IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

SPECIAL SECTION ON EMERGING TRENDS, ISSUES, AND CHALLENGES IN ENERGY-EFFICIENT CLOUD COMPUTING

Received December 16, 2017, accepted February 9, 2018, date of publication March 2, 2018, date of current version April 4, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2808320

SLA-Aware Energy Efficient Resource Management for Cloud Environments

SAAD MUSTAFA[1], KASHIF BILAL[1], SAIF UR REHMAN MALIK[2], AND SAJJAD A. MADANI[2]
[1]Department of Computer Science, COMSATS Institute of Information Technology, Abbottabad 22060, Pakistan
[2]Department of Computer Science, COMSATS Institute of Information Technology, Islamabad 45550, Pakistan

Corresponding author: Saad Mustafa (saadmustafa@ciit.net.pk)

ABSTRACT Cloud computing provides online services to customers using pay as you go model. The Cloud computing enables customers to outsource the large and complex tasks to the cloud data centers for the execution and result generations. Cloud data centers host the incoming tasks by providing resources, such as CPU, RAM, storage, and bandwidth. As the large data centers provide the basic resources to hosted tasks, they also consume a huge amount of energy, which leads to higher operating cost and CO_2 traces. Therefore, research community felt the need to provide energy-efficient solutions that reduce the impact of the aforementioned issues. Consequently, researchers proposed many solutions, and majority of them are based upon the concept of consolidation. Consolidation techniques place the incoming tasks on minimum possible servers, thus increasing the resource utilization and decreasing energy consumption. In this paper, we use the same workload consolidation concept and present two techniques that reduce energy consumption while ensuring the negotiated quality-of-service. Moreover, we enhanced two existing techniques by improving the energy efficiency and introducing service level agreement (SLA) awareness to minimize the overall SLA violations. Performance evaluation of the proposed techniques is done based on fluctuating workloads, and results show that our techniques outperform existing techniques in terms of energy efficiency, SLA compliance, and performance assurance at the network level. Moreover, correctness of the proposed techniques is demonstrated by modeling and verifying them with the help of high-level Petri Nets, SMT-Lib, and Z3 solver.

INDEX TERMS Cloud computing, energy efficiency, resource management, resource allocation, SLA awareness.

I. INTRODUCTION

Cloud computing is emerging as one of the popular solutions for on-demand and dynamic resource provisioning [1]. The provisioning and maintenance of cloud resources are done with the help of resource management (RM) techniques [2], [3]. RM techniques are responsible to keep the track of free resources and assign the resources from the free pool to incoming tasks. Along with certain advantages, RM techniques introduce a set of research challenges that need to be addressed. One of the key research challenges in RM is the energy efficiency, and researchers are focusing on this challenge [4]–[6]. Energy efficiency not only minimizes the overall the expenditure of the cloud data centers (DCs), but also have a positive effect on environment [7], [8]. CO_2 and Green House Gases (GHG) play an important role in environmental pollution [12]–[14], and reports published in 2007 highlight the fact that Information and Communication Technology (ICT) industry produced about 2% of total CO_2 emissions. Moreover, recent report predicts that the CO_2 emissions of ICT industry will increase to 12% by 2020 [15]. On the other hand, big ICT companies like Google are already hosting hundred and thousands of servers and in 2010 these companies consumed about 271.8 billion kWh electricity [9], [10]. Furthermore, recent reports suggest that the energy consumption of ICT industry will grow to about 1963 billion kWh in 2020 [11].

In computational DCs, servers are considered as key energy consumers. Servers consume about 80% of the total energy, whereas, the rest of 20% is consumed by networking and storage devices [16]. The main reason for the aforementioned fact is the frequent use of servers for hosting and computing of users' tasks. Recently, researchers have proposed various energy-efficient solutions that tend to reduce energy consumption at the server level [17]. The proposed solutions

can be broadly characterized into hardware level and software level solutions. Hardware level solutions use dynamic voltage and frequency scaling (DVFS) technique [18], [24]–[26]. Whereas, software level techniques provide energy efficiency by using workload consolidation approach [19], [27]–[32]. DVFS technique, scales the energy based on the computational load that is placed on the server. Conversely, workload consolidation technique places the workload on minimum possible servers which increases the resource utilization and VM density on the servers. Increased resource utilization reduces the number of active servers that leads to reduction in energy consumption. Though, workload consolidation minimizes energy consumption, but such kind of technique may lead to service level agreements (SLA) violations if desired quantity of resources are not available [20], [21]. Moreover, it is noticed that there is a scope to further enhance the energy efficiency in workload consolidation techniques by selecting the better servers.

In this paper, we present two workload consolidation techniques that attempt to reduce energy consumption and resultant SLA violations. Available Capacity and Power (ACP), and Required Capacity and Power (RCP) are based on best fit decreasing (BFD) algorithm [22] that is traditionally used for online bin packing. BFD algorithm is considered a good VM placement algorithm because it places the VMs with higher resource requirements first. Remaining VMs with lower resource requirements are placed on the servers with left-over resources. Moreover, our proposed techniques focus on available CPU capacity and energy that are still to be consumed rather than overall CPU capacity and maximum energy consumption. This provides a clear view that how much energy will be consumed by a certain VM. Moreover, our proposed techniques also ensure agreed QoS level by using the threshold mechanisms [23]. Threshold mechanism preserves a certain capacity of resources free so that those resources can be assigned to a VM in case the resource requirements of that particular VM are increased. Furthermore, two existing energy-efficient techniques are enhanced to improve energy efficiency and SLA violations. Following are the major contribution of this paper:

- The detailed analysis of the selected energy-efficient resource management heuristics is provided using same environments and assumptions.
- Two new SLA-aware energy-efficient resource management techniques, ACP and RCP, are proposed that optimize energy, SLA violations and network load simultaneously.
- Extensions to PCABFD and EPOBF heuristics are proposed. The proposed heuristics aim to enhance performance in terms of energy efficiency, SLA violations, and performance degradation due to migrations.
- Time and space complexity analysis of proposed and selected solution is presented.
- Proposed techniques are formally verified and modeled by the help of High-level Petri Nets, SMT-Lib, and Z3 Solver.

Algorithm 1 Pseudocode for Proposed ACP Algorithm

ACP Algorithm

Input: *srvList(S), vmList(V), threshold*;
Output: VM allocation

1) $V_s' \leftarrow$ sort all $v \in V$ in order of CPU requirement from higher to lower
2) $S_i' \leftarrow$ sort all $s \in S$ in order of utilization from higher to lower
3) **for** all the $vs \in V_s'$ **do**
4) $Max\_ratio \leftarrow 0$
5) $Allocated\_host \leftarrow$ Null
6) $comp\_req_s \leftarrow$ computational requirements of VM v_s
7) **for** all $pi \in S_i'$ where utilization of $pi +$ $comp\_req_s < threshold$ **do**
8) $Available\_CPU\_cap_i \leftarrow$ Avialable CPU capacity of pi
9) $Max\_CPU\_cap_i \leftarrow$ Maximum CPU capacity of pi
10) $Peak\_Energy_i \leftarrow$ Peak energy consumption of pi
11) $Current\_Energy_i \leftarrow$ Current energy consumption of pi
12) $R_{ACP} \leftarrow Available\_CPU\_cap_i$ $/(Peak\_Energy_i - Current\_Energy_i)$
13) **if** $Available\_CPU\_cap_i == Max\_CPU\_cap_i$ and $Allocated\_host$! = Null
14) **Break**
15) **end if**
16) **if** $R_{ACP} > Max\_ratio$
17) $Allocated\_host \leftarrow pi$
18) $Max\_ratio \leftarrow R_{ACP}$
19) **end if**
20) **end for**
21) **if** $Allocated\_host \neq$ Null
22) place v_s on $Allocated\_host$
23) **end if**
24) **end for**
25) $Allocation =$ get_allocation( )
26) *return Allocation*

The rest of the paper is organized as follows: In Section II, the related work done in the area of resource provisioning is presented. Section III describes the system model. In Section IV, energy efficient resource allocation algorithms are discussed. Section V highlights the proposed strategies, and Section VI represent the formal verification of the proposed techniques. Performance evaluation is discussed in Section VII. The paper concludes in Section VIII.

II. RELATED WORK

Energy efficiency is considered as one of the major research challenges while performing resource management in cloud environments [6]. Several solutions are provided by the researchers to handle the aforesaid issue that are either based

Algorithm 2 Pseudocode for Proposed RCP Algorithm

RCP Algorithm

Input: *srvList(S), vmList(V), threshold*;
Output: VM allocation

1) $V_s' \leftarrow$ sort all $v \in V$ in order of CPU requirement from higher to lower
2) $S_i' \leftarrow$ sort all $s \in S$ in order of CPU capacity from higher to lower
3) **for** all the $vs \in V_s'$ **do**
4) $Max\_ratio \leftarrow 0$
5) $Allocated\_host \leftarrow$ Null
6) $comp\_req_s \leftarrow$ computational requirements of VM v_s
7) **for** all $pi \in S_i$ where utilization of $pi +$ $comp\_req_s < threshold$ **do**
8) $Before\_Alloc\_Energy_i \leftarrow$ Energy consumption of pi before allocation of vs
9) $Available\_CPU\_cap_i \leftarrow$ Avilable CPU capacity of pi
10) $Max\_CPU\_cap_i \leftarrow$ Maximum CPU capacity of pi
11) $After\_Alloc\_Energy_i \leftarrow$ Energy consumption of pi after allocation of vs
12) $R_{RCP} \leftarrow comp\_req_s/(After\_Alloc\_Energy_i - Before\_Alloc\_Energy_i)$
13) **if** $Available\_CPU\_cap_i = Overall\_CPU\_cap_i$ and $Allocated\_host$! = Null
14) **Break**
15) **end if**
16) **if** $R_{RCP} > Max\_ratio$
17) $Allocated\_host \leftarrow pi$
18) $Max\_ratio \leftarrow R_{RCP}$
19) **end if**
20) **end for**
21) **if** $Allocated\_host \neq$ Null
22) add_allocation (vs, $Allocated\_host$)
23) **end if**
24) **end for**
25) $Allocation =$ get_allocation( )
26) *return Allocation*

on DVFS or workload consolidation techniques. A DVFS based scheduling algorithm is proposed by Wu *et al.* in [18]. The proposed algorithm improves overall resource utilization, leading to higher energy-efficiency. Alnowiser *et al.* [24] offer a DVFS based solution that uses a weighted round robin algorithm to monitor, consolidate, and migrate the hosted VMs. The weighted round robin algorithm provides a consolidation based solution, whereas, DVFS minimizes the energy consumption by matching the processor frequency and voltage. In [25], a DVFS based solution is presented for CPU-intensive Bag-of-Task applications. Moreover, a scheduler is used along with the DVFS solution to reduce overall energy consumption and fulfill the completion deadline. Ren *et al.* [26] provide a game theory based DVFS multi-objective framework that intends to minimize energy consumption. A game theory based module is used for resource management, whereas, DVFS minimizes the energy consumption at the server level.

Along with DVFS based solutions, researchers are also presenting workload consolidation based techniques that reduce the energy consumption by consolidating the workload on the least possible servers. Mertzios *et al.* [27], minimize the energy consumption by consolidating VMs that have overlapping processing times. The consolidation process applies to servers who are hosting the VMs at that specific time. Feller *et al.* [28] provide an energy-efficient ant colony optimization (ACO) based solution that attempts to minimize energy consumption along with basic resource management operations, such as placement, consolidation, and migration. Lee and Zomaya [29] propose two workload consolidation based techniques that provide energy efficiency by increasing processor utilization. The basic difference between both the techniques is of cost function. One of the proposed techniques selects a server based on resource utilization, whereas, the second picks a server by calculating a difference between its real energy consumption and least energy consumption. Addis *et al.* [30] provide an energy-efficient solution based on a hierarchical framework discussed in [31]. The technique is based on two types of managers that are used to manage and maintain the resources on the hand. Central manager (CM) categorizes the available servers based on the class of services and selects a suitable server for the task based on task's class. Alternatively, application managers (AM) perform VM migration, capacity allocation, frequency scaling, and load balancing. In [32], a multi-tier virtualized environment is provided for resource management in cloud environments. Along with the environment, the authors provide a workload prediction mechanism that predicts changes in workload, and based on that prediction; resources are assigned to each VM.

Researchers have also presented SLA-aware energy-efficient solutions that intend to minimize SLA violations along with energy consumption. Researchers in [33] recommend an SLA-aware DVFS solution that provides energy efficiency without violating SLA. Each server is equipped with DVFS module, and hybrid optimization technique is used to handle issues, such as load balancing, resource allocation, and VM placement. A multi-resource energy efficient model is proposed by Li *et al.* in [34]. Proposed model uses a modified particle swarm optimization (PSO) technique for workload consolidation. Moreover, threshold mechanisms are used to handle SLA violations and VM migrations. In [23], Beloglazov *et al*. offer four dynamic threshold mechanisms to detect server overloading. Aforesaid mechanisms are medium absolute deviation (MAD), local regression (LR), interquartile range (IQR), and local regression robust (LRR). One of these mechanisms is used along with workload consolidation technique to avoid server overloading and SLA violations.

Kansal and Chana [35] propose a multi-objective firefly optimization based algorithm that tries to minimize VM migrations and energy consumption. Aforesaid technique transfers the VM that with highest resource

requirements to the server who has lowest resource utilization. Ficco *et al.* [36] present a bio-inspired coral-reefs optimization and game theory based techniques that attempt to ensure SLA-aware cloud resource optimization. Bio-inspired coral-reefs optimization is used to model the elasticity of cloud resources, whereas, game theory is used to optimize the cloud resources on the basis of the user's demand. Zhou *et al.* [37] present resource management techniques that attempt to reduce energy consumption and SLA violation simultaneously. The authors use three thresholds to divide the servers into little loaded, lightly loaded, moderately loaded, and highly loaded categories. VMs are only migrated from highly loaded servers to the less loaded server, whereas, other two categories are not disturbed to prevent SLA violations. Moreover, the authors propose two threshold and three VM selection policies for migrations. Radhakrishnan and Kavitha [38] present a resource management technique that uses genetically weight optimized artificial neural network to minimize energy consumption and VM migrations. The proposed algorithm predicts the available capacity of resources in the near future, and resource management is done on the basis of that prediction. Castro *et al.* [39] attempt to reduce the energy consumption in clouds that is consumed by both CPU and RAM. Two workload consolidation based approaches are proposed by the authors that decide about the hosting server based on its CPU and RAM usage. Along with consolidation techniques, threshold mechanisms are used to reduce the resultant SLA violations.

All the above-mentioned techniques are designed to provide energy efficiency, and minimize the overall expenditure of cloud data centers. However, the performance of these techniques can still be improved as most of the techniques are based on overall CPU capacity, and energy consumption of a server. Moreover, these techniques do not consider the energy that will be consumed for the available CPU capacity or against the resources required by the VM. Furthermore, the majority of aforementioned energy-efficient techniques ignores SLA violations that may be encountered due to workload consolidation. Therefore, to handle these issues, we are proposing new techniques that select a server based on the aforesaid facts. We also present SLA-aware versions of proposed solutions to minimize the SLA violations that are resulted due to workload consolidation. Besides that, we have also enhanced existing energy-efficient solution to reduce the issue of SLA violations.

III. SYSTEM MODEL

The focus of this work is to manage resources in cloud environments. Therefore, we are considering a large data center that uses N heterogeneous servers. Each of the servers is represented by the CPU capacity represented in Millions of Instructions per Second (MIPS), random-access memory (RAM), and network bandwidth [23]. Multiple users can submit requests for an M number of VMs. SLA is agreed with a service provider and in case of SLA violation, service providers pay a penalty. Moreover, network attached storage (NAS) is used to enable live VM migration instead of simple storage. Furthermore, global and local managers are used to manage resources. The local manager is placed on each server and is the part of the virtual machine manager (VMM). Local manager is responsible to keep the track of all the resources on a server, resizing of VM, and VM migration. Whereas, global manager is placed on the master node and keeps the record of overall system resources.

A. MULTI-CORE CPU ARCHITECTURES

In our model, we are using multi-core environment in which each server has n cores with a capacity of m MIPS. Therefore, the capacity of a server can be represented by $n*m$ MIPS. Moreover, we are not using the concept of parallelization to host parts of a VM on two different cores. Therefore, the size of a single VM can be at most equal to the size of a single core.

B. ENERGY MODEL

According to Beloglazov *et al.* [20], power in data centers is mainly consumed by CPU, memory, network interfaces, and disk storage of computing nodes. However, power consumption by servers can be defined by a linear relationship between CPU utilization and power consumption. But recent studies show that introduction of multi-core CPUs and virtualization of resources, force the manufacturers to equip the servers with the large amount of memory. This large amount of memory is dominating the power consumption in servers [23]. Moreover, modeling a precise analytical model for power consumption in aforementioned environment is quite a difficult job. Therefore, as shown in the Table 1, we are using a real power consumption data that is provided by the results of SPECpower benchmark. SPECpower is a benchmark provided by the Standard Performance Evaluation Corporation (SPEC), that evaluates the performance and power of the computers which are categorized as servers. Furthermore, due to variability in workload, the utilization of the CPU may change with time, and is expressed as a function of time $u(t)$. Therefore, the total energy (E) consumed by a server (S) can be expressed as an integral of the power consumption function as shown in the equation below:

$$E = \int_{t_0}^{t_1} P\left(u\left(t\right)\right) dt \tag{1}$$

where, P is power consumed by the server against the CPU utilization (u) at a given time t.

IV. ENERGY EFFICIENT BFD TECHNIQUES

Studies have proven that BFD algorithms are good for online bin packing compared to other similar solutions, and researchers are using this strategy for VM placement. BFD algorithm sorts all the received tasks in descending order, and places one by one on the selected servers. All the BFD based algorithms work in the same way; however, the server selection criteria are different for each algorithm.

TABLE 1. Power consumption of servers under consideration at various loads.

Server	Power Consumption against CPU Utilization										
	Idle	10%	20%	30%	40%	50%	60%	70%	80%	90%	100%
IBM Server x3250	41.6	46.7	52.3	57.9	65.4	73	80.7	89.5	99.6	105	113
HP ProLiant ML110 G4	86	89.4	92.6	96	99.5	102	106	108	112	114	117
HP ProLiant ML110 G5	93.7	97	101	105	110	116	121	125	129	133	135
IBM Server x3550	58.4	98	109	118	128	140	153	170	189	205	222

In this section, we discuss the basic working of all the existing BFD algorithms that are used to provide energy-efficient solutions. The algorithms considered in this study are:

- Modified Best-Fit Decreasing (MBFD)
- Power and Computing Capacity-Aware Best Fit Decreasing (PCABFD)
- Energy-aware and Performance per watt Oriented Best Fit (EPOBF)

A. MODIFIED BEST FIT DECREASING (MBFD)

MBFD [20] is one of the well-known energy-efficient versions of BFD algorithm. MBFD works in the same manner as that of traditional BFD algorithms, however, server selection criterion of both the algorithms is different. Instead of selecting a server based on CPU capacity and power, MBFD selects a server who displays a minimum energy increase after the VM placement. The energy of the server is calculated by using the following equation.

$$P = 0.7 * P_{max} + 0.3 * P_{max} * U \tag{2}$$

Where P is the energy of a server at a given time, P_{max} is maximum power that the server can consume, and U is the utilization of the server. Moreover, to calculate the difference between power consumption before and after VM placement can be calculated by the help of following formula.

$$P_{diff} = P_{AP} - P_{BP} \tag{3}$$

Here, P_{AP} is the power consumed by the server after VM allocation, and P_{BP} is the power before VM allocation. A server with a minimum value of P_{diff} is selected for VM hosting.

B. POWER AND COMPUTING CAPACITY-AWARE BEST FIT DECREASING (PCABFD)

PCABFD [40] is an energy-efficient variant of BFD algorithm that keeps the check on both energy and capacity of the server. A server is selected on the basis of the ratio between power and CPU capacity of the server. A server with the minimum ratio is selected to host a VM. Equation 4 is used to calculate an aforementioned ratio.

$$R = P_{max}/CPU_{max} \tag{4}$$

here, P_{max} is the maximum power, and CPU_{max} is the maximum CPU capacity of the server.

C. ENERGY-AWARE AND PERFORMANCE PER WATT ORIENTED BEST FIT (EPOBF)

Quang-Hung *et al.* [41] present energy-efficient algorithm that selects a server based on CPU capacity and power. The cost function of the algorithm selects a server based on the ratio between total CPU capacity, and energy increase after VM placement. Based on the value calculated with the help of equation 5, the server with maximum ratio is selected to host the VM.

$$R_{E2} = CPU_{max}/P_{diff} \tag{5}$$

where, CPU_{max} is the maximum CPU capacity of the server and P_{diff} is calculated using equation 3.

D. SELECTED TECHNIQUES

An extensive study is carried out to evaluate the performance of selected techniques based upon various workloads. Results show that PCABFD and EPOBF techniques perform better in terms of energy compared to MBFD algorithm. However, energy efficiency of both the techniques can be further improved by the offloading the servers that are underutilized. Presently, both the techniques do not use any kind of mechanism to identify the underutilized servers. Moreover, both techniques lack in terms of SLA-awareness as no mechanism is used to avoid the SLA violations that are incurred due to workload consolidation. Furthermore, network level performance degradation due to excessive migrations is not considered by aforementioned two techniques. Both techniques select a random VM for the migration without considering its consequences on the performance at the network level. Therefore, in this study we are focusing on aforesaid issues and providing enhanced versions of both the techniques that will improve energy efficiency, and reduce SLA performance degradation at server and network levels.

E. ENHANCED TECHNIQUES

Proposed enhanced PCABFD and EPOBF use the lower threshold (LT) mechanism to identify the underutilized servers. Lower threshold mechanism keeps the check on the utilization of the server. As shown in the Figure 1, if the utilization of server is below the set threshold value, all the hosted VMs are migrated to other servers, and offloaded server is switched off. Conversely, an upper threshold (UT) mechanism is used to keep the check on the over-utilized server. UT mechanism avoids the SLA violations by keeping the utilization below a given threshold value, as shown in Figure 2. In case of upper threshold breach, VM or a set

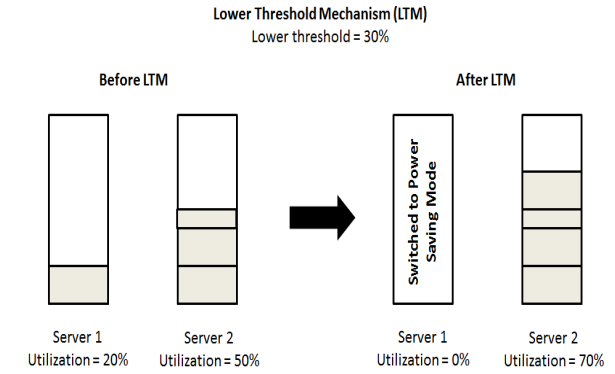

FIGURE 1. Working of lower threshold (LT) mechanism.

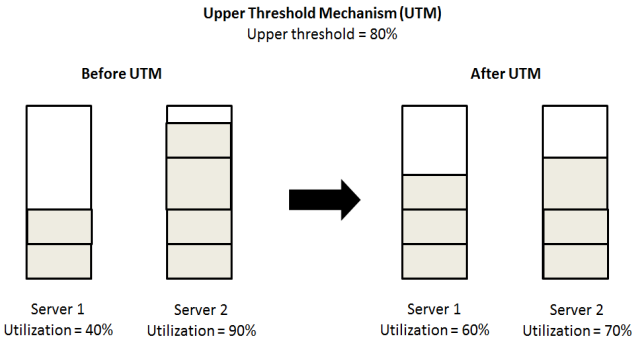

FIGURE 2. Working of upper threshold (UT) mechanism.

of VMs is migrated from the over-utilized server to other servers, and SLA violation is avoided.

To avoid the performance degradation at the network level, minimum migration technique (MMT) policy is used. MMT picks a VM for migration, which needs least time for migration. The selection of such a VM minimizes the load incurred at the network level that is experienced due to migrations. Details of used threshold mechanisms and migration technique are given below.

1) DYNAMIC THRESHOLD MECHANISM

In this study, we are using a median absolute deviation (MAD) based dynamic threshold mechanism to reduce energy consumption and SLA violations. We selected MAD because it is robust and resilient to outliners compared to the standard deviation [23]. Moreover, MAD uses the previous knowledge for the generation of new threshold value. Following mechanism is used to generate a MAD value from the given set of univariate data set $X_1, X_2, \ldots, X_n$.

$$MAD = median_i(|X_i - median_j(X_j)|) \tag{6}$$

here, MAD is the median of absolute values of the deviations of the data's median. However, the threshold (T_u) on the basis of MAD value can be generated by using the following equation.

$$T_u = 1 - s.MAD \tag{7}$$

where $s \in \mathbb{R}^+$ is a safety parameter that is used to control the behavior of the method. If the value of s is low, energy consumption will be minimized, and SLA violations will increase.

2) STATIC THRESHOLD MECHANISM

The static threshold mechanism is used to set static values for the upper and lower threshold mechanism. Static thresholds do not change during the course of the simulation and remain fixed. We have set the lower threshold value to 30% and upper threshold value to 80%. In case of threshold violation, VM(s) will be migrated to other server(s). However, results reveal that static threshold mechanism suffers when dynamic and fluctuating workloads are used.

3) MIGRATION POLICY

We are using MMT to select a VM that consumes minimum time for the migration. Following equation is used to compare the migration time of two VMs. According to [23], migration time depends on the RAM used by the VM, and total bandwidth available for the hosting server. We can use the following equation to compare the migration time of two VMs.

$$v \in V_j \,|\, \forall a \in V_j, \frac{RAM_u(v)}{BW_j} \leq \frac{RAM_u(a)}{BW_j} \tag{8}$$

where, V_j is set of VMs hosted on server j, $RAM_u(v)$ is the RAM currently used by VM v, $RAM_u(a)$ is the RAM currently used by VM a, and BW_j is the available bandwidth for server j.

V. PROPOSED TECHNIQUES

In recent times, virtualization has become an integral part of the cloud data center. It enables service providers to share and assign resources on-demand. VMs can be logically resized and consolidated if they are not using all the assigned resources. This can help in minimizing energy consumption by switching the idle server to sleep mode. Moreover, it also enables cloud data centers to perform live migration of VMs from one server to another. However, the majority of current resource allocation techniques designed for cloud data centers provide high performance by meeting SLAs and do not consider energy efficiency. Therefore, there is a need to devise techniques that consider both performance and energy efficiency while performing resource management.

A. AVAILABLE CAPACITY AND POWER BASED TECHNIQUE (ACP)

Our proposed ACP algorithm is SLA-aware energy-efficient RM technique that minimizes both energy consumption and resultant SLA violations. ACP algorithm is based on PCABFD algorithm and uses the same steps to place a VM on a selected server. However, the server selection criterion of ACP is different as compared to previous BFD algorithms. In the proposed technique, a server is selected on the basis of available CPU capacity and the energy that will be consumed

for that capacity. However, existing BFD techniques are based on overall CPU capacity and maximum energy consumption. Existing techniques are totally ignoring the current state of the servers, yet resulting in more energy consumption. Equations 9 and 10 are used to select the hosting server.

$$R_{ACP} = CPU_{Available}/(P_{max} - P_{BP}) \quad (9)$$

$$CPU_{Available} = CPU_{Max} - CPU_{Utilized} \quad (10)$$

here, $CPU_{Available}$ is the available CPU capacity of the server, and it can be calculated by using equation 10. P_{max} is the maximum power that a server can consume, and P_{BP} is the power consumption of the server before hosting the respective VM. Server with maximum R_{ACP} value will be selected for VM hosting. We are also using the threshold based mechanism to avoid SLA violations that are incurred due to limited available resources. The pseudocode of ACP is presented in Algorithm 1.

In the proposed algorithm, initially all VMs $v \in V$ are sorted in descending order based on CPU requirements and stored in the sorted list V_s'. Similarly, the list of servers is sorted in descending order based upon the CPU capacity, and are stored in S_i'. In the next step, for each VM $vs \in V_s$', the following steps are repeated. A server is selected from the list, and for each server, we check if the cumulative value of *pi's* CPU utilization and $comp\_req_s$ is below the given threshold. If the cumulative value is below the threshold, then the server can host the VM, and we check server's available CPU capacity, maximum CPU capacity, peak energy consumption, and current energy consumption of the server. Aforementioned details of the server are saved in variables $Available\_CPU\_cap_i$, $Max\_CPU\_cap_i$, $Peak\_Energy_i$ and $Current\_Energy_i$ so that they can be used to calculate R_{ACP} by using equation 9. We also keep the check that VM should be placed on a used server, and if no used server is available, then an unused server should be used. After calculating the R_{ACP}, we compare *Max_ratio* and if R_{ACP} is higher than the *Max_ratio*, we assign the value of R_{ACP} to *Max_ratio*, and set the server *pi* as a *Allocated_host*. The aforementioned steps are repeated until all the VMs are assigned.

B. REQUIRED CAPACITY AND POWER BASED TECHNIQUE (RCP)

RCP is also similar to ACP and uses the same mechanism to place VMs on servers. However, RCP selects a server that uses minimum energy against the requested CPU capacity by the VM, as shown in equation 11. None of the existing algorithms consider the demand and resultant energy consumption while selecting a server. Moreover, if the value of the ratio for multiple servers is same, then a server with minimum energy consumption is selected for the hosting of VM. In previous algorithms, there was no such mechanism and any server that meets the selection criteria can be selected for the VM hosting.

$$R_{RCP} = CPU_{Required}/(P_{AP} - P_{BP}) \quad (11)$$

here, $CPU_{Required}$ is the capacity of the CPU that VM demands, P_{BP} is the power consumed by the server before the hosting of the VM, and P_{AP} is the power that server will consume after the hosting of VM. Algoithm 2 shows the pseudocode of RCP.

In the proposed algorithm, initially all VMs $v \in V$ are sorted in descending order based on CPU requirements and stored in the sorted list V_s'. Similarly, the list of servers is sorted in descending order based on the CPU capacity, and are stored in S_i'. In the next step, for each VM $vs \in V_s$', the following steps are repeated. A server is selected from the list, and for each server we check if the cumulative value of *pi's* CPU utilization and $comp\_req_s$ is below the given threshold. If the cumulative value is below the threshold, then the server can host the VM, and we check server's available CPU capacity, maximum CPU capacity, peak energy consumption, and current energy consumption of the server. Aforementioned details of the server are saved to in variables $Available\_CPU\_cap_i$, $Max\_CPU\_cap_i$, $Before\_Alloc\_Energy_i$ and $After\_Alloc\_Energy_i$, so that they can be used to calculate R_{RCP} by using equation 11. We also keep the check that VM should be placed on a used server, and if no used server is available, then an unused server should be used. After calculating the R_{RCP}, we compare *Max_ratio* and if R_{RCP} is higher than the *Max_ratio*, we assign the value of R_{RCP} to *Max_ratio* and set the server *pi* as a *Allocated_host*. The same steps are repeated for all the servers and a server with maximum R_{RCP} is selected to host the VM.

C. COMPLEXITY ANALYSIS OF PROPOSED ACP AND RCP ALGORITHMS

1) TIME COMPLEXITY OF ACP AND RCP ALGORITHMS

The time complexity of proposed ACP and RCP algorithms is calculated as follows. First of all, algorithms sort the received VMs in descending order based on their respective CPU requirement. If we have n number of VMs then the time complexity will be $O(n \cdot log(n))$. In the second step, the servers are sorted in descending order based on their CPU utilization. Therefore, the time complexity of sorting m servers is $O(m \cdot log(m))$. The outer loop will have n iteration to place the VMs on the server. Moreover, in case of best-case scenario, the inner loop will execute u times as each time a used server will be available to host a server. However, in a worst-case scenario, the inner loop will execute m number of times due to non-availability of used servers. Therefore, best and worst case time complexities of both the algorithm are given in equation 12 and 13 respectively.

$$O(\left[n\,(\log\,(n) + u)\right] + m \times \log(m)) \quad (12)$$

$$O(\left[n\,(\log\,(n) + m)\right] + m \times \log(m)) \quad (13)$$

2) SPACE COMPLEXITY OF ACP AND RCP ALGORITHMS

Space complexity of ACP and RCP algorithms is calculated using equation 14.

$$O(2n + m) \quad (14)$$

TABLE 2. Time and space complexities of selected and proposed techniques.

Resource Management Technique	Time Complexity		Space Complexity
	Best Case Scenario	Worst Case Scenario	
ACP	$O([n(\log(n) + u)] + m \times \log(m))$	$O([n(\log(n) + m)] + m \times \log(m))$	$O(2n + m)$
RCP	$O([n(\log(n) + u)] + m \times \log(m))$	$O([n(\log(n) + m)] + m \times \log(m))$	$O(2n + m)$
MBFD	$O([n(\log(n) + m)] + m \times \log(m))$	$O([n(\log(n) + m)] + m \times \log(m))$	$O(2n + m)$
PCA-BFD	$O([n(\log(n) + m)] + m \times \log(m))$	$O([n(\log(n) + m)] + m \times \log(m))$	$O(2n + m)$
EPOBF	$O([n(\log(n) + m)] + m \times \log(m))$	$O([n(\log(n) + m)] + m \times \log(m))$	$O(2n + m)$

here, n is the number of VMs to be placed, and m is the total number of servers. The space complexity of ACP and RCP algorithms is very moderate as only require a list of size n to store VMs in sorted order and a list of size m to store sorted list of servers. Moreover, we use a list of size n to store the VM to server mapping. Time and space complexities of all the selected techniques are presented in Table 2.

VI. FORMAL VERIFICATION

The procedure of validating the accuracy of the system under consideration is known as verification. Correctness of the system can be determined on the basis of two parameters: (i) properties and (ii) specifications [42]. To verify our proposed techniques, we are using bounded model checking technique [43] along with Z3 solver and SMT-Lib.

A. SMT-LIB AND Z3 SOLVER

Formal verification and automated reasoning about decision problems can be well solved and coded by the help of Satisfiability Modulo Theories (SMT) [44]. SMT makes the logical formula of decidability problem and on the basis of decidability background theory, decides its satisfiability. However, Boolean Satisfiability Solvers (SAT) is also used for the same purpose, but it is propositional satisfiability solver. SMT solvers support several theories, such as, equality and un-interpreted functions, linear arithmetic over rationals ($LA\mathbb{Q}$), linear arithmetic over integers ($LA\mathbb{Z}$), non-linear arithmetic over reals (*NLA*), over arrays (*AR*), bit vectors, and combinations. Moreover, for the verification of systems, SMT-Lib provides an input platform for the various solvers Satisfiability modulo theories: introduction [44]. An abstract model is developed to represent the behavioral specifications of the system, and with the help of SMT solvers, bounded model checking is applied to the aforesaid model to check its bounded symbolic execution [44]. There are some solvers that provide a support for SMT-Lib, namely, Z3, CVC4, OpenSMT, Beaver, MathSAT5, and Boolector. The basic difference between these solvers can be at the interface level, logic level, theory level, or at the level of input formulae [45].

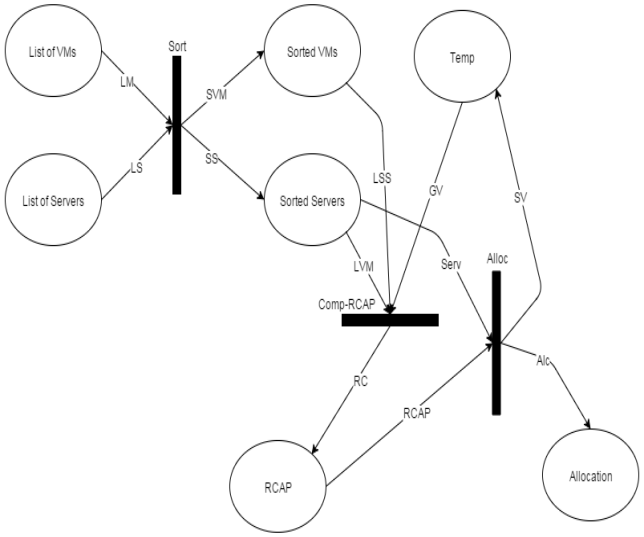

FIGURE 3. HLPN model for the proposed resource management algorithms.

B. HIGH-LEVEL PETRI NETS (HLPN)

HLPN are used for the graphical and mathematical modeling of systems, such as stochastic and parallel [46]. HLPN help with systems simulations, and analysis of system behavior and structural properties through the mathematical representation. HLPN are used to analyze the processing of information, interconnection between various procedures and components, and information flow between the procedures. Therefore, in this study, we are using a HLPN for the formal verification of our proposed techniques. HLPN is represented by a set of seven tuples, such as $N = (P, T, F, \emptyset, R, L, M_0)$, where:

1) A set of finite places is represented by P.
2) T represents set of finite transitions, such that $P \cap T = \emptyset$.
3) F shows the flow relation between a place and transition or a transition and place. Represented as $F \subseteq (P \times T) \cup (T \times P)$.
4) $\emptyset$ highlights the mapping of places to data types, and can be represented as $\emptyset : P \rightarrow data\ types$.
5) R represents the rules that are used to map the transitions to logical formulae, such as: $R : T \rightarrow formula$.
6) L are the labels that are used along with flows, and are represented as $L : F \rightarrow label$.
7) M_0 is the initial state from where the flow starts and is shown as $M : P \rightarrow tokens$.

C. MODELING AND ANALYSIS OF PROPOSED ALGORITHM

The HLPN model of proposed RM algorithms is illustrated in Figure 3. To begin with, the system modeling, we first need to define the set of places P and the corresponding data types. Figure 3 shows that there are seven places in the model, whereas, Table 3 illustrates the places and the

TABLE 3. Places and data types mappings.

Places	Mappings
φ (List of Servers)	$(SID \times CPU_{Cap} \times P_{EN} \times C_{EN} \times CPU_{avlb})$
φ (List of VMs)	$(VMID \times CPU_{req})$
φ (Sorted VMs)	$(VMID \times CPU_{req})$
φ (Sorted Servers)	$(SID \times CPU_{Cap} \times P_{EN} \times C_{EN} \times CPU_{avlb})$
φ (RACP)	$(R_{AP} \times SID \times VMID)$
φ (Temp)	$(Max - ratio \times AllocatedHost \times Threshold)$
φ (Alloc)	$(SID \times VMID)$

corresponding data types' mappings. Now, we will define rules, pre-conditions, and post-conditions for the given model.

$$\begin{aligned} R\,(Sort) &= \forall ls \in LS, \forall lm \in LM, \forall ss \in SS\, \forall svm \in SVM| \\ ss\,[1] &:= Sort\,(ls\,[1], ls\,[2]) \wedge ss\,[2] \\ &:= Sort\,(ls\,[1], ls\,[2]) \wedge \\ SS' &= SS \cup \{(ss\,[1], ss\,[2], ss\,[3], ss\,[4], ss\,[5]) \wedge \\ svm\,[1] &:= Sort\,(lm\,[1], lm\,[2]) \wedge svm\,[2] \\ &:= Sort\,(lm\,[1], lm\,[2]) \wedge \\ SVM' &= SVM \cup \{(svm\,[1], svm\,[2]) \end{aligned} \tag{15}$$

The rules (as in (15)) which are mapped on transition $R\,(Sort)$ explain the process when the servers, and the VM(s) are sorted in descending order according to the utilization and requirements, respectively. Once the serves and the VM(s) are sorted, we compare the utilization of the servers and the computational requirements of the VM with the selective threshold (as depicted in (16)) of transition $R(Comp{-}RCAP)$). If the aforesaid result is less than the threshold, then we compute the R_{ACP} value of the selected server.

$$\begin{aligned} R\,(Comp - RCAP) &= \forall lss \in LSS, \forall lvm \in LVM, \forall gv \\ &\quad \in GV, \forall rc \in RC| \\ lss\,[2] + lvm\,[2] &< gv\,[3] \Longrightarrow rc\,[1] \\ &:= \frac{lss\,[5]}{lss\,[3] - lss\,[4]} \wedge rc\,[2] \\ &:= lss\,[1] \wedge rc\,[3] := lvm[1] \wedge RC' \\ &= RC \cup \{(rc\,[1], rc\,[2], rc\,[3])\} \end{aligned} \tag{16}$$

In (17), the rules that are mapped on transition $R\,(Alloc)$ compare the R_{ACP} value generated in (16) with Max_ratio. If the result of R_{ACP} value is less than the Max_ratio, then the servers are selected, and the values are updated accordingly. Finally, the VM(s) is allocated to the servers.

$$\begin{aligned} R\,(Alloc) &= \forall rca \in Rcap, \forall sv \in SetVar, \forall gv \\ &\quad \in GetVar, \forall alc \in ALC| \\ rca\,[1] &> gv\,[1] \Longrightarrow sv\,[2] \\ &:= rca\,[2] \wedge sv\,[1] := rca\,[1] \wedge \\ SV' &= SV \cup \{(sv\,[1], sv\,[2], sv\,[3])\} \wedge \\ ALC' &= ALC \cup \{(rca\,[1], rca\,[2], rca\,[3])\} \end{aligned} \tag{17}$$

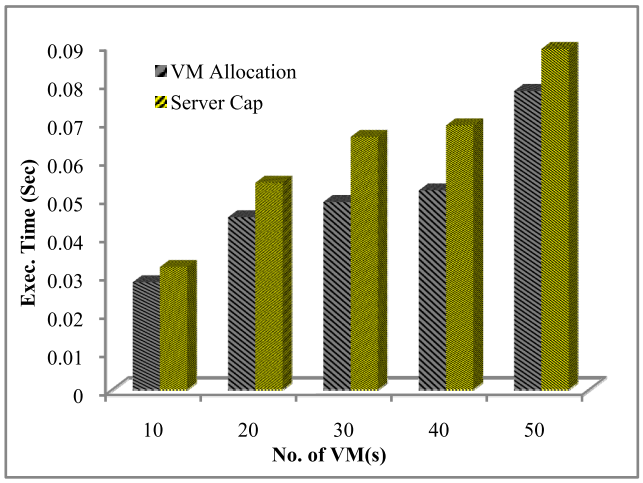

FIGURE 4. Verification time taken by the Z3 solver to prove the properties.

D. MODEL VERIFICATION PROPERTY

In order to highlight the correctness of our HLPN models, we first translated the HLPN models into Z3 Solver and SMT-lib. Moreover, we identified two properties to identify the correctness of our models. After that Z3 solver is used to check whether the model satisfies the properties or not. The Figure 4 shows the time required by the Z3 Solver to verify the correctness of the properties. The first property is *VM Allocation*, whether all the VMs are allocated or not. The second property is *Server Cap*, which inspects that the VM is only allocated to those servers that have the capacity. The solver generates positive results, indicating that the properties are verified, which means the models of the algorithms are working correctly.

VII. EXPERIMENTAL EVALUATION

In this section, we will discuss the performance evaluation of our proposed techniques along with the selected techniques. We have conducted a comparative study with existing state of the art workload consolidation based energy efficient techniques.

A. EXPERIMENTAL SETUP

We are considering a cloud data center for evaluation of our techniques. Though, a large-scale data center would have been a better choice but due to limited resources, we have used a simulation based approach and conducted extensive simulations on CloudSim [47]. CloudSim is a well-known open source simulation tool that is used for the performance evaluation of energy-efficient resource management techniques. CloudSim provides numerous parameters that evaluate the energy efficiency of technique along with SLA-awareness and network performance. In this study, evaluation of proposed techniques carried out on the basis of parameters, such as, energy consumption, performance degradation due to migration and SLA violations. These parameters are discussed in detail in the next section.

TABLE 4. Server configurations.

Server	CPU Model	Number of Cores	Clock Rate (MHz)	RAM (GB)
IBM Server x3550	2 x Intel Xeon 5675	2 x 6	3067	16
IBM Server x3250	Intel Xeon 3470	4	2933	8
HP ProLiant ML110G5	Intel Xeon 3075	2	2660	4
HP ProLiant ML110 G4	Intel Xeon 3040	2	1860	4

TABLE 5. VM configurations.

VM Type	CPU (MIPS)	RAM (GB)
Micro instance	500	0.613
Small instance	1000	1.7
Extra-large instance	2000	3.75
High-CPU medium instance	2500	0.85
High-memory extra large	3000	6

TABLE 6. Workload characteristics.

Workloads	Number of VMs	Median	St. Dev
W1	1052	6 %	17.09 %
W2	898	5 %	16.83%
W3	1061	4 %	15.57%
W4	1516	5 %	12.78%
W5	1078	6 %	14.14%
W6	1463	6 %	16.55%
W7	1358	6 %	15.09%
W8	1233	6 %	15.07%
W9	1054	6 %	15.15%
W10	1033	4 %	15.21%

In this study, we are using a setup of total 800 heterogeneous servers belonging to four types of multi-core servers as shown in Table 4. Each category of servers has different CPU model, clock rate, cores, and RAM. We have taken 200 servers from each category of servers that host VMs similar to Amazon EC2 instances [48]. All the VMs differ in terms of CPU and RAM requirements. High-memory extra-large VMs require the highest amount of CPU and RAM capacity, whereas, micro instance demands minimum capacity as shown in Table 5. In addition to that, we are using real-world workloads for the better evaluation of our techniques. Aforesaid workloads are the actual workloads that were received by PlanetLab on 10 different days [49]. Table 6 represents the details of these workloads that are based on the CPU utilization of 500 worldwide spread servers, hosting more than 1000 VMs at a time.

B. PERFORMANCE METRICS

In order to assess the performance of selected and proposed techniques, we are using following performance evaluation metrics.

1) ENERGY CONSUMPTION

The first metric we consider in our study is the overall energy consumed by data center's servers. For the calculation of energy, an energy model discussed in Section III is used, and total energy consumption is calculated by using equation 1.

2) PERFORMANCE DEGRADATION DUE TO VIRTUAL MACHINE MIGRATIONS (PDM)

We are using this metric to check the performance of our techniques at the network level. *PDM* highlights the effects of migrations on the performance of the network. *PDM* can help us in selecting a migration technique that incurs less load on the network. *PDM* can be calculated by using the following equation.

$$PDM = \frac{1}{M}\sum_{j=1}^{M}\frac{C_{d_j}}{C_{r_j}} \tag{18}$$

where, M is the number of VMs, C_{d_j} is the estimated performance degradation of VM (j) and C_{r_j} is the total capacity demanded by the VM (j) during its execution time.

3) SLA VIOLATION TIME PER ACTIVE HOST (SLATAH)

Another parameter considered in this study is SLA violation time per active host. *SLATAH* is the cumulative during which the server experienced the SLA violations. It can be calculated by using equations 19.

$$SLATAH = \frac{1}{N}\sum_{i=1}^{N}\frac{T_{s_i}}{T_{a_i}} \tag{19}$$

where, N is number of hosts, T_{s_i} is the total SLA violation time during which server's (i) utilization was 100% and T_{a_i} is the total active time of the server (i).

4) SLA VIOLATION (SLAV)

The fourth parameter we are considering in this study is the SLA violation. SLA defines the characteristics of the QoS that should be provided to the customer. We are using *SLAV* metric defined in [20] to calculate the SLA violations that are encountered during the given time. *SLAV* can be calculated by using the equation 20 which depends on *SLATAH* and *PDM*.

$$SLAV = SLATAH * PDM \tag{20}$$

C. SIMULATION RESULTS

All the selected and proposed resource management techniques are designed to minimize energy consumption. However, PCABFD and EPOBF do not consider the SLA violations that result due to workload consolidation. Therefore, as discussed in Section IV, we present the various variants of aforesaid techniques that attempt to increase energy efficiency, and reduce SLA violations and performance degradation that is faced due to excessive migrations. In the next section, performance evaluation of aforesaid variants is carried out based on above-mentioned parameters.

D. PERFORMANCE EVALUATION OF PCABFD AND EPOBF VARIANTS

In this section, we compare the PCABFD and EPOBF techniques with the variants we have proposed in this study. Details of all the techniques are presented in Table 7.

TABLE 7. Variants of PCABFD and EPOBF.

Technique Name	VM Allocation Technique	Migration Technique	Lower Threshold	Upper Threshold
PCABFD	PCABFD	RS	No	No
EPOBF	EPOBF	RS	No	No
PCABFD/ MMT	PCABFD	MMT	No	No
EPOBF/ MMT	EPOBF	MMT	No	No
EPCABFD	PCABFD	MMT	Yes	No
EEPOBF	EPOBF	MMT	Yes	No
SEPCABFD	PCABFD	MMT	Yes	Yes
SEEPOBF	EPOBF	MMT	Yes	Yes

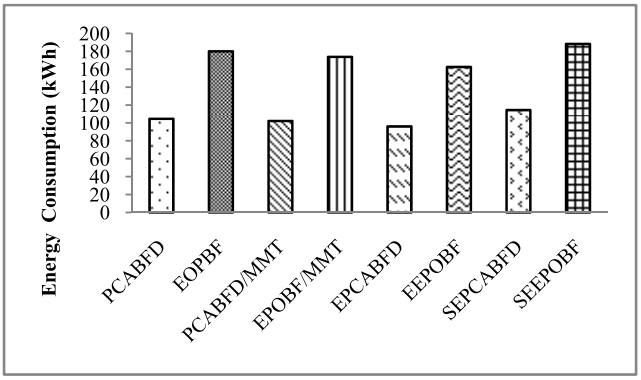

FIGURE 5. Energy consumption of various variants of PCABFD and EPOBF.

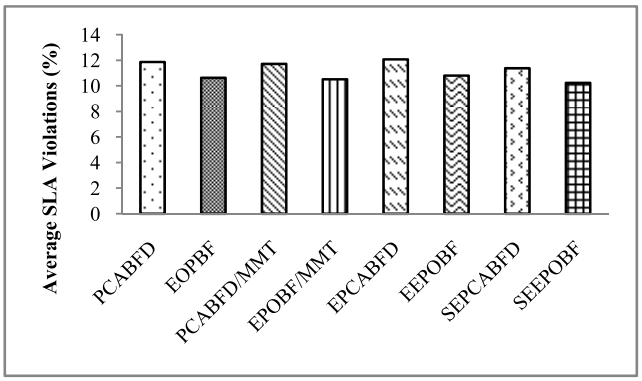

FIGURE 6. Average SLA violations of various variants of PCABFD and EPOBF.

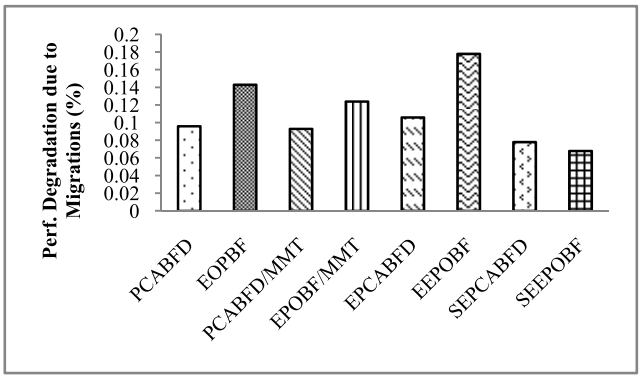

FIGURE 7. Figure 7: Performance degradation due to migrations of various variants of PCABFD and EPOBF.

1) ENERGY CONSUMPTION

Figure 5 presents the energy consumption comparison of PCABFD, EPOBF and their variants. The results highlight that proposed energy-efficient variants outperformed the original techniques. It can be seen that EPCABFD is the most energy efficient whereas SLA-aware version of EPOBF (SEEPOBF) is the least energy-efficient variant among the said techniques. EPCABFD consumes about 8% less energy than original PCABFD and 16% less compared to its SLA-aware version (SEPCABFD). Moreover, it consumes almost 47% less energy compared to EPOBF and 41% less than EEPOBF. On the other hand, EEPOBF consumes 10% less energy than EPOBF and 14% less than SEEPOBF. Furthermore, most energy-efficient technique (EPCABFD) utilizes 49% less energy in comparison to the least energy-efficient technique, i.e., SEEPOBF.

2) AVERAGE SLA VIOLATIONS

Average SLA violations of the variants under discussion are shown in Figure 6. It can be clearly seen that SLA-aware versions of the techniques perform better than their respective counterparts. SEEPOBF has the least SLA violations, whereas EPCABFD has highest SLA violations. When we compare both of them, SEEPOBF has at least 15% less violation compared to EPCABFD. Moreover, SLA-aware version of EPOBF has 5.5% fewer violations than EEPOBF and about 4% less compared to original EPOBF. In comparison to SLA-aware version of SEPCABFD, SEEPOBF has 10% fewer violations. It is evident from the discussed results that techniques that use upper thresholds perform better in terms of SLA violation due to the availability of some free resources.

3) PERFORMANCE DEGRADATION DUE TO MIGRATION

Figure 7 shows the performance degradation due to migrations. The results show that due to better VM selection criteria, MMT helps in minimizing the performance degradation. Moreover, the upper threshold mechanism also assists in the reduction of performance degradation. The upper threshold decreases the SLA violations which ultimately reduce the need to migrate the VMs. It can be seen from the results that SEEPOBF clearly outperforms other techniques in terms of the said metric, whereas, its energy efficient counterpart EEPOBF displays the worst results. SEEPOBF has 13% less performance degradation compared to second best SEPCABFD, and 52% less than original EPOBF. Moreover, if we compared the best and worst techniques, then SEEPOBF has almost 62% less performance degradation compared to EEPOBF.

4) SLA VIOLATION TIME PER ACTIVE HOST

Figure 8 shows the time during which active hosts faced the SLA violations. The results highlight that SLA-aware

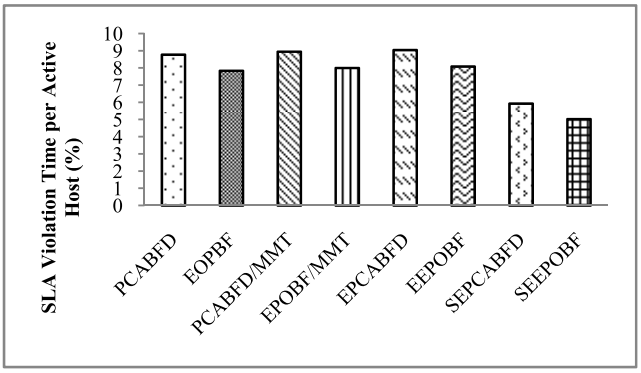

FIGURE 8. SLA violation time per active host of various variants of PCABFD and EPOBF.

TABLE 8. Details of evaluated techniques.

Technique Name	VM Allocation Technique	Migration Technique	Lower Threshold	Upper Threshold
EPCA	PCABFD	MMT	Yes	No
EEPOBF	EPOBF	MMT	Yes	No
ACP	ACP	MMT	Yes	No
RCP	RCP	MMT	Yes	No
SEPCA	PCABFD	MMT	Yes	Yes
SEEPOBF	EPOBF	MMT	Yes	Yes
SACP	ACP	MMT	Yes	Yes
SRCP	RCP	MMT	Yes	Yes
MBFD	MBFD	MMT	Yes	Yes

variants perform well due to less SLA violations. SEEPOBF is best in terms of both SLA violations and SLATAH, whereas, EPCABFD has the highest violation time. It can be noticed from the Figure 8 that SEEPOBF has 15.5% less violation time compared to SEPCABFD, whereas, it has 36% and 38% less violation time compared to EEPOBF and EPOBF respectively. If we compare the other SLA-aware technique, SEPCABFD has 34% less violation time compared to EPCABFD and 32% fewer violations compared to PCABFD. It is evident from the results that SLA-aware versions perform better than their counterparts.

E. PERFORMANCE EVALUATION OF ACP AND RCP ALGORITHMS

In this section, we will be discussing the results of selected energy efficient and SLA-aware RM techniques, along with our proposed ACP and RCP techniques. All the proposed and selected techniques analyzed in this section use both energy consumption and CPU capacity as selection criteria. Selected techniques that are being used for this evaluation are already discussed in the previous section. EPCA and EEPOBF are the enhanced energy efficient versions of PCA and EPOBF techniques. Whereas, SEPCA and SEEPOBF are the SLA-aware enhanced versions of both the techniques. On the hand, ACP and RCP are the energy efficient versions of the proposed techniques, whereas, the SACP and SRCP are the SLA-aware versions. Details of each technique are presented in Table 8. Moreover, to analyze the impact of static and dynamic thresholds on RM, aforesaid techniques are evaluated on the basis of these thresholds. For static thresholds, lower threshold is set to 30% and upper threshold is set to 80%, whereas, dynamic threshold uses the MAD. The results of aforesaid techniques based on selected performance evaluation parameters are discussed below.

1) ENERGY CONSUMPTION

Figure 9 presents the energy consumption of selected techniques based on static and dynamic thresholds. The results show that our proposed ACP and RCP techniques outperform other selected techniques in terms of energy consumption. Both techniques use LT mechanism to identify and offload the underutilized nodes. Moreover, both the techniques select a server based on its real-time energy and resource utilization, which improves the decision making during VM placement. On the other hand, existing techniques only consider the maximum energy consumption and CPU capacity during VM placement phase. Lack of updated resource information results in poor VM placement that leads to higher-energy consumption. Moreover, our SLA-aware versions of the proposed techniques consume more energy compared to their counterparts because of the use of UT mechanism that keeps some of the resources free. Hence, forcing the system to use more servers for VM hosting.

The results of the Figure 9 show that our proposed RCP technique is best in terms of energy consumption, whereas, ACP is second best and our proposed energy efficient variant of PCA (EPCA) with slight difference is at third best. RCP consumes about 54%, 46%, 9% and 5% less energy compared to MBFD, EEPO, EPCA and ACP respectively. In comparison to SLA-aware RCP, RCP consumes 14% less energy than its SLA-aware counterpart. On the other hand, ACP consumes 51%, 43%, 15%, and 4% less energy compared to MBFD, EEPO, SACP and EPCA respectively. If we compare the impact of the static and dynamic threshold mechanisms on the energy consumption, it can be noted that the difference ranges between 3% to 6%. Therefore, it can be said that dynamic thresholds also have a positive impact on energy efficiency. Dynamic threshold performs better than the static threshold because they are not fixed and keep on changing, therefore, if the SLA violations remain low, upper threshold can be increased. This allows the server to host more VMs which leads to reduction in the number of active server and energy consumption.

Figure 9 shows the maximum, average and minimum energy consumption values of each techniques, whereas, Figure 10 shows the standard devation of energy consumption based on the workloads which are used to evaluate the selected and proposed techniques. It is evident from the Figure 10 that energy efficient versions have the slightly better results compared to SLA-aware

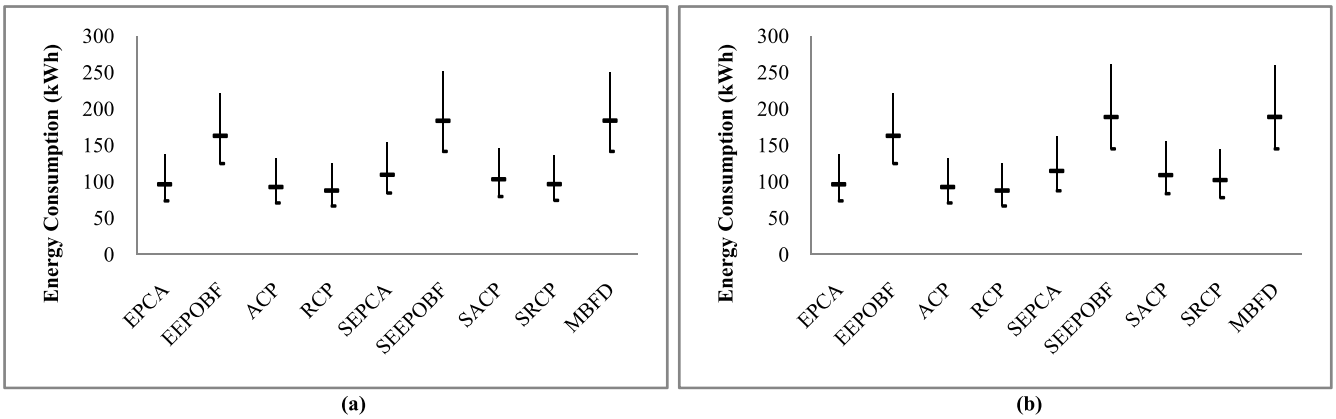

FIGURE 9. Energy consumption (a) Max, average and min of all workloads with MAD based thresholds, (b) Max, average and min of all workloads with static thresholds.

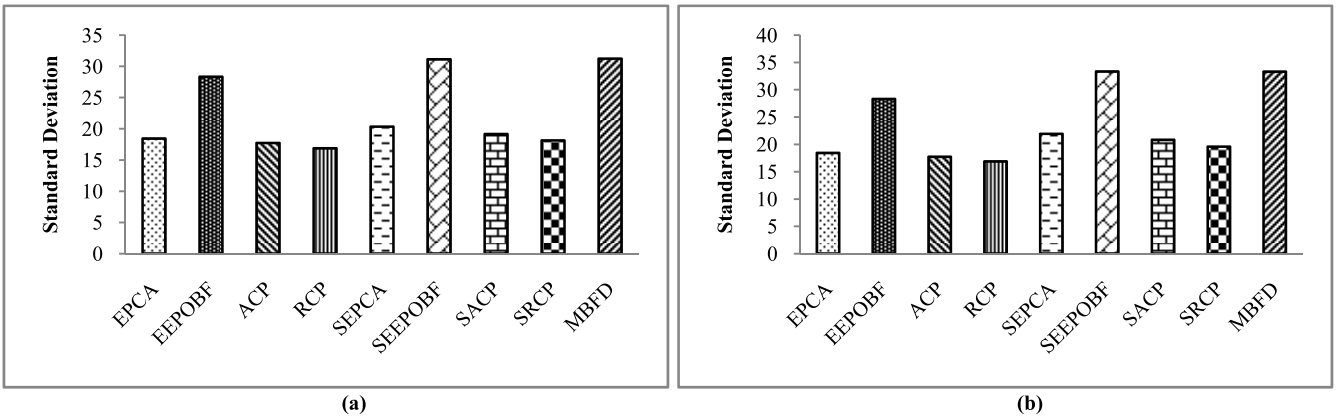

FIGURE 10. Standard deviation of energy consumption on (a) All workloads with MAD based thresholds, (b) All workloads with static thresholds.

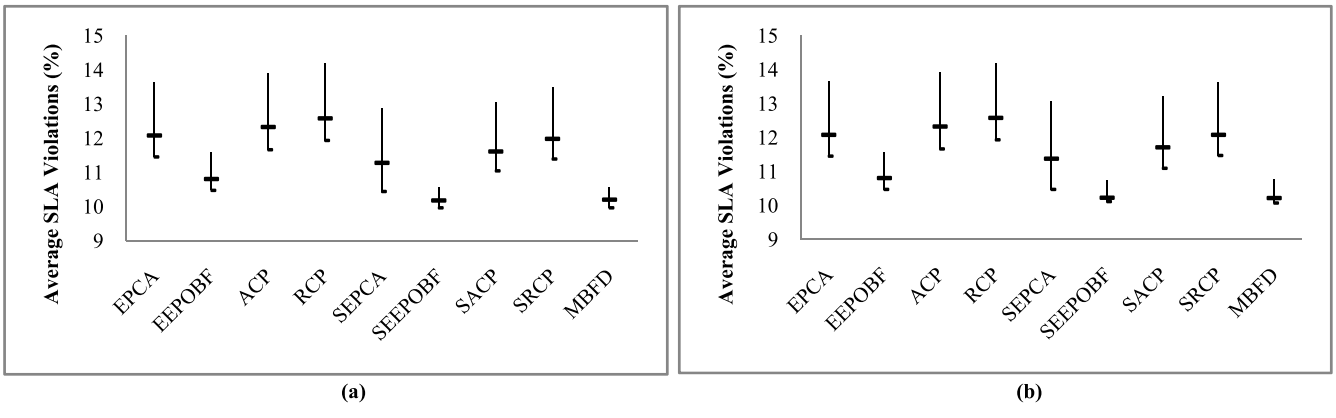

FIGURE 11. Average SLA violations (a) Max, average and min of all workloads with MAD based thresholds, (b) Max, average and min of all workloads with static thresholds.

counterparts. Moreover, the variation in the values of EPCA, ACP, RCP, SEPCA, SACP and RCP, are less compared to EEPOBF, SEEPOBF and MBFD algorithms. It shows that in terms of energy, proposed and PCA based solutions are more stable than the MBFD and EPOBF based solutions.

2) AVERAGE SLA VIOLATIONS

Figure 11 shows the performance comparison of the proposed and selected techniques based on SLA violations. Techniques like SEEPOBF and MBFD that do not perform workload consolidation properly, and use more servers have enough resources to accommodate the fluctuating demands of VMs.

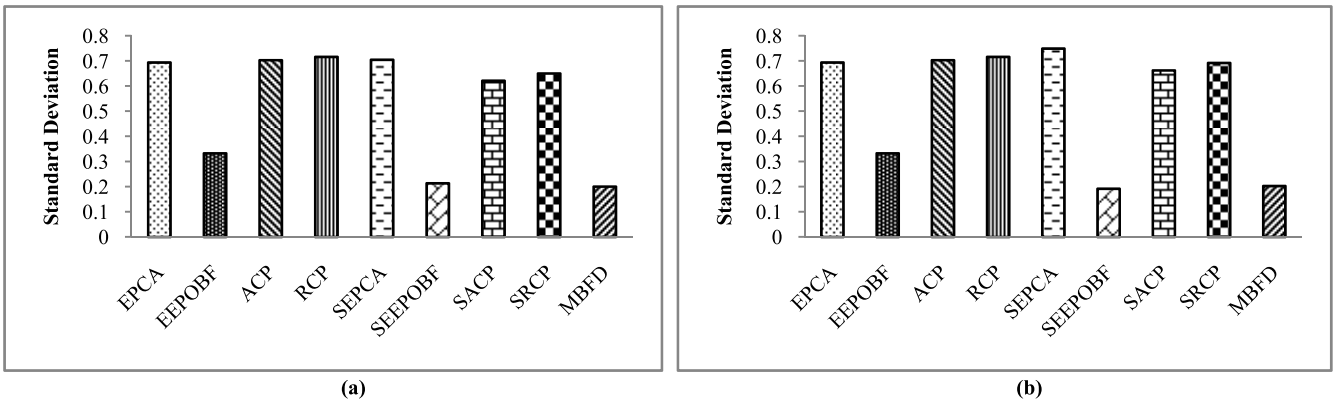

FIGURE 12. Standard deviation of average SLA violations on (a) All workloads with MAD based thresholds, (b) All workloads with static thresholds.

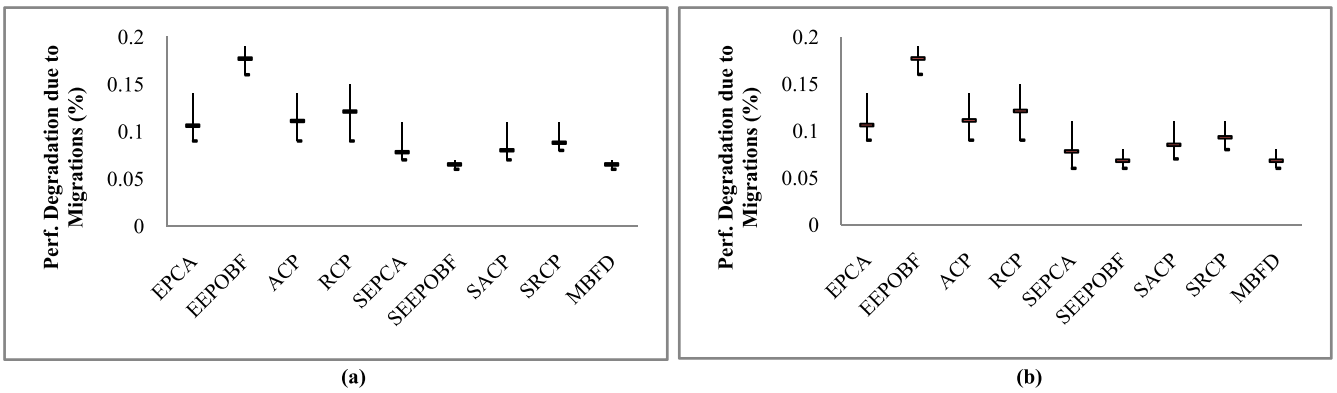

FIGURE 13. Performance degradation due to migrations (a) Max, average and min of all workloads with MAD based thresholds, (b) Max, average and min of all workloads with static thresholds.

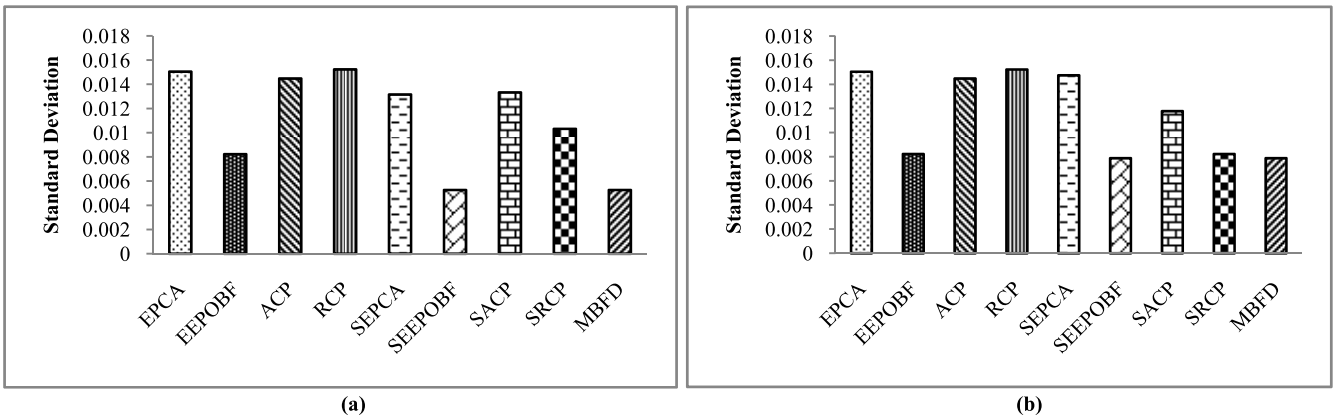

FIGURE 14. Standard deviation of performance degradation due to migrations on (a) All workloads with MAD based thresholds, (b) All workloads with static thresholds.

The fact that can be noticed from Figure 11 is the impact of dynamic and static thresholds on selected and proposed techniques. It can be clearly seen that dynamic threshold mechanism performs well in terms of SLA violations due to varying nature. If the higher SLA violations have occurred in previous rounds, then the value of the threshold will be set low, and SLA violations will be avoided. A static threshold mechanism does not have such intelligence and remains static throughout the course of the simulation.

It is evident from the Figure 11 that MBFD and SEPOBF have approximately the same amount of SLA violations. However, they outperform other techniques. MBFD has 17%, 14%, 11%, and 5.5% fewer SLA violations compared to SRCP, ACP, SEPCA, and EEPOBF respectively. Moreover, SLA-aware versions of the proposed and selected techniques perform better than their energy-efficient counterparts in terms of SLA violations. SACP reduces the SLA violations by 5% compared to ACP, whereas, SRCP has 4%, fewer SLA

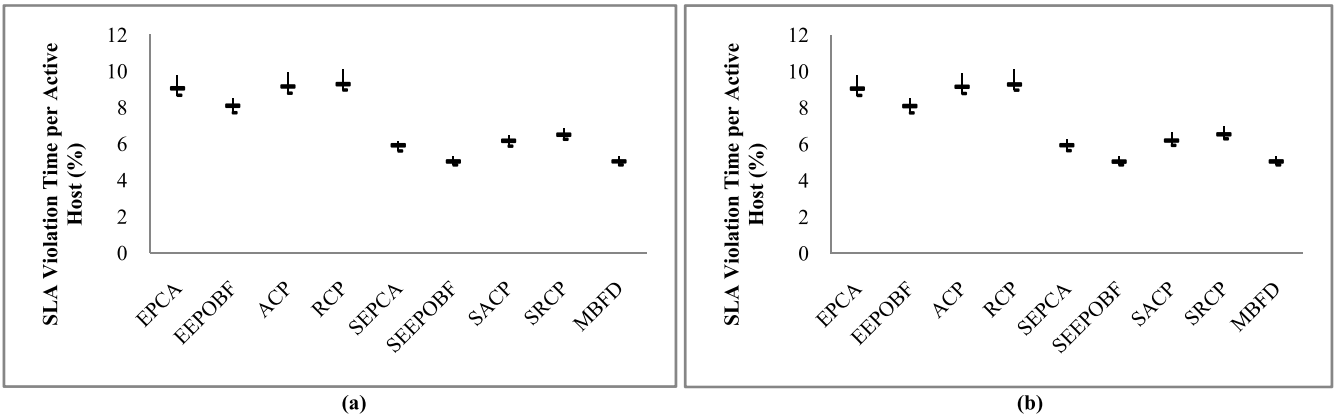

FIGURE 15. SLA violation time per active host (a) Max, average and min of all workloads with MAD based thresholds, (b) Max, average and min of all workloads with static thresholds.

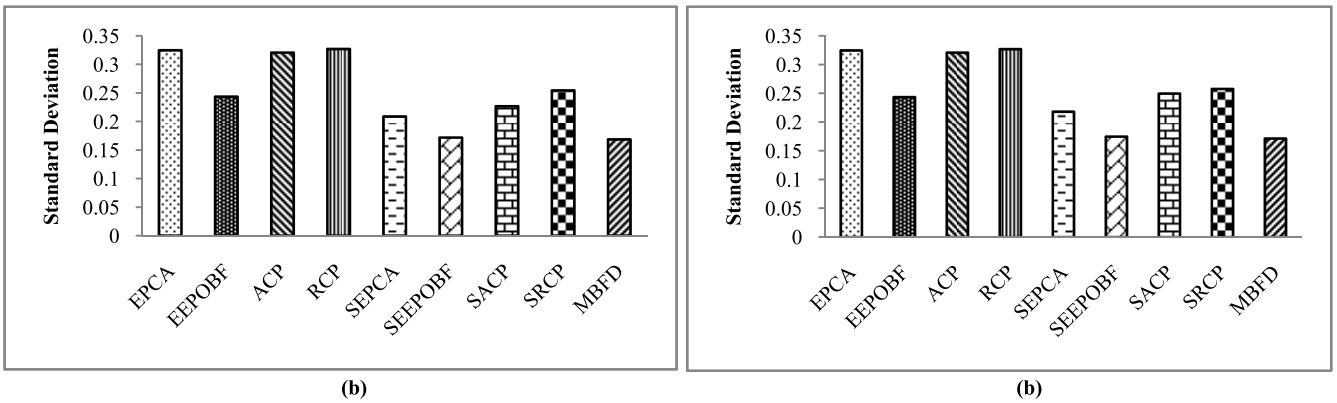

FIGURE 16. Standard deviation of SLA violation time per active host on (a) All workloads with MAD based thresholds, (b) All workloads with static thresholds.

violations compared to RCP. Similarly, SEPCA has 6% fewer SLA violations compared to energy-efficient EPCA. Furthermore, if we compare the SLA violations of SLA-aware techniques based on static and dynamic UT, the difference of SLA violations ranges between 0.5% to 1%.

The maximum, average and minimum percentages are shown in Figure 11, whereas, Figure 12 shows the standard deviation of average SLA violations based on different workloads. The results show that in terms of SLA violations, EEPOBF, SEEPOBF and MBFD algorithms outperform their counterparts. Other solutions perform well in terms of energy consumption, but they suffer in terms of SLA violations due to aggressive workload consolidation. Moreover, SLA-aware versions of the techniques perform slightly better than their energy efficient versions.

3) PERFORMANCE DEGRADATION DUE TO MIGRATION

Comparison of selected techniques on the basis of performance degradation due to migrations is presented in Figure 13. The results highlight that MBFD and SEEPOBF have minimum performance degradation due to migrations. MBFD has 27% less performance degradation compared to SRCP, whereas, it has 20% and 13% less performance degradation compared to the SACP and SEPCA respectively. Moreover, SEEPOBF decreases the performance degradation significantly compared to the energy-efficient counterparts EEPOBF. SEEPOBF has 61% less performance degradation compared to EEPOBF, whereas, SEPCA has 26%, the SACP has 24% and SRCP has 23% compared to their respective energy-efficient versions. Furthermore, if we compare the results of the techniques that are using static and dynamic threshold mechanisms, the difference is between 5% to 6%.

Although the maximum, average and minimum values are evident from the Figure 13, but the variation from the mean value is shown in the Figure 14. It can be noticed from the Figure 14 that EEPOBF, SEEPOBF and MBFD algorithms have less variation compared to other counterparts due to less aggressive workload consolidation and SLA violations. Moreover, if we compare the SLA-aware and energy efficient techniques, it can be seen that SLA-aware versions have less variation than energy efficient versions.

4) SLA VIOLATION TIME PER ACTIVE HOST

Figure 15 shows the SLA violation time per active host of the selected and proposed techniques. Results show that SLA-aware versions of the techniques outperform the energy efficient techniques due to less SLA violations. It is evident from the results that SEEPOBF and MBFD are best in terms of SLA violation time. SEEPOBF is slightly better than MBFD algorithm; however, it has 15%, 18% and 23% less violation time compared to SEPCA, SACP, and SRCP,

respectively. Moreover, compared to its energy efficient version, it has 38% less SLA violation time. Compared to worst in term of violation time, SEEPOBF has about 46% less violation time compared to RCP. Results further highlight that the SLA-aware versions of all the techniques perform better than their respective energy efficient versions in terms of SLA violations time. If we compare the results of techniques on the basis of dynamic and static thresholds, it is noticed that dynamic threshold based SLA-aware techniques perform better than a static threshold based techniques. The reason behind the better performance is dynamically changing values of CPU utilization thresholds.

The standard deviation of the SLA violation time is presented in Figure 16. The results show that SLA-aware versions are more stable than energy efficient versions due to better violation avoidance. Moreover, EEPOBF, SEEPOBF and MBFD algorithms have less variations compared to their respective counterparts. Results indicate that the upper threshold mechanism and free resources efficiently handle the issue of SLA violations.

VIII. CONCLUSIONS

In this paper, we jointly handle the issue of energy consumption and SLA violation that a service provider faces while providing resources in a cloud environment. Our work provides an in-depth overview of existing BFD based energy-efficient algorithms and evaluates their performance. In addition to that, we enhanced the existing energy-efficient techniques to further improve the energy efficiency. Moreover, SLA-awareness is also introduced in the selected techniques, and results show that SLA violations are reduced significantly. Furthermore, new resource management techniques proposed in this paper reduce energy consumption and SLA violations by considering the real time states of energy and CPU capacity of servers. Real time states assist in better selection of server which led to reduction in energy consumption and SLA violations.

We have also concluded that the dynamic threshold mechanisms play a key role in enhancing the performance of resource management techniques. MAD based dynamic threshold mechanism performs better than the static threshold mechanism in terms of all the parameters, such as, energy consumption, performance degradation due to migration, and SLA violations. The reason behind the better performance of the dynamic threshold mechanism is the use of previous data, and intelligently changing the values based on the changing behavior of workloads. We have also concluded that migration technique helps in improving the performance by selecting a VM that has the minimum effect on the network performance. We noticed that MMT selects a VM with minimum migration time; therefore, it performs better than RC.

In the future, we intend to improve the performance of the proposed techniques by considering another resource management metrics, such as, network load, fault tolerance, profit maximization and load balancing. Network load minimization while performing VM placement can further improve the performance at the network level. Moreover, fault tolerant energy efficient resource management can be the new and interesting research challenge that we are considering at the moment.

REFERENCES

[1] B. P. Rimal, E. Choi, and I. Lumb, "A taxonomy and survey of cloud computing systems," in *Proc. 5th Int. Joint Conf. INC, IMS IDC*, Aug. 2009, pp. 44–51.

[2] P. Mell and T. Grance, "The NIST definition of cloud computing (draft)," Nat. Inst. Standards Technol., Gaithersburg, MD, USA, Tech. Rep. 800-145, p. 7, 2011.

[3] V. Medina and J. García, "A survey of migration mechanisms of virtual machines," *ACM Comput. Surv.*, vol. 46, no. 3, 2014, Art. no. 30.

[4] S. Mustafa, B. Nazir, A. Hayat, A. R. Khan, and S. A. Madani, "Resource management in cloud computing: Taxonomy, prospects, and challenges," *Comput. Electr. Eng.*, vol. 47, pp. 186–203, Oct. 2015.

[5] P. Rygielski and S. Kounev, "Network virtualization for QoS-aware resource management in cloud data centers: A survey," *PIK—Praxis Inf. Verarbeitung Kommunikation*, vol. 36, no. 1, pp. 55–64, 2013.

[6] J. Shuja *et al.*, "Survey of techniques and architectures for designing energy-efficient data centers," *IEEE Syst. J.*, vol. 10, no. 2, pp. 507–519, Jun. 2016.

[7] L. Wang and S. U. Khan, "Review of performance metrics for green data centers: A taxonomy study," *J. Supercomput.*, vol. 63, no. 3, pp. 639–656, 2013.

[8] J. Shuja, S. A. Madani, K. Bilal, K. Hayat, S. U. Khan, and S. Sarwar, "Energy-efficient data centers," *Computing*, vol. 94, no. 12, pp. 973–994, 2012.

[9] J. Koomey, *Growth in Data Center Electricity Use 2005 to 2010*. Oakland, CA, USA: Analytics Press, Jul. 2011. [Online]. Available: http://www.analyticspress.com/datacenters.html

[10] K. Bilal, S. U. Khan, and A. Y. Zomaya, "Green data center networks: Challenges and opportunities," in *Proc. 11th Int. Conf. Frontiers Inf. Technol. (FIT)*, Dec. 2013, pp. 229–234.

[11] K. Bilal, S. U. R. Malik, S. U. Khan, and A. Y. Zomaya, "Trends and challenges in cloud datacenters," *IEEE Cloud Comput.*, vol. 1, no. 1, pp. 10–20, May 2014.

[12] R. Bianchini and R. Rajamony, "Power and energy management for server systems," *Computer*, vol. 37, no. 11, pp. 68–76, Nov. 2004.

[13] A. Khosravi, S. K. Garg, and R. Buyya, "Energy and carbon-efficient placement of virtual machines in distributed cloud data centers," in *Euro-Par 2013 Parallel Processing* (Lecture Notes in Computer Science), vol. 8097. Berlin, Germany: Springer, 2013, pp. 317–328.

[14] G. Koutitas and P. Demestichas, "Challenges for energy efficiency in local and regional data centers," *J. Green Eng.*, vol. 1, no. 1, pp. 1–32, 2010.

[15] M. Web, "SMART 2020: Enabling the low carbon economy in the information age," Climate Group, London, U.K., Tech. Rep., 2008. [Online]. Available: https://www.theclimategroup.org/sites/default/files/archive/files/Smart2020Report.pdf

[16] R. Brown, "Report to congress on server and data center energy efficiency: Public law 109-431," Lawrence Berkeley Nat. Lab., Berkeley, CA, USA, Tech. Rep. LBNL-363E.

[17] A. Hameed *et al.*, "A survey and taxonomy on energy efficient resource allocation techniques for cloud computing systems," *Computing*, vol. 98, no. 7, pp. 751–774, Jul. 2016.

[18] 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.

[19] R. W. Ahmad, A. Gani, S. H. A. Hamid, M. Shiraz, A. Yousafzai, and F. Xia, "A survey on virtual machine migration and server consolidation frameworks for cloud data centers," *J. Netw. Comput. Appl.*, vol. 52, pp. 11–25, Jun. 2015.

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

[21] R. Rajavel and M. T, "SLAOCMS: A layered architecture of SLA oriented cloud management system for achieving agreement during resource failure," in *Proc. 2nd Int. Conf. Soft Comput. Problem Solving (SocProS)*, 2012, pp. 801–809.

[22] S. Mustafa, K. Bilal, S. A. Madani, N. Tziritas, S. U. Khan, and L. T. Yang, "Performance evaluation of energy-aware best fit decreasing algorithms for cloud environments," in *Proc. IEEE Int. Conf. Data Sci. Data Intensive Syst.*, Dec. 2015, pp. 464–469.
[23] A. Beloglazov and R. Buyya, "Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in cloud data centers," *Concurrency Comput., Pract. Exper.*, vol. 24, no. 13, pp. 1397–1420, Sep. 2012.
[24] A. Alnowiser, E. Aldhahri, A. Alahmadi, and M. M. Zhu, "Enhanced weighted round robin (EWRR) with DVFS technology in cloud energy-aware," in *Proc. Int. Conf. Comput. Sci. Comput. Intell.*, Mar. 2014, pp. 320–326.
[25] R. N. Calheiros and R. Buyya, "Energy-efficient scheduling of urgent bag-of-tasks applications in clouds through DVFS," in *Proc. IEEE 6th Int. Conf. Cloud Comput. Technol. Sci.*, Dec. 2014, pp. 342–349.
[26] Y. Ren *et al.*, "Balancing performance, resource efficiency and energy efficiency for virtual machine deployment in DVFS-enabled clouds: An evolutionary game theoretic approach," in *Proc. GECCO*, 2014, pp. 1205–1212.
[27] G. B. Mertzios, M. Shalom, A. Voloshin, P. W. H. Wong, and S. Zaks, "Optimizing busy time on parallel machines," in *Proc. IEEE 26th Int. Parallel Distrib. Process. Symp. (IPDPS)*, May 2012, pp. 238–248.
[28] E. Feller, L. Rilling, and C. Morin, "Energy-aware ant colony based workload placement in clouds," in *Proc. IEEE/ACM Int. Conf. Grid Comput. (GRID)*, Sep. 2011, pp. 26–33.
[29] Y. C. Lee and A. Y. Zomaya, "Energy efficient utilization of resources in cloud computing systems," *J. Supercomput.*, vol. 60, no. 2, pp. 268–280, 2012.
[30] B. Addis, D. Ardagna, B. Panicucci, M. S. Squillante, and L. Zhang, "A hierarchical approach for the resource management of very large cloud platforms," *IEEE Trans. Depend. Sec. Comput.*, vol. 10, no. 5, pp. 253–272, Sep./Oct. 2013.
[31] T. Nowicki, M. S. Squillante, and C. W. Wu, "Fundamentals of dynamic decentralized optimization in autonomic computing systems," in *Self-Star Properties in Complex Information Systems* (Lecture Notes in Computer Science), vol. 3460. Berlin, Germany: Springer, 2005, pp. 204–218.
[32] D. Ardagna, B. Panicucci, M. Trubian, and L. Zhang, "Energy-aware autonomic resource allocation in multitier virtualized environments," *IEEE Trans. Serv. Comput.*, vol. 5, no. 1, pp. 2–19, Jan./Mar. 2012.
[33] S. Ali, S.-Y. Jing, and S. Kun, "Profit-aware DVFS enabled RM of IaaS cloud," *Int. J. Comput. Sci. Issues*, vol. 10, no. 2, pp. 237–247, 2013. [Online]. Available: http://www.ijcsi.org/contents.php?volume=10&&issue=2
[34] H. Li, G. Zhu, C. Cui, H. Tang, Y. Dou, and C. He, "Energy-efficient migration and consolidation algorithm of virtual machines in data centers for cloud computing," *Computing*, vol. 8, no. 3, pp. 303–317, 2016.
[35] N. J. Kansal and I. Chana, "Energy-aware virtual machine migration for cloud computing—A firefly optimization approach," *J. Grid Comput.*, vol. 14, no. 2, pp. 327–345, 2016.
[36] M. Ficco, C. Esposito, F. Palmieri, and A. Castiglione, "A coral-reefs and Game Theory-based approach for optimizing elastic cloud resource allocation," *Future Gener. Comput. Syst.*, vol. 78, pp. 343–352, Jan. 2018. [Online]. Available: http://dx.doi.org/10.1016/j.future.2016.05.025
[37] Z. Zhou, Z. Hu, and K. Li, "Virtual machine placement algorithm for both energy-awareness and sla violation reduction in cloud data centers," *Sci. Program.*, vol. 2016, Mar. 2016, Art. no. 15. [Online]. Available: https://www.hindawi.com/journals/sp/2016/5612039/
[38] A. Radhakrishnan and V. Kavitha, "Energy conservation in cloud data centers by minimizing virtual machines migration through artificial neural network," *Computing*, vol. 98, no. 11, pp. 1185–1202, 2016.
[39] P. H. P. Castro, V. L. Barreto, S. L. Corrêa, L. Z. Granville, and K. V. Cardoso, "A joint CPU-RAM energy efficient and SLA-compliant approach for cloud data centers," *Comput. Netw.*, vol. 94, pp. 1–13, Jan. 2016.
[40] N. Tziritas, C.-Z. Xu, T. Loukopoulos, S. U. Khan, and Z. Yu, "Application-aware workload consolidation to minimize both energy consumption and network load in cloud environments," in *Proc. 42nd IEEE Int. Conf. Parallel Process. (ICPP)*, Oct. 2013, pp. 449–457.
[41] N. Quang-Hung, N. Thoai, and N. Son, "EPOBF: Energy efficient allocation of virtual machines in high performance computing cloud," in *Transactions on Large-Scale Data- and Knowledge-Centered Systems XVI* (Lecture Notes in Computer Science), vol. 8960. 2014, pp. 71–86.
[42] S. U. R. Malik, S. U. Khan, and S. K. Srinivasan, "Modeling and analysis of state-of-the-art vm-based cloud management platforms," *IEEE Trans. Cloud Comput.*, vol. 1, no. 1, p. 1, Jan./Jun. 2013, doi: 10.1109/TCC.2013.3.
[43] A. Biere, A. Cimatti, E. M. Clarke, O. Strichman, and Y. Zhu, "Bounded model checking," *Adv. Comput.*, vol. 58, pp. 117–148, 2003.
[44] L. de Moura and N. Bjørner, "Satisfiability modulo theories: Introduction and applications," *Commun. ACM*, vol. 54, no. 9, pp. 69–77, Sep. 2011.
[45] C. Barrett, A. Stump, and C. Tinelli, "The SMT-LIB standard: Version 2.0," in *Proc. 8th Int. Workshop Satisfiability Modulo Theories*, 2010, pp. 1–14.
[46] S. U. R. Malik, K. Bilal, S. U. Khan, B. Veeravalli, K. Li, and A. Y. Zomaya, "Modeling and analysis of the thermal properties exhibited by cyberphysical data centers," *IEEE Syst. J.*, vol. 11, no. 1, pp. 163–172, Mar. 2017, doi: 10.1109/JSYST.2015.2493565.
[47] 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.
[48] *Amazon Elastic Computing Cloud (EC2)*. Accessed: Sep. 15, 2017. [Online]. Available: http://aws.amazon.com/ec2/instance-types
[49] 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.

SAAD MUSTAFA received the B.S. and M.S. degrees in computer science from the COMSATS Institute of Information Technology, Abbottabad, Pakistan, in 2007 and 2010, respectively, where he is currently pursuing the Ph.D. degree with the Department of Computer Science. His research interests include resource management, energy efficient systems, cloud computing, and wireless networks.

KASHIF BILAL received the Ph.D. degree in electrical and computer engineering from North Dakota State University, Fargo, ND, USA, in 2014. He has been with the COMSATS Institute of Information Technology, Abbottabad, Pakistan, since 2004. His research domain encompasses topics related to data center networks, distributed computing, wireless networks, and expert systems.

SAIF UR REHMAN MALIK received the Ph.D. degree from the Department of Electrical and Computer Engineering, North Dakota State University, Fargo, ND, USA, in 2014. He is currently an Assistant Professor with the Department of Computer Science, COMSATS Institute of Information Technology, Islamabad, Pakistan. His research interest revolves around the application of formal methods in large-scale computing systems, software engineering, and security protocols.

SAJJAD A. MADANI received the M.S. degree in computer sciences from the Lahore University of Management Sciences and the Ph.D. degree from the Vienna University of Technology. He is currently with the COMSATS Institute of Information Technology as an in-charge Virtual Campus and an Associate Professor. His areas of interests include low power wireless sensor networks and green computing. He has published over 80 papers in peer-reviewed international conferences and journals.