Energy and QoS aware resource allocation for heterogeneous sustainable cloud datacenters

Yuyang Peng, Dong-Ki Kang, Fawaz Al-Hazemi, Chan-Hyun Youn

Department of Electrical Engineering, Korea Advanced Institute of Science and Technology, Daejeon, Republic of Korea

Abstract

As the demand on Internet services such as cloud and mobile cloud services drastically increases recently, the energy consumption consumed by the cloud datacenters has become a burning topic. The deployment of renewable energy generators such as Photovoltaic (PV) and wind farms is an attractive candidate to reduce the carbon footprint and, achieve the sustainable cloud datacenters. However, current studies have focused on geographical load balancing of Virtual Machine (VM) requests to reduce the cost of brown energy usage, while most of them have ignored the heterogeneity of power consumption of each cloud datacenter and the incurred performance degradation by VM co-location. In this paper, we propose Evolutionary Energy Efficient Virtual Machine Allocation (EEE-VMA), a Genetic Algorithm (GA) based metaheuristic which supports a power heterogeneity aware VM request allocation of multiple sustainable cloud datacenters. This approach provides a novel metric called powerMark which diagnoses the power efficiency of each cloud datacenter in order to reduce the energy consumption of cloud datacenters more efficiently. Furthermore, performance degradation caused by VM co-location and bandwidth cost between cloud service users and cloud datacenters are considered to avoid the deterioration of Quality-of-Service (QoS) required by cloud service users by using our proposed cost model. Extensive experiments including real-world traces based simulation and the implementation of cloud testbed with a power measuring device are conducted to demonstrate the energy efficiency and performance assurance of the proposed EEE-VMA approach compared to the existing VM request allocation strategies.

1. Introduction

The electric energy consumption of datacenters is accounted to be 1.5% of the worldwide electricity usage in 2010, and the energy cost is a primary fraction of a datacenter’s maintenance expenditure [1,2]. Therefore, there is a growing push to improve the energy efficiency of the data centers behind cloud computing [3,4]. Traditionally, datacenters get their power supply from the utility grid which is generated by dirty energy generators, such as coal, or nuclear plants [15]. These conventional energy generators not only produce much carbon but also increase the operation cost for datacenters. Towards addressing this inefficiency, a promising solution receiving spotlights is the incorporation of renewable energy generators such as PhotoVoltaic (PV) and wind turbines into the design of datacenters (i.e., achieving “sustainable” datacenters which reduce not only the electricity cost but also the carbon footprint). Renewable energy generator is becoming drastically an attractive candidate for designing...
green datacenters in academia. Recently, researchers have proposed several studies to integrate renewable energy sources into cloud datacenters. The cost optimization model considering both of renewable energy source and cooling infrastructure is proposed to realize the potential of sustainable cloud datacenters [9]. They propose the demand shifting which schedules non-interactive workload to maximize the utilization of renewable power source. The energy storage management of sustainable cloud datacenters has been proposed to minimize the cloud service provider’s electricity cost [10,11]. The scheduling scheme for parallel batch jobs has been proposed in order to maximize the utilization of green energy consumption while ensuring the Service Level Agreements (SLAs) of requests [16]. However, there are still remaining challenges to achieve the energy efficient sustainable cloud datacenters.

First, each cloud datacenter have heterogeneous server architecture, i.e., they require different power consumption even for serving of the same amount of workload. The server heterogeneity is caused by hardware upgrades, capacity extension, and the replacement of peripheral devices [6–8]. However, traditional cloud datacenter management schemes assume that all the cloud datacenters have homogeneous server architecture with same power efficiency although this assumption is unrealistic for most cloud resource providers. Second, two issues of greening cloud datacenters and Quality-of-Service (QoS) assurance are conflicting goals in the resource management. Especially, the performance degradation might be induced by VM co-location interference when multiple VM instances are running on common physical server in cloud datacenters [12]. As more VM instances are packed into common servers, the required number of active servers is decreased, while the resource contention is deteriorated. This means that the energy consumption is reduced with sacrificing the QoS assurance of the processing for VM requests. It is important to find a desirable tradeoff between above two goals corresponding to the dynamic workload level.

To solve these challenges, we propose an Evolutionary Energy Efficient Virtual Machine Allocation (EEE-VMA) approach which depends on an energy optimization model for sustainable cloud datacenters having heterogeneous power efficiency with renewable energy generators. This paper proposes four contributions as belows.

First, our approach tries to find a near optimized solution of VM request allocation by applying Genetic Algorithm (GA) with consideration for both of renewable energy cost and traditional utility grid cost. The fundamental strategy adopted in EEE-VMA as an energy saving scheme is Dynamic Right Sizing (DRS) which is for making cloud datacenters be power-proportional (i.e., consumes power only in proportion to the workload level) by adjusting the number of active servers in response to actual workload (i.e., to adaptively “right-size” the datacenter) [3,5]. In DRS, the energy saving can be achieved through allowing idle servers that do not have any running

Fig. 1. Cloud environment consists of multiple cloud datacenters and Cloud Request Brokers (CRBs) with Cloud Request Broker Manager (CRBM).
VM instances to be low-power mode (e.g., sleep or hibernation). Note that our proposed energy consumption model for the EEE-VMA approach includes a switching cost of DRS which is incurred by toggling a server from low-power mode into active mode (i.e., awaken transition). This makes our proposed approach more practical for energy efficient cloud datacenter in real world.

Second, in our proposed EEE-VMA approach, in order to adopt the heterogeneous power efficiency of each cloud datacenters, we propose a novel metric called powerMark to quantize the power efficiency of servers by measuring their power consumption at each utilization level of resources such as CPU, memory, and I/O bandwidth. Especially, we compute powerMark for serveral types of server by measuring their power consumption for processing CPU-intensive applications. Through powerMark, we are able to determine the allocation priority of each cloud datacenter based on their power efficiency so as to improve the performance of energy saving.

Third, we achieve the significant energy saving of cloud datacenters while minimizing the performance degradation caused by VM co-location interference through our EEE-VMA approach. The workload model including both of the number of co-located VM instances and the resource utilization which are key factors reflecting VM co-location interference is applied to the cost model of the EEE-VMA approach. Moreover, we consider the bandwidth cost between cloud service users and cloud datacenters as an additional part contributing the QoS deterioration of VM request processing [31,32]. The desirable cloud datacenter selection for each VM request assignment are conducted with consideration for both of energy saving and QoS assurance corresponding to the dynamic workload level.

Finally, we conduct extensive experiments through simulations at various workload levels based on real-world traces such as dynamic capacity of renewable energy and electricity prices of traditional grid power [9,18–20], and the implementation of testbed with a power measuring device called Yocto-Watt to measure a real power consumption of several cloud server types [21].

The rest of the paper is organized as follows. Section 2 gives an overview of the proposed system architecture of multiple cloud datacenters and cloud request brokers. In Section 3, the objective cost model including workload and energy consumption model with powerMark are formulated. Our EEE-VMA approach based on Genetic Algorithm is proposed to obtain the approximated optimal solution minimizing the total cost of cloud datacenters in Section 4. Section 5 shows the various experimental results that demonstrate the effectiveness of our proposed approach based on real-world traces. The conclusion is given in Section 5.

2. System architecture and design

Our considered cloud environment including multiple Cloud Request Brokers (CRBs) which support mesh networking with distributed multiple cloud datacenters is depicted in Fig. 1. There are $h$ CRBs and $m$ cloud datacenters with $h \times m$ communication links. In each cloud datacenter, the information of resource utilization, the available renewable energy, and the power consumption of each server are collected through monitoring modules, power measuring devices, and reported to the Cloud Request Broker Manager (CRBM) which is responsible for solving the allocation of VM requests submitted to CRBs. The CRBM has two modules: the powerMark analyzer and the EEE-VMA solver. The powerMark analyzer is responsible for capturing the power efficiency of each cloud datacenter through our proposed novel metric called powerMark. We describe this metric in detail in Section 3. The EEE-VMA solver is responsible for finding a near optimal solution of VM request allocation from CRB to the cloud datacenter. The solution derived by the EEE-VMA solver based on the amount of submitted VM requests in

### Table 1
Set of key notations.

<table>
<thead>
<tr>
<th>Notation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\mathcal{DC}$</td>
<td>The set of cloud datacenters</td>
</tr>
<tr>
<td>$\mathcal{CRB}$</td>
<td>The set of CRBs</td>
</tr>
<tr>
<td>$\mathcal{F}$</td>
<td>The set of flavor types of VM request supported by cloud resource provider</td>
</tr>
<tr>
<td>$\mathcal{RC}$</td>
<td>The set of resource components such as CPU and memory</td>
</tr>
<tr>
<td>$\Lambda(t)$</td>
<td>The set of VM requests arrived at whole CRBs at time $t$</td>
</tr>
<tr>
<td>$X(t)$</td>
<td>Resource allocation plan of VM requests from CRBs to cloud datacenters at time $t$, which determines the destined cloud datacenters for each VM request</td>
</tr>
<tr>
<td>$\mathcal{M}(t)$</td>
<td>DRS plan of cloud datacenters at time $t$, which determines the number of active servers of cloud datacenters</td>
</tr>
<tr>
<td>$\mathcal{S}(t)$</td>
<td>A solution including resource allocation plan $X(t)$ and DRS plan $\mathcal{M}(t)$</td>
</tr>
<tr>
<td>$\mathcal{D}(t)$</td>
<td>Performance degradation of cloud datacenter $DC_j$, by CPU resource contention at time $t$</td>
</tr>
<tr>
<td>$\mathcal{UR}_j$</td>
<td>The set of predetermined resource utilization levels</td>
</tr>
<tr>
<td>$\mathcal{pwM}_{rc}(t)$</td>
<td>An average power consumption per an unit level of utilization of resource component $rc \in \mathcal{RC}$ of servers in the cloud datacenter $DC_j$</td>
</tr>
<tr>
<td>$\mathcal{pivots}$</td>
<td>The predetermined pivot server used as a criterion of resource capacity</td>
</tr>
<tr>
<td>$\epsilon(t)$</td>
<td>The energy consumption of cloud datacenter $DC_j$ at time $t$</td>
</tr>
<tr>
<td>$\epsilon_{\text{total}}(t)$</td>
<td>The total cost of whole cloud datacenters at time $t$</td>
</tr>
<tr>
<td>$J_{\text{EEE-VMA}}(\cdot)$</td>
<td>The objective function to get $\epsilon_{\text{total}}(t)$ in the EEE-VMA solver</td>
</tr>
</tbody>
</table>
each CRB and the reported information from each cloud datacenter is delivered to whole CRBs and all the submitted VM requests are allocated to their destined cloud datacenters.

The owner of cloud datacenters has to minimize costs for resource operation while boosting benefits which can be realized since cloud service users have a good reputation on observed QoS by cloud services. In this paper, our EEE-VMA solver tries to find a solution to minimize the total cost of resource operation including three sub cost models: energy consumption cost, bandwidth cost, and performance degradation cost. In the perspective of energy consumption cost, the EEE-VMA solver tries to maximize the utilization of renewable energy with consideration on the dynamic capacity of each renewable energy generator since the price of renewable energy is much cheaper than the one of grid energy.

VM requests from CRBs are preferably allocated to cloud datacenters which have the higher capacity of renewable energy and the higher power efficiency (i.e., the lower powerMark value). In the perspective of bandwidth cost, the EEE-VMA tends to route VM requests to cloud datacenters having the cheaper bandwidth cost. Obviously, different pairs of CRB and cloud datacenter have different bandwidth cost according to the hop distance and the amount of transferred data of routed VM requests. Therefore, it is clear that VM requests need to be allocated to the closest cloud datacenter to their source CRB in order to minimize the bandwidth cost. To simplify our model, we assume that the transferred data size of each VM request is known to the EEE-VMA solver in CRBM beforehand.

In the perspective of performance degradation cost, the EEE-VMA solver tries to spread whole VM requests over multiple cloud datacenters in order to avoid QoS deterioration of VM request processing. In cloud datacenters, the VM co-location interference is the key factor that makes servers undergo severe performance degradation [12,22]. The VM co-location interference is caused by resource contention which can be reflected mainly by the number of co-located VM instances and resource utilization of them. In brief, the VM co-location interference grows bigger as more VM instances are co-located on the common server and the higher resource utilization is occurred. Therefore, VM requests have to be scattered in order to try its hardest to avoid performance degradation by VM co-location interference. Because of the complexity of optimization for aggregated cost model, the EEE-VMA solver adopts metaheuristic based on GA to obtain near optimal solution of VM requests allocation within the acceptable computation time. In next section, we propose a mathematical model to describe the cost of cloud datacenter and describe the metric powerMark in detail. The set of involved key notations are shown in Table 1.

3. Problem formulation

3.1. Workload model

There are many different kinds of workloads in cloud datacenters which can be classified into two categories: interactive or transactional (delay-sensitive) and non-interactive or batch (delay-tolerant) workload [9]. The interactive workloads such as Internet web services and multimedia streaming services have to be processed within a certain response time defined by service users. They are often network I/O intensive jobs which have less impact to the power consumption of servers. In contrast, the batch workloads such as scientific applications and big data analysis can be scheduled to process anytime as long as the whole tasks are finished before the predetermined deadline. They are usually computation intensive jobs that require a lot of CPU utilization causing a significant power consumption of servers. In this paper, we are interested in the computation intensive batch workloads since they have a greater influence to server power consumption than interactive workloads. We assume that all the VM requests have computation intensive workloads, and the resource contention is always occurred in CPU resource. A workload $\lambda_i(t) \in \Lambda_i(t)$ denotes the number of arrived VM requests with a required flavor type (e.g., instance type such as m3.medium or c4.large in Amazon EC2) $F_k \in \mathcal{F}$ at the CRB $\in \mathcal{CRB}$ at time $t$ [29]. We use $r^{rc}_{i(t)}$ to denote the required amount of resource component $rc \in \mathcal{RC}$ by a VM request with flavor type $F_k$ where $\mathcal{RC} = \{rc_{CPU}, rc_{MEM}\}$. For example, $r^{rc}_{cpu}$ such that $F_k = m3.medium$ and $rc = rc_{CPU}$ represents the required number of CPU cores by a VM request of which the flavor type is m3.medium. When multiple VM requests are arrived at the CRB, then the CRB would decide in which cloud datacenters each VM request should be routed for processing. We assume no data buffering at the CRB so that whenever a VM request arrives at the CRB, it would be routed to a cloud datacenter for processing immediately [11].

We denote the number of VM requests with the flavor type $F_k$ routed from the CRB $\in \mathcal{CRB}$ to $D_j \in \mathcal{DC}$ at time $t$ as $X^{jk}(t)$, which is derived by a resource allocation plan for cloud datacenters, $X(t)$. Then we have the following constraints:

$$\sum_{\forall DC_j \in \mathcal{DC}} X^{jk}_i(t) = \lambda^i(t), \forall CRB_i \in \mathcal{CRB}, \forall F_k \in \mathcal{F}, \forall t \quad (1)$$

$$0 \leq X^{jk}_i(t) \leq \lambda^i(t), \forall CRB_i \in \mathcal{CRB}, \forall F_k \in \mathcal{F}, \forall DC_j \in \mathcal{DC}, \forall t \quad (2)$$

Above Eq. (1) means that the total number of VM requests arrived at CRBs must agree with the one of whole VM requests allocated to cloud datacenters. Another constraint we should consider is a resource capacity of the cloud datacenter. Each cloud datacenter only can accommodate VM requests within their resource capacity (e.g., the total number of CPU cores). Then, we have the following constraints

$$\sum_{\forall CRB_i \in \mathcal{CRB}} \sum_{\forall F_k \in \mathcal{F}} r^{rc}_{cpu}(t) \cdot m_i(t), \forall DC_j \in \mathcal{DC}, \forall t \quad (3)$$

$$\sum_{\forall CRB_i \in \mathcal{CRB}} \sum_{\forall F_k \in \mathcal{F}} r^{rc}_{mem}(t) \cdot m_i(t), \forall DC_j \in \mathcal{DC}, \forall t \quad (4)$$

<table>
<thead>
<tr>
<th>Table 1</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\lambda_i(t)$</td>
<td>Number of arrived VM requests</td>
</tr>
<tr>
<td>$\mathcal{CRB}$</td>
<td>Cloud datacenters</td>
</tr>
<tr>
<td>$\mathcal{F}$</td>
<td>Flavors of VM requests</td>
</tr>
<tr>
<td>$\mathcal{RC}$</td>
<td>Resource components</td>
</tr>
<tr>
<td>$r^{rc}_{cpu}$</td>
<td>Required number of CPU cores</td>
</tr>
<tr>
<td>$r^{rc}_{mem}$</td>
<td>Required number of memory</td>
</tr>
<tr>
<td>$X^{jk}_i(t)$</td>
<td>Number of VM requests</td>
</tr>
<tr>
<td>$\mathcal{DC}$</td>
<td>Cloud datacenters</td>
</tr>
<tr>
<td>$m_i(t)$</td>
<td>Resource capacity of cloud datacenters</td>
</tr>
</tbody>
</table>

Y. Peng et al. / Optical Switching and Networking 23 (2017) 225–240
$0 \leq m_j(t) \leq N(D_j), \forall D_j \in \mathcal{D}, \forall t$ (5)

where $r^k_\text{CPU}$ and $r^k_\text{RC}$ are required CPU cores and memory size of VM request with flavor type $F_k \in \mathcal{F}$. $scp_\text{RC}$ is the physical capacity of resource component $rc \in \mathcal{R}_C$ of an arbitrary server in the cloud datacenter $D_j$. Constraints (3) and (4) represent that allocated VM requests can not always exceed the capacity of resource provided by cloud datacenter $D_j$. We use $m_j(t)$ to denote the number of active servers in cloud datacenter $D_j$ at time $t$ and it is determined by a DRS plan $M(t)$, and its upper bound is $N(D_j)$ which is the number of total physical servers in the cloud datacenter $D_j$. The constraint (5) represents that $m_j(t)$ can be determined in the range of $0$ to $N(D_j)$ through the DRS plan. $m_j(t) = 0$ means that whole servers in the cloud datacenter $D_j$ are in the sleep state, while $m_j(t) = N(D_j)$ means that they are in the active state at time $t$.

Next, we consider a VM co-location interference to build a performance degradation model of resource allocation in cloud datacenter [12]. The VM co-location interference implies that the virtualization of cloud supports resource isolation explicitly when multiple VM requests are running simultaneously on common PM, but it does not mean the assurance of performance isolation between VM requests internally. In the perspective of CPU resource, physical CPU cores of the server are not pinned to each running VM request, but assigned dynamically. The switching overhead by the dynamical CPU assignment policy might cause the undesirable performance degradation of allocated VM requests. Moreover, the CPU resource contention aggravates the performance degradation since it is very difficult to isolate the cache space of CPU. There is a strong relationship between VM co-location interference and the number of co-located VMs in PM [12]. The more co-located VM instances, the more severe VM co-location interference is occurred. Based on [12], we estimate the performance degradation $D_j(t)$ of the cloud datacenter $D_j \in \mathcal{D}$ by the CPU resource contention at time $t$ as follows.

$$D_j(t) = \sum_{\forall \mathcal{C}_R \in \mathcal{C}, \forall \mathcal{F}_k \in \mathcal{F}, \forall r_c \in \mathcal{R}_C} L^k(t) \cdot r^k_\text{RC} \cdot \frac{(\mu_{\text{CPU}} - \mu_{\text{CPU}}(t)) + \overline{\mathcal{F}}_j(t))}{scp_{\text{RC}(r_c)} \cdot m_j(t)}$$

where $\overline{\mathcal{F}}_j(t)$ is an average allocated time slice determined by Hypervisor [25,28] for VM requests allocated to the cloud datacenter $D_j$ at time $t$. We use $\overline{\mu_{\text{CPU}}(r_c)}(t)$ to denote an average utilization of assigned virtual resources of whole VM requests allocated to cloud datacenters at time $t$. Note that in Eq. (5), $\overline{\mathcal{F}}_j(t)$ and $r^k_\text{RC} \cdot \mu_{\text{CPU}}(t)$ can be known in advance, while $\overline{\mu_{\text{CPU}}(r_c)}(t)$ can not be recognized beforehand, until the utilization of CPU resource is measured through the internal monitoring module of each server in cloud datacenters at time $t$ [24]. Therefore it is required to use the historical information of CPU resource utilization of VM requests to find optimal solution of resource allocation for the current workload. As shown in Fig. 1, the data repository module is responsible for collecting and storing the monitoring information of resource utilization of each VM request to estimate the future demand. Our EEE-VMA solver uses the historical data of the resource utilization from the data repository module in each cloud datacenter to estimate the expected performance degradation of solution candidates.

3.2. Energy consumption model

3.2.1. The renewable energy model

The renewable energy such as the PV and wind energy is more sustainable than the traditional grid power, and its price is low and the less carbon is emitted [9]. There are two models to achieve the sustainable cloud datacenters by deploying the renewable energy generation. One is on-site deployment of renewable energy generation at the datacenter facility itself. For example, Apple has built its own local biogas fuel cells and two 20-MW solar arrays in Maiden, NC and they have been powered by 100% renewable energy sources [26,27]. Such on-site renewable energy generator can alleviate energy losses due to the transmission and distribution of generated energy, but its energy potential depends greatly on the location of the cloud datacenter. Another model is building the renewable energy generator at off-site facilities. It has the flexibility to locate the generator in a location with good weather (e.g., strong wind speed or bright sunshine), but the significant transmission losses of energy can be occurred. In this paper, we use the first model which has been adopted by most major datacenter owners.

We denote $rwe_j(t)$ and $rpe_j(t)$ as dynamic capacity of renewable wind energy and renewable photovoltaic energy of the cloud datacenter $D_j \in \mathcal{D}$ at time $t$, respectively. Obviously, it is required to forecast the future capacity of renewable energy to achieve energy efficient resource management of cloud datacenters since they are usually intermittent and irregular. Therefore, we estimate the future capacity of renewable energy generation by using the historical data from the data repository module in the cloud datacenter through calculating an Exponentially Weighted Moving Average (EWMA) values. The detailed descriptions of the EWMA based forecasting scheme for estimated capacity of renewable energy is omitted in this paper.

3.2.2. Heterogeneous power consumption model

We propose a novel power efficient metric called $\text{powerMark}$ to evaluate the heterogeneous power consumption of cloud datacenters. Servers consist of each cloud datacenter have heterogeneous architecture, which implies that the specification of their resources are different, consequently, even though they process the same application, for which each required power consumption might be different [8,13]. To describe $\text{powerMark}$ in detail, we propose Definition 1 and 2 as belows,
Definition 1. (powerMark):

The powerMark $pwM_{rc}$ is an average power consumption per an unit level of utilization of resource component $rc \in RC$ of servers in the cloud datacenter $DC_i$.

Definition 2. (pivot server):

The predetermined pivot server $pivotS$ used as a criterion of resource capacity for normalizing powerMark of each cloud datacenter.

Moreover, we propose the novel concept pivot server $pivotS$ in Definition 2 to normalize the powerMark of each cloud datacenter. For simplicity, we assume that servers in the same cloud datacenter has power-Homogeneity to each other. To obtain the powerMark value, we predetermined the set of resource utilization levels $UR = \{ ur_1, ur_2, ..., ur_k \}$. The powerMark $pwM_{rc}$ represents the power efficiency of servers in the cloud datacenter $DC_i$ with respect to the certain resource component $rc \in R$ by calculating an arithmetic mean of power consumption measured at each resource utilization level $ur_j \in UR \forall ur_j > 0$. For example, we set $UR = \{ ur_1 = 0.1, ur_2 = 0.2, ..., ur_9 = 0.9 \}$ and $rc = rcCPU$, then the power consumption of server is measured at each CPU utilization level $0.1, 0.2, ..., 0.9$ respectively. Based on the data of measured power consumption, the powerMark $pwM_{rc}$ is given by

$$pwM_{rc} = \frac{1}{|UR|} \sum_{ur_k \in UR} \frac{pw_{rc, ur_k}}{ur_k}, \quad (7)$$

$$npwM_{rc} = \frac{1}{|UR|} \sum_{ur_k \in UR} \frac{scp_{rc, ur_k}}{scp_{rc}} \cdot \frac{pw_{rc, ur_k}}{ur_k}, \quad (8)$$

where $pw_{rc, ur_k}$ is the power consumption of servers in the cloud datacenter $DC_i$ at the utilization level $ur_k$ of the resource component $rc$. $npwM_{rc}$ is the normalized value of $pwM_{rc}$ based on the capacity of resource component $rc$ of the pivot server $pivotS$ where $\frac{scp_{rc}}{scp_{rc}} \cdot \frac{pw_{rc, ur_k}}{ur_k}$ is the normalized value of $pw_{rc, ur_k}$. The lower powerMark represents the higher power efficiency of the cloud datacenter and, with larger $|UR|$, powerMark can accurately describe the power efficiency of the cloud datacenter. In order to investigate the availability of powerMark, we conduct the preliminary experiment to obtain power consumption of heterogeneous servers with running VM requests processing computation-intensive jobs on a real test bed. There are 3 server types to investigate the heterogeneity of power consumption. The hardware specifications of each server type are shown in Table 2.

Table 2

<table>
<thead>
<tr>
<th>Server types</th>
<th>CPU architecture</th>
<th>CPU cores</th>
<th>CPU clocks (GHz)</th>
<th>Cache size (kB)</th>
<th>Memory size (GB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Server-1</td>
<td>Intel i5-760</td>
<td>4</td>
<td>2.8</td>
<td>8192</td>
<td>3</td>
</tr>
<tr>
<td>Server-2</td>
<td>Intel i5-4590</td>
<td>4</td>
<td>3.3</td>
<td>6144</td>
<td>8</td>
</tr>
<tr>
<td>Server-3</td>
<td>Intel i7-3770</td>
<td>8</td>
<td>3.4</td>
<td>8192</td>
<td>16</td>
</tr>
</tbody>
</table>

Fig. 3 shows the calculated normalized powerMark $npwM$ values with $rc = rcCPU$ of Server-1, 2, and 3 based on Eq. (8). Note that the difference of normalized powerMark $npwM$ values among servers of Fig. 3 is bigger than the one of powerMark $pwM$ values of Fig. 2. The Server-3 has the smallest value of $npwM$, which means that this server has the best power efficiency among three servers, and this is in concordance with the results in Fig. 2. Based on result curves in Figs. 2 and 3, we conclude that our proposed metric powerMark is simple and useful to represent the relative power efficiency of heterogeneous cloud datacenters in practice.

3.2.3 Dynamic right sizing model

To achieve power-proportional cloud datacenter which consumes power only in proportion to the workload, we consider DRS approach which adjusts the number of active servers by turning them on or off dynamically [3]. Obviously, there is no need to turn all the servers in cloud datacenter on when the total workload is low. In DRS approach, the state of servers which have no running applications can be transit to the power saving mode (e.g., sleep or hibernation) in order to avoid wasting energy as shown in Fig. 4. In order to successfully deploy DRS approach onto our system, we should consider the switching overhead for adjusting the number of active servers (i.e., for turning sleep servers on again).

The switching overhead includes: (1) additional energy consumption by transition from sleep to active state (i.e.,
awnen transition); (2) wear-and-tear cost of server; (3) fault occurrence by turning sleep servers on when toggled is high [3]. We only consider the energy consumption as the overhead by DRS execution. Therefore, we define a constant $\alpha_{\text{awaken}}$ to denote the amount of energy consumption for awaken transition of servers. Then the total energy consumption $e_j(t)$ of cloud datacenter $DC_j$ at time $t$ is defined as follows,

$$
e_j(t) = \sum_{v_{\text{active}} \in s} \left( \rho_{\text{grid}} \cdot \frac{\sum_{v_{\text{CRB}} \in C} \sum_{x_j \in d} t_{\text{dc}} \cdot x_j^k(t) \cdot \frac{\rho_{\text{dsk}}}{\rho_{\text{bw}}} \cdot m_j(t)}{\text{scp}_v \cdot m_j(t)} \right) + \alpha_{\text{awaken}} \cdot (m_j(t) - m_j(t-1))^+, \forall DC_j \in DC, \forall t \tag{9}\n$$

where $(x)^+ = \max(0, x)$. The first term of the right hand side in (9) represents an energy consumption for using servers to serve VM requests allocated to the cloud datacenter $DC_j$ at time $t$ and the second term represents an energy consumption for awaken transition of sleeping servers. Especially, the second term implies that a frequent changes in the number of active servers might increase the undesirable waste of energy. Note that the overhead by transition from active to sleep state (i.e., asleep transition) is ignored in our model since a time required for asleep transition is relatively short compared to the one for awaken transition.

### 3.3. The cloud datacenter cost minimization problem

We build a cost model based on workload model and energy consumption model proposed in Sections 3.1 and 3.2. We focus on minimizing the total cost including three sub costs: (1) energy cost; (2) performance degradation cost; (3) bandwidth cost. In our energy cost model, to simplify it, we assume that the price for renewable energy usage is zero in this paper (strictly, the real price is not zero since the investment expense and the maintenance expenditure for renewable energy generation equipments are required to deploy the renewable energy generator onto the cloud datacenter). Generally, the price of power grid and the capacity of the generated renewable energy are time-varying according to the electricity market and the location of the cloud datacenter [17,19]. We use $c_j^p(t)$ to denote the energy cost of cloud datacenter $DC_j$ at time $t$ as follows,

$$
c_j^p(t) = \rho_{\text{grid}}(t) \cdot \left( e_j(t) - r_{\text{dsk}}(t) - r_{\text{bw}}(t) \right)^+, \forall DC_j \in DC, \forall t \tag{10}\n$$

where $\rho_{\text{grid}}(t)$ denotes the time-varying price of power grid at time $t$. Next, the performance degradation cost can be determined by the total performance degradation of the cloud datacenter based on Eq. (6). When we use $\rho_{\text{perf}}$ to denote the constant of penalty price for performance degradation, then the performance degradation cost of the cloud datacenter $DC_j$ at time $t$, $c_j^{\text{perf}}(t)$ is given by,

$$
c_j^{\text{perf}}(t) = \rho_{\text{perf}} \cdot D_j(t), \forall DC_j \in DC, \forall t \tag{11}\n$$

Note that $\rho_{\text{perf}}$ is a constant in contrast with $\rho_{\text{grid}}(t)$ which is dynamically changed according to time. Third, the bandwidth cost is the one for the data transfer between the cloud service users closed to CRBs and VM requests allocated on servers in cloud datacenters. Obviously, different links between CRB and cloud datacenter require the different bandwidth cost. The bandwidth cost is determined by the network distance (e.g., hop distance) and the transferred data size. We use $c_j^{\text{bw}}(t)$ to denote the bandwidth cost of the cloud datacenter $DC_j$ at time $t$ as given by,

$$
c_j^{\text{bw}}(t) = \sum_{v_{\text{CRB}} \in C} \sum_{v \in DC} \rho_{\text{bw}}(t) \cdot \left( x_j^k(t) \cdot ds_j \right), \forall DC_j \in DC, \forall t \tag{12}\n$$

where $\rho_{\text{bw}}^j$ denotes is the bandwidth cost coefficient of the communication link between the cloud request broker $CRB_i$ and the cloud datacenter $DC_j$, and $ds$ denotes the transferred data size of VM request with flavor type $F_k$.

---

**Fig. 2.** Normalized power consumption results of Server-1, 2, 3 under execution of Montage applications as an example.

**Fig. 3.** Results of normalized powerMark of Server-1, 2, 3 based on Eq. (7).
Obviously, as the hop distance between the CRBi and DCj grows longer, \( \rho_{bw}^{ij} \) is also increased. (12) implies that the allocation of more VM requests to the cloud datacenter which is far away (i.e., has long hop distance) from the source CRB increases the bandwidth cost \( cbw_j(t) \). It is advantageous for bandwidth cost saving to allocate VM requests to the nearest cloud datacenter to their source CRB.

Consequently, we focus on minimizing the total cost of whole cloud datacenters through our proposed approach of the EEE-VMA solver. We use \( c_{total}(t) \) to denote the total cost of whole cloud datacenters at time \( t \), which includes the energy costs, the performance degradation cost and the bandwidth cost. Then we define the objective function \( f_{\text{EEE-VMA}}(S(t)) \) to calculate the total cost determined by the solution \( S(t) = \{ X(t) = \{ x_{1,1}^1(t), x_{1,1}^2(t), \ldots, x_{DC,F}^{X \in \mathcal{S}_f, B \in \mathcal{B}}(t) \}, M(t) = \{ m_1(t), m_2(t), \ldots, m_{DC}(t) \} \} \) at time \( t \) as belows,

\[
 f_{\text{EEE-VMA}}(S(t)) : c_{total}(t) = \sum_{V_{DC}, B \in \mathcal{B}} (c_f(t) + c_{bw}(t)) s.t. (1) - (5)
\]

To solve this function, we propose the EEE-VMA approach based on GA in order to find an approximated optimal solution for VM request allocation. In next section, we describe our algorithm in detail.

### 3.4. Evolutionary energy efficient virtual machine allocation

In this section, we propose EEE-VMA approach based on GA which is one of efficient metaheuristics to solve a complex optimization problem. In order to successfully deploy GA onto the EEE-VMA, we should define accurate strategies for GA and set their appropriate parameters. To do this, we consider five basic steps of GA as follows.

#### 3.4.1. Encoding scheme

A chromosome (i.e., individuals in the population) features the solution \( S(t) \) of our datacenter management scheme in cloud datacenters. The format of genes in the chromosome is described as an integer value. The chromosome includes multiple genes which are divided into two parts: the first part is for the VM request allocation plan \( X(t) \); and the second part is for the DRS plan \( M(t) \).

In the first part of the chromosome, gene values represent the number of VM requests allocated to the cloud datacenter at time \( t \). For example, as shown in Fig. 5, the gene \( 1_{210} \) in the chromosome represents \( x_{1,1}^1(t) = 210 \) which means that the number of allocated VM requests with flavor type \( F_1 \) from the CRB1 to DC1 is 210 at time \( t \). In the second part of the chromosome, gene values represent the number of active servers in the cloud datacenter at time \( t \). In Fig. 5, the gene \( CRB1 \times F1 \times DC1 + 3200 \) in the chromosome means that the number of active servers in the cloud datacenter DC1 is 3200 at time \( t \).

#### 3.4.2. Initialization

In the first generation \( g = 1 \), GA in the EEE-VMA approach begins with randomly generated populations according to submitted VM requests at each CRB. To reduce the computation time for GA execution, the range of value for each gene can be predetermined based on (2), and (5).

#### 3.4.3. Evaluation

In EEE-VMA approach, we use (13) to evaluate the performance of each chromosome (i.e., solution) in the population. The fitness value of a chromosome is inversely
related to the cost value. The higher fitness function value implies the higher performance of the chromosome. Note that if a certain chromosome violates any of constraints (1)-(5), then its cost value is counted as “positive infinity”. Otherwise, the chromosome which has the smallest cost value among all the chromosomes in the generated population at \( g = g_{\text{Max}} \) (a max step of generation) is chosen as an optimal solution \( S^*(t) \) finally.

3.4.4. Selection

There are several candidate schemes for selection of appropriate solutions in GA. Especially, we adopt the roulette-wheel selection which determines the probability of each chromosome to be chosen according to their fitness function value. This scheme tends to preserve superior solutions and evolve them in the next generation [30].

3.4.5. Crossover

The role of crossover is to generate offspring from two parents by cutting certain genes of parents and conducting recombination of each gene fragment. The offspring inherits characteristics of each parent. Our EEE-VMA approach adopts the simple crossover scheme by which the first half of the first parent and the second half of the second parent are aggregated to genes of their offspring. Note that crossover has to be conducted separately on each part of the chromosome since it has two parts of the VM request allocation plan and the DRS plan.

3.4.6. Mutation

It is necessary to ensure the diversity of the generated population at all generation steps in order to avoid local minima problem in GA. At each generation step \( g \), gene values constituting chromosome can be modified randomly according to the predetermined probability \( p_{\text{mut}} \). If \( p_{\text{mut}} \) is too large, the superior genes inherited from parents can be lost, otherwise, the diversity of population might be lower when \( p_{\text{mut}} \) is too small. It is important to determine the appropriate \( p_{\text{mut}} \) in order to maintain the diverse and superior population. However, we do not consider this issue since it is out of scope in this paper.

The proposed GA for EEE-VMA approach is described in Algorithm 1. In order to get the near optimal solution \( x_{t}^{i} \) of datacenter management for VM requests arrived at CRBs at time \( t \), the state information of servers in all the datacenters at time \( t - 1 \) is required. If the current time \( t = 0 \), then we assume that the previous state of all the servers is active (i.e., all the servers are switched on). In line 02, we initialize the candidate population \( \text{cand}_t \) (\( g = 1 \)) with the population size \( ps \) (represents the limit number of chromosomes in the population) randomly. The population \( \text{cand}_t \) evolves until \( g = g_{\text{Max}} \) to generate the final population \( \text{pop}_t = g_{\text{Max}}(t) \) to search the near optimal solution \( x_{t}^{f} \) as shown from line 03 to 29. Two parent chromosomes \( S^i(t) \) and \( S^j(t) \) are released from \( \text{temp}_t \) to produce an offspring \( S^k(t) \) from line 06 to 10. In line 11, each offspring in the set \( \text{offspring}_t \) is mutated by modifying each gene according to the probability \( p_{\text{mut}} \) to maintain the diversity of the population. We check constraints (i.e., Eqs. (1)-(5)) of each solution \( S^k(t) \) in \( \text{cand}_t \) are whether violated or not in line 14. If they are violated, the corresponding solution has the cost value counted as “positive infinity”. Otherwise, the objective function value of the solution is calculated through \( \hat{f}_{\text{EEE-VMA}}(\cdot) \) in line 17. If we find the solution \( S^k(t) \) having the objective function value \( c^k(t) \) which is smaller than the predetermined threshold value \( c^\text{thr} \), then we count \( S^k(t) \) as the near optimal solution \( S^*(t) \) and the algorithm 1 is finished. Otherwise, chromosomes to be preserved until the next generation are chosen from the current population through the Iterative Roulette-wheel Selection (Algorithm 2) procedure as shown in line 23. When reaching the max step of generation \( g_{\text{Max}} \), the near optimal solution \( S^*(t) \) having minimum cost value of \( \hat{f}_{\text{EEE-VMA}}(\cdot) \) in \( \text{pop}_t = g_{\text{Max}}(t) \) is found and returned to the EEE-VMA solver in our system in line 30.
The IterativeRwSelection for Algorithm 1 is described in Algorithm 2. From line 02 to 07, the cost values which have “positive_infinity” are released from \( \text{cand}_C^g(t) \) and added to \( \text{illeg}_C^g(t) \) since solutions which do not violate constraints (i.e., (1)–(5)) are preferentially considered as candidates to be preserved until the next generation. The maximum (worst) and minimum (best) objective function values are found from \( \text{cand}_C^g(t) \) in line 09 and 10. Fitness values of each solution are calculated as shown in line 13. Through this equation, the fitness value of the best solution comes out as \( \alpha \) times of one of the worst solution. The selection pressure which represents the difference between fitness values of superior solutions and inferiors is increased as \( \alpha \) is increased. The sum of fitness values of each solution, \( SF \) is updated in line 15.

Algorithm 1.

Input : VM requests arrived at CRBs at time \( t \), \( \Lambda(t) \)
Output : Approximated optimal solution of resource allocation at time \( t \), \( S^*(t) \)

01: \( g = 1, g : \) generation step
02: initialize randomly \( \text{cand}_\text{pop}^g(t) = \{ S^{g,1}(t), S^{g,2}(t), ..., S^{g,ps}(t) \} \)
\[ ps : \text{limit size of population, even number} \]
03: while \( g \leq g\text{Max} \) do, \( g\text{Max} : \text{max step of generation} \)
04: \( \text{temp}_\text{pop}^g(t) = \text{copy}(\text{cand}_\text{pop}^g(t)) \)
05: \( k = ps + 1 \)
06: while \((\text{temp}_\text{pop}^g(t) \neq \text{empty}) \) do
07: release two arbitrary chromosomes \( S^{g,i}(t), S^{g,j}(t) \) from \( \text{temp}_\text{pop}^g(t) \)
08: \( \text{offspring}^g(t) = \cup \{ S^{g,k}(t) = \text{crossover}(S^{g,i}(t), S^{g,j}(t)) \} \)
09: \( k++ \)
10: end while
11: \( \text{offspring}^g(t) = \text{mutation}(\text{offspring}^g(t), pr_m) \)
12: \( \text{cand}_\text{pop}^g(t) = \cup \text{offspring}^g(t) \)
13: for \forall S^{g,i}(t) \in \text{cand}_\text{pop}^g(t) = \{ S^{g,1}(t), ..., S^{g,ps}(t), S^{g,ps+1}(t), ..., S^{g,ps+\text{Max}}(t) \} \)
14: if \( S^{g,i}(t) \) violates Eq. (1) ~ (5) then
15: \( c^{g,i}(t) = \text{positive_infinity} \)
16: end if
17: else \( c^{g,i}(t) = f_{\text{EEE-VM}}(S^{g,i}(t)) \)
18: if \( c^{g,i}(t) \leq c^{thr} \) then
19: \( S^*(t) = S^{g,i}(t) \) and exit;
20: end if
21: \( \text{C}^g(t) = \cup \{ c^{g,i}(t) \} \)
22: end for
23: \( \text{chIdxSet}^g(t) = \text{IterativeRwSelection}(\text{C}^g(t), ps) \)
24: for \( \forall i \in \text{chIdxSet}^g(t) \)
25: \( \text{pop}^g(t) = \cup \{ S^{g,i}(t) \in \text{cand}_\text{pop}^g(t) \} \)
26: end for
27: \( \text{cand}_\text{pop}^{g+1}(t) = \text{pop}^g(t) \)
28: \( g++ \)
29: end while
30: \( S^*(t) = \arg\min_{\forall S^{g=gMax,i}(t) \in \text{pop}^{g=gMax}(t)} \{ f_{\text{EEE-VM}}(S^{g=gMax,i}(t)) \} \)
Algorithm 2.

Input: The set of cost values at generation $g$ at time $t$, $C^g(t)$
The predetermined population size $ps$

Output: The set of selected chromosome indices, $\text{chIdxSet}^g(t)$

01: $\text{cand}_C^g(t) = \text{copy} \left( C^g(t) = \{ c^{g,1}(t), \ldots, c^{g,ps}(t), c^{g,ps+1}(t), \ldots, c^{g,ps+r}(t) \} \right)$

02: for $\forall \text{cand}_C^{g,i}(t) \in \text{cand}_C^g(t)$ do

03: if $\text{cand}_C^{g,i}(t) = \text{positive_infinity}$ then

04: $\text{illeg}_C^g(t) = \cup \{ \text{cand}_C^{g,i}(t) \}$

05: $\text{cand}_C^g(t) = \setminus \{ \text{cand}_C^{g,i}(t) \}$

06: end if

07: end for

08: while $|\text{chIdxSet}^g(t)| < \min \{ ps, |\text{cand}_C^g(t)| \}$ do

09: $\text{cand}_C_{\max}^g(t) = \max(\text{cand}_C^g(t))$

10: $\text{cand}_C_{\min}^g(t) = \min(\text{cand}_C^g(t))$

11: $\text{sumOfFitness SF} = 0$

12: for $\forall \text{cand}_C^{g,i}(t) \in \text{cand}_C^g(t)$

13: $f^{g,i}(t) = \left( \text{cand}_C_{\max}^g(t) - \text{cand}_C^{g,i}(t) \right) + \frac{\left( \text{cand}_C_{\max}^g(t) - \text{cand}_C_{\min}^g(t) \right)}{\alpha - 1}$, $\alpha > 1$

14: $FV^g(t) = \cup \{ f^{g,i}(t) \}$

15: $SF += f^{g,i}(t)$

16: end for

17: $\text{selectionPoint SP} = \text{random}(0, sf)$

18: $\text{cumulativeSum QS} = 0$

19: for $\forall f^{g,i}(t) \in FV^g(t)$

20: $QS = QS + f^{g,i}(t)$

21: if $SP \leq QS$ then

22: $\text{chIdxSet}^g(t) = \cup \{ i \}$

23: $FV^g(t) = \setminus \{ f^{g,i}(t) \}$

24: end if

25: break;

26: end for

27: end while

28: while $|\text{chIdxSet}^g(t)| < ps$ do

29: release $\text{cand}_C^{g,i}(t)$ randomly from $\text{illeg}_C^g(t)$

30: $\text{chIdxSet}^g(t) = \cup \{ i \}$

31: end while

32: return $\text{chIdxSet}^g(t)$

This value represents the total size of roulette-wheel, and each solution is assigned to spaces on the roulette-wheel. That means that the selection probability of each solution is proportional to the size of their assigned spaces. The selection procedure of the roulette-wheel is described from line 19 to 26. At every step, the cumulative summation $QS$ is updated according to the fitness value $f^{g,i}(t)$ in $FV^g(t)$. If the selection point $SP$ is smaller than $QS$ with the latest update by $f^{g,i}(t)$ (i.e., $\sum_{k=1}^{i-1} f^{g,k}(t) \leq SP \leq \sum_{k=1}^{i} f^{g,k}(t)$), then the index $i$ of $S^g(i)$ in $\text{cand}_C^g(t)$ is added to $\text{chIdxSet}^g(t)$. If the total
number of chosen chromosomes by the roulette-wheel procedure is not sufficient (i.e., the cardinality of \( \text{chIdxSet}\_g\_t(t) \) is smaller than the predetermined size of the population \( p_S \)), then we supplement \( \text{chIdxSet}\_g\_t(t) \) by randomly putting out the indices of chromosomes from \( \text{illeg}\_C\_g\_t(t) \). After all the procedures are finished, then \( \text{chIdxSet}\_g\_t(t) \) is finally returned to the Algorithm 1 in line 32.

4. Performance evaluation

In this section, we evaluate the performance of our proposed EEE-VMA approach based on both of simulation analysis and experiments on real testbeds. To highlight the benefits of our design for renewable and QoS aware workload management, we perform a numerical simulation based on real-world traces of renewable energy capacity.

4.1. Dynamic capacity of renewable energy

We consider three locations to employ the raw data in order to build a capacity trace of renewable energy including wind energy: Oak Ridge National Lab (Eastern Tennessee); University of Arizona (Tucson, Arizona); University of Nevada, Las Vegas (Paradise, Nevada) \cite{11,33}. We obtain the capacity traces of wind energy at those three locations based on \cite{33} that collects data of wind speed every day. The capacity traces of each location at EST
05:20–17:54 on September 9, 2015 are shown in Fig. 6. Fig. 6(a), (c), and (e) shows the wind speed of each location and we can find that it is fluctuated a lot even during short period. We assume that each generator has 30 wind turbines and the amount of generated wind energy is estimated based on the wind power prediction scheme from [34]. Then the curves of the amount of available wind energy are shown in Fig. 6(d), (e), and (f).

4.2. Energy price description

As mentioned earlier, only the grid power price is considered since we assume that the renewable energy price is free. The grid power price is dynamically changed according to the electricity consuming time. We use the electricity price information in our simulation based on the real time pricing during 24 h in the electricity market which is shown in Fig. 7 [23,35]. Note that the electricity price is high from 6 a.m. to 14 p.m., and from 19 p.m. to 21 p.m. The electricity usage is usually increased during these periods due to the needs of industrial and household appliances. In our simulation, each cloud datacenter randomly has the electricity pricing curve among datacenter 1, 2, and 3 in Fig. 7.

4.3. Cloud resource description

The total number of multiple cloud datacenters is nine, and each datacenter owns $2 \times 10^3$ homogeneous servers in this paper. In perspective of VM instance specifications, we adopt the policy of Amazon EC2 Web Services (AWS), our cloud datacenters support the set of flavor types $\mathcal{F} = \{F_1 = \text{CPU} = 2\text{ cores, mem} = 4 \text{ GB}, F_2 = (4, 8), F_3 = (8, 16), F_4 = (16, 32)\}$ and each VM request has an arbitrary flavor type $F_k \in \mathcal{F}$ randomly [29]. As mentioned in Section 3, each cloud datacenter has heterogeneous server architecture, they have different powerMark value in the range of 200–500 based on results in Fig. 3.

4.4. Workload scenario

Our considered workload includes two parts: the number of VM requests $\Lambda(t)$, and their required resource utilization $\nu(t)$ at time $t$. The number of VM requests $\Lambda(t)$ can be defined as from $3 \times 10^3$ to $100 \times 10^3$ in this paper. Obviously, as $\Lambda(t)$ is increased, both of energy consumption and performance degradation are also increased.
In perspective of resource utilization, we only consider the resource component $rc = rc_{CPU}$ and ignore the resource component $rc_{mem}$ since the energy consumption and performance degradation caused by $rc_{mem}$ is negligible compared to ones by $rc_{CPU}$. We use the real traces of CPU resource utilization measured by the monitoring module with several benchmark applications on the physical machines. Fig. 8 shows the CPU resource utilization of running benchmarks including a mixture of pbzip2, iozone3, netperf on VM instances.

Fig. 10. Active server ratio of cloud datacenters including servers with heterogeneous (a) and homogeneous power efficiency (b).

Fig. 11. Performance degradation of CPU contention by co-located VM requests in Server-1 (a), 2 (b), and 3 (c).
4.5. GA Parameters for EEE-VMA approach

We consider a population size $ps$ with a range from $10^2$ to $10^4$, the max step of generation $gMax$ in the range of 100 to 1000, and the mutation probability 0.001, 0.005 and 0.01 in the Algorithm 1 and 2. As the parameters such $ps$ and $gMax$ are increased, the performance of the derived solution is increased, but its computation need is also deteriorated.

4.6. Traditional resource management schemes

To demonstrate that our proposed approach outperforms existing resource management schemes, we compare the EEE-VMA approach to both of VM consolidation and VM balancing based allocation approaches. The VM consolidation approach tries to pack multiple VM requests as many as possible in the common physical server. This scheme tends to reduce the number of active servers. Therefore, the energy saving performance is increased, while the performance degradation is deteriorated. In contrast, the VM balancing approach splits VM requests over multiple cloud datacenters. This scheme avoids the performance degradation of resource contention by VM request co-location, but causes the large energy consumption due to a lot of active servers.

Figs. 9, 10, and 11 show the performance of our proposed EEE-VMA approach and existing VM balancing and consolidation approaches at $ps = 10^2$, $gMax = 500$, and mutation probability is 0.001. Fig. 9 shows the total cost in Eq. (13) of the VM balancing, VM consolidation and our proposed EEE-VMA approach at different workload offered load level. Fig. 9(a) shows the curves of total cost of all the approaches assuming that each cloud datacenter has heterogeneous power efficiency. Our proposed approach achieves the improvements of the cost saving performance about 8% and 53% compared to VM consolidation and VM balancing approaches, respectively. However, the difference of the cost saving performance between traditional approaches and our EEE-VMA approach in Fig. 9 (b) assuming that each cloud datacenter has homogeneous power efficiency is relatively small compared to the one in Fig. 9(a). The EEE-VMA approach achieves the improvements of the cost saving performance about 10% and 15% compared to VM consolidation and VM balancing approaches, respectively. Note that our EEE-VMA approach further improves the performance of energy saving in the heterogeneous cloud datacenters since it uses powerMark value which can rank the power efficiency of each cloud datacenter to maximize the energy efficiency of resource allocation. However, our proposed approach still has the better performance than ones of existing approaches even under the assumption of homogeneous power efficiency of each cloud datacenter. Fig. 10 shows the active server ratio of cloud datacenters by our EEE-VMA approach and existing resource management approaches. In Fig. 10(a), the average active server ratio of the EEE-VMA approach is under 30%, while the ones of VM balancing is closed to 60%. Our EEE-VMA approach considers both of energy consumption and performance degradation of VM requests, while the VM balancing only focuses on the performance degradation. Note that the energy saving performance of VM consolidation is worse than the one of the EEE-VMA approach even though the VM consolidation focuses on the energy consumption of cloud datacenters. This is because our EEE-VMA approach allocates VM requests to power efficient cloud datacenters preferentially based on their powerMark values, while the VM consolidation randomly assigns VM requests to cloud datacenters. In Fig. 10(b), the active server ratio of VM consolidation is lower than the one of our EEE-VMA approach, this is because the VM consolidation approach only focuses on the energy consumption of cloud datacenter, but the EEE-VMA approach avoids unacceptable performance degradation of running VM requests through Eq. (11).

Fig. 11 shows the performance degradation of allocated VM requests in each cloud datacenter by the EEE-VMA, VM consolidation, and VM balancing approaches based on the server types. The performance degradation is calculated by Eq. (6). In the perspective of performance degradation, the VM balancing approach outperforms the others including our proposed EEE-VMA approach. The VM balancing tries to spread submitted VM requests over whole cloud datacenters as fair as possible, therefore the CPU resource contention of co-located VM requests can be minimized. In Server-1 type, the performance degradation of VM balancing is lower than the ones of the EEE-VMA approach and the VM consolidation by 40% and 60%, respectively. In Server-2 type, the performance degradation of VM balancing is lower than ones of the EEE-VMA approach and the VM consolidation by 55% and 62%, respectively. Finally, in Server-3 type, the VM balancing approach can improve the performance degradation about 39% and 55% compared to the EEE-VMA approach and the VM consolidation, respectively.

5. Conclusions

In this paper, we introduced the EEE-VMA approach for greening cloud datacenters with renewable energy generators. We proposed a novel energy efficient metric powerMark to classify the power efficiency of heterogeneous servers in cloud datacenters and built a considerate cost model considering switching overheads in order to reduce efficiently the energy consumption of servers without a significant performance degradation by co-located VM requests and DRS execution. We deployed the iterative roulette-wheel algorithm for GA of the EEE-VMA approach in order to solve the complex objective function of our cost model. Through various experimental results based on simulation and Openstack platform justify that our proposed algorithms are supposed to be deployed for prevalent cloud data centers. In the perspective of total cost, our EEE-VMA approach can improve the average cost by 28% compared to existing resource management schemes at all the workload level. With the increase of the computation investment for GA in EEE-VMA approach, our proposed approach can get arbitrarily close to the optimal value.
Acknowledgments

This work was supported by 'The Cross-Ministry Giga KOREA Project' of the Ministry of Science, ICT and Future Planning, Korea [GK13P0100, Development of Tele-Experience Service SW Platform based on Giga Media].

References

[27] Amazon Web Services, [https://aws.amazon.com].
[37] Amazon Web Services, [https://aws.amazon.com].
[40] Measurement and Instrumentation Data Center (MIDC), [http://www.nrel.gov/midc/].