategy.

Table 1: Acronyms and full names of terms.

Acronym	Full name
ST	Saturation threshold
VMR	VM maximum amount of resource
VRA	VM resource allocation
VRAR	VM resource allocation rate
VRR	VM resource reservation
HRR	Host resource reservation
HMR	Host maximum amount of resource
SD	Saturation degree
HROR	Host resource occupancy rate
VRO	VM resource occupancy
HRO	Host resource occupancy

3.1. Overloaded Hosts Detecting Method.

3.1. Overloaded Hosts Detecting Method. Theoretically, a host only will be identified as overloaded if the total CPU demands of all VMs on it exceed its total CPU capacity at some point. But performance degradation and SLA violations are already inevitable on it at this time. In order to prevent these from happening, overloaded hosts detecting method is proposed in practice by setting an upper threshold. When the CPU utilization of a host exceeds this threshold, it will be identified as overloaded, and next, some of the VMs must be migrated from it to other hosts to reduce its CPU utilization to normal. Therefore, overloaded hosts detecting method can avoid SLA violations caused by the sudden increase of CPU demands of some VMs.

Clearly, in overloaded hosts detecting process, how to determine the appropriate value of upper threshold is the key problem. At first, it should be noted that using the same upper threshold to determine whether the state of the host is overloaded is not reasonable even for hosts with the same configuration. Because the number, type, and CPU usage of VMs on them are different. Two extreme examples are given in Figure 2 to illustrate the irrationality of using a common upper threshold for all hosts.

For illustration purposes, the actual unit of CPU capacity is not used in the examples. Shaded areas represent idle CPU capacity. The two hosts H_1 and H_2 have the same total CPU capacity of 100 and the same upper threshold of 80%. VM_1 is the only VM on H_1, and the maximum amount of CPU can be required by VM_1 is 100. In the present moment, the CPU usage of VM_1 is 90, so the CPU utilization of H_1 is 90%, which is larger than the upper threshold. H_1 is judged as overloaded and VM_1 must be migrated to other hosts. However, as long as no new VMs are created on H_1, even if the CPU usage of VM_1 increases to its maximum amount, the CPU demand on H_1 will not exceed its total CPU capacity. Furthermore, the migration of VM_1 is unwise, as its CPU usage is huge; first, the time and cost of this migration are extremely large; second, it easily leads to the overload of the destination host of the migration at the next time point. For this situation, we think that keeping VM_1 on H_1 and just not allowing H_1 to add new VMs is better than treating H_1 as overloaded. There are 60 VMs on H_2, the maximum amount of CPU can be required by each VM is 100. At the present moment, the CPU usage of each VM is 1, so the CPU utilization of H_2 is 60%, which is smaller than the upper threshold, and H_2 is judged as not overloaded. However, if the CPU usage per VM increases slightly, for example by 1, at the next time point, the CPU demand on H_2 will increase to 120, which exceeds its total CPU capacity. Then the upper threshold is ineffective for H_2 at the present moment. Therefore, hosts should be judged separately by giving each

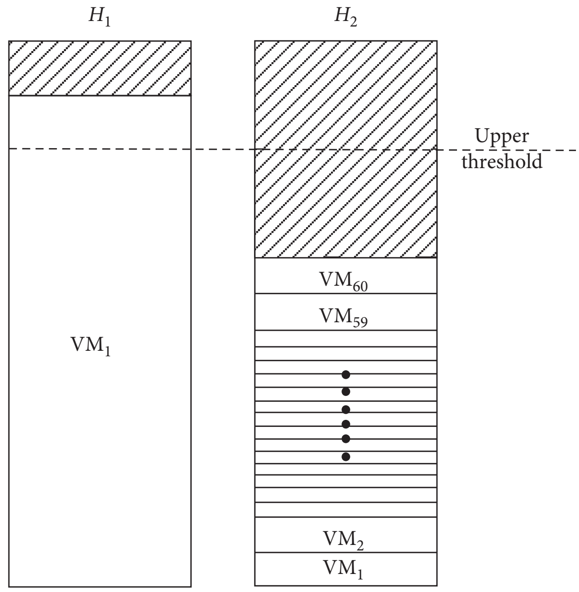


Figure 2: Examples for illustrating the irrationality of using common upper threshold.

of them an independent threshold. In addition, when the CPU utilization of a host exceeds some certain threshold, it may not need VM migrations, and just limiting the creation of new VMs can prevent it from overloading.

Similarly, the total CPU utilization of a host is not a complete reflection of its state, so taking host as the basic unit of investigation to get the upper threshold is also irrational. The actual situation of every VM on the host should be reflected directly in the calculation of the threshold. Based on these considerations above, we introduce a new threshold named Saturation Threshold (ST). Each host has its own private ST, and ST changes dynamically with the actual situation of every VM on the host. Before giving the calculation formula of ST, several concepts need to be clarified.

3.1.1. VM Maximum Amount of Resource (VMR). VMR is determined by the type a VM belongs to, and it is equal to the maximum amount of CPU available for that type. Since all the VM types are known and fixed, VMR of a VM is also a known fixed value.

3.1.2. VM Resource Allocation (VRA) and VM Resource Allocation Rate (VRAR). In the actual situation, in order to reduce operating costs, instead of allocating VMR to a VM, the cloud provider only allocate the amount of CPU that it needs in a moment for the VM to use. Therefore, VRA equals to the actual CPU usage of a VM. Then, $VRAR$ is calculated as

$$\mathrm{VRAR} = \frac{\mathrm{VRA}}{\mathrm{VMR}}. \quad (1)$$

3.1.3. VM Resource Reservation (VRR) and Host Resource Reservation (HRR). Each VM is treated as an independent object when calculating VRR. Depending on CPU request and usage of a VM, a part of CPU capacity is reserved on the host for its future usage. Since the VMR of a VM is fixed, with the increase of its VRA, the growth space of its future CPU demand decreases, and accordingly, its VRR should be reduced. Based on this fact, the calculation formula of VRR is given as

$$\mathrm{VRR} = (1 - \mathrm{VRAR}) \times \mathrm{VRA}. \quad (2)$$

The sum of VRRs of all VMs on a host is called the HRR, so as the number of VMs increases, HRR also increases.

Finally, the calculation formula of ST is given as

$$\mathrm{ST} = 1 - \frac{\mathrm{HRR}}{\mathrm{HMR}}. \quad (3)$$

In equation (3), the acronym HMR stands for Host Maximum amount of Resource; it is the total CPU capacity that the host can provide. The definition of HRR is closely related to the number of VMs and the CPU usage of each VM on a host, so using HRR in the calculation of ST means we also take these two parameters into account in our overloaded hosts detecting method.

If the CPU utilization of a host exceeds its ST, the host is marked as saturated. VRR of a VM is not really allocated to it and HRR of a host is a part of CPU capacity that can be shared by all VMs on it. Therefore, it should be noted that immediate VM migrations are not required on a saturated host; it simply no longer accepts VM allocation.

When judging whether a host is overloaded, the changing trend of its CPU utilization is also a parameter that cannot be ignored. In order to add this into consideration, a concept named Saturation Degree (SD) is introduced. It is the extent to which its CPU utilization exceeds ST and can be calculated as

$$\mathrm{SD} = \frac{\mathrm{Utilizaion} - \mathrm{ST}}{\mathrm{ST}}. \quad (4)$$

If a host stays saturated and its SD increases continuously at n consecutive monitoring points (the points at which SD remains the same are excluded), the state of the host will be changed from saturation to overload. n is an adjustable parameter, its value can be optimized and finally determined through experiments.

The pseudocode of our overloaded hosts detecting method is shown in Algorithm 1. It is referred to as Dynamic Independent Saturation Threshold (DIST) method. In order to get ST, SD, and the CPU utilization of a host, all VMs on it will be traversed. So, the time complexities of them are $O(N)$. The rest of DIST uses numerical comparisons to determine if the host is overloaded, and the time complexity is $O(1)$. Therefore, the time complexity of DIST for one host is $O(N)$, where N is the number of VMs on the host, and then, the time complexity of the entire overloaded hosts detecting process for all hosts is $O(M \times N)$, where M refers to the number of hosts.

3.2. Underutilized Hosts Detecting Method. Migrating all VMs from underutilized hosts and then shutting them off or setting them to deep sleep mode is an efficient way to

```
Input: host
Calculate ST and SD of the host
utilization = host.getCpuUtilization ( );
if utilization > ST
    if SD > lastSD
        saturated_count + +;
    else if SD < lastSD
        saturated_count = 0;
    end if
    if saturated_count < n
        host_state = Saturated;
        lastSD = SD;
    else
        host_state = Overloaded;
        MigrationList.add (host.getVMsToMigrate ( ));
        saturated_count = 0;
        lastSD = 0;
    end if
else
    saturated_count = 0;
    lastSD = 0;
end if
return
```

Algorithm 1: DIST.

increase CPU utilization and reduce energy consumption of a cloud data center. In the proposed underutilized hosts detecting method, all active hosts except saturated ones should be put into a candidate host set for detection. We first get the priorities of all candidate hosts and then try to migrate all VMs from the host with highest priority to other unsaturated hosts while keeping them not overloaded. If the entire migration process is successfully completed, the host with highest priority is marked as underutilized, and it will be turned off or switched to deep sleep mode after all VM migrations are done. Otherwise, it will remain active. The host will be removed from the candidate host set after detection. Meanwhile, some candidate hosts have just accepted the migrated VMs, so the candidate host set and the priority of all candidate hosts should be updated. The underutilized hosts detecting method does not terminate until there is no host in the candidate set.

In priority calculation process, unlike previous works which simply use CPU utilization of a host to decide its priority, we want to take more factors into consideration to improve effectiveness. Therefore, a new indicator named Host Resource Occupancy Rate (HROR) is proposed. To explain it, the definition of Resource Occupancy is given at first.

3.2.1. VM Resource Occupancy (VRO) and Host Resource Occupancy (HRO). VRO is the sum of VRA and VRR of a VM. HRO is equal to the sum of VROs of all VMs on that host.

The calculation formula of HROR is

$$\mathrm{HROR} = \frac{\mathrm{HRO}}{\mathrm{HMR}}. \qquad (5)$$

The reasons for using HROR to calculate the priority of a host are as follows. First, besides the CPU utilization of hosts at present, their possible future CPU demands and maximum CPU capacities are also critical for prioritizing them. Second, it is important to take the number of VMs into account in the priority calculation method, because the more VMs a host has, the greater the probability that it will not be underutilized in the future. Comprehensive consideration of them can obviously improve the effectiveness of underutilized hosts detection. It should be noted that, according to their definitions, VRO can simultaneously reflect the actual CPU usage of the VM at present and the CPU capacity should be reserved for its future usage; HRO is related the number of VMs on the host; the calculation of HROR uses HMR. Therefore, using HROR to calculate the priority of a host is much better than using the CPU utilization undoubtedly.

In addition, the impact of variation trend of HROR is another important factor which should be taken into consideration in priority calculation. Specifically, for hosts with approximately equal HROR values at one monitoring point, the one should have higher priority if its HROR values are likely to decrease at the next monitoring point. We use the Naive Bayesian classifier to get the probability that HROR decreases. Then based on HROR and the probability, an indicator named Adjusted Host Resource Occupancy Ratio (AHROR) is proposed.

In the Naive Bayesian classifier, we need data samples to form a training set, and each sample is represented by a $m + 1$ dimension vector $(a_1, a_2, \ldots, a_m, c)$. Each vector consists of n feature attributes and a class label. There may exist k classes, so the range can be expressed as $\{C_1, C_2, \ldots, C_k\}$. After training, for a sample which has no class label, the classifier will predict that it belongs to the class which has the highest posterior probability conditioned on the sample vector.

As we intended to use the historical data of HROR to predict its probabilities of decreasing or not decreasing at the next monitoring point, according to the Naive Bayesian classifier, the direct method is to choose the historical data as the features of sample vector. Suppose $x_t, x_{t+1}, \ldots, x_{t+m}$ are $m+1$ HRORs observed from preceding monitoring points in time $t, t+1, \ldots, t+m$, then we get the input feature vector $X = (x_t, x_{t+1}, \ldots, x_{t+m})$. The variation of HROR can be divided into two types, decreasing and not decreasing, so the range of the class is $\{0, 1\}$. Specifically, the class 1 is the state of decreasing and the class 0 is the state of not decreasing. Similarly, for simple and efficient use of the input vector X, the vector X will be transformed to vector $Y = (y_1, y_2, \ldots, y_m)$ using the rule shown in the following equation:

$$y_i = \begin{cases} 1, & \text{if } x_{t+i-1} > x_{t+i}, \\ 0, & \text{otherwise,} \end{cases} (1 \le i \le m). \tag{6}$$

For an input vector Y, our goal is to calculate $P(1 \mid Y)$ and $P(0 \mid Y)$:

$$P(1 \mid Y) = \frac{P(Y \mid 1) P(1)}{P(Y)}, \tag{7}$$

$$P(0 \mid Y) = \frac{P(Y \mid 0) P(0)}{P(Y)}. \tag{8}$$

The class conditional probabilities $P(Y|1)$ and $P(Y|0)$ can be calculated by the following equations:

$$P(Y \mid 1) = \prod_{i=1}^{m} P(y_i \mid 1), \tag{9}$$

$$P(Y \mid 0) = \prod_{i=1}^{m} P(y_i \mid 0). \tag{10}$$

The probabilities $P(y_i \mid 1)$ and $P(y_i \mid 0)$ can be got based on training samples.

$$P(y_i \mid 1) = \frac{P(1, y_i)}{P(1)} = \frac{s_{1,y_i}}{s_1}, \qquad P(y_i \mid 0) = \frac{P(0, y_i)}{P(0)} = \frac{s_{0,y_i}}{s_0}, \tag{11}$$

where s_{1,y_i} is the number of training samples of class 1 having the value y_i for its ith feature, and s_1 is the number of training samples belonging to class 1; s_{0,y_i} is the number of training samples of class 0 having the value y_i for its ith feature, and s_0 is the number of training samples belonging to class 0. For the special case where s_{1,y_i} or s_{0,y_i} is 0, Laplace smoothing can be used to solve it.

Besides, in (7) and (8), $P(1)$ and $P(0)$ are the class prior probabilities which can be estimated by the following equations:

$$P(1) = \frac{s_1}{s}, \tag{12}$$

$$P(0) = \frac{s_0}{s}, \tag{13}$$

where s is the total number of training samples.

Finally, the calculation of AHROR is given in equation (14). A host with a smaller AHROR should have higher priority:

$$\text{AHROR} = \begin{cases} \text{HROR}, & \text{if } P(Y \mid 1) P(1) \\ & \le P(Y \mid 0) P(0), \\ (1 - 0.1 \times P(1 \mid Y)) \times \text{HROR}, & \text{otherwise.} \end{cases} \tag{14}$$

Since the purpose of introducing variation trend of HROR is just to distinguish priorities of hosts with approximately equal HROR, we multiply $P(1 \mid Y)$ by 0.1 to reduce its weight. Though $P(1 \mid Y)$ cannot be got according to the formulas, we can get $P(Y \mid 1)P(1)$ and $P(Y \mid 0)P(0)$, and $P(Y)$ can be treated as a nonzero constant. In addition, $P(1 \mid Y) + P(0 \mid Y) = 1$, then the second case of equation (14) can be transformed into the following equation:

$$\begin{aligned} (1 - 0.1 \times P(1 \mid Y)) \times \text{HROR} &= \left(1 - 0.1 \times \frac{P(1 \mid Y)}{P(1 \mid Y) + P(0 \mid Y)}\right) \times \text{HROR} \\ &= \left(1 - 0.1 \times \frac{P(Y \mid 1) P(1)}{P(Y \mid 1) P(1) + P(Y \mid 0) P(0)}\right) \times \text{HROR}. \end{aligned} \tag{15}$$

In our experiment, the interval of measurements is five minutes, and the workload data of last nearest one hour is enough for predicting the state of next monitoring point, so we let $m = 12$. For each prediction, we use the last nearest 24 measurements to form 13 sample vectors as a training sample set, so the AHRORs of the first 24 monitoring points are equal to their HRORs.

The pseudocode of our underutilized hosts detecting method is shown in Algorithm 2. It is referred to as Combined Weight Prioritization (CWP) method. In order to get AHROR of a host, all VMs on it will be traversed, so the time complexity is $O(N)$. The rest of CWP uses double circulation to determine if the host is underutilized, and the time complexity is $O(M \times N)$. Therefore, the time

```
Input: hostList
Put all active hosts except saturated ones into candidatehostList
Calculate AHROR of each host in candidatehostList
candidatehostList.sortByDecreasingAHROR( );
for (host: candidatehostList)
    for (VM: VMList)
        if (getNewVMPlacement(VM) == null)
            Destroy all VM reallocations of the host;
            continue;
        end if
    end for
    host_state = Underutilized;
    Update candidatehostList;
end for
return
```

Algorithm 2: CWP.

complexity of CWP is $O(M \times N)$, where N is the number of VMs, while M refers to the number of hosts.

3.3. VM Selection Method. Determining which VMs to migrate from an overloaded host has a direct impact on the number and cost of migrations, i.e., inappropriate selections can lead to extra SLA violations, which in turn can increase energy consumption. In our consideration, for the VMs on an overloaded host, the one with bigger VRA and smaller VRAR should be prioritized for migration. Bigger VRA means it takes up a lot of CPU at present, and smaller VRAR means it has larger growth space of CPU demand in the future. This rule makes the current migration of this VM makes more sense for making the host running properly in the future, and accordingly, the total number and cost of migrations will be reduced.

Considering that using two separate factors makes the selection process difficult, it is better to find one factor that can reflect them both simultaneously. According the definition of VRR, for the two VMs with the same VRA, the one has smaller VRAR must has larger VRR. So, VM with bigger VRA and VRR should be selected first. VRO is the sum of VRA and VRR, so the selection should be based on VRO. In conclusion, the VM with larger VRO should have higher priority to be selected for migration.

The proposed VM selection method is referred to as Minimize Number and Cost of Migrations (MNCM) method. VRO of each VM is already obtained in the previous part, and all VMs on the overloaded host will be traversed for selecting proper ones; therefore, the time complexity of MNCM is $O(N)$, where N is the number of VMs.

4. Experimental Setup

In this section, the simulator, hosts and VMs characteristics, workload data, and performance metrics in our experiment are described in detail.

4.1. Simulator. It is essential to evaluate the proposed energy and SLA-efficient resource management strategy and compare it with the previous works on a large-scale data center infrastructure. However, experimentation on a real cloud data center is expensive and time-consuming. Moreover, real cloud data centers are proprietary and invisible to consumers. The experiment results are often difficult to reproduce and analyze. In addition, the influence of network and data transmission cannot be ignored, which will lead to inaccurate evaluation of energy consumption. To solve this issue, many simulators based on modeling and simulation technology are designed. They can provide an experimental environment which is very close to a real data center, and they make it much easier to evaluate and compare different resource management strategies. Considering the modern open-source CloudSim toolkit can provide reproducible results to check the cloud strategies and has very good support for energy consumption mode [30], CloudSim 4.0 is chosen as the experimental platform in this paper. More details of CloudSim are given in [31, 32].

4.2. Configuration of Hosts and VMs. In the simulation, we implement a data center which contains 800 heterogeneous hosts: half of them are HP ProLiant G4, the other half are HP ProLiant G5. The specific characteristics of these two types of servers [11] are listed in Table 2. Referring Amazon EC2, we set up four types of VMs, and their characteristics [11] are depicted in Table 3. Initially, the resources of each VM are allocated according to the resource requirements defined by its type. Then, less resources are allocated to VMs according to their workload during their lifetime dynamically, which can create opportunities for VM dynamic consolidation.

4.3. Workload Traces. To make the experiment more convincing, it is necessary to use real workload data. In this paper, the data used is derived from the CoMon project which is a monitoring infrastructure for PlanetLab [33]. We use 10 days' workload traces collected from more than 1000 VMs on 800 hosts located at more than 500 places throughout the world [11] as shown in Table 4. These traces characterize CPU utilization in 5 min intervals of the VMs.

Table 2: Configuration of two types of hosts.

Host type	CPU type	Cores	Frequency (MHz)	RAM (GB)
HP ProLiant G4	Intel Xeon 3040	2	1860	4
HP ProLiant G5	Intel Xeon 3075	2	2660	4

Table 3: Configuration of four types of VMs.

VM type	CPU (MIPS)	RAM (MB)
High-CPU medium	2500	870
Extra large	2000	1740
Small	1000	1740
Micro	500	613

Table 4: Characteristics of 10 days' workload traces (based on CPU utilization).

Date	Num. of VMs	Mean (%)	SD (%)
March 3, 2011	1052	12.31	17.09
March 6, 2011	898	11.44	16.83
March 9, 2011	1061	10.70	15.57
March 22, 2011	1516	9.26	12.78
March 25, 2011	1078	10.56	14.14
April 3, 2011	1463	12.39	16.55
April 9, 2011	1358	11.12	15.09
April 11, 2011	1233	11.56	15.07
April 12, 2011	1054	11.54	15.15
April 20, 2011	1033	10.43	15.21

4.4. Performance Metrics. The goal of an energy-aware resource management strategy is to minimize the power consumption and SLA violation of the data center. To verify its effectiveness, we choose energy consumption, SLA violation metrics, energy efficiency, number of VM migrations, and number of host shutdowns as performance metrics to evaluate our strategy.

4.4.1. Energy Consumption. In comparison to other resources like memory, disk storage, and network, it has been shown that the energy consumption of a host is mostly consumed by its CPU. Even if the DVFS technique is applied, the energy consumption of a host can be approximated by a linear model with its CPU utilization. However, the introduction of modern multicore CPUs, large memory, and big hard disks makes the traditional linear model inaccurate and makes the establishment of accurate analysis model to describe the energy consumption of the host complex. Therefore, we use the real data of energy consumptions of HP ProLiant G4 and HP ProLiant G5 under different CPU utilizations derived from SPECpower benchmark (http://www.spec.org/powerssj2008/). The details of the data are shown in Table 5.

Table 5: Power consumption of hosts under different CPU utilizations.

CPU utilization (%)	Power consumption (W)	
	HP ProLiant G4	HP ProLiant G5
0	86	93.7
10	89.4	97
20	92.6	101
30	96	105
40	99.5	110
50	102	116
60	106	121
70	108	125
80	112	129
90	114	133
100	117	135

4.4.2. SLA Violation Metrics. The values of SLA violation metrics are key indicators to evaluate QoS of data center. The CPU demand of a VM arbitrarily varies over time, and SLA violations will be caused if the host is oversubscribed. Two metrics have been introduced in [11] to depict SLA violation. They are SLA violation Time per Active Host (SLATAH) and Performance Degradation due to Migration (PDM). VMs cannot be provided with their CPU demands if the host is experiencing the 100% CPU utilization, so SLATAH is SLA violations due to overutilization of hosts. PDM is the negative impact on the performance of a VM caused by its live migration process. The definitions of them are given as

$$\mathrm{SLATAH} = \frac{1}{M}\sum_{i=1}^{M}\frac{T_{s_i}}{T_{a_i}}, \tag{16}$$

$$\mathrm{PDM} = \frac{1}{N}\sum_{j=1}^{N}\frac{C_{d_j}}{C_{r_j}}, \tag{17}$$

where M and N denote the number of hosts and VMs in a data center, respectively; T_{s_i} is the time during which the host's CPU utilization reaches 100%; T_{a_i} is the total active time of the host; C_{d_j} is the estimated performance degradation of VM_j caused by VM migrations, and according to Dumitrescu and Foster [34], it is set to 10% of CPU utilization during the total migration time of VM_j; C_{r_j} is the total CPU capacity requested by VM_j.

As the two metrics are independent and equally important, SLA Violation (SLAV) is calculated by multiplying them together as

$$\mathrm{SLAV} = \mathrm{SLATAH} \times \mathrm{PDM}. \tag{18}$$

4.4.3. Energy Efficiency. A good energy-aware resource management strategy should minimize power consumption and SLAV simultaneously. However, the two metrics have a relationship of restricting each other, using them individually is hard to give an intuitive judgment of how good or bad a strategy is compared with others. Therefore, the energy efficiency (EE) proposed in [29] as shown in (19) is used as the other metric. Obviously, the strategy with bigger EE value performs better.

$$\mathrm{EE} = \frac{1}{(\text{energy consumption} \times \mathrm{SLAV})}. \tag{19}$$

4.4.4. Number of VM Migrations. VM migration is an expensive operation as it brings data transmission burden to the network and resources are occupied on both sources and destination hosts during the migration process.

4.4.5. Number of Host Shutdowns. Reduction in the number of switching state of hosts can lead to additional energy saving in data center, so a smaller value of number of host shutdowns represents the strategy has a better performance.

5. Experimental Results and Analysis

In this section, we first present the impact of the parameter n in the overloaded hosts detecting method, on the performance of the proposed resource management strategy and determine the optimal value for it. Then, the performance of our strategy is evaluated relying on the aforementioned metrics, and the experimental results are analyzed in comparison to some benchmark strategies.

5.1. Determine the Optimal Value of Parameter. As mentioned in Section 3.1, in DIST, the state of the host will be changed from saturation to overload if it stays saturated and its SD increases continuously at n consecutive monitoring points (the points at which SD remains constant are excluded). Theoretically, when the value of n is very small, the hosts are easy to get into overloaded state, and in the extreme case when the n is equal to 1, there is no host belonging to the saturated state, because as long as a host conforms to the criteria of saturated state, it is judged as overloaded and the VM selection and migration processes on it begins. Therefore, the number of VM migrations is large, and SLA violations and energy consumption caused by VM migrations are also very large. With the increase of n value, fewer and fewer hosts can change from saturated to overloaded state, the number of VM migrations will decrease and so do the SLA violations and energy consumption caused by them. However, when the value of n is too large, some saturated state hosts cannot be timely converted into overloaded state for VM migrations; then the resource requests of some VMs on them may not be satisfied, resulting in increasing SLA violations and energy consumption. Finally, when n exceeds a certain critical value, all hosts in saturated state will not become overloaded, and the number of VM migrations, SLA violations, and energy consumption all reach definite values and do not change with the increase of n value.

In order to find the most suitable value for n, we study it with the first three of the ten PlanetLab workload traces and using the metrics as evaluation criteria. The impact of n on the all metrics has been studied; however, for the sake of space, we only show the impact on energy consumption, SLA violations, energy efficiency, and number of VM migrations metrics. Moreover, we find that when n approaches 10, the values of all the metrics have been very stable, and the results obtained when n is 1 differ greatly from the results obtained when n is other values. So, in order to show the critical data more clearly, we draw the results with n values from 2 to 10 in Figures 3–6, and results obtained when n is 1 are listed separately in Table 6.

From Table 6, it is obvious that when n is 1, huge number of VM migrations occur in all three groups of experiments, and accordingly, the values of SLAV and energy consumption are also very large, and the values of energy efficiency are low. The change trend of the data in the figures basically conforms to our theoretical analysis above. It can be clearly seen from the figures that when n increases from 2 to 3, the values of number of VM migrations, SLAV and energy consumption decreases a lot, and the values of energy efficiency increases a lot, and when $n \geq 6$, the value of the four metrics tend to be stable. It should be noted that, though the value of energy consumption basically unchanged when n increases from 3 to 6, the values of number of VM migrations and SLAV first decrease in a certain extent when n increases from 3 to 5, and then increase when n increases to 6. Accordingly, when n increases from 3 to 6, the energy consumption first increases and then decreases and reaches the maximum when n is 5. As is clear from the above descriptions, we consider 5 as the most suitable value of n to reduce both energy consumption and SLA violations.

5.2. Comparison to Benchmark Strategies. In this section, the proposed strategy is compared with five existing energy–saving strategies which use THR (with static utilization threshold 0.8), LR (with safety parameter 1.2), IQR (with safety parameter 1.5), MAD (with safety parameter 2.5), and LAOD (with safety parameter 0.9) [24] in overloaded hosts detecting phase, respectively, and use a simple method (SM) [11] in the underutilized hosts detecting phase and MMT in the VM selection phase. Additionally, our strategy and all the comparing strategies use PABFD method in VM placement phase. The five overloaded hosts detecting methods have been explained in Section 2. Safety parameter is used to control aggressiveness of these methods for consolidating VMs. The smaller the parameter, the lower the energy consumption, but the higher the level of SLA violations caused by the consolidation. The value of the safety parameter selected for each method has shown to be optimal [11]. In SM, the host with minimum CPU utilization relative to the others will be considered as underutilized if all the VMs on it can be migrated onto others while keeping them not overloaded. MMT selects the VM that requires the minimum migration time relative to the others. The migration time is estimated as the amount of RAM utilized divided by the available network bandwidth. PABDF first sorts the VMs based on their CPU utilizations in an unincreased order and then allocates each VM to the host which will have the least increase in power caused by the allocation.

In the following, we use DIST/CWP/MNCM to represent our strategy, and the comparing strategies are THR/SM/MMT, LR/SM/MMT, MAD/SM/MMT, IQR/SM/MMT, and LAOD/SM/MMT. For each strategy, experiments are executed using the 10 days of workload traces depicted in Table 4 separately. The comparison of energy consumption, SLA metrics, number of VM migrations, as well as number

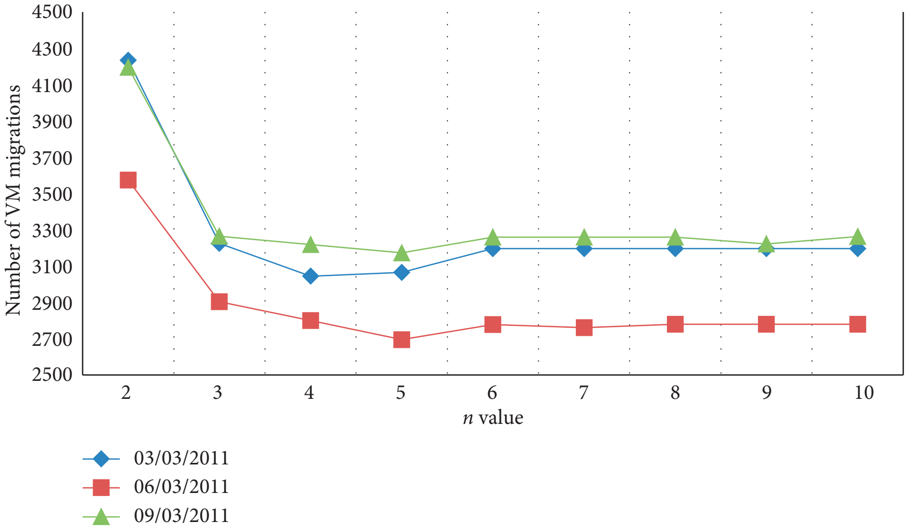


Figure 3: Impact of n on number of VM migrations. Comparison of number of VM migrations when *n* is assigned different values in DIST, using first three of the ten PlanetLab workload traces.

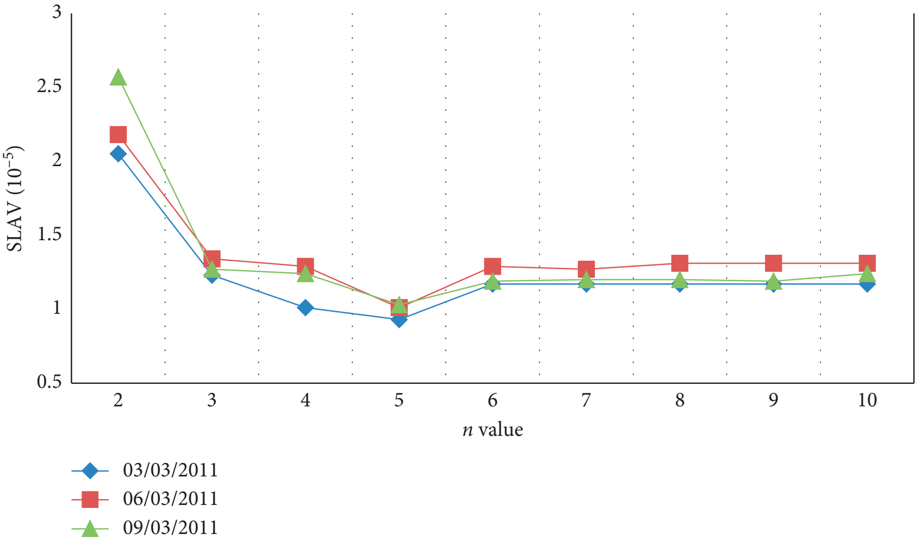


Figure 4: Impact of n on SLAV. Comparison of SLAV when *n* is assigned different values in DIST, using first three of the ten PlanetLab workload traces.

of host shutdowns of these strategies are reported in Figures 7–10. Each value in the bar graphs is the average value of ten results obtained using 10 days of data.

From Figure 7, it is obvious that the proposed strategy has a much smaller number of VM migrations compared with other strategies. Specifically, relative to the proposed strategy, LAOD/SM/MMT has the minimum difference and LR/SM/MMT has the maximum difference in the number of VM migrations. The range of difference reached 21359 to 24929, with a reduction rate range of 86.74% to 88.41%. The reason can be explained as follows: first, DIST and CWP consider the uniqueness of each host according to the actual situation of VMs on it when determining whether it is overloaded or underutilized; second, in DIST, a saturated host no longer accepts a VM allocation, which reduces the chance that it becomes overloaded and requires VM migrations; third, besides the current CPU usage, MNCM takes the future growth space of CPU demand into account. These methods make the host detecting results and the VM selection results more effective, and then the number of VM migrations is reduced.

As we can see in Figure 8, the proposed strategy also has a significant advantage on the number of host shutdowns. Since the proposed strategy properly chooses the underutilized hosts and the VMs need to be migrated from overloaded hosts, many unnecessary and incorrect migrations and the restarting of some previously shutdown hosts are prevented. As a result, it shuts down a much smaller number of hosts than the other strategies do. Compared to the proposed strategy, LR/SM/MMT has the minimum

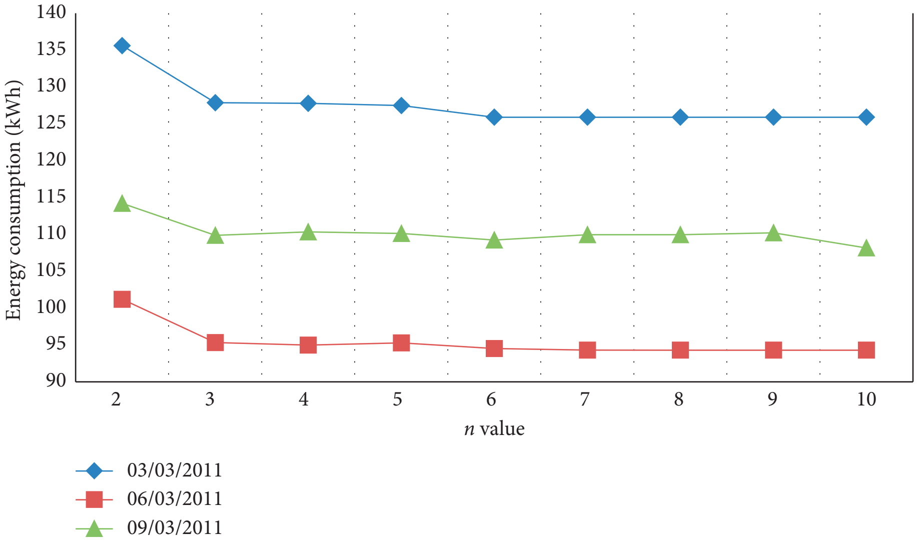


Figure 5: Impact of n on energy consumption. Comparison of energy consumption when *n* is assigned different values in DIST, using first three of the ten PlanetLab workload traces.

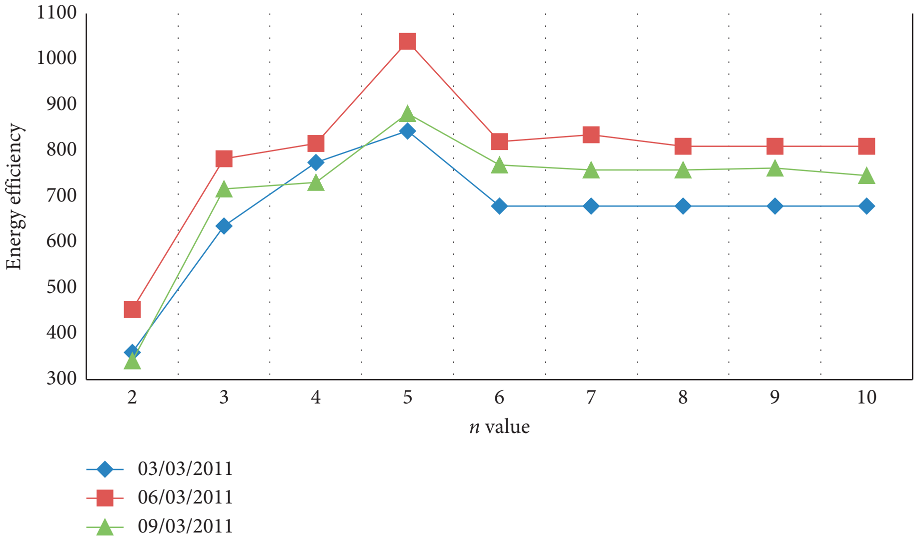


Figure 6: Impact of n on energy efficiency. Comparison of energy efficiency when *n* is assigned different values in DIST, using first three of the ten PlanetLab workload traces.

Table 6: Performance of the proposed strategy when *n* is 1 in DIST.

Date	Number of VM migrations	SLAV (10^{-5})	Energy consumption (kWh)	Energy efficiency
March 3, 2011	22898	9.69	190.91	54.06
March 6, 2011	17321	9.80	141.44	72.14
March 9, 2011	19179	10.13	159.95	61.72

difference and IQR/SM/MMT has the maximum difference in the number of host shutdowns. The range of difference reached 4261 to 4954, with a reduction rate range of 84.28% to 86.17%.

To save space, the comparisons of the proposed strategy to the benchmark strategies on the three SLA metrics are all shown in Figure 9. According to the results, the proposed strategy has smaller values of SLAV, PDM, and SLATAH, and compared to it, LAOD/SM/MMT has the minimum difference and LR/SM/MMT has the maximum difference in the SLA metrics. The range of difference reached 2.292 to 4.002, 0.28 to 0.44, and 2.35 to 3.53, with the ranges of reduction rates of 70.31% to 80.52%, 43.75% to 55%, and 46.81% to 56.94%, respectively. First, through introducing the saturated state, DIST prevents the CPU utilization of hosts from reaching 100%; consequently, SLATAH

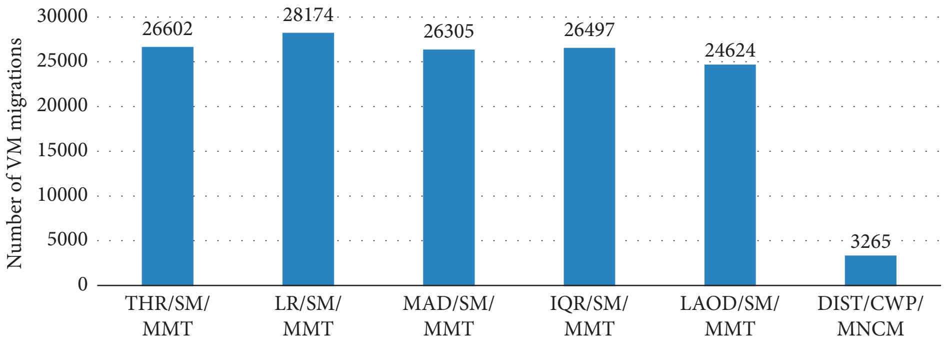


Figure 7: Comparison of number of VM migrations of strategies using PlanetLab workload traces.

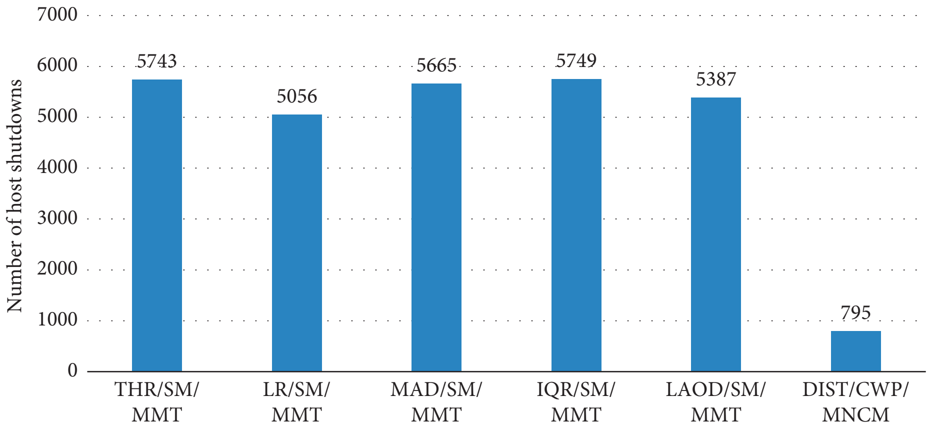


Figure 8: Comparison of number of host shutdowns of strategies using PlanetLab workload traces.

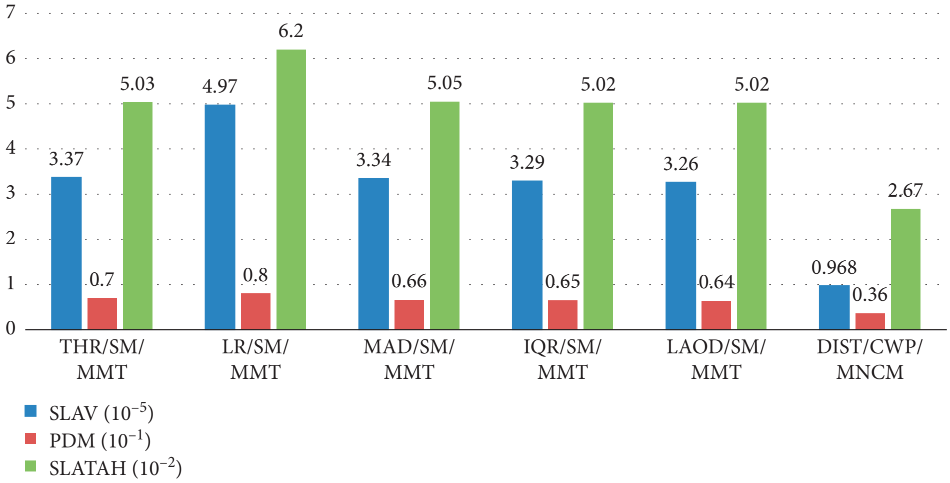


Figure 9: Comparison of SLA metrics of strategies using PlanetLab workload traces.

decreases; second, the number of VM migrations of the proposed strategy is much smaller than other strategies, and then the time cost and performance degradation due to migration are smaller; thus, PDM decreases. Therefore, SLAV formed by multiplying the two metrics is also reduced.

Figure 10 depicts the comparison on energy consumption of different strategies. Notably, the proposed strategy has smaller energy consumption value than others. Specifically, relative to the proposed strategy, LR/SM/MMT has the minimum difference and THR/SM/MMT has the maximum difference in the energy consumption. The range of the difference is 37.48 kWh to 64.12 kWh, and the range of the reduction rate is 23.16% to 34.02%. Since the proposed strategy has smaller values of above metrics in comparison to

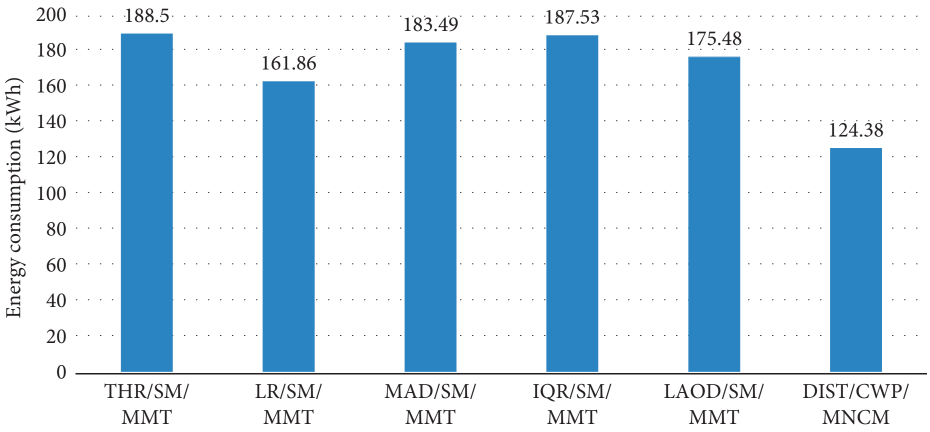


Figure 10: Comparison of energy consumption of strategies using PlanetLab workload traces.

Table 7: The respective effects of the three methods presented.

Strategy	SLAV (10^{-5})	Energy consumption (kWh)	Energy efficiency
LR/SM/MMT	6.94	130.89	110.09
DIST/SM/MMT	1.68	108.29	549.67
LR/CWP/MMT	3.89	73.44	342.12
LR/SM/MNCM	5.95	68.80	244.28
DIST/CWP/MNCM	1.39	96.94	742.13

the other strategies, and VM migrations and switching hosts state ON/OFF can produce extra energy consumption; this result is easy to understand.

In addition, in order to see the specific effect of our strategy in reducing energy consumption, we run NPA and DVFS to get their energy consumption values as benchmarks because they do not involve VM migration. NPA uses no energy management measures during workloads processing, and its energy consumption is 2410.8 kWh, and DVFS consumes 829.5 kWh. In comparison, the proposed strategy reduces energy consumption by 94.84% and 85.01%.

The above simulation results have shown that our strategy using the proposed three methods together produces much better performance compared to other combinations of existing methods. Then to demonstrate the validity and reliability of each of them, we combine them separately with other benchmark methods to compose various strategies. Extensive experiments are conducted for testing them, but to save space, only a selection of representative and illustrative results is listed in the paper. Table 7 shows the results of some experiments using the workload traces on April 20, 2011, and LR, SM, and MMT are taken as the benchmark methods for three phases of dynamic VM consolidation.

The first row is the baseline strategy, and the second to fourth rows are the strategies use one of the three methods. From these results, the three proposed methods work better than their corresponding benchmark methods. DIST greatly reduces SLAV as it introduces the saturated state for hosts, which prevents the CPU utilization of hosts from reaching 100%; CWP cut almost half of SLAV and energy consumption because it considers more factors and uses the Naive Bayesian classifier for prediction; and MNCM cut almost half of energy consumption because it has more comprehensive consideration in the selection of VMs. And, more remarkable, they can be well integrated. From the results shown in the last row, our strategy, DIST/CWP/MNCM, has the best result on energy efficiency compared to other combinations, that is, using them together can achieve the best overall performance.

6. Conclusions

For the energy consumption problem in cloud data centers, this study put forward a threshold-based energy and SLA-efficient resource management strategy to make a trade-off between energy consumption and SLA violation. For the subproblems in dynamic VM consolidation, the overloaded hosts detecting method DIST, the underutilized hosts detecting method CWP, and the VM selection method MNCM are proposed. Benefits from these methods are that the chance that hosts are being overloaded is reduced and underutilized hosts are turned off as much as possible. Meanwhile, the numbers of VM migrations and host shutdowns are well controlled. Therefore, energy consumption and SLA violation of the cloud data center are both reduced. The results of simulation experiments show that our strategy outperforms comparing strategies significantly on all evaluation metrics. As future work, more resource types, such as memory and network bandwidth, will be considered in addition to the CPU. Furthermore, we plan to further improve the performance of our strategy by using machine learning algorithms to predict future workloads based on historical data.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (grant nos. 11372067 and 61772112) and the Innovation Foundation of Science and Technology of Dalian (grant no. 2018J11CY010).

References

[1] M. Armbrust, A. Fox, R. Griffith et al., “Above the clouds: a Berkeley view of cloud computing,” Report No. UCB/EECS-2009-28, Department Electrical Engineering and Computer Sciences, University of California, Berkeley, Berkeley, CA, USA, 2009.

[2] R. Buyya, C. S. Yeo, S. Venugopal, J. Broberg, and I. Brandic, “Cloud computing and emerging IT platforms: vision, hype, and reality for delivering computing as the 5th utility,” *Future Generation Computer Systems*, vol. 25, no. 6, pp. 599–616, 2009.

[3] M. Tarahomi and M. Izadi, “A prediction-based and power-aware virtual machine allocation algorithm in three-tier cloud data centers,” *International Journal of Communication Systems*, vol. 32, no. 3, p. e3870, 2019.

[4] A. Beloglazov, R. Buyya, Y. C. Lee, and A. Zomaya, “A taxonomy and survey of energy-efficient data centers and cloud computing systems,” in *Advances in Computers*, pp. 47–111, Elsevier, Amsterdam, Netherlands, 2011.

[5] M. Blazek, H. Chong, W. Loh, and J. G. Koomey, “Data centers revisited: assessment of the energy impact of retrofits and technology trends in a high-density computing facility,” *Journal of Infrastructure Systems*, vol. 10, no. 3, pp. 98–104, 2004.

[6] J. D. Moore, J. S. Chase, P. Ranganathan, and R. K. Sharma, “Making scheduling “cool”: temperature-aware workload placement in data centers,” in *Proceedings of the USENIX Annual Technical Conference, General Track*, pp. 61–75, Anaheim, CA, USA, April 2005.

[7] L. A. Barroso and U. Hölzle, “The case for energy-proportional computing,” *Computer*, vol. 40, no. 12, pp. 33–37, 2007.

[8] P. Barham, B. Dragovic, K. Fraser et al., “Xen and the art of virtualization,” *ACM SIGOPS Operating Systems Review*, vol. 37, no. 5, pp. 164–177, 2003.

[9] C. Clark, K. Fraser, S. Hand et al., “Live migration of virtual machines,” in *Proceedings of the NSDI’05 Conference on Symposium on Networked Systems Design & Implementation*, Boston, MA, USA, May 2005.

[10] R. Buyya, S. K. Garg, and R. N. Calheiros, “SLA-oriented resource provisioning for cloud computing: challenges, architecture, and solutions,” in *Proceedings of the International Conference on Cloud & Service Computing*, Hong Kong, China, December 2011.

[11] A. Beloglazov and R. Buyya, “Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in cloud data centers,” *Concurrency and Computation: Practice and Experience*, vol. 24, no. 13, pp. 1397–1420, 2012.

[12] N. Khattar, J. Sidhu, and J. Singh, “Toward energy-efficient cloud computing: a survey of dynamic power management and heuristics-based optimization techniques,” *The Journal of Supercomputing*, vol. 75, no. 8, pp. 1–61, 2019.

[13] X. Zhu, D. Young, B. J. Watson et al., “1000 islands: integrated capacity and workload management for the next generation data center,” in *Proceedings of the 2008 International Conference on Autonomic Computing*, pp. 172–181, IEEE, Chicago, IL, USA, June 2008.

[14] D. Gmach, J. Rolia, L. Cherkasova, G. Belrose, T. Turicchi, and A. Kemper, “An integrated approach to resource pool management: policies, efficiency and quality metrics,” in *Proceedings of the 2008 IEEE International Conference on Dependable Systems and Networks with FTCS and DCC (DSN)*, pp. 326–335, IEEE, Anchorage, AK, USA, June 2008.

[15] A. Beloglazov and R. Buyya, “Energy efficient allocation of virtual machines in cloud data centers,” in *Proceedings of the 2010 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing*, pp. 577-578, IEEE, Melbourne, VIC, Australia, May 2010.

[16] A. Beloglazov and R. Buyya, “Energy efficient resource management in virtualized cloud data centers,” in *Proceedings of the 2010 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing*, pp. 826–831, IEEE Computer Society, Melbourne, VIC, Australia, May 2010.

[17] A. Beloglazov and R. Buyya, “Adaptive threshold-based approach for energy-efficient consolidation of virtual machines in cloud data centers,” in *Proceedings of the 8th International Workshop on Middleware for Grids, Clouds and e-Science—MGC’10*, p. 4, Bangalore, India, November 2010.

[18] E. Arianyan, H. Taheri, and S. Sharifian, “Novel energy and SLA efficient resource management heuristics for consolidation of virtual machines in cloud data centers,” *Computers & Electrical Engineering*, vol. 47, pp. 222–240, 2015.

[19] R. Yadav, W. Zhang, H. Chen, and T. Guo, “Mums: energy-aware VM selection scheme for cloud data center,” in *Proceedings of the 2017 28th International Workshop on Database and Expert Systems Applications (DEXA)*, pp. 132–136, IEEE, Lyon, France, September 2017.

[20] R. Yadav and W. Zhang, “MeReg: managing energy-SLA tradeoff for green mobile cloud computing,” *Wireless Communications and Mobile Computing*, vol. 2017, Article ID 6741972, 11 pages, 2017.

[21] R. Yadav, W. Zhang, K. Li, C. Liu, M. Shafiq, and N. K. Karn, “An adaptive heuristic for managing energy consumption and overloaded hosts in a cloud data center,” *Wireless Networks*, vol. 26, pp. 1–15, 2018.

[22] R. Yadav, W. Zhang, O. Kaiwartya, P. R. Singh, I. A. Elgendy, and Y. C. Tian, “Adaptive energy-aware algorithms for minimizing energy consumption and SLA violation in cloud computing,” *IEEE ACCESS*, vol. 6, pp. 55923–55936, 2018.

[23] S. B. Melhem, A. Agarwal, N. Goel, and M. Zaman, “Markov prediction model for host load detection and VM placement in live migration,” *IEEE ACCESS*, vol. 6, pp. 7190–7205, 2018.

[24] M. Ranjbari and J. A. Torkestani, “A learning automata-based algorithm for energy and SLA efficient consolidation of virtual machines in cloud data centers,” *Journal of Parallel and Distributed Computing*, vol. 113, pp. 55–62, 2018.

[25] S. K. Abd, S. A. R. Al-Haddad, F. Hashim, A. B. H. J. Abdullah, and S. Yussof, "An effective approach for managing power consumption in cloud computing infrastructure," *Journal of Computational Science*, vol. 21, pp. 349–360, 2017.

[26] H. Duan, C. Chao, G. Min, and W. Yu, "Energy-aware scheduling of virtual machines in heterogeneous cloud computing systems," *Future Generation Computer Systems*, vol. 74, pp. 142–150, 2016.

[27] H. Li, G. Zhu, C. Cui, H. Tang, Y. Dou, and C. He, "Energy-efficient migration and consolidation algorithm of virtual machines in data centers for cloud computing," *Computing*, vol. 98, no. 3, pp. 303–317, 2016.

[28] M. Ghobaei-Arani, A. A. Rahmanian, M. Shamsi, and A. Rasouli-Kenari, "A learning-based approach for virtual machine placement in cloud data centers," *International Journal of Communication Systems*, vol. 31, no. 8, p. e3537, 2018.

[29] Z. Zhou, J. Abawajy, M. Chowdhury et al., "Minimizing SLA violation and power consumption in Cloud data centers using adaptive energy-aware algorithms," *Future Generation Computer Systems*, vol. 86, pp. 836–850, 2018.

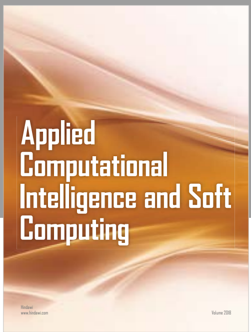
[30] A. Beloglazov, J. Abawajy, and R. Buyya, "Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing," *Future Generation Computer Systems*, vol. 28, no. 5, pp. 755–768, 2012.

[31] R. N. Calheiros, R. Ranjan, A. Beloglazov, C. A. F. De Rose, and R. Buyya, "CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms," *Software: Practice and Experience*, vol. 41, no. 1, pp. 23–50, 2011.

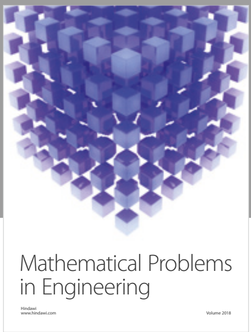
[32] R. Buyya, R. Ranjan, and R. N. Calheiros, "Modeling and simulation of scalable cloud computing environments and the CloudSim toolkit: challenges and opportunities," in *Proceedings of the 2009 International Conference on High Performance Computing & Simulation*, pp. 1–11, IEEE, Leipzig, Germany, June 2009.

[33] K. Park and V. S. Pai, "CoMon," *ACM SIGOPS Operating Systems Review*, vol. 40, no. 1, pp. 65–74, 2006.

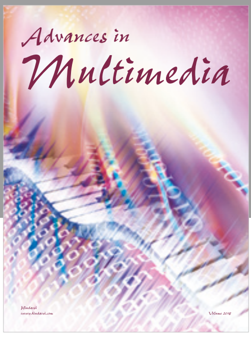
[34] C. L. Dumitrescu and I. Foster, "GangSim: a simulator for grid scheduling studies," in *Proceedings of the CCGrid 2005. IEEE International Symposium on Cluster Computing and the Grid*, pp. 1151–1158, IEEE, Wales, UK, May 2005.

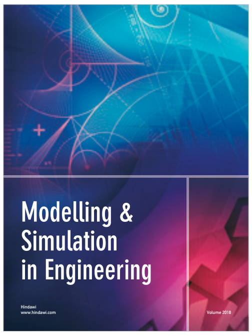

Applied Computational Intelligence and Soft Computing

The Scientific World Journal


Mathematical Problems in Engineering

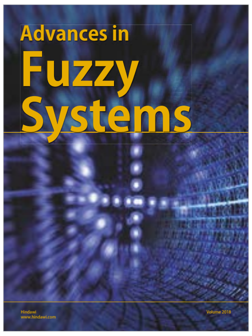
Journal of Engineering

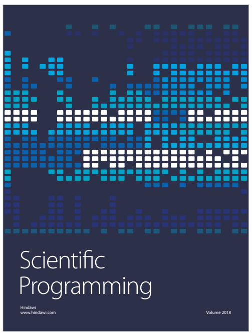

Advances in Multimedia

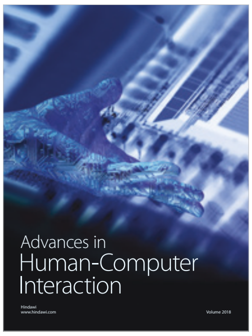

Modelling & Simulation in Engineering

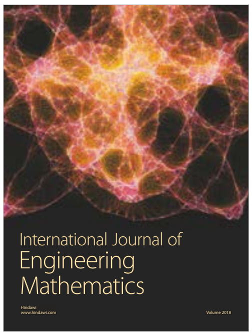
Advances in Artificial Intelligence

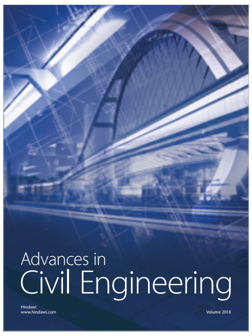
International Journal of Reconfigurable Computing

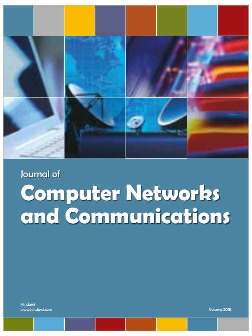

Advances in Fuzzy Systems


Scientific Programming


Advances in Human-Computer Interaction

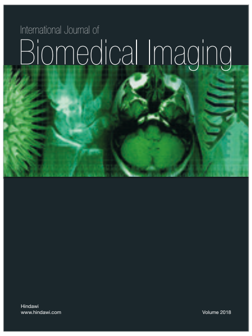

International Journal of Engineering Mathematics


Advances in Civil Engineering

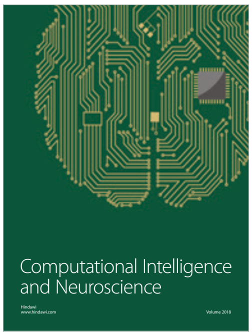

Journal of Computer Networks and Communications

Journal of Robotics

International Journal of Computer Games Technology


International Journal of Biomedical Imaging

Journal of Electrical and Computer Engineering


Computational Intelligence and Neuroscience