J Supercomput (2016) 72:688–717
DOI 10.1007/s11227-015-1603-9

Novel heuristics for consolidation of virtual machines in cloud data centers using multi-criteria resource management solutions

Ehsan Arianyan[1] · **Hassan Taheri**[1] · **Saeed Sharifian**[1]

Published online: 31 December 2015

Abstract Increasing demand for acquiring diverse range of services has led to the establishment of huge energy hungry cloud data centers all around the world. Cloud providers face with major concerns to reduce their energy consumption while ensuring high quality of service based on the Service Level Agreement (SLA). Consolidation is proposed as one of the most effective techniques for online energy saving in cloud environments with dynamic workloads. This paper proposes novel proactive online resource management policies to optimize energy, SLA, and number of migrations in cloud data centers. More precisely, this paper proposes new prediction algorithm for determination of overloaded hosts as well as novel multi-criteria decision making techniques to select virtual machines. The results of simulations using CloudSim simulator shows up to 98.11 % reduction in the output metric which is representative of energy consumption, SLA violation, and number of migrations, in comparison with state of the art.

Keywords Cloud computing · Consolidation · Data center · Energy consumption · SLA violation · Migration

✉ Hassan Taheri
htaheri@aut.ac.ir

Ehsan Arianyan
ehsan_arianyan@aut.ac.ir

Saeed Sharifian
sharifian_s@aut.ac.ir

[1] Department of Electrical and Electronics Engineering, Amirkabir University of Technology, Tehran, Iran

1 Introduction

Cloud computing is one of the most popular buzzwords in today's enterprise [1]. Cloud computing brings modern technologies together to provide a vast range of service types for diverse users. Cloud computing is the use of cloud resources (hardware and software) that are delivered as a service over a network (typically the Internet) [2]. Cloud provides a managed pool of resources which includes storage, processing power and software services [3]. As a key feature of Cloud computing, consumers only pay for the services used and as offered by cloud providers that intelligently provide computing capabilities to quickly increase or decrease computing power as business needs change [4]. The research and development community has quickly reached consensus on core concepts of Cloud computing such as on-demand computing, elastic scaling, elimination of up-front capital and operational expenses, and establishing pay-as-you-go business model for information technology services [5]. A cloud typically consists of multiple resources possibly distributed and heterogeneous [6]. However, capacity management and demand prediction in cloud environments, where applications have variable and dynamic needs, are especially complicated and consequently resource management in Cloud computing is one of the most important challenges [7].

Considering various goals that sometimes are contradicted with each other makes the resource management problem in cloud data centers a challenging issue which needs tuning some trade-offs between targets [8]. The two most important requirements that have to be considered for resource allocation in cloud environments are energy consumption and Service Level Agreements (SLA) fulfillment. The SLA is an agreement that specifies the quality of service (QoS) between a service provider and service consumer, and usually includes the service price, with the level of QoS adjusted by the price of the service [9]. On the other hand, continuous increase in energy consumption of modern data centers raises a great concern for both governments and service providers [8]. Apart from the overwhelming operating costs and the total cost of acquisition (TCA) caused by high energy consumption, another concern is the environmental impact in terms of carbon dioxide (CO_2) emissions [10].

Due to the heterogeneity of cloud resources, and also the fact that the cloud users may have sporadic and dynamic resource usage, the cloud environment is highly dynamic [8]. However, virtualization technology which is the platform of Cloud computing facilitates the process of resource management in cloud environments. Virtualization is an important feature of Cloud computing that allows providing multiple Virtual Machines (VMs) on a single physical machine as well as migration of VMs [11]. Various kinds of applications running simultaneously on a Physical Machine (PM) have different resource requirements which lead to variable workloads on PMs. Hence, it is prevalent that VMs do not consume their maximum amount of resources all the time. Therefore, the actual resource utilization is much less than the installed capacity in a data centre. On the other hand, PMs consume their near maximum power consumption when they are idle. Therefore, consolidation of VMs on the least possible PMs and switching idle PMs off is the most novel method to save energy which is made true by the arisen of live migration technology in which VMs can be reallocated in an on-line manner [12–15]. However, the obligation of providing high quality of ser-

vice to cloud customers leads to the necessity in dealing with the energy-performance trade-off, as aggressive consolidations may lead to performance degradation [12].

Consolidation is on-line optimization of VMs placements on PMs for progressive online resource allocation in cloud data centers. The basic online consolidation problem in cloud data centers is divided into four parts [12]: (1) distinguishing the time that a host is overloaded; (2) distinguishing the time that a host is underloaded; (3) selecting VMs for migration from overloaded hosts; and (4) VM placement for the VMs that are selected for migration. This paper focuses on the first and the third phases and proposes novel heuristics for them.

One criticism of much of the literature on resource management problem in cloud environments is that they focus on CPU as the main system parameter and develop their models and algorithms based on only the CPU consumption rate. However, ignoring other important system parameters such as RAM and network bandwidth leads to wrong allocations. For instance, a PM could be idle in terms of CPU but overloaded in terms of memory for example, leading a wrong decision. Besides, modern multi-core processors are much more power-efficient than previous generations, whereas memory technology does not show any significant improvements in energy efficiency [16]. The increased number of cores in servers combined with the rapid adoption of virtualization technologies creates the ever growing demand to memory and makes memory one of the most important components of focus in the power and energy usage optimization [17]. The same condition can be applied to disk storage and network devices in modern cloud data centers. These facts unveil that it is essential to take into account the usage of multiple system resources in the energy-aware resource management [16]. The main contributions of this paper are proposing Window Moving Average (WMA) policy for determination of overloaded PMs and Multi-criteria TOPSIS with Prediction VM Selection (MTPVS) policy for VM selection from overloaded PMs. WMA and MTPVS investigate energy-performance efficient solutions by considering important input resource parameters including CPU, RAM, and network bandwidth. MPTVS takes advantage of a multi-criteria algorithms based on a modified version of the Technique for Order of Preference by Similarity to Ideal Solution (TOPSIS) [18] for VM selection in cloud data centers which not only eliminates the hotspot quickly, but also minimizes the SLA violations due to VM migrations. Besides, MTPVS selects VMs based on their predicted resource capacities rather than their current utilizations, which notably improves the output results.

The main contributions of this paper are:

- Proposing a novel multi-criteria VM selection method namely Multi-criteria TOPSIS with Prediction VM Selection (MTPVS) policy that selects the VMs to be migrated from overloaded PMs to both eliminating the hotspots quickly and minimizing the SLA violations due to VM migrations.
- Proposing Window Moving Average Policy (WMA) for detection of overloaded PMs that considers all input criteria including CPU, RAM, and network bandwidth in decision process and reduces the occurrence of VMs' migrations caused by instantaneous load peaks.
- Considering all important parameters as well as their weights in MTPVS policy.

- Proposing a simple and functional mechanism to compute weights of different resource types.

This paper begins by reviewing related works in Sect. 2. It will then goes on to describe the input parameters which are considered in resource management problem in Sect. 3. Section 4 presents our system model. Section 5 presents our proposed resource management policies. Section 6 assesses the applicability of our proposed solutions using CloudSim simulator. Finally, our concluding remarks and future directions are presented in Sect. 7.

2 Motivation and related work

There is a wide area of research that addresses the consolidation solution for energy and performance management in large scaled cloud data centers in which the workload is consolidated on a minimum number of PMs and the idle PMs are switched off. The main targets for comparing the efficiency of the algorithms are energy consumption by physical nodes and SLA violations, however, these targets are typically negatively correlated as energy can usually be decreased by the cost of the increased level of SLA violations [12]. In other words, energy consumption and SLA violation have an intrinsic trade-off which requires meticulous resource management algorithms to simultaneously minimize them.

The first work that have investigated large-scale virtualized data centers has been proposed in [19]. In addition to the hardware scaling and VMs consolidation, they have proposed a new power management method for virtualized systems called "soft resource scaling". In addition, they have suggested dividing the resource management problem into local and global levels. In the local level, the algorithms monitor power management of guest VMs. On the other hand, global policies coordinate multiple physical machines. In this paper, the target system is heterogeneous, the workload used to validate the system is arbitrary, and the goal of the proposed model is minimizing energy consumption as well as satisfying performance requirements.

The authors in [12] have conducted competitive analysis and proved competitive ratios of optimal online deterministic algorithms for the single VM migration and dynamic VM consolidation problems. They have divided the problem of dynamic VM consolidation into four parts for the first time: (1) determining when a host is considered as being overloaded; (2) determining when a host is considered as being underloaded; (3) selection of VMs that should be migrated from an overloaded host; and (4) finding a new placement of the VMs selected for migration from the overloaded and underloaded hosts. They have proposed novel adaptive heuristics for all parts. They have used Power Aware Best Fit Decreasing (PABFD) algorithm to solve resource allocation problem in the fourth part which is similar to MBFD policy that they adopted in their previous work [16].

The authors in [8] have proposed Enhanced Optimization (EO) policy as a novel resource management procedure in cloud data centers. The main idea behind EO policy is solving the resource allocation problem for the VMs that are selected to be migrated from either overloaded or underloaded PMs in one step rather than in separate steps for each one. Besides, they have introduced a solution based on Technique for Order of

Preference by Similarity to Ideal Solution (TOPSIS) for optimizing different targets in cloud data centers at the same time including energy consumption, SLA violation, and number of VM migrations. Based on this idea, they have proposed TOPSIS Power and SLA Aware Allocation (TPSA) and TOPSIS-Available Capacity-Number of VMs-Migration Delay (TACND) policies as novel multi-criteria algorithms for resource allocation and determination of underloaded PMs in cloud data centers, respectively.

The authors in [6] have presented two energy-conscious task consolidation heuristics, which aim to maximize resource utilization and explicitly take into account both active and idle energy consumption. Their heuristics assign each task to the resource on which the energy consumption for executing the task is explicitly or implicitly minimized without the performance degradation of that task. They have considered that CPU utilization directly relates to energy consumption and based on this assumption they have developed two energy-conscious task consolidation heuristics.

The authors in [20] have presented an efficient cloud resource provisioning approach, which is beneficial for the Software as a Service (SaaS) users, SaaS provider and cloud resource provider. They have modeled a cloud ecosystem in which the SaaS provider leases resources from cloud providers and also leases software as services to SaaS users. The SaaS providers aim at minimizing the payment of using VMs from cloud providers, and want to maximize the profit earned through serving the SaaS users' requests. The cloud provider is to maximize the profit without exceeding the upper bound of energy consumption of cloud provider for provisioning VMs to SaaS provider. Their proposed optimal cloud resource provisioning algorithm includes two sub-algorithms at different levels: interaction between the SaaS user and SaaS provider at the application layer and interaction between the SaaS provider and cloud resource provider at the resource layer.

The authors in [11] have proposed efficient consolidation algorithms which can reduce energy consumption and at the same time the SLA violations in some cases. They have introduced an efficient SLA-aware resource allocation algorithm that considers the trade-off between energy consumption and performance. Their proposed resource allocation algorithm takes into account both host utilization and correlation between the resources of a VM with the VMs present on the host. Moreover, they have proposed a novel algorithm for determination of underloaded PMs in the process of resource management in cloud data centers considering host CPU utilization and number of VMs on the host.

The authors in [21] have investigated the problem of power- and performance-efficient resource management in virtualized data center environments. The goal of this paper is to maximize the resource provider's revenue by minimizing power consumption and SLA violation simultaneously. They have addressed the resource management problem using a sequential optimization model and proposed solutions using a limited look-ahead control to estimate future system states over a prediction slot by the help of Kalman filter. They have explored a heterogeneous environment and their considered workload is arbitrary. Decision goals to be optimized are the following: the number of VMs to be provisioned for each service; the CPU share allocated to each VM; the number of servers to switch on or off; and a fraction of the incoming workload to distribute across the servers hosting each service.

The authors in [7] have proposed performance analysis based resource allocation scheme for the efficient allocation of virtual machines on the cloud infrastructure. They have proposed an efficient algorithm that follows a best fit strategy for allocation of virtual machine requests to the physical host nodes. To achieve this, they have designed a performance analysis scheme of each host node considering the number of cores and specification of CPU and memory size.

The authors in [22] have explored the problem of dynamic placement of applications in virtualized systems. Their goal is to minimize the power consumption while meeting the requested SLA. The proposed solution contains three managers and an arbitrator. The arbiter coordinates managers' actions and makes allocation decisions. Performance manager gathers applications information and resize VMs according to current resource requirements and the SLA. Power manager handles hardware power states and applies DVFS when it is necessary. Migration manager coordinates live migration of VMs. The considered target system is heterogeneous and the proposed model is for arbitrary workloads.

The authors in [16] have proposed an architectural framework and principles for energy-efficient Cloud computing aimed at the development of energy-efficient provisioning of cloud resources, while meeting QoS requirements defined by the SLA. They divided the VM allocation problem in two parts: the first part is the admission of new requests for VM provisioning and placing the VMs on hosts, whereas the second part is the optimization of the current VM allocation. They have modeled the first part as a bin packing problem and solved it by Modified Best Fit Decreasing (MBFD) algorithm in which they first sort all VMs in decreasing order of their current CPU utilizations, and allocate each VM to a host that provides the least increase of power consumption due to this allocation. Moreover, they have stated that the optimization of the current VM allocation is carried out in two steps: at the first step they select VMs that need to be migrated, at the second step the chosen VMs are placed on the hosts using the MBFD algorithm.

In sum, the main drawback of all the aforementioned studies is lack of ability to handle multiple system resources apart from CPU. However, this study not only considers all important criteria, but also proposes novel algorithms for simultaneous application of the predicted value of input criteria in decision process which notably improves the output results.

3 Input system parameters

Since our target data center is heterogeneous, all effective system parameters should be taken into consideration in decision making process. In our model, a server can be overloaded with respect to one or more of system's parameters. In other words, it is viable that while a server is over utilized regarding to one specific parameter, the utilization of other system parameters be normal. For instance, network interface may become the bottleneck of the system when there are some network-intensive virtual machine operations that concurrently transmit large data. Consequently, to balance overall system response time, all important parameters should be considered. Six major parameters, considered in this study in decision making process, are listed in Table 1. C_{CPU} specifies the computational power of machines which is determined

Table 1 Input parameters for resource management

No.	Parameter	Description	Unit
1	C_{CPU}	CPU clock speed multiplied by the number of CPU cores	MIPS
2	C_{RAM}	RAM capacity	GB
3	C_{NET}	Network bandwidth	Gbps
4	P_{CPU}	Division of requested CPU MIPS of a VM by the available CPU MIPS in a PM	%
5	P_{RAM}	Division of requested RAM capacity of a VM by the available RAM capacity in a PM	%
6	P_{NET}	Division of requested network bandwidth of a VM by the available network bandwidth in a PM	%

as CPU clock speed multiplied by the number of CPU cores defined in MIPS. C_{RAM} defines the capacity of RAM. C_{NET} symbolizes capacity of network bandwidth which determines the amount of data that can pass through a network interface per unit of time. P_{CPU} is the percentage of CPU utilization that is computed by dividing the requested CPU of a VM by the available CPU capacity in a PM. P_{RAM} is the percentage of RAM utilization that is computed by dividing the requested RAM capacity of a VM by the available RAM capacity in a PM. P_{NET} is the percentage of network bandwidth utilization that is computed by dividing requested network bandwidth of a VM by the available network bandwidth in a PM.

4 System model

The target system consists of data centers with heterogeneous resources which are hosts of various users with different applications who want to run multiple heterogeneous VMs on data center nodes, resulting in a dynamic mixed workload on each PM. VMs and PMs are characterized with CPU computation power defined in Millions Instructions Per Second (MIPS), RAM, Disk capacity, and Network bandwidth. The target system model is shown in Fig. 1. This model is defined in [8] and is a modified version of the model described in [8,12]. The central manager is the resource manager for the whole data center that manages resource distribution among VMs in the data center. In addition, it resizes VMs according to their resource needs, and decides when and which VMs should be migrated from PMs. The agents which are implemented in hypervisors are connected to the central manager through network interfaces and have responsibility for monitoring PMs as well as sending gathered information to the central manager. Hypervisor performs actual resizing and migration of VMs as well as changes in power modes of the PMs. Our system model includes two important parts: a central manager similar to the global manager defined in [12] and the agents similar to the local manager defined in [12]. The main difference between our model and the one proposed in [12] is that both the decision on VMs resizing and also the decision on when and which VMs should be migrated are made in central manager rather than

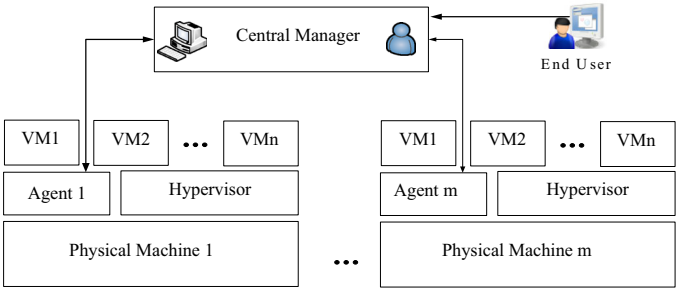

Fig. 1 System model [8]

in agents which results in having a more holistic view in the decision making process. However, if the central manager runs on a single PM and that PM fails, there is no fault-tolerant policy. Therefore, we propose running the central manager on a VM instead of a PM and use FT (fault tolerance) and HA (High Available) capabilities which are possible; thanks to virtualization technology.

4.1 Consolidation procedure

This study adopts the resource management procedure defined in [12] which divides the process of on-line consolidation problem in cloud data centers into four main phases. Algorithm 1 depicts the consolidation procedure based on these four phases.

First, PMs are searched one by one to find overloaded PMs until there is no more hotspot. Resource utilization values of each PM are predicted based on the resource utilization history of PMs, using the proposed prediction algorithm. If the prediction algorithm forecasts for a PM that its utilization will become more than 100 %, then this PM is determined as an overloaded PM. After that, some VMs residing on overloaded PMs are selected for migration based on the proposed policy for VM selection from overloaded PMs until elimination of hot spots. In the following step, selected VMs are categorized based on their CPU utilization. Then, a resource allocation procedure is executed for the sorted VMs to find their migration destination using PABFD allocation policy [12,16]. PABFD policy finds the PM that both have enough resource to host the VM as well as the least power increase after allocation of a VM. If the control system finds a proper destination for a VM, then the couple of VM and its new host are added to the migration map.

Following that, underloaded PMs are determined. In this step, overloaded PMs, switched off PMs, and the PMs that are to be the migration destination in the migration map are excluded from the searching list of underutilized PMs. Moreover, overloaded PMs, and switched off PMs are excluded from the list of PMs for finding new VM placement. In each searching step to find underloaded PMs, the defined policy for determination of underloaded PMs is executed and a PM is selected as a candidate of being underloaded. VMs from underloaded PMs are added to the migration list until the controlling system cannot find any underloaded PM. In the following step, selected VMs from underloaded PMs are sorted based on their CPU utilization. If the control system can find proper PMs as probable migration destinations for all the

VMs residing on an underloaded PM using PABFD policy, then all its VMs and their founded hosts are added to the VM's migration list. Otherwise, none of the VMs are added to the VM's migration list. Finally, the migration process is initiated based on the final migration map.

Algorithm 1: Consolidation Procedure.

```
1  Input: all PMs and VMs.
2  Output: migrationMap.
3  overUtilizedHosts = getOverUtilizedHosts(getHostList());
4  vmsToMigrate = getVmsToMigrateFromOverUtilizedHosts(overUtilizedHosts);
5  sortedVmsToMigrate = sortVmsToMigrate(vmsToMigrate);
6  migrationMap = getNewVmPlacement(sortedVmsToMigrate, overUtilizedHosts);
7  excludedHostsForFindingUnderUtilizedHost.addAll(overUtilizedHosts);
8  excludedHostsForFindingUnderUtilizedHost.addAll(switchedOffHosts);
9  excludedHostsForFindingUnderUtilizedHost.addAll(extractHostListFromMigrationMap(migrationMap));
10 excludedHostsForFindingNewVmPlacement.addAll(overUtilizedHosts);
11 excludedHostsForFindingNewVmPlacement.addAll(switchedOffHosts);
12 While (true) {
13  if (numberOfHosts == excludedHostsForFindingUnderUtilizedHost.size())
14    break;
15  underUtilizedHost = getUnderUtilizedHost(excludedHostsForFindingUnderUtilizedHost);
16  if (underUtilizedHost == null)
17    break;
18  excludedHostsForFindingUnderUtilizedHost.add(underUtilizedHost);
19  excludedHostsForFindingNewVmPlacement.add(underUtilizedHost);
20 vmsToMigrateFromUnderUtilizedHost=getVmsToMigrateFromUnderUtilizedHost(underUtilizedHost);
21  if (vmsToMigrateFromUnderUtilizedHost.isEmpty())
22   continue;
23 sortedVmsToMigrateFromUnderUtilizedHost=sortVmsToMigrate(vmsToMigrateFromUnderUtilizedHost);
24 newVmPlacement=getNewVmPlacementFromUnderUtilizedHost(vmsToMigrateFromUnderUtilizedHost,exc
   ludedHostsForFindingNewVmPlacement);
25 excludedHostsForFindingUnderUtilizedHost.addAll(extractHostListFromMigrationMap(newVmPlacement));
26 migrationMap.addAll(newVmPlacementFromUnderUtilizedHosts);
27 }end
28 return migrationMap;
```

4.2 Power and energy models

Traditionally, recent studies [16,21] have subscribed to the belief that power consumption by servers can be approximated by a linear relationship with CPU utilization. This approximation comes from the idea that CPU is the major power consumer in a data center. A serious weakness with this argument, however, is that by introducing multi-core CPUs with modern power management techniques, as well as utilization of virtualization technique, CPU is not the only major power consumer in data centers anymore [12]. This fact combined with the difficulty of modeling power consumption in modern data centers, makes building precise analytical models a complex research problem [12]. Hence, instead of using a complex analytical model for power consumption of a server, we utilize real data on power consumption provided by the results of the SPECpower benchmark [6]. Table 2 shows the power consumption of the servers used in this study which is provided in [12].

In addition, energy consumption is modeled as the summation of power consumed during a period of time according to Eq. (1) which is widely used in the literature such as [12,16].

$$E(t) = \int_t P(t)\,\mathrm{d}t \tag{1}$$

Table 2 Power consumption of considered servers for different loads (kW) [12]

Server	Idle	10 %	20 %	30 %	40 %	50 %	60 %	70 %	80 %	90 %	100 %
HP ProLiant G3	105	112	118	125	131	137	147	153	157	164	169
HP ProLiant G4	86	89.4	92.6	96	99.5	102	106	108	112	114	117
HP ProLiant G5	93.7	97	101	105	110	116	121	125	129	133	135
IBM Server x3250	41.6	46.7	52.3	57.9	65.4	73	80.7	89.5	99.6	105	113

4.3 SLA violation metrics

Quality of service requirements are commonly formalized in the form of SLAs, which can be determined in terms of such characteristics as minimum throughput or maximum response time delivered by the deployed system [12]. As these characteristics can vary for different applications, it is necessary to define a workload independent metric that can be used to evaluate the SLA delivered to any VM deployed in an Infrastructure as a Service (IaaS) such as OTF (Overload Time Fraction) metric defined in [13]. In this study, we use a modified version of SLA Violation (SLAV) metric introduced in [12] as defined in Eq. (2) which is composed of multiplication of two metrics: the SLA violation time per active host (SLATAH) and performance degradation due to migration (PDM) as defined in Eq. (3).

$$\text{SLAV} = \text{SLATAH} \times \text{PDM} \tag{2}$$

$$\text{SLATAH} = \frac{1}{N}\sum_{i=1}^{N}\frac{T_{\text{S}_i}}{T_{\text{a}_i}}, \quad \text{PDM} = \frac{1}{M}\sum_{j=1}^{M}\frac{C_{\text{d}_j}}{C_{\text{r}_j}} \tag{3}$$

In the default SLATAH metric defined in [12], T_{s_i} is the total time during which the host i has experienced the utilization of 100 %; however, we define T_{s_i} as the total time during which allocated resource to the VMs is lower than their requested resource; This amendment to the default SLATAH metric is because that it is highly probable that a PM experience the utilization of 100 % but at the same time all the VMs receive their required resources leading to no SLA violations. In addition, it is probable that although a PM is not experiencing the utilization of 100 %, but the allocated resource to a VM be less than its requested resource. T_{a_i} is the total time during which the host i has been in the active state; N is the number of PMs; C_{d_j} is the estimate of the performance degradation of the VM_j caused by migrations which is estimated as 10 % of the average CPU utilization in MIPS during all migrations of the VM_j; C_{r_j} is the total CPU capacity requested by the VM_j during its lifetime; and M is the number of VMs.

5 Proposed policy for determination of overloaded PMs

In this section we present Window Moving Average (WMA) policy as a proposed heuristic for determination of overloaded PMs which is one of the most important resource management sub problems for consolidation of cloud data centers. One of the key advantages of WMA policy is that it considers all important system's criteria

including CPU, RAM, and network bandwidth in determination of overloaded PMs. Therefore, it is conscious of overutilization regarding all important system resource types.

5.1 Window moving average (WMA) policy

Window moving average predicts the resource utilization for CPU, RAM, and network bandwidth based on their saved utilization history. WMA policy is a modified version of Moving Average algorithm, a well-known time series prediction technique that is also used as a type of finite impulse response filter [23]. The performance of this technique is widely evaluated in literature such as [23,24]. Moving Average model is also a special case of general ARIMA model [25]. The default Moving Average technique basically builds a linear model for forecasting using the current values. But, what happens if there is some noise present in the given time series. The aim of Window Moving Average (WMA) policy is basically elimination of the noise or sudden spikes present in the resource utilization data of time series. WMA policy takes noise into account when forecasting the data and reduces the effect of both noise and sudden spikes by computing the average of two separate time interval windows. More precisely, instead of considering only one recent value of utilization as the new value of time series, average of recent values in a specific window size of time series is considered as an estimation of the new value. Likewise, the average of old values in a specific window size of time series is considered as an estimation of the old value. Finally, similar to the default moving average technique, the final predicted utilization value for CPU, RAM, and network bandwidth are computed by a combination of the estimation of the old value and the estimation of the new value using Eqs. (4), (5), and (6).

$$\hat{U}_{\text{CPU}} = k \times \frac{\sum_{i \in \text{Window 1}} U^i_{\text{CPU}}}{\text{Size (Window 1)}} + (1-k) \times \frac{\sum_{i \in \text{Window 2}} U^i_{\text{CPU}}}{\text{Size (Window 2)}} \tag{4}$$

$$\hat{U}_{\text{RAM}} = k \times \frac{\sum_{i \in \text{Window 1}} U^i_{\text{RAM}}}{\text{Size (Window 1)}} + (1-k) \times \frac{\sum_{i \in \text{Window 2}} U^i_{\text{RAM}}}{\text{Size (Window 2)}} \tag{5}$$

$$\hat{U}_{\text{NET}} = k \times \frac{\sum_{i \in \text{Window 1}} U^i_{\text{NET}}}{\text{Size (Window 1)}} + (1-k) \times \frac{\sum_{i \in \text{Window 2}} U^i_{\text{NET}}}{\text{Size (Window 2)}} \tag{6}$$

where $\hat{U}_{\text{CPU}}$, $\hat{U}_{\text{RAM}}$, and $\hat{U}_{\text{NET}}$ are the predicted utilization of CPU, RAM, and network bandwidth, respectively; the coefficient k has the same function as the constant defined in familiar moving average algorithm. More precisely, k is a coefficient that specifies the weight of the estimation of the recent samples and the estimation of the old ones on the predicted utilization of a specific resource type; and U^i_{CPU}, U^i_{RAM}, and U^i_{NET} are the ith utilization value of CPU, RAM, and network bandwidth which are saved in the history, respectively. WMA policy considers multi-criteria including CPU, RAM, and network bandwidth. Therefore, if the WMA policy forecasts for a PM that utilization of either one of its resource types ($\hat{U}_{\text{CPU}}$, $\hat{U}_{\text{RAM}}$, or $\hat{U}_{\text{NET}}$) will be more than 100 %, then this PM is determined to be overloaded.

6 Proposed policies for VM selection from overloaded PMs

In this section we present our proposed policies for VM selection from overloaded PMs including Maximum Requested Resource (MRR), Minimum Downtime Migration (MDM), and Multi-Criteria TOPSIS with Prediction VM Selection (MTPVS). It is important to note that all of these policies take advantage WMA policy for prediction of resource utilizations. More precisely, one of the main strength of proposed policies compared with state of the arts is that they select VMs for migration based on their predicted utilizations rather than their current utilizations.

6.1 Maximum requested resource (MRR) policy

Maximum requested resource addresses the problem of minimizing SLA violation as well as number of VM migrations as depicted in algorithm 2. Since it is considered that our target system faces mostly with CPU shortage, this policy selects VMs based on their predicted CPU capacity request. This policy repeats the selection of a VM with the highest predicted requested CPU capacity until the elimination of hotspot. Due to selection of a VM with the highest requested CPU capacity, MRR quickly eliminates the hotspot occurred by CPU shortage in a PM which decreases the SLA violation. Besides, since this policy eliminates the hotspot with scheduling the migration of lower number of VMs that have higher CPU capacities instead of migration of large number of VMs with less CPU capacity, the number of VMs migrations decreases.

Algorithm 2: MRR policy for VM selection

```
1  Input: Candidate VMs to be selected for migration from overloaded PMs.
2  Output: Selected VMs to be migrated.
3  Maximum Requested CPU capacity = Min Value.
4  While(hotspot exist) {
5    For all the VMs hosting on the overloaded PM do{
6      If (Predicted Requested CPU capacity of this VM > Maximum Requested CPU capacity){
7        Maximum Requested CPU capacity = Requested CPU capacity of this VM.
8        Selected VM = this VM.
9      }end
10   }end
11   VMList.add(Selected VM).
12 }end
13 Return the VMList
```

6.2 Minimum downtime migration (MDM) policy

Live-migration is one of the key enablers of resource management in cloud data centers. A live-migration instance usually takes a few seconds to a few minutes to complete. Among all procedures for live-migration, memory content transmission takes the longest time and thus most affects the migration performance [26]. In order to be effective, a live-migration technique should finish the migration process as fast as possible while minimizing the QoS degradations in the migrated VMs. Three prevalent approaches for transferring memory contents used for VM migration are stop-and-

copy, pre-copy, and post-copy migration schemes. A stop-and-copy migration method transmits all memory contents before resuming the migrated VM on the destination PM. A pre-copy approach iteratively transfers modified pages to the destination PM and enables the VM in migration to suspend only for transferring a small number of modified pages after copying the entire memory region to the destination node [26]. In the post-copy migration scheme, the hypervisor sends only the minimal and essential memory contents and information of a migrating VM to the destination and resumes the execution of the VM [26]. In this approach when the migrated VM needs to access untransferred pages, a page fault occurs. In response to the page fault, the hypervisor in the destination node makes requests of the missing pages to the source node, and in return the source node hypervisor transmits them to the destination [26]. Since famous hypervisors such as Xen utilize pre-copy scheme for live VM migration, which allows migrating an OS with near-zero downtime, a pre-copy approach is implemented in this paper similar to [12]. Moreover, the migration time is estimated as the amount of RAM utilized by the VM divided by the spare network bandwidth available for the host. In all these migration techniques, two important parameters that affect both the downtime of the migrating VM and the migration time are amount of transferred memory as well as the available network bandwidth in source and destination. Hence, as depicted in algorithm 3, the major goal of MDM is simultaneous minimization of the total downtime occurred during migration process as well as migration duration. Therefore, MDM considers RAM, and network bandwidth parameters in decision making process and selects a VM with the lowest migration delay from an overloaded PM. The main difference between MDM and Minimum Migration Time (MMT) policy proposed in [12] is that MDM makes decision based on the predicted capacity of all resource types instead of the current CPU utilization.

Algorithm 3: MDM policy for VM selection

Input: Candidate VMs to be selected for migration from overloaded PMs.
Output: Selected VMs to be migrated.
Minimum Migration Delay = Max Value.
While(hotspot exist){
  For all the VMs hosting on the overloaded PM do{
    $\text{Migration Delay}_{VM} = \frac{\text{PredictedRAM}_{VM}}{\text{PredictedAvailable NetworkBandwidthof the PM}}$
    If (Migration Delay of this VM < Minimum Delay){
      Minimum Delay = Migration Delay of this VM.
      Selected VM = this VM.
    }end
  }end
  VMList.add(Selected VM).
}end
Return the VMList.

6.3 Multi-criteria TOPSIS with prediction VM selection (MTPVS) policy

Multi-criteria TOPSIS with prediction VM selection is proposed for multi-criteria VM selection from overloaded PMs in consolidation process of cloud data centers.

MTPVS takes advantage of a modified version of the Technique for Order of Preference by Similarity to Ideal Solution (TOPSIS) [18] as a multi-criteria decision making algorithm. MTPVS simultaneously apply the idea proposed in MRR and MDM policies by selecting the VMs that both have the highest predicted requested resource and at the same time have the minimum predicted migration delay. By doing so, not only the hotspot is eliminated quickly with less SLA violation and lower number of VM migrations, but also the SLA violations due to migration process is minimized. Besides, MTPVS selects VMs for migration based on the predicted utilizations of their requested resource utilization using WMA prediction method rather than superficially based on the current CPU utilizations.

MTPVS is a multiple parameter method to identify solutions from a finite set of alternatives based upon simultaneous distance minimization from an ideal point and distance maximization from a nadir point [27]. More precisely, the chosen VM should have the shortest distance from the ideal positive point (VM^{+}) and the farthest distance from the negative ideal point (VM^{-}). VM^{+} and VM^{-} are formed as composite of best and worst values of different system parameters, respectively, among all the VMs. Distance from each of these poles are measured in the Euclidean distance.

All the predicted information assigned to the virtual machines in time slot t form a Decision Matrix $\overrightarrow{\mathrm{DM}}$ as shown in Eq. (7).

$$\overrightarrow{\mathrm{DM}} = \begin{bmatrix} P_{\mathrm{cpu}}^{\mathrm{VM}^1} & P_{\mathrm{ram}}^{VM^1} & P_{\mathrm{net}}^{\mathrm{VM}^1} & C_{\mathrm{cpu}}^{\mathrm{VM}^1} & C_{\mathrm{ram}}^{\mathrm{VM}^1} & C_{\mathrm{net}}^{\mathrm{VM}^1} \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ P_{\mathrm{cpu}}^{\mathrm{VM}^i} & P_{\mathrm{ram}}^{\mathrm{VM}^i} & P_{\mathrm{net}}^{\mathrm{VM}^i} & C_{\mathrm{cpu}}^{\mathrm{VM}^i} & C_{\mathrm{ram}}^{\mathrm{VM}^i} & C_{\mathrm{net}}^{\mathrm{VM}^i} \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ P_{\mathrm{cpu}}^{\mathrm{VM}^m} & P_{\mathrm{ram}}^{\mathrm{VM}^m} & P_{\mathrm{net}}^{\mathrm{VM}^m} & C_{\mathrm{cpu}}^{\mathrm{VM}^m} & C_{\mathrm{ram}}^{\mathrm{VM}^m} & C_{\mathrm{net}}^{\mathrm{VM}^m} \end{bmatrix} \tag{7}$$

Where $\mathrm{VM}^1, \mathrm{VM}^2, \ldots, \mathrm{VM}^m$ are the VMs that MTPVS is to sort them; $P_{\mathrm{res}}^{VM^j}$ is the resource utilization of jth VM in percent; $C_{\mathrm{res}}^{\mathrm{VM}^j}$ is the resource capacity of jth VM; and res can be CPU, RAM, or network bandwidth. $\overrightarrow{DM}$ is the decision matrix which consists of alternatives and criteria. Alternatives are $\mathrm{VM}^1, \mathrm{VM}^2, \ldots, \mathrm{VM}^m$, and criteria are the parameters defined in Table 1. Each value of $\overrightarrow{DM}$ matrix indicates the rating of a specific alternative according to one criterion.

To select the best VM for migration we go through the following steps:

Step 1: The data of the decision matrix $\overrightarrow{DM}$ come from different sources, so it is necessary to normalize it in order to transform it into a dimensionless matrix, which allow the comparison of the various criteria. Therefore, we first normalize the decision matrix $\overrightarrow{DM}$ to have dimensionless decision matrix $\overrightarrow{\underline{DM}}$. The purpose of decision matrix normalization is to make matrix entries free of unit so that they can take part in our computations. Therefore, the decision matrix is made dimensionless by dividing each entry by maximum value of each column according to Eq. (8). After this step, the $\overrightarrow{\underline{DM}}$ matrix consists of normalized values which represent the relative ratings of the alternatives.

$$\overrightarrow{\underline{DM}} = \begin{bmatrix} \frac{P_{cpu}^{VM^1}}{P_{cpu}^{max}} & \frac{P_{ram}^{VM^1}}{P_{ram}^{max}} & \frac{P_{net}^{VM^1}}{P_{net}^{max}} & \frac{C_{cpu}^{VM^1}}{C_{cpu}^{max}} & \frac{C_{ram}^{VM^1}}{C_{ram}^{max}} & \frac{C_{net}^{VM^1}}{C_{net}^{max}} \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ \frac{P_{cpu}^{VM^i}}{P_{cpu}^{max}} & \frac{P_{ram}^{VM^i}}{P_{ram}^{max}} & \frac{P_{net}^{VM^i}}{P_{net}^{max}} & \frac{C_{cpu}^{VM^i}}{C_{cpu}^{max}} & \frac{C_{ram}^{VM^i}}{C_{ram}^{max}} & \frac{C_{net}^{VM^i}}{C_{net}^{max}} \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ \frac{P_{cpu}^{VM^m}}{P_{cpu}^{max}} & \frac{P_{ram}^{VM^m}}{P_{ram}^{max}} & \frac{P_{net}^{VM^m}}{P_{net}^{max}} & \frac{C_{cpu}^{VM^m}}{C_{cpu}^{max}} & \frac{C_{ram}^{VM^m}}{C_{ram}^{max}} & \frac{C_{net}^{VM^m}}{C_{net}^{max}} \end{bmatrix} \tag{8}$$

Step 2: In the next step, VM^+ and VM^- are determined. Before that, type of each attribute should be defined. In general, the criteria can be classified into two types: benefit and cost. The benefit criterion means that a higher value is better, while for the cost criterion is the opposite. In other words, larger values for a benefit type attribute leads to less distance from VM^+ and more distance from VM^-, while the opposite condition is hold for a cost type variable. Since we want to select a VM that has smaller data volume, RAM capacity is marked as cost type. In other words, the more memory dedicated to a virtual machine, the more cost we should pay for migration. Therefore, MTPVS algorithm searches for a VM that has the lowest memory to avoid transferring large data over interconnection network. However, CPU and NET parameters are considered to have benefit type. More precisely, MTPVS selects a VM with higher predicted CPU capacity to quickly eliminate the hotspot and minimize the SLA violation and number of VM migrations. Therefore, VM_{res}^+ and VM_{res}^- are defined using Eqs. (9) and (10) respectively.

$$VM_{res}^+ = \left\{ P_{cpu}^+, P_{ram}^-, P_{net}^+, C_{cpu}^+, C_{ram}^-, C_{net}^+ \right\} \tag{9}$$

$$VM_{res}^- = \left\{ P_{cpu}^-, P_{ram}^+, P_{net}^-, C_{cpu}^-, C_{ram}^+, C_{net}^- \right\} \tag{10}$$

Where P^+ and C^+ are the maximum values in each column of $\overrightarrow{\underline{DM}}$, and P^- and C^- are the minimum values in each column of $\overrightarrow{\underline{DM}}$ matrix.

Step 3: In this step, the score of each individual alternative regarding each criteria is computed based on its relative distance from ideal solutions (VM_{res}^+ and VM_{res}^-) to make comparisons possible. The relative distance for each resource type of a VM from VM_{res}^+ and VM_{res}^- are calculated using Eq. (11).

$$Score_{res}^{VM^j} = \frac{\sqrt{(VM_{res}^j - VM_{res}^-)^2}}{\sqrt{(VM_{res}^j - VM_{res}^-)^2} + \sqrt{(VM_{res}^j - VM_{res}^+)^2}} \tag{11}$$

Where $Score_{res}^{VM^j}$ shows the score of a specific resource type of jth VM, and res can be any of the parameters defined in Table 1. The more distance a VM has from VM^-, the more the value of nominator of Eq. (11) becomes and consequently the score value is larger. Similarly, the less distance a VM has from VM^+, the less the value of denominator of Eq. (11) becomes and accordingly the score value is larger.

Step 4: In this step, the individual score of each alternative regarding different criteria are combined to obtain an overall score for each alternative separately. In addition, in this step, the importance of each criterion is considered in the score computations by application of a weight for each criterion. Therefore, we compute the total score of a VM using Eq. (12).

$$\text{Score}(\text{VM}^j) = \sum_{\text{res}=1}^{\#\text{Res}} \text{Weight}_{\text{res}} \times \text{Score}_{\text{res}}^{\text{VM}^j} \tag{12}$$

where $\text{Score}(\text{VM}^j)$ is the average closeness of jth VM to the ideal solutions, $\text{Weight}_{\text{res}}$ is importance of each criterion of type res; res can be any of CPU, RAM, or network bandwidth; $\text{Weight}_{\text{res}}$ is computed using Eq. (13); and #Res is the number of considered resources.

Step 5: Rank the VMs according to their score and select the one with the highest score. The VM with the highest score has the maximum distance from VM^- and the minimum distance from VM^+.

6.3.1 Weight computation for different criteria

Different criteria considered in MTPVS policy have different importance in the final score. However, finding an optimized weight for different criteria is a wide research area by itself. In this study, we propose using a simple functional weighting procedure which computes the weights of each parameter based on the average utilization of all system resources in a data center according to Eq. (13). The idea behind the proposed equation is that the higher the utilization of a specific resource type, the more likely that the system confronted with hotspot along this resource type. Therefore, adoption of the proposed weighting equation results in selection of the VMs that eliminate the hotspot along this resource type faster.

$$\text{Weight}_{\text{Res}} = \frac{\bar{U}_{\text{Res}}(t)}{\sum_{\text{res}=1}^{\#\text{Res}} \bar{U}_{\text{res}}(t)} \tag{13}$$

where $\bar{U}_{\text{Res}}(t)$ is the average utilization of a specific resource in a data center at simulation time t, and #Res is the number of considered resources. Res can be either of CPU, RAM, or network bandwidth.

7 Performance evaluation

In this section, we discuss a performance evaluation of the heuristics presented in this paper. We compare our solutions with recent energy aware consolidation studies which are close to our study including [12] and [11] as benchmarks. Similar to our study, they consider the four phase resource management process introduced in [12].

Table 3 Configuration of servers

Server	CPU model	Cores	Frequency (MHz)	RAM (GB)
HP ProLiant G3	Intel Pentium D930	2	3000	4
HP ProLiant G4	Intel Xeon 3040	2	1860	4
HP ProLiant G5	Intel Xeon 3075	2	2660	4
IBM Server x3250	Intel Xeon 3470	4	2933	8

Table 4 VM types (four Amazon EC2 VM types) [4]

VM type	CPU (MIPS)	RAM (GB)
High-CPU medium instance	2500	0.85
Extra-large instance	2000	3.75
Small instance	1000	1.7
Micro instance	500	0.613

7.1 Experiment setup

Since our target system is a generic Cloud computing environment, it is vital to analyze it on a large-scale virtualized data center infrastructure. However, implementing and evaluating the proposed algorithms on such an infrastructure is very expensive and time-consuming. Moreover, executing repeatable large-scale experiments to analyze and compare the results of proposed algorithms is really hard. Therefore, we have used simulation for performance evaluation. We have utilized an extension of CloudSim toolkit [28] and its entire provided infrastructure as our simulation platform. Adopting CloudSim toolkit enables us to perform repeatable experiments on large-scale virtualized data centers. Besides, it is a modular and extensible open source toolkit which has built-in capability to implement and compare energy aware algorithms in cloud environments.

In our infrastructure setup which has real configurations, we have simulated a Cloud computing infrastructure comprising a data center with 800 installed heterogeneous physical machines including 200 HP ProLiant ML110 G3, 200 HP ProLiant ML110 G4, 200 HP ProLiant ML110 G5, and 200 IBM Server x3250. Characteristics of these machines are depicted in Table 3. Power consumptions of physical machines are computed based on the data described in Sect. 4.3. VMs are supposed to correspond to four Amazon EC2 VM types as shown in Table 4. Since using real workload for simulation experiments is important, we consider 10 days data of CoMon project [29]. This data contains CPU utilization in 5-min intervals of more than a thousand VMs that are located at more than 500 servers around the world (Table 5). During the simulations, each VM is randomly assigned a workload trace from one of the VMs from the corresponding day. WMA predicts future utilizations in which k is set to be 0.3; size of window 1 and window 2 are set to be $\frac{1}{3}$ and $\frac{2}{3}$ of history length respectively; and the history length is equal to 30.

Table 5 Workload data characteristics (CPU utilization) [29]

Date	Number of VMs	Mean (%)	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

7.2 Performance metrics

To make our results comparable with the algorithms presented by Beloglazov and Buyya, we consider ESV metric defined in [12] which is shown in Eq. (14). Moreover, to assess the simultaneous minimization of energy, SLA violation, and number of VMs' migrations, we use the metric defined in [8] which is shown in Eq. (15).

$$\text{ESV} = \text{Energy} \times \text{SLAV} \tag{14}$$

$$\text{ESM} = \text{ESV} \times \text{Migrations count} \tag{15}$$

7.3 Simulation results

The default on-line consolidation process in cloud data centers include four main phases [12]. In this section, a reference scenario consisting of a combination of the best policies reported in [12] for these phases including Local Regression (LR) for the first phase, a simple method (SM) for the second phase, Minimum Migration Time for the third phase (MMT), and Power Aware Best Fit Decreasing (PABFD) policy for the fourth phase is compared with the scenario described in [11] as well as with our proposed policies. In Sect. 7.3.1 the policies for determination of overloaded PMs are compared with each other; in Sect. 7.3.2 the policies for VM selection from overloaded PMs are compared; and in Sect. 7.3.3 the combination of best policies proposed in this study as well as in [12] and [11] are compared.

7.3.1 Evaluation of policies for determination of overloaded PMs

In this section we compare our proposed WMA policy with four other policies proposed in [12] including Local Regression (LR), Local Regression Robust (LRR), interquartile range (IQR), and Median Absolute Deviation (MAD). Ten experiments are executed separately for the 10 days of workloads depicted in Table 5 and their median results for energy consumption, SLA violation, number of VM migrations, execution time

Table 6 Output results for different policies for determination of overloaded PMs

Policy	LR	WMA	LRR	IQR	MAD
Energy consumption (Kwh)	46.03	44.49	46.56	46.62	46.41
SLAV ($\times 10^{-5}$)	74.628	21.83	72.462	104.349	67.832
ESV($\times 10^{-3}$)	36.686	10.23	32.873	52.198	33.654
Number of VM migrations	7005	3217	6598	9182	6413
ESM	257.78	32.98	217.58	479.18	215.90
Execution time (s)	0.02996	0.02950	0.02902	0.03768	0.03164

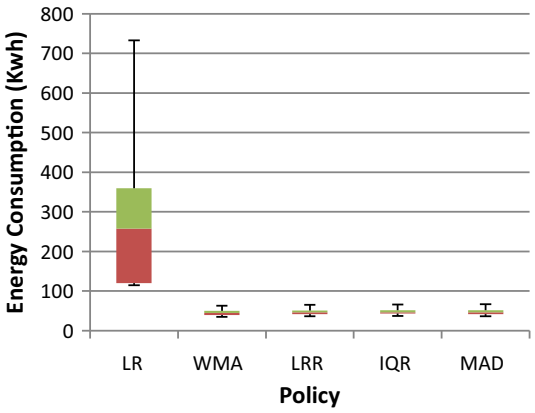

Fig. 2 Energy consumption of different policies for determination of overloaded PMs

as well as ESV and ESM metrics are reported in Table 6. Figure 2 shows the energy consumption; Fig. 3 shows the value of SLA violation; Fig. 4 depicts the value of ESV metric; Fig. 5 shows the overall number of VM migrations; Fig. 6 depicts the ESM metric; and Fig. 7 shows the median value for average execution time of the whole resource management process.

As depicted in Figs. 2, 3, and 5, the results for WMA policy regarding energy consumption, SLA violation, and number of VM migrations are prominently lower than other policies. Consequently, ESV and also ESM metrics for WMA policy are much less in comparison with LR, LRR, IQR, and MAD as shown in Figs. 4 and 6, respectively. More precisely, it can be inferred from Table 6 that adoption of WMA policy leads to 3.34, 70.74, 54.07, 72.11, and 87.2 % reductions in energy consumption, SLA violation, number of VM migrations, ESV metric, and ESM metric, respectively, in comparison with LR policy. This observation can be described by the fact that WMA policy both considers multiple resource types in decision process and also has more accurate predictions of the resource utilizations which notably improve the output results. In addition, it can be deduced from Fig. 7 that the execution times of all the evaluated policies are near each other.

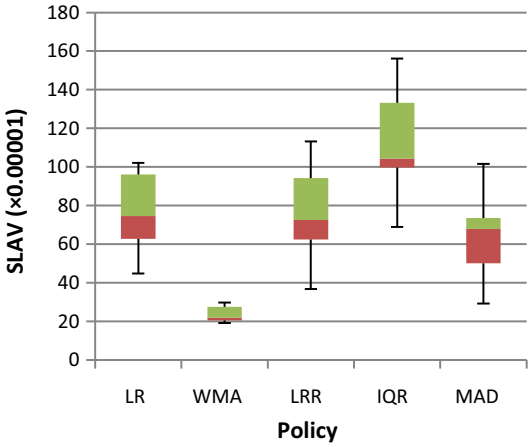

Fig. 3 SLA violation of different policies for determination of overloaded PMs

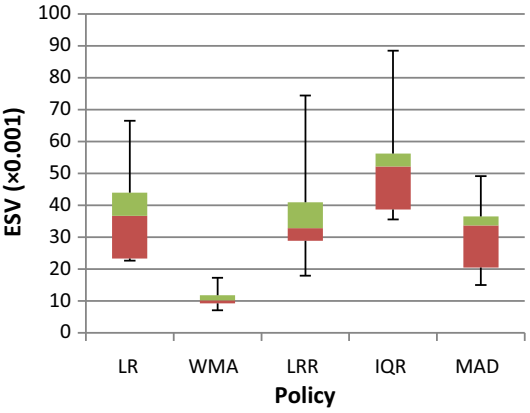

Fig. 4 ESV of different policies for determination of overloaded PMs

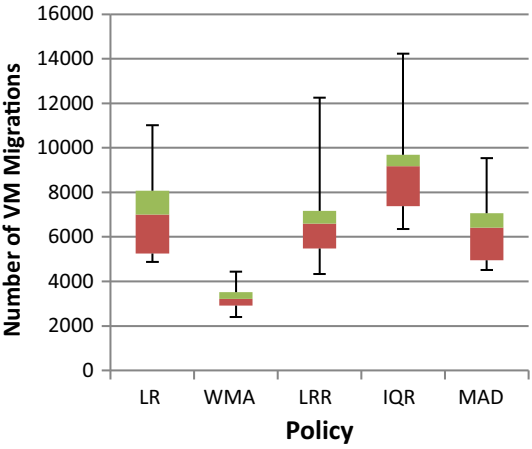

Fig. 5 Number of VM Migrations of different policies for determination of overloaded PMs

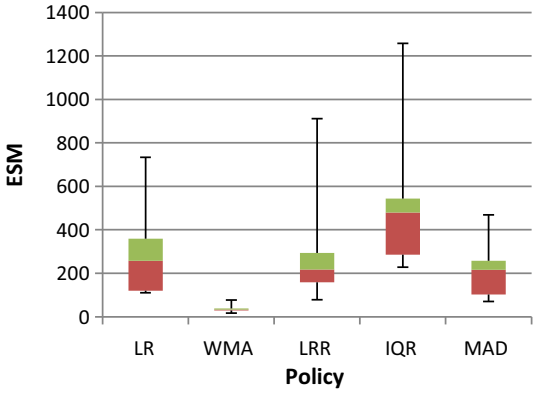

Fig. 6 ESM of different policies for determination of overloaded PMs

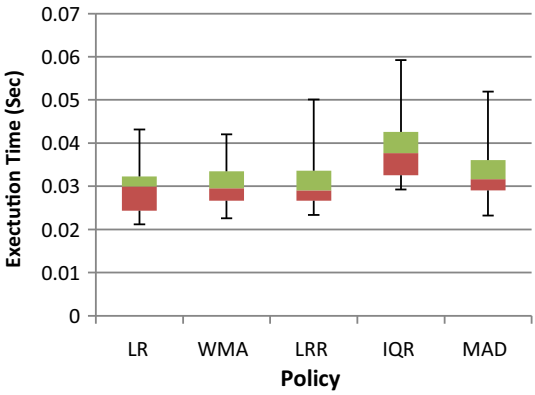

Fig. 7 Execution time of different policies for determination of overloaded PMs

7.3.2 Evaluation of proposed policies for VM selection from overloaded PMs

In this section we compare our proposed policies for selection of VMs from overloaded PMs including MRR, MDM, and MTPVS with two other policy proposed in [12] including Minimum Migration Time (MMT) and Maximum Correlation (MC). Ten experiments are executed separately for the 10 days of workloads depicted in Table 5 and their median results for energy consumption, SLA violation, number of VM migrations, execution time as well as ESV and ESM metrics are reported in Table 7. Figure 8 shows the energy consumption; Fig. 9 shows the value of SLA violation; Fig. 10 depicts the value of ESV metric; Fig. 11 shows the overall number of VM migrations; Fig. 12 depicts the ESM metric; and Fig. 13 shows the median value for average execution time of the whole resource management process.

MRR, MDM, and MTPVS policies have a global view of the system because they take all important system parameters introduced in Table 1 into consideration. Consequently, the ESV and ESM parameters for these policies are lower than other policies, as shown in Figs. 10 and 12, respectively. Moreover, as depicted in Fig. 9, due to consideration of all important system parameters as well as their importance in decision process, the total SLA violation of MRR, MDM, and MTPVS is much less

Table 7 Output results of different VM selection Policies

Policy	MMT	MRR	MDM	MTPVS	MC
Energy consumption (Kwh)	46.03	45.00	45.78	43.51	45.25
SLAV ($\times 10^{-5}$)	74.628	33.211	51.799	19.413	64.316
ESV ($\times 10^{-3}$)	36.686	15.573	25.494	8.709	27.048
Number of VM migrations	7005.5	3650.5	5149	2644	5273.5
ESM	257.78	57.36	131.20	22.77	145.38
Execution time (s)	0.02996	0.02241	0.02497	0.02139	0.02803

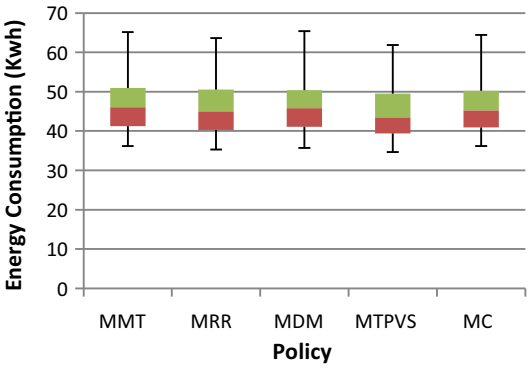

Fig. 8 Energy consumption of different VM selection policies

than other policies. Likewise, the same condition is hold for ESV, the number of VM's migration, and ESM as depicted in Figs. 10, 11, and 12, respectively.

On the other hand, as depicted in Figs. 8, 9, and 11, the results for MTPVS policy regarding energy consumption, SLA violation, and number of VM migrations are prominently less than other policies. As a result, ESV and also ESM metrics for MTPVS policy are much less in comparison with other policies as shown in Figs. 10 and 12, respectively. More precisely, it can be inferred from Table 7 that adoption of MTPVS policy leads to 5.5, 72.05, 62.25, 76.26, and 91.16 % reductions in energy consumption, SLA violation, number of VM migrations, ESV metric, and ESM metric, respectively, in comparison with MMT policy. This observation can be described by the fact that MTPVS policy aggregates the idea behind MRR and MDM policies to select a VM with the highest predicted CPU capacity and the least migration delay. In addition, MTPVS takes advantage of a multi-criteria decision making algorithm which finds a solution through simultaneous distance maximization from a negative ideal point as well as distance minimization from positive ideal point which notably improves the output results. Besides, MTPVS selects VMs for migration based on the predicted utilizations of their requested resource utilization using WMA prediction method rather than superficially based on the current CPU utilizations. In addition, MTPVS applies the importance of each system's criteria in decision process. Moreover, it can be inferred from Fig. 13 that adopting MTPVS policy leads to the least execution

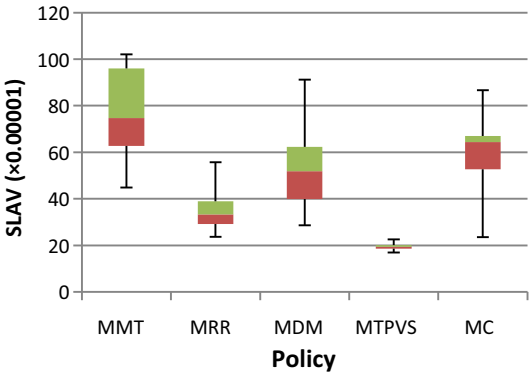

Fig. 9 SLA violation of different VM selection policies

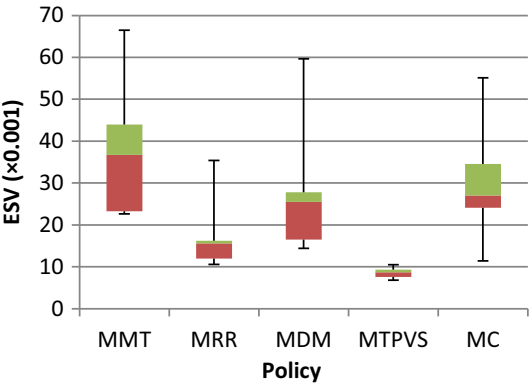

Fig. 10 ESV of different VM selection policies

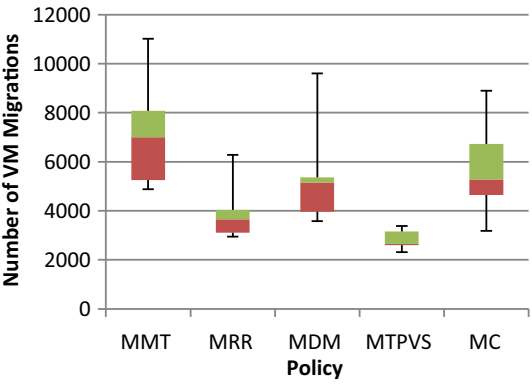

Fig. 11 Number of VM migrations of different VM selection policies

time in comparison with other policies. This observation can be described by the fact that MTPVS policy takes advantage of simpler mathematical calculations with lower complexities in comparison with other policies.

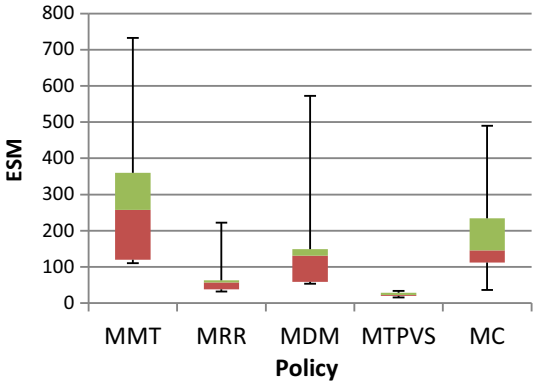

Fig. 12 ESM of different VM selection policies

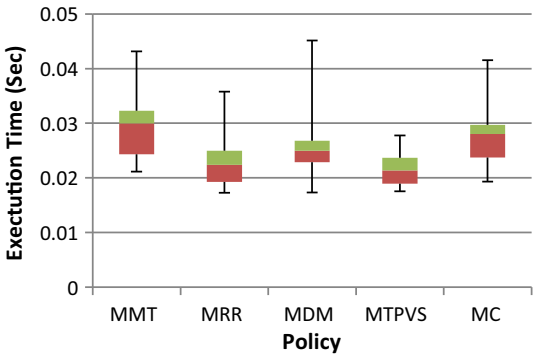

Fig. 13 Average execution time of different VM selection policies

7.3.3 Evaluation of combination of proposed policies for resource management

In this section we compare three scenarios consisting of combination of our best proposed policies for resource consolidation process in cloud data centers with the ones proposed in [12] and [11]. In this section we define a four segmented naming format, depicted in Table 8, for the notation of the scenarios assessed in this section. The sections of the naming format are arranged according to the four phases of consolidation procedure proposed in [12]. The notations are constructed by connecting the abbreviation of the policies used for each phase using slash lines.

Best combination of policies proposed in [12] include LR, SM, MMT, and PABFD for four phases of consolidation process. Therefore, other scenarios are compared with LR/SM/MMT/PABFD scenario (scenario 1) as a reference scenario. The LR/VDT/MMT/UMC scenario (scenario 2) proposed in [11] is similar to the ones proposed in [12] except that it uses VM-based dynamic threshold (VDT) policy for determination of underloaded PMs and utilization and minimum correlation (UMC) policy for resource allocation. The difference between our WMA/SM/MTPVS/PABFD scenario (scenario 3) and the one proposed in [12] is that it adopts WMA policy for determination of overloaded PMs as well as MTPVS policy for VM selection from overloaded PMs. Ten experiments are executed separately for

Table 8 The notation used for combinations of best proposed policies

Scenario number	Policy abbreviation
Scenario 1	LR/SM/MMT/PABFD
Scenario 2	LR/VDT/MMT/UMC
Scenario 3	WMA/SM/MTPVS/PABFD

Table 9 Output results for combination of best policies for different phases of resource management process

Policy	LR/SM /MMT/PABFD	LR/VDT/ MMT/UMC	WMA/SM/ MTPVS/PABFD
Energy consumption (Kwh)	46.03	45.39	51.39
SLAV ($\times 10^{-5}$)	74.628	68.228	3.829
ESV ($\times 10^{-3}$)	36.686	30.988	1.9993
Number of VM migrations	7005.5	5619.5	2334.5
ESM	257.78	174.83	4.85
Execution time (s)	0.02996	0.04576	0.05467
ESM improvement (%)	0	32.17	98.11

the 10 days of workloads depicted in Table 5 and their median results for energy consumption, SLA violation, number of VM migrations, execution time as well as ESV and ESM metrics are reported in Table 9. Figure 14 shows the energy consumption in the data center; Fig. 15 shows the value of SLA violation incurred to the system due to resource shortage as well as performance degradation due to migration; Fig. 16 depicts the value of ESV metric which can be used to infer the simultaneous improvement of energy consumption and SLA violation; Fig. 17 shows the overall number of VM migrations executed in the system during simulation time; Fig. 18 depicts the ESM metric which can be used to measure simultaneous improvement of energy consumption, SLA violation, and number of VM migrations; and Fig. 19 shows the median value for average execution time of the whole resource management process.

It can be inferred from Figs. 15, 16, 17, and 18 that our proposed scenario (scenario 3) prominently has the best performance regarding SLA violation, ESV metric, number of VM migrations, and ESM metric, respectively. More precisely, it can be inferred from Table 9 that adoption of scenario 3 leads to 58.68, 66.67, 94.5, and 98.11 % reductions in SLA violation, number of VM migrations, ESV metric, and ESM metric, respectively, in comparison with the reference scenario (scenario 1). However, as shown in Fig. 14, the total energy consumption of scenario 3 scenario is slightly more than other policies. This observation can be described by the existence of an intrinsic trade-off between energy consumption and SLA violation. More precisely, since energy and SLA violation are negatively correlated, the SLA violation is decreased by the cost of a small increase in energy consumption. But, the objective of a Cloud resource management system is simultaneous optimization of energy consumption, SLA violation, and number of migrations which can be inferred from ESM

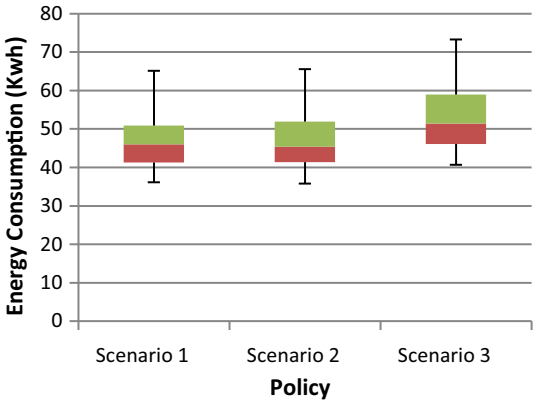

Fig. 14 Energy consumption of combination of best policies for resource management process

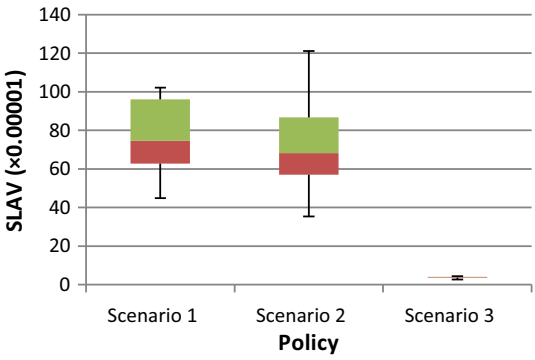

Fig. 15 SLA violation of combination of best policies for resource management process

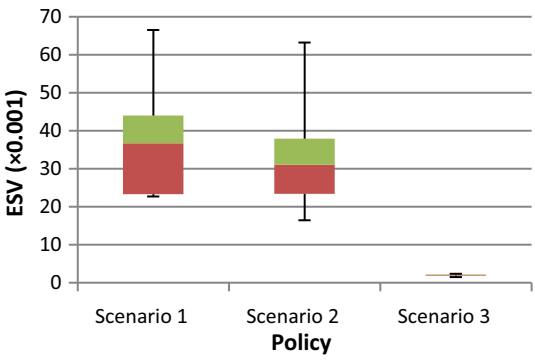

Fig. 16 ESV of combination of best policies for resource management process

metric. In this direction, scenario 3 shows the best performance in comparison with other scenarios, as shown in Fig. 18. Moreover, it can be deduced from Fig. 19 that adopting scenario 3 leads to a bit more execution time in comparison with other scenarios. This observation can be described by the fact that scenario 3 considers more criteria in decision process in comparison with other policies which leads to more execution time.

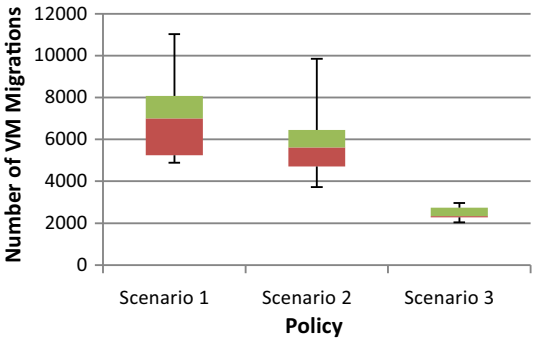

Fig. 17 Number of VM migrations of combination of best policies for resource management process

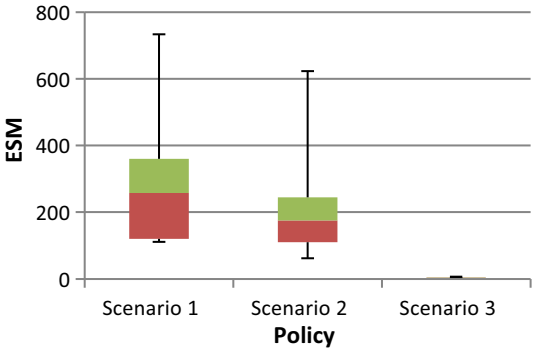

Fig. 18 ESM of combination of best policies for resource management process

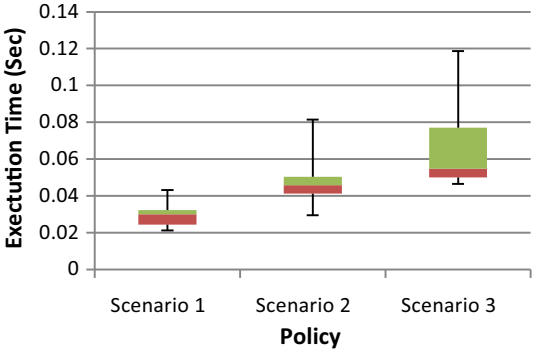

Fig. 19 Average execution time of combination of best policies for resource management process

7.3.4 Statistical analysis

In this section a statistical analysis is presented for the best algorithm combinations and benchmark algorithms. Based on the Ryan–Joiners normality test, ESM values of all three type scenarios (LR/SM/MMT/PABFD, LR/VDT/MMT/UMC and WMA/SM/MTPVS/PABFD) follow a normal distribution with the $P > 0.1$. Table 10 shows results based on paired t tests for all three aforementioned scenarios. Results show that there is statistically significant difference between these algorithms. The t

Table 10 Comparison of the algorithms using paired *t* tests

Policy 1 (ESM)	Policy 2 (ESM)	Difference ($\times 10^3$)	*P* value
LR/SM/MMT/PABFD (257.78)	WMA/SM/MTPVS/PABFD (4.85)	0.271 (0.155, 0.387)	<0.001
WMA/SM/MTPVS/ PABFD (4.85)	LR/VDT/MMT/UMC (174.83)	0.238 (0.111, 0.365)	< 0.001

Table 11 Comparison of the best algorithm combinations and benchmark algorithms regarding ESM metric

Policy	ESM	95 % CI
LR/SM/MMT/PABFD	276.43717	159.9861, 392.8883
LR/VDT/MMT/UMC	243.58346	116.5291, 370.6379
WMA/SM/MTPVS/PABFD	4.9134504	4.212525, 5.614375

tests have shown that the usage of the WMA/SM/MTPVS/PABFD scenario leads to a statistically significantly lower value of the ESM metric with the $P < 0.001$. Table 11 compares the best algorithm combinations and benchmark algorithms regarding the mean values of the ESM metric along with 95 % confidence intervals. From the observed results, we can conclude that the WMA/SM/MTPVS/PABFD scenario has the best performance regarding ESM metric.

8 Conclusion

There are serious concerns for cloud providers to reduce their energy consumption while ensuring a high level of adherence to service level agreements. This paper has focused on consolidation problem as an efficient resource management solution to reduce energy consumption in cloud data centers. This study has proposed novel heuristics for two main phases of consolidation problem including WMA policy for determination of overloaded PMs and MRR, MDM, and MTPVS policies for VM selection from overloaded PMs. One of the main strength of the proposed policies is consideration of all important system's criteria as well as their importance in decision process. Another main advantage of proposed policies is decision making based on the predicted capacity of the system's criteria rather than their current utilizations. Moreover, taking advantage of WMA policy, this paper has reached more accurate prediction values for system's resource types. Furthermore, this paper has proposed a novel method for computation of weights of different system's resource types. The experimental results obtained from extensive evaluations using CloudSim simulator have proven that our policies significantly outperform existing consolidation solutions regarding energy consumption, SLA violation, and number of VMs' migrations. The research work is planned to be followed by implementing the proposed policies using real cloud infrastructure management products such as OpenStack. Another direction for future research is investigation of novel algorithms for on-line VM placement on

heterogeneous data centers of different cloud service providers over wide area network connections.

References

1. Islam S, Keung J, Lee K, Liu A (2012) Empirical prediction models for adaptive resource provisioning in the cloud. Future Gen Comput Syst 28(1):155–162
2. Manvi SS, Krishna Shyam G (2014) Resource management for infrastructure as a service (IaaS) in cloud computing: a survey. J Netw Comput Appl 41:424–440
3. Nathani A, Chaudhary S, Somani G (2012) Policy based resource allocation in IaaS cloud. Future Gen Comput Syst 28(1):94–103
4. Durao F, Carvalho JFS, Fonseka A, Garcia VC (2014) A systematic review on cloud computing. J Supercomput 68:1321–1346
5. Jing S-Y, Ali S, She K, Zhong Y (2013) State-of-the-art research study for green cloud computing. J Supercomput 65:445–468
6. Lee YC, Zomaya AY (2012) Energy efficient utilization of resources in cloud computing systems. J Supercomput 60(2):268–280
7. Lee HM, Jeong Y-S, Jang HJ (2013) Performance analysis based resource allocation for green cloud computing. J Supercomput 69:1013–1026
8. Arianyan E, Taheri H, Sharifian S (2015) Novel energy and SLA efficient resource management heuristics for consolidation of virtual machines in cloud data centers. Comput Electr Eng 47:222–240
9. Son S, Jung G, Jun SC (2013) An SLA-based cloud computing that facilitates resource allocation in the distributed data centers of a cloud provider. J Supercomput 64(2):606–637
10. Beloglazov A, Buyya R, Lee YC, Zomaya A (2011) A taxonomy and survey of energy-efficient data centers and cloud computing systems. Adv Comput 82(2):47–111
11. Horri A, Mozafari MS, Dastghaibyfard G (2014) Novel resource allocation algorithms to performance and energy efficiency in cloud computing. J Supercomput 69(3):1445–1461
12. Beloglazov A, Buyya R (2012) Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in Cloud data centers. Concurr Comput Pract Exp 24(13):1397–1420
13. Beloglazov A, Buyya R (2013) Managing overloaded hosts for dynamic consolidation of virtual machines in cloud data centers under quality of service constraints. IEEE Trans Parallel Distrib Syst 24(7):1366–1379
14. Esfandiarpoor S, Pahlavan A, Goudarzi M (2014) Structure-aware online virtual machine consolidation for datacenter energy improvement in cloud computing. Comput Electr Eng
15. Beloglazov A, Buyya R (2012) Managing overloaded hosts for dynamic consolidation of virtual machines in cloud data centers under quality of service constraints
16. Beloglazov A, Abawajy J, Buyya R (2012) Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing. Future Gen Comput Syst 28(5):755–768
17. Minas L, Ellison B (2009) Energy efficiency for information technology: how to reduce power consumption in servers and data centers. Intel Press, USA
18. Chen C-T (2000) Extensions of the TOPSIS for group decision-making under fuzzy environment. Fuzzy Sets Syst 114(1):1–9
19. Nathuji R, Schwan K (2007) VirtualPower: coordinated power management in virtualized enterprise systems. ACM SIGOPS Oper Syst Rev 41(6):265–278
20. Li C (2012) Optimal resource provisioning for cloud computing environment. J Supercomput 62(2):989–1022
21. Kusic D, Kephart JO, Hanson JE, Kandasamy N, Jiang G (2009) Power and performance management of virtualized computing environments via lookahead control. Cluster Comput 12(1):1–15
22. Verma A, Ahuja P, Neogi A (2008) pMapper: power and migration cost aware application placement in virtualized systems. In: Middleware 2008. Springer, Berlin, pp 243–264
23. Xiao Z, Song W, Chen Q (2013) Dynamic resource allocation using virtual machines for cloud computing environment. IEEE Trans Parallel Distrib Syst 24(6):1107–1117
24. Yang D, Cao J, Fu J, Wang J, Guo J (2013) A pattern fusion model for multi-step-ahead CPU load prediction. J Syst Softw 86(5):1257–1266
25. De Gooijer JG, Hyndman RJ (2006) 25 years of time series forecasting. Int J Forecast 22(3):443–473

26. Jeong J, Kim S-H, Kim H, Lee J, Seo E (2013) Analysis of virtual machine live-migration as a method for power-capping. J Supercomput 66(3):1629–1655
27. Tzeng G-H, Huang J-J (2011) Multiple Attribute Decision Making: Methods and Applications. CRC Press, Boca Raton
28. Calheiros RN, Ranjan R, Beloglazov A, De Rose CA, Buyya R (2011) CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. Softw Pract Exp 41(1):23–50
29. Park K, Pai VS (2006) CoMon: a mostly-scalable monitoring system for PlanetLab. ACM SIGOPS Oper Syst Rev 40(1):65–74