Renewable and Sustainable Energy Reviews 167 (2022) 112782

Contents lists available at ScienceDirect

Renewable and Sustainable Energy Reviews

journal homepage: www.elsevier.com/locate/rser

Utilizing power consumption and SLA violations using dynamic VM consolidation in cloud data centers

Umer Arshad [a], Muhammad Aleem [b], Gautam Srivastava [c,d], Jerry Chun-Wei Lin [e,*]

[a] The University of Lahore, Lahore 54000, Pakistan
[b] National University of Computer and Emerging Sciences, Islamabad 44000, Pakistan
[c] Brandon University, Brandon, MB R7A 6A9, Canada
[d] China Medical University, 40402, Taichung, Taiwan
[e] Western Norway University of Applied Sciences, 5063, Bergen, Norway

ARTICLE INFO

Keywords:
Cloud computing
Service level agreement
Energy efficiency
VM migrations
Real-time cloud data centers

ABSTRACT

Cloud Computing services can be accessed anytime, anywhere via the Internet. The overwhelming growth of cloud data centers over the past decade has increased their costs as energy demands have risen. As a result, higher carbon dioxide emissions and other greenhouse gasses are putting a strain on our ecosystem. The main objective of this study is to reduce the power consumption in cloud computing with no or negligible trade-offs in quality of service. This paper presents a new algorithm called the energy efficiency heuristic using virtual machine consolidation to minimize the high energy consumption in the cloud. By setting two thresholds, hosts are classified into three main classes. The designed model reallocates virtual machines from one physical host to another to minimize energy consumption. The results of the proposed algorithm have been obtained in terms of virtual machine migrations, performance degradation caused by migration, service level agreement violations, and execution time, showing a significant improvement over state-of-the-art techniques.

1. Introduction

Many individuals and businesses worldwide use Cloud Computing (CC) [1–3] to obtain storage and computing services accessible via the Internet. The general use of cloud computing has increased manifold due to its easy accessibility, excellent scalability, cost efficiency, and reliability [4–6]. Cloud services can be used anytime and anywhere via the Internet [7]. Cloud models are divided into three services: Software, Platform, and Infrastructure as a Service [8]. Software-as-a-Service (SaaS) allows users to utilize online software applications over the Internet; an example is Google-Drive, Dropbox, Facebook, etc. Platform-as-a-Service (PaaS) provides a framework for programmers to develop their customized cloud applications in Microsoft Azure and Google Application Engine. Infrastructure-as-a-Service (IaaS) provides virtualized computing resources such as Central Processing Unit (CPU), bandwidth, storage, and memory in the form of Virtual Machines (VMs) [9–11]. The benefits associated with each model are unique and different from one another, and the cloud promises to meet the needs of different types of businesses [12]. The cloud user and provider often agree on specific terms and conditions for the use of cloud services, called a Service Level Agreement (SLA). A SLA often describes the requirements and the Quality of Service (QoS) between the Cloud users and the service provider [13].

Given the large cloud data centers of today, colossal energy consumption is one of the main problems [14,15]. A lot of energy is needed in a data center to run the cooling systems. In addition, computing, storage, and networking equipment consume a lot of power. The energy consumed ultimately reflects high CO_2 emissions that impact the biosphere [4]. According to a report by the Natural Resource Defense Council [5], data centers in the United States of America consume about 91 billion kWh of electricity, and this number is estimated to increase to 200 billion kWh by 2030.

The increase in electricity consumption also increases the cost of the business model and consequently lowers productivity [16]. According to a study [4], Cloud Computing (CC) consumes more energy than most countries worldwide. To illustrate, if we consider `Cloud` as a country, it would be the fifth-largest country in terms of energy consumption [4]. In the cloud, the main components that contribute to this enormous energy consumption are CPU, memory, networking, storage, cooling and power consumption, etc., as shown in Fig. 1.

As highlighted in Fig. 1, there is an urgent need to reduce energy consumption without compromising QoS. Recent advances in hardware

* Corresponding author.
E-mail addresses: umer.arshad@cs.uol.edu.pk (U. Arshad), m.aleem@nu.edu.pk (M. Aleem), SRIVASTAVAG@brandonu.ca (G. Srivastava), jerrylin@ieee.org (J.C.-W. Lin).

https://doi.org/10.1016/j.rser.2022.112782
Received 26 April 2022; Received in revised form 3 July 2022; Accepted 7 July 2022
Available online 27 July 2022

Nomenclature

Abbreviations

EEHVMC	Energy Efficiency Heuristic with Virtual Machine Consolidation
QoS	Quality of Service
VM	Virtual Machine
VMM	Virtual Machine Migration
PDM	Performance Degradation caused by VM migration
SLATAH	Service Level Agreement Violations Time per Active Host
IQR	inter-quartile range
HOL	Host Over-Loaded
HML	Host Medium-Loaded
HUL	Host Under-Loaded
MIPS	Millions of Instructions Per Second
MSU	Minimum Size Utilization
CATR	Cumulative Available-to-Total Ratio
EVMC	Energy-Aware VM Consolidation
SABFD	Space Aware Best Fit Decreasing
EEOM	Energy Efficiency Optimization of Virtual Machine Migrations
RALBA	Resource Aware Load Balancing Algorithm
EFT	Earliest Finish Time
ETSA	Energy-Efficient Task Scheduling Algorithm
MU	Maximum Utilization
MMT	Minimum Migration Time MMT
RS	Random Selection
MC	Maximum Correlation
DVMC	Dynamic Virtual Machine Consolidation
SLAV	Service Level Agreement Violations
DVFS	Dynamic Voltage and Frequency Scaling
QoS	Quality of Service
SLA	Service Level Agreement
CPU	Central Processing Unit
IaaS	Infrastructure-as-a-Service
PaaS	Platform-as-a-Service
SaaS	Software-as-a-Service
SOTA	State-of-the-art
DNN	Deep Neural Networks
CC	Cloud Computing

technology have reduced the energy consumption of a CC system [18–20].

One of the fundamental mechanisms for energy conservation is Dynamic Voltage and Frequency Scaling (DVFS). This approach automatically changes the voltage and frequency to reduce processor heat dissipation and lowers the power consumption. In addition, the reduced heat generation allows cooling systems to be turned off, saving more energy [21,22]. Another related approach to energy conservation in cloud computing is Dynamic Virtual Machine Consolidation (DVMC). The DVMC approach reallocates VMs from one host to another to reduce the number of active hosts in the data center by putting inactive hosts into power-saving mode to conserve the power [23–25].

This study presents an energy-conserving framework based on the concept of VM Consolidation. The proposed framework is a heuristic model. A heuristic approach speeds up the process to locate a suitable solution. A heuristic is an approach for problem-solving and is effective

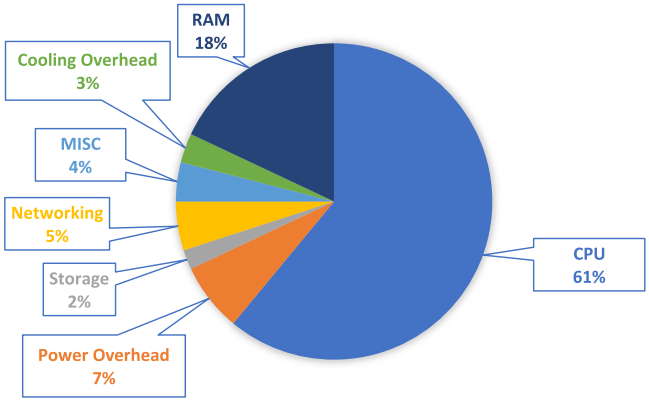

Fig. 1. Breakdown of power consumption in data centers [17].

where the classic techniques are slow, or an exact solution is not desirable (considering the employed overhead). The purpose of a heuristic is to find a solution in a reasonable time that is good enough to solve the problem at hand [26]. The heuristic solution is not optimal, however, the solution provides approximately the exact answer. Therefore, the proposed scheme is designed as a heuristic approach to quickly finds the placement strategy to conserve energy in the Cloud. Almost 79% of the consumed power in the cloud comprises CPU and memory; hence these two most important aspects are the building blocks of the proposed scheme.

The proposed Energy Efficiency Heuristic using VM Consolidation (EEHVMC) reduces power consumption and SLA violations (SLAV). The main idea is to classify host machines based on CPU and memory usage. By setting two thresholds (related to CPU and memory utilization), the host machines are categorized into three main classes: Host Over-Loaded (HOL), Host Medium-Loaded (HML), and Host Under-Loaded (HUL) machines. As of HOL, we migrate VMs to the HML to minimize power consumption in the cloud data centers. In the HML, all VMs are kept unchanged. From the HUL hosts, the proposed approach reassigns the VMs to the HML, and the inactive hosts are put into power-saving mode [27].

The results show that the EEHVMC approach significantly minimizes power consumption and reduces SLAV. In summary, the contributions of this work are summarized below:

- Detail analysis of the literature examines the strengths and weaknesses of existing VM consolidation heuristics.
- A novel scheduling mechanism, EEHVMC consolidates VMs on host machines to reduce power consumption, VM migrations, performance degradation, and Service Level Agreement Violations in the cloud.
- Experimentation and evaluation of the intended approach compared to state-of-the-art VM consolidation heuristics.

The remainder of the paper is organized as follows: Section 2 discusses related work. Section 3 states the performance model used in this study, SLAV, and PDM. Then, Section 4 presents the system architecture, EEHVMC algorithm, and complexity analysis. Experimental results and discussions are presented in Section 5. Finally, Section 6 concludes the paper and explains future work.

2. Related work

Beloglazov et al. [4] conducted a competitive study to mitigate the problems associated with VMM and DVMC. The results state that developing a randomized or adaptive approach is essential to improving the performance of optimal deterministic algorithms. In addition, novel adaptive heuristics were proposed based on a retrospective study of

the resource method for VM performance- and energy-efficient consolidation. The technique significantly reduces energy consumption while ensuring high compliance with SLA. CPU, memory, network interfaces, and disk storage are essential for determining to compute node energy consumption. This approach mitigates energy consumption only for CPU usage, while memory, network interfaces, and disk storage are the components responsible for host power consumption in cloud computing.

Dynamic Virtual Machine Consolidation (DVMC) approach [6] reduces SLAV and power consumption in the cloud. This approach is classified into four main areas: Underload, Overload Detection, VM Selection, and Placement. The first part of this approach is to detect whether the host is overloaded or not. If the host is overloaded, depending on the upper threshold, a VM migrates to the other physical host. The VM selection policies are Minimum Migration Time (MMT), Random Selection (RS), Maximum Correlation (MC), and Maximum Utilization (MU). All VMs must migrate to another physical host if the host is underloaded. The proposed approach does not consider CPU, memory- and I/O-intensive tasks running on a VM.

Energy-Efficient Task Scheduling Algorithm (ETSA) [7] consists of three parts: Estimation, Normalization, and, Selection and Execution. The ETSA approach reduces energy consumption in several ways. The normalization part determines the smallest normalized total value resulting from the combination of end time and utilization. The execution task is assigned to the resources that have the smallest normalized total value. The ETSA technique strikes a balance between completion time and utilization and provides more reliable results. This approach balances workload and completion time but does not consider SLA.

In the first phase of the intended approach, scheduling focuses on VM computational capabilities. The VM with the Earliest Finish Time (EFT) is selected for job mapping in the second phase [11]. Resource Aware Load Balancing Algorithm (RALBA) improves resource utilization, reduces makespan, and minimizes execution time. However, RALBA is unsuitable for the dependent task, and quality of service is not part of this approach.

Adaptive three-threshold framework energy-aware algorithms categorize hosts into four categories: less, little, normal, and overloaded hosts. When a host is overloaded, virtual machines must migrate to the less busy host. All VM remain unchanged if a host is normal and a little busy. When the host is less active, all VM migrate to a less busy host. Zhou et al. [12] only considered the CPU and I/O intensive tasks on the VM. If the task is CPU intensive, the VM with a maximum CPU ratio to memory usage is selected. If the task is I/O-bound, the VM selection multiplies by the CPU and memory utilization. Zhou et al. considered the CPU and I/O-intensive tasks but ignored the memory-intensive tasks on the overloaded host. This approach minimized power consumption based only on the CPU workload, while CPU and memory are the main components responsible for host power consumption in a CC data center.

The research presented in [13] proposes a scheduling technique for CC systems that is cost-effective and saves energy. While the approach minimizes schedule gaps by performing approximate computations using per-core DVFS on different multi-core processors, it also accounts for input errors in component tasks. This study aims to maintain quality at the desired standard and provide a cost-effective solution that provides energy efficiency and timelessness with precision. However, this approach increases potential network traffic.

Energy Efficiency Optimization of Virtual Machine Migrations (EEOM) [15] consists of three steps: VM selection, trigger time, and host location. The EEOM technique migrates some lightly and heavily loaded VMs to another physical host. The inactive host is put into a power-saving mode so that the host's power consumption can minimize. The EEOM approach tries to reduce the number of running hosts but neglects the remaining factors such as cooling system, network traffic, and migration costs.

The authors aim to reduce power consumption, and SLA violations in the Space Aware Best Fit Decreasing (SABFD) [21] approach. This approach selects a VM for migration with maximum CPU utilization and places a VM on the host with lower computational power, i.e., Millions of Instructions Per Second (MIPS). It minimizes power consumption based only on CPU utilization. The SABFD does not examine the type of applications running on the VM and cannot reduce data center power consumption and minimize SLA.

Buyya et al. [28] proposed an approach to mitigate the problem of energy consumption in the cloud. This approach helps to reduce data center energy consumption and enables low-cost cloud production. This paper presents and implements an energy-aware resource allocation algorithm based on a VM consolidation mechanism. Experimental results show that this technique is efficient compared to other energy-aware approaches; however, it introduces significant overhead. Moreover, this approach does not consider the application types running on the VMs.

Liu et al. [29] presented a VM consolidation approach for a cloud computing environment. The central concept was to minimize the VMMs and thus minimize energy consumption in the cloud. The main aspects achieved by the proposed approach were: reducing the possibility of host overload, avoiding unwanted VMMs, and consequently minimizing the total number of VMMs. The proposed method improves the resource utilization of the host machines. It works very well under different workload traces. Therefore, this approach satisfies the requirement of minimizing the cost of data centers for resource providers. This technique's prospects are beneficial for service providers and end-users.

Uddin et al. [30] correspond to the proposition of a server consolidation technique to increase the efficiency of pre-installed server machines and their utility by shifting them to virtual server machines to promote eco-friendly and energy-efficient cloud data servers. A novel virtualized task scheduling algorithm evenly distributes tasks from physical server machines to virtual machines. The experimental results show that 30% could increase the efficient utilization of resources (on the deployed VMs). Moreover, the study showed that the least amount of servers used (i.e., up to 50%), resulting in significant energy savings.

Energy-Aware VM Consolidation (EVMC) [31] system implements a resource parameter-based scheme to regulate overutilized hosts in a virtual cloud environment. The comparisons of VMs and hosts determine for analyzing overloaded hosts, while the Cumulative Available-to-Total Ratio (CATR) uses to determine the underutilized hosts. Transfer VMs to appropriate hosts; VM placement uses a criterion based on normalized resource parameters of hosts and virtual machines. Several tests were performed on many virtual machines using traces from PlanetLab workloads to calculate the performance of VM consolidation. The results show that the EVMC approach is on par with other well-known methods by improving energy savings, SLA violations, and the number of VM migrations.

The GradeCent algorithm [32] uses the Stochastic Gradient Descent technique. This technique promotes an upper CPU utilization threshold for detecting overloaded hosts using an actual CPU workload. In addition, the authors proposed a dynamic VM selection algorithm, i.e., Minimum Size Utilization (MSU), to select the VMs of an overloaded host for VM consolidation. Gradient and MSU maintain the tradeoff between minimizing energy consumption and maximizing QoS among the specified SLA objectives. The proposed algorithms focus on increasing energy saving and violation of SLA by 23% and 27.5% on average, respectively, compared to the baseline methods.

In [33], authors present a frequency-aware management technique for controlling processors' dynamic and static power in data centers that operate virtual machines. A frequency-aware model capable of determining the best frequency ratio for reducing processor energy usage. The energy consumption of a data center may be enhanced with this model in place by altering the processor's rate to meet the appropriate frequency ratio. This paper devises a management strategy for intelligently adjusting the frequency ratio to save energy while compliant with virtual machine frequency requirements. The result

Table 1
Summary of literature review techniques.

Heuristics	Strengths	Weaknesses
Optimal Online Deterministic Algorithms [4]	Minimize the total number of active hosts, live migrations, and performance degradation	Minimize energy base only CPU utilization whereas neglects memory, and disk storage
Dynamic Virtual Machines Consolidation Algorithm [6]	Meet end-to-end performance requirements, energy-efficient, cost-effective	CPU, memory and I/O intensive tasks are not considered running on the VM
Energy-Efficient Task Scheduling Algorithm [7]	Improve resource utilization, consider heterogeneous tasks	SLA violations and temperature of host increase
Resource Aware Load Balancing Algorithm [11]	Improve resource utilization, minimal scheduling overhead, reduced makespan	SLA is not considered, not efficient for dependent tasks
Adaptive Energy-Aware Algorithm [12]	Minimize SLA and consider migration overhead	They do not consider the total amount of resource cost
An Energy-Efficient, QoS-Aware and Cost-Effective Scheduling [13]	Maintain QoS, maximize resource, energy and cost savings	Bandwidth and network traffic is also increasing
Energy Efficiency heuristic of VM Migrations [15]	CPU and memory are considered for migration purpose	Cooling system, network traffic, and migration cost are not considered
Space Aware Best Fit Decreasing [21]	Minimize resource waste, energy optimize	VM placement base on CPU utilization whereas memory is not considered
Energy-Aware Data Centre Resource Allocation [28]	CPU and memory are considered energy efficiency heuristic	Operational cost increase, neglect the application type running in the VMs
Energy-Efficient and QoS DVM Consolidation [29]	Reduce the amount of VMs migrations, low operating costs, meet SLA	Neglects the essential factors like workload, type of host, and temperature
Virtualized Task Scheduling Algorithm [30]	Reduce time, infrastructure overhead, operational costs	Required load balancing algorithm, including server and network in the data center
Energy-aware VM consolidation [31]	Improvement in Quality of Service, Meet SLA	Performance degradation
GradeCent algorithm [32]	Minimize live migration and execution time	Neglects VM placement policy
Frequency-aware DVFS model [33]	Energy efficiency of a data center maximize by adjusting the processor's frequency	Decreasing the CPU frequency will reduce the system performance
Energy Optimization Algorithm [34]	better performance in contrast to the interquartile range and local regression algorithms, high throughput	SLA violation

shows that a modest static power percentage leads to excellent energy-saving performance after studying the relationships between frequency ratio and energy usage.

This paper [34] presents an Energy Optimization Algorithm (EOA) to optimize energy without losing performance. In this technique, we identify the overload by looking at the whole workload usage of the data center. With this method, the performance-to-power ratio increase. Achieve high throughput; virtual machines must migrate less frequently. This EOA's primary purpose is to decrease the number of live migrations while preserving performance.

In summary (see Table 1), most studies [4,6,12,13,21,28–30] target to minimize energy consumption only in terms of CPU, while storage is an important component of energy consumption in cloud-host machines. Some of the proposed approaches [7,11,15,31–34] use VM migration mechanisms from overloaded host machines to underloaded hosts. However, these approaches do not define appropriate thresholds (set at runtime) to detect whether host machines are overloaded or underloaded. In addition, existing approaches do not consider the types of applications running on the VMs, which can lead to incorrect migration decisions that result in fewer energy savings and more SLA violations.

3. Evaluation models and metrics

3.1. Power model

Several studies [4,6,13,25] mainly target CPU-intensive tasks to model energy saving in a cloud data center. Memory, networking, bandwidth, cooling systems, storage system, and other specialized computing devices such as GPUs have significant energy requirements in CC data centers. Currently, our study targets the most energy-intensive resources in a cloud data center, namely CPU and memory. CPU and memory consume almost 79% of the power in the cloud data center [17], as shown in Fig. 1. Therefore, the proposed approach targets CPU and memory most energy-intensive resources in a cloud. Moreover, this study assumes two important cloud application classes: memory-intensive and CPU-intensive. The proposed scheduling heuristic considers the most prominent energy-hungry resources and guides appropriate placement/mapping schemes for VMs. The total power consumption of a physical server is composed of two components: P_s and P_u, as shown in Eq. (1) [35,36]. P_s is the fixed power consumption of the server regardless of whether VMs are operating or not, and P_u is the dynamic power utilized by the VMs running on it.

$$P_{(total)} = [P_s + P_u], \tag{1}$$

As given in Eq. (2) [36], P_{cpu} is the amount of power consumed by the CPU in the physical host computer, while P_{memory} is the amount of power consumed by the memory in the physical host machine.

$$P_{(u)} = [P_{cpu} + P_{memory}] \tag{2}$$

Several recent studies [37–40] highlight that in a host machine, almost 70% of the power is consumed when a host is idle compared to other fully utilized hosts. This fact justifies that when an inactive host turns off, it saves power significantly, resulting in excellent energy efficiency. Therefore, the energy model [4] used (in our proposed VM placement or scheduling heuristic) is the energy consumed by both active and inactive hosts, as shown in Eq. (3) [28].

$$\begin{aligned} P_{(u)} = {} & [K \times P_{maxcpu} + (1-K) \times P_{maxcpu} \times u] \\ & + [K \times P_{maxmemory} + (1-K) \times P_{maxmemory} \times u], \end{aligned} \tag{3}$$

where P_{maxcpu} is the maximum power consumption of a host (taking into account the computationally intensive tasks performed on the CPUs) when it is fully utilized [37,38]. $P_{maxmemory}$ is the maximum power consumption of a host machine (taking into account the memory-intensive tasks performed in memory) when it is fully utilized [39,40]. There are several studies [37–40] show that a host machine, although underutilized, also consumes a significant amount of energy, about 70% compared to a busy host machine. Therefore, the energy model [4] used represents this fact with the term K. The

remaining host machines (machines that are not idle) are represented by the term *1-K*. The notation *u* represents the current CPU and memory utilization of an idle host machine.

However, energy utilization may vary (i.e., increase or decrease) as CPU and the memory usage of the busy host machines vary. Therefore, the model emphasizes that CPU and memory utilization is a function of time and is represented as the term $u = (t)$. The busy host machines have different energy requirements at various execution times, depending on the percentage of CPU used during each execution. Therefore, the total energy represented by the term *E* consumption of a host machine can be interpreted by the integral power consumption function over some time, as shown in Eq. (4) [28] of the model used.

$$E = \int_{t}^{\infty} P(u(t))dt \tag{4}$$

3.2. SLA violations

SLA is an agreement between cloud providers and users. This agreement specifies the requirements, price, and QoS between the two parties [41]. Meeting the quality of service requirements is essential for cloud computing environments. These are usually formalized in terms of an SLA that can determine properties such as *minimum throughput* or *maximum response time* provided by the deployed system. The following two matrices are used to measure the SLA level in the Infrastructure as a service cloud model [28]:

(1): As shown in Eq. (5) [42], when an active host is being utilized 100%, then Service Level Agreement Violations Time per Active Host (SLATAH) can be represented as:

$$SLATAH = \frac{1}{N} \times \sum_{i=1}^{N} \frac{T_{si}}{T_{ai}} \tag{5}$$

One of the concerning aspects of SLATAH is that when a host serving an application is fully utilized (i.e., 100%), it limits the application performance [43]. The Eq. (5) [42] uses *N* as the number of hosts, T_{si} is the total time that the host machine *i* experienced full utilization (i.e., 100%) leading to SLAV, T_{ai} is the total time that the host *i* is served in the active state (i.e., serving VMs).

(2): The Performance Degradation caused by the VM Migration (PDM) as shown in Eq. (6) [42] can be formulated as follows:

$$PDM = \frac{1}{M} \times \sum_{j=1}^{M} \frac{C_{dj}}{C_{rj}} \tag{6}$$

M is the number of VMs; C_{dj} is the estimate of *Performance Degradation of VM j due to migrations* (PDM), and C_{rj} is the total CPU capacity requested by the VM *j* during its lifetime. The value of C_{dj} during the experiments was estimated to be about 10% of the CPU workload in Millions Instructions Per Second (MIPS) estimated (during all migrations of the VM *j*). Both the *SLATAH* and *PDM* metrics are independent and equally important to characterize the level of SLAV. Therefore, in this study, we propose a hybrid metric that includes both VMM and performance degradation as a result of host overloading [44]. The combined metric was presented as service level agreement violations, which is calculated as shown in Eq. (7) [28].

$$SLAV = SLATAH \times PDM \tag{7}$$

3.3. Performance

By comparing the efficiency of the algorithms with the literature, a new metric can be defined by calculating the product of energy consumption along the SLAV. *Energy* (E) and SLAV, represented in Eq. (8) [28,45].

$$ESV = E \times SLAV \tag{8}$$

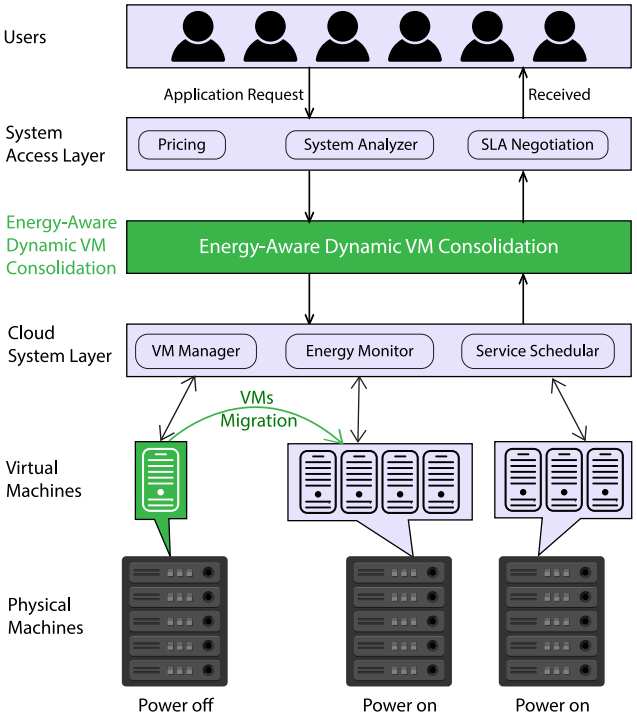

Fig. 2. Cloud computing system architecture [17].

4. Proposed energy efficiency heuristic using VM consolidation (EEHVMC)

The proposed technique is based on VM consolidation and placement heuristic to conserve energy. Like a typical heuristic based mechanism, the proposed Energy Efficiency Heuristic using VM Consolidation (EEHVMC) employ various methods to produce solutions in a reasonable time. The EEHVMC approach is designed generically and suits the various classical cloud computing framework and the specialized cloud platforms such as hadoop, spark, etc.

Cloud users can access applications anytime, anywhere via the Internet. **System Access Layer** acts as an interface between consumers and the cloud infrastructure. **Energy-Aware Dynamic VM Consolidation** moves VM from one physical host to another host to minimize power consumption. **Multiple VM** can fulfill accepted requests on a single machine and dynamically power on and off. **Physical Machines** create VM resources using hardware infrastructure to meet service requests.

The EEHVMC approach uses CPU and memory to reduce power consumption and SLAV. By defining two thresholds, T_{high} and T_{low}, the hosts in the data centers classify into three main classes; Host Over-Loaded (HOL), Host Medium-Loaded (HML), and Host Under-Loaded (HUL). First, CPU and memory utilization of the data center's host compared with the defined threshold. If $CU_{HI} \geq T_{high} \parallel MU_{HI} \geq T_{high}$, then the host is overloaded. Some of the VMs on the HOL should migrate to the HML to reduce power consumption. If $T_{high} \leq CU_{HI} \geq T_{low} \parallel T_{high} \leq MU_{HI} \geq T_{low}$ then the hosts are medium loaded and all VMs remain unchanged. In HUL, the proposed technique collects all VM and assigns them to HML to reduce the number of active hosts and put the remaining inactive hosts to sleep, as shown in Fig. 2.

4.1. An adaptive utilization threshold

Thresholds represent dynamic values affected by the computing environment (resources in the cloud data center). The inter-quartile range (IQR) defines the threshold and splits the data set into quartiles.

Table 2
EEHVMC flow chart abbreviations.

Variables	Description
HostList	Total number of Host
VMList	Total number of VM
CU_{HI}	CPU Utilization of ith Host
MU_{HI}	Memory Utilization of ith Host
CU_{vi}	CPU Utilization of ith VM
MU_{vi}	Memory Utilization of ith VM
T_{high}	High Threshold
T_{low}	Low Threshold
HOL	Host Over-Loaded
HML	Host Medium-Loaded
HUL	Host Under-Loaded
VM_i	First VM on a certain Host
VM_M	Last VM on a certain Host
CRU_H	Current Resource Utilization of Host
LRU_H	Less Resource Utilization of Host

The difference between a data set's upper and lower quartile is the following step to determine the interquartile range. First, arrange the data in ascending order. The second step is calculating the ordered set's median (Q2). The third step is to separate the data in half and find the median of the first half of the ordered set (lower quartile Q1) and the median of the second half (upper quartile Q3). The final step is $IQR = UpperQuartile - LowerQuartile$. We present a strategy that uses two thresholds: an upper threshold and a lower threshold. The lower threshold calculates using the median of the first half of the ordered dataset (host utilization). In contrast, the upper threshold calculates using the median of the second half of the ordered data set (host utilization). For example, the proposed approach uses two thresholds T_{high} and T_{low}. When a host observes the CPU or memory utilization, it compares them to the defined thresholds for that particular data center. If the threshold T_{high} exceeds, the proposed algorithm assumes that the current host machine is overloaded, which means that VM migrations from that host machine are required. If the threshold is below the threshold T_{low}, the host machine of a particular data center is underloaded, so VMs migrate to other hosts (see Table 2).

4.2. CPU and memory intensive tasks

There are two types of applications that run on virtual machines: *CPU* and *memory-bound* applications. If most of an application's execution time is used for compute-intensive operations, it is called a CPU-intensive application (i.e., the CPU resource is used in most cases). The VM with the maximum CPU consumption (due to most computationally intensive tasks) is selected for migration. Of these selected VMs, the VM with the ultimate memory consumption pattern is selected first. Transferring a large amount of memory may consume more resources, but due to the host's memory source limitation, the slower migration will likely result in more delays. For example, the new task might not be processed due to insufficient memory, resulting in a higher rate of system violations. Effectively migrating such a VM also frees up more memory on the host, which can use to allocate new incoming tasks.

4.3. Host over-loaded

If $CU_{HI} \geq T_{high} \parallel MU_{HI} \geq T_{high}$, then the host is overloaded. Each virtual machine in a host is given CPU (CU_{vi}) and memory utilization (MU_{vi}), then it is compared to the defined thresholds, e.g., T_{high} and T_{low}. If $CU_{vi} \geq T_{high} \parallel MU_{vi} \geq T_{high}$, then the VM is overloaded.

The proposed EEHVMC system detects which type of application consumes the most power. Suppose most of the power consumption is related to CPU-bound applications (VMs with high CPU utilization). In that case, the VMs migrates to the HML that consumes fewer CPU resources to reduce the CPU-intensive load from the overloaded and underloaded hosts. After the VMs migrates, the host workloads are updated accordingly. Similarly, the memory-intensive VMs migrates to HML, which consumes fewer memory-related resources. The host machine utilization in terms of memory usage is updated for both host machines involved in the migration, as shown in Fig. 3.

4.4. Host medium-loaded

If $T_{high} \leq CU_{HI} \geq T_{low} \parallel T_{high} \leq MU_{HI} \geq T_{low}$, then the host is medium loaded, and all virtual machines remain unchanged.

4.5. Host under-loaded

As shown in Fig. 3, Host Under-Loaded (HUL) moves all VMs to medium-loaded hosts to reduce the number of active hosts, and inactive hosts are forced to low-power mode to reduce energy consumption.

4.6. Virtual machine selection and placement

In this study, we assume that the workload can be CPU or memory intensive. A task is CPU intensive if its completion depends mainly on the use of CPU resources, while a task that spends most of its time interacting with memory (i.e., spends most of its time performing load/store operations) is said to be memory intensive. In the Scheduling model, the earliest finish time of all most significant jobs determines by considering all VMs in the first stage. The second stage selects the VM with the maximum CPU or memory utilization and assigns it to the concerned VM in the host medium load. The ready time of virtual machines updates after each schedule, and this process repeats until all jobs execute successfully. We propose a new VMs selection method named MRCU (Maximum ratio of CPU utilization to memory utilization) to select VMs for migration when CPU-intensive tasks overload a host. Let the CPU and memory utilization of i VM by C^u_{vm} *and* M^u_{vm} respectively. Let CPU and memory utilization of any VM e be referred by C^e_{vm} *and* M^e_{vm} respectively. The MRCU technique chooses a VM v from the host for migration if it fits the following criteria:

$$\frac{C^u_{vm}}{M^u_{vm}} > \frac{C^e_{vm}}{M^e_{vm}} \tag{9}$$

Eq. (9) [12] shows that the lower M^u_{vm} value, the higher C^u_{vm} value, higher C^u_{vm} / M^u_{vm} value is. As a result, Eq. (9) chooses the VM with the highest C^u_{vm} / M^u_{vm} value to migrate, because higher CPU usage indicates more energy consumption. When transferring possible VMs, the MRCU technique considers both the CPU and memory factors. For example, if the server is overloaded with CPU -intensive tasks, the power consumption of CPU (s) will account for most of the total energy consumption compared to the other components of the host machine. Therefore, the algorithm selects a VM with the highest CPU value for migration, since a higher CPU workload means higher power consumption (the objective is to save energy).

Assuming the tasks are memory intensive, the virtual machine with the highest memory use selects for migration. Although migrating extensive memory data may consume more resources because the host's memory source is overloaded, delayed migration will likely cause more significant damage. For example, the new task may not be received due to insufficient memory, increasing the rate of system corruption. Timely migration of such virtual machines can also free up additional RAM on the host, which can use for the new task. The scheduling model shifts the memory usage of the overloaded host and sends these tasks to HML, which uses less memory in HML, and then changes the memory resources in the medium-loaded host.

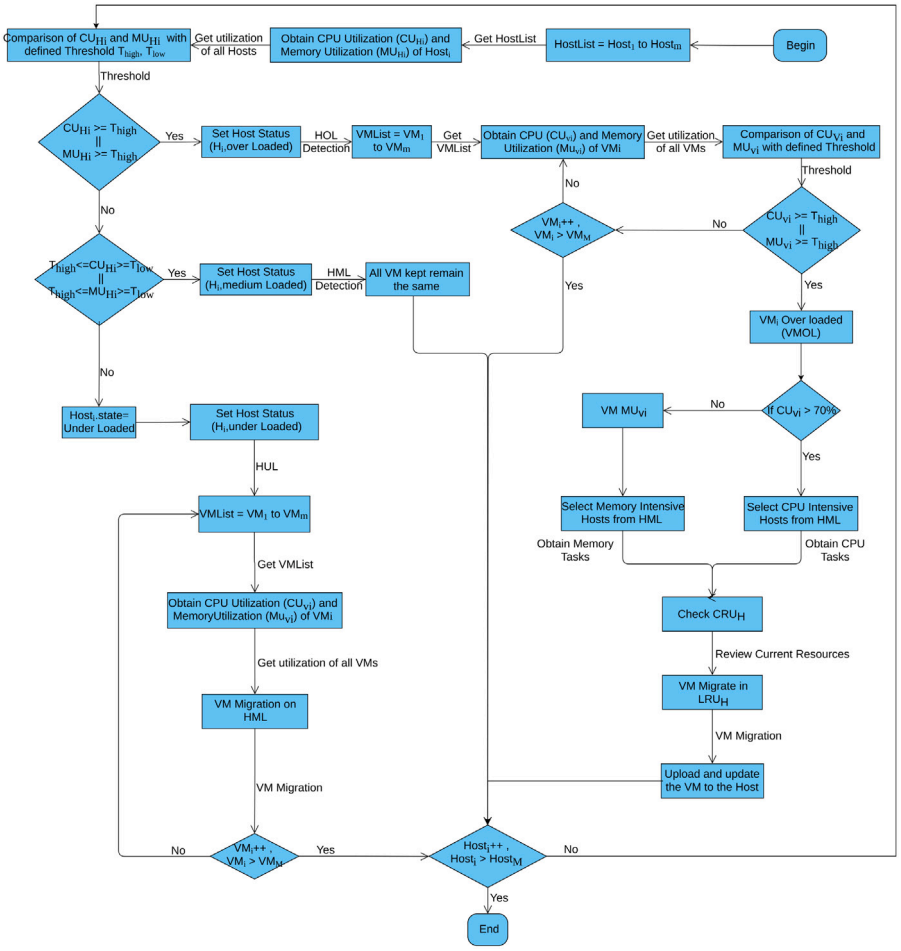

Fig. 3. Energy efficiency heuristic using vm consolidation.

4.7. EEHVMC algorithm

The proposed methodology reduces both power consumption and Service Level Agreement (SLA). The "vmlist" is a collection of VMs in the physical host and the "hostlist" is a list of hosts in the cloud data center (see Algorithm 1). Thresholds T_{high} and T_{low} have been set, which divide the host into three main classes (see Algorithm 1).

EEHVMC return MigrationMap scheme in which VM placement policy scheme is saving. When the VM migrates, it checks the next VM from HOL (lines 10–17, Algorithm 1).

If $T_{high} \leq cU_{Hi} \geq T_{low} \parallel T_{high} \leq mU_{Hi} \geq T_{low}$, then it will be considered as HML. In HML, there is no requirement for any migration. All of the VMs on that host are left undisturbed (lines 18–19, Algorithm 1).

If both conditions are not met, the host is classified as underutilized (lines 20–21, Algorithm 1). A loop is executed in HUL to determine the CPU and memory usage of this VM (lines 22, Algorithm 1). To save energy wasted by inactive hosts in the HUL (lines 23–25, Algorithm 1), the algorithm collects all VM from the HUL and moves them to the HML to reduce the number of active hosts and shut down the remaining empty hosts. EEHVMC return MigrationMap scheme in which VM placement policy scheme is saving. When the VM migrates, it checks the next virtual machines from HUL (lines 26–29, Algorithm 1) (see Table 3).

"Hostlist" is a collection of hosts stored in variable *m*, while `vm-list` is a collection of VMs stored in variable *n* (lines 1–2, Algorithm 1). A loop is executed in the variable "hostlist" (line 3, Algorithm 1) and determines CPU and the memory usage of that host (lines 4, Algorithm 1). If CPU or the memory usage is greater than the T_{high} values, then it is a HOL (lines 5–6, Algorithm 1). A loop is made in the "vmlist" (line 7) and determines CPU and the memory utilization of that VM (lines 8, Algorithm 1). It is a VM over-loaded (lines 9, Algorithm 1) if CPU or the memory usage is above the T_{high} values. A migration scheme is mapped that sends a VM from HOL to HML.

Table 3
EEOVMS algorithms abbreviations.

Variables	Description
Hostlist	Total number of Host
vmlist	Total number of VM
VM_i	First VM on a certain Host
VM_M	Last VM on a certain Host
cU_{HI}	CPU Utilization of ith Host
mU_{HI}	Memory Utilization of ith Host
cU_{vj}	CPU Utilization of jth VM
mU_{vj}	Memory Utilization of jth VM
T_{high}	High Threshold
T_{low}	Low Threshold
vmOverLoaded	Virtual Machine Over-Loaded

4.8. Algorithm complexity

We store the host and VM list in a separate data structure that takes $O(nlogn)$ time. We retrieve the virtual machine from the data structure and check its load for each host, which also requires constant time complexity. Then we select the VMs stored in the structure to distribute the load. This also takes $O(n)$ time, where n is the number of VMs. So

input : hostlist, vmlist, T_{high}, T_{low}
output: migration scheme
$m \leftarrow getCount(hostlist)$;
$n \leftarrow getCount(vmlist)$;
for $i \leftarrow 1$ **to** m **do**
  $cU_{Hi} \leftarrow getcU_{Hi}(host)$ and $mU_{Hi} \leftarrow getmU_{Hi}(host)$;
  if $cU_{Hi} \geq T_{high} || mU_{Hi} \geq T_{high}$ **then**
    $hostOverLoaded \leftarrow getHostOverLoaded$;
    for $j \leftarrow 1$ **to** n **do**
      $cU_{vj} \leftarrow getcU_{vj}(vm)$ and $mU_{vj} \leftarrow getmU_{vj}(vm)$;
      if $cU_{vj} \geq T_{high} || mU_{vj} \geq T_{high}$ **then**
        $vmOverLoaded \leftarrow getVmOverLoaded$;
        $vmToMigrate \leftarrow getVmToMigrate(hostOverLoaded)$;
        $migrationMap \leftarrow$
         $getNewVmPlacement(vmsToMigrate)$;
        return $migrationMap$;
      else
        vm++;
      end
    end
  else if $T_{high} \leq cU_{Hi} \geq T_{low} || T_{high} \leq mU_{Hi} \geq T_{low}$ **then**
    $hostMediumLoaded \leftarrow addToHostMediumLoaded$;
  else
    $hostUnderLoaded \leftarrow getHostUnderLoaded$;
    for $k \leftarrow 1$ **to** n **do**
      $cU_{vk} \leftarrow getcU_{vk}(vm)$ and $mU_{vk} \leftarrow getmU_{vk}(vm)$;
      $vmToMigrate \leftarrow getVmToMigrate(hostUnderLoaded)$;
      $migrationMap \leftarrow getNewVmPlacement(vmsToMigrate)$;
      return $migrationMap$;
    end
  end
end

Algorithm 1: Energy efficiency heuristic using vm consolidation

we check each host from the list of m number of host machines, and then select a VM for each host that exceeds the threshold of n number of VMs. Thus, the other load steps consume the constant time; hence, the total complexity of the proposed approach is $O(nlogn) + O(m \times n)$, which can be written as $O(m \times n)$, where m is the number of hosts and n is the number of VMs. If $m = n$, then the time complexity is $O(n^2)$.

5. Experimental results and analysis

This paper's approaches belong to heuristics; A heuristic technique is an approach to problem-solving that uses a practical method or various shortcuts to produce solutions. State-of-the-art (SOTA) Deep Neural Networks (DNNs) are the best patterns you can use for any specific task. A DNN can recognize SOTA based on its speed, precision, or interest metric. It is costly to train due to complex data models. Furthermore, deep learning requires expensive GPUs and hundreds of machines. It expands the users' cost as the number of VMs migration rises [46]. The proposed approach EEHVMC minimizes the number of VMs migration, so the SLAV and cost decrease. These approaches are part of the cloud simulator: DVFS, IQR_MC, IQR_MMT, MAD_MC, and MAD_MMT. The common thing in those approaches is that they only use CPU utilization and neglect other parts of power consumption. The proposed heuristic approach uses a cloud simulator to minimize power consumption. Hosts, MIPS, Cores, RAM, and other aspects of the parameter are the same as cloud simulator, and then we compare them with related techniques. The Energy Efficiency Heuristic Virtual Machine Consolidation (EEHVMC) testing results were compared to the other VM consolidation strategies such as *Energy optimize algorithms & DVFS* [13], *IQR_MC* [4], *IQR_MMT* [4], *MAD_MC* [6], *MAD_MMT* [6], and *SABFD* [21].

Table 4
Configuration of the simulation environment.

Simulator/version	CloudSim version 3.0.2
Datasets	Synthetic – I [4], GoCJ [47]
Energy optimize algorithms	DVFS [13], IQR_MC [4], IQR_MMT [4], MAD_MC [6], MAD_MMT [6], SABFD [21]
Performance parameters	Energy consumption, VMs migration, PDM, Average SLA, Execution time
Total cloud host machines	800
Total Virtual Machines	800 heterogeneous VMs
Total simulation limit	4800 s

Table 5
Workload characteristics.

Date	Number of virtual machines
03/March/2011	1052
06/March/2011	898
09/March/2011	1061
22/March/2011	1516
25/March/2011	1078
03/April/2011	1463
09/April/2011	1358
11/April/2011	1233
12/April/2011	1054
20/April/2011	1033

5.1. Experimental setup

CloudSim Toolkit [42], a novel simulation framework, was selected as the simulation platform for the CC environment. In addition, using CloudSim offers two advantages: It supports on-demand resource provisioning and management, as well as virtual environment modeling and energy-aware simulation, including the ability to simulate service applications with dynamic workloads [48,49]. Table 4 shows the setup details for the simulation environment used. The experiments are conducted with 800 VMs hosted on 800 host machines within a cloud better to understand the concept of VMM and energy efficiency.

5.1.1. Realistic dataset based on PlanetLab

The PlanetLab dataset [4] enables the behavior modeling of Cloud system components, including VMs, data centers, and resource provisioning policies.

1. **Hosts Features:** The extensions used in the current study were built using the CloudSim toolkit (version 3.0.3). The cloud consisted of heterogeneous hosts, some HP ProLiant G4 hosts, and the others HP ProLiant G5 hosts. HP ProLiant G4 hosts consist of 1860 MIPS, 2 CPU cores, and 4 GB of RAM, whereas HP ProLiant G5 hosts consist of 2860 MIPS, 2 CPU cores, and 4 Gb of RAM based on [4].
2. **Virtual Machines Features:** The functionalities of the VMs are based on Amazon EC2 instance models [50]. CPU high, large, small, and micro instances are the four categories of VMs used. CPU high instance consists of 2500 MIPS and 870 GB of RAM, whereas CPU large instance consists of 2000 MIPS and 1740 GB of RAM. Similarly, CPU small instance consists of 1000 MIPS and 1740 GB of RAM, whereas the CPU micro instance consists of 500 MIPS and 613 GB of RAM based on [4].
3. **Workload Characteristics:** The experiments were carefully conducted using an existing system's workload traces to produce more realistic results. Data for the tests came from Planet Lab's CoMon project [51]. In addition, thousands of VMs from servers at over 500 sites worldwide use data on CPU consumption. Table 5 lists the characteristics of the dataset in detail [4].

Table 6
Google cloud jobs dataset.

Type of jobs	MI	% of Jobs
Type of jobs	MI	% of Jobs
Small	15k–55k	20
Medium	59k–99k	40
Large	101k–135k	30
Extra-large	150k–337.5k	4
Huge	525k–900k	6

5.1.2. Gocj dataset

The GoCJ dataset [47] contains a variety of jobs. Table 6 shows the different categories of tasks in the GoCJ dataset i.e., *small*, *medium*, *big*, *extra-large*, and *huge*. It also shows the characteristics and specifications of the host and VM that run in the GoCJ dataset.

5.1.3. Benchmark heuristics

The following is an overview of the other prominent approaches used for experimental evaluation.

- **DVFS** [13]: *Dynamic Voltage Frequency Scaling* is a method to reduce power consumption by automatically changing the frequency and voltage;
- **IQR_MC** [4]: *InterQuartile Range* is utilized to detect overloading on the host, and the *Maximum Correlation* policy is utilized for migration;
- **IQR_MMT** [4]: *InterQuartile Range* is utilized to detect overloading on the host, and the *Minimum Migration Time* policy is utilized for migration;
- **MAD_MC** [6]: *Median Absolute Deviation* is utilized to indicate overload on the host, and the *Maximum Correlation* policy is utilized for migration [44].
- **MAD_MMT** [6]: *Median Absolute Deviation* is used to identify overloading on the host, and the *Minimum Migration Time* policy is utilized for migration purpose;
- **SABFD** [21]: This method selects a VM for migration that has the highest CPU usage and is placed in the host with the fewest MIPS.

5.1.4. Performance parameters

The following performance metrics used to evaluate the outcomes of the proposed approach:

- **Energy Consumption kWh**: Data centers are huge buildings consisting of many physical machines that store and retrieve data. A data center consumes over 91 billion kWh of electricity [5];
- **VM Migrations**: Transferring a Virtual Machine (VM) from one physical host to another. VM Manager keeps track of VMs in the cloud and their availability;
- **Performance Degradation caused by the Migration**: PDM refers to the general performance degradation that occurs in VMs as a result of live migrations;
- **SLA Violations**: The final SLAV simplify by lowering one of the parameters, PDM, or SLATAH;
- **Execution Time**: Execution time is when a task execution takes from start to finish.

5.2. Experimental results

The simulations compare DVFS [13], SABFD [21], and energy-aware strategies (e.g. IQR, MAD) [4,6]. The proposed research is compared with the most popular algorithms, such as IQR, MAD and VM selection techniques, MC [4] and MMT [6]. The proposed algorithm, EEHVMC, checks and calculates the host's threshold based on CPU and memory usage. Compared to the previous techniques, the proposed algorithm EEHVMC consumes the least amount of energy.

5.3. Realistic PlanetLab dataset

The characteristic of hosts and VMs are specified in host and virtual machines features part. Fig. 4 shows that EEHVMC (24.34 kWh) has the lowest energy usage, followed by DVFS (29.79 kWh), IQR_MC (27.06 kWh), IQR_MMT (27.29 kWh), MAD_MC (26.49 kWh), MAD_MMT (26.64 kWh), and SABFD (28.38 kWh). The percentage improvement of the proposed approach EEHVMC compared to the other approaches followed by DVFS (22.39%), IQR_MC (11.18%), IQR_MMT (12.12%), MAD_MC (8.83%), MAD_MMT (9.45%), and SABFD (16.50%).

The Dynamic Virtual Machine Consolidation (DVMC) method shows that as the number of VM migrations increases, so does the cost. Consequently, we need to reduce the number of VM migrations. DVFS approach automatically changes the voltage and frequency to reduce processor heat generation and lower power consumption. DVFS is a frequency-aware model capable of determining the best frequency ratio for reducing processor energy usage. DVFS approach does not include the VM migration process; therefore, as shown in Figs. 5 and 10, their result is 0, but the power consumption is too much compared to other approaches as shown in Figs. 4 and 9. As shown in Fig. 5, the EEHVMC strategy results in fewer migrations compared to the other related approaches, which are based on migration-based mechanisms. Regarding VM migration, our approach is better than IQR_MC by 13.13%, IQR_MMT by 9.67%, MAD_MC by 8.38%, MAD_MMT by 5.98%, and SABFD by 16.12%.

Performance Degradation caused by VMs migration (PDM) increases if the number of VMs migration rises. Therefore, we need to care about it that the migration will remain low, so PDM remains to decrease; the DVFS approach is not part of VM migration, so PDM remains 0, as shown in Figs. 6 and 11. The proposed EEHVMC approach reduces the number of live migrations while lowering PDM. Fig. 6 shows that EEHVMC has the least performance degradation (i.e., 0.14) compared with IQR_MC (0.18), IQR_MMT (0.17), MAD_ MC (0.17), MAD_MMT (0.16), and SABFD (0.19). Our approach as per PDM is up by 28.57% than IQR_MC, 21.43% than IQR_MMT, 21.43% than MAD_MC, 14.29% than MAD_MMT and 35.71% than SABFD.

Service Level Agreement Violations increase if one parameter, PDM or SLATAH, grows. Compared to other methods, DVFS does not require any VMs migration, so as a result,

PDM remains 0, so it does not affect any SLAV process. As shown in Figs. 7 and 12, the development of DVFS is 0 compared to the other approaches. Fig. 7 illustrates that EEHVMC reduces SLA violations when compared to other methods. The figure clearly shows that EEHVMC has the lowest SLA violations (i.e., 9.01%) compared with IQR_MC (10.23%), IQR_MMT (10.12%), MAD_MC (10.1%), MAD_MMT (10%), and SABFD (10.89%).

DVFS takes less time than other approaches, which is why VM migration executes early. But the difference is it takes more power consumption, as shown in Figs. 4 and 9. Fig. 8 shows that the proposed approach EEHVMC requires less execution time compared to other approaches. As per Execution time, our approach is better by 5.284% than IQR_MC, 5.186% than IQR_MMT, 4.228% than MAD_MC, 2.700% than MAD_MMT, and 6.948% than SABFD.

5.4. GoCJ dataset

The GoCJ dataset comprises different task sizes such as *small*, *medium*, *large*, *extra large*, and *huge* generated using dynamic task length thresholds. The GoCJ task types [47] are listed in Table 6.

The EEHVMC uses the least amount of power (i.e., 16.23 kWh) followed by DVFS (20.75 kWh), IQR_MC (18.9 kWh), IQR_MMT (18.79 kWh), MAD_MC (17.58 kWh), MAD_MMT (17.23 kWh), and SABFD (19.23 kWh), as shown in Fig. 9. With regards to energy consumption, our approach is higher than DVFS by 27.85%, IQR_MC by 16.45%, IQR_MMT by 15.77%, MAD_MC by 8.32%, MAD_MMT by 6.16%, and SABFD by 18.48%.

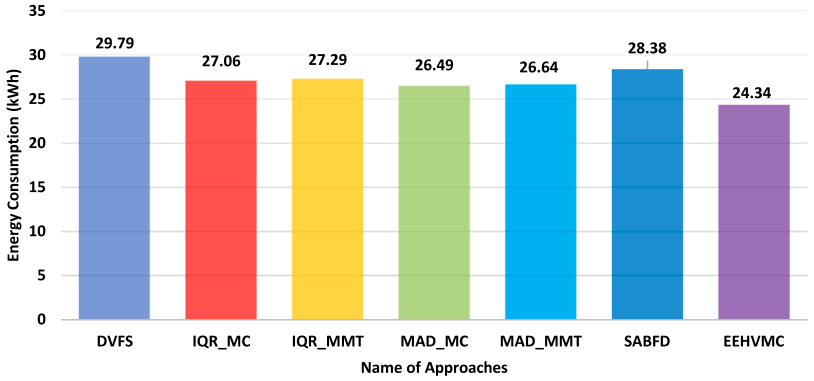

Fig. 4. Energy consumption — synthetic dataset.

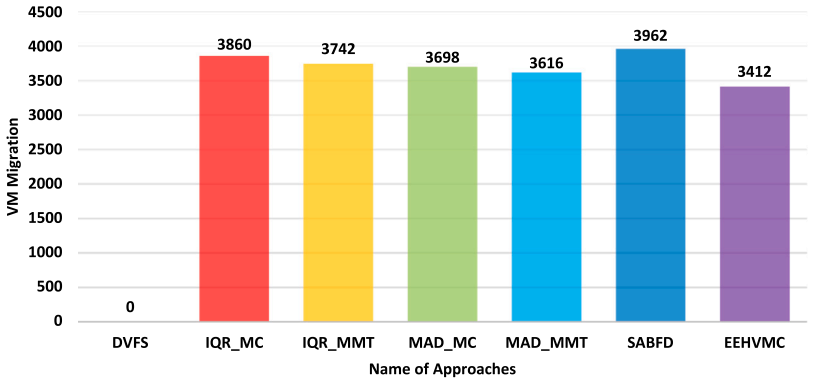

Fig. 5. Virtual machine migrations — synthetic dataset.

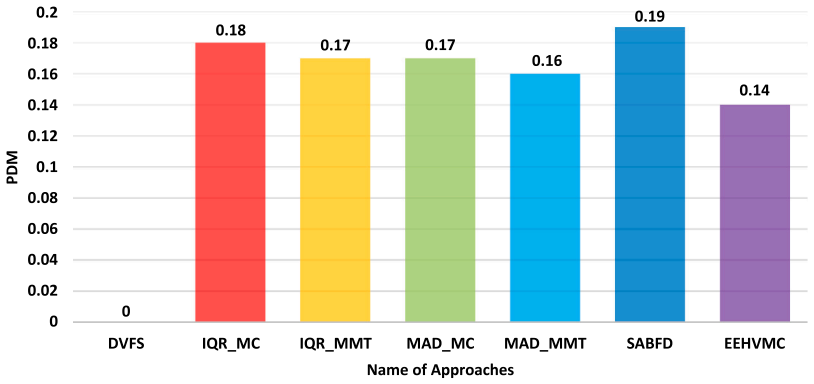

Fig. 6. Performance degradation — synthetic dataset.

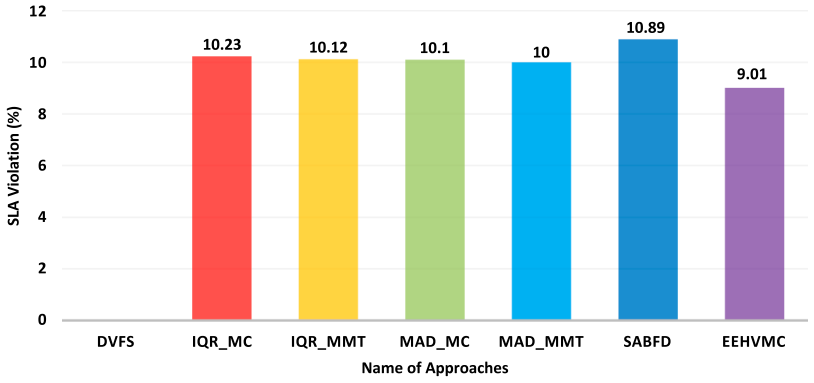

Fig. 7. Service level agreement violations — synthetic dataset.

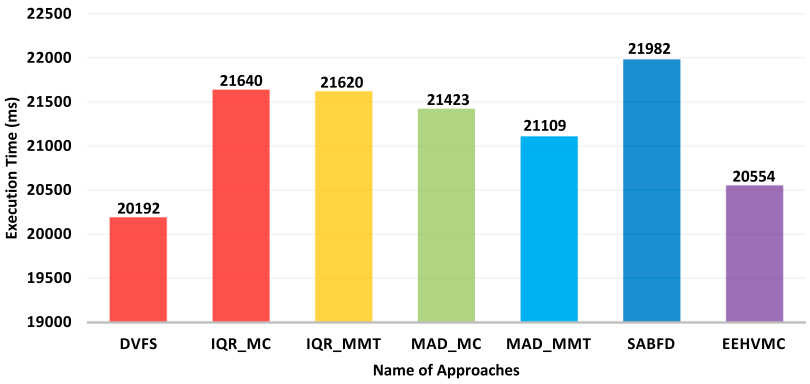

Fig. 8. Execution time — synthetic dataset.

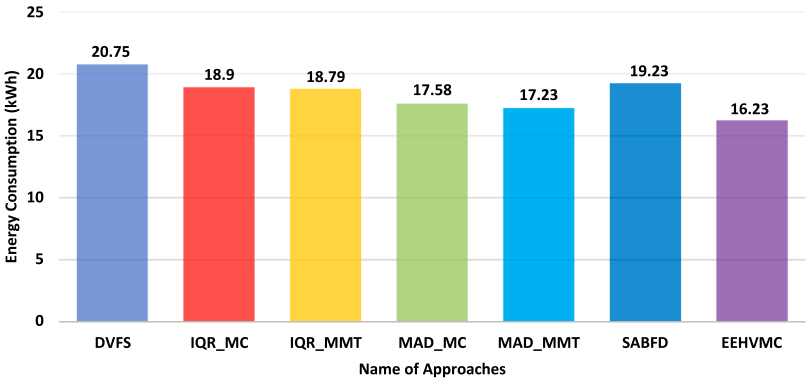

Fig. 9. Energy consumption — GoCJ dataset.

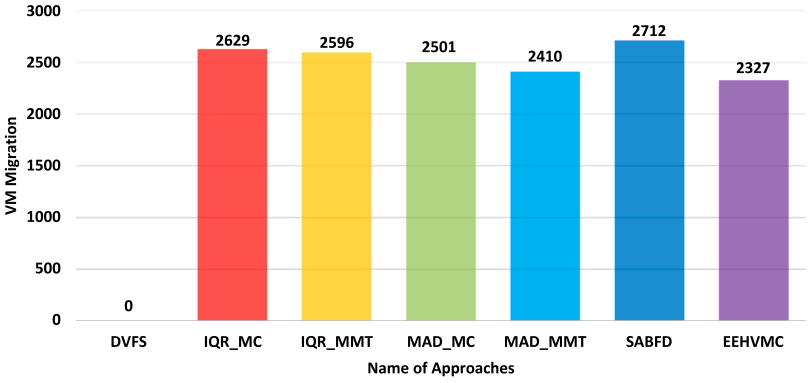

Fig. 10. Virtual machine migrations — GoCJ dataset.

The DVMC technique shows that as the number of VM migrations increases, the cost also increases. As shown in Figs. 5 and 10, DVFS approach does not include the VM migration process; therefore, their result is 0. EEHVMC requires fewer VM migrations than the other approaches, as shown in Fig. 10.

In respect of VM migration improvement percentage of our approach is 100% by DVFS, 12.98% by IQR_MC, 11.56% by IQR_MMT, 7.48% by MAD_MC, 3.57% by MAD_MMT, and 16.54% by SABFD.

PDM will increase if the quantity of VMs migration rises. Therefore, we want to care approximately that the migration will continue to be low, so PDM stays decrease; the DVFS technique is not always a part of VM migration, so PDM remains 0, as proven in Figs. 6 and 11. Performance degradation is mitigated by EEHVMC reducing the number of live migrations.

EEHVMC suffers the least performance degradation (i.e., 0.15) compared to IQR_MC (0.18), IQR_MMT (0.17), MAD_ MC (0.16 kWh), MAD_MMT (0.16), and SABFD (0.19), as shown in Fig. 11. In connection with PDM our approach is better than 20% by IQR_MC, 13.33% by IQR_MMT, 6.67% by MAD_MC, 6.67% by MAD_MMT and 26.67% by SABFD.

SLAV remains 0 in DVFS because there is no involvement of Performance caused by VM Migration (PDM), as shown in Figs. 7 and 12. Fig. 12 shows that the energy efficiency heuristic using virtual machine consolidation (10.2%) has the lowest SLA violations compared to IQR_MC (10.9%), IQR_MMT (10.8%), MAD_ MC (10.6%), MAD_MMT (10.6%), and SABFD (10.89%).

Compared to other methods, DVFS takes less time to execute. This is because DVFS does not provide for migrations, which results in better execution performance, but also consumes more energy.

EEHVMC takes less time to execute compared to the other methods. The reason is that less PDM and VMM are required, so it runs faster. As shown in Fig. 13, the proposed approach takes less time to execute than

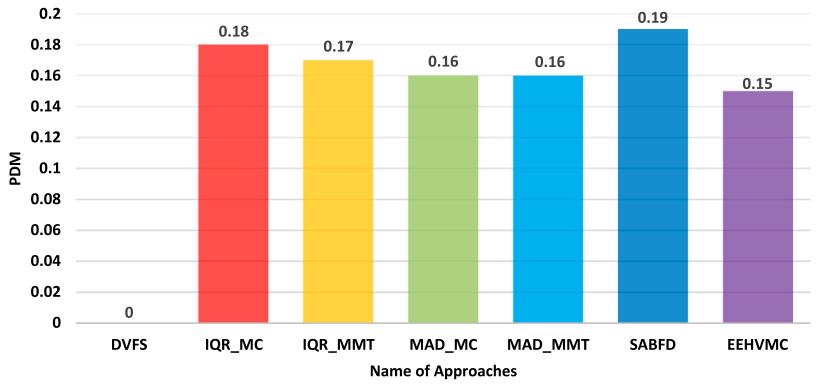

Fig. 11. Performance degradation — GoCJ dataset.

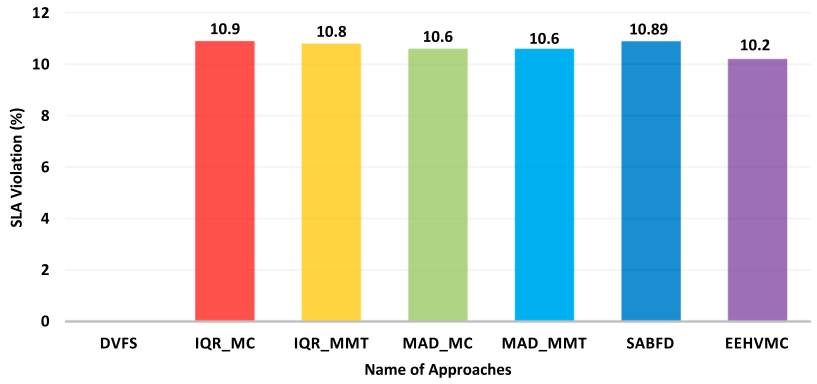

Fig. 12. Service level agreement violations — GoCJ dataset.

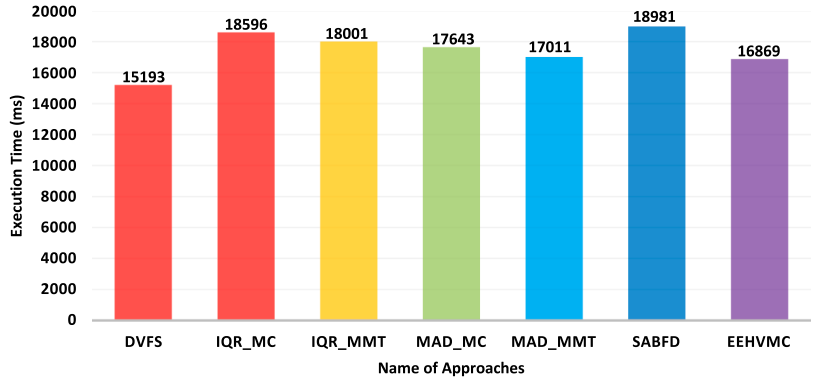

Fig. 13. Comparison of execution time.

alternative strategies. Regarding execution time, our approach is up by 9.94% than DVFS, 10.24% than IQR_MC, 6.71% than IQR_MMT, 4.59% than MAD_MC, 0.84% than MAD_MMT, and 12.52% than SABFD.

5.5. Result and discussion

The fundamental concept behind the proposed technique "EEHVMC" is to classify cloud hosts based on CPU and memory usage. The host classifies into three main classes based on the two thresholds of CPU and memory usage: HOL, HML, and HUL. EEHVMC has the lowest energy consumption compared to DVFS, IQR_MC, IQR_MMT, MAD_MC, MAD_MMT, and SABFD, as shown in Figs. 4 and 9. According to the DVMC, the VM migration cost increases with the number of VM migrations. Therefore, the technique that requires fewer VM migrations leads to better computational performance. The results presented in the previous section (e.g., Figs. 5 and 10) show that the proposed EEHVMC technique requires fewer VM migrations and saves more energy compared to the related approaches.

If the number of Virtual Machine Migration (VMM) increases, Performance Degradation caused by VM migration (PDM) will increase. The proposed technique has the most negligible performance degradation compared to other approaches, as shown in Figs. 6 and 11. The final SLAV simplifies [4] by lowering one of the parameters, PDM or SLATAH.

$$SLAV = SLATAH \times PDM \tag{10}$$

Moreover, the proposed approach reduces the frequency of migrations and PDM, resulting in a low SLAV. EEHVMC has the lowest SLAV compared to the other techniques, as shown in Figs. 7 and 12. Compared to the other methods, DVFS takes less time to execute. The reason is that there is no VMM or PDM, so it runs faster than the different approaches (mentioned in the previous section). Figs. 8

and 13 show that the proposed approach takes less time to execute than the alternative techniques such as IQR_MC, IQR_MMT, MAD_ MC, MAD_MMT, and SABFD.

6. Conclusions

People and businesses worldwide use cloud computing to manage and store data over the Internet. As cloud computing data centers have become more prevalent, the power consumption of the host and other infrastructures has increased. There is a need to reduce power consumption without compromising the Quality of Service. This paper presents the Energy Efficiency Heuristic with Virtual Machine Consolidation (EEHVMC), which reduces power consumption while reducing SLA violations. The host classifies into three main categories based on the two thresholds: *Host Over-Loaded*, *Host Medium-Loaded*, and *Host Under-Loaded*. Over-loaded hosts consume more energy than other hosts in the data center, so specific virtual machines must move from over-loaded to medium-loaded hosts. All VMs that move from under-loaded to medium-loaded and empty hosts are put into power-saving mode to reduce the number of active hosts. Compared to state-of-the-art, the EEHVMC process minimizes power consumption and SLA violations. CPU and memory are the hosts' components used to consume power. Still, other parts are used to consume energy, like network bandwidth, storage, cooling overhead, and power overhead. We will minimize power consumption by considering these parts' network bandwidth, GPU, storage, cooling overhead, and power overhead in future work. The proposed approach only finds CPU and memory-intensive tasks in the virtual machine. There are other tasks as well which are part of virtual machines, like I/O intensive tasks. Suppose the I/O intensive tasks consume more power than the other parts, mainly CPU and memory. In that case, it will not consider this task. We plan to consider I/O intensive tasks to reduce power consumption at the data center level. Hadoop and Spark are models that use in CC. EEHVMC does not follow any of these models. In the future, we can use this approach in Hadoop or the Spark model.

CRediT authorship contribution statement

Umer Arshad: Conceptualization, Methodology, Writing – original draft. **Muhammad Aleem:** Writing – review & editing, Supervision. **Gautam Srivastava:** Investigation. **Jerry Chun-Wei Lin:** Formal analysis, Writing – review & editing.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Jerry Chun-Wei Lin reports a relationship with Western Norway University of Applied Sciences that includes: employment.

Data availability

Data will be made available on request.

Acknowledgments

This paper is partially supported by the Western Norway University of Applied Sciences, Bergen, Norway.

References

[1] Lin JCW, Djenouri Y, Srivastava G, Li Y, Yu PS. Scalable mining of high-utility sequential patterns with three-tier MapReduce model. ACM Trans Knowl Discov Data 2021;16(3):1–26.
[2] Wu JMT, Srivastava G, Wei M, Yun U, Lin JCW. Fuzzy high-utility pattern mining in parallel and distributed hadoop framework. Inform Sci 2021;553:31–48.
[3] Beloglazov A, Buyya R. Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in cloud data centers. Concurr Comput: Pract Exper 2012;24:1397–420;

Lin JCW, Djenouri Y, Srivastava G. Efficient closed high-utility pattern fusion model in large-scale databases. Inf Fusion 2021;76:122–32.
[4] Beloglazov A, Buyya R. Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in cloud data centers. Concurr Comput: Pract Exper 2012;24:1397–420.
[5] Andrae AS, Edler T. On global electricity usage of communication technology: trends to 2030. Challenges 2015;6:117–57.
[6] Fard SYZ, Ahmadi MR, Adabi S. A dynamic VM consolidation technique for QoS and energy consumption in cloud environment. J Supercomput 2017;73:4347–68.
[7] Panda SK, Jana PK. An energy-efficient task scheduling algorithm for heterogeneous cloud computing systems. Cluster Comput 2019;22:509–27.
[8] Mhouti AEl, Erradi M, Nasseh A. Using cloud computing services in e-learning process: Benefits and challenges. Educ Inf Technol 2018;23:893–909.
[9] Choudhary A, Rana S, Matahai KJ. A critical analysis of energy efficient virtual machine placement techniques and its heuristic in a cloud computing environment. Procedia Comput Sci 2016;78:132–8.
[10] Masdari M, Nabavi SS, Ahmadi V. An overview of virtual machine placement schemes in cloud computing. J Netw Comput Appl 2016;66:106–27.
[11] Hussain A, Aleem M, Khan A, Iqbal MA, Islam MA. RALBA: a computation-aware load balancing scheduler for cloud computing. Cluster Comput 2018;21:1667–80.
[12] Zhou Z, Abawajy J, Chowdhury M, Hu Z, Li K, Cheng H, Alelaiwi AA, Li F. Minimizing SLA violation and power consumption in cloud data centers using adaptive energy-aware algorithms. Future Gener Comput Syst 2018;86:836–50.
[13] Stavrinides GL, Karatza HD. An energy-efficient, QoS-aware and cost-effective scheduling approach for real-time workflow applications in cloud computing systems utilizing DVFS and approximate computations. Future Gener Comput Syst 2019;96:216–26.
[14] Khalid YN, Aleem M, Ahmed U, Islam MA, Iqbal MA. Troodon: A machine-learning based load-balancing application scheduler for CPU–GPU system. J Parallel Distrib Comput 2019;132:79–94.
[15] Zhou Z, Yu J, Li F, Yang F. Virtual machine migration algorithm for energy efficiency heuristic in cloud computing. Concurr Comput: Pract Exper 2018;30:e4942.
[16] Bui DM, Yoon Y, Huh EN, Jun S, Lee S. Energy efficiency for cloud computing system based on predictive heuristic. J Parallel Distrib Comput 2017;102:103–14.
[17] Barroso LA, Hölzle U, Ranganathan P. The datacenter as a computer: Designing warehouse-scale machines. Synth Lect Comput Archit 2018;13:i–189.
[18] Mezmaz M, Melab N, Kessaci Y, Lee YC, Talbi EG, Zomaya AY, Tuyttens D. A parallel bi-objective hybrid metaheuristic for energy-aware scheduling for cloud computing systems. J Parallel Distrib Comput 2011;71:1497–508.
[19] Cheng C, Li J, Wang Y. An energy-saving task scheduling strategy based on vacation queuing theory in cloud computing. Tsinghua Sci Technol 2015;20:28–39.
[20] Hussain A, Aleem M, Islam MA, Iqbal MA. A rigorous evaluation of state-of-the-art scheduling algorithms for cloud computing. IEEE Access 2018;6:75033–47.
[21] Wang H, Tianfield H. Energy-aware dynamic virtual machine consolidation for cloud datacenters. IEEE Access 2018;6:15259–73.
[22] Ilager S, Ramamohanarao K, Buyya R. ETAS: Energy and thermal-aware dynamic virtual machine consolidation in cloud data center with proactive hotspot mitigation. Concurr Comput: Pract Exper 2019;31:e5221.
[23] Lin W, Liang C, Wang JZ, Buyya R. Bandwidth-aware divisible task scheduling for cloud computing. Softw - Pract Exp 2014;44:163–74.
[24] Gawali MB, Shinde SK. Task scheduling and resource allocation in cloud computing using a heuristic approach. J Cloud Comput 2018;7:1–16.
[25] Mohamadi Bahram Abadi R, Rahmani AM, Alizadeh SH. Server consolidation techniques in virtualized data centers of cloud environments: A systematic literature review. Softw - Pract Exp 2018;48:1688–726.
[26] Shukla K, Nefti-Meziani S, Davis S. A heuristic approach on predictive maintenance techniques: Limitations and scope. Adv Mech Eng 2022;14:6.
[27] Sayadnavard MH, Haghighat AT, Rahmani. A reliable energy-aware approach for dynamic virtual machine consolidation in cloud data centers. J Supercomput 2019;75:2126–47.
[28] Buyya R, Beloglazov A, Abawajy J. Energy-efficient management of data center resources for cloud computing: a vision, architectural elements, and open challenges. In: Parallel and distributed processing techniques and applications, Vol. 106. 2010, p. 116–24.

[29] Liu Y, Sun X, Wei W, Jing W. Enhancing energy-efficient and QoS dynamic virtual machine consolidation method in cloud environment. IEEE Access 2018;6:31224–35.
[30] Uddin M, Hamdi M, Alghamdi A, Alrizq M, Memon MS, Abdelhaq M, Alsaqour R. Server consolidation: A technique to enhance cloud data center power efficiency and overall cost of ownership. Int J Distrib Sens Netw 2021;17:1550147721997218.
[31] Khan MA. An efficient energy-aware approach for dynamic VM consolidation on cloud platforms. Cluster Comput 2021;21:1–18.
[32] Yadav R, Zhang W, Li K, Liu C, Laghari AA. Managing overloaded hosts for energy-efficiency in cloud data centers. Cluster Comput 2021;33:1–15.
[33] Mao J, Peng X, Cao T, Bhattacharya T, Qin X. A frequency-aware management strategy for virtual machines in DVFS-enabled clouds. Sustain Comput: Inform Syst 2022;33:100643.
[34] Kanagasubaraja S, Hema M, Valarmathi K, Kumar N, Kumar BPM, Balaji N. Energy optimization algorithm to reduce power consumption in cloud data center. In: International conference on advances in computing, communication and applied informatics: informatics and systems, Vol. 66. 2022, p. 1–8.
[35] Zolfaghari R, Rahmani AM. Virtual machine consolidation in cloud computing systems: Challenges and future trends. Wirel Pers Commun 2020;115:2289–326.
[36] Gu C, Huang H, Jia X. Power metering for virtual machine in cloud computing-challenges and opportunities. IEEE Access 2014;2:1106–16.
[37] Gandhi A, Harchol-Balter M, Das R, Lefurgy C. Optimal power allocation in server farms. ACM SIGMETRICS Perform Eval Rev 2009;37:157–68.
[38] Kusic D, Kephart JO, Hanson JE, Kandasamy N, Jiang G. Power and performance management of virtualized computing environments via lookahead control. Cluster Comput 2009;11:1–15.
[39] Raghavendra R, Ranganathan P, Talwar V, Wang Z, Zhu X. No power struggles: coordinated multi-level power management for the data center. In: Proceedings of the 13th international conference on Architectural support for programming languages and operating systems, Vol. 9. 2008, p. 48–59.
[40] Verma A, Ahuja P, Neogi A. pMapper: power and migration cost aware application placement in virtualized systems. In: ACM/IFIP/USENIX international conference on distributed systems platforms and open distributed processing, Vol. 18. 2008, p. 243–64.
[41] Piraghaj SF, Dastjerdi AV, Calheiros RN, Buyya R. ContainerCloudSim: An environment for modeling and simulation of containers in cloud data centers. Softw - Pract Exp 2017;47:505–21.
[42] Calheiros RN, Ranjan R, Beloglazov A, De Rose CA, Buyya R. CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. Softw - Pract Exp 2011;41:23–50.
[43] Hussain A, Aleem M, Iqbal MA, Islam MA. Investigation of cloud scheduling algorithms for resource utilization using cloudsim. Comput Inform 2019;38:525–54.
[44] Alsbatin L, Öz G, Ulusoy AH. A novel physical machine overload detection algorithm combined with quiescing for dynamic virtual machine consolidation in cloud data centers. Int Arab J Inf Technol 2020;17:358–66.
[45] Khan AA, Zakarya M, Khan R. Energy-aware dynamic resource management in elastic cloud datacenters. Simul Model Pract Theory 2019;92:82–99.
[46] Saxena D, Singh AK. OFP-TM: an online VM failure prediction and tolerance model towards high availability of cloud computing environments. J Supercomput 2022;78:8003–24.
[47] Hussain A, Aleem M. GoCJ: Google cloud jobs dataset for distributed and cloud computing infrastructures. Data 2018;3:38.
[48] Ibrahim M, Iqbal MA, Aleem M, Islam MA. SIM-cumulus: An academic cloud for the provisioning of network-simulation-as-a-service (NSaaS). IEEE Access 2018;6:27313–23.
[49] Zolfaghari R, Sahafi A, Rahmani AM, Rezaei R. An energy-aware virtual machines consolidation method for cloud computing: Simulation and verification. Softw - Pract Exp 2021;12:157–62.
[50] Iqbal MA, Aleem M, Ibrahim M, Anwar S, Islam MA. Amazon cloud computing platform EC2 and VANET simulations. Int J Ad Hoc Ubiquitous Comput 2019;30:127–36.
[51] Shirvani MH, Rahmani AM, Sahafi A. A survey study on virtual machine migration and server consolidation techniques in DVFS-enabled cloud datacenter: taxonomy and challenges. J King Saud Univ-Comput Inf Sci 2020;32:267–86.