Simulation Modelling Practice and Theory 104 (2020) 102127

Contents lists available at ScienceDirect

Simulation Modelling Practice and Theory

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

A novel energy-aware resource management technique using joint VM and container consolidation approach for green computing in cloud data centers

Niloofar Gholipour[a], Ehsan Arianyan[b,*], Rajkumar Buyya[c]

[a] Department of Computer Engineering, Science and Research Branch, Islamic Azad University, Tehran, Iran
[b] ICT Research Institute, Tehran, Iran
[c] Cloud Computing and Distributed Systems (CLOUDS) Laboratory, School of Computing and Information Systems, The University of Melbourne, Australia

ARTICLE INFO

Keywords:
Cloud computing
Consolidation
Containerization
Datacenter
Energy consumption
Resource management

ABSTRACT

Cloud computing is being rapidly adopted for managing IT services as a notable solution due to diverse beneficiaries such as automatically optimized resource management as well as modern service delivery models. The container as a service has been recently introduced by cloud providers as a new service apart from traditional cloud services. Containers enable applications to run and deploy on isolated virtual space, and the operating system kernel is shared among them. Also, containerization has some attributes such as scalability, highly portable properties, and lightweight, for those reasons, it is applied for running isolated applications. Reducing energy consumption, as well as their CO_2 emissions, are great deals for cloud providers. In this direction, consolidation is recommended as a vital energy-aware approach in cloud data centers. Previously, independent virtual machine migration or container migration was proposed in the literature for green computing in cloud data centers. However, this paper proposes a new cloud resource management procedure based on a multi-criteria decision-making method that takes advantage of a joint virtual machine and container migration approach concurrently. The results of simulations using ContainerCloudsim simulator validates the applicability of the proposed approach which shows notable reductions in energy consumption, SLA violation, and number of migrations in comparison with the state-of-the-art algorithms.

1. Introduction

Cloud computing is an Internet-based service delivery model for providing on-demand access to data, computation, and applications as utility services from anywhere. In this direction, one of the principal cloud computing's characteristics is elasticity, which is used in response to the ongoing variation in the customer's requests. On the other hand, the rapid growth of data needs more massive storage, and this reason leads to creating larger data centers in both quantity and scale to satisfy customer's requests. Consequently, energy consumption is increasing for cooling and operation. Furthermore, this will raise concern for governments and owners of the cloud [1]. Nowadays there are more than four types of cloud service models and the traditional ones are Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS) [2]. In the last years, the rise of cloud computing has spawned many of "X as a service" cloud computing services, one of which is a Container as a Service (CaaS) [3]. Users can easily access these

* Corresponding author.
E-mail address: ehsan_arianyan@itrc.ac.ir (E. Arianyan).

https://doi.org/10.1016/j.simpat.2020.102127
Received 18 March 2020; Received in revised form 22 May 2020; Accepted 26 May 2020
Available online 04 June 2020
1569-190X/

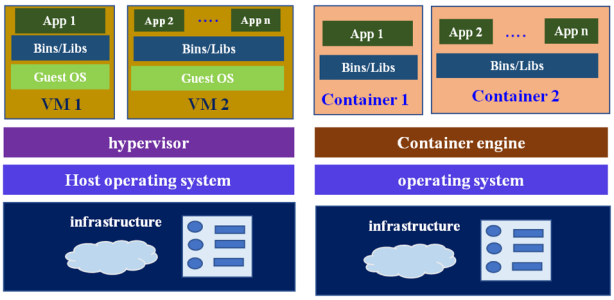

Fig. 1. traditional hypervisor architecture on the left and a container-based architecture on the right [6]

services on a pay-as-you-use basis without any geographical restrictions [4].

PaaS supplies the platform for the users to develop their application without any concern about the technologies and required underlying infrastructure. Along with the advantage of the PaaS model, there are some drawbacks, which limits the utilization. The PaaS model makes it possible for customers to concentrate only on their code without the need to be worry about the runtime environment, operating system, and maintenance costs. On the other hand, the platform's characteristics limit the development of applications in PaaS environments. For instance, in order to run Java applications on Google Application Engine (GAE), developers must make sure that their utilized third-party libraries are compatible with GAE. Hence, CaaS is proposed to solve these issues and limitations in the PaaS service model [3]. The axiom of CaaS is that permits processes and their resources to be isolated from the rest of the system without requiring any hardware [5]. Containers, as the building blocks of the CaaS model, provide an isolated virtual environment for running applications.

The host operating system kernel is shared among containers which interact with each other via system standard calls [3]. As containers can be utilized denser in comparison with virtual machines, they increase the need for efficient cloud resource utilization. Containerization, which is a lightweight solution for deployment and management of packaged portable and interoperable applications that provide the capability to develop, test, and deploy applications on container hosts to execute a large number of servers. Moreover, containerization supplies the ability to interconnect containers. Another significant advantage of containerization is simplifying the step from a single host to clusters of containerized applications over multiple clusters in multiple clouds [6]. Fig. 1 shows the differences between the traditional hypervisor architecture and container-based architecture. In traditional architecture, each Virtual Machine (VM) executes on virtual hardware and a kernel. However, in container-based architecture, the same OS and kernel are shared among all containers [6]. In container-based architecture, there is no need for monitoring media such as hypervisor that exists in the traditional one. The resources are more utilized by the shared kernel, which in return reduces the overhead of container's startups and shutdowns. The relations between containers are performed via standard system calls, which are much faster than hypervisor-based communications [3]. It is important to note that the architecture of the two top right containers are different to show that it is probable that either one application or more than one applications running in a container. Consolidation is an important feature that is achievable thanks to the capabilities provided by virtualization technology including live migration of VMs and containers [7, 8]. In the server consolidation method, several virtual machines and containers are packed in the minimum number of physical machines in order to turn off or switch the status of the idle hosts to sleep mode to minimize the energy consumption [9, 10]. One concern in consolidation is tuning the trade-off between energy efficiency and quality of service (QoS). So, it is essential to have a consolidation algorithm that balances between Service Level Agreements (SLA) and energy efficiency [11]. There are two types of consolidation, including static and dynamic consolidation [12]. In static consolidation, during the allocation of VMs to Physical Machines (PMs) for processing, no migration is executed. While, in dynamic consolidation, the VMs are dynamically migrated from one PM to another based on the current resource utilization. This process will continue until an energy-efficient placement with the minimum number of active PMs is found [12].

Previous studies have concentrated on VM consolidation and container consolidation separately. Also, the authors in [3] verified that container consolidation is more energy-efficient than VM consolidation. However, we focus on joint VM and container consolidation in this paper and show that this solution is more energy-efficient than the separate consolidation of either VMs or containers. The novelties of this paper are as follows:

- Ø Proposing the notion of the joint container and VM consolidation solution.
- Ø Proposing a new architecture and flowchart consisted of seven sub-problems to handle the joint VM and container migration problem.
- Ø Proposing the multi Criteria migration decision (JVCMMD) policy to decide whether the VMs or the containers should be migrated that simultaneously optimizes energy consumption, number of VM and container migrations and SLA violations.

The rest of the paper is organized as follows: related works are reviewed in section 2. Section 3 presents system models, including the data center model and the metrics, which are used to evaluate the efficiency of the proposed policies. Section 4 presents our proposed solution for joint VM and container resource management in cloud data centers. Section 5 assesses the applicability of our proposed solutions using the ContainerCloudsim simulator. Finally, conclusion and future works are presented in section 6.

2. Related Work

Energy-efficient resource management in cloud environments is a hot topic and is vastly addressed by researchers. This section summarizes some of the previous works in the literature, which are especially focused on consolidation and migration techniques.

The authors in [13] have proposed optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation in Cloud data centers. Furthermore, they have investigated a novel adaptive heuristic for dynamic VM consolidation based on an analysis of historical data from the VM's resource usage. Also, they validated the efficiency of their proposed algorithms by simulating them in the CloudSim simulator and showed that their proposed solutions reduce energy consumption significantly and meet the SLA, simultaneously. They have divided the problem of dynamic VM consolidation into four parts: (1) determining when a host is considered as being overloaded; (2) determining when a host is considered as being underloaded; (3) VM selection; and (4) finding the new placement of the VMs selected for migration. According to their experiment results, their proposed local regression-based algorithm for the determination of overloaded hosts combined with the minimum migration time algorithm for VM selection outperforms the other dynamic VM consolidation algorithms.

The authors in [1] have explored a novel fuzzy multi Criteria and objective resource management solution that takes advantage of dynamic voltage and frequency scaling methods as well as consolidation approach. In this paper, they have considered decreasing power consumption and performance loss as their goals for the VM migration problem. Additionally, they evaluated their proposed algorithms by the CloudSim simulator. They verified that they had a significant reduction in energy consumption, SLA violation, and a number of migrations in comparison with state of the art. Moreover, they considered the essential Criteria for counting CPU, RAM, and network bandwidth in all their proposed algorithms. Besides, they have considered four consolidation subproblems including 1 & 2) determining over-load and under-load hosts, 3) VM selection to migrate, and 4) VM placement. They have proposed power and SLA Fuzzy Weighted TOPSIS (PSFWT) allocation policy in their scenario for VM placement considering eight criteria in the decision process. Moreover, they proposed the DVFS aware consolidation optimization (DCO) policy for resource management procedure as well as utility-based DVFS governor (UDG), which modifies the voltage and frequency of processors according to the utilization of servers.

The authors in [3] have investigated the energy efficiency of servers and proposed a new framework that consolidates containers on VMs. Moreover, they presented a container consolidation problem and evaluated their algorithms regarding performance metrics, including energy consumption, SLA violation, average container migration rate, and an average number of created VMs. Furthermore, they have modelled the CaaS environment and the power optimization problem in container CloudSim simulator. They considered three subproblems for consolidation: 1) detecting over-loaded and under-loaded hosts, 2) container selection for migrating, 3) container placement. Their experiment results show that the combination of their proposed correlation-aware placement algorithm (MCore) for determination of over-loaded and under-loaded hosts with the underload and over-load thresholds of 70% and 80%, respectively, combined with the selected biggest container to migrate (Maximum Usage) algorithm for container selection, outperforms other algorithms.

The authors in [14] have considered minimizing power consumption and performance loss as their goals. The authors have proposed a new comprehensive cloud resource management approach. Besides, they have proposed multi-criteria algorithms for two stages of consolidation subproblems, including 1) determining under-loaded hosts and, 2) VM placement. They have presented Enhanced Optimization (EO) policy for online resource management procedure and TOPSIS Power and SLA Aware Allocation (TPSA) policy for VM placement. Additionally, they considered multiple resources such as CPU, network, RAM, and Disk in their work. They have simulated their proposed algorithms in the CloudSim simulator to validate that their proposed policy has a notable influence on the reduction of energy consumption, number of VM migrations, and SLA violation in comparison with the state of the arts.

The authors in [15] proposed new proactive online resource management policies to optimize energy, SLA, and a number of migrations in cloud data centers. They have considered the minimization of power consumption and performance loss as their goals. Furthermore, they have addressed two phases of consolidation subproblems, including 1) detecting over-loaded host and, 2) VM selection for migrating. They have presented Window Moving Average Policy (WMA) as a novel forecasting algorithm for the designation of overloaded hosts as well as Multi-criteria TOPSIS with Prediction VM Selection (MTPVS) as a novel multi-criteria decision-making technique to select virtual machines for migration. In their proposed policies, they have considered multi-criteria counting RAM, CPU, and network bandwidth. In addition, they have utilized the CloudSim simulator to validate that their proposed algorithms have a significant reduction in SLA violation, energy consumption, and the number of migrations in comparison with state of the art.

The authors in [16] have addressed multi-target resource allocation for cloud data centers that applied a comprehensive view of the resource allocation problem. This paper has considered minimizing power consumption and performance loss as their goals. Also, they have introduced power, SLA violation, and Number of *VM* Migrations Genetic Algorithm (PSNGA) policy as a novel heuristic multi-criteria solution for energy-efficient resource allocation algorithm based on a genetic algorithm. Energy consumption of both IT equipment and cooling systems were their goals in addition to CPU, RAM, and network. Moreover, they have utilized the CloudSim simulator to confirm that their proposed algorithm outperforms state of the art regarding energy consumption, SLA violation, number of virtual machine migrations, and execution time.

The authors in [17] have considered minimizing power consumption and performance loss as their goals. In this study, they have investigated the addition of two new phases to the default on-line resource management process, including the VM sorting phase and condition evaluation phase. They have Proposed Multi-Criteria TOPSIS Sorting with Prediction (MCTP) policy as a new technique for VM sorting phase along with Minimum Downtime Migration Optimization (MDMO) policy as a new approach for the condition evaluation phase. Besides, this paper implied the shortage of other resource management methods, including non-consideration of all

essential and useful system metrics as well as not having load forecasting models. Additionally, in this study, they have considered multiple essential parameters, including CPU, RAM, and network. They have addressed four phases of consolidation subproblems, including 1 & 2) determining over-loaded and under-loaded host, 3) VM selection, and 4) VM placement. They have confirmed the applicability of their proposed policies via CloudSim simulator and showed that their proposed solutions have a notable influence on diminishing energy consumption as well as decreasing SLA violations and the number of VMs' migration in cloud data centers.

The authors in [18] have presented an integrated procedure to manage energy consumption and brownout in container-based cloud data centers. They have offered a brownout-based architecture by deactivating optional containers in applications or micro services temporarily to reduce energy consumption. Moreover, they have presented several policies to find the appropriate containers to be deactivated. Also, they appraised their performance in an archetype system. Their proposed Lowest Utilization Container First (LUCF) policy chooses a set of containers with the lowest utilization that reduces the utilization to less than the overloaded threshold of a host. Besides, their proposed Minimum Number of Containers First (MNCF) policy selects the minimum number of containers while reducing energy consumption in order to deactivate fewer services. For evaluating their proposed policies, they exploited the real traces in an archetype system. As their results show, their proposed approach has the minimum energy consumption in comparison with the approaches without power-saving techniques, brownout-overbooking approach, and auto-scaling approach, respectively while meeting the Quality of Service.

The authors in [19] have investigated a renewable energy-aware multi-indexed job classification and scheduling scheme using Container as-a-Service (CaaS) for data centers. As they mentioned, the workloads which have enough amount of renewable energy with them to handle incoming jobs are transferred to the datacenter from different devices. Consequently, they have designed a renewable energy-based container consolidation and host selection scheme. They have applied the Google workload traces in their evaluating and proved that their proposed approach has higher energy saving in comparison with the state of the arts.

The authors in [20] have investigated a general formulation of the elastic provisioning of Virtual machines for Container Deployment (EVCD) as an Integer Linear Programming problem. This approach optimizes the multiple QoS metrics, deployment time, and deployment cost as well as reallocating the containers at runtime. Moreover, they have considered the proposed formulation as a benchmark and have evaluated the algorithms such as Greedy first fit and Round-Robin, which applied for container deployment problem. They have evaluated and highlighted the drawbacks of the two heuristics. Their experiments have confirmed that the round-robin heuristic produces the highest deployment time and, working on a static pool of resources, is the most expensive solution for running containers. Meanwhile, the greedy first fit reduces the costs notably. As they have explored, since these heuristics are not aware of the characteristics of containers and VMs, they cannot determine an optimized allocation. They have designed and implemented a simulation program; their experiment results confirmed that their proposed algorithm could improve the efficiency of resource utilization.

The authors in [21] have explored a novel QoS-aware VM consolidation approach that adopts a method based on resource utilization history of virtual machines in a cloud environment. They have presented an efficient algorithm for finding an underutilized host as well as an efficient SLA-aware algorithm for finding new VM placement, namely Utilization and Minimum Correlation (UMC). In UMC, a VM will be migrated to the destination host if its CPU utilization has the lowest correlation with all the VMs CPU utilization, which are located on that host. Besides, they have utilized the VM-based dynamic threshold (VDT) algorithm periodically to recognize the underloaded hosts. They have appraised their proposed algorithm using the CloudSim simulator. Their simulation results demonstrate improvements in energy consumption and QoS metrics.

A comparison of our work and related works regarding their considered consolidation level (VM, container, or joint VM and container) as well as optimization goal (SLA, energy consumption, and number of either VM or container migration) is depicted in Table1. As shown in Table 1, our study is the only one that considers consolidating both VM and container in its proposed solution and also considers both VM and container migrations as its optimization goal.

Table 1
Comparison of our work and related works - Images

approach #REF	consolidation level			optimization goals				
	VM	container	joint VM & container	SLA	energy	number of migration		
						VM migration	container migration	joint VM&container
[1]	✓	×	×	✓	✓	✓	×	×
[3]	×	✓	×	✓	✓	✓	✓	×
[13]	✓	×	×	✓	✓	✓	×	×
[14]	✓	×	×	✓	✓	✓	×	×
[15]	✓	×	×	✓	✓	✓	×	×
[16]	✓	×	×	✓	✓	✓	×	×
[17]	✓	×	×	✓	✓	✓	×	×
[18]	×	✓	×	✓	✓	×	×	×
[19]	×	✓	×	✓	✓	×	✓	×
[20]	✓	✓	×	×	×	×	×	×
[21]	✓	×	×	✓	✓	✓	×	×
this paper	✓	✓	✓	✓	✓	✓	✓	✓

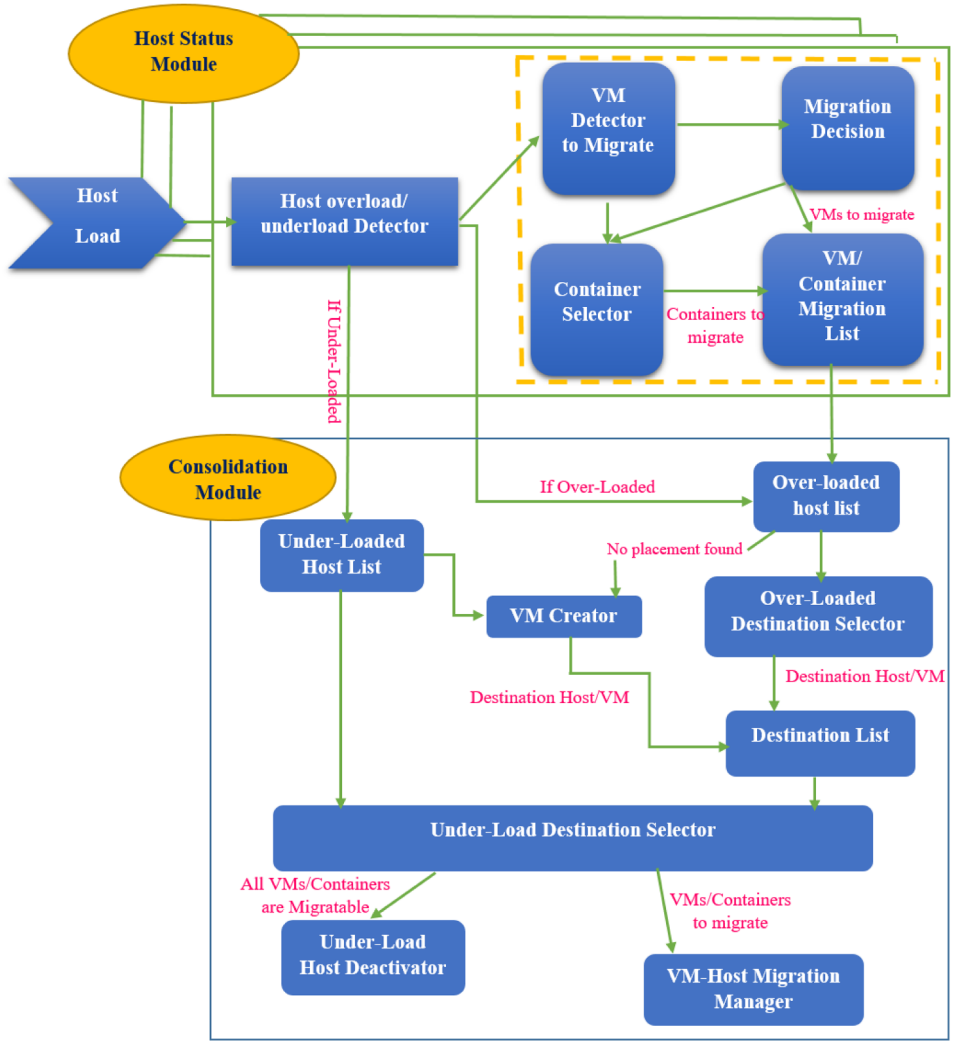

Fig. 2. proposed system model and flowchart.

3. Proposed System Model

Our proposed system model is shown in Fig. 2 with its components. It is an extended version of the system model defined in [3], which targets the CaaS environment, and applications are performed on containers. Containers are executed inside VMs which are hosted on PMs. Moreover, containers are defined by the demand of disk, network, CPU, and memory similar to VMs and PMs [3]. The goal of this model is consolidating containers on the minimum number of VMs and, accordingly, the smallest number of physical servers. The framework consists of the 'Host' module, which is executed on all PMs and the 'Consolidation' module, which is executed on a central PM.

Consolidation module

The consolidation module is set up on a separate node that decides proper destinations for the selected containers and VMs to be migrated. In the consolidation module, the overloaded destination selector is responsible for finding appropriate destinations for both containers and VMs in VM/container migration list. Moreover, the VM creator component approximates the number of required VMs to be initialized in the next processing window. It is performed according to the number of containers that are not assigned to any appropriate host or VM as the destination. The under-load destination selector component finds the best destinations for containers and VMs using the host selection algorithm. If it finds proper hosts as destinations for all the containers and VMs, then the information of destination hosts will be sent to the under-load host deactivator module which switches the under-loaded hosts off and also sends this information to the VM-Host migration manager. VM-Host migration manager is responsible for triggering the migration of VMs and containers.

Host module

The host status module exists in each active host in the data center and is consisted of five main components that are introduced in

this section. The host over-load and under-load detector components have the same functionality as their counterpart introduced in the system model proposed in [3]. However, the rest of the components in the host module are new components proposed in this paper to be added to the model proposed in [3].

Host Over-load/ Under-Load Detector

In this component, both over-load and under-load detection algorithms are implemented, which will be discussed in section 5.3. This component identifies the hosts' status as being either over-loaded or under-loaded by checking their resource utilization. The Identification information (ID) of both over-loaded and under-loaded hosts are sent to the 'VM detector to migrate' component. Also, the IDs of the over-loaded hosts are sent to the 'over-load host list' component, and the IDs of the under-load hosts are sent to the 'under-load host list' component in the consolidation module.

VM detector to migrate

In this component, the VMs which should be migrated from under-loaded and over-loaded hosts are identified using the algorithms discussed in section 5.3. The IDs of the candidate VMs for migration are sent to the 'migration decision' and 'container selector' components.

Migration decision

In this component, the VMs that should be migrated from over-loaded and under-loaded hosts are selected using our proposed algorithm introduced in section 4. Then, the IDs of the selected VMs are sent to both 'container selector' and 'container/VM migration list' components.

Container selector

In this component, containers that should be migrated from both over-loaded and under-loaded hosts are selected. All the containers that are located in under-loaded hosts but their hosting VMs are not selected for migration, are chosen for migration. For the over-loaded hosts, the containers are selected using the maximum usage (MU) policy [3]. Finally, the IDs of the selected containers are sent to the 'VM/container migration list' component.

VM/container migration list

In this component, the information of the containers and the VMs which are chosen to be migrated are registered, and this information is forwarded to the 'over-load host list' component in the consolidation module.

3.1. Power and energy Models

The power consumption of the data center at time t is calculated using Eq. (1) [3]:

$$P_{dc}(t) = \sum_{i=1}^{Ns} Pi(t) \tag{1}$$

Where *Ns* is the number of servers in the data center, and *Pi*(*t*) is the power consumption of the i'th server in the data center.

We estimate the power consumption of servers based on a linear relationship between power and CPU utilization of servers similar to previous related works [3]. This estimation comes from the idea that CPU is the major power consumer in a data center. We compute $U_{i,\,t}$ as the utilization of server *i* at time t using $U_{i,t} = \sum_{j=1}^{Nvm} \sum_{k=1}^{Nc} U_{c(k,j)}(t)$, where *Nvm* is the number of virtual machines and *Nc* is the number of containers hosting in VMs. Therefore, the power consumption of the i'th server is estimated through Eq. (2) [3].

$$P_i(t) = \begin{cases} P_i^{idle} + (P_i^{max} - P_i^{idle}) * U_{i,t}, & N_{vm} > 0 \\ 0, & N_{vm} < 0 \end{cases} \tag{2}$$

Where P_i^{idle} is the power consumption of a server in idle state, and P_i^{max} is the power consumption of a server at its full utilization, and total energy consumption is defined as follows in Eq. (3) [22]:

$$E = \int_{t0}^{t1} P(U(t))dt. \tag{3}$$

3.2. SLA Violation Metrics

In cloud computing environments, meeting QoS requirements is one of the momentous issues [13]. QoS provisions are commonly formalized in the type of SLAs that can be specified in terms of such characteristics as maximum response time or minimum throughput delivered by the deployed system. Because of the wide difference of the application in terms of these characteristics, specifying an independent workload metric is essential, which can be applied to measure the SLA delivered to any VM deployed in an IaaS such as SLA violation time per active host (SLATAH) metric defined in [1]. We use the SLA metrics introduced in [13] that is approximated through Eq. (4), which is composed of multiplication of two metrics: 1) the SLA violation time per active host (SLATAH) and, 2) performance degradation due to migration (PDM) as defined in Eq. (5).

$$SLAV = SLATAH.\ PDM \tag{4}$$

$$SLATAH = \frac{1}{N} \sum_{i=1}^{N} \frac{Tsi}{Tai},\ PDM = \frac{1}{M} \sum_{j=1}^{M} \frac{Cdj}{Crj} \tag{5}$$

Where Ts_i is the total time during which the host i has experienced the utilization of 100%; Ta_i is the total time during which the host i has been in the active state; N is the number of PMs; Cd_j is approximation of the performance degradation of VM_j caused by migrations which are estimated as 10% of the average CPU utilization in Million Instruction Per Second (MIPS) during all migrations of the VM_j; Cr_j is the total CPU capacity requested by the VM_j during its lifetime, and M is the number of VMs.

4. Joint VM and Container Multi-Criteria Migration Decision (JVCMMD) policy

The major drawback of current consolidation solutions in cloud data centers is that they do not consider joint container and VM migration in their decision making process. To overcome this limitation, this paper proposes a new joint VM and container consolidation procedure which divides the cloud resource management problem into seven sub-problems including 1) over-loaded host detection, 2) under-loaded host detection, 3) identifying whether VMs or containers should be migrated by determining the candidate VMs that should be migrated from over-loaded and underloaded hosts, 4) VM selection from candidate VMS for migration, 5) VM placement, 6) container selection for migration, and 7) container placement. In addition, this paper proposes a novel policy named Joint VM and Container Multi Criteria Migration Decision (JVCMMD) technique for the third sub-problem which is described in this section.

Our proposed consolidation procedure for migration decision is shown in Algorithm 1. First, the over-loaded hosts are found using Local Regression (LR) policy [13]. Second, the under-loaded hosts are found using Simple Method (SM) policy [13]. Third, our proposed JVCMMD policy is applied to decide whether the VMs or containers should be migrated from over-loaded and under-loaded hosts by determining the candidate VMs for migration. Fourth, the VMs that should be migrated among the candidate VMs are selected using Minimum Migration Time (MMT) policy [13]. Fifth, new destination hosts are determined for selected migrating VMs using TOPSIS Power and SLA aware Allocation (TPSA) algorithm [14]. The VMs that were selected for migration are removed from the initial candidate VM list and the list of remaining VMs that were needed to be migrated but are not selected for migration is built. Sixth, the migrating VMs that no suitable destination host is found for them are addressed and appropriate containers hosting on them are selected for migration using Maximum Usage (MU) policy [3]. Seventh, new destinations are found for the selected containers using Correlation Threshold Host Selection Algorithm (CORHS) with Least Full Host Selection Algorithm (LFHS) as alternative algorithm [3]. In the rest of this section we describe the JVCMMD policy in detail.

JVCMMD policy takes advantage of TOPSIS as a multi-criteria algorithm and considers six criteria depicted in Table 2 in its decision. This policy ranks all the VMs that are the candidate of migration using the method described in this section and selects the VMs with the highest score. The criteria considered in the JVCMMD policy have either benefit or cost types. The more the value of criteria with the benefit type, and the lower the value of criteria with the cost type, the closer is the answer to the optimum point.

JVCMMD ranks a VM at the highest position that passes the following conditions simultaneously: (1) the selected VM for migration has the most correlation with other deployed VMs in the PM, (2) the selected VM has the highest CPU utilization, (3) the selected VM is the most over-loaded one, (4) the selected VM has the least MIPS capacity, (5) the selected VM has the greatest number of containers, and (6) the selected VM has the least storage capacity.

A VM is determined to be over-loaded using Eq. (6) [3]:

$$\sum_{i=0}^{NOC} ContainerMIPS_i > \frac{HostMIPS}{\#CPUCoreHost \times \#CPUCoresVM} \tag{6}$$

Where $ContainerMIPS_i$ is the utilized computing capacity of i'th container in MIPS, HostMIPS is the total computing capacity of the PM hosting the VM in MIPS. #CPUCoreHost is the number of processing elements of the PM hosting the VM, and #CPUCoresVM is the number of processing elements of the candidate VM. The rationale behind considering six decision criteria shown in Table 2 in JVCMMD policy is as follows. 1) Applying resource correlation (RC) in JVCMMD policy is based on the idea that the higher the correlation between applications that use the same resources on an oversubscribed server, the higher the possibility of the server to become overloaded [23].

So, a VM should be selected that has the most correlation with other VMs hosting in a PM. 2) Furthermore, VM CPU utilization (VMCU) is defined as a benefit type because of the more the CPU utilization of a VM, the more the chance of resource cutting, and accordingly, the more the chance of SLA violation. Therefore, this VM should be selected with a higher priority for migration. 3) The VM overload parameter (OLVM) is defined as a benefit type because when the computing capacity of a VM is over-utilized due to high resource usage of the containers hosting on it, the probability of resource shortage in the hosting PM and consequently SLA violation

Table 2
considered Criteria in JVCMMD policy

No.	Notation	Parameter	Description	Cost/ benefit
1	RC	Resource correlation	Resource correlation of a VM with other deployed VMs in a physical machine	benefit
2	VMCU	VM CPU Utilization	The CPU utilization of a VM	benefit
3	OLVM	Over-load VM	The sum of MIPS of containers hosting in a VM.	benefit
4	VMM	VM MIPS	The computing capacity of a VM in MIPS.	cost
5	NOC	Number of Container	The number of available containers in a VM	benefit
6	VMS	VM Storage	The storage capacity of a VM	cost

increases. So, the overloaded VM should be selected for migration with higher priority. 4) The VM MIPS (VMM) is considered as a cost type because of the higher the computing capacity of a VM, the higher the risk of resource shortage to host containers. 5) Also, the selection of a VM with the least number of containers (NOC) leads to the condition in which there are fewer competent containers to access the shared resources, and consequently, the SLA violation reduces. 6) Moreover, by the selection of a VM with the least storage capacity (VMS), the migration overhead due to storage transmission is decreased.

All the information assigned to the VMs in time slot t forms a decision matrix *Multi-Criteria Decision Matrix (MCDM)*, as shown in Eq. (7).

$$\overline{MCDM} = \begin{bmatrix} RC_{VM1} & CU_{VM1} & OL_{VM1} & M_{VM1} & NOC_{VM1} & S_{VM1} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ RC_{VMi} & CU_{VMi} & OL_{VMi} & M_{VMi} & NOC_{VMi} & S_{VMi} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ RC_{VMn} & CU_{VMn} & OL_{VMn} & M_{VMn} & NOC_{VMn} & S_{VMn} \end{bmatrix}. \tag{7}$$

In this matrix VM_1, ... VM_n are the candidate VMs for migration. RC_{VM}, CU_{VM}, OL_{VM}, M_{VM}, NOC_{VM}, *and* S_{VM} are RC, VMCU, OLVM, VMM, NOC, and VMS criteria defined in Table 2. In order to select the best VM, the subsequent process is followed:

Step 1: First, we normalize the decision matrix *MCDM* to have dimensionless decision matrix *MCDMNorm,* as shown in Eq. (8). The decision matrix is made dimensionless by dividing each entry by the maximum value of each column using equation $Cri_{Nor}^{VMi} = \frac{CrVMi}{Crmax}$ in which Cri is the i'th criterion shown in Table 2; Nor is an abbreviation for normalized; Crvmi and Crmax are the criterion value of VMi and the maximum value for the criterion, respectively.

$$\overline{MCDM} = \begin{bmatrix} \frac{RC_{VM1}}{RC_{\max}} & \frac{CU_{VM1}}{CU_{\max}} & \frac{OL_{VM1}}{OL_{\max}} & \frac{M_{VM1}}{M_{\max}} & \frac{NOC_{VM1}}{NOC_{\max}} & \frac{S_{VM1}}{S_{\max}} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \frac{RC_{VMi}}{RC_{\max}} & \frac{CU_{VMi}}{CU_{\max}} & \frac{OL_{VMi}}{OL_{\max}} & \frac{M_{VMi}}{M_{\max}} & \frac{NOC_{VMi}}{NOC_{\max}} & \frac{S_{VMi}}{S_{\max}} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \frac{RC_{VMn}}{RC_{\max}} & \frac{CU_{VMn}}{CU_{\max}} & \frac{OL_{VMn}}{OL_{\max}} & \frac{M_{VMn}}{M_{\max}} & \frac{NOC_{VMn}}{NOC_{\max}} & \frac{S_{VMn}}{S_{\max}} \end{bmatrix} \tag{8}$$

Step 2: In the next step, VM^+ and VM^- are determined. Before determining VM^+ and VM^-, the type of each attribute should be defined. As shown in Table 2, each attribute can be considered to have either benefit or cost type. Larger values for a benefit type attribute lead to less distance from VM^+ and more distance from VM^-, while the opposite condition exists for a cost type variable. Therefore, VM^+ and VM^- are defined in Eq. (9) and Eq. (10), respectively.

$$VM^- = \{RC_{NORM}^-, VMcu_{NORM}^-, OLVM_{NORM}^-, VMM_{NORM}^+, NOC_{NORM}^-, VMS_{NORM}^+\} \tag{9}$$

$$VM^- = \{RC_{NORM}^-, VMcu_{NORM}^-, OLVM_{NORM}^-, VMM_{NORM}^+, NOC_{NORM}^-, VMS_{NORM}^+\} \tag{10}$$

Where CRI_{NOR}^+ and $CRI^-{}_{NOR}$ are the maximum and minimum values in each column of *MCDMNorm*, respectively, adopting the value of CRI_{NOR}^+ or $CRI^-{}_{NOR}$ to compute VM^+ and VM^- depends on the type of the criterion which can be either benefit or cost as shown in the fifth column of Table 2. More precisely, if the criterion has a benefit type, then CRI_{NOR}^+ is equal to the maximum value in each column of *MCDMNorm,* and $CRI^-{}_{NOR}$ is equal to the minimum value in each column of *MCDMNorm* as defined in Eq. (11). Moreover, an opposite condition is applied to cost type Criteria to compute CRI^+Norm and CRI^-Norm as defined in Eq. (12). Moreover, the Criterion is one of the parameters specified in Table 2.

$$CRI_{Norm}^+ = \text{Max}\, CRI_{Norm} \;\; CRI \in Benefit \;\; . \;\; Min \;\; CRI_{Norm} \;\; CRI \in Cost \tag{11}$$

$$CRI_{Norm}^- = \text{Min}\, CRI_{Norm} \;\; CRI \in Benefit \;\; . \;\; Max \;\; CRI_{Norm} \;\; CRI \in Cost \tag{12}$$

Step 3: The distances from the positive ideal solution D_{VM+} and the negative ideal solution D_{VM-} for VMi are calculated using Eq. (13) and Eq. (14), respectively.

$$D_{VMi}^+ = \sum_{j=1}^{n} D\,(CRI_{Norm}^{VMi}, CRI_{Norm}^+) \tag{13}$$

$$D_{VMi}^- = \sum_{j=1}^{n} D\,(CRI_{Norm}^{VMi}, CRI_{Norm}^-) \tag{14}$$

Step 4: the score of VM is calculated using Eq. (15):

$$Score_{VMi} = \frac{D_{VMi}^-}{D_{VMi}^+ + D_{VMi}^-} \tag{15}$$

Where $Score_{VMi}$ shows the score of i'th VM. The more distance a VM has from VM^-, the more the value of the nominator of Eq. (15), and consequently, the larger is the score value. Similarly, the less distance a VM has from VM^+, the less the value of the denominator of Eq. (15), and accordingly, the larger is the score value.

Step 5: Finally, the VMs which have the highest score than the average score of the whole of the VMs are selected for migration.

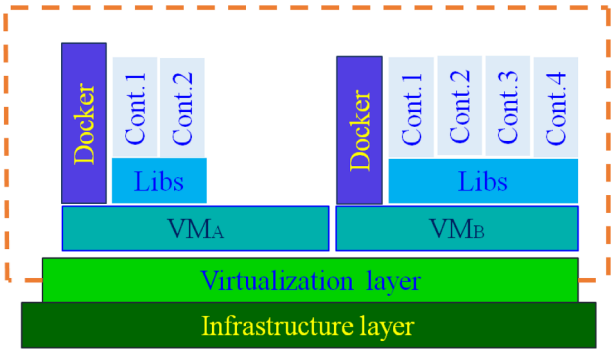

Fig. 3. Virtualized environment in container CloudSim [3]

5. Performance Evaluation

We evaluate performance of our proposed solution and compare it with recent energy-aware resource allocation studies that are close to our study, including [3, 13, 14], and [21] as benchmarks.

5.1. Experiment Setup

Since implementing and evaluating the proposed algorithms in a real environment is very expensive and time-consuming, it is necessary to analyze it on a virtualized data center infrastructure [24]. Furthermore, it is difficult to perform and compare duplication of large scale examinations on real infrastructures. Hence, we have used simulation for evaluation of our proposed algorithms. We take advantage of "Container CloudSim" for simulation [3]. Container CloudSim is an extended version of the CloudSim simulator. This toolkit provides an environment for evaluation of resource management paradigms such as container scheduling, container consolidation, and placement. The layered architecture of Container CloudSim including infrastructure, virtualization, VM, library, and container layers are depicted in Fig. 3.

Container CloudSim enables researchers to simulate different resource management techniques in virtual environments containing system/container and operating system/ virtual machine virtualization [3]. Our infrastructure setup has a real configuration of a cloud computing infrastructure including a data center with 800 installed heterogeneous physical machines consisting of 200 HP ProLiant ML110 G4, 200 HP ProLiant ML110 G5, 200 IBM Server x3250, and 200 IBM Server x3550. The characteristics of these machines are depicted in Table 3. Power consumptions of physical machines are computed based on the model described in section 3.1. Configuration of containers and VMs are depicted in Tables 4, and 5, respectively. In order to evaluate the algorithms considering the aforementioned simulation setup and configurations, we applied the workload traces from PlanetLab [25]. This workload contains CPU utilization in 5-min intervals of more than a thousand VMs that are located at more than 500 servers around the world. The characteristics of the data for each day are shown in Table 6. During the simulations, each VM is randomly assigned a workload trace from one of the VMs to the corresponding day.

5.2. Performance Metric

A solution with the lowest energy consumption, SLA violation, and the number of migrations is the best one. As these objectives are negatively correlated [1], similar to benchmark studies, to consider the simultaneous minimization of energy and SLA violation, the ESV metric defined in [13] is adopted. Also, to consider the simultaneous minimization of energy, SLA violation, and number of VM migrations, the ESM metric defined in [14] is adopted. Moreover, in order to assess the number of container migrations along with energy consumption and SLA violation, we utilize the ESCM metric defined in Eq. (18) which is computed by multiplication of the value of the mentioned objectives.

$$ESV = Energy.\ SLAviolation \tag{16}$$

$$ESM = Energy.\ SLAV.\ Number of VM migration \tag{17}$$

Table 3
The configuration of the servers

Server types	CPU Number	Memory (GB)	CPU(Cores)	Population
Hp ProLiant Ml110 G4	2	4	Intel Xeon 3040	200
Hp ProLiant Ml110 G5	2	4	Intel Xeon 3075	200
IBM server x3550	2×6	16	2x intel Xeon X5675	200
IBM server x3250	4	8	intel Xeon X3470	200

Table 4
The configuration of containers

Container type	CPU (cores)	CPU (MIPS)	Memory (GB)
1	1	4658	128
2	1	9320	256
3	1	18636	512

Table 5
VM types

VM types	CPU (cores)	CPU [1.5 GHz) (MIPS] (mapped on 18636 MIPS Per core)	Memory (GB)
1	2	2 cores	1
2	4	4 cores	2
3	1	1 core	4
4	8	8 cores	8

Table 6
Workload data characteristics [25]

Date	Number of VMs	Workload Mean (%)	Workload SD (%)
03/03/2011	1052	12.31	17.09
06/03/2011	898	11.44	16.83
09/03/2011	1061	10.70	15.57
22/03/2011	1516	9.26	12.78
25/03/2011	1078	10.56	14.14
03/04/2011	1463	12.39	16.55
09/04/2011	1358	11.12	15.09
11/04/2011	1233	11.56	15.07
12/04/2011	1054	11.54	15.15
20/04/2011	1033	10.43	15.21

$$ESCM = Energy.\ SLAV.\ Number of container migration \tag{18}$$

5.3. Simulation Results

In this section, we evaluate our proposed policy by comparing it with five other benchmark scenarios proposed in the state of the arts, including [3, 13, 14], and [21]. We have numbered scenarios from one to six, as depicted in the first column of Table 7. Our proposed solution is scenario 6. In the third column of Table 7, we have defined a segmented naming format for the notation of the scenarios assessed in this section. The number of sections in the naming format is equal to the number of sub-problems considered in each paper for a consolidation problem. The notations are constructed by connecting the abbreviation of the policies used for each sub-problem using slash lines. The six considered scenarios are defined as follows.

Scenario 1. There are four segmented VM consolidation solution with the combination of the best algorithms proposed in [3] including static threshold policy (Static THR) with 70% threshold for determination of under-loaded host and 80% threshold for determination of over-loaded hosts, Maximum Usage (MU) for VM selection, and correlation host selection policy (CORHS) to find new destinations for the VMs that should be migrated. If the correlation of the VMs and the destination PM is not available, the least full host selection (LFHS) policy is used as an alternative.

Scenario 2. There are four segmented VM consolidation solution with the combination of the best algorithms proposed in [13] including Local regression (LR) policy for determination of over-loaded PMs, simple method (SM) for determination of under-loaded hosts, minimum migration time (MMT) policy to choose the VMs that should be migrated, and power-aware best fit decreasing

Table 7
the characteristics of scenarios

Scenario number	Name of scenario	Policy abbreviation
1	VM migration [3]	Static THR/MU/CORHS, LFHS
2	VM migration [13]	LR/SM/MMT/PABFD
3	VM migration [14]	LR/TACND/MMT/TPSA
4	VM migration [21]	LR/VDT/MMT/UMC
5	Container migration [3]	Static THR/MU/CORHS, LFHS
6	Joint VM and container migration	LR/SM/MMT/JVCMMD/MU/TPSA/CORHS, LFHS

(PABFD) policy for finding new placement for the VMs to be migrated.

Scenario 3. There are four segmented VM consolidation solution with the combination of the best algorithms proposed in [14] including LR policy for determination of over-loaded PMs, TOPSIS-Available Capacity and Number of VM and migration Delay (TACND) for designation of underloaded PMs, MMT policy for selection of the VMs that should be migrated, and TOPSIS power and SLA aware allocation (TPSA) for VM placement.

Scenario 4. There are four segmented VM consolidation solution with the combination of best algorithms proposed in [21] including LR policy for determination of over-loaded PMs, VM based Dynamic Threshold (VDT) for designation of underloaded hosts, MMT policy for selection of the VMs that should be migrated, and host Utilization and Minimum Correlation (UMC) for VM placement.

Scenario 5. There are four segmented container consolidation solution with the combination of the best algorithms proposed in [3] including Static threshold with 70% and 80% thresholds for determination of underloaded hosts, MMT policy for selection of the VMs that should be migrated, and host Utilization and Minimum Correlation (UMC) for VM placement. under-loaded and over-loaded PMs, respectively, maximum usage (MU) policy for container selection for migration, and CORHS placement policy with LFHS as alternative algorithm for container placement.

Scenario 6. There are seven segmented Joint VM and container consolidation solution proposed in this paper including LR for determination of over-loaded hosts, SM for determination of under-loaded hosts, MU policy for identification of containers for migration, MMT policy for identification of the VMs for migration, JVCMMD policy for making the decision whether VMs or containers should be migrated, CORHS policy for container placement with LFHS policy as its alternative algorithm when the correlation of the VMs and the destination PM is not available, and TPSA policy for finding new placements for VMs.

Ten experiments are performed separately for the ten days of workloads depicted in Table 6 and their median results for energy consumption, number of VM migrations, number of container migrations, SLAV, ESV, ESM, and ESCM metrics are reported in Table 8. Figs. 4-10 show the box plot for energy consumption, the value of SLAV, the value of ESV metric, the number of VM migration, the value of ESM metric, the number of container migration, and the value of ESCM metric for different scenarios, respectively. A box plot—also called a box and whisker plot—displays the five-number summary of a set of data. The five-number summary is the minimum, first quartile, median, third quartile, and maximum. In a box plot, we draw a box from the first quartile to the third quartile. The yellow and green boxes show the distance between first quartile to the median and between median to the third quartile, respectively. A vertical line goes through the box at the median. The whiskers go from each quartile to the minimum or maximum.

According to Figs. 4–9 and Fig. 10, it can be inferred that our proposed solution leads to better performance regarding the number of VM migration, number of container migration, SLA violation, ESV metric, ESM metric, and ESCM metric in comparison with other scenarios. We compare the results of our proposed solution with the best previous ones in two categories which were based on either only VM migrations (scenario 3) or only container migrations (scenario 5). More precisely, it can be deduced from Table 8 that adoption of our proposed scenario (scenario 6) leads to 9.9%, 52.83%, 69.76%, 2.4%, and 67.17%, decrease in energy consumption, SLA violation, ESV metric, number of VM migrations, and ESM metric, respectively, in comparison with the best previous scenario based on only VM migrations (scenario 3). Also, it can be inferred from Table 8 that adoption of our proposed scenario (scenario 6) leads to 39.96%, 99.99%, 99.99%, 53.24%, 99.99% decrease in energy consumption, SLA violation, ESV metric, number of container migrations, and ESCM metric, respectively, in comparison with the best previous scenario based on only container migrations (scenario 5). So, the obtained results validate the applicability of our proposed scenario for consolidation in cloud data centers. This observation can be described by the fact that our proposed solution proposes a joint VM and container consolidation solution and takes advantage of the JVCMMD policy to determine whether VMs or containers should be migrated. Moreover, JVCMMD policy takes advantage of both VM and container migration to consolidate VMs so that the VMs that were selected for migration but no suitable destination host is found for them are also migrated by migrating their containers, which notably improves the results. On the other hand, as can be seen in Table 8, scenario 6 shows a negligible more energy consumption than scenario 4, but the SLA violation and the number of VM migrations in scenario 4 is extremely more than scenario 6. More precisely, it can be deduced from the results shown in Table 8 that adoption of our proposed scenario (scenario 6) leads to 99.99%, 99.99%, 85.4%, and 99.99% decrease in SLA violation, ESV metric, number of VM migrations, and ESM metric, respectively, in comparison with scenario 4, while having a negligible increase in energy consumption. It is important to note that the TPSA policy makes it possible to tune a trade-off between output targets, including energy consumption, SLA violation, and the number of migrations [14].

Table 8
Median values of output results for six consolidation solutions

Scenario number	Energy Consumption (KWH)	Number of VM migration	Number of container migration	SLA Violation ($\times 10^2$)	ESV	ESCM	ESM	ESCM improvement (%)	ESM Improvement (%)
1	81.938	1770	0	1.35	95.87	0	170495.4	_	99.99
2	65.562	730.5	0	0.4615	24.85	0	18757.16	_	99.99
3	54.64	79.1	0	0.000106	0.0086	0	0.6269	_	67.17
4	49.17	531.1	0	0.5294	28.39	0	18836.1	_	99.99
5	81.935	0	408.5	0.4415	25.38	11640.74	0	99.99	_
6	49.19	77.2	191	0.00005	0.0026	0.90574	0.2058	_	_

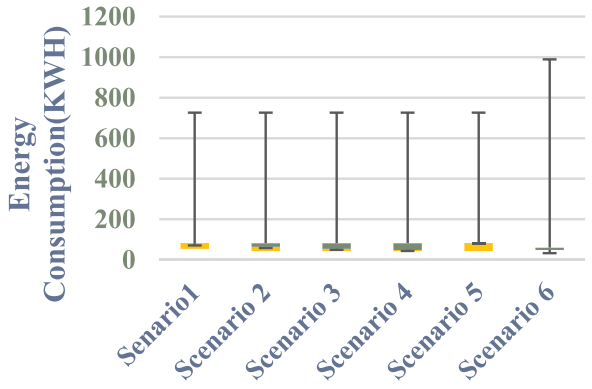

Fig. 4. Energy consumption of different policies.

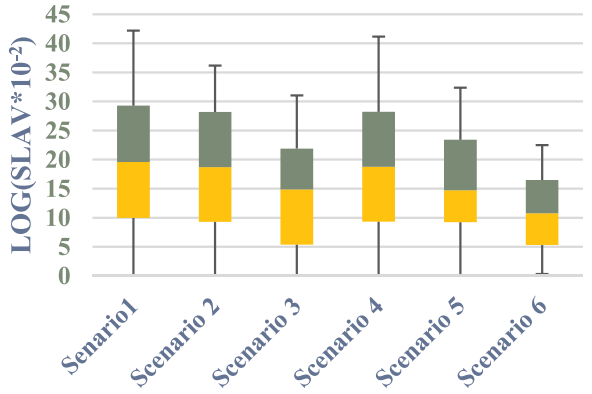

Fig. 5. SLA violation (SLAV) of different policies.

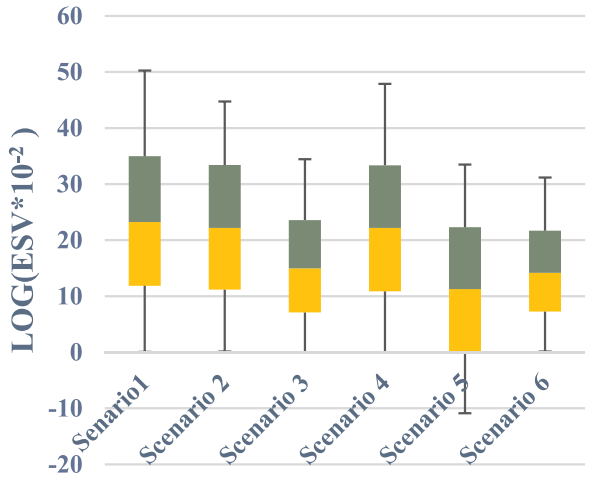

Fig. 6. ESV of different policies.

5.3.2. Statistical Analysis

The statistical analysis of the obtained results is presented in this section. According to the Ryan-Joiner's normality test, the values of the ESM metric produced by all six studied scenarios follow a normal distribution with the P-value > 0.1. Table 9 demonstrates the result of paired t-tests of our proposed algorithm (scenario 6) and the benchmark algorithms. It can be inferred from the obtained results that there is a statistically significant difference between our proposed algorithm and benchmark algorithms. The t-tests show that our proposed algorithm causes a statistically notably lower value of the ESM metric with the P-value < 0.001. Table 10 compares our proposed algorithm and benchmark algorithms considering the mean values of the ESM metric along with 95% Confidence Intervals (CI). According to Table 10, we can conclude that our proposed algorithm has the best performance regarding ESM metrics.

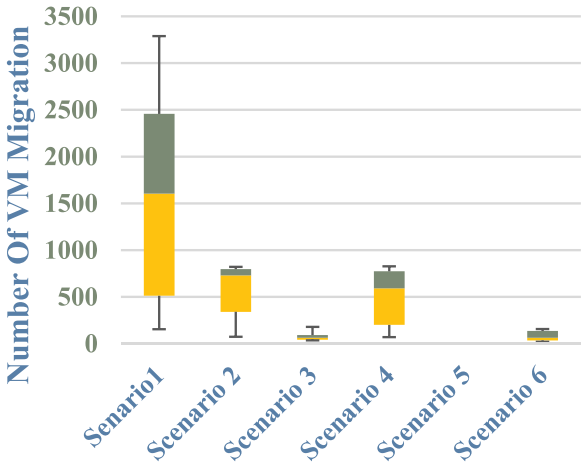

Fig. 7. Number of VM Migrations of different policies.

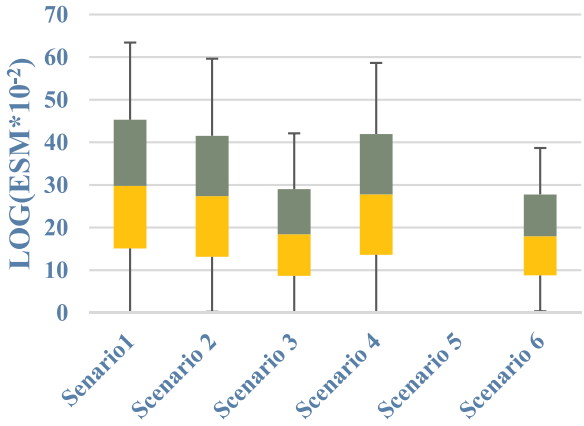

Fig. 8. ESM of different policies.

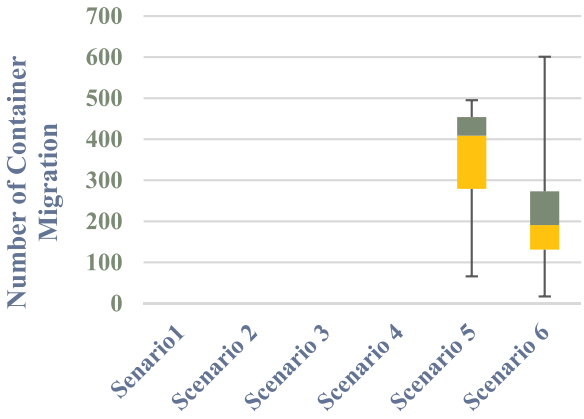

Fig. 9. Number of Container Migrations of different policies.

6. Conclusions and Future Work

The rapid adoption of cloud computing has led to the construction of large-scale data centers that consume a tremendous amount of power. Improving the energy efficiency of cloud data centers is a continuous challenge that can increase the cloud providers' return on investment (ROI) along with the reduction of greenhouse gas emissions. However, despite the increasing popularity of container data centers, the energy efficiency of resource management algorithms in this deployment model is not deeply studied in the literature.

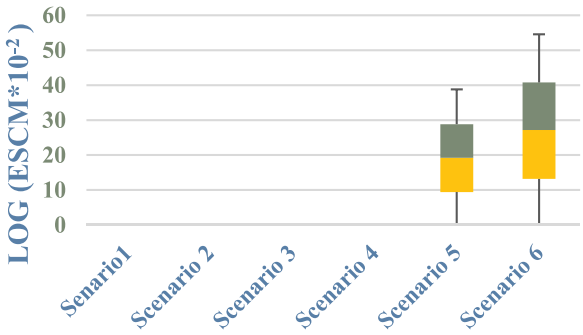

Fig. 10. ESCM of different policies.

Table 9
Comparison of our proposed and benchmark algorithms using paired t tests regarding ESM metric.

Policy 1 (ESM)	Policy 2 (ESM)	Difference	P value
Scenario 6 (0.2058)	**Scenario 1** (170495.4)	266.95(141.71, 392.2047)	p-value < 0.001
Scenario 6 (0.2058)	**Scenario 2** (18757.16)	0.265064(0.2076, 0.322443)	p-value < 0.001
Scenario 6 (0.2058)	**Scenario 3** (0.626909)	0.002881(0.001629, 0.004132)	p-value < 0.001
Scenario 6 (0.2058)	**Scenario 4** (18836.1)	10.25056(5.27506, 1522606)	p-value < 0.001
Scenario 6 (0.2058)	**Scenario 5** (0)	___	p-value < 0.001

Table 10
Comparison of our proposed and benchmark algorithms regarding the mean values and 95% confidence intervals of ESM metric

Scenario number	policy	ESM	95% CI
1	Static THR/MU/CORHS, LFHS	14171308.1	(-10877859, 3293449.04)
2	LR/SM/MMT/PABFD	20769.7	(9292.81, 30062.51)
3	LR/TACND/MMT/TPSA	164.0936	(-87.8179, 76.27572)
4	LR/VDT/MMT/UMC	527507.1	(-467595, 59912.3)
5	Static THR/MU/CORHS, LFHS	0	0
6	LR/SM/MMT/JVCMMD/MU/TPSA/CORHS, LFHS	1.174103	(-0.4985, 0.675598)

Algorithm 1
: Consolidation Procedure.

Input: All VMs, All Hosts, All Containers,
Output: MigrationMap.
Identify over-utilized hosts using **LR** policy.
Identify under-utilized hosts using **SM** policy.
Identify switched off hosts.
for each OL/UL host **do**
  Candidate VMs for migration using **JVCMMD** policy.
  Select VMs to be migrated from candidate VMs, using **MMT** policy.
    if VMs are selected **then**
      Place VMs on appropriate hosts using **TPSA** policy and put them in the migration map.
    else
      Identify the containers of migrating VMs that no suitable destination is found for them, using **MU** policy.
      Place containers using **CORHS/LFHS** policy and put them in the migration map.
    end if.
end for.
return migration map.

In this study, we addressed the problem of green resource management in container-based cloud data centers. We concentrated on multi important objectives, including energy consumption, SLA violation, and the number of both VM and container migrations. We took advantage of simultaneous container and VM consolidations to improve the previous obtained results in the literature. In this regard, we proposed the joint VM and container consolidation solution. More precisely, first of all, we proposed the joint VM and container consolidation flowchart. We divided the joint VM and container consolidation problem into eight sub problems. Furthermore, we proposed a multi-criteria migration decision (JVCMMD) algorithm to make a decision about whether the containers should be migrated or the VMs.

The results of the experiments obtained from an extensive assessment of our proposed solutions using an extension of the

Container CloudSim simulator validated that our proposed solution outperforms existing solutions. More precisely, our proposed LR/SM/MMT/JVCMMD/MU/TPSA/CORHS, LFHS scenario demonstrated notable 99.28% and 99.99% reductions in ESM and ESCM metrics. In future work, we plan to evaluate the proposed algorithms in a real cloud infrastructure such as OpenStack. Another research direction is considering multi-criteria, including RAM, network bandwidth, memory, and cooling system rather than only CPU criterion. We also plan to investigate the heuristic algorithms for joint VM and container consolidation problems.

References

[1] E. Arianyan, H. Taheri, V. Khoshdel, Novel fuzzy multi objective DVFS-aware consolidation heuristics for energy and SLA efficient resource management in cloud data centers, Journal of Network and Computer Applications 78 (2017) 43–61.
[2] B. Varghese, R Buyya, Next generation cloud computing: New trends and research directions, Future Generation Computer Systems 79 (2018) 849–861.
[3] S.F. Piraghaj, A.V. Dastjerdi, R.N. Calheiros, R. Buyya, ContainerCloudSim: An environment for modeling and simulation of containers in cloud data centers, Software: Practice and Experience 47 (2017) 505–521.
[4] G. Sakellari, G. Loukas, A survey of mathematical models, simulation approaches and testbeds used for research in cloud computing, Simulation Modelling Practice and Theory 39 (2013) 92–103.
[5] M.J. Scheepers, Virtualization and containerization of application infrastructure: A comparison, in: 21st Twente Student Conference on IT. 2014).
[6] C. Pahl, Containerization and the paas cloud, IEEE Cloud Computing 2 (2015) 24–31.
[7] T. Kaur, I. Chana, Energy efficiency techniques in cloud computing: A survey and taxonomy, ACM computing surveys (CSUR 48 (2015) 1–46.
[8] G.G. Castañé, A. Nunez, P. Llopis, J. Carretero, E-mc2: A formal framework for energy modelling in cloud computing, Simulation Modelling Practice and Theory 39 (2013) 56–75.
[9] F. Farahnakian, T. Pahikkala, P. Liljeberg, J. Plosila, H. Tenhunen, Utilization prediction aware VM consolidation approach for green cloud computing, 2015 IEEE 8th International Conference on Cloud Computing, IEEE, 2015, pp. 381–388.
[10] D. Borgetto, H. Casanova, G. Da Costa, J.-M. Pierson, Energy-aware service allocation, Future Generation Computer Systems 28 (2012) 769–779.
[11] Z. Zhou, J. Abawajy, M. Chowdhury, Z. Hu, K. Li, H. Cheng, A.A. Alelaiwi, F. Li, Minimizing SLA violation and power consumption in Cloud data centers using adaptive energy-aware algorithms, Future Generation Computer Systems 86 (2018) 836–850.
[12] M.A. Haghighi, M. Maeen, M. Haghparast, An Energy-Efficient Dynamic Resource Management Approach Based on Clustering and Meta-Heuristic Algorithms in Cloud Computing IaaS Platforms, Wireless Personal Communications 104 (2019) 1367–1391.
[13] A. Beloglazov, R. Buyya, Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in cloud data centers, Concurrency and Computation: Practice and Experience 24 (2012) 1397–1420.
[14] E. Arianyan, H. Taheri, S. Sharifian, Novel energy and SLA efficient resource management heuristics for consolidation of virtual machines in cloud data centers, Computers & Electrical Engineering 47 (2015) 222–240.
[15] E. Arianyan, H. Taheri, S. Sharifian, Novel heuristics for consolidation of virtual machines in cloud data centers using multi-criteria resource management solutions, The Journal of Supercomputing 72 (2016) 688–717.
[16] E. Arianyan, H. Taheri, S. Sharifian, Multi Target Dynamic VM Consolidation in Cloud Data Centers Using Genetic Algorithm, J. Inf. Sci. Eng. 32 (2016) 1575–1593.
[17] E. Arianyan, H. Taheri, S. Sharifian, M. Tarighi, New six-phase on-line resource management process for energy and sla efficient consolidation in cloud data centers, Int. Arab J. Inf. Technol. 15 (2018) 10–20.
[18] M. Xu, A.N. Toosi, R. Buyya, Ibrownout: an integrated approach for managing energy and brownout in container-based clouds, IEEE Transactions on Sustainable Computing 4 (2018) 53–66.
[19] N. Kumar, G.S. Aujla, S. Garg, K. Kaur, R. Ranjan, S.K. Garg, Renewable energy-based multi-indexed job classification and container management scheme for sustainability of cloud data centers, IEEE Transactions on Industrial Informatics 15 (2018) 2947–2957.
[20] M. Nardelli, C. Hochreiner, S. Schulte, Elastic provisioning of virtual machines for container deployment, Proceedings of the 8th ACM/SPEC on International Conference on Performance Engineering Companion, 2017, pp. 5–10.
[21] A. Horri, M.S. Mozafari, G. Dastghaibyfard, Novel resource allocation algorithms to performance and energy efficiency in cloud computing, The Journal of Supercomputing 69 (2014) 1445–1461.
[22] A. Beloglazov, J. Abawajy, R. Buyya, Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing, Future generation computer systems 28 (2012) 755–768.
[23] A. Verma, G. Dasgupta, T.K. Nayak, P. De, R. Kothari, Server workload analysis for power minimization using consolidation, Proceedings of the 2009 conference on USENIX Annual technical conference, 2009, p. 28 USENIX Association.
[24] T. Guérout, T. Monteil, G. Da Costa, R.N. Calheiros, R. Buyya, M. Alexandru, Energy-aware simulation with DVFS, Simulation Modelling Practice and Theory 39 (2013) 76–91.
[25] K. Park, V.S. Pai, CoMon: a mostly-scalable monitoring system for PlanetLab, ACM SIGOPS Operating Systems Review 40 (2006) 65–74.