IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received March 15, 2020, accepted March 30, 2020, date of publication April 2, 2020, date of current version April 15, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2985234

CPU and RAM Energy-Based SLA-Aware Workload Consolidation Techniques for Clouds

BEENISH GUL[1], IMRAN ALI KHAN[1], SAAD MUSTAFA[1], OSMAN KHALID[1], SYED SAJID HUSSAIN[1], DARREN DANCEY[2], AND RAHEEL NAWAZ[3]

[1]Department of Computer Science, COMSATS University Islamabad, Abbottabad Campus, Abbottabad 22010, Pakistan
[2]Department of Computing and Mathematics, Manchester Metropolitan University, Manchester M15 6BH, U.K.
[3]Department of Operations, Technology Events, and Hospitality Management Manchester, Metropolitan University, Manchester M15 6BH, U.K.

Corresponding author: Osman Khalid (osman@cuiatd.edu.pk)

ABSTRACT Cloud computing offers hardware and software resources delivered as services. It provides solutions for dynamic as well as "pay as you go" provision of resources. Energy consumption of these resources is high which leads to higher operational costs and carbon emissions in data centers. A number of research studies have been conducted on energy efficiency of data centers, but most of them concentrate on single factor energy consumption, i.e., energy consumed by CPU only, and energy consumption by Random Access Memory (RAM) is neglected. However, recently the focus has been turned towards impact of energy consumption by RAM on data centers. Studies have shown that RAM consumes about 25% of joint energy consumed by a server's CPU and RAM. In this paper, two energy-aware virtual machine (VM) consolidation schemes are proposed that take into account a server's capacity in terms of CPU and RAM to reduce the overall energy consumption. The proposed schemes are compared with existing schemes using CloudSim simulator. The results show that the proposed schemes reduce the energy cost with improved Service Level Agreement (SLA).

INDEX TERMS Cloud computing, energy efficiency, multi-factor energy consumption, resource allocation, virtualization, workload consolidation.

I. INTRODUCTION

Cloud computing is a shared computing paradigm that aims to provide number of services including computing, web hosting, and storage under a single platform which are otherwise offered by different service providers [1]–[3]. Most of the businesses have shifted to cloud based solutions to make use of "pay as you go" service model, where a subscriber will only pay a cost of resources used [2]. Such elasticity offered by a cloud service model deliver services with an advantage of cost saving by eliminating the requirement of creating and maintaining customer's private infrastructures [4].

Cloud computing uses virtualized hardware, which assists a physical machine (PM) to operate with multiple virtual machines (VMs) having different resources' types and distributions. A cloud hosts several applications on VMs and each VM on a PM has different workloads, which vary with time. Dynamic workloads may result in overloading, i.e., resource demand may increase on the PM beyond the resources allocated to customers. Alternatively, some PMs may remain under-loaded and idle. This kind of PM load imbalance adversely effects the performance of VMs, which ultimately results in more energy consumption and violation of service level agreement (SLA) between customers and service provider.

To minimize the energy consumption and uphold the SLA with the client, workload from overburdened servers is shifted to underutilized servers. The workload consolidation can be performed using various energy efficient resource management (RM) techniques [5]–[10]. Energy efficiency in cloud data centers (DCs) is one of the key challenges to minimize the expense incurred by high energy consumption and it is also important to reduce the CO_2 emissions [11]–[14].

It is reported that 2% of the total CO_2 emissions has been caused by information communication technology (ICT) industry which is likely to increase up to 12% by the year

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

2020 [15]. The large ICT companies, such as Yahoo, Google, Microsoft, etc. host thousands of servers that consume higher energy. It is reported that in 2010 alone, 271.8 billion kWh electricity was consumed by ICT companies, whereas the DCs in USA consumed up to 70 billion kWh in 2014, which is projected to raise up to 73 billion kWh of electricity by the year 2020 [16].

Energy consumption of a typical PM in a DC is about 80% of the overall energy, while other storage and networking devices consume rest of the 20% of total energy [17]. Various RM techniques have been proposed by the researchers to reduce energy consumption by ensuring efficient utilization of resources [18]. Generally, to reduce the energy consumption of a DC, researchers used the strategies that focused particularly on the energy consumption of single resource, i.e., CPU [14], [19]. However, rapid growth in multi-core architectures and the virtualization itself is now more prone to greater energy consumption due to having larger sizes of RAM [19]. Moreover, of total server's energy, RAM consumes up to 25% as reported in [19]. Subsequently, the research community also started considering RAM-based energy consumption for efficient energy management in DCs.

In this paper, we present two energy-aware techniques for VM consolidation. The proposed schemes consider energy consumption by RAM along with CPU. Moreover, the schemes utilize a threshold mechanism in order to keep some resources free to tackle the increased resource demands at run time. We subdivide the resource allocation problem into two components: (a) host selection and (b) VM placement. The proposed techniques take into account a PM's capacity and energy consumption while placing VMs on a server. Moreover, consideration of a server's capacity while placing VMs on the host improves the resource management which is indicated as improvement in our energy graphs.

To summarize, following are our main contributions:

1) This paper presents detailed analysis of the selected energy-efficient resource management techniques using cloud environments.
2) Two new energy-efficient SLA-aware resource management techniques namely MaxCap and RemCap are proposed to optimize energy and handle SLA violations by balancing the network load.
3) The proposed algorithms aim to improve performance in terms of energy efficiency, SLA violations, and address performance degradation due to migrations.
4) We also present the time and space complexity analysis of the proposed techniques.

The rest of the paper is organized as follows. Section II explains the related work. Representation of system model is explained and elaborated in Section III. We discuss our proposed techniques in section IV. In Section V, results and evaluations are discussed, and finally, Section VI concludes the paper followed by future direction.

II. LITERATURE REVIEW

Over the last decade, cloud computing has encompassed a large number of applications, e.g., [20]–[25]. Numerous challenges have been faced by the researchers while performing resource allocation in cloud DCs and one of the most critical issues is energy [26]. To handle such issues, a number of solutions are proposed in the literature and these solutions are either dynamic voltage frequency scaling (DVFS) based or they are workload consolidation based. Wu *et al.* present a DVFS based method which improves the overall utilization of resources resulting in improved energy efficiency [27]. Alnowiser *et al.* provide the solution using concept of weighted round robin algorithm [28]. This algorithm monitors, consolidate, and migrate the overloaded and underloaded VMs that are hosted on PM. The weighted round robin utilizes consolidation method for energy efficient resource scheduling to minimize energy consumption by matching voltage and frequency of the processor. In [29], the authors offer a solution for CPU intensive applications. These applications are packed as bag of tasks. A scheduler along with the proposed solution is used for reducing overall power consumption and completes the task within the deadline. The authors in [30] present the multi-objective algorithm based on game theory that aims to mitigate the overall power consumption. The game theory factor is responsible for efficient resource management while decreasing the energy consumption on server level.

Mertzios *et al.* provide a workload consolidation method to minimize the energy consumption at server level [31]. The VMs having overlapped time of processing are consolidated onto server(s) to minimize the power consumption. The authors in [32] present an energy-efficient resource allocation method based on ant colony optimization (ACO) that intends to control the power consumption besides using very basic operations of resource management, like VM placement, workload consolidation, and VMs migration. Authors in [33] aim to optimize processor utilization in the proposed workload consolidation schemes to improve energy efficiency. Using cost functions, the schemes perform server selection based on utilization and difference in server's power consumption under varying loads.

Addis *et al.* in [34] offer solution to energy efficient resource management based on idea of hierarchical framework as discussed in [35]. The algorithm divides the managers into two types for managing and maintaining the server's resources. The servers are classified based on services' classes, and best available server is selected for task assignment depending upon task's class. A server's application manager is used to migrate the VMs, allocate the capacity, perform frequency scaling, and perform load balancing across the servers. In [36], the authors provide a multi-tier virtualized cloud environment to manage the resources. A workload prediction method is used to detect the fluctuations in workloads. These predictions are used to assign the workloads to VMs.

Authors in [37] propose SLA-aware energy efficient resource management solution based on DVFS that intends to provide energy efficiency with guaranteed SLA. The DVFS modules are integrated in all the PMs which utilize hybrid optimization to address load balancing, resource allocation, and VM placement across the servers. An energy-efficient multi-resource model is presented by Li *et al.* in [38]. The model works on the principal of modified particle swarm optimization (PSO) for consolidating the workload. To handle SLA violations and to control the migration of VMs, a threshold mechanism is used. Kansal and Chana [39] in their work present a multi-objective firefly optimization-based model that aims to decrease the number of VM migrations and thus reducing overall energy consumption. The proposed technique migrates the VM with highest resource requirements to the server with least resource utilization.

Bi *et al.* [40] proposed an SLA based study for optimizing the profit of virtualized cloud data centers (VCDC). They also proposed a dynamic meta-heuristic hybrid algorithm to maximize the profit and reduce energy costs. The algorithm is developed with the concept of annealing and particle swarm optimization (PSO). Results show an increase in profit and lowering of energy costs. Khoshkholghi *et al.* [41] aimed to improve the utilization of resources, such as CPU, RAM, and bandwidth through dynamic VM consolidation in cloud data centers. The authors in their study proposed algorithms to reduce the energy consumption and improve the performance of computing resources in overall data centers based on SLA. Results show an improvement in energy costs as well as performance of data centers. In [42], the authors proposed a dynamic provision of VMs in virtualized application services. They proposed a hybrid queuing model for determining the number of VMs and to allocate each tier of application services. Results show improvement in performance and reduced energy costs.

Castro *et al.* [19] in their study aim to reduce the power consumption of servers in clouds. Their model calculates overall energy by adding up the energy consumption of both CPU and RAM. The power difference of a server before allocation and after allocation is considered while placing VMs on the server which has never been picked up for VM placement. To control SLA violations, the authors used various threshold mechanisms. Recently, a multi-factor energy-efficient resource allocation model has been proposed by authors in [14]. The proposed model takes into account joint power consumption of CPU and RAM while taking VM hosting decisions. In the aforementioned work, capacity of server is considered along with the power consumption while selecting server for hosting of VMs.

All above cited techniques are designed to deliver energy efficient resource allocation which ultimately minimizes the overall cost of cloud DCs. However, these techniques still have some performance limitations when multiple factors are considered simultaneously, e.g., SLA and energy consumption. Moreover, majority of work is not considering the energy consumed by RAM in addition to CPU. Therefore, in this work we propose energy-efficient techniques that optimally utilize CPU and RAM based on energy model to reduce energy consumption. We also propose SLA-aware versions of our energy-aware techniques to handle the SLA violations that may occur due to workload consolidation.

III. SYSTEM MODEL

Proposed system model in cloud environment is presented in Fig 1. We consider a cloud network of DCs consisting of a large number of heterogeneous servers [19], [39]. Every server has its processing speed, memory, storage-capacity, and energy consumption. Each individual server is represented by CPU capacity which is calculated in Million Instructions Per Second (MIPS), a Random Access Memory (RAM) and a bandwidth. Servers contain local disks to host Operating Systems (OS) whereas Network Attached Storage (NAS) is used to store VMs and to enable live migration of VMs. Large number of cloud users can submit request for M number of VMs where each VM consists of its own load of CPU, memory utilization, and network transfer rate [43]. For the management of resources, proposed system consists of two layers: (a) a global manager and (b) a local manager [19] as shown in Fig 1. Carbon emission directories are maintained for keeping energy efficiency information. Global manager on central node also known as green broker is subdivided into task scheduler, task selector, cost calculator, application's profile, and carbon emission calculators. Each server has local manager, which keeps track of that server. That information is sent to green broker/global manager at central node which places the workload on different servers based on collected information. User sends service request to green broker for allocation of service. Green broker keeps track of server's utilization, carbon emission directories, and application profiles. When a request arrives, green broker checks the application profile and the requirements of user. Based on information received from servers, carbon emission directories, private cloud, and available services, the green broker assigns the required service to the user. For the whole process, SLA is agreed with the service provider where service provider is responsible of providing services with agreed terms and conditions, and in case of SLA violations, the service provider is penalized. Power model and architecture of proposed techniques are presented in the subsequent subsection. Table 1 contains the notations and their meanings used in our model.

A. CPU ARCHITECTURES

In our proposed model, each server is considered to have n cores with processing power of m MIPS. A server's capacity c is calculated as $m \times n$ MIPS. Moreover, separate cores to host parts of a VM are not used in our model. Therefore, processing power of a single VM at most equals the processing power of a single core.

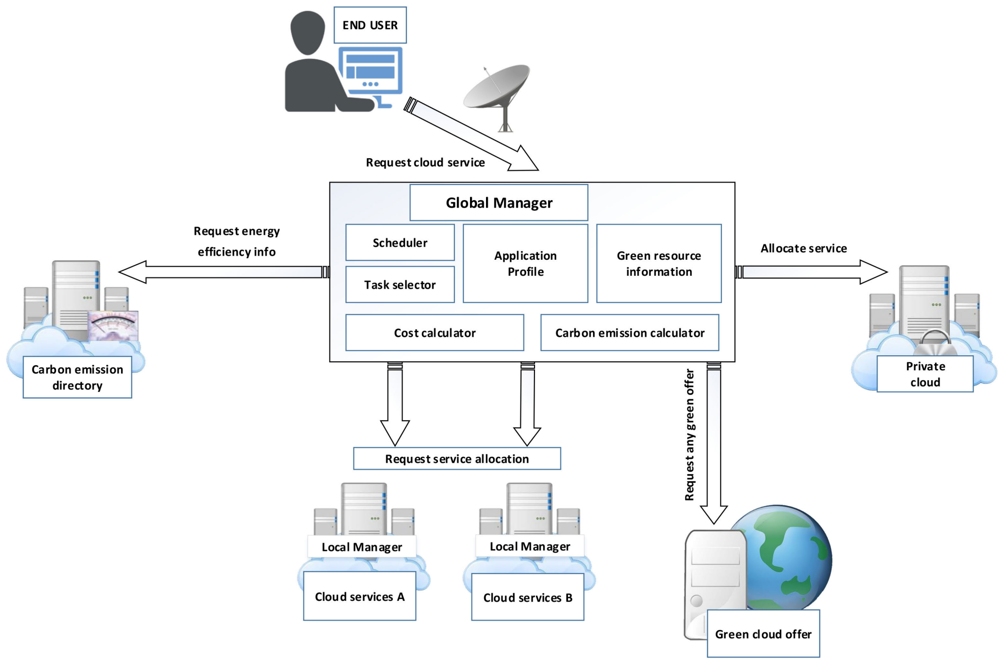


FIGURE 1. Green architecture for energy efficient cloud environments.

TABLE 1. Notations and their meanings.

Notation	Meaning
E_{CPU}	Energy consumption by CPU
E_{RAM}	Energy consumption by RAM
E_{bgp}	Background power
E_{op}	Operational power
E_{asb}	Active standby
E_{apd}	Active power down
CPU_{utl}	Utilization of CPU
CPU_{idl}	Idleness of CPU
E_{br}	Power required for attempting read command
E_{bw}	Power required for attempting write command
RAM_B	Bandwidth of memory
CPU_{max}	Maximum CPU capacity
P_{max}	Peak power consumption of server
$CPU_{Available}$	Available capacity of CPU calculated in MIPS
PM	Physical machine
P_{rem}	Remaining power a PM can consume
M_j	Memory size of VM j
c	Network link capacity
c_{dj}	Performance degradation estimation
c_{rj}	CPU capacity required by VM
PDM	Performance degradation due to migration
M	Number of VMs
$SLATAH$	SLA violations time per active host
$SLAV$	SLA violations

B. POWER ESTIMATION MODEL

The power model computes a server's power consumption which is based on joint power consumed by CPU and RAM. The power consumed by other components, such as network interface and disks is ignored. The total power can be calculated as:

$$E_{ALL} = E_{CPU} + E_{RAM}. \tag{1}$$

The formula below expresses the total energy (E_{CPU}) consumed by a server. So, the overall energy consumption by the CPU (E_{CPU}) can be expressed as:

$$E_{CPU} = k \times P_{max} + (1 - k) \times P_{max} \times CPU_{utl}. \tag{2}$$

Here, P_{max} denotes the maximum power, and k defines the fraction of time the server remains idle. This fraction is normally noted as 70% of the total time [19]. The time server stays active is expressed by $1 - k$, whereas CPU_{utl} represents the CPU utilization.

A real-data from SPEC power benchmark is employed to estimate the CPU consumption [44]. Two servers are used namely: (a) HP-Proliant ML110 G5 and (b) HP-Proliant ML110 G4. In the aforementioned benchmark, CPU is the key energy consumer. Therefore, hard disk and display are turned off after one minute, while network communications are minimized and memory pages of applications are kept locked in physical RAM. Two key components constitute the power consumption of RAM, namely: (a) background power (E_{bgp}) and (b) operational power (E_{op}). It is calculated as:

$$E_{RAM} = E_{bgp} + E_{op}. \tag{3}$$

E_{bgp} is used for changing the memory states [45]. Inspired by [19], proposed model employs only two states known as Active Standby (E_{asb}) and Active Power-down (E_{apd}). Aforesaid states present a tradeoff between latency time and energy consumption. E_{asb} has a highest energy consumption and least latency. Alternatively, energy consumption of E_{apd} is 39% lower than E_{asb} with the higher cost of latency. We also assume that the RAM stays in E_{asb} when CPU is processing, and the state of the RAM changes to E_{apd} when CPU is not in use. Considering these factors, E_{bgp} can be calculated as:

$$E_{bgp} = CPU_{utl} \times E_{asb} + CPU_{idl} \times E_{apd}. \tag{4}$$

Here in (4), CPU_{utl} is the utilization of CPU (varies from 0 to 1) in a given time span and CPU_{idl} is the idleness of CPU in the same time. Likewise, E_{op} depends on two factors i.e., memory read/write attempts and frequency of memory operations. E_{op} is basically the product of power, memory, and the bandwidth required for commands of read/write that may occur in processing. It can be determined as:

$$E_{op} = RAM_B \times \frac{E_{br} + E_{bw}}{2} \times CPU_{utl} \times U(0, 1). \tag{5}$$

where, RAM_B is bandwidth of the memory, E_{br} and E_{bw} are the power requirements for attempting read command and write command, respectively, and $U(0, 1)$ is a uniformly distributed random value.

IV. JOINT CPU AND RAM BASED VM CONSOLIDATION

Energy-efficient VM consolidation can be achieved by: (a) overloaded server's detection for migration of some VMs to other servers, (b) underloaded server's detection for migration of some VMs to shut down other servers, (c) VM selection to migrate from overloaded servers to under-loaded servers, and (d) VM placement to migrate from overloaded and underloaded servers. In our study, we considered the aforementioned cases (c) and (d). Therefore, we propose two energy-aware and SLA-aware techniques to improve the energy consumption and to handle resultant SLA violations, respectively.

Our proposed *MaxCap* and *RemCap* server selection techniques use the lower threshold (LT) mechanism to identify the underutilized servers. LT mechanism keeps check on the utilization of a server provided by a local manager that is placed on each server to keep track of CPU and RAM utilization. If the utilization of server is below the set threshold value, then the server is considered as an under-loaded server and it is switched off after migrating all the hosted VMs to other servers. 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. So, when workload on the server exceeds, then the server is considered as overloaded server.

A. MAXIMUM CAPACITY AND POWER TECHNIQUE (MaxCap)

We propose an energy-aware scheme namely MaxCap that intends to improve the energy consumption of DCs. In the proposed scheme, server selection for VM hosting is based on maximum CPU capacity and maximum power consumption. Whereas, the existing schemes have not considered the capacity factor during server selection for hosting the VMs, which as a result produces higher power consumption. Server with minimum energy cost after VM placement is selected for hosting of upcoming VMs. The following equation performs server selection:

$$MaxCap = \frac{CPU_{max}}{P_{max}}. \tag{6}$$

Here, CPU_{max} is the maximum CPU capacity calculated in MIPS, while P_{max} is the peak power consumption of server in watts. The server selected for hosting the VM will be the one with the maximum *MaxCap*. A threshold mechanism is also used in our work to avoid SLA-violations when availability of computational resources is limited. The pseudocode for *MaxCap* is presented in Algorithm 1.

In Line 1, the VMs V are sorted in descending order of CPU requirements. Servers S are sorted in descending order of utilization (Line 2), so that the VM with larger workload is placed on the first utilized server where it fits. For sorted VMs V', the following steps are repeated for each VM $v \in V'$ in Line 3–Line 23. A server s is selected from the sorted list of servers S' and cumulative value is computed consisting of CPU utilization and computational requirement of VM (Line 7). If the value is less than given value of threshold, only then the server will be selected for hosting the VMs. The values of peak energy consumption, maximum CPU capacity, and available CPU capacity are stored in the respective variables (Line 8–Line 10). Using (5), the aforementioned variables are used to calculate the *MaxCap* ratio (Line 11). The algorithm makes sure that a VM is placed on a server that is already in use, and when no such server is available, the VM is placed on an unused server (Line 12–Line 14). The *MaxCap* is compared with *Max_ratio* and is overwritten by *Max_ratio* if *MaxCap* is greater, and s is set to an *Allocated_host* (Line 15–Line 18). If none of the used servers is selected, then the allocation is performed to the unused server (Line 20–Line 22). The above steps are repeated for remaining servers available in the list S' and a server is selected with maximum value of *MaxCap* for hosting of VMs.

B. REMAINING CAPACITY AND POWER (RemCap)

This scheme attempts to improve the CPU and RAM's energy consumption. The *RemCap* utilizes CPU's available capacity and a server's power consumption. A server with minimum energy cost after VM placement is selected for the next VM hosting. *RemCap* is calculated as:

$$RemCap = \frac{CPU_{Available}}{P_{rem}}. \tag{7}$$

In the above equation, $CPU_{Available}$ is the available capacity of a CPU calculated in MIPS whereas P_{rem} is remaining power in watts a PM can consume. Server offering maximum *RemCap* value is selected for hosting the new VM. Algorithm 2 performs the host selection and VM placement based on two factors i.e., remaining power and server's capacity. The pseudocode for *RemCap* is presented in Algorithm 2.

The VMs and servers' lists are sorted in descending order based on CPU requirements and utilizations, respectively (Line 1–Line 2). For each VM v, a cumulative value for a

Algorithm 1 Pseudocode for *MaxCap* Algorithm

```
Input: $srvList(S)$, $vmList(V)$, $threshold$;
Output: Allocation of VMs
1: $V' \leftarrow getSortedVMs(V)$
2: $S' \leftarrow getSortedServers(S)$
3: for each $v \in V'$ do
4:     $Max\_ratio \leftarrow 0$
5:     $Allocated\_host \leftarrow Null$
6:     $comp\_req_v \leftarrow getCompReq(v)$
7:     for each $s \in S'$ such that $utilization(s) + comp\_req_v < threshold$ do
8:         $P_{max} \leftarrow getPeakEnergyConsumption(s)$
9:         $CPU_{max} \leftarrow getMaxCPUCapacity(s)$
10:        $CPU_{Available} \leftarrow getAvailCPUCapacity(s)$
11:        $MaxCap \leftarrow CPU_{max}/P_{max}$
12:        if $CPU_{Available} = CPU_{max}$ and $Allocated\_host \neq Null$ then
13:            break
14:        end if
15:        if $MaxCap > Max\_ratio$ then
16:            $Allocated\_host \leftarrow s$
17:            $Max\_ratio \leftarrow MaxCap$
18:        end if
19:    end for
20:    if $Allocated\_host \neq Null$ then
21:        $add\_allocation(v, Allocated\_host)$
22:    end if
23: end for
24: $Allocation \leftarrow get\_allocation()$
25: return $Allocation$
```

server s is computed based on CPU utilization and computational requirement of v (Line 7). If the value is less than given value of threshold, then server will be selected for hosting the VM. The details of energy, maximum CPU capacity, and available CPU capacity are extracted, and used in calculation of *RemCap* ratio (Line 8–Line 12). The algorithm makes sure that VMs are placed on servers that are already in use (Line 16–Line 19) and new server is utilized if and only if no such server exists in the server's list (Line 21–Line 23). The above steps are repeated for the servers available in list and a server with maximum value of *RemCap* is selected for hosting of VMs.

C. DISCUSSIONS

In our proposed algorithms, the formulas (6) and (7) are used to compute the parameters, namely *MaxCap* and *RemCap*, respectively. These parameters provide the maximum MIPS (capacity) against minimum watts (energy). The MIPS/Watts ratio provided by the aforementioned parameters is used to select servers for efficient VM placement. The difference between both the *MaxCap* and *RemCap* is that the *MaxCap* server selection technique places VMs on a server that provides maximum MIPS/watts, i.e., the server that consumes minimum power and provides maximum capacity to host the VMs. Whereas, the *RemCap* server selection technique places the VMs on a server based on real-time utilization, i.e., remaining capacity and power that a server can consume. However, the existing *CREW* technique selects the server only based on power consumption of the CPU and RAM and ignores the capacity of the server. In *CREW*, just a power difference before and after placement of VMs on a server is taken. The server that consumes less power after placement of VMs is chosen for hosting the VMs. On the contrary, in our proposed techniques, we are placing the VMs on servers that are already in use and an unused server is utilized if and only if no used server can host more VMs. Such a server whose remaining capacity and remaining power provides the maximum MIPS/watt ratio is selected to host the VMs. When we compare our results with the existing CREW technique, there is significant improvement in energy graphs of our proposed techniques.

D. SPACE AND TIME COMPLEXITY

The space complexity of proposed techniques namely: *MaxCap* and *RemCap* is computed as: $O(2p + q)$, where, p denotes the number of VMs that are to be placed on servers, q is used to represent the number of servers.

The time complexity of our proposed *MaxCap* and *RemCap* algorithms is calculated by using following steps. First of all, received VMs are sorted in descending order

Algorithm 2 Pseudocode for *RemCap* Algorithm

```
Input: $srvList(S)$, $vmList(V)$, $threshold$;
Output: Allocation of VMs
1:  $V' \leftarrow getSortedVMs(V)$
2:  $S' \leftarrow getSortedServers(S)$
3:  for each $v \in V'$ do
4:      $Max\_ratio \leftarrow 0$
5:      $Allocated\_host \leftarrow Null$
6:      $comp\_req_v \leftarrow getCompReq(v)$
7:      for each $s \in S'$ such that $utilization(s) + comp\_req_v < threshold$ do
8:          $Energy_{\_before\_alloc} \leftarrow getBeforeAllocEnergy(s)$
9:          $Energy_{\_after\_alloc)} \leftarrow getAfterAllocEnergy(s)$
10:         $CPU_{max} \leftarrow getMaxCPUCapacity(s)$
11:         $CPU_{Available} \leftarrow getAvailCPUCapacity(s)$
12:         $RemCap \leftarrow CPU_{Available}/(Energy_{\_after\_alloc} - Energy_{\_before\_alloc})$
13:         if $CPU_{Available} = CPU_{max}$ and $Allocated\_host \neq Null$ then
14:             break
15:         end if
16:         if $RemCap > Max\_ratio$ then
17:             $Allocated\_host \leftarrow s$
18:             $Max\_ratio \leftarrow MaxCap$
19:         end if
20:     end for
21:     if $Allocated\_host \neq Null$ then
22:         $add\_allocation(v, Allocated\_host)$
23:     end if
24: end for
25: $Allocation \leftarrow get\_allocation()$
26: return $Allocation$
```

according to their requirements of CPU and RAM. In case of p number of VMs, time complexity will be $O(p.log(p))$. After sorting VMs, in the second step, the servers on the basis of their CPU and RAM utilization are sorted in descending order. Therefore, sorting q servers will exhibit time complexity of $O(q.log(q))$. After sorting both the VMs and servers in descending order, the VM placement will take place for which the outer loop will be p times. When we compare the best-case time complexity of the algorithm, the inner loop will have s iterations as each time a used server will be available to host a VM. Alternatively, in worst-case scenario, the inner loop will have q iterations as no used server is available. The best-case and worst-case time complexities of Algorithm 1 and Algorithm 2, respectively, are expressed as:

$$O([p(log(p) + s)] + q \times log(q)).$$
$$O([p(log(p) + q] + q \times log(q)).$$

The time and space complexities of the proposed and selected techniques are presented in Table 2.

V. EXPERIMENTAL EVALUATIONS

We discuss the experimental environment, performance evaluation, and results of our energy-aware and SLA-aware techniques.

TABLE 2. Time and space complexity.

Technique	Time Complexity		Space Complexity
	Best Case	Worst Case	
$CREW$	$O([p(log(p) + q)] + q \times log(q))$	$O([p(log(p) + q)] + q \times log(q))$	$O(2p + q)$
$MaxCap$	$O([p(log(p) + s)] + q \times log(q))$	$O([p(log(p) + q)] + q \times log(q))$	$O(2p + q)$
$RemCap$	$O([p(log(p) + s)] + q \times log(q))$	$O([p(log(p) + q)] + q \times log(q))$	$O(2p + q)$

A. EXPERIMENTAL SETUP

This section evaluates the impact of power consumption by RAM on overall energy for workload consolidation. For experiments, CloudSim simulator is used [46] which allows modeling and simulations of DCs on large scale [46]. Moreover, CloudSim simulates the real-world workloads, VM instances, and servers' configuration. A real dataset from SPECpower benchmark [44] is employed in our simulation in order to estimate the energy consumption of CPU. The dataset consists of open source real-world workloads provided by the PlanetLab. These workloads were collected by PlanetLab over 10 days timespan by making use of approx. 500 servers and each of the workload has more than 1000 tasks. Moreover, we used this standard benchmark dataset so that the other researchers should be able to compare their techniques using the same dataset.

TABLE 3. Servers configuration.

Server	CPU Model	Number of Cores	Clock Rate (MHz)	RAM (GB)
HP ProLiant ML110G5	Intel Xeon 3075	2	2660	32
HP ProLiant ML110G4	Intel Xeon 3075	2	1860	32

TABLE 4. VM sizes.

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 instance	3000	6

TABLE 5. Workloads.

Workloads	Number of VMs	Median	St. Deviation
W1	1052	6%	17.09%
W2	898	6%	16.83%
W3	1061	5%	15.57%
W4	1516	4%	12.78%
W5	1078	5%	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%
W11	1044	6%	15.30%

We used two types of servers namely HP-Proliant ML110 G4 and HP-Proliant ML110 G5. A DC having 800 servers is considered consisting of the aforementioned types of servers as shown in Table 3. Servers in this setup have RAM of 32GB. We have taken VM sizes from Amazon EC2 instances [47] given in Table 4. Planetlab workloads are used as shown in Table 5.

1) PERFORMANCE METRICS

The following performance metrics are used to compare our heuristic with existing heuristics.

a: ENERGY CONSUMPTION

Joint CPU and RAM energy consumption by PMs in DCs is considered as a first metric. Our power model is discussed earlier in Section III.

b: VM MIGRATIONS

Migration of VMs is the method of migrating pages of VMs from across the servers at runtime. Initially, hypervisor copies the pages of memory, then migrates that VM in several cycles. Once the VM migration is completed, the VM starts resuming its paused processes without interruption. We consider a VM j with memory of size M_j and network link capacity c from source to destination. For migrating the $VM(j)$, time needed can be computed as:

$$T_{M_j} = \frac{M_j}{c}. \tag{8}$$

VM migration effects the performance of applications negatively due to downtime faced during migrations. To estimate the performance degradation resulting from migrations (Cd_j) we employ the approach as discussed in [19].

c: SLA VIOLATIONS

Three metrics are used for quantifying the level of service for VMs. The first metric is SLA violations Time per Active Host (SLATAH). If N represent the number of servers, T_{si} is time when full utilization has been experienced by the server i and T_{ai} be the total time that server i stays active, then SLATAH is computed as follows:

$$SLATAH = \frac{1}{M}\sum_{i=1}^{M}\frac{T_{si}}{T_{ai}}. \tag{9}$$

Performance degradation due to migrations (PDM) is the second metric we have considered. For calculating PDM, let M represents VMs' count, C_{dj} be estimated value of performance degradation, and C_{rj} be the required CPU capacity for VM during its lifetime. PDM is computed as:

$$PDM = \frac{1}{M}\sum_{j=1}^{n}\frac{C_{dj}}{T_{rj}}. \tag{10}$$

Third and the last metric is SLA violation (SLAV) which is product of (9) and (10), defined as follows:

$$SLAV = SLATAH \times PDM. \tag{11}$$

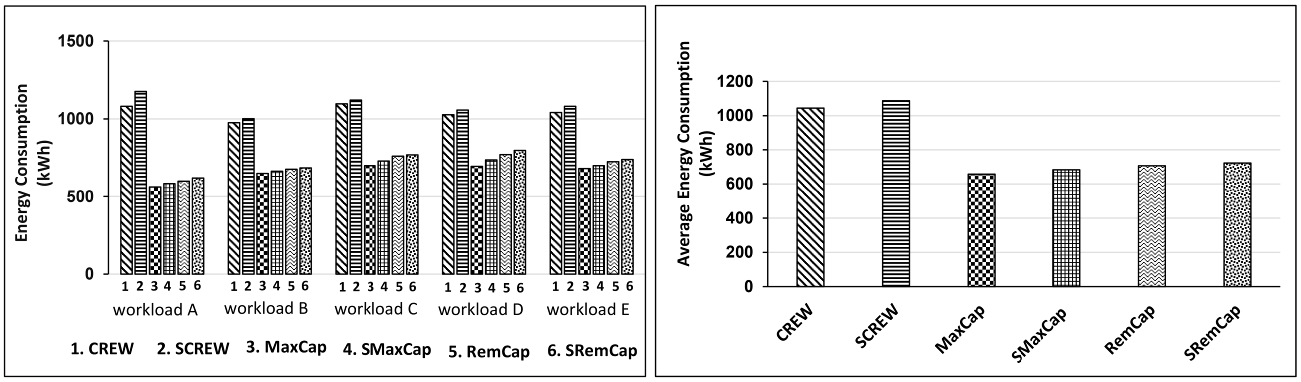
B. PERFORMANCE EVALUATION

We compare our proposed energy-aware and SLA-aware schemes with existing energy-aware and SLA-aware techniques *CREW* and *SCREW* [19]. These techniques take into account power consumption of server and are considered as pioneer in measuring the energy consumption by RAM. Pedro *et al.* asserted that RAM consumes 25% of a server's energy [19]. In earlier studies, there is no power model designed considering joint power consumption by CPU and RAM. The authors' work is also cited by recent studies, such as [14] and [48], due to the novelty of the topic and efficiency of energy results. We have also worked in the same area, and selected *CREW* and *SCREW* as a benchmark for comparisons of our results.

Our proposed energy-aware schemes *MaxCap* and *RemCap* show improvement in energy consumption at the cost of SLA. Whereas our SLA-aware versions *SMaxCap* and *SRemCap* reduce the SLA violations occurred due to workload consolidation, as well as reduce performance degradation due to migrations in *MaxCap* and *RemCap*. We used various PlanetLab workloads for our simulations [49]. Details of migration policies and mechanisms for setting threshold values for existing and proposed techniques are provided in Table 6.

d: DYNAMIC THRESHOLD VALUES

We utilized median absolute deviation (MAD) based dynamic threshold mechanism as it is comparatively more resilient



(a) (b)

FIGURE 2. Energy consumption: (a) by various energy-aware and SLA aware techniques and (b) average energy consumption.

TABLE 6. Existing and proposed techniques.

Technique Name	Migration Technique	Lower Threshold	Upper Threshold
CREW [19]	MMT	Yes	No
MaxCap	MMT	Yes	No
RemCap	MMT	Yes	No
SCREW [19]	MMT	Yes	Yes
SMaxCap	MMT	Yes	Yes
SRemCap	MMT	Yes	Yes

and robust to outliers than standard deviation [50]. MAD is used to control energy consumption and SLA violations. Moreover, MAD generates new values of threshold with the help of previous knowledge gained from server's utilization. These values can be obtained using following mechanism from the given uni-variate set of data $X_1, X_2, \cdots X_n$. The MAD can be computed as:

$$MAD = median_i(|X_i - median_j(X_j)|). \tag{12}$$

In above equation, X_i is the current utilization of CPU, while X_j is the set of previous utilizations of CPU. The dynamic threshold value T_u can be computed by using the following threshold formula:

$$Tu = 1 - s \cdot MAD. \tag{13}$$

Here, $s \in R^+$ is a parameter used to control the threshold behavior. The value of s ranges from 0 to 1. Moreover, energy consumption is directly proportional to s. With low value of s, the energy consumption will also decrease.

Value of s and energy consumption are directly proportional to each other, i.e., if the value of s is low, energy consumption will also be decreased. The value of s is inversely proportional to SLA, such that with the lower value of s (and low energy consumption) the SLA violations will be high.

e: STATIC THRESHOLD

This threshold is utilized to set upper limit of server utilization. Static threshold remains fixed and do not vary with time during simulation. In our study, we have fixed the upper threshold to 80%, and VM migrations will happen beyond this defined limit [39]. However, in case the dynamic workloads are used, static threshold mechanism may suffer. Experiment results are discussed in subsequent subsections.

1) ENERGY CONSUMPTION

Fig. 2 presents the comparison of energy consumption for various PlanetLab workloads. Comparisons of our proposed energy-aware schemes (MaxCap and RemCap) and SLA-aware schemes (SMaxCap and SRemCap) are performed with existing energy-aware and SLA-aware techniques, namely: CREW and SCREW [19], respectively. Results show that proposed energy-aware and SLA-aware schemes outperform the existing energy-aware and SLA-aware schemes. Improvement in results is due to improving energy efficiency which is made possible by decreasing the number of active servers. Moreover, proposed schemes monitor resource utilization to gather updated information while the existing schemes consider only maximum energy consumption for initial host selection. In our study, capacity of CPU is considered during VM placement. Server that provides maximum RAM and CPU capacities per watt power is selected for VM placement. In this way, energy consumption is improved as compared to existing CREW. For instance, MaxCap consumes 37% less energy than CREW and RemCap consumes 32% less energy than CREW. The proposed SLA-aware version SMaxCap consumes 35% less energy than SCREW and SRemCap consumes 31% less energy than the SLA-aware versions SCREW. Fig. 2b displays the consolidated average energy consumption on different workloads having different requirements of resources for each workload.

2) AVERAGE SLA VIOLATIONS

Performance comparisons for SLA violations is presented in Fig. 3. The SLA-aware SMaxCap and SRemCap show better performance in terms of SLA violations when compared with energy-aware MaxCap and RemCap. The better performance is due to the utilization of upper threshold that allows

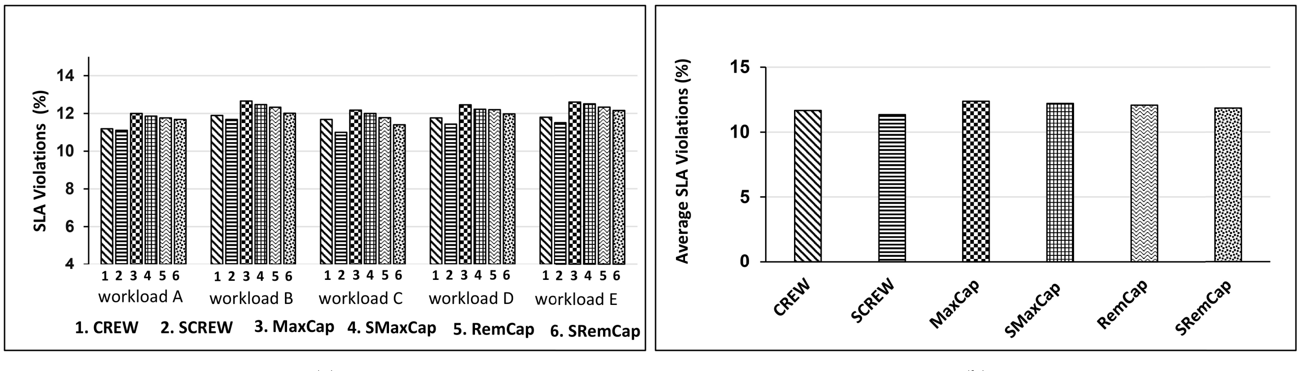


FIGURE 3. SLA violations: (a) by various energy-aware and SLA aware techniques and (b) average SLA violations.

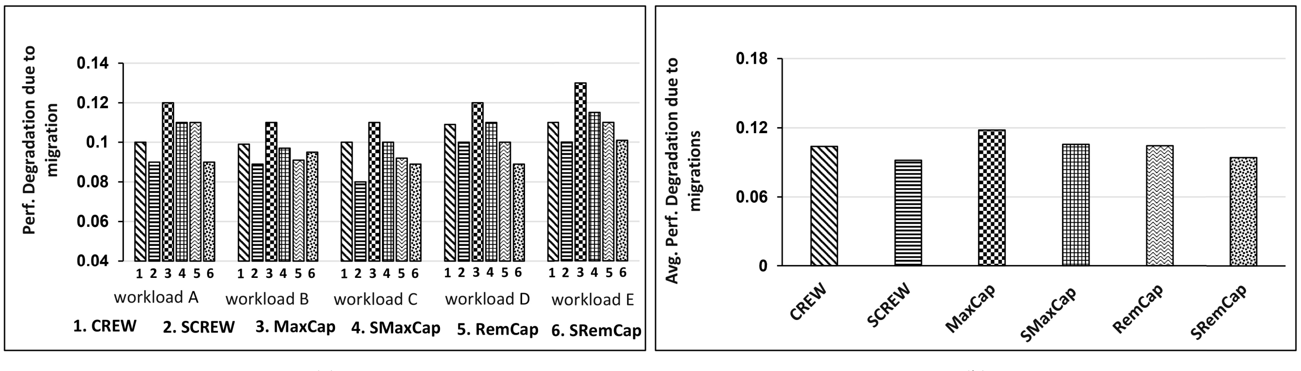


FIGURE 4. Performance degradation: (a) by various energy-aware and SLA aware techniques and (b) average performance degradation.

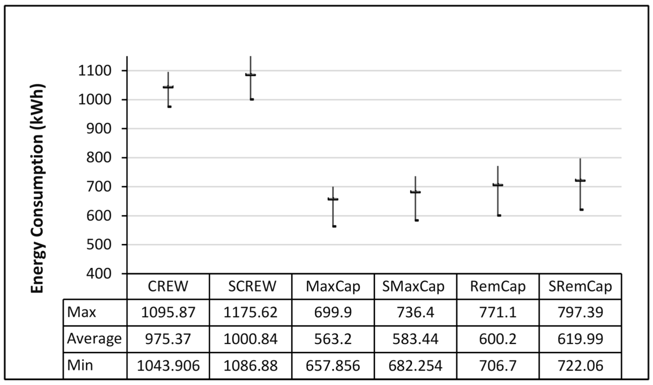
some resources to be free, and when required, the resources are available, thus avoiding the SLA violations. The MaxCap and RemCap show 9% to 7% greater SLA violations than SCREW, respectively. However, the SLA-aware versions decrease the violations by 3%. The better performance of SCREW is due to its non-aggressive behavior in workload consolidation. The techniques with non-aggressiveness do not utilize the resources fully, leaving some resources free on each server. This helps later to accommodate the increased demands of VMs. However, SCREW consumes more energy than our proposed schemes, so there is a tradeoff in energy consumption and SLA violations. Fig. 3b presents the consolidated graphs of average SLA violations by both proposed and existing techniques on different workloads.

3) PERFORMANCE DEGRADATION DUE TO VM MIGRATIONS

Fig. 4 presents the comparison of performance degradation due to migrations for existing and proposed techniques. It can observed that efficient management of resources and the selected VM migration policy i.e., Minimum Migration Time (MMT) for placement of VMs helps to reduce the performance degradation. Moreover, using upper threshold mechanism for resources minimizes the SLA violations because such mechanism keeps some resources free that are available to fulfill the increased resource demands at runtime, thus avoiding performance degradation due to VM migration. In Fig. 4, SRemCap exhibits lowest average performance degradation compared to its proposed counterparts MaxCap and RemCap. Moreover, in Fig. 4b, SRemCap shows better performance than SMaxCap. However, our SLA-aware SMaxCap and SRemCap show greater performance degradation than SCREW. The reason for such behavior is that primary aim of the proposed techniques is to reduce energy consumption. Therefore, when we try to reduce energy consumption, this will increase the SLA violations, and performance degradation is proportional to SLA violations. However, as we see in Fig. 2, the energy consumption of existing schemes CREW and SCREW is greater than our proposed schemes.

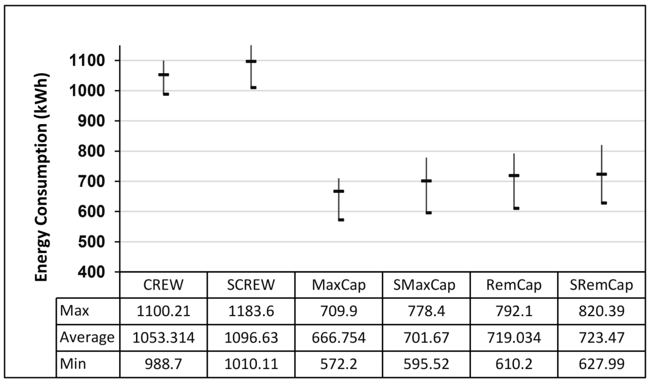
4) EFFECT OF DYNAMIC AND STATIC THRESHOLD

Fig. 5 presents the maximum, average, and minimum energy consumption of proposed and selected energy-aware and SLA-aware techniques using dynamic and static threshold. From Fig. 5a and Fig. 5b, it is evident that our proposed MaxCap and RemCap have improved performance in terms of energy as compared to selected techniques such as CREW. To identify the underloaded server we used lower



	CREW	SCREW	MaxCap	SMaxCap	RemCap	SRemCap
Max	1095.87	1175.62	699.9	736.4	771.1	797.39
Average	975.37	1000.84	563.2	583.44	600.2	619.99
Min	1043.906	1086.88	657.856	682.254	706.7	722.06

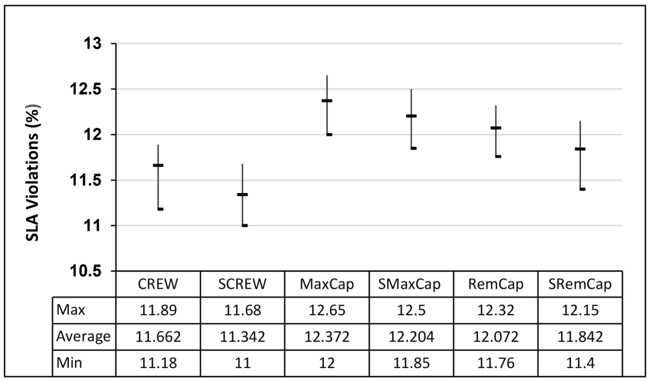
(a)



	CREW	SCREW	MaxCap	SMaxCap	RemCap	SRemCap
Max	1100.21	1183.6	709.9	778.4	792.1	820.39
Average	1053.314	1096.63	666.754	701.67	719.034	723.47
Min	988.7	1010.11	572.2	595.52	610.2	627.99

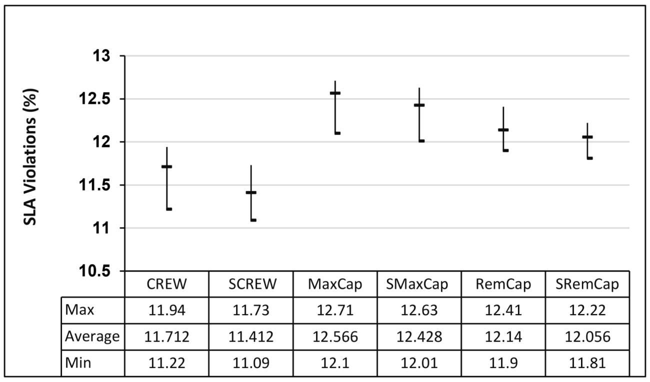
(b)

FIGURE 5. **Energy consumption (a) Max, average, and min of all workloads with dynamic thresholds (b) Max, average, and min of all workloads with static thresholds.**



	CREW	SCREW	MaxCap	SMaxCap	RemCap	SRemCap
Max	11.89	11.68	12.65	12.5	12.32	12.15
Average	11.662	11.342	12.372	12.204	12.072	11.842
Min	11.18	11	12	11.85	11.76	11.4

(a)



	CREW	SCREW	MaxCap	SMaxCap	RemCap	SRemCap
Max	11.94	11.73	12.71	12.63	12.41	12.22
Average	11.712	11.412	12.566	12.428	12.14	12.056
Min	11.22	11.09	12.1	12.01	11.9	11.81

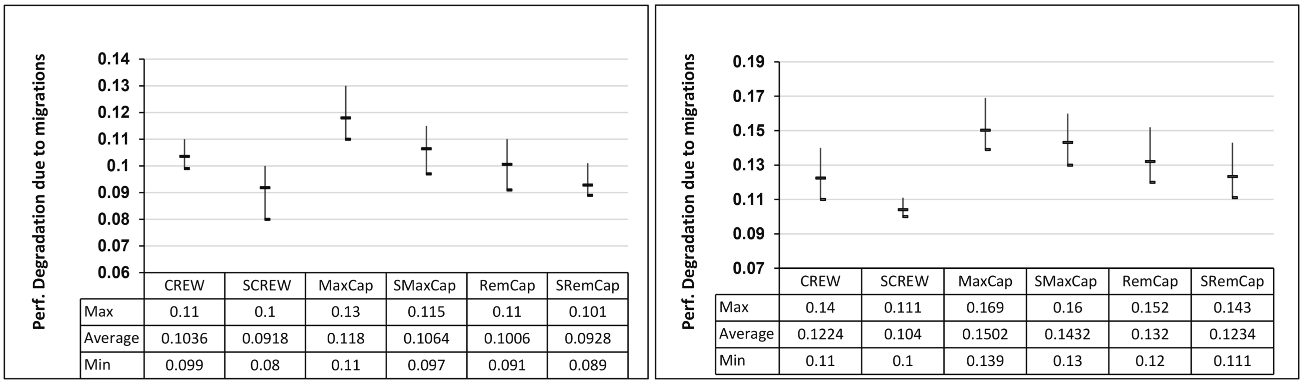
(b)

FIGURE 6. **SLA Violations (a) Max, average, and min of all workloads with dynamic thresholds (b) Max, average, and min of all workloads with static thresholds.**

threshold mechanism. Experimental findings show that difference of the power ranges between 1% to 3% and there is a positive impact of dynamic threshold on efficient utilization of resources. Dynamic threshold keeps changing according to the situation, thus lowering the SLA violations and increasing the upper threshold. Increase in upper threshold permits server to accommodate more VMs, thereby decreasing the number of active servers which ultimately reduces the energy consumption. In other words, when value of threshold is high, server will able to host more VMs, thus reducing the number servers and overall energy consumption. However, reduction in active servers may also increase the SLA violations, because, free resources may not be available on a server to fulfill the increased resource demands by hosting VMs.

Comparison of the maximum, average, and minimum SLA violation of both selected and proposed techniques with dynamic and static thresholds are presented in Fig. 6. The proposed SMaxCap and SRemCap schemes perform better than their energy-aware versions (MaxCap and RemCap) using dynamic thresholds in terms of SLA violation. Dynamic thresholds have shown better performance as they are adjustable according to the server's condition. If higher SLA violations are experienced in previous allocations, then threshold value will be reconfigured for future allocations to avoid the SLA violations. Static thresholds on the other hand are fixed values which cannot be changed at real-time. In addition, it can be seen that SLA-aware versions of selected and proposed techniques have shown better performance in terms of SLA as performance of SLA-aware versions with dynamic threshold is improved up to 1% to 2% than their counterparts i.e., MaxCap and RemCap. Difference of MaxCap and SMaxCap is 0.2% while the difference of RemCap and SRemCap is 1%.

Fig. 7 presents the comparison carried out on performance degradation due to migration by various selected and proposed schemes using dynamic and static thresholds. This comparison is performed taking maximum, average, and minimum values of performance degradation with static and dynamic thresholds which is presented in Fig. 7a and Fig. 7b, respectively. Graphs show that SLA-aware versions have lower variations with dynamic threshold as compared to static threshold. Moreover, proposed techniques perform better with dynamic threshold. It is clear that 1% to 4% performance has been improved by the SLA-aware versions



	CREW	SCREW	MaxCap	SMaxCap	RemCap	SRemCap
Max	0.11	0.1	0.13	0.115	0.11	0.101
Average	0.1036	0.0918	0.118	0.1064	0.1006	0.0928
Min	0.099	0.08	0.11	0.097	0.091	0.089

(a)

	CREW	SCREW	MaxCap	SMaxCap	RemCap	SRemCap
Max	0.14	0.111	0.169	0.16	0.152	0.143
Average	0.1224	0.104	0.1502	0.1432	0.132	0.1234
Min	0.11	0.1	0.139	0.13	0.12	0.111

(b)

FIGURE 7. Average performance degradation due to migrations (a) Max, average, and min of all workloads with dynamic thresholds (b) Max, average, and min of all workloads with static thresholds.

(SMaxCap and SRemCap) with dynamic threshold as compared to energy-aware versions (MaxCap and RemCap).

To summarize, results show that for the placement of VMs, the capacity of both CPU and RAM cannot be neglected. Moreover, taking into account the capacity of CPU and RAM is also critical to save energy. SLA versions of our proposed techniques displayed reduced SLA violations which improves the performance by reducing VM migrations. Our proposed scheme's primary objective is to reduce energy consumption. However, there is a tradeoff exists between energy conservation and SLA violations. If we reduce energy consumption, it will definitely increase SLA violations and migrations (as we have to migrate VM if the desired capacity of resources are not available on the server).

VI. CONCLUSION AND FUTURE WORK

In this paper, for efficient resource management, we added the capacity of CPU and RAM into domain along with energy consumption to improve the server selection for VM placement. We compared our proposed energy-aware schemes with their counter parts and also with existing techniques. Results display that our energy-aware versions namely, MaxCap and RemCap improves in terms of energy consumption whereas our proposed SLA-aware versions i.e., SMaxCap and SRemCap handled 6% to 8% resultant SLA violations occurred during consolidation of VMs. It has also been observed that techniques perform better with dynamic threshold as compared to static threshold, and the dynamic threshold has a positive impact on minimizing SLA violations. In our future work, we will consider the energy consumed by the Network Interface Card (NIC) in addition to CPU and RAM, as it has major role during the migration of VM. Moreover, in addition to energy and SLA, other metrics will also be considered, such as, network load, load balancing, and fault tolerance, etc.

REFERENCES

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

[2] *Microsoft Azure*. Accessed: Jan. 21, 2020. [Online]. Available: http://www.windowsazure.com

[3] K. Bilal, S. U. R. Malik, O. Khalid, A. Hameed, E. Alvarez, V. Wijaysekara, R. Irfan, S. Shrestha, D. Dwivedy, M. Ali, U. S. Khan, A. Abbas, N. Jalil, and S. U. Khan, "A taxonomy and survey on green data center networks," *Future Gener. Comput. Syst.*, vol. 36, pp. 189–208, Jul. 2014.

[4] D. Kondo, B. Javadi, P. Malecot, F. Cappello, and D. P. Anderson, "Cost-benefit analysis of cloud computing versus desktop grids," in *Proc. IEEE Int. Symp. Parallel Distrib. Process.*, May 2009, pp. 1–12.

[5] M. R. Mesbahi, M. Hashemi, and A. M. Rahmani, "Performance evaluation and analysis of load balancing algorithms in cloud computing environments," in *Proc. 2nd Int. Conf. Web Res. (ICWR)*, Apr. 2016, pp. 145–151.

[6] E. Arzuaga and D. R. Kaeli, "Quantifying load imbalance on virtualized enterprise servers," in *Proc. 1st Joint WOSP/SIPEW Int. Conf. Perform. Eng. (WOSP/SIPEW)*, 2010, pp. 235–242.

[7] L. Chen, H. Shen, and K. Sapra, "RIAL: Resource intensity aware load balancing in clouds," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, Apr. 2014, pp. 1294–1302.

[8] A. Sallam and K. Li, "A multi-objective virtual machine migration policy in cloud systems," *Comput. J.*, vol. 57, no. 2, pp. 195–204, Feb. 2014.

[9] M. Tarighi, S. A. Motamedi, and S. Sharifian, "A new model for virtual machine migration in virtualized cluster server based on fuzzy decision making," 2010, *arXiv:1002.3329*. [Online]. Available: http://arxiv.org/abs/1002.3329

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

[11] S. Mustafa, B. Nazir, A. Hayat, A. U. 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.

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

[13] 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, Mar. 2013.

[14] B. Gul, I. A. Khan, S. Mustafa, O. Khalid, and A. U. R. Khan, "CPU–RAM-based energy-efficient resource allocation in clouds," *J. Supercomput.*, vol. 75, no. 11, pp. 7606–7624, Nov. 2019.

[15] H. Rong, H. Zhang, S. Xiao, C. Li, and C. Hu, "Optimizing energy consumption for data centers," *Renew. Sustain. Energy Rev.*, vol. 58, pp. 674–691, May 2016.

[16] S. Arman, S. Smith, D. Sartor, R. Brown, M. Herrlin, J. Koomey, E. Masanet, N. Horner, and W. Lintner. (2016). *United States Data Center Energy Usage Report*. Accessed: Jan. 21, 2020. [Online]. Available: https://www.osti.gov/biblio/1372902

[17] B. Richard, "Report to congress on server and data center energy efficiency: Public law 109-431," Ernest Orlando Lawrence Berkley Nat. Lab., Univ. California, Oakland, CA, USA, Tech. Rep., 2008, pp. 109–431.

[18] A. Hameed, A. Khoshkbarforoushha, R. Ranjan, P. P. Jayaraman, J. Kolodziej, P. Balaji, S. Zeadally, Q. M. Malluhi, N. Tziritas, A. Vishnu, S. U. Khan, and A. Zomaya, "A survey and taxonomy on energy efficient resource allocation techniques for cloud computing systems," *Computing*, vol. 98, no. 7, pp. 751–774, Jul. 2016.

[19] 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.

[20] R. Yunus, O. Arif, H. Afzal, M. F. Amjad, H. Abbas, H. N. Bokhari, S. T. Haider, N. Zafar, and R. Nawaz, "A framework to estimate the nutritional value of food in real time using deep learning techniques," *IEEE Access*, vol. 7, pp. 2643–2652, 2019.

[21] H. Waheed, S.-U. Hassan, N. R. Aljohani, J. Hardman, S. Alelyani, and R. Nawaz, "Predicting academic performance of students from VLE big data using deep learning models," *Comput. Hum. Behav.*, vol. 104, no. 1, Mar. 2020, Art. no. 106189.

[22] P. Thompson, R. Nawaz, J. McNaught, and S. Ananiadou, "Enriching news events with meta-knowledge information," *Lang. Resour. Eval.*, vol. 51, no. 2, pp. 409–438, Jun. 2017.

[23] F. Anwaar, N. Iltaf, H. Afzal, and R. Nawaz, "HRS-CE: A hybrid framework to integrate content embeddings in recommender systems for cold start items," *J. Comput. Sci.*, vol. 29, no. 1, pp. 9–18, Nov. 2018.

[24] H. Qadir, O. Khalid, M. U. S. Khan, A. U. R. Khan, and R. Nawaz, "An optimal ride sharing recommendation framework for carpooling services," *IEEE Access*, vol. 6, pp. 62296–62313, 2018.

[25] S. Ayyaz, U. Qamar, and R. Nawaz, "HCF-CRS: A hybrid content based fuzzy conformal recommender system for providing recommendations with confidence," *PLoS ONE*, vol. 13, no. 10, 2018, Art. no. e0204849.

[26] A. Beloglazov and R. Buyya, "Energy efficient allocation of virtual machines in cloud data centers," in *Proc. 10th IEEE/ACM Int. Conf. Cluster, Cloud Grid Comput.*, May 2010, pp. 577–578.

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

[28] 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.*, vol. 1, Mar. 2014, pp. 320–326,

[29] 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.

[30] Y. Ren, J. Suzuki, C. Lee, A. V. Vasilakos, S. Omura, and K. Oba, "Balancing performance, resource efficiency and energy efficiency for virtual machine deployment in DVFS-enabled clouds: An evolutionary game theoretic approach," in *Proc. Conf. Companion Genetic Evol. Comput. Companion—GECCO Comput.*, 2014, pp. 1205–1212.

[31] G. B. Mertzios, M. Shalom, A. Voloshin, P. W. H. Wong, and S. Zaks, "Optimizing busy time on parallel machines," *Theor. Comput. Sci.*, vol. 562, pp. 524–541, Jan. 2015.

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

[33] 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, May 2012.

[34] 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. Dependable Secure Comput.*, vol. 10, no. 5, pp. 253–272, Sep. 2013.

[35] T. M. S. Nowicki and C. W. S. Wu, "Fundamentals of dynamic decentralized optimization in autonomic computing systems," in *Proc. Self-star Workshop*. Berlin, Germany: Springer, 2004, pp. 204–218.

[36] D. Ardagna, B. Panicucci, M. Trubian, and L. Zhang, "Energy-aware autonomic resource allocation in multitier virtualized environments," *IEEE Trans. Services Comput.*, vol. 5, no. 1, pp. 2–19, Jan. 2012.

[37] S. Ali, S. Y. Jing, and S. Kun, "Profit-aware DVFS enabled resource management of IaaS cloud," *Int. J. Comput. Sci. Issues*, vol. 10, no. 2, p. 237, 2013.

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

[39] 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, Jun. 2016.

[40] J. Bi, H. Yuan, W. Tan, M. Zhou, Y. Fan, J. Zhang, and J. Li, "Application-aware dynamic fine-grained resource provisioning in a virtualized cloud data center," *IEEE Trans. Autom. Sci. Eng.*, vol. 14, no. 2, pp. 1172–1184, Apr. 2017.

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

[42] J. Bi, H. Yuan, M. Tie, and W. Tan, "SLA-based optimisation of virtualised resource for multi-tier Web applications in cloud data centres," *Enterprise Inf. Syst.*, vol. 9, no. 7, pp. 743–767, Oct. 2015.

[43] J.-K. Dong, H.-B. Wang, Y.-Y. Li, and S.-D. Cheng, "Virtual machine placement optimizing to improve network performance in cloud data centers," *J. China Universities Posts Telecommun.*, vol. 21, no. 3, pp. 62–70, Jun. 2014.

[44] *SPEC Power Benchmark*. Accessed: Jan. 21, 2020. [Online]. Available: https://www.spec.org/power_ssj2008

[45] M. Ranjbari and J. A. Torkestani, "A learning automata-based algorithm for energy and SLA efficient consolidation of virtual machines in cloud data centers," *J. Parallel Distrib. Comput.*, vol. 113, pp. 55–62, Mar. 2018.

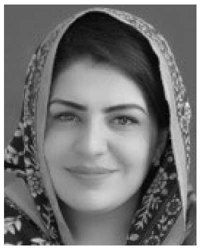
[46] The CLOUDS Lab. (2009). *CloudSim: A Framework for Modeling and Simulation of Cloud Computing Infrastructures and Services*. Accessed: Jan. 21, 2020. [Online]. Available: http://www.cloudbus.org/cloudsim/

[47] *Amazon EC2*. Accessed: Jan. 21, 2020. [Online]. Available: https://aws.amazon.com/ec

[48] S. Mustafa, K. Bilal, S. U. R. Malik, and S. A. Madani, "SLA-aware energy efficient resource management for cloud environments," *IEEE Access*, vol. 6, pp. 15004–15020, 2018.

[49] PlanetLab. (2015). *An Open Platform for Developing, Deploying, and Accessing Planetary Scale Services*. Accessed: Jan. 21, 2020. [Online]. Available: http://www.planet-lab.org

[50] N. Chung, X. D. Zhang, A. Kreamer, L. Locco, P.-F. Kuan, S. Bartz, P. S. Linsley, M. Ferrer, and B. Strulovici, "Median absolute deviation to improve hit selection for genome-scale RNAi screens," *J. Biomolecular Screening*, vol. 13, no. 2, pp. 149–158, 2008.



BEENISH GUL received the B.S. and M.S. degrees in computer science from COMSATS University Islamabad, Abbottabad campus, Pakistan, in 2013 and 2018, respectively. Her research interests include resource management in clouds, energy efficient systems, fog computing, edge computing, and wireless networks.

IMRAN ALI KHAN received the master's degree from Gomal University, Dera Ismail Khan, Pakistan, and the Ph.D. degree from the Graduate University Chinese Academy of Sciences, China. He is currently an Associate Professor and the Head of the Department Computer Science, COMSATS University Islamabad, Abbottabad campus. He has produced over 50 publications in journal of international repute and presented articles in International conferences. His research interests include wired and wireless networks, and distributed systems.

SAAD MUSTAFA received the M.Sc. and Ph.D. degrees in computer science from COMSATS University Islamabad, Pakistan, in 2010 and 2018, respectively. He is currently an Assistant Professor with COMSATS University Islamabad. His research interests include resource management/allocation in cloud, fog and edge, application mapping, energy-efficient systems, and network performance evaluation.

OSMAN KHALID received the master's degree from the Center for Advanced Studies in Engineering and the Ph.D. degree from North Dakota State University, USA. He is currently an Assistant Professor with COMSATS University Islamabad, Abbottabad campus. His research interests include recommender systems, network routing protocols, the Internet of Things, and fog computing. His website is: http://osman.pakproject.com

SYED SAJID HUSSAIN received the Master of Science degree in computer science from COMSATS University Islamabad (CUI), Pakistan, in 2007, and the Ph.D. degree from FernUniversität in Hagen, Germany, in 2013. He is currently an Assistant Professor with CUI, Abbottabad campus. His research interests include collaborative computing, human–computer interaction, and distributed systems.

DARREN DANCEY is the interim Head of the School for Computing, Mathematics, and Digital Technology. His substantive post is the Head of the Division: Computer Science and Information Systems. He has a research background in artificial intelligence with a Ph.D. in artificial neural networks. His teaching interests include software development. He has recently taught courses in data structures and algorithms, comparative programming languages, and applied courses, such as website development and mobile application development. In recent years, he has concentrated on creating collaborations and knowledge exchange between universities and industry with a focus on the SME sector. He has led several large projects funded by the Innovate U.K., the Digital R&D Fund for the Arts, and the European Research Council. He is also on the organizing committee for the Manchester Raspberry Pi Jam and sit on the BCS Manchester branch committee.

RAHEEL NAWAZ is the Director of Digital Technology Solutions and a Reader of analytics and digital education with Manchester Metropolitan University (MMU). He has founded and/or headed several research units specializing in Artificial Intelligence, Digital Transformations, Data Science, Digital Education, and Apprenticeships in Higher Education. He has led on numerous funded research projects in the U.K., European Union, South Asia, and Middle East. He holds adjunct or honorary positions with several research, higher education, and policy organizations, in U.K. and overseas. He regularly makes media appearances and speaks on a range of topics including Digital Technologies, Artificial Intelligence, Digital Literacy, and Higher Education. He was an Army Officer. Before becoming a full-time academic, he served in various senior leadership positions in the private Higher and Further Education sector.