Computer Science and Information Systems 17(1): 29–50 https://doi.org/10.2298/CSIS181122036A

# Efficient Virtual Machine Placement Algorithms for Consolidation in Cloud Data Centers

Loiy Alsbatin[1,2], Gürcü Öz[1], and Ali Hakan Ulusoy[3]

[1] Department of Computer Engineering, Faculty of Engineering, Eastern Mediterranean University,
Famagusta, North Cyprus via Mersin 10 Turkey,
loiy.alsbatin@gmail.com, gurcu.oz@emu.edu.tr
[2] Department of Computer Science, Collage of Computing and Information Technology, Shaqra University,
Riyadh, Saudi Arabia
[3] Department of Information Technology, School of Computing and Technology, Eastern Mediterranean University,
Famagusta, North Cyprus via Mersin 10 Turkey,
alihakan.ulusoy@emu.edu.tr

**Abstract.** Dynamic Virtual Machine (VM) consolidation is a successful approach to improve the energy efficiency and the resource utilization in cloud environments. Consequently, optimizing the online energy-performance tradeoff directly influences quality of service. In this study, algorithms named as CPU Priority based Best-Fit Decreasing (CPBFD) and Dynamic CPU Priority based Best-Fit Decreasing (DCPBFD) are proposed for VM placement. A number of VM placement algorithms are implemented and compared with the proposed algorithms. The algorithms are evaluated through simulations with real-world workload traces and it is shown that the proposed algorithms outperform the known algorithms. The simulation results clearly show that CPBFD and DCPBFD provide the least service level agreement violations, least VM migrations, and efficient energy consumption.

**Keywords:** Cloud computing, energy consumption, dynamic consolidation, virtualization

## 1. Introduction

Dynamic Virtual Machine (VM) consolidation effectively improves the energy efficiency and resource utilization in data centers. Reallocating VMs from an overloaded Physical Machine (PM) maximizes the utilization and energy efficiency with providing a high Quality of Service (QoS). The goal of consolidation of VMs ensures an efficient utilization that can be achieved through the use of VM migrations across different PMs. Power consumption of USA data centers has increased by 62.5% from 2005 to 2013 and expected to increase by 150% in 2020 [1]. Most of the energy consumption of data centers is consumed by computing resources. Accordingly, resource management is important to ensure that the applications efficiently utilize the available computing resources. Switching the idle nodes to sleep mode to eliminate the idle power consumption can achieve a reduction in energy consumption.

One efficient way to improve the utilization of cloud data center resources is the dynamic consolidation of VMs [2-11]. The dynamic consolidation reallocates VMs periodically using migration to reduce the number of active PMs required to handle requests. The objective of this approach is mainly to minimize energy consumption and maximize of QoS provided by the system.

It is complex to solve dynamic VM consolidation problem analytically as a whole [3, 4]. In general, the problem can be decomposed into tasks as following [8]:

1. PM underload detection: This is the phase when a PM is considered as being underloaded, so all VMs running on an underloaded PM should be migrated to other PMs and the underloaded PM should be switched to the sleep mode (to reduce the number of active PMs).
2. PM overload detection: This is the phase when a PM is considered as being overloaded, so some VMs running on an overloaded PM should be migrated to another active PM (to avoid violation QoS requirements).
3. VM selection: This is the phase to select VMs to be migrated from the overloaded PM.
4. VM placement: This is the phase to place selected VMs for migration on another active PM.

In this study, we mainly focus on VM placement problem. Algorithms named as CPU Priority based Best-Fit Decreasing (CPBFD) and Dynamic CPU Priority based Best-Fit Decreasing (DCPBFD) are proposed for VM placement. We implemented a number of placement algorithms to compare with the proposed algorithms using real workload traces.

The rest of the paper is organized as follows. The related work and the system model are discussed in Sections 2 and 3, respectively. The metrics used to show the performance of the algorithms are described in Section 4. The proposed VM placement algorithms are presented in Section 5. In Section 6, the experimental setup, evaluations and results are discussed. Finally, we conclude the results and discuss the future work in Section 7.

## 2. Related Work

The two main types of energy efficient resource management algorithms in the cloud data centers are constraint energy consumption algorithm [12, 13] and energy efficiency (energy consumption and Service Level Agreement (SLA) violation) algorithm [7, 8, 14, 15]. The constraint energy consumption algorithm aims to minimize the energy consumption, but this type of algorithm does not consider SLA violation at all or focuses a little on it. Therefore, this type of algorithm does not meet the requirements of users. For example, two heuristic algorithms are proposed by Lee and Zomaya [12] to reduce the energy consumption, but the algorithms do not consider SLA violation. Similarly, Kang and Ranka [13] proposed an energy-saving algorithm, but it also does not consider SLA violation. The main goal of the energy efficiency algorithm is to reduce the energy consumption and SLA violation in data centers. Several VM placement algorithms are proposed in [7, 8, 14, 15]. These algorithms reduce SLA violation and save energy consumption, but SLA violation remains at a high level. In our previous study [16], we proposed dynamic VM consolidation based on a PM overload detection algorithm and a combination of PM overload detection algorithm and VM quiescing to minimize number of VM migrations according to QoS requirements. The goal of the model is to improve utilization of resources, SLA, and energy efficiency in cloud data centers. However, this model does not focus on energy consumption.

In the past few years, many approaches to the dynamic consolidation of VMs have been proposed [2-11]. Comparative studies of various existing consolidation of VM algorithms using real-world workload traces were presented. Some of VM consolidation algorithms based on different heuristics on the legitimate PM were analyzed in [17]. A scheduling algorithm to assign VMs to PMs in a data center was proposed in [18]. The goal was to improve energy efficiency by taking into consideration the conflicts between the costs of VM migration and CPU and disk utilizations. Four models named as the migration model, the energy model, the application model, and the target system model were presented to identify the conflicts.

An adaptive threshold-based algorithm was proposed by Deng et al. in [19]. The overload threshold of CPU utilization and the average utilization of active PMs were used for PM underload detection algorithm, and minimum average utilization difference of the data center was used for VM placement algorithm. Several dynamic VM consolidation algorithms were proposed by Khoshkholghi et al. in [20] to improve the utilization, energy consumption and SLA violations based on the CPU, RAM and bandwidth. They used an iterative weighted linear regression method for PM overload detection and a vector magnitude squared of resources for PM underload detection. They also proposed SLA and power-aware VM selection algorithm and VM placement algorithm. PM overload and underload detection algorithms and VM placement algorithm based on dynamic thresholds and probable future load were proposed by Shaw et al. in [21]. They used simple exponential smoothing technique to predict CPU utilization and calculate dynamic upper and lower utilization thresholds. A VM consolidation algorithm with utilization prediction of multiple resource types based on the local history of PMs was proposed by Nguyen et al. in [22] to improve the energy efficiency of cloud data centers. Two adaptive energy-aware algorithms for minimizing SLA violation and maximizing energy efficiency in cloud data centers were proposed by Zhou et al. in [23]. CPU, memory resources and application types were considered during the deployment of VMs.

Managing resource allocation to improve response time using control loops at the server and cluster levels were applied in [24]. The server migrated a VM if the server's resource capacity was not enough to meet SLA of application. An adaptive heuristics energy-aware algorithm that used an upper threshold of CPU utilization for PM overload detection and dynamic VM selection algorithms was proposed in [25]. A greedy consolidation algorithm based on VM placement algorithm was proposed in [26] to improve the network usage and performance of applications in the data centers. The greedy consolidation algorithm reduced the number of migrations and speed up the placement decisions. In [27], two algorithms which could be used together for live migration of multiple VMs were proposed. The proposed VM migration depended on three factors that were the cost of migration, the expected distribution of workload and the state of PM after migration. The algorithms distributed the workload efficiently in the system. In spite of that, the research did not discuss how to meet SLA. In [3], the problem of dynamic VM placement was solved by a heuristic bin packing algorithm. However, SLA cannot be met because of unforeseeable workloads and instability.

A dynamic consolidation of VMs for web applications was implemented in [10]. In this study, the response time was used to define SLA. Weighted linear regression was applied to get the future workload and improve the distribution of workload. VM consolidation algorithms under QoS expectations were evaluated using the CloudSim toolkit showing high improvement of cost savings and energy efficiency using dynamic workload scenarios [7, 8]. They proposed maximum correlation, random selection and Minimum Migration Time (MMT) policies for VM selection from the overloaded PM and utilized interquartile range, median absolute deviation, robust local regression and Local Regression (LR) algorithms for PM overload detection. Simple Method (SM) was used to find underloaded PM which was with the least resource utilization. For VM placement, they proposed Power-Aware Best-Fit Decreasing (PABFD) algorithm, which based on sorting VMs by CPU utilization in decreasing order and placing a VM in PM that will have the minimum expected increasing in power consumption. The results showed that the combination of LR for PM overload detection and MMT for VM selection had better performance in the number of VM migrations, energy consumption, and SLA violations.

In this research, LR method was used for PM overload detection, SM policy was used for PM underload detection, MMT policy was used for VM selection, and for comparison purposes with proposed CPBFD and DCPBFD VM placement algorithms we used PABFD, First-Fit Decreasing (FFD) [28-30] and Best-Fit Decreasing (BFD) [29, 30] algorithms which are well-known algorithms for bin-packing problem. VM placement algorithm based on FFD sorts VMs by CPU utilization in decreasing order and places a VM in the first PM that will fit it [30]. VM placement algorithm based on BFD sorts VMs by CPU utilization in decreasing order and places a VM in PM that will have the maximum CPU utilization after allocating VM [30].

## 3. System Model

We use the system model presented in [8] to evaluate VM placement algorithms by using the CloudSim [31] toolkit. The system consists of *X* heterogeneous PMs in a large-scale data center. Characteristics of each PM are defined by CPU performance denoted by Random Access Memory (RAM), Millions Instructions Per Second (MIPS), and network bandwidth. The storage of servers for VM live migration is network attached storage. Multiple independent users request for supplying *Y* VMs characterized by requirements denoted by RAM, MIPS and network bandwidth.

As shown in Fig. 1, the system includes global and local managers. The local manager on each PM monitors CPU utilization of PM using VM Monitor (VMM). VMM decides when and which VMs should be migrated to other PMs. The global manager which acts as the controller in the system collects information of the utilization of PMs from the local managers and decides VM placement, and VMMs migrate VMs and change the power mode of PMs.

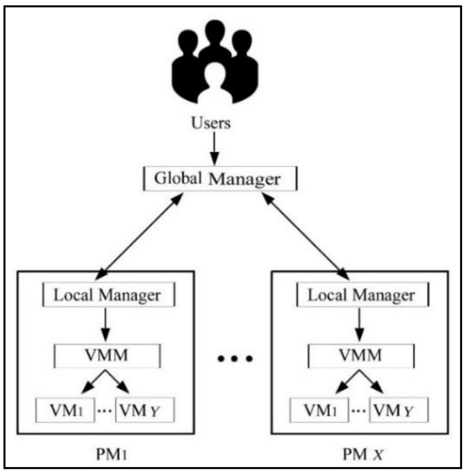

**Fig. 1.** The system model [8]

## 4. Metrices

### 4.1. Power Model

In the data centers, power consumption of PMs is usually defined by CPU, cooling systems, power supplies, memory and disk storage [32]. The power consumption of PMs can be defined using linear relationship of CPU utilization even if dynamic voltage and frequency scaling is used [7, 33, 34]. Due to the fact that the limited number of the frequency and voltage states of a CPU and other system components, such as network interfaces and memory, the voltage and frequency scaling are not used. Since analytical models of power consumption is a complex research problem for modern multi-core CPUs [8], real power consumption benchmark results provided by the SPECpower [35] are used.

The work in [7, 36] shows that a PM when it is idle uses approximately 70% of its maximum energy consumption. As presented in [7, 36], the power consumption of PM can be defined as

$$P(u) = 0.7 \times P_{max} + 0.3 \times P_{max} \times u \qquad (1)$$

where $P_{max}$ is the maximum power of a fully utilized PM and is set to 250 W as presented in [33, 34]. $u$ is CPU utilization. Since CPU utilization changes over time, it is presented as a function of time as $u(t)$. As presented in [7], energy consumption can be obtained as

$$\mathrm{E} = \int_t P(u(t))dt. \qquad (2)$$

Since the energy consumption of a PM is determined by CPU utilization, we take CPU utilization into consideration in the proposed VM placement algorithms to reduce the energy consumption in the system [6].

## 4.2. Cost of Live Migration of VMs

Live migration of VMs transfers VMs between PMs without suspension. However, the large number of live VM migrations may drop the performance of applications. So, the number of VM migrations should be reduced. The behavior of applications causes performance degradation. We use cost model for VM migration presented in [6] to avoid performance degradation. The authors stated in [6] that CPU utilization can be increased by 10% for each VM migration. So, each VM migration can cause SLA violations. Therefore, VM migrations should be reduced, and VM with minimum memory should be selected.

## 4.3. SLA Violation Metrics

In cloud computing environments, it is highly important to meet QoS requirements. QoS is usually defined in the form of SLA that is defined through some characteristics such as maximum response time or minimum throughput of the system [8]. To evaluate QoS requirements, SLA metric is defined as a workload independent metric for any loads in Infrastructure as a Service (IaaS). Two metrics are used to measure SLA violations: The fraction of time when CPU utilization of PM has been 100%, SLA violation Time per Active Host/PM (SLATAH) as shown in (3); and Performance Degradation due to Migrations (PDM) as shown in (4) [8],

$$\mathrm{SLATAH} = \sum_{i=1}^{X} \frac{T_{s_i}}{T_{a_i}} \tag{3}$$

$$\mathrm{PDM} = \frac{1}{Y} \sum_{j=1}^{Y} \frac{C_{d_j}}{C_{r_j}} \tag{4}$$

where *X* represents the number of PMs, $T_{s_i}$ is the time when the utilization of *i*-th PM is 100% which leads to an SLA violation, $T_{a_i}$ is the time when i-th PM is active, *Y* represents the number of VMs, $C_{d_j}$ is the estimated performance degradation caused by *j*-th VM migrations, $C_{r_j}$ is the total CPU capacity requested by j-th VM. $C_{d_j}$ is estimated to be 10% of CPU utilization in MIPS during the j-th VM migrations [8].

The level of SLA violations is independently characterized by both SLATAH and PDM metrics. So, we use a metric presented in [8] that includes performance degradation caused by both overloaded PM and VM migrations, denoted as SLA Violation (SLAV) that is calculated as

$$\mathrm{SLAV} = \mathrm{SLATAH} \times \mathrm{PDM}. \tag{5}$$

VM consolidation objective is to reduce both energy consumption (E) and SLAV. ESV metric presented in [8] that equals the product of energy consumption and SLA violations is used as

$$\mathrm{ESV} = \mathrm{E} \times \mathrm{SLAV}. \tag{6}$$

SLAV and energy consumption are the most important metrics that should be minimized to improve efficiency of resources [7, 8]. SLA violation has a negative relation with the energy consumption in the cloud data center [37]. Therefore, ESV is used for performance evaluation of all algorithms to show the trade-off between energy consumption and SLA violation [7, 8, 37].

## 5. VM Placement

We propose novel VM placement algorithms based on giving priority of PM with highest CPU utilization between two sided limits, then giving priority to PM with CPU utilization outside the two-sided limits and nearest to the two-sided limits. Second priority is given to PMs with CPU utilization outside the two-sided limits, since selecting PMs with low load or high load leads to less active PMs and less energy consumption than waking up PMs from sleep mode. We modified well-known BFD algorithm [29] to be suitable for VM placement by limiting the upper CPU utilization threshold [38-42] and lower CPU utilization threshold and implementing the abovementioned priority. We denote proposed algorithm that used static upper and lower CPU utilization thresholds as CPBFD, and proposed algorithm that used dynamic upper and lower CPU utilization thresholds as DCPBFD. Not setting an upper limit for the CPU utilization of allocated PMs may cause frequent overloading of allocated PMs, which leads to performance degradation and increases the number of VM migrations. We propose to limit the upper threshold of CPU utilization of allocated PM to avoid performance degradation caused by VM migrations to PM with high load and to minimize the number of VM migrations. Furthermore, not setting a lower limit for CPU utilization of allocated PMs may cause allocating underloaded PMs, which leads to more active PMs and more energy consumption. We propose to limit the lower threshold of CPU utilization of allocated PM to improve energy consumption.

### 5.1. CPBFD Algorithm

CPU resource utilization model of PM of CPBFD algorithm is shown in Fig. 2. In CPBFD, all VMs are sorted in the decreasing order of their current CPU utilizations and allocate each VM to a PM with maximum CPU utilization less than upper CPU utilization threshold *a* and more than lower CPU utilization threshold *b*. If there are no PMs with CPU utilization between upper and lower threshold, a PM with the nearest CPU utilization to upper or lower thresholds will be allocated. The priority may be given to PM with CPU utilization nearest to upper threshold or lower threshold by adjusting parameter *c*. PM with CPU utilization that nearest to *c* and outside of *a* to *b* range will be selected.

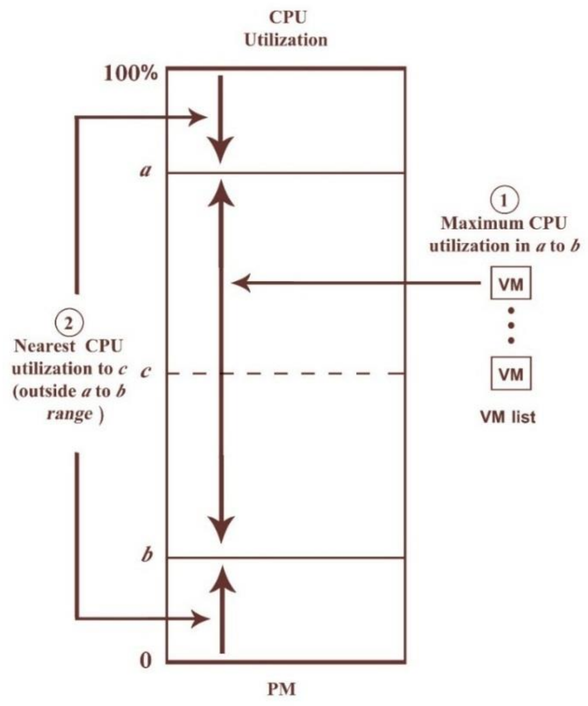
CPU Utilization
100%
a
c
b
0
PM
1
Maximum CPU utilization in a to b
VM
VM
VM list
2
Nearest CPU utilization to c (outside a to b range )

**Fig. 2.** CPU utilization model of PM of CPBFD and DCPBFD algorithm

There is no specific optimal upper CPU threshold value among the researchers [38-42]. Selecting high upper CPU threshold may significantly drop the performance of VMs running on a PM, while selecting low upper CPU threshold value makes consolidation inefficient to reduce energy consumption. Furthermore, there is no specific optimal lower CPU threshold value. So, selecting the suitable value for upper and lower CPU threshold is important. The upper and lower threshold of CPU utilization modified according to parameter *a* and *b*, respectively. CPBFD VM placement algorithm is shown in Algorithm 1.

Algorithm 1: CPBFD VM Placement Algorithm

```
Input: pmList, vmList, $a$, $b$, $c$
Output: allocatedPm
 1; vmList.sortDecreasingUtilization()
 2: for each vm in vmList do
 3:   maxCpu = min
 4:   minCpu = max
 5:   allocatedPm = null
 6:   for each pm in pmList do
 7:     if pm is excluded pm then
 8:       continue
 9:     end if
10:     if pm has enough resources for vm then
```

```
11:       if pm.Cpu $\neq$ 0 and pm is over utilized after allocation vm then
12:         continue
13:       end if
14:       if pm.Cpu $\leq$ a and pm.Cpu $\geq$ b then
15:         if pm.Cpu > maxCpu then
16:           allocatedPm = pm
17:           maxCpu = pm.Cpu
18:         end if
19:       else if Abs(pm.Cpu - c) < minCpu then
20:         allocatedPm2 = pm
21:         minCpu = Abs(pm.Cpu - c)
22:       end if
23:     end if
24:   end for each
25:   if allocatedPm = null then
26:     allocatedPm = allocatedPm2
27:   end if
28:   if allocatedPm $\neq$ null then
29:     allocation.add(vm,allocatedPm)
30:   end if
31: end for each
32: return allocatedPm
```

## 5.2. DCPBFD Algorithm

CPU resource utilization model of PM of DCPBFD algorithm is the same as that used in CPBFD shown in Fig. 2. The only difference between DCPBFD and CPBFD is that upper and lower CPU utilization thresholds in DCPBFD are dynamic, but in CPBFD they are static as discussed in Section 5.1. In DCPBFD, all VMs are sorted in the decreasing order of their current CPU utilizations and allocate each VM to a PM with maximum CPU utilization less than dynamic upper CPU utilization threshold *a* and more than dynamic lower CPU utilization threshold *b*. If there are no PMs with CPU utilization between upper and lower threshold, a PM with the nearest CPU utilization to upper or lower thresholds will be allocated. The priority may be given to PM with CPU utilization nearest to upper threshold or lower threshold by adjusting parameter *c*. PM with CPU utilization that is nearest to *c* and outside of *a* to *b* range will be selected.

There is no specific optimal upper and lower CPU utilization threshold values among the researchers. In DCPBFD, the proposed optimal value of dynamic upper and lower CPU utilization threshold *a* and *b* are based on fixed highest and lowest CPU utilization values, Median Absolute Deviation (MAD) of historical values of PM CPU utilization, and median of historical values of PM CPU utilization divided by number of VMs.

The fixed highest CPU utilization is used for upper CPU threshold as highest CPU threshold, and fixed lowest CPU utilization is used for lower CPU threshold as lowest CPU threshold. MAD is defined as the median of the absolute deviations from the median of historical values of PM CPU utilization. MAD gives an idea about CPU utilization variability, which is important to calculate suitable upper and lower CPU threshold. Median of historical values of PM CPU utilization divided by number of

VMs gives an idea about the utilization of VMs that will be allocated to PM, which is also important to calculate suitable upper and lower CPU thresholds. $a$ and $b$ are calculated as shown in (7) and (8), respectively.

$$a = HighestCpu\ - s \times (CpuMAD + CpuMedian/Number\ of\ VMs) \tag{7}$$

$$b = LowestCpu + s \times (CpuMAD + CpuMedian/Number\ of\ VMs) \tag{8}$$

where $s$ is a parameter that allows the adjustment of the dynamic upper and lower CPU utilization limits, the lower $s$, the wider two sided limits and the less the energy consumption, but the higher the level of SLA violations caused by the consolidation.

DCPBFD VM placement algorithm is shown in Algorithm 2.

```
Algorithm 2: DCPBFD VM Placement Algorithm
   Input: pmList, vmList, highestCpu, lowestCpu, c, s
   Output: allocatedPm
 1: vmList.sortDecreasingUtilization()
 2: for each vm in vmList do
 3:   maxCpu = min
 4:   minCpu = max
 5:   allocatedPm = null
 6:   for each pm in pmList do
 7:     if pm is excluded pm then
 8:       continue
 9:     end if
10:     if pm has enough resources for vm then
11:       if pm.Cpu ≠ 0 and pm is over utilized after allocation vm then
12:           continue
13:       end if
14:       if (pm.CPUHistory.length ≥ 12) then
15:           a = highestCpu – s × (pm.CPUHistoryMAD + pm.CPUHistoryMedian / #ofvm)
16:           b = lowestCpu + s × (pm.CPUHistoryMAD + pm.CPUHistoryMedian / #ofvm)
17:       else
18:         a = highestCpu - 0.05
19:         b = lowestCpu + 0.05
20:       end if
21:       if pm.Cpu ≤ a and pm.Cpu ≥ b then
22:         if pm.Cpu > maxCpu then
23:           allocatedPm = pm
24:           maxCpu = pm.Cpu
25:         end if
26:       else if Abs(pm.Cpu - c) < minCpu then
27:         allocatedPm2 = pm
28:          minCpu = Abs(pm.Cpu - c)
29:        end if
30:      end if
31:   end for each
32:   if allocatedPm = null then
33:       allocatedPm = allocatedPm2
```

```
34:   end if
35:   if allocatedPm ≠ null then
36:     allocation.add(vm,allocatedPm)
37:   end if
38: end for each
39: return allocatedPm
```

Calculation of MAD starts when at least 12 historical values of CPU utilization of PM are obtained. 12 is used as a safe value to calculate MAD [8]. We suggest having the initial values of *a* and *b* before obtaining the first MAD value. The initial value of *a* is adjusted to (highest CPU threshold – 5%), and the initial value of *b* is adjusted to (lowest CPU threshold + 5%), which are supposed to be appropriate to avoid that *a* reaches the highest CPU threshold and *b* reaches the lowest CPU threshold.

## 6. Performance Evaluation

### 6.1. Experiment Setup

A cloud computing user accesses infinite computing resources. Large scale and repeatable experiments which are necessary to analysis and compare the algorithms is very difficult on a real-world infrastructure [8]. We use simulations for ensuring the repeatability of experiments. We use the CloudSim toolkit [14, 31] as a simulation platform that allows the energy consumption modeling on cloud computing environments.

As presented in [8], we simulate a data center containing 800 heterogeneous PMs. PM types are HP ProLiant ML110 Generation 4 and HP ProLiant ML110 Generation 5. CPU frequencies of the servers are mapped onto MIPS rating: 1,860 MIPS for each core of HP ProLiant ML110 G4 server, and 2,660 MIPS for each core of HP ProLiant ML110 G5 server. Each server is modeled to have 1 GB/s network bandwidth. VM characteristics correspond to Amazon EC2 instance types including Extra Large Instance (3.75 GB, 2,000 MIPS); High-CPU Medium Instance (0.85 GB, 2,500 MIPS); Micro Instance (613 MB, 500 MIPS); and Small Instance (1.7 GB, 1,000 MIPS). Initialization of VMs allocation is done according to the resource requirements of VM types. However, VM's useless resources during the lifetime according to the workload create opportunities for dynamic consolidation.

### 6.2. Workload Data

To make the evaluation of simulation applicable, we use real-world workload traces provided as a monitoring infrastructure for PlanetLab [43], which is a part of the CoMon project. We use CPU utilization traces presented in [8] from more than a thousand VMs running on PMs located in more than 500 places around the world. Utilization is collected every 5 minutes. Random 10 days from the collected workload traces are used in the simulations. The workload characteristics for each day are presented in Table 1. In the simulations, each VM is randomly assigned a workload trace from one of VMs from the corresponding day. VM consolidation is not limited by the memory bounds to avoid the constraint on the consolidation.

**Table 1.** Workload data characteristics [8]

| Workload | Number of VMs | Mean of CPU utilization | Standard deviation of CPU utilization | Median of CPU utilization |
|---|---|---|---|---|
| 1 | 1052 | 12.31% | 17.09% | 6% |
| 2 | 898 | 11.44% | 16.83% | 5% |
| 3 | 1061 | 10.70% | 15.57% | 4% |
| 4 | 1516 | 9.26% | 12.78% | 5% |
| 5 | 1078 | 10.56% | 14.14% | 6% |
| 6 | 1463 | 12.39% | 16.55% | 6% |
| 7 | 1358 | 11.12% | 15.09% | 6% |
| 8 | 1233 | 11.56% | 15.07% | 6% |
| 9 | 1054 | 11.54% | 15.15% | 6% |
| 10 | 1033 | 10.43% | 15.21% | 4% |

## 6.3. Simulations Results

The algorithms are evaluated using the CloudSim and the workload traces presented in Section 6.2. We compare proposed CPBFD algorithm to FFD, BFD and PABFD. We simulate all combinations of SM underload detection algorithm, LR overloading detection algorithm, MMT VM selection policy and four VM placement algorithms (FFD, BFD, PABFD and proposed CPBFD). Algorithms that are used in dynamic VM consolidation problem are shown in Fig. 3.

For the proposed CPBFD algorithm, two-sided limits (*a* and *b*) of CPU utilization is varied from wider to narrower based as 90% to 10%, 80% to 20%, 70% to 30% and 60% to 40%. Moreover, *c* parameter is varied to give more priority for low CPU utilization, equal priority to low or high CPU utilization, and more priority for high CPU utilization as 0.45, 0.50, and 0.55, respectively. According to these variations, combinations of *a*, *b* and *c* parameters are varied as (0.9, 0.1, 0.45), (0.9, 0.1, 0.5), (0.9, 0.1, 0.55), (0.8, 0.2, 0.45), (0.8, 0.2, 0.5), (0.8, 0.2, 0.55), (0.7, 0.3, 0.45) (0.7, 0.3, 0.5), (0.7, 0.3, 0.55), (0.6, 0.4, 0.45) (0.6, 0.4, 0.5), and (0.6, 0.4, 0.55). The purposes of these variations are to get better upper and lower CPU utilization threshold and better priority for low or high CPU utilization for the proposed CPBFD algorithm and to use these better parameters in proposed DCPBFD algorithm.

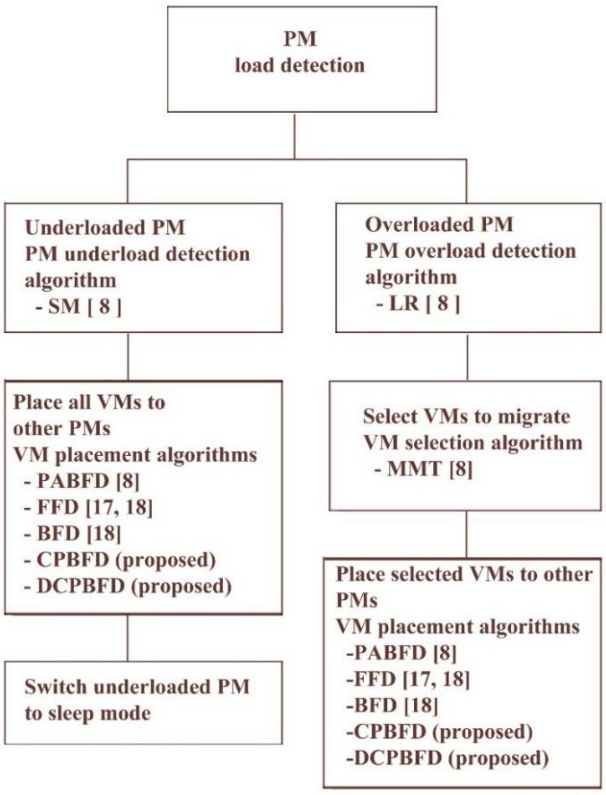

**Fig. 3.** Dynamic VM consolidation problem and algorithms in cloud data centers.

Figs. 4 to 9 show the average results with 95% confidence intervals of ESV metric, energy consumption, SLAV metric, number of migrations, PDM metric, and SLATAH metric for all algorithms combination in 10 workload cases, respectively. Results in Figs. 4 to 9 show that CPBFD leads to better results regarding all parameters compared to FFD. CPBFD leads to better of ESV metric, SLAV metric, number of migrations, PDM metric, and SLATAH metric compared to BFD and PABFD. Moreover, BFD leads to better results regarding ESV metric, SLAV metric, number of migrations, PDM metric compared to FFD and PABFD. On the other hand, PABFD only leads to the least energy consumption compared to all algorithms. Fig. 4 shows that CPBFD has better ESV on average approximately 61.1%, 54.2% and 17.5% than FFD, PABFD, and BFD, respectively. Fig. 5 shows that CPBFD has less energy consumption on average approximately 3.8% than FFD and more energy consumption on average approximately 0.75%, and 3% than BFD, and PABFD, respectively. Fig. 6 shows that CPBFD has less SLAV on average approximately 54.2%, 56.7% and 19% than FFD, PABFD, and BFD, respectively. Fig. 7 shows that CPBFD has fewer number of migration on average approximately 22.1%, 27.4%, and 8.3% than FFD, PABFD, and BFD, respectively. Fig. 8 shows that CPBFD has less PDM on average approximately 35.9%, 52.5% and 12.7% than FFD, PABFD, and BFD, respectively. Fig. 9 shows that CPBFD has less SLATAH on average approximately 19.3%, 8% and 13.1% than FFD, PABFD, and BFD, respectively.

CPBFD_70,30,55 has better ESV and SLAV on average approximately 7.7% and 8.7% than CPBFD with other parameters. CPBFD_80,20,55 has better efficient energy consumption on average approximately 0.8% than CPBFD with other parameters. CPBFD with ($a$ = 70, $b$ = 30) of CPU has better ESV on average approximately 14.5%, 3% and 4.3% than CPBFD with ($a$ = 90, $b$ = 10), CPBFD with ($a$ = 80, $b$ = 20), and CPBFD with ($a$ = 60, $b$ = 40), respectively. CPBFD with ($a$ = 70, $b$ = 30) of CPU has better SLAV on average approximately 15.9%, 3.9% and 3.9% than CPBFD with ($a$ = 90, $b$ = 10), CPBFD with ($a$ = 80, $b$ = 20), and CPBFD with ($a$ = 60, $b$ = 40), respectively. CPBFD with ($a$ = 80, $b$ = 20) has almost the same energy consumed by CPBFD with ($a$ = 90, $b$ = 10), and better efficient energy consumption on average approximately 0.7%, and 1.5% than CPBFD with ($a$ = 70, $b$ = 30), and CPBFD with ($a$ = 60, $b$ = 40).

CPBFD_80,20,55 has almost the same energy consumed by BFD, and more energy consumption on average approximately 2.3% than PABFD, but CPBFD_80,20,55 has better ESV and SLAV on average approximately 62.2% and 63.9% than PABFD. This means even if we consider that ESV metric is modified to the product of energy consumption powered by 20 and SLAV (modified ESV = $E^{20} \times$ SLAV), CPBFD_80,20,55 will still better than PABFD in regard of modified ESV.

The energy consumption changes slightly compared to SLA violation. Therefore, the impact of SLA violation on ESV metric is greater than energy consumption. However, the suitable value of $a$, $b$ and $c$ should be selected to make a tradeoff between meeting QoS and improving energy efficiency. From the simulation results, we observe that CPBFD with ($a$ = 70, $b$ = 30) provides best ESV metric, SLA violations and number of migrations. In addition, CPBFD with ($a$ = 80, $b$ = 20) provides best efficient energy consumption. Moreover, CPBFD when $c$ parameter equals 0.55 leads to a little better ESV metric, SLA violations and energy consumption. Furthermore, we observe that selecting moderate two-sided limits of CPU utilization between ($a$ = 70, $b$ = 30) and ($a$ = 80, $b$ = 20) for CPBFD is better than selecting too wide ($a$ = 90, $b$ = 10) or too narrow ($a$ = 60, $b$ = 40) sided limits of CPU utilization.

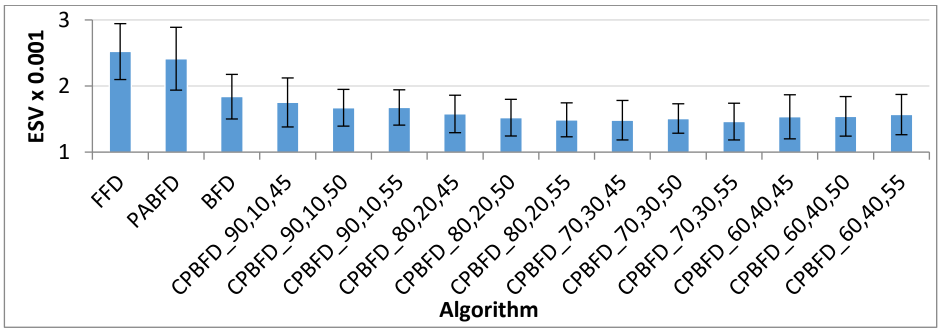

**Fig. 4.** ESV metric of VM placement algorithms

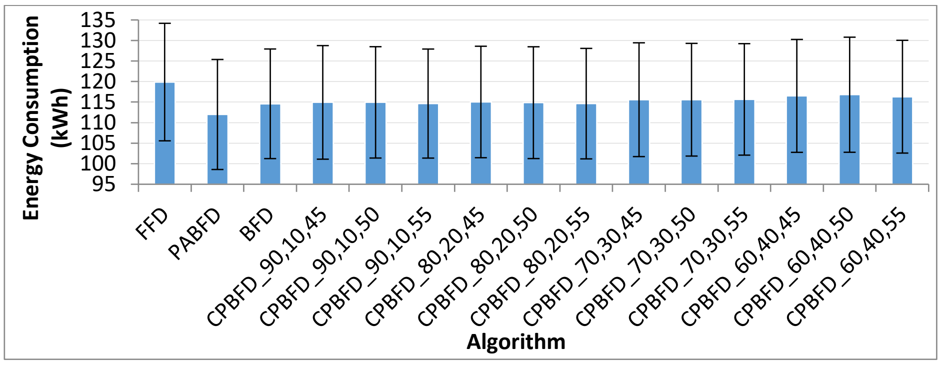

**Fig. 5.** Energy consumption of VM placement algorithms

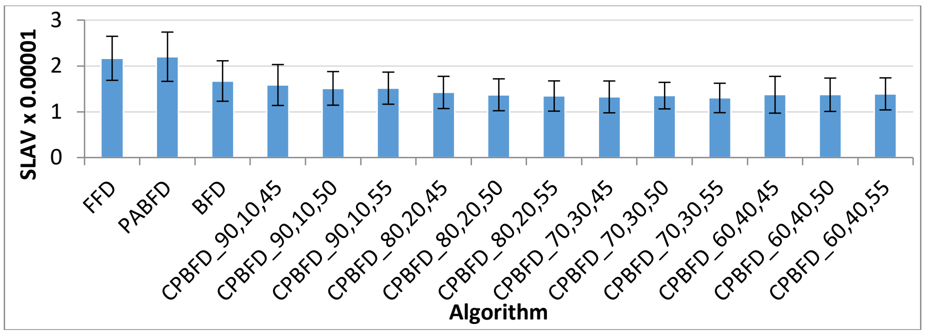

**Fig. 6.** SLAV metric of VM placement algorithms

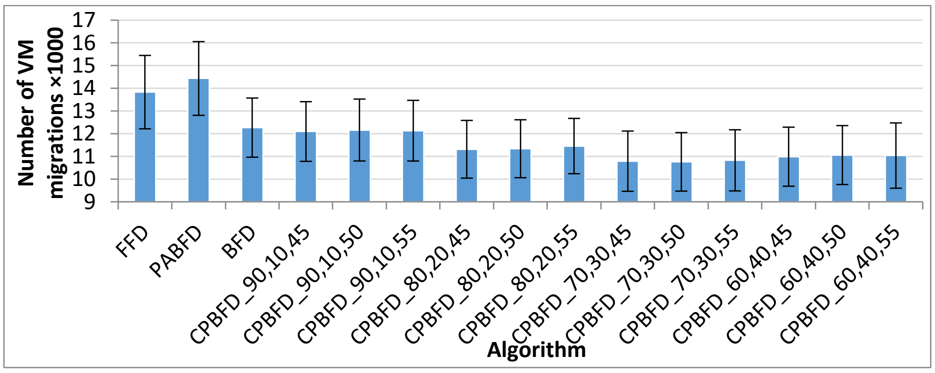

**Fig. 7.** Number of VM migrations of VM placement algorithms

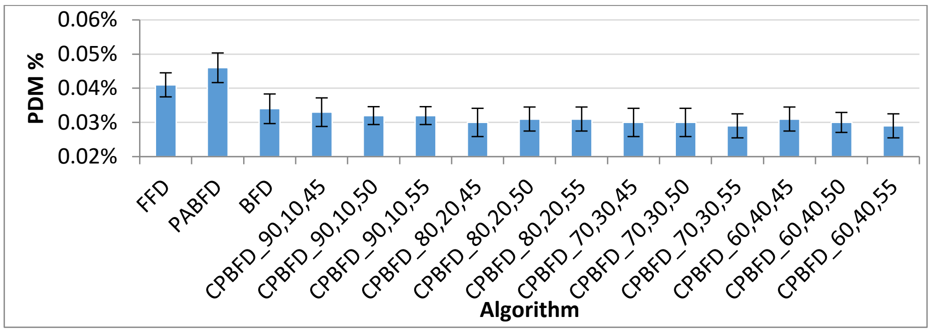

**Fig. 8.** PDM metric of VM placement algorithms

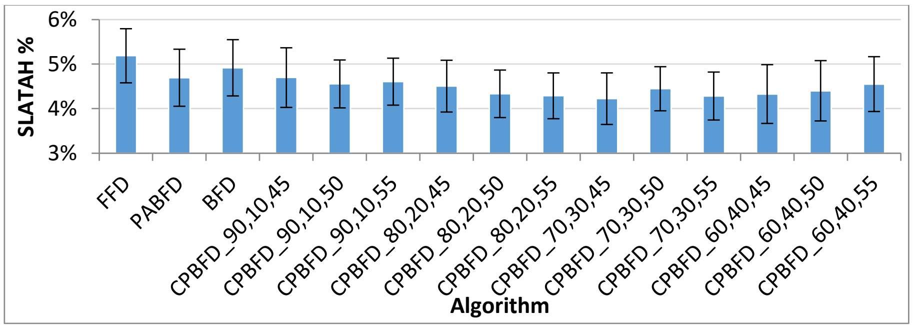

**Fig. 9.** SLATAH metric of VM placement algorithms

For the proposed DCPBFD algorithm, the fixed highest CPU utilization used for higher CPU utilization limit (*a*) is set to 80%, and the fixed lowest CPU utilization used for lower CPU utilization limit (*b*) is set to 20%, which are suitable moderate values according to the results obtained from CPBFD algorithm. *s* parameter of two-sided limits (*a* and *b*) of CPU utilization is varied as 1, 0.75, and 0.5. Moreover, *c* parameter is set to 0.55 to give more priority for high CPU utilization, which gives the best result according to the results obtained from CPBFD algorithm. According to these variations, combinations of the fixed highest CPU utilization, the fixed lowest CPU utilization, *c* and *s* parameters are varied as (0.8, 0.2, 0.55, 1), (0.8, 0.2, 0.55, 0.75), and (0.8, 0.2, 0.55, 0.5).

Figs. 10 to 15 show the average results with 95% confidence intervals of ESV metric, energy consumption, SLAV metric, number of migrations, PDM metric, and SLATAH metric for DCPBFD algorithm with all combination of parameters, and CPBFD_80,20,55, CPBFD_75,25,55, and CPBFD_70,30,55 that have the best results compared to CPBFD with other parameters. Results in Figs. 10 to 15 show that DCPBFD_80,20,55,75 leads to better results compared to DCPBFD with other parameters, and CPBFD_75,25,55 leads to better results compared to CPBFD with other parameters. Results in Figs. 10, 12, 13, 14, and 15 show that DCPBFD_80,20,55,75 leads to better of ESV metric, SLAV metric, number of migrations, PDM metric, and SLATAH metric compared to DCPBFD and CPBFD with other parameters. Moreover, results in Fig. 11 show that CPBFD_80,20,55 provides a little better efficient energy consumption compared to DCPBFD and CPBFD with other parameters. We observe that selecting moderate two-sided limits of CPU utilization for DCPBFD by selecting moderate highest CPU utilization and lowest CPU utilization and adjusting *s* parameter to moderate value is better than selecting too wide or too narrow sided limits of CPU utilization.

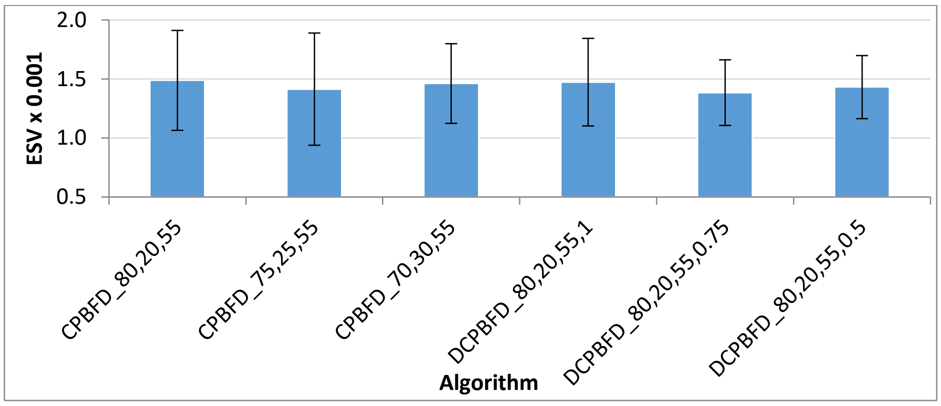

**Fig. 10.** ESV metric of CPBFD and DCPBFD algorithms

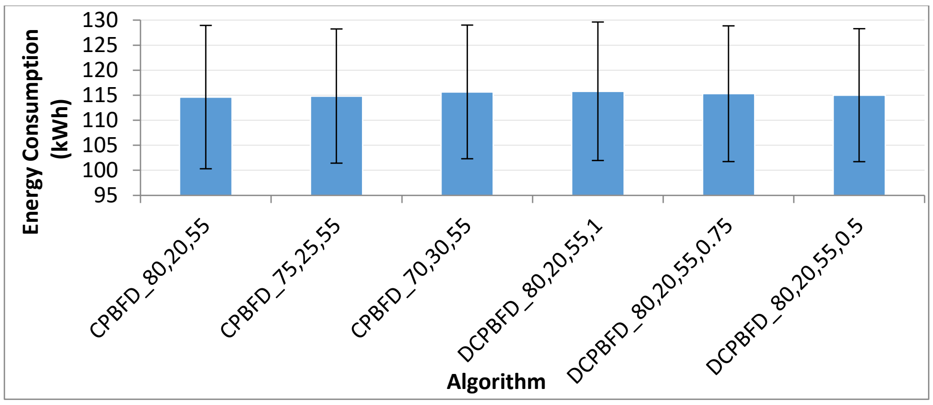

**Fig. 11.** Energy consumption of CPBFD and DCPBFD algorithms

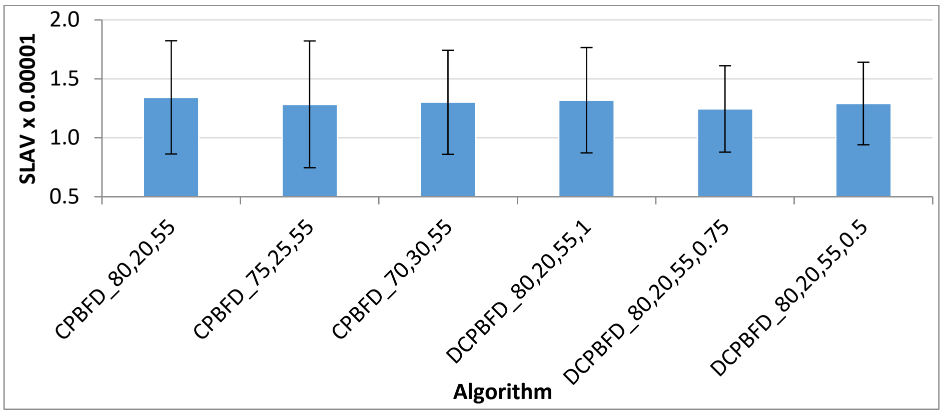

**Fig. 12.** SLAV metric of CPBFD and DCPBFD algorithms

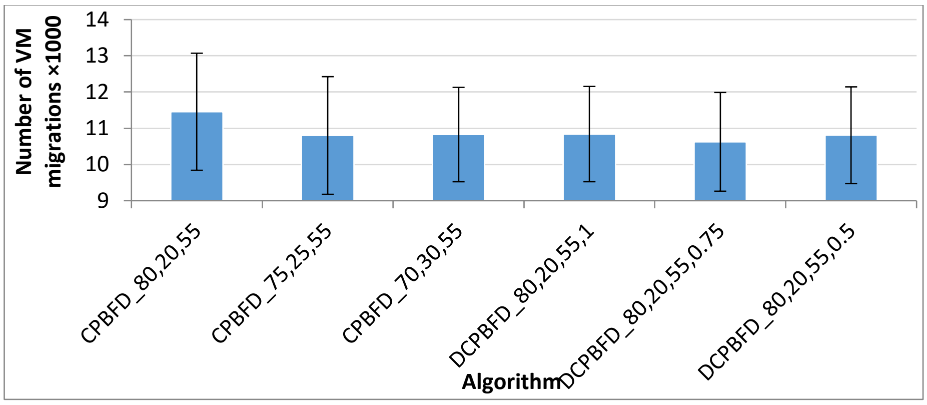

**Fig. 13.** Number of VM migrations of CPBFD and DCPBFD algorithms

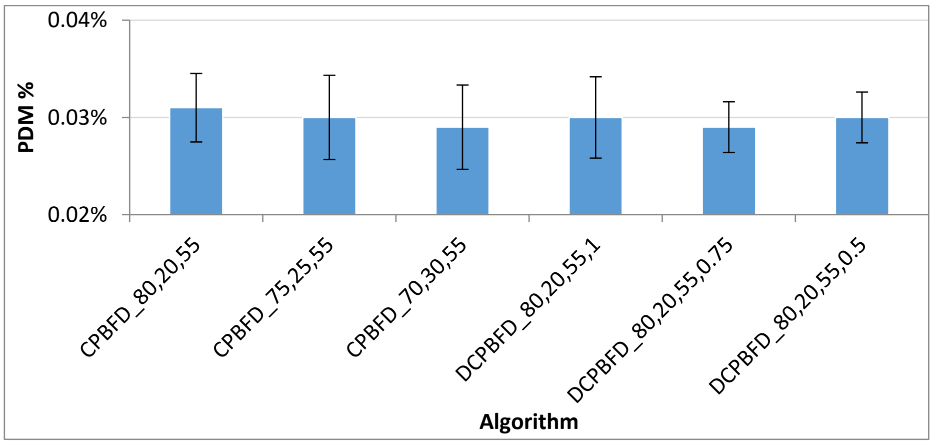

**Fig. 14.** PDM metric of CPBFD and DCPBFD algorithms.

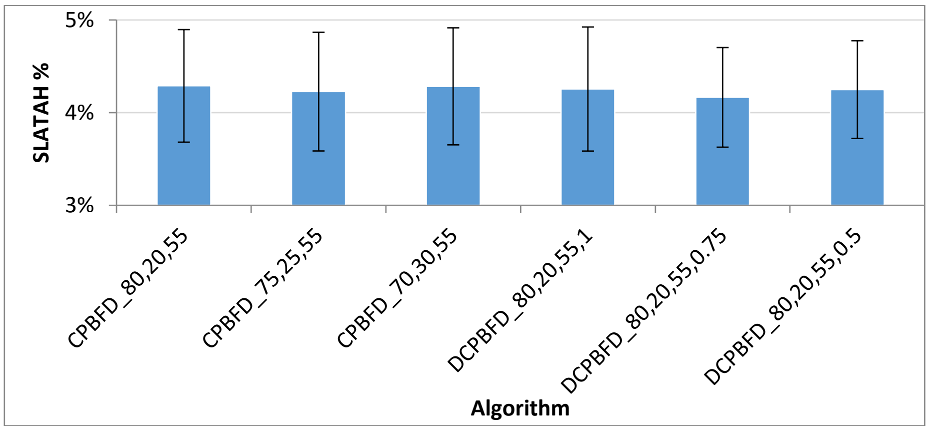

**Fig. 15.** SLATAH metric of CPBFD and DCPBFD algorithms.

## 7. Conclusion and Future Work

The goal of the proposed CPBFD and DCPBFD VM placement algorithms is to improve energy efficiency, and SLA in cloud data centers. A number of VM placement algorithms are implemented to compare with the proposed algorithm. We evaluate the algorithms through simulations with real-world workload traces. CPBFD and DCPBFD algorithms produce better results by avoiding VM migrations to PM with high load that may cause SLA violations or low load, which lead to more active PMs and more energy consumption. The simulation results show that CPBFD and DCPBFD with moderate two-sided limits of CPU utilization provide the least ESV, least SLA violations, least VM migrations, and efficient energy consumption. However, PABFD algorithm leads to a little better energy consumption than CPBFD and DCPBFD algorithms.

As a future work, we plan to extend our research by using a software framework for dynamic and energy efficient consolidation of VMs applied in existing cloud deployments and in research on dynamic consolidation of VMs to optimize the resource utilization and energy efficiency.

## References

1. Weber, W. D., Fan, X., Barroso, L. A.: Powering the data center. U.S. Patent No. 8,595,515. Washington, DC: U.S. Patent and Trademark Office. (2013)
2. Beloglazov, A., Buyya, R.: Managing overloaded hosts for dynamic consolidation of virtual machines in cloud data centers under quality of service constraints. IEEE Transactions on Parallel and Distributed Systems, Vol. 24, No.7, 1366-1379. (2013)
3. Verma, A., Ahuja, P., Neogi, A.: pMapper: power and migration cost aware application placement in virtualized systems. In Proceedings of ACM/IFIP/USENIX International Conference on Distributed Systems Platforms and Open Distributed Processing, Springer Berlin Heidelberg, 243-264. (2008)

4. Jung, G., Hiltunen, M. A., Joshi, K. R., Schlichting, R. D., Pu C.: Mistral: Dynamically managing power, performance, and adaptation cost in cloud infrastructures. In Proceedings of IEEE 30th International Conference on Distributed Computing Systems, 62-73. (2010)
5. Gmach, D., Rolia, J., Cherkasova, L., Kemper, A.: Resource pool management: Reactive versus proactive or lets be friends. Computer Networks, Vol. 53, No. 17, 2905-2922. (2009)
6. Fu, X., Zhou, C.: Virtual machine selection and placement for dynamic consolidation in Cloud computing environment. Frontiers of Computer Science, Vol. 9, No. 2, 322-330. (2015)
7. Beloglazov, A., Abawajy, J., Buyya, R.: Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing. Future Generation Computer Systems, Vol. 28, No. 5, 755-768. (2012)
8. Beloglazov, A., Buyya, R.: Optimal online deterministic algorithms and adaptive Heuristics for energy and performance efficient dynamic consolidation of virtual machines in cloud data centers. Concurrency and Computation: Practice and Experience, Vol. 24, No. 13, 1397-1420. (2012)
9. Han, G., Que, W., Jia, G., Shu, L.: An efficient virtual machine consolidation scheme for multimedia cloud computing. Sensors, Vol. 16, No. 2, 246-246. (2016)
10. Guenter, B., Jain, N., Williams, C.: Managing cost, performance, and reliability tradeoffs for energy-aware server provisioning. In Proceedings of 30'st Annual IEEE Intl. Conf. on computer communications (INFOCOM), 1332-1340. (2011)
11. Alsbatin, L., Oz, G., Ulusoy, A. H.: An overview of energy-efficient cloud data centres. In Proceedings of the International Conference of computer and applications (ICCA2017), Dubai, United Arab Emirates, 211- 214. (2017)
12. Lee, Y. C., Zomaya, A. Y.: Energy conscious scheduling for distributed computing systems under different operating conditions. IEEE Transactions on Parallel and Distributed Systems, Vol. 22, No. 8, 1374–1381. (2011)
13. Kang, J., Ranka, S.: Dynamic slack allocation algorithms for energy minimization on parallel machines. Journal of Parallel and Distributed Computing, Vol. 70, No. 5, 417–430. (2010)
14. Buyya, R., Ranjan, R., Calheiros, R.N.: Modeling and simulation of scalable cloud computing environments and the CloudSim toolkit: Challenges and opportunities. In Proceedings of 2009 Conference on High Performance Computing & Simulation Conference, 1-11. (2009)
15. Beloglazov, A., Buyya, R.: Energy efficient resource management in virtualized cloud data centers. in Proceedings of the 10th IEEE/ACM International Symposium on Cluster, Cloud, and Grid Computing, 826–831. (2010)
16. Alsbatin, L., Oz, G., Ulusoy, A. H.: A Novel Physical Machine Overload Detection Algorithm Combined with Queiscing for Dynamic Virtual Machine Consolidation in Cloud Data Centers. The International Arab Journal of Information Technology, Vol. 17, No. 3. (2020). Available online: https://iajit.org/PDF/May%202020,%20No.%203/16897.pdf.
17. Kaushar, H., Ricchariya, P., Motwani, A.: Comparison of SLA based energy efficient dynamic virtual machine consolidation algorithms. International Journal of Computer Applications, Vol. 102, No. 16. (2014)
18. Sharifi, M., Salimi, H., Najafzadeh, M.: Power-efficient distributed scheduling of virtual machines using workload-aware consolidation techniques. Journal of Supercomputing, Vol. 61, No.1, 46-66. (2012)
19. Deng, D., He, K., Chen, Y.: Dynamic virtual machine consolidation for improving energy efficiency in cloud data centers. In Proceedings of 4th international conference on cloud computing and intelligence systems. (2016)
20. Khoshkholghi, M. A., Derahman, M. N., Abdullah, A., Subramaniam, S., Othman, M.: Energy-efficient algorithms for dynamic virtual machine consolidation in cloud data centers. IEEE Access, Vol. 5, 10709-10722. (2017)

21. Shaw, S. B., Singh, A. K.: Use of proactive and reactive hotspot detection technique to reduce the number of virtual machine migration and energy consumption in cloud data centre. Computers & Electrical Engineering, Vol. 47, 241-254. (2015)
22. Nguyen, T. H., Francesco, M. D., Yla-Jaaski, A.: Virtual Machine Consolidation with Multiple Usage Prediction for Energy-Efficient Cloud Data Centers. IEEE Transactions on Services Computing, Vol. 99, 1-14. (2017)
23. Zhou, Z., Abawajy, J., Chowdhury, M., Hu, Z., Li, K., Cheng, H., Alelaiwi, A. A., Li, F.: Minimizing SLA violations and power consumption in cloud data centers using adaptive energy-aware algorithms. Future Generation Computer Systems, Vol. 86, 836-850. (2018)
24. Wang, X., Wang, Y.: Coordinating power control and performance management for virtualized server clusters. IEEE Transactions on Parallel and Distributed Systems, Vol. 22, No. 2, 245-259. (2011)
25. Yadav, R. Zhang, W.: MeReg: Managing Energy-SLA Tradeoff for Green Mobile Cloud Computing. Wireless Communications and Mobile Computing, Vol. 2017. (2017)
26. Kakadia, D., Kopri, N., Varma, V.: Network-aware virtual machine consolidation for large data centers. In Proceedings of 3rd International Workshop on Network-Aware Data Management. (2013)
27. Forsman, M., Glad, A., Lundberg, L., Ilie, D.: Algorithms for automated live migration of virtual machines. Journal of Systems and Software, Vol. 101, 110-126. (2015)
28. Yue, M.: A simple proof of the inequality FFD (L)< 11/9 OPT (L)+ 1,for all l for the FFD bin-packing algorithm. Acta Mathematicae Applicatae Sinica (English Series), Vol. 7, No. 4, 321-331. (1991)
29. Coffman, E. G., Garey, M. R., Johnson, D. S.: Approximation algorithms for bin-packing: A survey. Approximation algorithms for NP-hard problems, 46-93. (1996)
30. Shi, L., Furlong, J., Wang, R.: Empirical evaluation of vector bin packing algorithms for energy efficient data centers. in IEEE Symposium on Computers and Communications, 9-15. (2013)
31. Calheiros, R. N., Ranjan, R., Beloglazov, A., Rose, C. A., Buyya, R.: CloudSim: A toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. Software: Practice and Experience, Vol. 41, No. 1, 23-50. (2011)
32. Minas, L., Ellison, B.: Energy efficiency for information technology: How to reduce power consumption in servers and data centers. Intel Press. (2009)
33. Fan, X., Weber, W. D., Barroso, L.A.: Power provisioning for a warehouse-sized computer. ACM SIGARCH Computer Architecture News, Vol. 35, No. 2, 13-23. (2007)
34. Kusic, D., Kephart, J. O., Hanson, J. E., Kandasamy, N., Jiang, G.: Power and performance management of virtualized computing environments via lookahead control. Cluster Computing, Vol. 12, No. 1, 1-15. (2009)
35. Lange, K. D.: Identifying shades of green: The SPECpower benchmarks. Computer, Vol. 42, No.3, 95-97. (2009)
36. Beloglazov, A., Buyya, R.: Adaptive threshold-based approach for energy-efficient consolidation of virtual machines in cloud data centers. In Proceedings of the 8th International Workshop on Middleware for Grids, Clouds and e-Science. (2010)
37. Ammar, A., Luo, J, Tang, Z, Wajdy, O.: Intra-Balance Virtual Machine Placement for Effective Reduction in Energy Consumption and SLA Violation. IEEE Access, Vol. 7. (2019)
38. Murtazaev, A., Oh, S.: Sercon: Server consolidation algorithm using live migration of virtual machines for green computing. IETE Techical Review, Vol. 28, No. 3, 212-231. (2011)
39. Vogels, W.: Beyond Server Consolidation. ACM Queue, Vol. 6, No. 1, 20-26. (2008)
40. Magesh, H., Smith, J.: Server consolidation through virtualization with quad-core intel xeon processors. White Paper by Intel Corporation and Infosys Technologies. (2008)
41. Wood, T., Shenoy, P., Venkataramani, A., Yousif, M.: Black-box and gray-box strategies for virtual machine migration. 4th USENIX Symposium on Networked Systems Design & Implementation. (2007)

42. Song, Y., Wang, H., Li, Y., Feng, B., Sun, Y.: Multi-tiered ondemand resource scheduling for vm-based data center. 9th IEEE/ ACM International Symposium on Cluster Computing and the Grid. (2009)
43. Park, K., Pai, V. S.: CoMon: A mostly-scalable monitoring system for PlanetLab. ACM SIGOPS Operating Systems Review, Vol. 40, No. 1, 65-74. (2006)

**Loiy Alsbatin** received his B.S. degree in Computer Engineering from Mutah University, Jordan, in 2008, his M.S. degree in Computer Engineering from Jordan University of Science and Technology (JUST), Jordan, in 2012, and his Ph.D. degree in Computer Engineering from Eastern Mediterranean University (EMU), in Famagusta, North Cyprus, in 2019. He is currently a faculty member in the Department of Computer Science at Shaqra University, Saudi Arabia. His current research interests include distributed system, cloud computing, resource management, and virtualization.

**Gurcu Oz** received her B.S, M.S. degrees from the Electrical and Electronic Engineering department and Ph.D. degree from the Computer Engineering Department of Eastern Mediterranean University, in Famagusta, North Cyprus. Currently, she is working in the Department of Computer Engineering of Eastern Mediterranean University. Her research interests include computer networks, wireless networks, distributed systems, cloud computing, system simulation, information security and network security.

**Ali Hakan Ulusoy** was born in Eskisehir, Turkey, on June 3, 1974. He graduated from the double major program of the department of Electrical and Electronic Engineering (EEE) and department of Physics in Eastern Mediterranean University (EMU) in 1996. He received his M.S. and Ph.D. degrees in EEE in EMU in 1998 and 2004, respectively. He joined Information Technology department, EMU, in 2004. His current research interests include wireless communications, receiver design, channel estimation, fuzzy systems, wireless networks, cloud computing, millimeter wave communications and healthcare system development.

*Received: November 22, 2018; Accepted: November 20, 2019*