IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received December 10, 2019, accepted December 20, 2019, date of publication December 23, 2019, date of current version January 2, 2020.

*Digital Object Identifier 10.1109/ACCESS.2019.2961786*

# A Virtual Machine Consolidation Algorithm Based on Ant Colony System and Extreme Learning Machine for Cloud Data Center

**FAGUI LIU, ZHENJIANG MA, BIN WANG, AND WEIWEI LIN**
School of Computer Science and Engineering, South China University of Technology, Guangzhou 510006, China

Corresponding authors: Zhenjiang Ma (cszhenjiangma@mail.scut.edu.cn) and Weiwei Lin (linww@scut.edu.cn)

This work was supported in part by the Engineering and Technology Research Center of Guangdong Province for Logistics Supply Chain and Internet of Things under Project GDDST[2016]176, in part by the Industrial Development Fund Project of Guangzhou under Project X2JSD8183470, in part by the National Natural Science Foundation of China under Grant 61772205 and Grant 61872084, and in part by the Major Program of Guangdong Basic and Applied Research under Project 2019B030302002.

**ABSTRACT** The energy consumption issue of large-scale data centers is attracting more and more attention. Virtual machine consolidation can significantly reduce energy consumption by migrating virtual machines from one physical machine to another. However, excessive virtual machine consolidation can lead to dangerous Service Level Agreement (SLA) violations. Therefore, how to balance between effective energy consumption and SLA violations avoidance effectively is a paradox to be mediated. The virtual machine consolidation problem is NP-hard. The traditional heuristic algorithm is easy to fall into the local optimal and some meta-heuristic algorithms can help to avoid it. However, the existing meta-heuristic algorithms are with high complexity. Therefore, we propose a lower complexity multi-population ant colony system algorithm with the Extreme Learning Machine (ELM) prediction (ELM_MPACS). The algorithm firstly predicts the host state employing ELM and then the virtual machine on the overloaded host will be migrated to the normal host, while the virtual machine on the underloaded host will be consolidated to another underloaded host with higher utilization. Multiple populations concurrently construct migration plans and local search further optimizes the results obtained by each population to reduce SLA violations. We compare ELM_MPACS with the benchmark, heuristic and meta-heuristic algorithms. The experimental results have shown that compared with these algorithms, our algorithm reduces energy consumption, migration times and SLA violations effectively.

**INDEX TERMS** Cloud computing, ant colony system, virtual machine consolidation, energy consumption, SLA.

## I. INTRODUCTION

Cloud computing is a prevailing computing paradigm, which provides computer hardware and software resources as a service to users over the Internet [1]. Generally, cloud computing can be divided into Infrastructure as a Service (IaaS), Platform as a Service (PaaS) and Software as a Service (SaaS) according to the type of service [2]. Cloud computing technology can optimize computing resources and provide flexible extensions. Any user can buy large amounts of computing resources to complete complex business demands at a low cost. However, the number and size of global data centers continue to increase to meet the growing business demands, so the energy consumption also increases dramatically [3]. For example, the average power consumption of a data center is equivalent to the power consumption of 25,000 households [4]. The issue of energy consumption has received widespread attention. It not only causes high operating costs but also leads to enormous damage to the ecological environment. Besides, cloud service providers should also ensure that users can enjoy reliable Quality of Service (QoS). The cloud service provider will be punished for violating the Service Level Agreements (SLA) specifying QoS objectives [5]. Therefore, how to reduce energy consumption while ensuring fewer SLA violations has become a big challenge for cloud service providers.

Virtualization technology is one of the critical technologies of cloud computing, which enables multiple virtual machines to share one physical machine [6]. If all the virtual machines are packed into a few physical machines, the energy

The associate editor coordinating the review of this manuscript and approving it for publication was Songwen Pei.

consumption in the data center will be significantly reduced [7]. Over the past few years, researchers have focused on Virtual Machine Placement (VMP) problem to reduce energy consumption by optimizing the placement. VMP is intended to ensure QoS and decrease energy consumption by building a reasonable mapping between multiple virtual machines and physical machines. The problem of VMP can be divided into static placement and dynamic placement [8]. Static placement refers to creating virtual machines on suitable physical machines and dynamic placement is a process of migrating the running virtual machine from one physical machine to another [9]. We focus on the dynamic virtual machine placement, which is also called dynamic virtual machine consolidation, to reduce energy consumption. However, placing an excessive number of virtual machines on the same physical machine will result in substantial SLA violations [10] and poor user experience. Therefore, we should pay more attention to the QoS requirements of users in the process of placing energy-aware virtual machines.

The two main concerns of virtual machine consolidation are determining the source and destination hosts of the migration and how to migrate virtual machines from source host to destination host. Some existing studies typically rely on static thresholds or dynamic thresholds to determine the host state by comparing current utilization with thresholds. Other studies use prediction techniques such as linear regression [11] and machine learning [12] to predict the host state in the next scheduling cycle to determine which hosts need to be migrated. Because Extreme Learning Machine (ELM) is accurate, fast, and generalized [13], we use ELM to predict host usage. The VMP problem is NP-hard [14], [15]. Some researchers put forward linear programming to solve virtual machine consolidation [16], [17]. The linear programming method is simple and can obtain accurate optimal solution, but poor in scalability. When the problem scale increases, the calculation time will increase greatly. Therefore, linear programming is not suitable to deal with NP-hard problems. Some researchers proposed heuristic algorithms to get approximate optimal solutions [18]–[20], but the heuristic algorithm may fall into the local optimal easily. Bio-inspired meta-heuristic algorithms, such as Ant Colony System (ACS), Genetic Algorithm (GA) and Particle Swarm Optimization (PSO), can not only avoid falling into a locally optimal solution but also get high-quality approximation in a reasonable time in dealing with NP-hard problems [21], [22]. However, some meta-heuristic algorithms such as GA and PSO are designed for continuous problems originally, and they require special encoding for combinatorial optimization problems. Therefore, we choose ACS which is designed for discrete problems. What's more, the ACS algorithm is a kind of swarm intelligence algorithm, which can be parallelized to speed the process.

To the best of our knowledge, some researchers used ACS [21], [23] to solve the virtual machine consolidation problem. However, the execution time is too long due to the large search space. Therefore, it is necessary to optimize the search space further. As a result, we propose an algorithm based on ELM and ACS called ELM_MPACS in this paper. First, the ELM algorithm is employed to predict underloaded and overloaded hosts. Then the multi-population ant colony system is applied to consolidate virtual machines. Finally, local search strategy and pheromone exchange rule are adopted to optimize the solution. The main contributions of this paper are as follows:

1. We present an ELM-based prediction algorithm. First, multiple ELM models are trained to reduce the error caused by random initialization. Then we choose the one with the least verification error for prediction.

2. We propose a virtual machine consolidation algorithm with lower complexity based on ACS. We dynamically choose the destination host for the virtual machine to reduce the complexity, avoiding the overhead of the pre-constructed migration tuples. Multiple populations construct concurrently and their respective migration schemes are optimized by the local search strategy. The pheromone exchange rule between different populations is used to increase the pheromones of the excellent combinations.

3. Different from the current work dealing with the virtual machines indiscriminately, we propose a new virtual machine migration strategy. The virtual machines on overloaded hosts are migrated to normal hosts, while the virtual machines on underloaded hosts are migrated to other underloaded hosts with higher utilization.

4. We conduct experiments on the real dataset of the CloudSim platform and compare ELM_MPACS with other algorithms. The experimental results have shown that our proposed algorithm effectively reduces energy consumption, the number of migrations and SLA violations.

The remainder of the paper is organized as follows. Section 2 reviews and discusses the relevant researches about energy consumption in the data center. We detail the power model, objective function and ELM prediction algorithm in Section 3. In Section 4, we present our ELM_MPACS algorithm. We describe experimental settings and analyze our experimental results in Section 5. The conclusions are in Section 6.

## II. RELATED WORK

The primary issue we should address about virtual machine consolidation is to decide the source and destination hosts of the migration. The host load changes dynamically in the data center. Lower usage can result in a waste of resources, while higher usage is prone to lead to SLA violations. Therefore, determining the source hosts for migration in the data center is the first step to solve the problem. Double static thresholds are adopted in [2]. The algorithm divided the hosts into three types, including the overloaded hosts, underloaded hosts and normal hosts. If the utilization of the host exceeded the upper threshold, it was overloaded. Below the lower threshold, it was underloaded. However, the method based on static threshold cannot adapt well to the dynamic changes of load in the data center. Beloglazov and Buyya [20] proposed the method of dynamic threshold based on historical utilization to detect overloaded hosts and finally migrated virtual machines on overloaded hosts to other hosts to implement virtual machine consolidation. Yadav and Zhang [24] proposed an algorithm that adjusted the upper threshold dynamically. The algorithm was employed to minimize the value of residual and not directly influenced by the outlier. Yadav *et al.* also [25] proposed two adaptive methods which were based on robust regression to set dynamic upper threshold. The first algorithm used gradient descent to minimize the cost function to get the global optimal, and the second algorithm was based on the idea that the host with the maximum correlation coeffient between the virtual machines was more likely to be overloaded. Zhou *et al.* [26] proposed an adaptive three-threshold method, which used the K-Means clustering algorithm to further subdivide the hosts into four types, which can better adapt to the dynamic changes of host load in the data center.

However, these papers only determine the overloaded host and the underloaded host by comparing the current utilization with the thresholds and do not predict the host state in the next period. If we can shut down the underloaded host in advance or migrate the virtual machines on the overloaded host beforehand, we will reduce energy waste and avoid SLA violations in the data center. Accurate prediction algorithms can prevent some unnecessary virtual machine migrations and the existing prediction algorithms proposed by researchers are mainly divided into two categories. One is based on linear regression [11], [27] while the other is based on machine learning like K-NNR [12] and ANN [28]. Linear regression methods can only capture linear features and machin learning methods such as neural networks can build nonlinear models flexibly but consume much more time [29], [30]. Compared with the traditional neural network, the ELM [13], [31] randomly initializes the weight and bias between the input and the hidden nodes, so the execution speed is fast. Besides, the generalization ability of ELM is also outstanding. There are a large number of physical machines in the data center that need to be scheduled periodically, so an accurate and efficient prediction algorithm is required. In addition, the load of the host in the data center changes dynamically, and migrations also cause load changes on the source and destination hosts. Linear prediction algorithms cannot predict this trend well, so we use an algorithm based on ELM to predict the host state in the next period.

Another critical issue that we should address is to determine the migration method. Researchers have proposed many approaches to migrating virtual machines from source hosts to appropriate destination hosts. Because the virtual machine consolidation problem is NP-hard, we divide these methods into two categories. One is non-meta-heuristic algorithms including linear programming and heuristic algorithm, and the other is meta-heuristic algorithms.

Linear programming or integer programming [16], [17] is one of the earliest proposed approaches to solving the VMP problem. This kind of algorithm is simple and can get the optimal solution, but it is challenging to get the optimal solution in a reasonable time when the scale of the problem increases. First Fit (FF) and Best Fit (BF) are well-known heuristic algorithms. Anand *et al.* [18] compared Integer Linear Programming (ILP) and First Fit Decreasing (FFD) algorithms considering the energy consumption caused by virtualization and migration. Murtazaev and Oh [19] proposed the SERCON method based on FF and BF, which considered minimizing the number of active hosts and migrations. However, most of the papers aforementioned only considered CPU or memory and rarely took into account the factor of the bandwidth. Lago *et al.* [32] noticed the impact of network bandwidth on performance and considered bandwidth resources in the heterogeneous network during the process of scheduling virtual machines. It shortened migration time by allocating bandwidth rationally and reduced energy consumption ultimately. Zhu *et al.* [33] considered three resources including CPU, memory and bandwidth. Besides, the authors designed different algorithms for virtual machine allocation, scheduling and optimization to minimize energy consumption. These two articles considered bandwidth resources and established more accurate and realistic models which reduced SLA violations and energy consumption in the data center. The traditional heuristic algorithms are easy to fall into the local optimal when dealing with the NP-hard problem. However, meta-heuristic algorithms have significant advantages in solving such issues; hence, many meta-heuristic algorithms have been applied to virtual machine consolidation.

Li *et al.* [34] considered the upper and lower thresholds of the CPU and hard disk resources. The authors applied the improved PSO to the problem of virtual machine consolidation to avoid falling into the local optimal. However, the above paper did not employ prediction methods and might cause unnecessary migrations. Chou *et al.* [35] put forward a resource allocation strategy based on PSO as well and employed the least-square method to predict resource utilization in the next period. GA was also applied to reduce energy consumption in the data center [36], [37]. Unlike paper [36], which used intelligent algorithms for resource allocation, [37] employed GA as a prediction algorithm to predict the state of the physical machine in the next period. A new meta-heuristic algorithm named salp swarm optimization was introduced in [38], which imitated the behavior of salp swarm. However, the author used real number encoding. When it converted real number into integer number, it could lead to a loss of accuracy. Li *et al.* [39] proposed an virtual machine consolidation method based on discrete Differential Evolution (DE). However, the author only considered the energy consumption and host overloading risk but ignored the number of migrations, which is of importance in the real data center.

However, some algorithms such as GA, PSO and DE are originally designed to solve continuous optimization problems, so the discretization is required for solving combinatorial optimization problems, which may result in a loss of accuracy. Ant colony optimization, a meta-heuristic algorithm, is designed for discrete optimization problems without special encoding [40], and it has also been applied to virtual machine consolidation problem. Farahnakian *et al.* [23] proposed an ACS-based virtual machine consolidation approach, aiming at maximizing the number of dormant physical machines and minimizing the number of virtual machine migrations. The ant traversed all possible tuples which are constructed as (source host, virtual machine, destination host) to yield an approximate optimal solution. Aryania *et al.* [21] extended [23]. The authors took into account the number of dormant physical machines and viewed the size of memory during VM migration as an essential factor. Ashraf and Porres [41] also improved [23]. It assigned different priorities for maximizing the number of dormant hosts and minimizing the number of migrations. These two independent populations were intended to optimize different goals. Moreover, the authors added neighborhood constraints, which asked the migration to occur in the neighborhood, to reduce the search space.

However, although the scope of the source and the destination hosts is narrowed by adding constraints in the paper [21], [23], it is still considerable. Since the ant will traverse all tuples predefined to find the destination host for each virtual machine, substantial tuples will be constructed when there are abundant underloaded hosts or overloaded hosts in the data center, which will significantly increase the search space. Moreover, the complexity of traversing tuples is higher than that of traversing only the destination hosts. In [41], the neighborhood constraints can narrow the search space, but it may make it unable to get the global optimal. To narrow the search space, reduce the complexity and get global optimal, we design a new ACS algorithm. The ant dynamically traverses the possible destination hosts for each virtual machine, instead of traversing all the tuples previously constructed or adding neighborhood constraints. Thus our proposed method narrows the search space, reduces the complexity and obtains the global optimal.

## III. PROBLEM FORMULATION

We need to do some preparations before consolidating virtual machine. A power model of the physical machine is established at first, which is the basis for evaluating the power consumption in the data center. Then we need to determine the objective function of the virtual machine consolidation problem, and we formulate the virtual machine consolidation problem into a multi-objective function, including migration times and energy consumption. Finally, we find out the overloaded hosts and the underloaded hosts to migrate the virtual machines. Therefore, a prediction module that classifies the hosts is also necessary. We will detail these three sub-modules in this section.

### A. POWER MODEL

The energy consumption components of a physical machine is composed of CPU, memory, hard disk and network communication components, among which CPU is the main energy-consuming component [2], [23], [42]. Therefore, we can use the energy consumption of the CPU to estimate the energy consumption of the whole system. In order to calculate the current power consumption more accurately, it is necessary to build a suitable power model. Because we use the CPU utilization to estimate the power consumption of the system, we need to study the power consumption relationship between CPU and physical machine. The existing papers [43], [44] indicate that there is a robust linear relationship between CPU utilization and the power consumption of the physical machine. Therefore, we establish the following linear power model.

$$P_j = P_j^{min} + u * \left(P_j^{max} - P_j^{min}\right), \tag{1}$$

which is equivalent to

$$P_j = P_j^{max} * u + P_j^{max} * k * (1 - u), \tag{2}$$

where $P_j$ is the power of the host, $P_j^{min}$ is the idle power of the host and $P_j^{max}$ is the peak power of the host. $u$ represents the current utilization of the host and $k$ is the ratio of the idle power and the peak power.

From equation (1), we can conclude that when the CPU utilization is 0, the idle host still consumes lots of power. Moreover, the paper [2] pointed out that idle power accounts for 70% of peak power. Therefore, if the idle physical machine can be shut down in time, the energy consumption will be significantly saved.

### B. OBJECTIVE FUNCTION

It is one of our primary motivations to reduce energy consumption for virtual machine consolidation, while the number of migrations is another factor that cannot be ignored. The performance of the physical machine will degrade due to excessive migrations, thereby affecting the user experience. If the virtual machines on the underloaded physical machines cannot be migrated in time, the hosts running at a low load mode will waste resources and increase energy consumption. If the virtual machines on the overloaded physical machines cannot be removed in time, continuous SLA violations will occur. Therefore, migration times have a significant impact on energy consumption and SLA violations. In this paper, we establish a multi-objective function considering both energy consumption and the number of migrations. Whenever constructing a migration plan, we calculate the ratio of the current power of the host to the maximum power of the host and then obtain the cumulative sum. So the final objective function is as follows:

$$f = \max\left(\frac{1}{Mig} + \frac{1}{\sum_{j=1}^{M} \frac{P_j}{P_j^{max}}}\right), \tag{3}$$

where $Mig$ is the number of migrations for the migration plan, $M$ is the number of physical machines. $P_j$ is the power of the physical machine after migrating, $P_j^{max}$ is the peak power of the host. $f$ is inversely proportional to $Mig$ and power consumption. The larger the number of migrations, the smaller the value of $f$. The higher the power consumption, the smaller the value of $f$. The unit of power consumption is in watts. To avoid the impact of dimension on calculations, we normalize the power consumption. The above formula is equivalent to

$$f = \max\left(\frac{1}{Mig} + \frac{1}{\sum_{j=1}^{M} k(1-u)+u}\right), \tag{4}$$

where $u$ is the current utilization of the host, and $k$ is the ratio of the idle power and peak power. The above function is the fitness function of the ant colony system algorithm, which is used to evaluate the schemes of the ants.

### C. ELM PREDICTION ALGORITHM

ELM is a single hidden layer feed-forward neural network. Different from the traditional neural networks, ELM is characterized by its random initialization of the bias and weight between the input layer and the hidden layer [13]. ELM is fast to train and more generalized. There are a large number of physical machines need to be periodicly scheduled in the data center, so an accurate prediction algorithm is necessary, and training speed is also required to be fast enough. Herein, we employ ELM to predict the utilization of the host in the next scheduling period. Firstly, several ELM models are trained with one part of the historical utilization, and then another part of the data is used to validate the models. The ELM model, with the least validation error, is used to predict the usage of the host in the next period. In this paper, the number of output neurons is set to one, instead of multiple output neurons for a classification problem.

We take an ELM with K hidden layer nodes as an example. Suppose there are $m$ samples with $n+1$ dimensions, the last dimension is used as the label, and the $i$th sample is recorded as $(\mathbf{x}_i, y_i)$, where $\mathbf{x}_i = [x_{i1}, x_{i2}, \ldots, x_{in}]^{\mathrm{T}}, i = 1, 2, \ldots, m$. A single hidden layer neural network with K hidden layer nodes can be expressed as

$$\sum_{j=1}^{K} \beta_j g\left(\mathbf{w}_j \cdot \mathbf{x}_i + b_j\right) = z_i, \quad i = 1, 2, \ldots, m, \tag{5}$$

where $g(\mathbf{x})$ is the activation function, and $\mathbf{w}_j = \left[w_{j1}, w_{j2}, \ldots, w_{jn}\right]$ is the weight of the $j$th hidden layer node relative to the input nodes. $\beta_j$ is the weight of the $j$th hidden layer node for the output nodes. $b_j$ is the bias of the $j$th hidden layer node and $z_i$ is the actual output. To minimize the training error, the following formula is established.

$$\sum_{i=1}^{m} \|z_i - y_i\| = 0, \tag{6}$$

which is equivalent to

$$\sum_{j=1}^{K} g\left(\mathbf{w}_j \cdot \mathbf{x}_i + b_j\right) \beta_j = y_i, \quad i = 1, 2, \ldots, m. \tag{7}$$

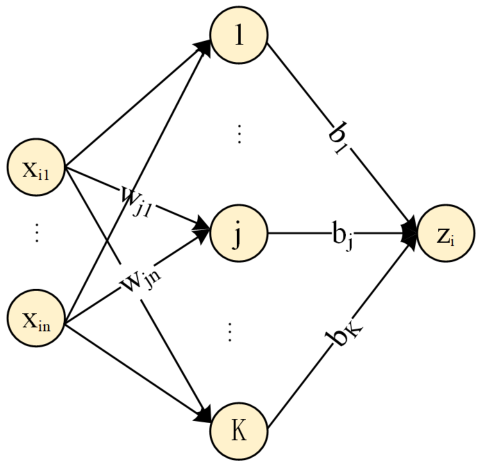


**FIGURE 1.** The model of the ELM.

written in matrix form

$$\mathbf{A}\boldsymbol{\beta} = \mathbf{Y}, \tag{8}$$

where

$$A = \begin{bmatrix} g(\mathbf{w_1} \cdot \mathbf{x_1} + b_1) & \cdots & g(\mathbf{w_K} \cdot \mathbf{x_1} + b_K) \\ & \vdots & \ddots \\ g(\mathbf{w_1} \cdot \mathbf{x_n} + b_1) & \cdots & g(\mathbf{w_K} \cdot \mathbf{x_n} + b_K) \end{bmatrix}, \tag{9}$$

$$\boldsymbol{\beta} = \begin{bmatrix} \beta_1 \\ \vdots \\ \beta_K \end{bmatrix}, \quad \mathbf{Y} = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix}. \tag{10}$$

The weight and bias between the input layer and the hidden layer are obtained by random initialization in ELM. Therefore, if the input data is determined, only the weight vectors of the hidden layer to the output layer are unknown. So we can get the weight vectors by solving the above equation. The inverse matrix of the singular and non-square form does not exist, but we can obtain a pseudo-inverse matrix of them. The pseudo-inverse matrix $A^{\dagger}$ of A can be computed like in [31], then we can calculate the value of $\boldsymbol{\beta}$.

$$\boldsymbol{\beta} = \mathbf{A}^{\dagger}\mathbf{Y}, \tag{11}$$

where $\mathbf{A}^{\dagger}$ is the pseudo-inverse matrix of A. We input the data to be predicted after figuring out $\boldsymbol{\beta}$, and then we can get predicted value $\widehat{y}$ according to (7).

According to extensive experiments, we take three ELMs in this paper. We set the number of hidden layer nodes to five and use sine function as activation function. Moreover, we take ten samples with three dimensions, of which 7/10 are used for training, 3/10 are used for verification. The network of the ELM is shown in Fig. 1.

We use ELM to detect overloaded and underloaded hosts and the detection algorithm is shown in Algorithm 1. The paper [23] set the threshold to 0.5 and 1.0, and the paper [11] set the threshold to 0.1 and 0.9. A large upper threshold will lead to frequent overload and severe SLA violations, while a small upper threshold will result in a waste of resources. If the lower threshold is too large, a large number of virtual

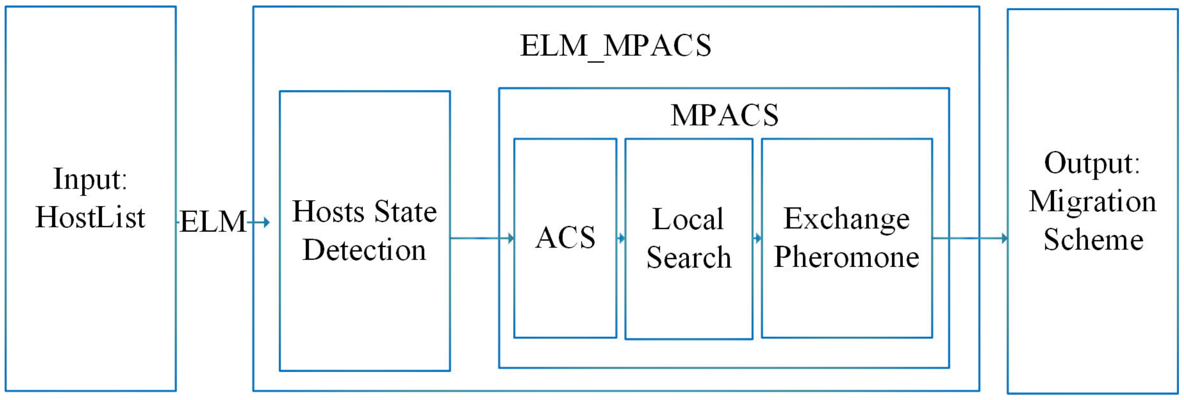


**FIGURE 2.** The relationship diagram of different parts of ELM_MPACS.

**Algorithm 1** Hosts State Dectection

```
Input: hostList
Output: underloadedHostList,
    normalHostList, overloadedHostList
1:  for host in hostList do
2:      if utilizationHistory.size $\leq 10$ then
3:          utilization = host.getUsedMips/host.getTotalMips
4:      else
5:          utilization = ELM(host)
6:      end if
7:      if utilization $> 0$ and utilization $\leq 0.3$ then
8:          underloadedHostList.add(host)
9:      else if utilization $> 0.3$ and utilization $\leq 0.8$ then
10:         normalHostList.add(host)
11:     else if utilization $> 0.8$ then
12:         overloadedHostList.add(host)
13:     end if
14: end for
```

machines will be migrated. If the lower threshold is too small, a large number of hosts with low usage will not be shut down in time. In order to obtain reasonable thresholds, we have experimentally verified that the upper and lower thresholds are set to 0.8 and 0.3, respectively. We can make a tradeoff between energy consumption and SLA violations in this interval. We execute the scheduling algorithm every five minutes. During the first ten periods, the current utilization of the host is used to determine the host state (line 1-3). When we collect enough data, we use ELM to predict host utilization (line 5). If the host utilization is greater than 0 and does not exceed 0.3, the host is considered to be underloaded (line 7-8). If the host utilization is greater than 0.3 and does not exceed 0.8, the host is the normal host (line 9-10). If the utilization of the host exceeds 0.8, it is considered to be overloaded (line 11-12).

## IV. MULTI-POPULATION ACS BASED ON ELM (ELM_MPACS)

Based on the preliminary work proposed in Section 3, we propose a multi-population ant colony system algorithm based on ELM (ELM_MPACS). The general idea of our ELM_MPACS algorithm is: firstly, we determine the underloaded hosts, normal hosts and overloaded hosts according to the ELM prediction model, and then apply the multi-population ACS to assign the destination host for the virtual machine to be migrated. According to the objective function (4), we evaluate each population's scheme and finally get the best solution. In this section, we first introduce the various parts of the virtual machine consolidation algorithm we proposed, and then show the complete ELM_MPACS algorithm and analyze its complexity finally. Fig. 2 shows the relationship between these parts.

### A. DEFINITION OF PHEROMONE AND HEURISTIC INFORMATION

In the real world, the pheromone is a kind of chemical substance that ants communicate with each other by it and ants find the source of food along the way by sensing other ants' pheromones [45]. The pheromone on the combination of a virtual machine and a physical machine in virtual machine consolidation represents the ant's favorability. Ants are more likely to choose the combination with a higher value of pheromone. The more pheromone of the combination accumulates, the more it has been selected in the previous iterations, meaning that it is a right choice. $\tau_{i,j}$ represents the value of pheromone of the combination $(VM_i, PM_j)$, where $VM_i$ is the $i$th virtual machine, and $PM_j$ is the $j$th physical machine. The initial pheromone is also very important, and we set the initial pheromone $\tau_0$ to

$$\tau_0 = \frac{1}{N}, \tag{12}$$

where $N$ is the number of virtual machines to be migrated.

In addition to the pheromone, heuristic information is another essential factor in ACS. The heuristic information $\eta_{i,j}$ indicates the expectancy that $VM_i$ is assigned to $PM_j$. The expectations of the same destination host for different virtual machines are different. In this paper, the virtual machine on the overloaded host is assigned to the normal host, and the virtual machine on the underloaded host is migrated to another underloaded host with higher utilization. On the one hand, to avoid overloading of destination hosts after the migration, the virtual machine on the overloaded host is biased toward the normal host with more resources. On the other hand, to shut down more underloaded hosts, the underloaded hosts with higher usage are better choices for VMs from underloaded hosts with lower usage. Therefore, we design different heuristics for virtual machines from various sources.

The heuristic for the overloaded hosts is defined as

$$\eta_{i,j} = \begin{cases} 1 - \dfrac{PU_j + VR_i}{PC_j}, & \text{if } PU_j + VR_i \leq PC_j \\ 0, & \text{otherwise}, \end{cases} \tag{13}$$

while that for the underloaded hosts is formulated as

$$\eta_{i,j} = \begin{cases} \dfrac{PU_j + VR_i}{PC_j}, & \text{if } PU_j + VR_i \leq PC_j \\ 0, & \text{otherwise}, \end{cases} \tag{14}$$

where $PU_j$ is the CPU resource used of $PM_j$ and $VR_i$ is the resource requested by $VM_i$. Constraint $PU_j + VR_i \leq PC_j$ is added to prevent the CPU resource requested from exceeding the total capacity of the host. Heuristic information is updated before selecting a destination host.

In the above formulas, $\frac{PU_j+VR_i}{PC_j}$ indicates the utilization of $PM_j$ after migration, and $1 - \frac{PU_j+VR_i}{PC_j}$ indicates the available resource ratio after migration. In the equation (13), the value of $\eta_{i,j}$ is proportional to available resource ratio; in the equation (14), higher utilization means higher $\eta_{i,j}$.

### B. PSEUDO-RANDOM-PROPORTIONAL RULE AND PHEROMONE UPDATING RULES

The ant prefers to choose the combinations with the largest product of the pheromone and the heuristic information. However, to avoid falling into local optimum, ants choose combinations based on pseudo-random-proportional rule. The pseudo-random-proportional rule can be expressed as follows

$$\text{r} = \begin{cases} \arg\max \left\{ \left[\tau_{i,j}^{\alpha}\right] \cdot \left[\eta_{i,j}^{\beta}\right] \right\}, & \text{if } q \leq q_0 \\ \text{R}, & \text{otherwise}, \end{cases} \tag{15}$$

where $q_0$ is a fixed value between 0 and 1, $\alpha$ and $\beta$ are weighting factors that determine the importance of two factors. $\tau_{i,j}$ and $\eta_{i,j}$ are the pheromone and heruistic information separately for migrating $VM_i$ to $PM_j$, and $\tau_{i,j}^{\alpha} \cdot \eta_{i,j}^{\beta}$ indicates that $\tau_{i,j}^{\alpha}$ and $\eta_{i,j}^{\beta}$ jointly determine the migration plan. If $q \leq q_0$, we select the host with the max product as the destination host. Otherwise, we choose the destination host according to the roulette wheel rule. The formula for calculating the probability of roulette is

$$p_{i,j} = \frac{\left[\tau_{i,j}^{\alpha}\right] \cdot \left[\eta_{i,j}^{\beta}\right]}{\sum_{PM_j \in \Theta} \left[\tau_{i,j}^{\alpha}\right] \cdot \left[\eta_{i,j}^{\beta}\right]}, \tag{16}$$

where $p_{i,j}$ represents the probability that the virtual machine $VM_i$ selects the destination $PM_j$. $\Theta$ is a set of destination physical machines. If $PM_j$ belongs to $\Theta$, the probability will be calculated. The probability is proportional to the value of $\tau_{i,j}^{\alpha} \cdot \eta_{i,j}^{\beta}$. If $q \leq q_0$, it is called exploitation, otherwise named exploration. Exploration can find more new options and avoid rapid convergence.

We select the destination host for each virtual machine according to the pseudo-random-proportional rule. During the process of migrating virtual machines on underloaded hosts, if $q \leq q_0$, the combination with the largest product of the heuristic information and the pheromone will be selected. The heuristic information is determined by the usage of resources, so the underloaded host with the highest utilization is preferred. In addition, if $q > q_0$, the combination with the largest product is also the most likely to be selected. Therefore, the migration of virtual machines from under-loaded hosts with lower utilization to under-loaded hosts with higher utilization will not cause excessive migrations.

After selecting the destination host according to the pseudo-random-proportion rule for the current virtual machine, the pheromone of the combination $(VM_i, PM_j)$ can be updated. Local pheromone updating is used to evaporate part of pheromone to avoid other ants selecting the same combination so that ants explore potential combinations. The rule of local pheromone updating is defined as follows

$$\tau_{i,j} = (1 - \rho) \cdot \tau_{i,j} + \rho \cdot \tau_0, \tag{17}$$

where $\rho$ is the pheromone evaporation coefficient, which is a number between 0 and 1 and determines the degree of pheromone evaporation. The larger $\rho$ is, the less the pheromone remains, and $\tau_0$ is the initial pheromone.

After all the ants in the population have built their plans, we will select the best one from each population according to (4). Then the fitness of the schemes of all the populations are compared, and the final scheme $S^+$ is selected to update global pheromone. The global pheromone updating rule can be defined as

$$\tau_{i,j} = (1 - \rho) \cdot \tau_{i,j} + \rho \cdot f(S^+), \tag{18}$$

where $S^+$ is the final scheme obtained above, $\rho$ is the pheromone evaporation coefficient. The global pheromone is to increase the pheromone concentration of each combination in the final scheme $S^+$, which is intended to reward these combinations and increase their probability of being selected in subsequent iterations.

### C. LOCAL SEARCH STRATEGY AND PHEROMONE EXCHANGE RULE

Although each population has already built its solution, there is no guarantee that every active host in the solution will be load balanced. If a heavily loaded host exchanges some virtual machines with a low-loaded host, then overload can be avoided. So we propose a local search strategy to achieve load balancing and reduce SLA violations. The basic idea of the local pheromone exchange rule is that after each population has built its migration plan, some exchange operations are performed within the respective schemes. The process

**Algorithm 2** LocalSearch

```
Input: $S$
Output: $S_l$
1:  for (i = 1; i < N*N; i++) do
2:      randomly choose ($VM_1$, $PM_1$),($VM_2$, $PM_2$)
3:      compute utilization difference1 between hosts
4:      exchange the destination hosts
5:      compute utilization difference2 between hosts
6:      if difference2 < difference1 then
7:          return ($VM_1$, $PM_2$), ($VM_2$, $PM_1$)
8:      else
9:          return ($VM_1$, $PM_1$), ($VM_2$, $PM_2$)
10:     end if
11: end for
```

**Algorithm 3** Exchange Pheromone

```
Input: $S_1$, $S_2$
Output: $S_1$, $S_2$
1: for combination1 in $S_1$ do
2:     for combination2 in $S_2$ do
3:         if combination1 == combination2 then
4:             temp = combination1.pheromone + combination2.pheromone;
5:             combination1.pheromone = temp;
6:             combination2.pheromone = temp;
7:         end if
8:     end for
9: end for
```

of local search is shown in Algorithm 2. If the termination iteration condition is not reached, each time we randomly select two combinations ($VM_1$, $PM_1$), ($VM_2$, $PM_2$) from the migration scheme $S$ (line 1-2). To make each virtual machine has the opportunity to exchange with other virtual machines, we set the number of iterations to $N^2$, and N is the number of virtual machines. Then we calculate the difference between the utilization of the two destination hosts (line 3), exchange their destination hosts, update the resources of the destination hosts and calculate the difference between them again (line 4-5). If the difference is reduced after the exchange, the load of the two hosts is believed more balanced than before and some SLA violations are avoided. Therefore, the original combination is replaced to output to $S_l$ (line 6-7). Otherwise, the original combination is output to $S_l$ (line 8-10). Finally, we get the optimized scheme $S_l$.

Local search has achieved a certain degree of avoidance of SLA violations. However, the local search does not produce a diversity of solutions. Moreover, the populations we introduced operate independently and do not cooperate with each other, so each population cannot learn valuable information from each other until now. Therefore, we design the pheromone exchange rule to allow one population to learn the excellent combinations from another, and retain these combinations as much as possible in future iterations. The pheromone exchange rule is shown in Algorithm3: we select a combination ($VM$, $PM$) from one scheme $S_1$, which is one of the population-constructed schemes, and then find out if there is the same combination in another scheme $S_2$ (line 1-2). If the two populations select the same combination (line 3), that combination is believed excellent, and their pheromone will be replaced by their updated pheromones (line 4-6). The probability of selecting this combination will increase in the next time. Finally, we get the updated schemes $S_1$ and $S_2$.

We can implement load balancing in the datacenter by the local search strategy. Moreover, the pheromone exchange strategy enables to preserve excellent combinations, increasing the probability of being selected in subsequent iterations. Therefore, we will optimize the migration plan and avoide SLA violations.

### D. VIRTUAL MACHINE CONSOLIDATION ALGORITHM

We have described all the components of our algorithm in detail so far, and we will explain the complete process of our proposed ELM_MPACS algorithm below. Our ELM_MPACS algorithm uses different placement rules for virtual machines on the overloaded and underloaded hosts. The virtual machines on the overloaded hosts will be migrated to the normal hosts with more available resources, while the virtual machines on the underloaded hosts will be migrated to the underloaded hosts with higher usage.

Algorithm 4 shows the pseudo-code for virtual machine consolidation. The algorithm inputs are various host lists obtained by Algorithm 1 and virtual machine list to be migrated, which are consisted of all virtual machines on the underloaded host and the virtual machines obtained from the overloaded host according to the principle of minimum migration time. Every population generates its scheme concurrently for each iteration, and every ant builds a scheme in the population (line 1-3), where $nI$ is the number of iterations, $nA$ is the number of ants in the population and $nC$ is the number of populations. If there are virtual machines to be migrated, we randomly select one of them to allocate destination host (line 4-15). If the source host of the virtual machine is overloaded, we use equation (13) to compute heuristic. And if the source host of the virtual machine is underloaded, we use equation (14) to compute heuristic. Then we assign the destination host for virtual machine according to pseudo-random-proportional rule (line 10-15). If the utilization of the destination host after the migration is lower than that of the source host for VMs from overloaded hosts, we will add the combination (VM, PM) to the temporary migration scheme $T$ (line 16-23). Each ant constructs its plan and calculates the fitness according to equation (4), and selects the scheme with the highest fitness (line 32) from each population. Then the local search strategy (line 33) stated by Algorithm 2 is applied to every best solution in each population and the pheromone exchange algorithm (line 35) shown by Algorithm 3 is used among populations. Finally, the highest-scoring scheme for all populations is the final scheme $S^+$ (line 36), and global pheromone is updated (line 37).

To avoid the load of destination host excessive capacity caused by virtual machine migration, we ask the utilization of the destination host cannot be higher than that of the source host after the migration (line 16-23). By contrast, the virtual machine from the underloaded host can only be migrated to the destination host with higher usage than that of the source host (line 24-28). By adding this constraint, the underloaded hosts with lower usage can be shut down more quickly, while those with higher usage will restore to normal ones gradually. It avoids unnecessary migrations and speeds up the migrating process at the same time.

As shown in Algorithm 4, we can conclude that the time complexity of the algorithm is $O(nI \cdot nA \cdot M \cdot N)$, where $nI$ is the number of iterations, $nA$ is the number of ants in the population, $M$ is the number of hosts, and $N$ is the number of virtual machines. Because multiple populations are concurrently executed, the number of the populations does not affect the complexity. The while loop traverses on $N$ virtual machines in line 4, and every VM needs to traverse on $M$ destination hosts in line 6 and line 8 to update heuristic information. Line 12 and line 14 select the destination host according to the pseudo-random-proportional rule and also need to traverse on $N$ destination hosts. Moreover, we deal with overloaded hosts and underloaded hosts separately, so the number of virtual machines and destination hosts traversed each time is smaller, which narrows the search space as well. The complexity of the local search algorithm is $O(N \cdot N)$. After all ants in the population have constructed the schemes, the local search will be executed. The complexity of the ant colony system is $O(nA \cdot M \cdot N)$ before executing the local search. Because $O(N \cdot N)$ is less than $O(nA \cdot M \cdot N)$, the local search does not affect the complexity. The complexity of pheromone exchange is $O(N \cdot N)$ as well. The pheromone exchange is carried out after each iteration and multiple populations construct solutions concurrently, so the pheromone exchange will not increase the complexity of ELM_MPACS, and the final complexity of the algorithm is $O(nI \cdot nA \cdot M \cdot N)$.

**Algorithm 4** Proposed ELM_MPACS Algorithm

```
Input: overloadedHostList, normalHostList,
    underloadedHostList, VMListToMigrate,
    $nI, nA, nC, \alpha, \beta, \tau_0, \rho, T = \emptyset, S^+ = \emptyset$
Output: S
 1: for i ∈ $[1, nI]$ do
 2:   for j ∈ $[1, nC]$ do
 3:     for k ∈ $[1, nA]$ do
 4:       while VMListToMigrate!= $\emptyset$ do
 5:         if VM.sourceHost is overloadedHost then
 6:           compute heuristic using (13)
 7:         else
 8:           compute heuristic using (14)
 9:         end if
10:         generate a random variable $q \in [0,1]$
11:         if $q \le q_0$ then
12:           choose a destination host using (15)
13:         else
14:           choose a destination host using (16)
15:         end if
16:         if VM.sourceHost is overloadedHost then
17:           update resources
18:           if utilization of destination host < utilization
              of source host then
19:             add $(VM, PM)$ to $T$
20:             update local pheromone using(17)
21:           else
22:             restore resources
23:           end if
24:         else if utilization of sourceHost < utilization of
            destionation host then
25:           add $(VM, PM)$ to $T$
26:           update local pheromone using (17)
27:           update resources
28:         end if
29:       end while
30:       compute the score of $T$ using (4)
31:     end for
32:     choose the best score among the T
33:     local search using Algorithm 2
34:   end for
35:   exchange pheromone using Algorithm 3
36:   $S^+ = \arg\max_{j \in nC} f(T)$
37:   update global pheromone using (18)
38: end for
```

## V. EXPERIMENTAL SETUP AND RESULTS ANALYSIS

### A. EXPERIMENT ENVIRONMENT

We demonstrate the advancement and effectiveness of our proposed ELM_MPACS algorithm in the CloudSim toolkit [46]. CloudSim is a scalable cloud simulation platform that supports energy-aware computing resources modeling and custom virtual machine consolidation methods. Our simulation experiments use two different specifications of dual-core physical machines with 400 for each type. Our simulated data center is also configured with four kinds of virtual machines of varying frequency. Table 1 and Table 2 show the specifications of various types of physical machines and virtual machines in the data center, including CPU frequency, bandwidth and memory.

**TABLE 1.** Physical machine types.

| Host Type | Frequency/MHz | Core | RAM/GB | BW/(Gbits/s) |
|---|---|---|---|---|
| Type1 | 1860 | 2 | 4 | 1 |
| Type2 | 2660 | 2 | 4 | 1 |

The dataset we used is from the CoMon system, which monitors the operation of the PlanetLab infrastructure and collects data from each node of the PlanetLab [47]. Table 3 shows details about the dataset. As shown in Table 3, the dataset recorded about 1000 virtual machines for ten days from March and April in 2011. To prove that our virtual machine consolidation algorithm is suitable for

**TABLE 2. Virtual machine types.**

| VM Type | Frequency/MHz | RAM/MB | BW/(Mbits/s) |
|---|---|---|---|
| Type1 | 2500 | 870 | 100 |
| Type2 | 2000 | 1740 | 100 |
| Type3 | 1000 | 1740 | 100 |
| Type4 | 500 | 613 | 100 |

**TABLE 3. PlanetLab data trace.**

| NO. | Date | Number of VMs |
|---|---|---|
| 1 | 3 March 2011 | 1052 |
| 2 | 6 March 2011 | 898 |
| 3 | 9 March 2011 | 1061 |
| 4 | 22 March 2011 | 1516 |
| 5 | 25 March 2011 | 1078 |
| 6 | 3 April 2011 | 1463 |
| 7 | 9 April 2011 | 1358 |
| 8 | 11 April 2011 | 1233 |
| 9 | 12 April 2011 | 1054 |
| 10 | 20 April 2011 | 1033 |

**TABLE 4. ELM_MPACS parameters.**

| $\alpha$ | $\beta$ | $\rho$ | $q_0$ | $nI$ | $nC$ | $nA$ |
|---|---|---|---|---|---|---|
| 1 | 1 | 0.1 | 0.9 | 3 | 3 | 3 |

large-scale data centers, we conducted experiments on the data of 20110322 because the number of virtual machines on this day is the most in the ten days. In our experiment, the scheduling period is set to five minutes, and the algorithm will be scheduled 288 times in 24 hours. The parameter settings in our proposed algorithm are shown in Table 4. We obtain these parameters through abundant experiments. Also, $\alpha$ and $\beta$ are the weight of pheromone and heuristic information, $\rho$ is the evaporation coefficient, $q_0$ is a constant, $nI$ is the number of iterations, $nC$ is the number of populations, and $nA$ is the number of ants in the population.

## B. EVALUATION METRICS

The total Energy Consumption (EC) of the data center is an essential metric for evaluating an algorithm. A host consumes different power at various CPU utilization, and the power specifications of the varying type of hosts under the same CPU utilization are also different. We use the power data given in SPECpower,[1] which is shown in Table 5. From Table 5, we can see that the power consumption of the host

[1] http://www.spec.org/power/ssj2008/

**TABLE 5. Host power consumption Information in Watt.**

| Host Type | 0% | 10% | 20% | 30% | 40% | 50% | 60% | 70% | 80% | 90% | 100% |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Type1 | 86 | 89.4 | 92.6 | 96 | 99.5 | 102 | 106 | 108 | 112 | 114 | 117 |
| Type2 | 93.7 | 97 | 101 | 105 | 110 | 116 | 121 | 125 | 129 | 133 | 135 |

in the idle state still occupies over 70% of the peak power consumption. The number of migrations (NM) refers to the total counts of migrations for each scheduling period during the entire running time. In real-world scenarios, virtual machine live migrations can cause poor user experience, so it is necessary to reduce the number of migrations. The number of migrations is expressed as

$$\mathrm{NM} = \sum_{k=1}^{T} Mig_k, \tag{19}$$

where the NM is the total number of migrations, $T$ is the total number of scheduling, $Mig_k$ refers to the number of migrations of $k$th scheduling period.

Performance Degradation due to Migration (PDM) is used to measure the impact of migration on host performance. Like the setting in [20], the estimator of the virtual machine performance degradation is set to the 10% CPU utilization. PDM is calculated as

$$\mathrm{PDM} = \frac{1}{N} \sum_{i=1}^{N} \frac{C_d^i}{C_r^i}, \tag{20}$$

where $N$ is the number of virtual machines, $C_d^i$ is an estimated value of performance degradation caused by migrations, and $C_r^i$ is the total resource requested by the virtual machine.

PDM per migration is used to measure the average PDM for all migrations. PDM is calculated as

$$\text{PDM per migration} = \frac{\mathrm{PDM}}{\mathrm{NM}}, \tag{21}$$

where PDM is Performance Degradation due to Migration, NM is the number of migrations.

SLA violation Time per Active Host (SLATAH) is the average ratio of the overload time to the total time of all active hosts, which indicates the overload situation of the entire data center. If the total resources requested by the virtual machine exceed the resources that the host can provide, the host without available resources is deemed as overloaded. SLATAH is formulated as

$$\mathrm{SLATAH} = \frac{1}{M} \sum_{j=1}^{M} \frac{T_o^j}{T_a^j}, \tag{22}$$

where $M$ is the number of physical machines, $T_o^j$ is the overload time, and $T_a^j$ is the running time of the host.

SLA violations (SLAV) occur when cloud service providers are unable to guarantee the quality of service in accordance with SLA. SLA violations will not only result in a decline in the user experience but also lead to punishment for cloud service providers. [20] proposed a measure of

**TABLE 6. Experimental results.**

| Algorithm | EC(kWh) | NM | SLAV$*10^{-7}$ | PDM$*10^{-5}$ | PDM per migration$*10^{-7}$ | SLATAH$*10^{-2}$ | ESV$*10^{-4}$ |
|---|---|---|---|---|---|---|---|
| IQR_MMT_1.5 | 202.76 | 28899 | 281 | 57.3 | 0.19828 | 4.9148 | 56.98 |
| THR_MMT_0.8 | 201.80 | 29165 | 320 | 62.5 | 0.2143 | 5.0006 | 64.58 |
| MAD_MMT_2.5 | 198.31 | 28243 | 283 | 57.5 | 0.20359 | 4.9253 | 56.12 |
| LR_MMT_1.2 | 176.57 | 31349 | 457 | 73.5 | 0.23446 | 6.2257 | 80.69 |
| MSE_MMT_4.0 | 165.09 | 4268 | 24 | 6.8 | 0.15933 | 3.473 | 3.96 |
| ACS_VMC | 144.91 | 24904 | 285 | —— | —— | —— | 41.30 |
| ST_MPACS | 138.59 | 5969 | 22 | 8.2 | 0.13738 | 2.602 | 3.05 |
| ELM_MPACS | 138.99 | 2881 | 9 | 4.2 | 0.14578 | 2.0459 | 1.25 |

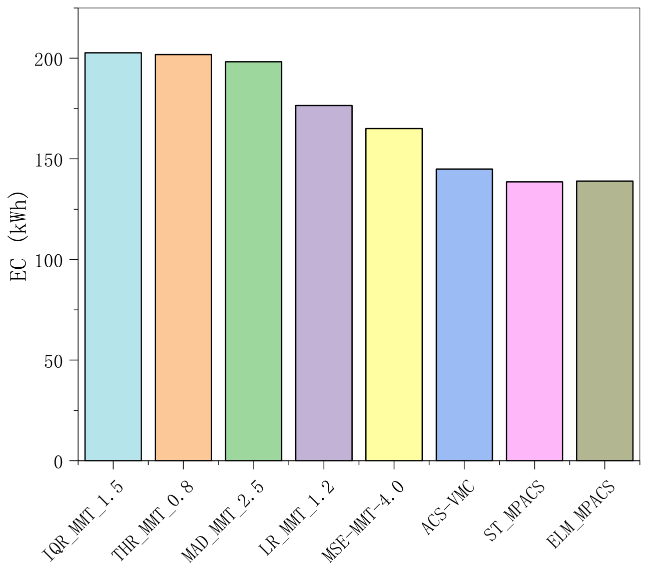


**FIGURE 3. Comparison of energy consumption.**

SLA violation consisting of SLATAH and PDM, which are independent of each other and equally important. The smaller the SLAV value, the smaller the number of SLA violations, and the better the QoS. The formula is as follows

$$SLAV = SLATAH * PDM. \tag{23}$$

EC and SLAV only show the performance of different algorithms from two aspects, and cannot comprehensively evaluate the performance of different algorithms. The ESV can weigh the energy consumption and SLAV, which is an objective evaluation of the pros and cons of different algorithms. ESV is defined as

$$ESV = EC * SLAV. \tag{24}$$

The ESV is proportional to these two values, and any metric increases will increase the ESV value. A smaller ESV value shows a better trade-off between energy consumption and the SLAV.

### C. ANALYSIS OF RESULTS

Our proposed algorithm ELM_MPACS is compared with four benchmark algorithms in CloudSim on the dataset "20110322". The host overload detection algorithms used by four algorithms are the Inter Quartile Range (IQR), the static threshold (THR), the Median Absolute Difference (MAD) and the Local Regression (LR), while the virtual machine selection algorithm uses Minimum Migration Time algorithm (MMT) [20]. The parameters attached to each name are security parameters, which determine the degree of virtual machine consolidation. We also compared ELM_MPACS with the latest MSE_MMT_4.0 algorithm [11] and ACS_VMC [23] to prove the advancement of our proposed algorithm. The formal is based on the heuristic algorithm, using Mean Square Error (MSE) as a correction of linear regression prediction; the latter is based on the ACS, employing the linear regression. Besides, to prove the validity of our ELM prediction module, we compared it with the algorithm ST_MPACS without ELM, where the upper and lower thresholds are set to 0.8 and 0.3, respectively. We run 20 experiments independently and take the mean as the final results. Table 6 shows our experimental results. From Fig. 3 to Fig. 9, the performance of these algorithms is compared in detail with each metric introduced in the previous section.

Fig. 3 shows that the two algorithms we propose have great advantages in terms of energy consumption. Compared with IQR_MMT_1.5, THR_MMT_0.8, MAD_MMT_2.5, LR_MMT_1.2, MSE_MMT_4.0 and ACS_VMC, ELM_MPACS reduced the energy consumption by 31.45%, 31.12%, 29.91%, 21.28%, 15.81% and 4.09% on the "20110322", respectively. Because energy consumption is a major factor in our proposed multi-objective function, we selects the one that minimizes energy consumption from multiple populations. IQR_MMT_1.5, THR_MMT_0.8, MAD_MMT_2.5 and LR_MMT_1.2 only set the upper threshold, while our algorithm employs double thresholds to deal with the virtual machines on the underloaded hosts, so the underloaded hosts are quickly shut down to reduce energy consumption. In addition, ACS_VMC handles the underloaded and the overloaded hosts in a unified manner, which cannot shut down the underloaded hosts in time, resulting in a waste of resources. Besides, we design different algorithms for the features of underloaded and overloaded hosts. We migrate a virtual machine from the underloaded host with lower utilization to the underloaded host with higher utilization, so that the host with lower utilization can be quickly shut down. The host with a higher utilization gradually recovers to the normal host. Therefore, our proposed

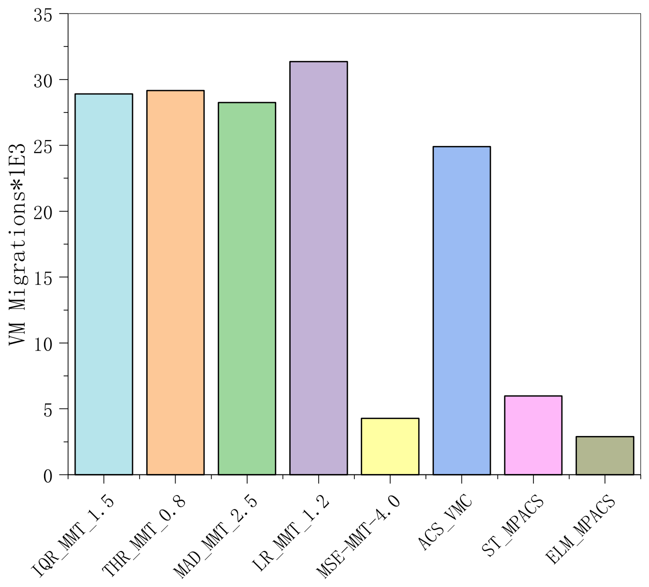


**FIGURE 4.** **Comparison of the number of migrations.**

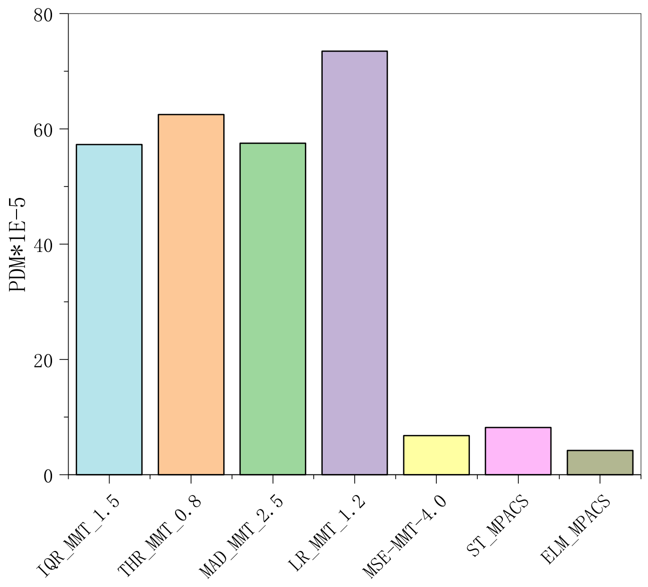


**FIGURE 5.** **Comparison of PDM.**

algorithm will greatly reduce energy consumption compared with other algorithms. The virtual machine consolidation algorithm ultimately determines the energy consumption, and the prediction algorithm directly affects the migration times and SLAV, so ELM_MPACS with ELM consumes almost the same energy as ST_MPACS without ELM. As can be seen from Fig. 3, our algorithms ELM_MPACS and ST_MPACS have the lowest energy consumption among all algorithms.

As shown in Fig. 4, we have fewer migrations than MSE_MMT_4.0 because the ELM prediction algorithm is more accurate than the robust linear regression with error correction, which avoids unnecessary migrations. Although robust linear regression has error correction, it cannot capture the nonlinear changes of the load in data center. In contrast, the ELM algorithm predicts the host load more flexibly, thus avoiding unnecessary migrations. The number of migrations of ELM_MPACS is smaller than that of ACS_VMC. Because ACS_VMC deals with the overloaded and underloaded hosts indiscriminately and traverse all the pre-constructed tuples, it results in a large number of unnecessary migrations. We migrate the virtual machines on the overloaded host to the normal host and migrate the virtual machines on the underloaded to another underloaded host, thus greatly reducing unnecessary migrations. Because ST_MPACS does not use the prediction algorithm, it cannot accurately predict the state of the host in the next period. Moreover, some virtual machines on the host that are currently overloaded or underloaded but will restore to the normal state in the next period are unnecessarily migrated. Therefore, our proposed ELM_MPACS has a smaller number of migrations than ST_MPACS.

We can conclude that our algorithm ELM_MPACS performs better than other algorithms on the PDM from Fig. 5. Compared with IQR_MMT_1.5, THR_MMT_0.8, MAD_MMT_2.5 and LR_MMT_1.2, our ELM_MPACS reduces by 92.67%, 93.28%, 92.70% and 94.29%, respectively. Furthermore, our algorithm reduces by 38.24% than MSE_MMT_4.0. Because our algorithm has the smallest number of migrations, the performance degradation caused by migration can be minimized. This is because PDM is a measure of the impact of migration on host performance, and the number of migrations directly affects the value of PDM. The previous analysis of the number of migrations in Fig. 4 shows that our algorithm significantly reduces the number of migrations, which greatly reduces the impact of migration on host performance. Therefore, the value of PDM of ELM_MPACS is minimal. And Fig. 6 shows the PDM per migration, which reflects the average migration cost. We can see that the values of PDM per migration of our proposed ST_MPACS and ELM_MPACS are less than that of other algorithms. We migrate the virtual machine from one underloaded host to another underloaded host with higher utilization, and migrate the virtual machine from the overloaded host to the normal host, which efficiently avoids fierce competition for resources. After the migrations, the destination hosts are not easily overloaded. Therefore, our virtual machine consolidation methods have the least migration costs and perform better than others.

Fig. 7 depicts the performance of each method on the SLATAH. In our algorithm, the virtual machine on the overloaded host preferentially selects a normal host with more available resources, and the algorithm asks that the utilization of the destination host is under that of the source host after the migration. After the virtual machine is migrated from the source host to the destination host, the source host will restore from the overload state to the normal state. Because the destination host has enough available resources, it will not overload due to migration. Also, our local search algorithm re-optimizes the placement scheme, rebalance the load between different hosts and avoid the risk of overloading

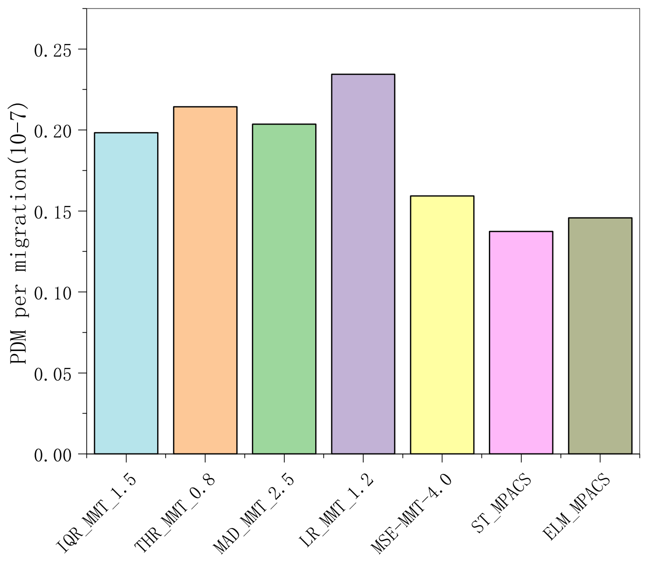


**FIGURE 6.** Comparison of PDM per migration.

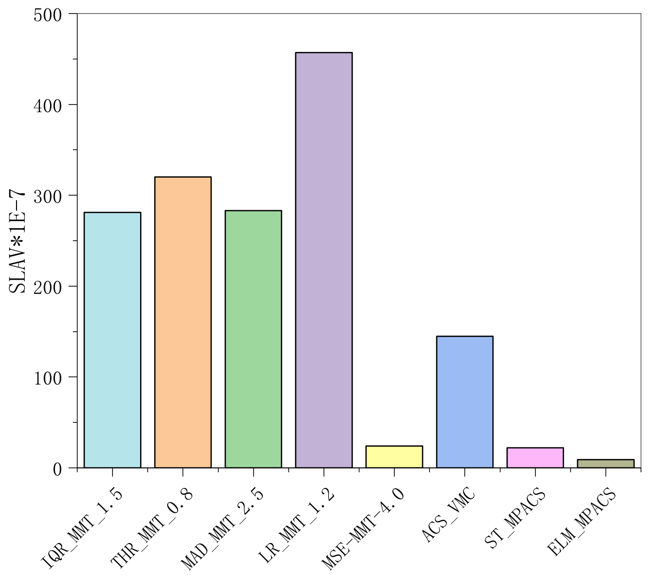


**FIGURE 8.** Comparison of SLAV.

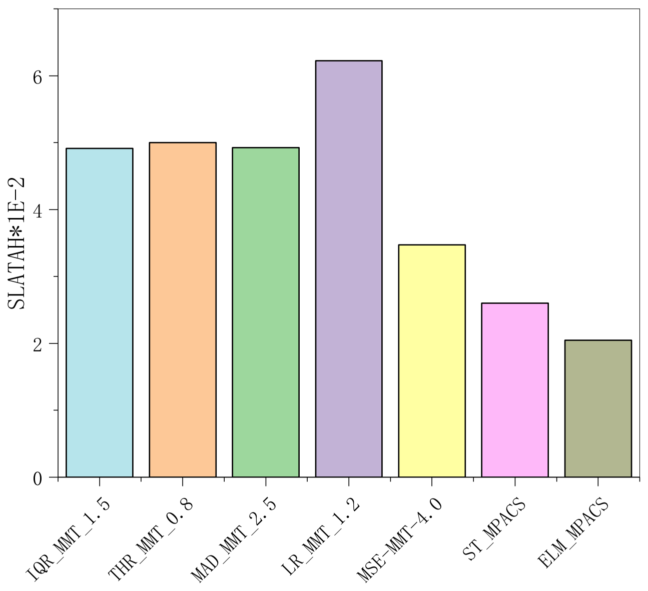


**FIGURE 7.** Comparison of SLATAH.

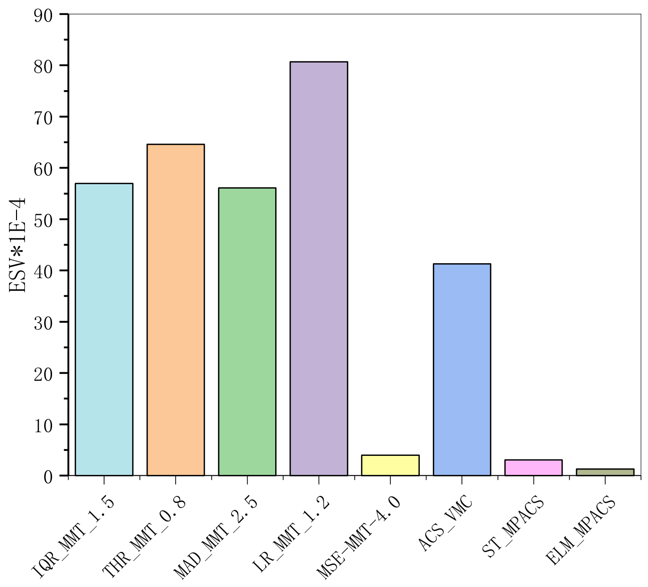


**FIGURE 9.** Comparison of ESV.

the destination host as much as possible due to the migration, so our ELM_MPACS can obtain a smaller value of SLATAH. Our prediction algorithm migrates some of the virtual machines on the host that might be overloaded in advance to avoid SLA violations. The ST_MPACS algorithm does not predict the host state in advance, so some virtual machines are migrated to the hosts that may be overloaded in the next period, which will aggravate the overload. Therefore, our ELM_MPACS algorithm performs better on SLATAH than any other algorithm.

SLAV is directly proportional to PDM and SLATAH, which jointly determine SLAV. According to the previous analysis, the values of PDM and SLATAH of our ELM_MPACS algorithm both are the smallest, so SLAV value is also the smallest. We can see that the SLAV of ELM_MPACS is only 3.16% of ACS_VMC compared with ACS_VMC from Fig. 8. Our ELM_MPACS algorithm can effectively reduce the number of migrations, identify overloaded hosts and migrate virtual machines on them in time, so our algorithm is indeed the best among these algorithms. ESV evaluated the performance of EC and SLAV comprehensively and is a more objective and all-round metric for each algorithm. From Fig. 9, we see that our ELM_MPACS algorithm is far better than IQR_MMT_1.5, THR_MMT_0.8, MAD_MMT_2.5 and LR_MMT_1.2 after considering energy consumption and SLAV. ACS_VMC focuses on reducing power consumption, and MSE_MMT_4.0 prefers to reduce SLAV. However, our algorithm consumes less energy than ACS_VMC and obtains a smaller SLAV value than MSE_MMT_4.0.

Moreover, we have decreased by 96.97% and 68.43% on ESV, respectively, compared with ACS_VMC and the latest MSE_MMT_4.0.

The experimental results show that our proposed ELM_MPACS effectively reduces the energy consumption in the data center, migration times, and SLAV. Moreover, the precise prediction of the ELM can avoid invalid virtual machine migrations and reduce SLA violations. The local search strategy optimizes the migration plan and further reduces the SLA violation, and the pheromone exchange between the populations retains some excellent combinations to some extent. Multiple populations construct schemes concurrently, allowing us to choose the best solution. Compared with the heuristic algorithm and the meta-heuristic algorithm, the experimental results prove the advancement and effectiveness of our algorithm.

## VI. CONCLUSION AND FUTURE WORK

In this paper, we have proposed a virtual machine consolidation algorithm based on ELM and ACS. Firstly, we build multiple ELM models to predict the host state for the next period. We used multi-population ACS algorithms with local search to select the destination host for the virtual machine one by one, which reduces the search space. Moreover, we analyzed the features of underloaded hosts and overloaded hosts and designed different heuristic information and migration rules. In order to prove the advancement of our proposed ELM_MPACS, we compared it with other seven algorithms of IQR_MMT_1.5, THR_MMT_0.8, MAD_MMT_2.5, LR_MMT_1.2, MSE_MMT_4.0, ACS_VMC and ST_MPACS on the CloudSim platform. We conclude that the prediction algorithm ELM can accurately predict the state of the host in the next scheduling cycle. The ELM can avoid unnecessary migrations and reduce SLA violations effectively according to the experiment results. The adoption of the local search algorithm further optimizes the solution, balances the loads between different hosts and reduces SLA violations. The pheromone exchange between populations accumulates the pheromone concentration of the excellent combinations, increasing the likelihood that the combination will be selected next time. Each time, the best solution is selected from multiple populations, which increases diversity and ensures convergence. Therefore, our algorithm effectively reduces energy consumption, migration times and SLA violations. In future, we want to apply our scheduling algorithm to real-world data centers to reduce energy consumption.

## REFERENCES

[1] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, and M. Zaharia, "A view of cloud computing," *Commun. ACM*, vol. 53, no. 4, pp. 50–58, 2010.

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

[3] E. Arianyan, H. Taheri, and S. Sharifian, "Novel energy and SLA efficient resource management heuristics for consolidation of virtual machines in cloud data centers," *Comput. Elect. Eng.*, vol. 47, pp. 222–240, Oct. 2015.

[4] M. Dayarathna, Y. Wen, and R. Fan, "Data center energy consumption modeling: A survey," *IEEE Commun. Surveys Tuts.*, vol. 18, no. 1, pp. 732–794, 1st Quart., 2016.

[5] D. Ardagna, G. Casale, M. Ciavotta, J. F. Pérez, and W. Wang, "Quality-of-service in cloud computing: Modeling techniques and their applications," *J. Internet Serv. Appl.*, vol. 5, no. 1, p. 11, 2014.

[6] S. Wang, A. Zhou, C.-H. Hsu, X. Xiao, and F. Yang, "Provision of data-intensive services through energy- and QoS-aware virtual machine placement in national cloud data centers," *IEEE Trans. Emerg. Topics Comput.*, vol. 4, no. 2, pp. 290–300, Apr./Jun. 2015.

[7] M. Pantazoglou, G. Tzortzakis, and A. Delis, "Decentralized and energy-efficient workload management in enterprise clouds," *IEEE Trans. Cloud Comput.*, vol. 4, no. 2, pp. 196–209, Apr./Jun. 2015.

[8] Q. Zhang, H. Wang, F. Zhu, S. Yi, K. Feng, and L. Zhai, "Energy-aware VM placement with periodical dynamic demands in cloud datacenters," in *Proc. IEEE 19th Int. Conf. High Perform. Comput. Commun.; IEEE 15th Int. Conf. Smart City; IEEE 3rd Int. Conf. Data Sci. Syst. (HPCC/SmartCity/DSS)*, Dec. 2017, pp. 162–169.

[9] H. Zhao, J. Wang, F. Liu, Q. Wang, W. Zhang, and Q. Zheng, "Power-aware and performance-guaranteed virtual machine placement in the cloud," *IEEE Trans. Parallel Distrib. Syst.*, vol. 29, no. 6, pp. 1385–1400, Jun. 2018.

[10] M. A. Khoshkholghi, M. N. Derahman, A. Abdullah, S. Subramaniam, and M. Othman, "Energy-efficient algorithms for dynamic virtual machine consolidation in cloud data centers," *IEEE Access*, vol. 5, pp. 10709–10722, 2017.

[11] L. Li, J. Dong, D. Zuo, and J. Wu, "Sla-aware and energy-efficient VM consolidation in cloud data centers using robust linear regression prediction model," *IEEE Access*, vol. 7, pp. 9490–9500, 2019.

[12] F. Farahnakian, T. Pahikkala, P. Liljeberg, J. Plosila, and H. Tenhunen, "Utilization prediction aware VM consolidation approach for green cloud computing," in *Proc. IEEE 8th Int. Conf. Cloud Comput. (CLOUD)*, Jun./Jul. 2015, pp. 381–388.

[13] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew, "Extreme learning machine: A new learning scheme of feedforward neural networks," *Neural Netw.*, vol. 2, pp. 985–990, Jul. 2004.

[14] M. Vasudevan, Y.-C. Tian, M. Tang, E. Kozan, and X. Zhang, "Energy-efficient application assignment in profile-based data center management through a repairing genetic algorithm," *Appl. Soft Comput. J.*, vol. 67, pp. 399–408, Jun. 2018.

[15] X.-F. Liu, Z.-H. Zhan, J. D. Deng, Y. Li, T. L. Gu, and J. Zhang, "An energy efficient ant colony system for virtual machine placement in cloud computing," *IEEE Trans. Evol. Comput.*, vol. 22, no. 1, pp. 113–128, Feb. 2018.

[16] S. Chaisiri, B.-S. Lee, and D. Niyato, "Optimal virtual machine placement across multiple cloud providers," in *Proc. IEEE Asia–Pacific Serv. Comput. Conf. (APSCC)*, Dec. 2009, pp. 103–110.

[17] B. Speitkamp and M. Bichler, "A mathematical programming approach for server consolidation problems in virtualized data centers," *IEEE Trans. Services Comput.*, vol. 3, no. 4, pp. 266–278, Oct./Dec. 2010.

[18] A. Anand, J. Lakshmi, and S. K. Nandy, "Virtual machine placement optimization supporting performance SLAs," in *Proc. IEEE 5th Int. Conf. Cloud Comput. Technol. Sci. (CloudCom)*, vol. 1, Dec. 2013, pp. 298–305.

[19] A. Murtazaev and S. Oh, "Sercon: Server consolidation algorithm using live migration of virtual machines for green computing," *IETE Tech. Rev.*, vol. 28, no. 3, pp. 212–231, 2011.

[20] 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 Comput., Pract. Exper.*, vol. 24, no. 13, pp. 1397–1420, Sep. 2012.

[21] A. Aryania, H. S. Aghdasi, and L. M. Khanli, "Energy-aware virtual machine consolidation algorithm based on ant colony system," *J. Grid Comput.*, vol. 16, no. 3, pp. 477–491, Sep. 2018.

[22] H. Xiao, Z. Hu, and K. Li, "Multi-objective VM consolidation based on thresholds and ant colony system in cloud computing," *IEEE Access*, vol. 7, pp. 53441–53453, 2019.

[23] F. Farahnakian, A. Ashraf, T. Pahikkala, P. Liljeberg, J. Plosila, I. Porres, and H. Tenhunen, "Using ant colony system to consolidate VMs for green cloud computing," *IEEE Trans. Services Comput.*, vol. 8, no. 2, pp. 187–198, Mar./Apr. 2015.

[24] R. Yadav and W. Zhang, "MeReg: Managing energy-SLA tradeoff for green mobile cloud computing," *Wireless Commun. Mobile Comput.*, vol. 2017, Dec. 2017, Art. no. 6741972.

[25] 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.

[26] Z. Zhou, J. Abawajy, M. Chowdhury, Z. Hu, K. Li, H. Cheng, A. A. Alelaiwi, and F. Li, "Minimizing SLA violation and power consumption in cloud data centers using adaptive energy-aware algorithms," *Future Gener. Comput. Syst.*, vol. 86, pp. 836–850, Sep. 2018.

[27] F. Farahnakian, P. Liljeberg, and J. Plosila, "LiRCUP: Linear regression based cpu usage prediction algorithm for live migration of virtual machines in data centers," in *Proc. 39th Euromicro Conf. Ser. Softw. Eng. Adv. Appl. (SEAA)*, Sep. 2013, pp. 357–364.

[28] G. J. L. Paulraj, S. A. J. Francis, J. D. Peter, and I. J. Jebadurai, "A combined forecast-based virtual machine migration in cloud data centers," *Comput. Electr. Eng.*, vol. 69, pp. 287–300, Jul. 2018.

[29] G. P. Zhang, "Time series forecasting using a hybrid ARIMA and neural network model," *Neurocomputing*, vol. 50, pp. 159–175, Jan. 2003.

[30] R. Moreno-Vozmediano, R. S. Montero, E. Huedo, and I. M. Llorente, "Efficient resource provisioning for elastic cloud services based on machine learning techniques," *J. Cloud Comput.*, vol. 8, no. 1, p. 5, 2019.

[31] G.-B. Huang, H. Zhou, X. Ding, and R. Zhang, "Extreme learning machine for regression and multiclass classification," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 42, no. 2, pp. 513–529, Apr. 2012.

[32] D. G. Lago, E. R. M. Madeira, and D. Medhi, "Energy-aware virtual machine scheduling on data centers with heterogeneous bandwidths," *IEEE Trans. Parallel Distrib. Syst.*, vol. 29, no. 1, pp. 83–98, Jan. 2018.

[33] W. Zhu, Y. Zhuang, and L. Zhang, "A three-dimensional virtual resource scheduling method for energy saving in cloud computing," *Future Gener. Comput. Syst.*, vol. 69, pp. 66–74, Apr. 2017.

[34] 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, Mar. 2016.

[35] L.-D. Chou, H.-F. Chen, F.-H. Tseng, H.-C. Chao, and Y.-J. Chang, "DPRA: Dynamic power-saving resource allocation for cloud data center using particle swarm optimization," *IEEE Syst. J.*, vol. 12, no. 2, pp. 1554–1565, Jun. 2018.

[36] G. Cao, "Topology-aware multi-objective virtual machine dynamic consolidation for cloud datacenter," *Sustain. Comput. Inform. Syst.*, vol. 21, pp. 179–188, Mar. 2019.

[37] F.-H. Tseng, X. Wang, L.-D. Chou, H.-C. Chao, and V. C. M. Leung, "Dynamic resource prediction and allocation for cloud data center using the multiobjective genetic algorithm," *IEEE Syst. J.*, vol. 12, no. 2, pp. 1688–1699, Jun. 2018.

[38] S. S. Alresheedi, S. Lu, M. A. Elaziz, and A. A. Ewees, "Improved multiobjective salp swarm optimization for virtual machine placement in cloud computing," *Human-Centric Comput. Inf. Sci.*, vol. 9, no. 1, p. 15, 2019.

[39] Z. Li, X. Yu, L. Yu, S. Guo, and V. Chang, "Energy-efficient and quality-aware VM consolidation method," *Futur. Gener. Comput. Syst.*, vol. 102, pp. 789–809, Jan. 2020.

[40] M. Dorigo, G. Caro, and L. Gambardella, "Ant algorithms for discrete optimization," *Artif. Life*, vol. 5, no. 2, pp. 137–172, Apr. 1999.

[41] A. Ashraf and I. Porres, "Multi-objective dynamic virtual machine consolidation in the cloud using ant colony system," *Int. J. Parallel, Emergent Distrib. Syst.*, vol. 33, no. 1, pp. 103–120, 2018.

[42] J. Wang, C. Huang, K. He, X. Wang, X. Chen, and K. Qin, "An energy-aware resource allocation heuristics for vm scheduling in cloud," in *Proc. IEEE 10th Int. Conf. High Perform. Comput. Commun. IEEE Int. Conf. Embedded Ubiquitous Comput.*, Nov. 2013, pp. 587–594.

[43] Q. Zheng, R. Li, X. Li, N. Shah, J. Zhang, F. Tian, K.-M. Chao, and J. Li, "Virtual machine consolidated placement based on multi-objective biogeography-based optimization," *Future Gener. Comput. Syst.*, vol. 54, pp. 95–122, Jan. 2016.

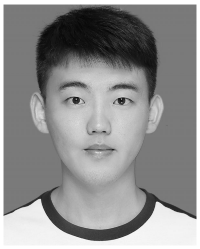
[44] W. Wang, Y. Jiang, and W. Wu, "Multiagent-based resource allocation for energy minimization in cloud computing systems," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 47, no. 2, pp. 205–220, Feb. 2017.

[45] M. Dorigo and L. M. Gambardella, "Ant colony system: A cooperative learning approach to the traveling salesman problem," *IEEE Trans. Evol. Comput.*, vol. 1, no. 1, pp. 53–66, Apr. 1997.

[46] R. 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," *Softw., Pract. Exper.*, vol. 41, no. 1, pp. 23–50, 2011.

[47] K. Park and V. S. Pai, "CoMon: A mostly-scalable monitoring system for PlanetLab," *ACM SIGOPS Oper. Syst. Rev.*, vol. 40, no. 1, pp. 65–74, 2006.

**FAGUI LIU** received the M.S. degree from Beihang University, in 1991, and the Ph.D. degree from the South China University of Technology, China, in 2006. She is currently a Professor with the School of Computer Science and Engineering, South China University of Technology. Her current research interests include service computing, the Internet of things, cloud computing, and big data.



**ZHENJIANG MA** received the B.S. degree from the Department of Information and Computing Science, Taiyuan University of Technology, in 2018. He is currently pursuing the master's degree with the School of Computer Science and Engineering, South China University of Technology. His research interests include cloud computing, cloud resource management, and virtual machine consolidation.

**BIN WANG** received the B.S. degree from the South China University of Technology, Guangzhou, China, in 2014, where he is currently pursuing the Ph.D. degree with the School of Computer Science and Engineering. His research interests include cloud computing, energy efficiency, and data mining.

**WEIWEI LIN** received the B.S. and M.S. degrees from Nanchang University, in 2001 and 2004, respectively, and the Ph.D. degree in computer application from the South China University of Technology, in 2007. He is currently a Professor with the School of Computer Science and Engineering, South China University of Technology. He has published more than 80 articles in refereed journals and conference proceedings. His research interests include distributed systems, cloud computing, big data computing, and AI application technologies. He is a Senior Member of CCF.