New Six-Phase On-line Resource Management Process for Energy and SLA Efficient Consolidation in Cloud Data Centers

Ehsan Arianyan, Hassan Taheri, Saeed Sharifian, and Mohsen Tarighi
Department of Electrical & Electronics Engineering, Amirkabir University of Technology, Iran

Abstract: *The rapid growth in demand for getting various services combined with dynamic and diverse nature of requests initiated in cloud environments have led to the establishment of huge data centers which consume a vast amount of energy. On the other hand, in order to attract more users in dynamic business cloud environments, providers have to provide high quality of service for their customers based on defined Service Level Agreement (SLA) contracts. Hence, in order to maximize their revenue, resource providers need to minimize both energy consumptions and SLA violations simultaneously. This study proposes a new six-phase procedure for on-line resource management process. More precisely, this study proposes addition of two new phases to the default on-line resource management process including VM sorting phase and condition evaluation phase. Moreover, this paper shows the deficiencies of present resource management methods which fail to consider all effective system parameters as well as their importance, and do not have load prediction models. The results of simulations using cloudSim simulator validates the applicability of our proposed algorithms in reducing energy consumption as well as decreasing SLA violations and number of VMs' migration in cloud data centers.*

Keywords: *Cloud computing, virtual machine, energy consumption, migration, cloudSim.*

Received August 2, 2014; accept December 20, 2015

1. Introduction

Cloud computing has recently been brought into focus in both academic and industrial communities [8]. Cloud computing is associated with a new paradigm for provisioning different computing resources, usually addressed from three fundamental aspects: Infrastructure as a Service (IaaS), Platform as a Service (PaaS) and Software as a Service (SaaS) [21]. Cloud environment is an efficient solution for both compute intensive and data intensive applications [10]. Due to the fact that the cloud users may have sporadic and dynamic resource usage, the cloud environment is highly dynamic [2]. Circumscribed resource utilization results in astonishingly high operational cost and energy usage [1]. The energy cost in a typical cloud data center is doubled every five years [7]. High energy consumption not only translates to a high operating cost, but also leads to higher carbon emissions [9].

The main portion of energy is consumed in the infrastructure of data centers. However, in typical data center deployments, server utilization is below 30%, but idle servers still consume 60% of their peak power [18]. Therefore, underutilization is the major cause of energy waste in cloud data centers. Consolidation of VMs on the least possible PMs and switching idle PMs off is the most novel method to save energy [2, 4, 9]. However, the obligation of providing high quality of service to cloud customers leads to the necessity in dealing with the energy-performance trade-off, as aggressive consolidations may lead to performance degradation [5].

Recent studies including [2, 5, 12] have utilized a consolidation solution for online optimization of VM placements which is consisted of four separated phases including:

1. Determining when a host is considered as being overloaded requiring migration of one or more *VMs* from this host.
2. Determining when a host is considered as being underloaded leading to a decision to migrate all VMs from this host and switch the host to the sleep mode.
3. Selection of *VMs* that should be migrated from an overloaded host.
4. Finding a new placement of the *VMs* selected for migration from the overloaded and underloaded hosts.

One of the important drawbacks of current studies is that they do not consider all important system parameters other than CPU in their decision making process. However, modern multi-core processors are much more power-efficient than previous generations, whereas memory technology does not show any significant improvements in energy efficiency [3]. This fact makes memory one of the most important components of focus in the power and energy usage optimization [16]. The same condition can be applied to 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 [3]. So, this study proposes MESMM[1] as a novel resource management model for cloud environments that not only consider CPU, RAM, and network bandwidth, but also improves the overall performance of on-line resource management process by addition of two new phases to the default consolidation problem and develops new heuristics for them. Two new defined phases are:

5. Sorting *VMs* to be allocated on *PMs*.
6. Condition evaluation before execution of optimization process. Executing *VM* sorting phase is important because it is probable that there would be not enough resources for all the VMs in the migration list and the allocation policy cannot find proper *PM* to host all the *VMs*. Besides, Execution of condition evaluation phase is important due to elimination of probable extra costs that can be incurred to the system.

The main contributions of this paper are:

- Proposing Consolidated Optimization (CO) and addition of two new phases to the on-line resource management process including VM sorting phase and condition evaluation phase.
- Proposing Multi-Criteria TOPSIS Sorting with Prediction (MCTP) policy as a new technique for VM sorting phase.
- Proposing Minimum Downtime Migration Optimization (MDMO) policy as a new technique for condition evaluation phase.
- Proposing Window Moving Average (WMA) policy as a new load prediction technique.
- Considering all important parameters in decision making process including CPU, RAM, and network bandwidth.
- Proposing a simple and functional mechanism to compute weights of different resource types.

This paper first reviews related works in section 2. Section 3 describes the input parameters which are considered in resource management problem. Section 4 presents our system model. Section 5 presents our proposed resource management policies. In section 6, the applicability of our proposed solutions is evaluated using cloudSim simulator. Finally, our concluding remarks are presented in section 7 as well as future directions.

2. Motivation and Related Work

As stated in [17], there is a wide area of research in resource management field in cloud computing. Hence, to make comparisons possible, we mention energy aware resource allocation studies which are close to our work.

In [19], authors have investigated power management techniques in the context of large-scale virtualized systems for the first time. In addition to the hardware scaling and *VMs* consolidation, they have proposed a new power management method for virtualized systems called “soft resource scaling.” Also, 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.

In [3], authors 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 Quality of Service (QoS) requirements. 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 allocations. 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 [5], authors 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 including.

1. Determining overloaded hosts.
2. Determining underloaded hosts.
3. VM selection for migration from an overloaded host.
4. VM placement.

They have proposed novel adaptive heuristics for all parts.

In [22], authors have proposed a system that uses virtualization technology to allocate data center resources dynamically based on application demands and support green computing by optimizing the number of servers in use. They aim to achieve two goals in their algorithm: overload avoidance and green computing. To reach these goals, they have designed a load prediction algorithm that can capture the future resource usages of applications accurately without looking inside the VMs. Furthermore, they have defined a server as a hot spot if the utilization of any

[1]Multi Criteria Energy SLA Migration Model

of its resources is above a static hot threshold and as a cold spot if the utilizations of all its resources are below a static cold threshold.

In [12], authors 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.

In [2], authors 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) for resource allocation and determination of underloaded PMs in cloud data centers, respectively.

In [15], authors 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.

In [14], authors 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.

In sum, the main drawback of all the aforementioned studies is non-consideration of all system parameters except CPU in decision process. However, our study not only considers all important criteria including CPU, RAM, and network bandwidth but also improves the overall performance of on-line resource management process by addition of two new phases to the default consolidation problem.

3. Input System Parameters

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. Consequently, this study considers the six major parameters which are listed in Table 1. C_{CPU} specifies the computational power which is determined 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. 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.

Table 1. Input parameters for resource management.

No.	Parameter	Description	Unit
1	C_{CPU}	CPU clock speed multiplied by the number of 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	%

4. System Model

4.1. Target System Model

We use the target system model defined in [2] as shown in Figure 1. This model includes two important parts: a central manager and the agents. The central manager is the resource manager of the data center which allocates VMs to available *PMs* in the data center. Also, it resizes *VMs* according to their resource needs, and decides when and which *VMs* should to be migrated from *PMs*. The agents which are implemented in hypervisors are connected to the central manager through network interfaces and have the responsibility to monitor the PM as well as sending gathered information for central manager.

4.2. Power and Energy Models

By emerging modern multi-core CPUs with novel power management methods and introduction of virtualization technology, CPU is not the only major power consumer in data centers anymore. This fact combined with the difficulty of modeling power consumption in modern data centers, makes building precise analytical models a complex research problem [5]. Therefore, we utilize real data on power consumption provided by the results of the SPEC power Benchmark [15]. Table 2 shows the power consumption of the servers used in this study which is provided in [5].

Moreover, energy consumption is the summation of power consumed during a period of time according to Equation 1.

$$E(t) = \int_t P(t)dt \quad (1)$$

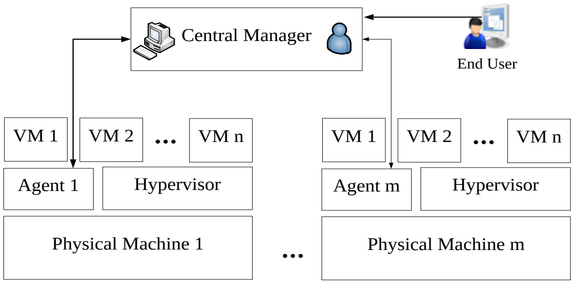

Figure 1. Proposed system model [2].

Table 2. Power consumption of considered servers for different loads (KWatts) [5].

Server	**Idle**	**10%**	**20%**	**30%**	**40%**	**50%**
HP ProLiant G4	86	89.4	92.6	96	99.5	102
HP ProLiant G5	93.7	97	101	105	110	116
Server	50%	60%	70%	80%	90%	100%
HP ProLiant G4	102	106	108	112	114	117
HP ProLiant G5	116	121	125	129	133	135

4.3. Average SLA Violation Metric

QoS requirements can be defined in terms of SLAs that are part of customer commitments and are described by such key performance metrics as minimal throughput and maximal response time [5]. 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 IaaS such as Overload Time Fraction (OTF) metric defined in [4]. However, this metric only considers CPU parameter and do not include multi parameters defined in this study. So, we introduce Multi Parameter SLA Violation (MPSV) that is capable of considering any number of input parameters according to Equation 2.

$$MPSV = \frac{\sum_{res=1}^{\#Res} \sum_i t_{0_i}^{res}(u_{ti})}{\sum_i t_{a_i}} \quad (2)$$

Where u_{ti} is the utilization of system parameter of type res at i'th saved history, containing the non-overloaded and overloaded states of a PM; $t_{0_i}^{res}$ is the time, during which the system parameter of type res of the PM has been overloaded, which is a function of u_{ti}; t_{ai} is the total time, during which the PM has been active; and res is acronym of resource which can be CPU, RAM, or network bandwidth.

4.4. Slav Metric

In order to compare our results with other studies, we use SLAV metric defined in [5] as a measure of the SLA violation due to both VM migration and resource shortage in a PM. SLAV is computed by multiplication of SLATAH and PDM. SLATAH is the percentage of time, during which active PMs have experienced the CPU utilization of 100 and PDM is the overall performance degradation by VMs due to migrations [5].

5. Proposed Resource Management Heuristics

MESMM takes advantage of CO policy and defines two new phases for consolidation problem. The two new defined phases are: (5) sorting VMs to be allocated on PMs, and (6) condition evaluation to execute optimization process. Moreover, MESMM takes advantage of WMA for load prediction of both PMs and VMs as well as MDMO policy for condition evaluation phase and Multi-Criteria TOPSIS Sorting with Prediction (MCTP) policy for VM sorting phase.

5.1. Consolidated Optimization Policy

The proposed system flowchart based on CO policy is depicted in Figure 2 in which the four boxes that make our flowchart different from state of the art are highlighted by drawing dashed lines around them. It is important to note that this flowchart is an extended version of our previous proposed EO policy presented in [2]. The boxes numbered 1 and 2 emphasize that the VMs to be migrated from either overloaded or underloaded PMs are gathered in the migration list and the final resource allocation is not yet arranged to be executed for them. The box numbered 3 indicates our proposed VM sorting phase in which our proposed MCTP policy is applied for sorting VMs. The box numbered 4 indicates our proposed condition evaluation phase in which our proposed MDMO policy is applied.

First, newly arrived VMs are placed on available PMs using TPSA policy [2]. In the next step, 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 WMA prediction algorithm [5]. So, if the WMA algorithm forecasts for a PM that utilization of either one of its resource types will be more than 100%, then this PM is determined to be an overloaded PM.

After that, VMs residing on overloaded PMs are selected for migration based on Minimum Migration Time (MMT) policy [5] until the elimination of hot spots. In the following step, a resource allocation procedure is executed for the sorted *VMs* to find their probable migration destination using TPSA allocation policy. If the control system finds a proper destination for a VM, then it is added to the migration list.

Following that, underloaded PMs are determined using TACND policy [2]. In each searching step to find underloaded PMs, 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. If the control system can find proper PMs as probable migration destinations for all the *VMs* residing on an underloaded PM using TPSA policy, then all its *VMs* are added to the VM's migration list. Otherwise, none of the *VMs* are added to the VM's migration list.

In the next step, all the *VMs* present in the migration list are sorted based on the *VM* sorting policy. Executing this step is important because it is probable that there would be not enough resources for all the VMs in the migration list and the allocation policy cannot find proper PM to host all the *VMs*. So, based on the defined sorting policy, the VMs are sorted so that to give higher ranks to the VMs that should have higher allocation priorities.

Then, a new placement is found for all the VMs in the migration list based on TPSA allocation policy. Major advantage of our proposed flowchart is that the VM placement step is executed in the final step after finding the complete list of *VMs* to be migrated either from overloaded or underloaded PMs, rather than in separate steps for them. Consequently, our placement has a holistic view of the whole probable allocations rather than executing *VM* allocations one by one.

At the final step, the condition evaluation phase is executed to assess the condition of the system and to analyze the final migrations list. The policy adopted in this phase determines whether the migration process should be initiated or not. Execution of this phase before initiating the migration process is important due to elimination of probable extra costs that can be incurred to the system. Following the positive decision taken in this step, the migration process is initiated.

One of the major advantages of CO policy is that, in contrast to EO policy, CO policy adopts the same policy for finding probable destinations for the VMs to be migrated from either overloaded or underloaded PMs as the one adopted for final *VM* placement process which notably improves the output results.

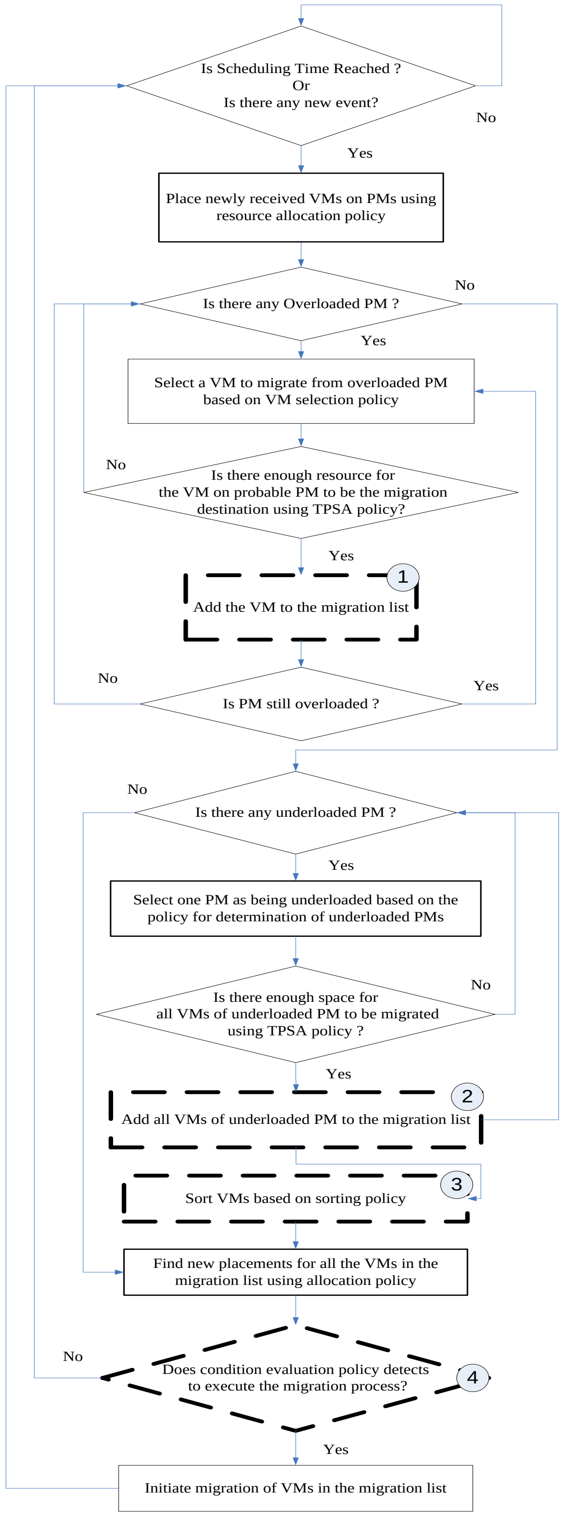

Figure 2. Proposed system flowchart.

5.2. WMA: Window Moving Average

WMA predicts the resource utilization regarding CPU, RAM, and network bandwidth based on their saved utilization history. WMA divides the saved history into two window parts and then finds the average of each window separately. In the next step, WMA predicts the future utilizations based on Equation 3.

$$\hat{U} = k \times \frac{{}_{i \in Window\,1} U_i}{Size(Window\,1)} + (1-k) \times \frac{{}_{i \in Window\,2} U_i}{Size(Window\,2)} \quad (3)$$

Where $\hat{U}$ the estimated utilization, k is a coefficient that specifies the weight of the recent samples and the old ones on the estimated utilization, and U_i is the i'th utilization value in the history. 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.

5.3. MCTP: Multi-Criteria TOPSIS Sorting with Prediction

MCTP is a multiple criteria method based on TOPSIS to sort VMs that are to be allocated by computing their score. MCTP sorts *VMs* so that the *VM* with the highest priority has 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 *VMs*. Distance from each of these poles are measured in the Euclidean distance.

MCTP computes the predicted values of the parameters depicted in Table 1 using the WMA policy. All the predicted information assigned to the virtual machines in time slot t form a decision matrix $\overrightarrow{MCTP}$ as shown in Equation 4.

$$\overrightarrow{MCTP} = \begin{bmatrix} P_{cpu}^{VM^1} & P_{ram}^{VM^1} & P_{net}^{VM^1} & C_{cpu}^{VM^1} & C_{ram}^{VM^1} & C_{net}^{VM^1} \\ \dots & \dots & \dots & \dots & \dots & \dots \\ P_{cpu}^{VM^i} & P_{ram}^{VM^i} & P_{net}^{VM^i} & C_{cpu}^{VM^i} & C_{ram}^{VM^i} & C_{net}^{VM^i} \\ \dots & \dots & \dots & \dots & \dots & \dots \\ P_{cpu}^{VM^m} & P_{ram}^{VM^m} & P_{net}^{VM^m} & C_{cpu}^{VM^m} & C_{ram}^{VM^m} & C_{net}^{VM^m} \end{bmatrix} \quad (4)$$

Where $VM^1, VM^2, \dots, VM^m$ are the VMs that MCTP is to sort them; $P_{res}^{VM^j}$ is the resource utilization of j'th VM in percent; $C_{res}^{VM^j}$ is the resource capacity of j'th VM; and res can be CPU, RAM, or network bandwidth.

In order to sort the VMs we go through the following steps:

Step1. First, we normalize the decision matrix $\overrightarrow{MCTP}$ to have dimensionless decision matrix $\overline{\overrightarrow{MCTP}}$. The purpose of decision matrix normalization is to make matrix entries free of unit so that they can take part in our computations. So, the decision matrix is made dimensionless by dividing each entry by maximum value of each column according to Equation 5.

$$\overline{\overrightarrow{MCTP}} = \begin{bmatrix} \frac{P_{cpu}^{VM^1}}{P_{cpu}^{\max}} & \frac{P_{ram}^{VM^1}}{P_{ram}^{\max}} & \frac{P_{net}^{VM^1}}{P_{net}^{\max}} & \frac{C_{cpu}^{VM^1}}{C_{cpu}^{\max}} & \frac{C_{ram}^{VM^1}}{C_{ram}^{\max}} & \frac{C_{net}^{VM^1}}{C_{net}^{\max}} \\ \dots & \dots & \dots & \dots & \dots & \dots \\ \frac{P_{cpu}^{VM^i}}{P_{cpu}^{\max}} & \frac{P_{ram}^{VM^i}}{P_{ram}^{\max}} & \frac{P_{net}^{VM^i}}{P_{net}^{\max}} & \frac{C_{cpu}^{VM^i}}{C_{cpu}^{\max}} & \frac{C_{ram}^{VM^i}}{C_{ram}^{\max}} & \frac{C_{net}^{VM^i}}{C_{net}^{\max}} \\ \dots & \dots & \dots & \dots & \dots & \dots \\ \frac{P_{cpu}^{VM^m}}{P_{cpu}^{\max}} & \frac{P_{ram}^{VM^m}}{P_{ram}^{\max}} & \frac{P_{net}^{VM^m}}{P_{net}^{\max}} & \frac{C_{cpu}^{VM^m}}{C_{cpu}^{\max}} & \frac{C_{ram}^{VM^m}}{C_{ram}^{\max}} & \frac{C_{net}^{VM^m}}{C_{net}^{\max}} \end{bmatrix} \quad (5)$$

Step 2. In the next step, VM^+ and VM^- are determined. Before determining VM^+ and VM^-, type of each attribute should be defined. Each attribute can be considered to have either benefit or cost type. 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. So, MCTP algorithm searches for a *VM* that has the lowest memory to avoid transferring large data over interconnection network. However, CPU and network bandwidth parameters are considered to have benefit type. So, VM_{res}^+ and VM_{res}^- are defined using Equations 6 and 7, respectively.

$$VM_{res}^+ = \left\{ P_{cpu}^+, P_{ram}^-, P_{net}^+, C_{cpu}^+, C_{ram}^-, C_{net}^+ \right\} \quad (6)$$

$$VM_{res}^- = \left\{ P_{cpu}^-, P_{ram}^+, P_{net}^-, C_{cpu}^-, C_{ram}^+, C_{net}^- \right\} \quad (7)$$

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

Step 3. The relative distance for each resource type of a VM from VM_{res}^+ and VM_{res}^- are calculated using Equation 8.

$$Score_{res}^{VM^j} = \frac{\sqrt{(VM_{res}^j - VM_{res}^-)^2}}{\sqrt{(VM_{res}^j - VM_{res}^-)^2} + \sqrt{(VM_{res}^j - VM_{res}^+)^2}} \quad (8)$$

Where $Score_{res}^{VM^j}$ shows the score of a specific resource type of j'th *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 Equation 8 becomes and consequently the score value is larger. Similarly, the less distance a *VM* has from VM^+, the less the value of denominator of Equation 8 becomes and accordingly the score value is larger.

Step 4. Compute the total score of a VM using Equation 9.

$$Score(VM^j) = \sum_{res=1}^{\#Res} Weight_{res} \times Score_{res}^{VM^j} \quad (9)$$

Where Score VM^j is the average closeness of j'th VM to the ideal solutions, $Weight_{res}$ is computed using Equation 10, and *#Res* is the number of considered resources.

Step 5. Sort the VMs according to their computed score. The *VM* with the highest score has the maximum distance from VM^- and the minimum distance from VM^+.

5.3.1. Parameters Weight Computation

As stated earlier, different parameters considered in a model have different importance on final decision. However, finding an optimized weight for different criteria is a wide research area by itself. One criterion could be that the more effective the input parameter is on output target value, the larger is its weight. Another criterion is a priority defined by system administrator or importance given by the user to a particular parameter over the others. 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 datacenter according to Equation 10.

$$Weight_{Res} = \frac{\overline{U}_{Res}(t)}{\sum_{res=1}^{\#Res} \overline{U}_{res}(t)} \quad (10)$$

where $\overline{U}_{Res}(t)$ 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 any of the parameters defined in Table 1.

5.4. MDMO: Minimum Downtime Migration Optimization

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 [13]. 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. Since famous hypervisors such as Xen, KVM, and *VM* ware 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 [5].

In all 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, the major goal of MDMO is simultaneous minimization of the downtime occurred during migration process for applications running in *VMs* as well as the total migration duration. So, MDMO evaluates the migrations' overhead incurred to the system due to the migration of *VMs* present in the migration list. More precisely, MDMO assesses the cost of migrating VMs in the migration list by computing the delay of transferring *VMs* from their host PM to their specified destination PM. The migration delay for each VM is estimated using Equation 11. MDMO decides to perform the migration of each VM if the computed migration delay for this *VM* is less than a predefined delay threshold which we call MDMO-threshold.

$$\mathrm{M}igrationDelay_{VM_i} = \frac{RAM\ Capacity_{VM_i}}{\min(BW_{PM_{source}}, BW_{PM_{destination}})} \quad (11)$$

Where $RAM\ Capacity_{VM_i}$ is the RAM capacity of VMi; $BW_{PM_{source}}$ and $BW_{PM_{destination}}$ are available network bandwidth in the source and destination PMs, respectively. So, migration delay is estimated as the RAM capacity defined for a *VM* divided by the minimum value between available spare network bandwidth for the PMs that are the source and destination of migration.

6. Performance Evaluation

In this section we define a seven segmented naming format, depicted in Table 3, for the notation of the scenarios assessed in this section. Different sections of the naming format are arranged according to the policy adopted for the whole resource management procedure, the policies adopted for the default four phases of consolidation procedure as well as the policies adopted for our proposed *VM* sorting and condition evaluation phases. The notations are constructed by connecting the abbreviation of the policies used for each section using slash lines. As shown in the first row of Table 3, a reference scenario scenario 1 consisting of a combination of the best policies reported in [5] for the default four phases of resource management process including Local Regression (LR) policy for determination of overloaded PMs, Simple Method (SM) policy for determination of underloaded PMs, Minimum Migration Time (MMT) policy for VM selection, and Power Aware Best Fit Decreasing (PABFD) policy for VM placement is considered as a base scenario. Besides, scenario 1 adopts Traditional four-phase Optimization (TO) process for the whole resource management procedure. Our proposed policies are compared with the reference scenario as well as with the policies proposed in [12] scenario 2 and [2] scenario 3. As shown in Table 3. [2, 5, 12] have used no policies for condition evaluation and *VM* sorting phases. Our proposed scenario scenario 4 takes advantage of TPSA policy for placement of migrating *VMs*, WMA policy for detection of overloading PMs, TACND policy for determination of underloaded PMs, MMT policy for *VM* selection, MCTP policy for *VM* sorting phase, and MDMO policy for condition evaluation phase.

Table 3. The notation of the proposed and benchmark policies.

Scenario Number	Policy Abbreviation
Scenario 1	TO/LR/SM/MMT/PABFD/-/-
Scenario 2	TO/LR/VDT/MMT/UMC/-/-
Scenario 3	EO/LR/TACND/MMT/TPSA/-/-
Scenario 4	CO/WMA/TACND/MMT/TPSA/MCTS/ MDMO

6.1. 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 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. So, we have used simulation for performance evaluation. We have chosen the CloudSim toolkit [4] as our simulation platform among available cloud computing simulators.

CloudSim is a modular open source took it library developed by the GRIDS laboratory of university of Melbourne for simulation of cloud computing scenarios [6]. It provides basic classes for describing data centers, virtual machines, applications, users, computational resources, and policies for management of diverse parts of the cloud systems such as scheduling and provisioning. Using CloudSim, enables us to perform repeatable experiments on large-scale virtualized data centers. However, the major limitation of CloudSim is the lack of a graphical user interface. Also, it only includes a basic and simplified network model and a limited workload generator.

In our infrastructure setup which has real configurations, we have simulated a data center comprising 800 installed heterogeneous physical machines half of which is supposed to be HP ProLiant ML110 G4 and the other half HP ProLiant ML110 G5 with the configurations depicted in Table 4. Power consumption by the physical machines is computed based on the models introduced in section 4. WMA predicts future utilizations in which k is set to be 0.3; size of window1 and window2 are set to be $\frac{1}{3}$ and $\frac{2}{3}$ of history length, respectively; and the history length is equal to 30. *VMs* are supposed to correspond to four Amazon *EC2 VM* types that have the configurations depicted in Table 5. Since using real workload for simulation experiments is important, we consider 10 days data of CoMon project [20]. 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 see Table 6. During the simulations, each *VM* is randomly assigned a workload trace from one of the *VMs* from the corresponding day.

Table 4. Configuration of servers.

Server	CPU model	Cores	Frequency (MHz)	RAM (GB)
HP ProLiant G4	Intel Xeon 3040	2	1860	4
HP ProLiant G5	Intel Xeon 3075	2	2660	4

Table 5. VM types (four Amazon EC2 VM types) [11].

VM type	CPU (MIPS)	RAM (GB)
High-CPU medium instance	2500	0.87
Extra-large instance	2000	3.75
Small instance	1000	1.74
Micro instance	500	0.613

Table 6. Workload data characteristics (utilizations) [20].

Date	Num. 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

6.2. Performance Metrics

Target parameters are energy consumption, SLA violation, and number of *VMs'* migrations which are measured using the models introduced in section 4.

Our ultimate goal is simultaneous minimization of energy, SLA violation, and number of VMs' migrations. So, we use the ESM metric defined in [2] that is representative of energy consumption, SLA violations, as well as number of *VMs'* migrations as defined in Equation 12. Besides, in order to to assess the simultaneous optimization of energy and SLA violation and also make our results comparable with the algorithms presented in [5, 12], we also consider the ESV parameter defined in [5].

$$ESM = Energy \times MPSV \times MigrationsCount \qquad (12)$$

6.3. Simulation Results

Ten experiments are executed separately for the 10 days of workloads depicted in Table 6 and their median results for energy consumption, number of *VM* migrations, execution time as well as ESV, and ESM metrics are reported in Table 7. Figure 3 shows the energy consumption; Figure 4 shows the value of SLA violation; Figure 5 shows the value of ESV metric; Figure 6 depicts the overall number of *VM* migrations; Figure 7 depicts the value of ESM metric; and Figure 8 represents the average execution time of different scenarios. It is important to note that the MDMO-Threshold is varied from 17 to 25 increasing by 1 and the output results reported in Table 7 for scenario 4 are the best one obtained when MDMO-Threshold is 25.

As depicted in Figures 4, 5, 6, and 7, adopting our proposed scenario leads to better performance regarding SLA violation MPSV metric, ESV metric, number of migrations, and ESM metric, respectively, in comparison with other scenarios. This observation can be described by the fact that adoption of our scenario leads to notable reductions in both SLA violations as well as the number of *VM* migrations as reported in Table 7. More precisely, our scenario has a holistic view for placing *VMs* on PMs, thanks to adoption of CO policy. Moreover, sorting *VMs* based on the MCTS policy which simultaneously considers multiple criteria in decision process results in finding smart prioritized allocations based on the computed scores for the *VMs* in the migration list. Besides, adding condition evaluation phase to the resource management process results in elimination of costly migrations. However, as depicted in Figure 3, the energy consumption for scenario 4 is slightly more than other scenarios. This observation can be described by the existence of an intrinsic trade-off between energy consumption and SLA violation which are typically negatively correlated. However, our goal is simultaneous minimization of energy consumption, SLA violation, and number of *VM* migrations that can be quantified using the ESM metric. It can be inferred from Table 7 that adoption of scenario 4 leads to 91.3% reductions in ESM metric, in comparison with the reference scenario.

Furthermore, as shown in Figure 8, adoption of scenario 4 leads to lower execution time in comparison with scenario 1 and scenario 2 but more execution time in comparison with scenario 3. This observation can be described by the fact that our scenario adds two new phases to the default four-phase on-line resource management process.

Table 7. Output results.

Scenario	Scenario1	Scenario2	Scenario3	Scenario4
Energy consumption(Kwh)	436.75	466.49	524.83	539.35
MPSV($\times 10^{-8}$)	28.35	22.10	10.07	5.00
ESV($\times 10^{-5}$)	12.505	10.102	5.264	2.501
Number of migrations	1302	982	689.5	593.5
ESM	0.1582	0.0939	0.0358	0.0138
Execution time (Sec)	0.0715	0.0753	0.0157	0.0332
ESM Improvement (%)	-	40.6	77.37	91.3

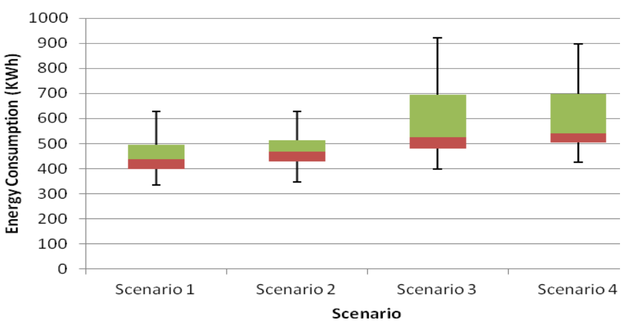

Figure 3. Energy consumption.

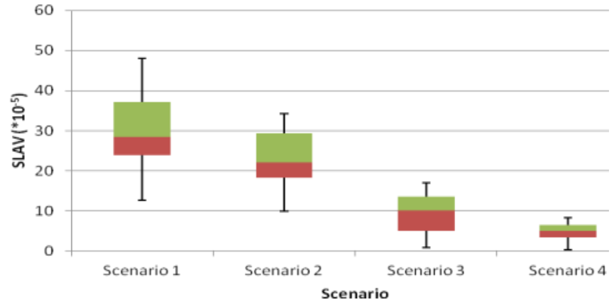

Figure 4. SLA violation.

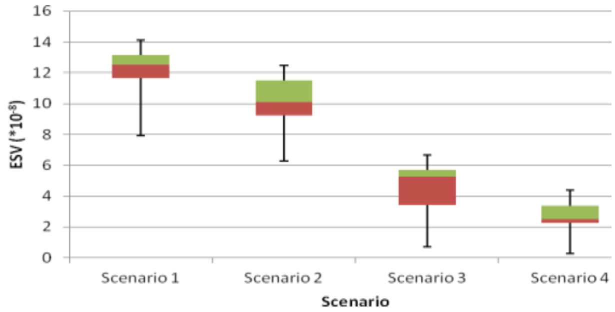

Figure 5. ESV.

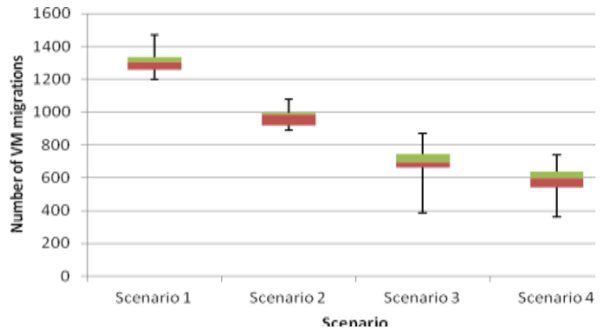

Figure 6. Number of migrations.

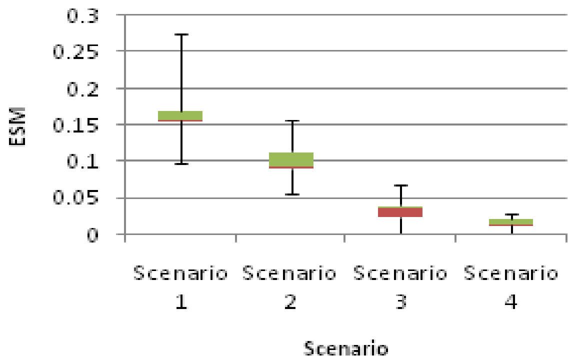

Figure 7. ESM.

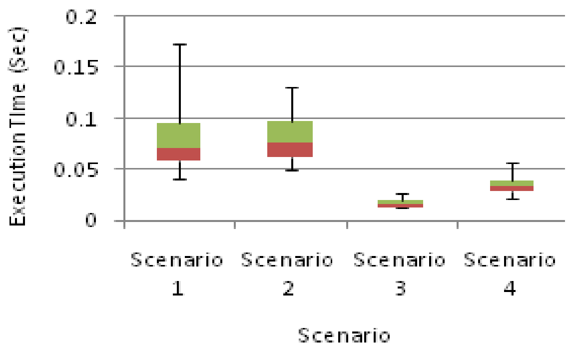

Figure 8. Average execution time.

7. Concluding Remarks and Future Directions

The major concern in cloud data centers is reducing energy consumption to diminish electricity bills as well as responding to regulations regarding cutting the carbon footprints. This paper addressed the consolidation problem in cloud data centers by proposing a novel six-phase procedure for the on-line resource management problem. This paper concentrated on important objectives in cloud ecosystems, which are energy consumption, SLA violation, and number of *VM* migrations. Moreover, this paper explained the central importance of considering different criteria such as CPU, RAM, and network bandwidth in decision making process. Besides, this paper proposed adding two new phases to the on-line resource optimization process which significantly improves the output results. More importantly, this paper proposed novel heuristics for the two new defined phases of resource management process that notably enhances the efficiency of resource management process in comparison with current techniques regarding both ESV and ESM metrics. More precisely, the results of experiments obtained from an extensive evaluation of proposed policies using Cloudsim simulator showed that adoption of our proposed scenario leads to 91.3% reduction in ESM metric, in comparison with the state of the art. The research work is planned to be followed by investigation of novel algorithms for on-line VM placement optimization between cloud service providers over wide area network connections.

References

[1] Ahmad R., Gani A., Hamid S., Shiraz M., Yousafzai A., and Xia F., "A Survey on Virtual Machine Migration and Server Consolidation Frameworks for Cloud Data Centers," *Journal of Network and Computer Applications*, vol. 52, pp. 11-25, 2015.

[2] Arianyan E., Taheri H., and Sharifian S., "Novel Energy and SLA Efficient Resource Management Heuristics for Consolidation of Virtual Machines in Cloud Data Centers," *Computers and Electrical Engineering*, vol. 47, no. 1, pp. 222-240, 2015.

[3] Beloglazov A., Abawajy J., and Buyya R., "Energy-Aware Resource Allocation Heuristics for efficient Management of Data Centers for Cloud Computing," *Future Generation Computer Systems*, vol. 28, no. 5, pp. 755-768, 2012.

[4] Beloglazov A. and Buyya R., "Managing Overloaded Hosts for Dynamic Consolidation of Virtual Machines in Cloud Data Centers Under Quality of Service Constraints," *IEEE Transactions on Parallel and Distributed Systems*, vol. 24, no. 7, pp. 1366-1379, 2012.

[5] Beloglazov A. and Buyya R., "Optimal Online Deterministic Algorithms and Adaptive Heuristics for Energy andPerformance Efficient Dynamic Consolidation of Virtual Machines in Cloud Data Centers," *Concurrency and Computation: Practice and Experience*, vol. 24, no. 13, pp. 1397-1420, 2012.

[6] Calheiros R., Ranjan R., Beloglazov A., De Rose C., and Buyya R., "CloudSim: a Toolkit for Modeling and Simulation of Cloud Computing Environments and Evaluation of Resource Provisioning Algorithms," *Software: Practice and Experience*, vol. 41, no. 1, pp. 23-50, 2011.

[7] Ebrahimirad V., Goudarzi M., and Rajabi A., "Energy-Aware Scheduling for Precedence-Constrained Parallel Virtual Machines in Virtualized Data Centers," *Journal of Grid Computing*, vol. 13, no. 2, pp. 233-253, 2015.

[8] Esfandiarpoor S., Pahlavan A., and Goudarzi M., "Structure-Aware Online Virtual Machine Consolidation for Datacenter Energy Improvement in Cloud Computing," *Computers and Electrical Engineering*, vol. 42, pp. 74-89, 2015.

[9] Farahnakian F., Ashraf A., Pahikkala T., Liljeberg P., Plosila J., Porres I., and Tenhunen H., "Using Ant Colony System to Consolidate VMs for Green Cloud Computing," *IEEE Transactions on Services Computing* , vol. 8, no. 2, pp. 187-198, 2015.

[10] Gao K., Wang Q., and Xi L., "Reduct Algorithm Based Execution Times Prediction inKnowledge Discovery Cloud Computing Environment," *The International Arab Journal of Information Technology*, vol. 11, no. 3, pp. 268-275, 2014.

[11] Gao Y., Guan H., Qi Z., Wang B., and Liu L., "Quality of Service Aware Power Management for Virtualized Data Centers," *Journal of Systems Architecture*, vol. 59, no. 4, pp. 245-259, 2013.

[12] Horri A., Mozafari M., and Dastghaibyfard G., "Novel Resource Allocation Algorithms to Performance and Energy Efficiency in Cloud Computing," *The Journal of Supercomputing*, vol. 69, no. 3, pp. 1445-1461, 2014.

[13] Jeong J., Kim S., Kim H., Lee J., and Seo E., "Analysis of VirtuaMachine Live-Migration as a Method for Power-Capping," *The Journal of Super Computing*, vol. 66, no. 3, pp. 1629-1655, 2013.

[14] Lee H., Jeong Y., and Jang H., "Performance Analysis Based Resource Allocation for Green Cloud Computing," *The Journal of Supercomputing*, vol. 69, no. 3, pp. 1013-1026, 2014.

[15] Lee Y. and Zomaya A, "Energy Efficient Utilization of Resources in Cloud Computing

Systems," *The Journal of Supercomputing*, vol. 60, no. 2, pp. 268-280, 2012.

[16] Luo L.,Wu W.,Tsai W., Di D., and Zhang F., "Simulation of Power Consumption of Cloud Data Centers," *Simulation Modelling Practice and Theory*, vol. 39, pp. 152-171, 2013.

[17] Manvi S. and Shyam G., "Resource Management for Infrastructure as a Service (IaaS) in Cloud Computing: A survey," *Journal of Network and Computer Applications*, vol. 41, pp. 424-440, 2014.

[18] Meisner D., Gold B., and Wenisch T., "PowerNap: Eliminating Server Idle Power," *in ACM Sigplan Notices*, vol. 44, no. 3, pp. 205-216, 2009.

[19] Nathuji R. and Schwan K., "VirtualPower: Coordinated Power Management in Virtualized Enterprise Systems," *ACM SIGOPS Operating Systems Review*, vol. 41, no. 6, pp. 265-278, 2007.

[20] Park K. and Pai V., "CoMon: a Mostly-Scalable Monitoring System for PlanetLab," *ACM SIGOPS Operating Systems Review*, vol. 40, no. 1, pp. 65-74, 2006.

[21] Tawfeek M., El-Sisi A., Keshk A., and Torkey F., "CloudTask Scheduling Based on ant Colony Optimization," *The International Arab Journal of Information Technology*, vol. 12, no. 2, pp. 129-137, 2015.

[22] Xiao Z., Song W., and Chen Q., "Dynamic Resource Allocation Using Virtual Machines for Cloud Computing Environment," *IEEE Transactions on Parallel and Distributed Systems*, vol. 24, no. 6, pp. 1107-1117, 2010.

Ehsan Arianyan received the M.S. degree from Amirkabir University of Technology, Tehran, Iran, in 2010. He is currently working toward the Ph.D. degree with the Department of Electrical Engineering. He is the author of more than 10 peer-reviewed papers as well as 3 books related to cloud computing. His research interests include cloud computing, parallel computing, and decision algorithms.

Hassan Taheri (M'90) received the M.S. and Ph.D. degrees from the University of Manchester, Manchester, U.K., in 1978 and 1988, respectively. He is currently an associate professor with the Department of Electrical Engineering, Amirkabir University of Technology. His research interests include cloud computing, teletraffic engineering, and quality of service in fixed and mobile communication networks.

Saeed Sharifian received the M.S. and Ph.D. degrees from the Amirkabir University of Technology, Tehran, Iran, in 2002 and 2009, respectively. He is now an assistant professor with the Department of Electrical Engineering, Amirkabir University of Technology. His research interests include high-performance web server architecture, parallel computing and programming, sensor networks, as well as performance modeling and evaluation.

Mohsen Tarighi received the M.S. and Ph.D. degrees from the Amirkabir University of Technology, Tehran, Iran, in 2008 and 2015, respectively. His research interests include cluster computing, virtualization, and decision algorithms.