Received November 26, 2018, accepted December 30, 2018, date of publication January 9, 2019, date of current version January 29, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2891567

SLA-Aware and Energy-Efficient VM Consolidation in Cloud Data Centers Using Robust Linear Regression Prediction Model

LIANPENG LI, JIAN DONG, DECHENG ZUO, AND JIN WU
Department of Computer Science and Technology, Harbin Institute of Technology, Harbin 150006, China

Corresponding author: Jian Dong (15B303010@hit.edu.cn)

This work was supported by the National Natural Science Foundation of China under Grant 61100029.

ABSTRACT Virtual machine (VM) consolidation provides a promising approach to save energy and to improve resource utilization in data centers. However, the aggressive consolidation of virtual machines may lead to service-level agreements (SLA) violation, which is essential for data centers and their users. Therefore, it is very meaningful to strike a tradeoff between power efficient and reduction of SLA violation level. In this paper, we propose a host overloading/underloading detection algorithm and a new VM placement algorithm based on our proposed robust simple linear regression prediction model for SLA-aware and energy-efficient consolidation of virtual machines in cloud data centers. Different from the native linear regression, our proposed methods amend the prediction and squint toward over-prediction by adding the error to the prediction; in this paper, we propose eight methods to calculate the error. We evaluate our proposed algorithms by extended the CloudSim simulator using PlanetLab workload and random workload. The experimental results show that our proposed model can reduce SLA violation rates by at most 99.16% and energy consumption by at most 25.43% for the real-world workload.

INDEX TERMS VM consolidation, linear regression, SLA-aware, energy-efficient, cloud data centers.

I. INTRODUCTION

Cloud data centers consume significant energy owing to the wide application of IT technology, such as big data and E business. There are thousands to millions of machines in cloud data centers, these infrastructure consume high energy which led to a large amount of carbon dioxide emissions contributing to the greenhouse effect [1]. The energy of cloud data centers has not been effectively utilized, Barroso and Holzle [2] analyzed the resource utilization of thousands of servers over a six-month period, and found that servers just use 10% to 50% of their resources at most of the time.

Virtual Machine (VM) consolidation is an efficient way towards energy conservation in cloud data centers. The VM consolidation technique is applied to migrate Virtual Machines (VMs) into lesser number of active Physical Machines (PMs), so that the PMs which have no VMs can be turned into sleep state. VM consolidation technique can reduce energy consumption of cloud data centers because of the energy consumption by the PM which is in sleep state is significantly lower than the energy consumption by the PM which is in active state [3].

However, aggressive consolidation of VMs can lead to performance degradation, because VMs share the underlying physical resources, and an application may encounter an unexpected resources requirement which may lead to increased response times or even failures. Before providing cloud services, cloud providers should sign Service Level Agreements (SLA) with customers, so it is essential to provide reliable Quality of Service (QoS) for cloud providers. Therefore, it is very meaningful to strike a tradeoff between energy and performance - minimizing energy consumption on the premise of meeting SLA.

A live migration [4] approach is presented in [5] to guarantee high-level of energy efficient and SLA. The authors propose that one of the optimization challenges is to decide which VMs to migrate, when to migrate, where to migrate, and when and which servers to turn on/off. To achieve this goal optimally, it is important to predict the future host state accurately, and make plan for migration of VMs based on the prediction. For example, if a host will be overloaded at next time unit, some VMs should be migrated from the host to keep the host from overloading, and if a host will be underloaded at next time unit, all VMs should be migrated from the host, so that the host can be turn off to save power. There is a linear relationship between the power consumption by servers and their CPU utilization, even when Dynamic Voltage and Frequency Scaling (DVFS) is applied [6], [7]. So in this paper, we focuses on predicting the host CPU utilization to determine when a host is overloaded or underloaded. A prediction model is proposed to forecast the future CPU utilization and named Robust Simple Linear Regression (RobustSLR) prediction model. The main objective of RobustSLR model is to minimize the power consumption and SLA violation level. The linear regression is a strong statistical method that used in machine learning schemes to estimate a prediction function. Different from the native simple linear regression, by adding the error directly or indirectly to the prediction, our proposed methods amend the prediction and squint towards over-prediction. In this paper, we have proposed eight methods to calculate the error. The experimental results show that our proposed methods can significantly reduce SLA violation rates while keeping energy cost efficient. The main contributions of this paper are the following.

1. We have proposed a RobustSLR model to predict the host future CPU utilization, it amends the prediction and squint towards over-prediction by adding the error to the prediction. We propose eight methods to calculate the error.
2. We propose a Host Overloading Detection algorithm based on our proposed RobustSLR model in order to minimize the power consumption and SLA violation.
3. We propose a Host Underloading Detection algorithm based on our proposed RobustSLR model in order to minimize the power consumption and SLA violation. We also propose an adaptive lower utilization threshold based on the interquartile range (IQR) in our Host Underloading Detection algorithm.
4. We propose a new VM placement algorithm which called RobustSLR Power Aware Best Fit Decreasing algorithm. In this algorithm, the future host load state is predicted using our RobustSLR algorithm, and the overloaded and underloaded hosts are excluded from the hostList for VM placement.
5. We propose eight live migration policies based on our RobustSLR model and we extend the Cloudsim [8] simulator for performance evaluation of our proposed algorithms.

The remainder of the paper is organized as follows. In Section 2 we discuss the related work. The proposed RobustSLR model is shown in Section 3. In Section 4 we introduce our SLA-aware and Energy-efficient VM consolidation based on RobustSLR Model. The simulation results are reported in Section 5. We conclude this paper in Section 6.

II. RELATED WORK

There are two main techniques in data centers for energy consumption management: Dynamic Voltage and Frequency Scaling (DVFS) technique and VM consolidation technique. The DVFS technique is to dynamically adjust the chip's operating frequency and voltage according to the different needs of the application program running on the chip for computing power, so as to achieve the purpose of energy saving. There are mainly three categories of DVFS technique: interval-based [9]–[12], inter-task [13], [14] and intratask methods [15], [16]. Interval-based methods adjust the processor frequency by predicting the future CPU utilization, inter-task methods assign different tasks to different speed of processors, and intra-task methods adjust the processor frequency according to the structure of programs. Although DVFS technology improves energy utilization, there is still much room for optimization.

The VM consolidation problem is a NP-hard problem [17], [18]. So VM consolidation problem is often formulated as an optimization problem with the objective to find a near optimal solution. The existing VM consolidation approaches can be mainly divided into two categories: threshold-based heuristics and decision-making based on statistical analysis of historical data. Threshold-based heuristics set appropriate threshold to predict the state of a host by comparing it with the threshold, and then decide the migration of VMs. The threshold-based heuristics method can also be divided into two categories: static threshold-based heuristics and dynamic threshold-based heuristics. Ahamed *et al.* [19] propose a static threshold-based heuristics method, they set two thresholds for host state prediction: an overloaded threshold and a underloaded threshold. A PM is overloaded if its CPU utilization is equal or greater than 90%, and then some VMs should be migrated to avoid overloaded at next time unit. Again, a PM is underloaded if its CPU utilization is equal or less than 10%, and then all VMs should be migrated to save power. However, the static threshold method is not suitable at most of the time, because the workloads in cloud data centers are dynamic and complicated.

Beloglazov and Buyya [5], Beloglazov *et al.* [20], and Xue *et al.* [25] propose two dynamic threshold-based heuristics policies: Median Absolute Deviation (MAD) and Inter Quartile Range (IQR). MAD is a measure of statistical dispersion, it is a robust measure of the variability of a univariate sample of quantitative data. IQR is a measure of statistical dispersion, being equal to the difference between 75th and 25th percentiles, or between upper and lower quartiles. The authors adjust the thresholds using above statistical methods based on the historical data of CPU utilization. Therefore, the thresholds can vary with the workloads of hosts. However, the inherent disadvantages of statistical methods have negative impact on the effectiveness. For example, if the past CPU utilizations of a host which are used for the IQR method are same, the upper threshold will be set to 100%, which obviously is an unsuitable threshold.

Different from the threshold-based heuristics, decision-making policies do not set threshold, they predict the state of a host using the historical data to train some functions. PRACTISE [22] is a neural network framework for predicting the resource utilization. The authors have found that PRACTISE has better prediction accuracy than ARIMA [23] and basic neural network model. They also have used PRACTISE to explore VM scheduling strategy to cater for peak demands in their other research [24]. Neural network prediction method is much more complicated, and is often used for medium term to long term prediction. Linear regression [25] is used to generate an estimated future resource utilization of a PM from analyzing its past resource utilization statistics. It is often used for short term prediction. A weighted linear regression method is used to predict the future workload in [26]. Different from other local regression or linear regression prediction methods, our proposed RobustSLR model amends the prediction and squint towards over-prediction by adding the error directly or indirectly to the prediction.

In [27] researchers propose an ant colony optimization meta-heuristic VM placement algorithm, they model the VM placement problem as a multidimensional bin packing problem, and they make VM placement plan according to the workload and the related resource requirements. Ma *et al.* [28] propose a multi-objective ant colony optimization VM placement algorithm, the optimization goal is the minimization of energy consumption and SLA violations. Beloglazov and Buyya [5], [29] propose a Power Aware Best Fit Decreasing (PABFD) algorithm for VM placement, they choose such a host which has the least power increasing for a VM after the VM is migrated to the host. The algorithm is essentially a greedy algorithm, so VMs are often consolidated aggressively. We propose a new VM placement algorithm by modifying the PABFD algorithm in this paper.

III. ROBUST SIMPLE LINEAR REGRESSION MODEL

In this section, we first give a brief introduction of the Simple Linear Regression Model; and then, we show the detail of our proposed Robust Simple Linear Regression Model.

A. SIMPLE LINEAR REGRESSION MODEL

Simple Linear Regression(SLR) is a statistical method that allows us to summarize and study relationship between two continuous (quantitative) variables: One variable, denoted x, is regarded as the predictor variable, the other variable, denoted y, is regarded as the response variable [30]. The purpose of SLR is finding "the best fitting line" which is called regression function shown in equation (1).

$$\hat{y}_i = b_0 + b_1 x_i \tag{1}$$

When we use equation (1) to predict the actual response y_i, the observed response for experimental unit i, the residual error can be calculated as following:

$$e_i = y_i - \hat{y}_i \tag{2}$$

One way of finding "the best fitting line" is using the least squares criterion method [30]. So we should minimize the Q value:

$$Q = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 \tag{3}$$

The values of b_0 and b_1 can be calculated if the equalities (4) and (5) are satisfied.

$$\frac{\partial Q}{\partial b_0} = -2\sum_{i=1}^{n}(y_i - b_1 x_i - b_0) = 0 \tag{4}$$

$$\frac{\partial Q}{\partial b_1} = -2\sum_{i=1}^{n}(y_i - b_1 x_i - b_0)x_i = 0 \tag{5}$$

then we get the least squares estimates for b_0 and b_1:

$$b_0 = \bar{y} - b_1 \bar{x} \tag{6}$$

$$b_1 = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n}(x_i - \bar{x})^2} \tag{7}$$

where $\bar{x}$ and $\bar{y}$ are the means of the x_i and y_i observations, respectively.

B. ROBUST SIMPLE LINEAR REGRESSION MODEL

There are two evaluation criterion for the SLR model prediction inaccuracies, the false negative prediction rate and the false positive prediction rate. The false negative prediction, which may lead to host overloaded, has a decisive impact on the performance of hosts; but the false positive prediction, which may lead to VM migration, also has an important impact on the performance of hosts. So we need to make tradeoff between the false negative prediction rate and the false positive prediction rate. Safety parameter is a method to reduce the adverse effects. The main idea of safety parameter is reserving a certain proportion of resources for inaccuracies on each server. However, safety parameter method, which keeps a static buffer for the dynamic host load, is not a good measure to minimise the influence of the false negative prediction [22]. So in this paper, we propose an adaptive dynamic method that amends the prediction and squint towards over-prediction by adding the error to the prediction. We propose eight methods to calculate the prediction error in this paper.

- Mean Square Error, typically is denoted as *MSE*, is shown in equation (8). We add *MSE* to the prediction directly, then we get the amendatory prediction which is shown in equation (9).

$$MSE = \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i^2)}{n-2} \tag{8}$$

$$\hat{y}_{n+1} = b_0 + b_1 x_{n+1} + MSE \tag{9}$$

- Root Mean Square Error, typically is denoted as *RMSE*, is shown in equation (10). We add *RMSE* to the prediction directly, then we get the amendatory prediction which is shown in equation (11).

$$RMSE = \sqrt{MSE} \tag{10}$$

$$\hat{y}_{n+1} = b_0 + b_1 x_{n+1} + RMSE \tag{11}$$

- Mean Absolute Error, typically is denoted as MAE, is shown in equation (12).

$$MAE = \frac{\sum_{i=1}^{n} |y_i - \hat{y}_i|}{n} \tag{12}$$

In our experiment, we set $n = 3$ and $n = 10$ to calculate MAE respectively, and the results are denoted as $MAE(3)$ and $MAE(10)$. We add MAE to the prediction directly, then we get the amendatory prediction which is shown in equation (13).

$$\hat{y}_{n+1} = b_0 + b_1 x_{n+1} + MAE \tag{13}$$

- Exponentially Weighted Moving Average Error, we denote it as $EWMAE$. In our research, we set $n = 3$ and $n = 10$ to build two models respectively, and the model formulas are shown in equation (14) and (15).

$$\begin{aligned} &EWMAE(3)\\ &\quad = 0.2|y_{i-2} - \hat{y}_{i-2}| + 0.3|y_{i-1} - \hat{y}_{i-1}| + 0.5|y_i - \hat{y}_i| \end{aligned} \tag{14}$$

$$\begin{aligned} &EWMAE(10)\\ &\quad = 0.06|y_{i-9} - \hat{y}_{i-9}| + 0.07 \sum_{k=0}^{1} |y_{i-8+k} - \hat{y}_{i-8+k}|\\ &\qquad + 0.08 \sum_{k=0}^{3} |y_{i-6+k} - \hat{y}_{i-6+k}| + 0.12|y_{i-2} - \hat{y}_{i-2}|\\ &\qquad + 0.16|y_{i-1} - \hat{y}_{i-1}| + 0.2|y_i - \hat{y}_i| \end{aligned} \tag{15}$$

We add $EWMAE$ to the prediction directly, then we get the amendatory prediction which is shown in equation (16).

$$\hat{y}_{n+1} = b_0 + b_1 x_{n+1} + EWMAE \tag{16}$$

- standard error of the slope estimate, typically is denoted as $se(b_1)$, is shown in equation (17). We add $se(b_1)$ to b_1, then we get the amendatory prediction which is shown in equation (18).

$$se(b_1) = \frac{RMSE}{\sqrt{\sum (x_i - \bar{x})^2}} \tag{17}$$

$$\hat{y}_{n+1} = b_0 + (b_1 + se(b_1)) x_{n+1} \tag{18}$$

- $(1 - \alpha)$Confidence Interval of the slope estimate. The formula is shown in equation (19).

$$(1-\alpha)CI_{slope} = b_1 \pm t_{(\alpha/2,n-2)} \times \left(\frac{RMSE}{\sqrt{\sum (x_i - \bar{x})^2}}\right) \tag{19}$$

the $t_{(\alpha/2,n-2)}$ is determined using a t-distribution with $n-2$ degrees of freedom. The value of $t_{(\alpha/2,n-2)}$ is such that the $(1-\alpha)$ confidence level is the area (probability) between $-t_{(\alpha/2,n-2)}$ and $+t_{(\alpha/2,n-2)}$ under the $t-$curve. In our research, we set $\alpha = 0.05$, to calculate the 95% Confidence Interval of the slope estimate, we denote it as $95\%CI_{slope}$. We add $95\%CI_{slope}$ to b_1, then we get the amendatory prediction which is shown in equation (20).

$$\hat{y}_{n+1} = b_0 + (b_1 + 95\%CI_{slope})) x_{n+1} \tag{20}$$

According to the eight adaptive dynamic methods which we have introduced above, we propose our Robust Simple Linear Regression algorithm for predicting the host CPU utilization according to the history of past CPU utilization. The RobustSLR algorithm calculates the host CPU utilization according to the n last CPU utilization in a host. The value of n is set to 3 or 10 in our simulation, and the interval of utilization measurements is five minutes. Algorithm 1 is the pseudo-code of RobustSLR algorithm.

Algorithm 1 Robust Simple Linear Regression (RobustSLR)

```
Input: The set of utilization history, $utilizationHistory$;
Output: The prediction of utilization, $prediction$;
1: $n = utilizationHistory.lenth$;
2: for $i = 0$ to $n - 1$ do
3:     $x[i] = i + 1$;
4:     $y[i] = utilizationHistory[i]$;
5: end for
6: $\bar{x} = \sum_i x_i/n$;
7: $\bar{y} = \sum_i y_i/n$;
8: calculate $b_0$ and $b_1$ according to equation (6) and (7);
9: if use $MSE$ to revise then
10:     calculate $MSE$ according to equation (8);
11:     calculate $prediction$ according to equation (9);
12: else if use $RMSE$ to revise then
13:     calculate $RMSE$ according to equation (10);
14:     calculate $prediction$ according to equation (11);
15: else if use $MAE(3)$ to revise then
16:     calculate $MAE(3)$ according to equation (12);
17:     calculate $prediction$ according to equation (13);
18: else if use $MAE(10)$ to revise then
19:     calculate $MAE(10)$ according to equation (12);
20:     calculate $prediction$ according to equation (13);
21: else if use $EWMAE(3)$ to revise then
22:     calculate $EWMAE(3)$ according to equation (14);
23:     calculate $prediction$ according to equation (16);
24: else if use $EWMAE(10)$ to revise then
25:     calculate $EWMAE(10)$ according to equation (15);
26:     calculate $prediction$ according to equation (16);
27: else if use $se(b_1)$ to revise then
28:     calculate $se(b_1)$ according to equation (17);
29:     calculate $prediction$ according to equation (18);
30: else if use $95\%CI_{slope}$ to revise then
31:     calculate $95\%CI_{slope}$ according to equation (19)
        using $\alpha = 0.05$;
32:     calculate $prediction$ according to equation (20);
33: else
34:     we should not be here, report some error;
35: end if
36: return $prediction$;
```

Initially, we use the host utilization history to calculate b_0 and b_1 according to equation (6) and (7)(line 1-8). Then, the prediction error and the amendatory prediction are calculated according to the eight adaptive dynamic methods(line 9-35). Finally, the prediction of host CPU utilization for the next time unit is returned for detecting whether the host is overloaded(line 36).

IV. SLA-AWARE AND ENERGY-EFFICIENT VM CONSOLIDATION BASED ON RobustSLR MODEL

The VM live migration problem can be divided into four parts: (1) detecting when a host is overloading; (2) detecting when a host is underloading; (3) selecting some VMs from the overloaded hosts and all VMs from the underloaded hosts for live migration; and (4) making a placement plan for all VMs which have been chosen from the overloaded and underloaded hosts. We discuss the details in the following sections.

A. HOST OVERLOADING DETECTION

We apply our proposed RobustSLR algorithm in host overloading detection. Algorithm 2 is the pseudo-code of Host Overloading Detection (HOD) algorithm.

Algorithm 2 Host Overloading Detection (HOD)

```
Input: $host$;
Output: $boolean$;
1: if $utilizationHistory.lenth < 10$ then
2:     return $host.getTotalRequestedMips() > 0.90 * host.getTotalMips()$;
3: else
4:     $prediction = RobustSLR(utilizationHistory)$ ;
5:     return $prediction >= 1$;
6: end if
```

Initially, a host is cosidered overloaded if the current CPU utilization is greater than 90% of the host resources if there are not enough data(at least 10) to be computed for the RobustSLR model(line 1-2). Then, the prediction is calculated according to the RobustSLR algorithm(line 4). Finally, whether the host is overloaded at the next time unit is returned according to the prediction(line 5).

B. HOST UNDERLOADING DETECTION

We propose an adaptive lower utilization threshold based on a robust statistic method, the interquartile range (IQR) method, which is the difference between the third and first quartiles:

$$IQR = Q_3 - Q_1 \tag{21}$$

We set a lower threshold using the IQR method:

$$T_l = 0.4(1 - s \cdot IQR) \tag{22}$$

where s is a safety parameter for VM consolidate. The higher s, the lower level of SLA violations, but the more the energy consumption, and vice versa.

Algorithm 3 Host Underloading Detection (HUD)

```
Input: $host$;
Output: $boolean$;
1: if $utilizationHistory.lenth < 10$ then
2:     return $host.getTotalRequestedMips() < 0.10 * host.getTotalMips()$;
3: else
4:     $T_l = 0.4(1 - s \cdot IQR)$;
5:     $prediction = RobustSLR(utilizationHistory)$ ;
6:     return $prediction < T_l$;
7: end if
```

Algorithm 3 describes the host underloading detection mechanism. Initially, a host is cosidered underloaded if the current CPU utilization is less than 10% of the host resources if there are not enough data (at least 10) to be computed for the RobustSLR model(line 1-2). Then, a lower threshold for CPU utilization is calculated according to equation (22), and the prediction is calculated according to the RobustSLR model(line 4-5). Finally, whether the host is underloaded at the next time unit is returned according to the prediction(line 6).

C. VM SELECTION

One or more VMs should be selected from the overloaded host, and all the VMs should be selected to migrate from the underloaded host. We choose the Minimum Migration Time (MMT) policy [5] that is widely used for VM selection in our research. The MMT policy chooses a VM which has the minimum migration time from source host to target host. From the formulation of MMT, we can find that the VM that takes the minimum memory is chosen. The overloaded host apply the MMT policie iteratively until it is not overloaded.

D. VM PLACEMENT

Beloglazov and Buyya [5], [29] proposed a Power Aware Best Fit Decreasing (PABFD) algorithm for VM placement, the algorithm sorts all the VMs in decreasing order according to their CPU utilizations firstly, after that it allocates each VM to such a host which has the least power increasing after the VM is migrated to the host. The algorithm is essentially a greedy algorithm, so VMs are often consolidated aggressively. We proposed a new VM placement algorithm by modifying the PABFD algorithm in this paper. We have modified the existing VM placement algorithm by adding the RobustSLR prediction algorithm into the PABFD. Algorithm 4 describes our RobustSLR Power Aware Best Fit Decreasing (RPABFD) algorithm. In our RPABFD algorithm, the future host load state is predicted using our RobustSLR algorithm, the overloaded and underloaded hosts are excluded from the host list for VM placement.

Algorithm 4 RobustSLR Power Aware Best Fit Decreasing (RPABFD)

```
Input: $hostList, vmList$;
Output: $allocation\ of\ VMs$;
1:  $vmList.sortDecreasingUtilization()$;
2:  for $vmList.contain(vm)$ is true do
3:      $min = MAX$;
4:      $alloctedHost = NULL$;
5:      for $hostList.contain(host)$ is true do
6:          if $HOD(host)$ is true then
7:              continue;
8:          end if
9:          if $HUD(host)$ is true then
10:             continue;
11:         end if
12:         if host can allocate resources for vm then
13:             $power = getPower(host, vm)$;
14:             if $power < min$ then
15:                 $alloctedHost = host$;
16:                 $min = power$;
17:             end if
18:         end if
19:         if alloctedHost is not NULL then
20:             $allocation.add(vm, allocatedHost)$;
21:         end if
22:     end for
23: end for
24: return $allocation$;
```

V. SIMULATION RESULTS AND ANALYSIS

A. EXPERIMENT SETUP

We have simulated a data center that comprises 400 HP ProLiant ML110 G4 servers and 400 HP ProLiant ML110 G5 servers using the CloudSim toolkit. The configuration of hosts are given in Table 1. The power consumption characteristics of the servers are shown in Table 2, and the characteristics of the VMs are listed in Table 3.

TABLE 1. Configuration of hosts.

Hosts	CPU type	Frequency(GHz)	Cores	RAM(GB)
HP Proliant G4	Intel Xeon 3040	1.86	2	4
HP Proliant G5	Intel Xeon 3075	2.86	2	4

B. WORKLOAD DATA

We evaluate our prediction Model using random workload and a real world workload:

- Random Workload: The users submit requests for provisioning of 50 heterogeneous VMs to the 50 hosts. Each application has 300 bytes input and 300 bytes output. Each VM runs the application and the CPU utilizations are generated according to a random variable. We run the simulation experiment for 24 hours.
- Real Workload (PlanetLab data): PlanetLab is the monitoring part of the CoMon project. It collected the CPU utilization data every five minutes from thousands of servers located at more than 500 places around the world [31]. We chose three different days of the workload traces in our simulations. Table 4 shows the characteristics of each workload.

C. PERFORMANCE METRICS

We have used several metrics to evaluate the performance of the algorithms. The main metrics are Energy Consumption by physical nodes and SLA Violation.

- Energy Consumption: Here we use the power mode based on the results of real data from the SPECpower benchmark [32]. The mode of energy consumption of the servers that we used in this paper is shown in Table 2. We can see that the energy consumption of the server in sleep state is much less than the sever in active state.
- SLA Violation: When a cloud provider fails to provide service to customers in accordance with service level agreement, a SLA violation will occur. SLAV [5] is an independent metric that can be measured by SLA violation time per active host (SLATAH) and performance degradation due to migration(PDM). The two metrics are independent and have the same effect on SLAV. So the SLAV metirc can be calculated as following:

$$SLAV = SLATAH \times PDM \tag{23}$$

We are going to introduce SLATAH and PDM below.

- SLA violation time per active host (SLATAH): When a host is experiencing the 100% utilization, it can not provide service, so SLATAH can be calculated as follows:

$$SLATAH = \frac{1}{N}\sum_{i=1}^{N}\frac{T_{oi}}{T_{ai}} \tag{24}$$

where N is the number of hosts; T_{oi} is the total time during which the host i is experiencing the 100% utilization; the total time of the host i which in active state is T_{ai}.

- Performance degradation due to migration (PDM): Live migration of VMs has negative impact on application performance. The PDM can be calculated as follows:

$$PDM = \frac{1}{N}\sum_{i=1}^{N}\frac{C_{di}}{C_{ri}} \tag{25}$$

where N is the number of VMs; C_{di} is the performance degradation of the VM i due to migrations, we set it as 10% of the CPU utilization according to [33]; C_{ri} is the total CPU capacity requested by the VM i.

- Average SLA violation: The metric can be calculated as follows:

$$Average\ SLAV = \frac{\sum_{k=1}^{N}(requestedMIPS) - \sum_{k=1}^{N}(allocatedMIPS)}{N} \tag{26}$$

where N is the number of VMs.

TABLE 2. Power consumption by the selected servers at different load levels in Watts.

Server	sleep	0%	10%	20%	30%	40%	50%	60%	70%	80%	90%	100%
HP Proliant G4	10	86	89.4	92.6	96	99.5	102	106	108	112	114	117
HP Proliant G5	10	93.7	97	101	105	110	116	121	125	129	133	135

TABLE 3. Four kinds of VM types.

VM type	CPU(MIPS)	RAM(GB)
High-CPU medium instance	2500	0.85
large instance	2000	1.70
Small instance	1000	1.70
Micro instance	500	0.61

TABLE 4. Workload data characteristics(CPU utilization).

Date	Hosts	VMs	Mean	St.dev.
03/03/2011	800	1052	12.31%	17.09%
22/03/2011	800	1516	9.26%	12.78%
20/04/2011	800	1033	10.43%	15.21%
random	50	50	—	—

- Overall SLA violation: The metric can be calculated as follows:

$$Overall\ SLAV = \frac{\sum_{k=1}^{N}(requestedMIPS) - \sum_{k=1}^{N}(allocatedMIPS)}{\sum_{k=1}^{N}(requestedMIPS)} \tag{27}$$

where N is the number of VMs.
- Number of VM migrations: The metric can be calculated as follows:

$$Migrations(F, t_1, t_2) = \sum_{i=1}^{N}\int_{t_1}^{t_2} Mig_i(F) \tag{28}$$

where N is the number of hosts, F is the placement of VMs at host i, $Mig_i(F)$ is the number of migration of host i from time t_1 to time t_2.
- Number of host shutdowns: The metric can be calculated as follows:

$$H = \frac{1}{n}\sum_{i=1}^{n} h_i \tag{29}$$

where h_i is the number of active hosts at the time i. H is the number of host shutdowns from time 1 to time n.

D. ANALYSIS OF SAFETY PARAMETER S

Using the workload data described above, we have simulated all combinations of the eight proposed host detection algorithms (MSE, RMSE, MAE(3), MAE(10), EWMAE(3), EWMAE(10), $se(b_1)$ and 95%CI) and the MMT VM selection algorithms. Moreover, because the safety parameter s in equation (22) is a key parameter for our model, so we first choose the best s for each combination through experiments. For each detection algorithm we have varied the parameters as follows: for MSE from 0.5 to 4.0 increased by 0.5, and for the others from 0.5 to 3.0 increased by 0.5. Figs.1 show the energy consumption, SLA violations under different value of s using the "20110303", "20110322" and "20110420" data sets.

Fig. 1 show that the lower s, the less the energy consumption, but the higher the level of SLA violations. Therefore, we have to choose the appropriate safety parameter s for dealing with the energy-performance tradeoff - minimizing energy consumption on the premise of meeting SLA. According to the experimental results, we set $s = 0.5$ for 95%CI-MMT-s algorithm, $s = 1.0$ for $se(b_1)$-MMT-s algorithm, $s = 1.5$ for RMSE-MMT-s algorithm, $s = 2.0$ for MAE(3)-MMT-s algorithm, $s = 2.5$ for MAE(10)-MMT-s, EWMAE(3)-MMT-s and EWMAE(10)-MMT-s algorithms, and $s = 4.0$ for MSE-MMT-s algorithm respectively.

we also do some experiments using the random workload, and get the similar results with the real workloads. Figs. 2-4 illustrate the energy consumption, SLA violations, number of VM migrations under different value of s for our eight proposed algorithms respectively.

E. COMPARISON WITH OTHER BENCHMARKS

We are interested in comparing our proposed model to the state-of-the-art algorithms using the PlanetLab data described above. We select "20110303" data set as the workload and our proposed eight algorithms with the appropriate safety parameter s which are MSE-MMT-4.0, RMSE-MMT-1.5, MAE(3)-MMT-2.0, MAE(10)-MMT-2.5, EWMAE(3)-MMT-2.5, EWMAE(10)-MMT-2.5, $se(b_1)$-MMT-1.0, and 95%CI-MMT-0.5 are chosen for comparison. The comparison benchamrks are NPA(None Power Aware) [8], DVFS [9], THR-MMT-1.0 [5], THR-MMT-0.8 [5], IQR-MMT-1.5 [5], MAD-MMT-2.5 [5], LR-MMT-1.2 [5], and LRR-MMT-1.2 [5]. The hosts which use the NPA policy consume their maximum power all the time. The THR-MMT-1.0 algorithm uses the fixed threshold of 100%. Table 5 show the details of experimental results, and Figs. 5-7 illustrate the Energy consumption, SLA violations, number of VM migrations for the main algorithms respectively.

From the simulation results, we have got the following conclusions: (1) VM consolidation technique significantly surpasses NPA and DVFS; (2) because of reducing the level of SLA violations observably, dynamic threshold-based heuristics algorithms considerably outperform the static threshold-based heuristics algorithm(THR-MMT-1.0), but there is no statistically significant difference with the

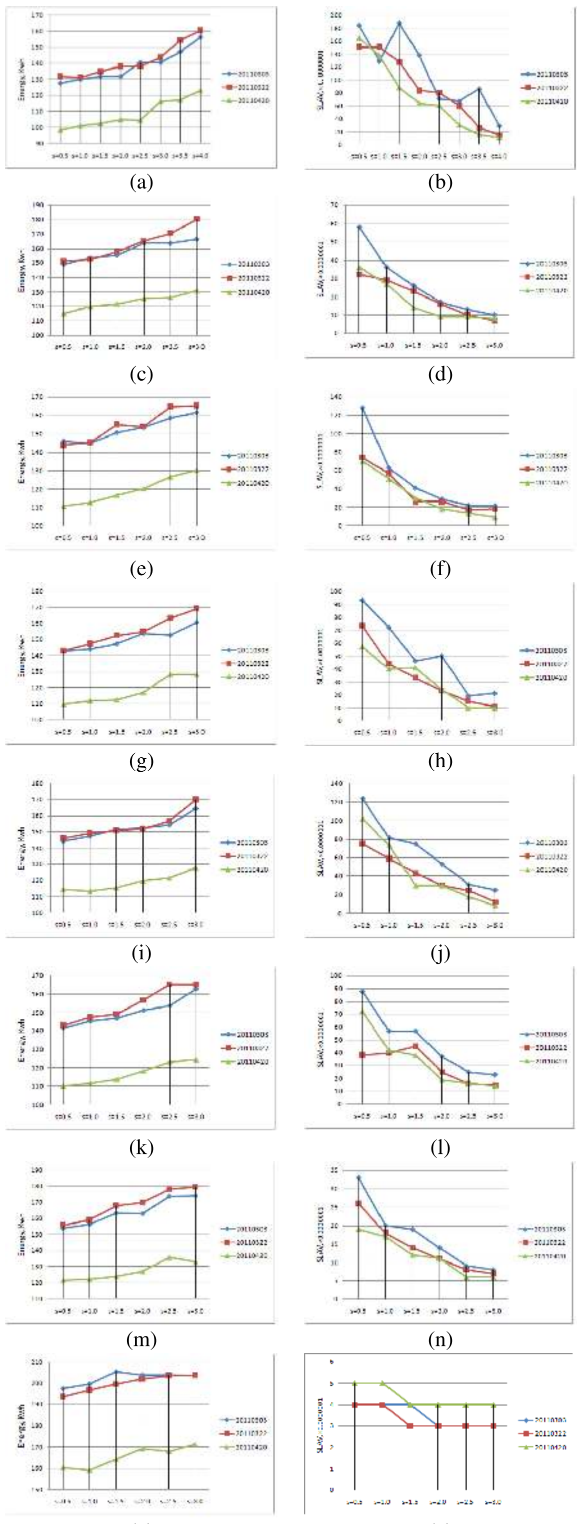

FIGURE 1. Energy consumption and SLAV under different value of *s* for real workload. (a) MSE-Energy. (b) MSE-SLAV. (c) RMSE-Energy. (d) RMSE-SLAV. (e) MAE(3)-Energy. (f) MAE(3)-SLAV. (g) MAE(10)-Energy. (h) MAE(10)-SLAV. (i) EWMAE(3)-Energy. (j) EWMAE(3)-SLAV. (k) EWMAE(10)-Energy. (l) EWMAE(10)-SLAV. (m) *se*(b_1)-Energy. (n) *se*(b_1)-SLAV. (o) 95%CI-Energy. (p) 95%CI-SLAV.

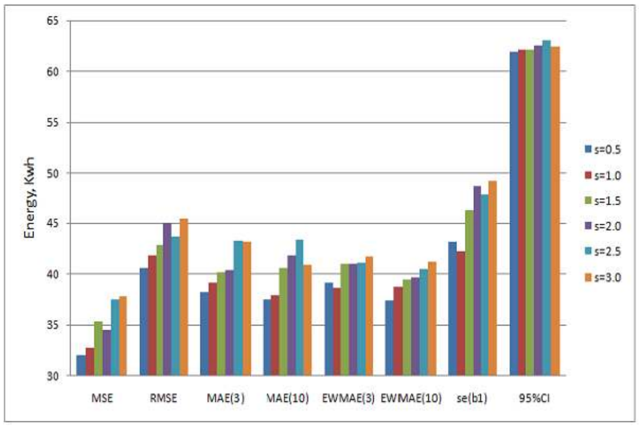

FIGURE 2. Energy consumption under different value of *s* for random workload.

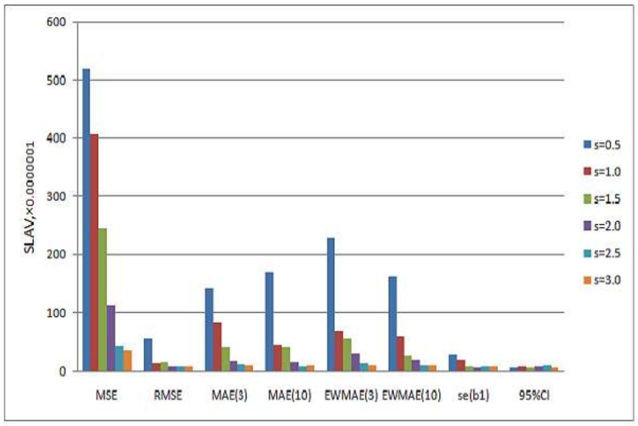

FIGURE 3. SLAV under different value of *s* for random workload.

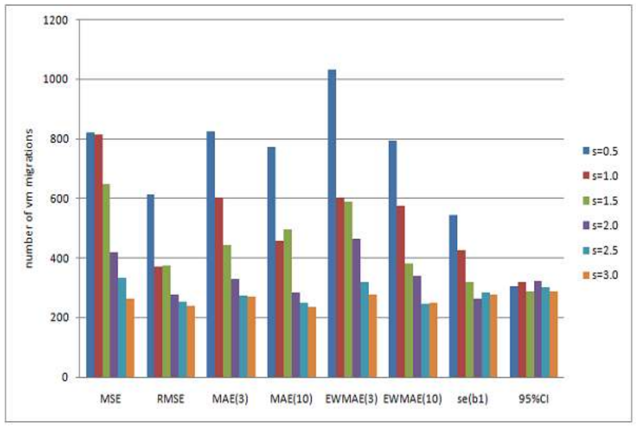

FIGURE 4. Number of VM migrations under different value of *s* for random workload.

static threshold-based heuristics algorithm with the threshold set to 80%(THR-MMT-0.8); (3) our proposed eight decision-making algorithms can significantly outperform the benchmarks of one static threshold-based heuristics algorithm with the threshold set to 80% and four dynamic VM consolidation policies including two dynamic threshold-based heuristics algorithms and two decision-making algorithms, they can reduce the metric of Energy consumption by at most 25.43%, average 17.10% , and they can reduce the

TABLE 5. Simulation results of the best algorithm combinations and benchmark algorithms.

Police	Energy (KWh)	SLAV ($\times 10^{-7}$)	VM migr.	SLATAH	PDM	Average SLAV	Overall SLAV	Host shutd.
NPA	2410.8	0	0	0%	0%	0%	0%	466
DVFS	787.84	0	0	0%	0%	0%	0%	466
THR-MMT-1.0	173.24	3088	48335	16.43%	0.19%	9.19%	0.58%	6491
THR-MMT-0.8	205.97	354	28843	4.97%	0.07%	10.11%	0.08%	6395
IQR-MMT-1.5	201.92	340	28350	4.91%	0.07%	10.1%	0.08%	6301
MAD-MMT-2.5	198.16	345	28162	4.93%	0.07%	10.13%	0.08%	6232
LR-MMT-1.2	176.15	478	28615	5.87%	0.08%	9.67%	0.15%	5483
LRR-MMT-1.2	176.15	478	28615	5.87%	0.08%	9.67%	0.15%	5483
MSE-MMT-4.0	156.34	29	3415	2.87%	0.01%	8.11%	0.34%	1055
RMSE-MMT-1.5	155.29	26	5714	2.5%	0.01%	8.24%	0.08%	1548
MAE(3)-MMT-2.0	153.49	29	5131	2.93%	0.01%	7.73%	0.17%	1406
MAE(10)-MMT-2.5	152.66	19	3902	2.06%	0.01%	8.11%	0.11%	1165
EWMAE(3)-MMT-2.5	154.15	31	5358	2.73%	0.01%	7.99%	0.08%	1428
EWMAE(10)-MMT-2.5	153.6	25	4731	2.43%	0.01%	7.92%	0.1%	1304
$se(b_1)$-MMT-1.0	156.31	20	5595	1.9%	0.01%	8.85%	0.04%	1561
95%CI-MMT-0.5	197.45	4	2854	0.59%	0.01%	10.08%	0.01%	1032

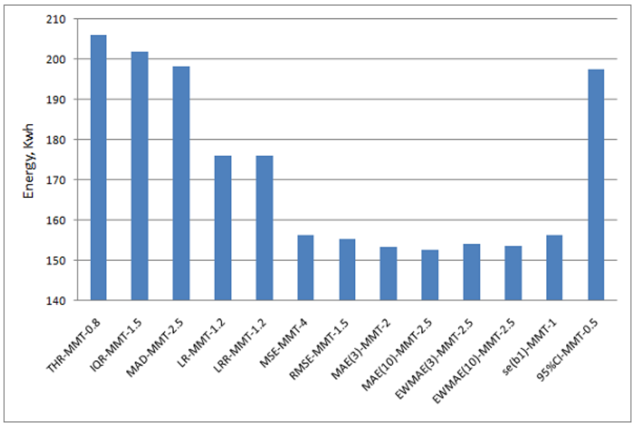

FIGURE 5. Energy consumption for the main algorithms.

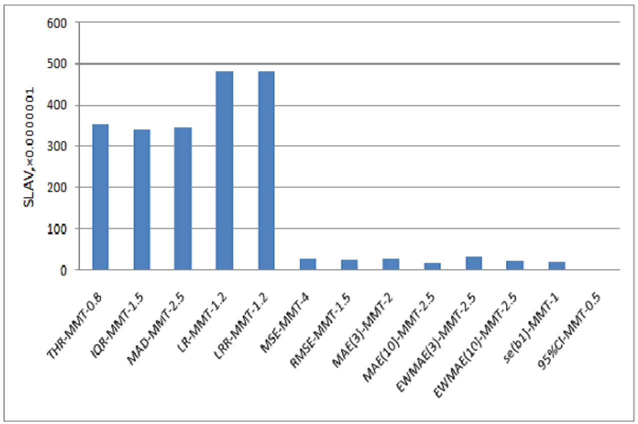

FIGURE 6. SLAV for the main algorithms.

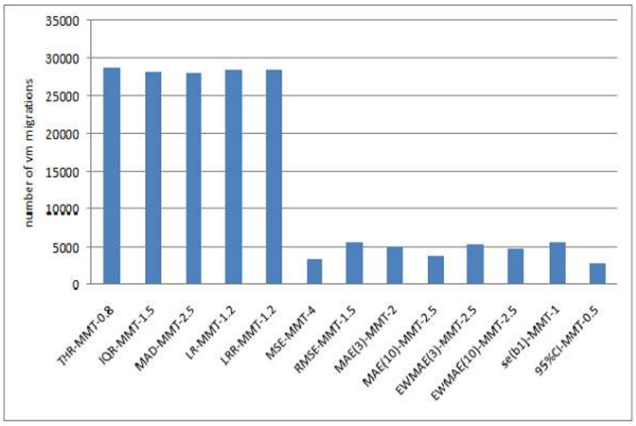

FIGURE 7. Number of VM migrations for the main algorithms.

metric of SLAV by at most 99.16%, average 94.27% , and they can reduce the metric of number of VM migrations by at most 90.11%, average 83.91%. (4) MSE-MMT-4.0, RMSE-MMT-1.5, MAE(3)-MMT-2.0, MAE(10)-MMT-2.5, EWMAE(3)-MMT-2.5, EWMAE(10)-MMT-2.5, and $se(b_1)$-MMT-1.0 algorithms are almost at the same level performance, but the MAE(10)-MMT-2.5 algorithm gets the best tradeoff between the Energy consumption and SLA violations; (5) 95%CI-MMT-0.5 algorithm consumes almost the same energy with the benchmarks dynamic threshold-based heuristics algorithms, but it gets the best SLA performance, it reduces SLA violations to almost 0.

VI. CONCLUDING REMARKS AND FUTURE DIRECTIONS

In this paper, we have proposed a RobustSLR model to predict the host future CPU utilization, and our model amends the prediction and squint towards over-prediction by adding the error directly or indirectly to the prediction. We have proposed eight methods to calculate the error. We have proposed a Host Overloading/Underloading Detection algorithm based on our proposed RobustSLR model in order to minimize the power consumption and SLA violation. We also have proposed a new VM placement algorithm which called RobustSLR Power Aware Best Fit Decreasing algorithm in our model.

We have extended the Cloudsim simulator for performance evaluation of our proposed algorithms using PlanetLab workload and random workload. The results of the experiments have shown that our proposed eight decision-making algorithms can significantly outperform the benchmarks of one static threshold-based heuristics algorithm with the threshold set to 80% and four dynamic VM consolidation policies including two dynamic threshold-based heuristics algorithms and two decision-making algorithms, they can reduce Energy consumption by at most 25.43% and SLA violations by at most 99.16% for the PlanetLab workload. As a future work, we plan to optimal our model to further improve energy efficient and reduce SLA violations considering the other resources, for example, the RAM and net.

REFERENCES

[1] (2011). *The Green Grid Consortium*. [Online]. Available: http://www.thegreengrid.org

[2] L. A. Barroso and U. Holzle, "The case for energy-proportional computing," *Computer*, vol. 40, no. 12, pp. 33–37, Dec. 2007.

[3] M. D. de Assunção, J.-P. Gelas, L. Lefèvre, and A.-C. Orgerie, "The green Grid'5000: Instrumenting and using a grid with energy sensors," in *Proc. 5th Int. Workshop Distrib. Cooper. Lab., Instrum. Grid (INGRID)*, Poznan, Poland, 2010, pp. 25–42.

[4] C. Clark *et al.*, "Live migration of virtual machines," in *Proc. 2nd Symp. Netw. Syst. Design Implement. (NSDI)*, Boston, MA, USA, 2005, pp. 273–286.

[5] 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, 2012.

[6] X. Fan, W.-D. Weber, and L.-A. Barroso, "Power provisioning for a warehouse-sized computer," in *Proc. 34th Annu. Int. Symp. Comput. Archit. (ISCA)*, New York, NY, USA, 2007, pp. 13–23.

[7] D. Kusic, J. O. Kephart, J. E. Hanson, N. Kandasamy, and G. Jiang, "Power and performance management of virtualized computing environments via lookahead control," *Cluster Comput.*, vol. 12, no. 1, pp. 1–15, Mar. 2009.

[8] 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.

[9] H. Hanson, S. W. Keckler, S. Ghiasi, K. Rajamani, F. Rawson, and J. Rubio, "Thermal response to DVFS: Analysis with an Intel Pentium M," in *Proc. Int. Symp. Low Power Electron. Design (ISLPED)*, vol. 7, Aug. 2007, pp. 219–224.

[10] C.-M. Wu, R.-S. Chang, and H.-Y. Chan, "A green energy-efficient scheduling algorithm using the DVFS technique for cloud datacenters," *Future Generat. Comput. Syst.*, vol. 37, pp. 141–147, Jul. 2014.

[11] A. Wierman, L. L. H. Andrew, and A. Tang, "Power-aware speed scaling in processor sharing systems," in *Proc. 28th Conf. Comput. Commun. (INFOCOM)*, Apr. 2009, pp. 2007–2015.

[12] L. L. H. Andrew, M. Lin, and A. Wierman, "Optimality, fairness, and robustness in speed scaling designs," in *Proc. ACM Int. Conf. Meas. Modeling Comput. Syst. (SIGMETRICS)*, 2010, pp. 37–48.

[13] K. Flautner, S. Reinhardt, and T. Mudge, "Automatic performance setting for dynamic voltage scaling," *Wireless Netw.*, vol. 8, no. 5, pp. 507–520, 2002.

[14] A. Weissel and F. Bellosa, "Process cruise control-event-driven clock scaling for dynamic power management," in *Proc. Int. Conf. Compil., Archit., Synth. Embedded Syst.*, 2002, pp. 238–246.

[15] S. Lee and T. Sakurai, "Run-time voltage hopping for low-power real-time systems," *Proc. 37th Annu. ACM/IEEE Design Autom. Conf. (DAC)*, Jun. 2000, pp. 806–809.

[16] J. R. Lorch and A. J. Smith, "Improving dynamic voltage scaling algorithms with PACE," *ACM SIGMETRICS Perform. Eval. Rev.*, vol. 29, no. 1, pp. 50–61, Jun. 2001.

[17] E. Feller, C. Morin, and A. Esnault, "A case for fully decentralized dynamic VM consolidation in clouds," in *Proc. IEEE 4th Int. Conf. Cloud Comput. Technol. Sci.*, Dec. 2012, pp. 26–33.

[18] T. Wood, P. Shenoy, A. Venkataramani, and M. Yousif, "Sandpiper: Black-box and gray-box resource management for virtual machines," *Comput. Netw.*, vol. 53, no. 17, pp. 2923–2938, Dec. 2009.

[19] F. Ahamed, S. Shahrestani, and B. Javadi, "Security aware and energy-efficient virtual machine consolidation in cloud computing systems," in *Proc. IEEE Trustcom/BigDataSE/ISPA*, Aug. 2016, pp. 1516–1523.

[20] 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.

[21] A. Beloglazov, "Energy-efficient management of virtual machines in data centers for cloud computing," Ph.D. dissertation, Dept. Comput. Inf. Syst., Univ. Melbourne, Parkville, VIC, Australia, 2013.

[22] D. Grimes *et al.*, "Robust server consolidation: Coping with peak demand underestimation," in *Proc. IEEE 24th Int. Symp. Modeling, Anal. Simulation Comput. Telecommun. Syst. (MASCOTS)*, Sep. 2016, pp. 271–276.

[23] 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. Softw. Eng. Adv. Appl.*, Sep. 2013, pp. 357–364.

[24] B. Guenter, N. Jain, and C. Williams, "Managing cost performance and reliability tradeoffs for energy-aware server provisioning," in *Proc. IEEE INFOCOM*, Apr. 2011, pp. 1332–1340.

[25] J. Xue, F. Yan, R. Birke, L. Y. Chen, T. Scherer, and E. Smirni, "PRACTISE: Robust prediction of data center time series," in *Proc. 11th Int. Conf. Netw. Service Management (CNSM)*, Nov. 2015, pp. 126–134.

[26] B. George, *Time Series Analysis: Forecasting and Control*, 3rd ed. New Delhi, India: Pearson Education, 1994.

[27] E. Feller, L. Rilling, and C. Morin, "Energy-aware ant colony based workload placement in clouds," in *Proc. 12th IEEE/ACM Int. Conf. Grid Comput. (GRID)*, Lyon, France, Sep. 2011, pp. 26–33.

[28] F. Ma, F. Liu, and Z. Liu, "Multi-objective optimization for initial virtual machine placement in cloud data center," *J. Inf. Comput. Sci.*, vol. 9, no. 16, pp. 5029–5038, 2012.

[29] A. Beloglazov and R. Buyya, "Managing overloaded hosts for dynamic consolidation of virtual machines in cloud data centers under quality of service constraints," *IEEE Trans. Parallel Distrib. Syst.*, vol. 24, no. 7, pp. 1366–1379, Jul. 2013.

[30] *STAT 501: Simple Linear Regression*. Accessed: 2018. [Online]. Available: https://onlinecourses.science.psu.edu/stat501/node/250

[31] 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.

[32] *The SPECpower Benchmark*. Accessed: 2018. [Online]. Available: http://www.spec.org/power/ssj2008/

[33] C. L. Dumitrescu and I. Foster, "GangSim: A simulator for grid scheduling studies," in *Proc. IEEE Int. Symp. Cluster Comput. Grid (CCGrid)*, vol. 2, May 2005, pp. 1151–1158.

LIANPENG LI received the B.S. and M.S. degrees from the Harbin Institute of Technology, in 2006 and 2010, respectively, where he is currently pursuing the Ph.D. degree in computer science. His research interests include parallel computation, cloud computing, and fault tolerant computers.

JIAN DONG received the Ph.D. degree in computer science and technology from the Harbin Institute of Technology, in 2008, where he was an Associate Professor with the School of Computer Science and Technology, from 2012 to 2017, and has been a Professor with the School of Computer Science and Technology, since 2017. His research interests include fault tolerant computers, parallel computation, and cloud computing.

DECHENG ZUO received the Ph.D. degree in computer science and technology from the Harbin Institute of Technology, in 2000, where he was an Associate Professor with the School of Computer Science and Technology, from 2003 to 2007, and has been a Professor with the School of Computer Science and Technology, since 2007. His research interests include computer architecture, fault tolerant computing, computer systems performance evaluation technology, wireless sensor networks, wearable computing, and mobile computing.

JIN WU received the bachelor's degree in computer science and the master's degree in software engineering from Nanjing University. He is currently pursuing the Ph.D. degree with the School of Computer Science and Technology, Harbin Institute of Technology. His research interests include mobile computing, cross-ISA virtualization, and cloud computing.