Service-Oriented Virtual Machine Placement Optimization for Green Data Center

Fan-Hsun Tseng[1] **· Chi-Yuan Chen**[2] **· Li-Der Chou**[1] **· Han-Chieh Chao**[2,3,4] **· Jian-Wei Niu**[5]

Published online: 9 May 2015

Abstract The first service-oriented virtual machine (VM) placement for green data center is designed in this work. Integer Linear Programming (ILP) is the problem design basis. The Tree algorithm is proposed to place VM role instances at the lowest communication cost, economizing the construction cost with fewer physical servers. Another Forest algorithm is also proposed for balancing the computation load between the physical machines. Both of the proposed algorithms are formulated on the graph theoretic technique and evaluated and analyzed using the Best Fit algorithm in the simulations. Although the total power consumption and average utility of both proposed algorithms are slightly impaired, the unnecessary outbound communication cost is significantly eliminated and decreased, especially in the immense number of services. The results show that the proposed Tree and Forest algorithms provide lower communication cost than the Best Fit algorithm, and achieve green communications for large scale environment such as cloud or green data center.

Keywords Service-oriented placement · Virtual machine · Data center · Integer linear programming · Load balancing

✉ Han-Chieh Chao
hcc@niu.edu.tw

Fan-Hsun Tseng
fanhsuntseng@ieee.org

Chi-Yuan Chen
chiyuan.chen@ieee.org

Li-Der Chou
cld@csie.ncu.edu.tw

Jian-Wei Niu
niujianwei@buaa.edu.cn

1 Department of Computer Science and Information Engineering, National Central University, Taoyuan, Taiwan

2 Department of Computer Science and Information Engineering, National Ilan University, I-Lan, Taiwan

3 Department of Electrical Engineering, National Dong Hwa University, Hualien, Taiwan

4 School of Information Science and Engineering, Fujian University of Technology, Fujian, China

5 School of Computer Science and Engineering, Beihang University, Beijing, China

1 Introduction

Virtualization technology was announced by the International Business Machines (IBM) Corporation in 1971 [1]. Many researchers have regarded it as the first step in migrating existing services into the cloud environment [2]. A cloud service is executed using various virtual machines (VMs). These VMs are packaged in a physical server or spread over several physical machines. Due to the increased popularity of cloud computing researchers have intensely investigated virtualization technology [3]. The renowned market research company Gartner [4] predicted this technique might be the first of ten important technologies in 2009 [5]. As mentioned, the virtualization technique is regarded as the principal step, fueling the evolution of cloud computing since 2009. Cloud computing was noted the most important technology Gartner predicted in 2010 [6] and 2011 [7], and it remained on the prediction list in 2012 [8]. According to state-of-the-art cloud development, it is obvious that virtualization technology must be extensively discussed and investigated.

Various virtual technique issues have been widely investigated widely. The problems that come from virtualization can be divided into two categories, the preliminary construction

[9–11] and the subsequent management [12–14]. For instance, the resource allocation [15] and provisioning problem [16] for VMs placement in the data center. Researchers conducted an optimization strategy between the deployment cost and system performance. Researchers proposed a blind scheduling algorithm [17] to maintain the fairness at each time point in mobile media cloud, and further proposed a blind online scheduling algorithm [18] to achieve asymptotic optimality. Other researchers studied the VM migration condition and strategy, such as [19, 20]. When physical machine resources are exhausted the virtual machine migrates from the current infrastructure to other available machines. However, the existing services and tasks should be continued uninterrupted. A novel technology for VM management called *Live Migration* [21] has become well-known in recent years.

This paper investigates the Service-Oriented VM Placement Optimization problem (or simply SOVMPO throughout this paper) defined based on Integer Linear Programming (ILP). We propose two algorithms named the Tree and Forest algorithms for solving the SOVMPO problem. Both proposed algorithms are formulated based on the graph theoretical technique. The Forest algorithm is proposed for balancing the traffic load between VMs, but wastes construction and communication cost. The Tree algorithm is designed to minimize the communication cost between VMs that belong to the same service. This research has three major contributions: 1) we formulate the service-oriented VM placement problem which is called SOVMPO and propose two algorithms based on graph theory for solving this problem; 2) the calculation time for both algorithms increases linearly with the increased number of services; 3). The Tree algorithm reduces the total power consumption, enhances the average utility of physical servers and reduces the communication cost as well.

This paper is organized as follows. Section II introduces current VM placement research and compares the different approaches. Section III defines the ILP model for optimizing the VM placement problem and explains the proposed algorithms. In section IV we present the optimization results and discuss the simulation results. Conclusions from this research are drawn in section V.

2 Background and related works

The VM data center related problem was studied extensively in past literature, such as VM placement, resource allocation, resource provisioning, minimizing construction cost and network traffic load. This paper focuses on minimizing the communication cost [22, 23] based service-oriented virtual machine placement in data center networks. The data center network infrastructure and communication cost are discussed and studied. The differences in VM role instances are introduced because a job or service may be executed using several VM role instances in cloud computing.

2.1 Data center infrastructure

The Cisco data center network infrastructure is a three-tier architecture [24] that includes the core common services, aggregation and server access. The data center infrastructure is shown in Fig. 1, composed of a tree structure according to the current data center architecture diagram. The network architecture and topology effect on the network resource in data center [25]. The three-layer architecture is adopted in most of enterprises and data centers. Most of intra-data traffic traverses within switch and physical server, thereby we consider and investigate the service-oriented VM placement. Although the fat-tree architecture is mentioned and studied in recent researches [26, 27], we adopted the original three-tier architecture structured as a tree topology in the data center infrastructure. Because this architecture is more suitable for fitting the real data center environment data center infrastructure modification is unneeded.

The top layer is the core router which connects with all middle-tier aggregations but does not connect with other core routers. The middle layer is the aggregation that only connects to one or two edge switches for access. This implies that the switches and physical servers under the middle-tier aggregation comprise a local area network (LAN). The bottom layer is the edge switch, with one or more physical servers accessing the edge switch for routing data information. Without loss of generality, the communication cost for two arbitrary physical servers in the same LAN is lower than two separate LANs. In other words, the cost of inbound communication within a LAN is lower than the cost of outbound communication between two LANs.

2.2 VM role instances

The identity of VM role instances is determined according to their working categories. For instance, the different VM role instances are in charge of different works. Windows Azure [28] has two VM roles which are web role and worker role.

The first type of VM instance is the web role instance. It executes the interactions between web applications or HTTP services. The web role instance is responsible for the communication between user and the operating process. On the other hand, the worker role instance is a background process that executes the tasks or jobs in Windows Azure. In order to achieve the elastic computing in cloud environment an application service or job may be executed by many VM instances. When the computing resource is insufficient in the same service type, the number of worker role instances can be increased and scaled up dynamically. Therefore, the number of worker role instances is always greater than the web role

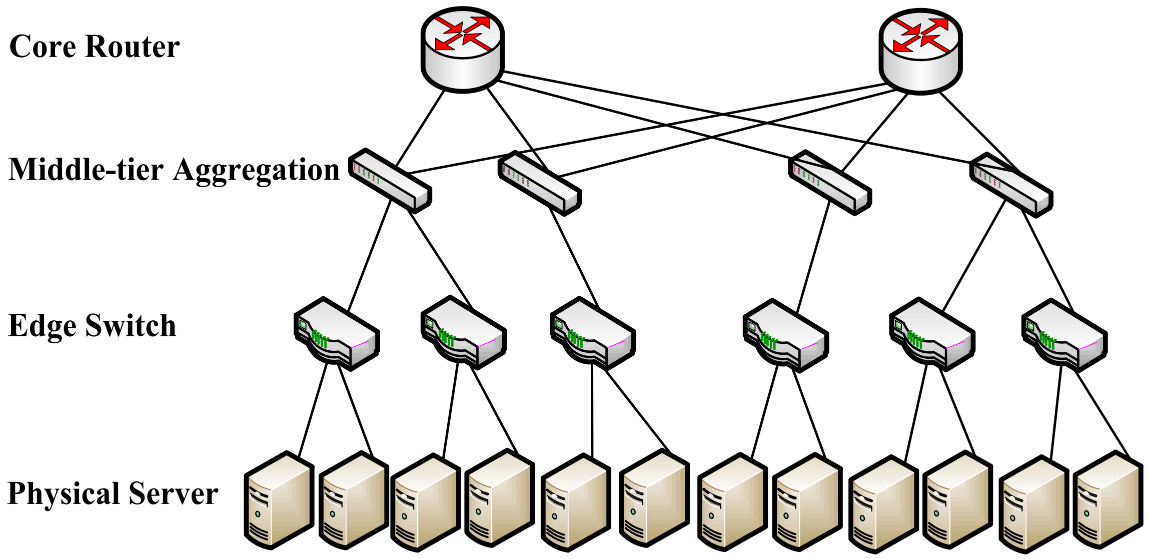

Fig. 1 Data center network infrastructure

instances. Two types of VM instances are considered by this research in the SOVMPO problem. The communication cost and utility rate are calculated from these VM role instances. We assume that a service is preceded by a web role and four worker role instances.

2.3 Related works

In the existing literature some researchers focused on placing VMs in different data center architectures [26, 27, 29] while others investigated the VM placement in distinct orientations [30, 31].

The works most related to ours are [26, 27, 30, 31]. In [26, 27], the three-tier architecture in data center is also considered. However, the researchers in [26] surveyed the influences of TCP Incast and congestion notification unlike the communication cost of inbound and outbound in our work. The traffic characteristic is not covered in [26] but we eliminate the costs from the outbound communication, which minimizes the traffic loads within different LANs. Although the data center architecture is considered, two parts differentiate [27] our work. First, in spite of the considered architecture, the VL2 [32], fat-tree [26] and BCube [33] architecture are adopted in [27], but we follow the three-tier architecture of traditional data center networks in this work. Second, the proposed work in [27] is based on traffic-aware VM placement while in our work the proposed algorithms are based on service-oriented VM placement. In [30, 31] both researches are based on application-aware viz. *AppAware*. Although the proposed scheme in [30] is based on the graph theoretical technique as our work, the authors focus on the application-aware VM migration unlike the data center placement problem in our work. The VM demand, physical machine capacity and communication are considered in [30] which is similar to our work, but the authors investigated the communication frequency performance unlike the communication cost in this paper. Finally, the distinct orientation towards VM placement is considered in [31], but some parts are still different. First of all, the *AppAware* orientation is completely differentiated from our service-oriented VM placement algorithms. The authors propose an application-aware VM placement algorithm based on the convex optimization theory unlike the proposed algorithms in this paper based on ILP model and graph theoretical technique. We propose a novel VM placement scheme based on service and investigate a vital issue minimizing the inbound and outbound communication costs which are frequently neglected in other studies.

3 Proposed algorithms

3.1 Problem definition

The virtual machine optimization problem for service-oriented cloud networks is formulated based on integer linear programming (ILP). The definitions of variables used in SOVMPO are listed in Table 1.

In the cloud infrastructure a tree structure is used to connect the hardware which includes the routers, middle-tier aggregations, switches and servers. The same structure is also applied to the virtual machines within the physical servers. Therefore, we let V be a set of nodes and E be a set of edges between nodes. Given a graph $G=(V,E)$, where $V=V_1\cup V_2$ and $E=E_1\cup E_2$. It can be easily shown that $V_1\cap V_2=\varnothing$ and $E_1\cap E_2=\varnothing$. Assume that $V_1=\{z_1,\ldots,z_k\}$ with $|V_1|=k$, and $V_2=\{q_1,\ldots,q_s\}$ with $|V_2|=s$.

The nodes in V_1 are the equipment in cloud infrastructure, which can be core routers, middle-tier aggregations, edge

Table 1 Definition of symbols

Variable	Definition
$V_1=\{z_1,\ldots,z_k\}$	Set of equipment in the cloud infrastructures
$\|V_1\|=k$	Number of equipment in the cloud infrastructures
$V_2=\{q_1,\ldots,q_s\}$	Set of virtual machines in the physical servers
$\|V_2\|=s$	Number of virtual machines in the physical servers
E_1	Set of links within cloud infrastructures
E_2	Set of links between a physical server and a virtual machine
$m_{CR(MA,ES,PS)}$	The core router (or middle-tier aggregation, edge router, physical server) is placed in V_1
$x_{i,\,j}$	The available links within an equipment i and an equipment j
$n_{Web(Worker)}$	The web role (or worker role) instance is placed in V_2
$y_{i,\,j}$	The available links within a physical server i and a virtual machine j
w_{qi}	The subordination of virtual machine q_j
$\xi^k_{q_i}$	The service type of virtual machine q_j
$C^k_{i,j}$	The communication cost between virtual machine q_i and q_j in service k
$u_{i,\,j}$	The utility of a virtual machine q_j in a physical server z_i
δ_i	The maximum utility of a physical server i

switches and physical servers. Hence, $z_i\in\{0, m_{CR},m_{MA},m_{ES}, m_{PS}\}$. We then consider the available links between z_i and z_j in V_1. Let available link

$$x_{i,j}=\begin{cases}1, & if\ (z_i,z_j)\in E_1\ and\ z_i\cdot z_j\neq 0\\ 0\,, & otherwise\end{cases}. \tag{1}$$

The nodes in V_2 are the virtual machines within the physical servers, which can be classified into web and worker roles within two types of instances. That is, $q_i\in\{0, n_{Web},n_{worker}\}$. The same structure is also applied to the virtual machine, in which each cloud service contains one worker role instance and may contain multiple worker role instances.

We next consider the available links between the physical server z_i and virtual machine q_j. The available links between web role instance q_i and worker role instance q_j are considered. Let available link

$$y_{i,j}=\begin{cases}1,\, if\ \left(z_i,q_j\right)\notin\varnothing\ and\ z_i\cdot q_j\neq 0\\ 1,\, if\ \left(q_i,q_j\right)\in E_2\ and\ q_i\cdot q_j\neq 0\\ 0,\ otherwise\end{cases}. \tag{2}$$

The web role instance q_i must be executed in a physical server, defined as $w_{q_j}=w_{PS}$, where $q_j\in n_{Web}$. However, a worker role instance q_j may subordinated a physical server or a web role instance, which is defined as

$$w_{q_j}=\begin{cases}w_{PS},\ if\ q_j\ is\ nested\ in\ a\ physical\ server\\ w_{web},\ if\ q_j\ is\ nested\ in\ a\ web\ role\ instance\\ 0,\ if\ q_j=0\end{cases}. \tag{3}$$

For service-oriented virtual machine optimization, because a single service may be executed by several virtual machines, we consider the communication cost between the virtual machine instances in a service. The virtual machine service type q_j is defined as $\xi^k_{q_j}$, where $k\in N$ and $q_i\neq 0$. Moreover, the utility of a virtual machine q_j in a physical server z_i is considered in our model, which is defined as $u_{i,j}=y_{i,j}w_{q_j}$.

In this given tree graph we assume that a communication cost exists for each available link between two vertices. The communication cost between virtual machine q_i and q_j in service k is defined as

$$Cost^k_{i,j}=\begin{cases}C^k_{i,j}, & if\ x_{i,j}\cdot y_{i,j}\neq 0\\ 0, & if\ x_{i,j}\cdot y_{i,j}=0\end{cases}. \tag{4}$$

The ILP model for SOVMPO problem is defined as follows.

Minimize

$$\sum\nolimits_{i=1}^{i}\sum\nolimits_{j=1}^{j}\sum\nolimits_{k=1}^{k}C^k_{i,j} \tag{5}$$

subject to

$$z_k\geq 0,\ \text{for}\ \forall k\in V \tag{6}$$

$$q_s\geq 0,\ \text{for}\ \forall s\in E \tag{7}$$

$$x_{i,j}\in\{0,1\},\ \text{for}\ \forall i,j\in E_1 \tag{8}$$

$$y_{i,j}\in\{0,1\},\ \text{for}\ \forall i,j\in E_2 \tag{9}$$

$$\xi^k_{q_j}\geq 0,\ \text{for}\ \forall k\in N \tag{10}$$

$$\sum u_{i,j}\leq\delta,\ \text{ for}\ \forall i\in V_1\ \text{and}\ \forall k\in V_2 \tag{11}$$

$$C^k_{i,j}\geq 0,\ \text{for}\ \forall i,j\in V \tag{12}$$

3.2 Tree algorithm

The Tree algorithm is proposed to optimize the web role instances and worker role instances such that the communication cost is minimized. Some notations are clarified before introducing the Tree algorithm.

The definition of the vector q, whose i-th element denotes the placement condition on the corresponding position. From this point of view, q_x denotes the role instance in the vector q corresponding to the vertex $x\in V_2$. For example, if a web role instance is placed on the vertex x, then from the indication vector point of view, $q_x= c_{web}$. This concept can be generalized so that q_X is well-defined, where $X\subseteq V_2$. The definition of vector z, whose i-th element denotes the physical server condition on the corresponding location. For example, z_x denotes the physical server in the vector z corresponding to the vertex $x\in V_1$. Moreover, we define the $U(z)$ as the utility of the physical server z.

Let $G=(V,E)$ be a graph. Denote X and Y as subsets of vertices. Then, $f_d(G,X,Y)$ is used to generate a descendant ordered list S of the elements in X acccording to the adjacent number of vertices in Y based on G. For example, if there are 3 and 4 adjacent vertices in Y for the vertex $x_1 \in X$ and $x_2 \in X$, respectively, then x_1 is behind x_2 in the ordered list S. Note that the order of the vertices with the same degree can be arbitrary. Since S is an ordered list, $S[i]$ is denoted as the i-th vertex in S. In addition, $N(G,x)$ is defined as the set of neighboring vertices of the vertex x on the graph G. $\gamma(G,X)$ be the vertex-induced subgraph of G with the set of vertices $V \backslash X$. After introducing the functions, the descriptions of Tree algorithm are shown as Algorithm 1.

```
Algorithm 1: Tree Algorithm ( $G$, $z$, $\delta$ )
Input: $G$: underlying graph
$z$: vector indicating the placement of web and worker role instance
   $\delta$: maximum utility of physical server
01 $\Omega_1=V_1$, $\Omega_2=V_2$, $\Delta=\phi$
02 repeat
03 $S=f_d(G, \Omega_1, \Omega_2)$
04 $\Delta=\Delta \cup S[1]$
05 $\Omega_1=\Omega_1 \backslash S[1]$ and $\Omega_2=\Omega_2 \backslash N(G, S[1])$
06 until $\Omega_2=\phi$
07 $G_r=\gamma(G,\Omega_1)$, where $G_r=(V_G, E_{G_r})$
08 $q_{V_{G_r}}=n_{worker}$, where $\xi^k_{V_{G_r}}=k$
09 while $G_r$ is disconnected and $U(z)<\delta$
10 $S=f_d(G_r, V_1 \backslash \Omega_1, \Omega_1)$
11 $G_r=\gamma(G, \Omega_1 \cup S[1])$
12 $z_{V_{G_r}}=n_{web}$
13 repeat
14 $S=f_d(G_r, V_{G_r}, \Omega_2)$
15 $z_{s[1]}=n_{web}$
16 until $U(z)=\delta$
```

Lines 2 to 8 are used to determine the service type in the subset of virtual machine. The chosen subset in the Tree algorithm is based on the number of covered virtual machines given as the same service type. Since the same virtual machine service type in lines 2 to 8 could be disconnected, lines 9 to 12 are used to link the unconnected virtual machines. In line 13 to 16, we examined the utility of a physical server. The physical server utility property should be limited within δ. Because the characteristic of our problem, the linear programming in the paper is always able to have a feasible solution.

The Tree algorithm performance is dominated by lines 2 to 6. A sorting of $O(n)$ elements is required in each iteration leading to $O(n\log n)$ time complexity. $O(n)$ iterations are needed in the worst case. Therefore, $O(n^2\log n)$ time complexity is required.

3.3 Forest algorithm

According to the Tree algorithm notations we introduce only the new functions used in the Forest algorithm. Let $BFS(G,x)$ be a subgraph constructed by applying breath-first-search on the graph G rooted at the vertex x. Let $f_{BFS,\xi^k}(G)$ be the graph composed of subgraphs constructed by service type $\xi^k_{q_j}$ breadth-first-search on the graph G. The Forest algorithm descriptions are shown as Algorithm 2 after introducing the functions.

```
Algorithm 2: Forest Algorithm ( $G$, $z$, $\delta$ )
Input: $G$: underlying graph
$z$: vector indicating the placement of web and worker role instance
   $\delta$: maximum utility of physical server
01 $\Omega_1=V_1$, $\Omega_2=V_2$, $\Delta=\phi$
02 repeat
03 $S=f_d(G, \Omega_1, \Omega_2)$
04 $\Delta=\Delta \cup S[1]$
05 $\Omega_1=\Omega_1 \backslash S[1]$ and $\Omega_2=\Omega_2 \backslash N(G, S[1])$
06 until $\Omega_2=\phi$
07 $G_r=\gamma(G,\Omega_1)$, where $G_r=(V_G, E_{G_r})$
08 $i=0$, $\Omega_1'=\Omega_1$
09 repeat
10 $i=i+1$
11 randomly choose a vertex x from $V_{G_r}$
12 $G_i^s=BFS(G_r, x)$, where $G_i^s=(V_i^s, E_i^s)$
13 $\Omega_1'=\Omega_1'/x$
14 until $\Omega_1'=\phi$
15 for $j=1$ to $i$
16 $S=f_d(G_j^s, \Omega_1, \Omega_2)$
17 $z_{s[1]}=n_{web}$ and $z_{s \backslash s[1]}=n_{worker}$, where $\xi^k_{V_{G_r}}=k$,
18 while $U(z) \leq \delta$
19 $S=f_d(G_r, \Omega_1, \Omega_2)$
20 $z_{s[1]}=n_{web}$
21 until $U(z)=\delta$
```

In Algorithm 2, lines 2 to 14 are used to construct the physical server subset on which either the web role instance or work role instance are appropriately executed. Note that the selected physical server is still empty now and the web role or work role instance placement needs to be determined later. Since the physical server selected in lines 2 to 14 could constitute a forest, line 15 to 17 are used to place at least one web role in each service type. Lines 18 to 21 are used to guarantee that the utility of vertex z does not exceed the maximum utility of a physical server.

The Forest algorithm performance is dominated by lines 2 to 6. A sorting of $O(n)$ elements is required in each iteration leading to $O(n\log n)$ time complexity. $O(n)$ iterations are needed in the worst case. Therefore, $O(n^2\log n)$ time complexity is required.

3.4 Proof of NP-hard problem

In this subsection we prove that the SOVMPO problem in cloud computing environment is a NP-hard problem.

Therefore it cannot be solved in polynomial time. In other words, there is no efficient algorithm for minimizing the communication cost between VM role instances. We find the traveling-salesman problem (TSP) [34] is quite similar to our SOVMPO problem. TSP is a classical and well-known NP-hard problem. A brief description is given below. A list of cities and their pair distances, the TSP problem is to find out the shortest potential route that visits each city once and then back to the original city. Given an undirected graph $G=(V,E)$, and each edge $(u,v)\in E$ has a non-negative integer cost $c(u,v)$. Then, find a Hamiltonian cycle of G with minimum cost $\sum c(u,v)$.

After introducing the TSP problem we prove that the proposed problem is polynomial time reducible to TSP problem, that is SOVMPO $\leq_p$ TSP. The given graph G in TSP problem corresponds to the given graph G in the SOVMPO problem, which means the geographic map is fully mapped to the cloud environment, including the infrastructure, physical and virtual machines. The set of cities V is mapped to the cloud infrastructures V_1 and VM role instance V_2. The set of edges E is mapped to the communication links E_1 and E_2. Finally, the non-negative integer cost $c(u,v)$ in TSP can be matched with the proposed communication cost $C_{i,j}^{k}$ between the virtual machines q_i and q_j in service k. In other words, to find a Hamiltonian cycle of G with minimum cost is equal to finding a service-oriented VM placement method with minimum cost. From the above explanation, the proposed SOVMPO problem can be reduced to the TSP problem. Because the TSP problem is NP-hard, we conclude that the SOVMPO problem in our approach is also a NP-hard problem.

3.5 Queuing analysis

Let λ, r_{source}, and n be the rate at which messages arrive at physical servers, the rate the virtual machines emit a message, and the average number of virtual machines associated with a physical, respectively. For simplicity, we assume that the processing time at physical machines can be neglected.

Afterwards, we can regard the physical server as an M/G/1 queuing system due to the fact that the arrival process is Poisson with a rate λ. Let the random variable s and p denote the message service time and the probability that the queue is idle when a message is incoming, respectively. In addition, assume that the rate at which the physical server emits the message is $r_{PS}=\frac{1}{T_{PS}}$. As for the service time s, with the probability p it will experience an average service time $\frac{T_{PS}}{2}$ and with the probability $1-p$ its average service time is T_{PS}. After the calculation, we can obtain

$$E[s]=p\cdot\frac{T_{PS}}{2}+(1-p)T_{PS}, \tag{13}$$

and

$$E\left[s^2\right]=p\cdot\frac{T_{PS}^2}{3}+(1-p)T_{PS}^2. \tag{14}$$

In addition, we can also know from the queuing theory that $p=1-\lambda\cdot s$. Then, we have

$$E[s]=\frac{T_{PS}}{2-\lambda\cdot T_{PS}}, \tag{15}$$

and

$$E\left[s^2\right]=(1-\lambda E[s])\frac{T_{PS}^2}{3}+\lambda\cdot E[s]\cdot T_{PS}^2. \tag{16}$$

As a result, the average delay d for each message at the physical server can be calculated as

$$d=E[s]+\frac{\lambda\left(E^2[s]+E[s^2]\right)}{2\cdot(1-\lambda E[s])}. \tag{17}$$

4 Simulation results

4.1 Planning case

We illustrate the Tree and Forest algorithm with a simulated optimization case in this section. In this VM optimization case there are 12 physical servers and 4 service types, which means that 4 web role instance are placed within 12 servers. Moreover, there are 14 worker role instances in 4 service types. The network topology of the cloud infrastructure is constructed as a fat-tree structure. The VM placement results for Tree and Forest algorithms are shown in Figs. 2 and 3.

Before explaining the optimization result we have four assumptions in our optimization algorithm. First, we assume that all network equipment are connected, which means there are no disconnection links between the current layer and upper layer. For instance, the physical server can connect with the edge switch and the edge switch is able to connect to the middle-tier aggregation. The same situation exists with the middle-tier aggregation and core router. Second, we assume that there are two kinds of communication costs between virtual machines. The first is the inbound communication cost which represents that the web role and work role instances exist within the same physical server. The second one is the outbound communication cost, which means that the work role instance is placed in a different physical server than the web role instance. Then, we utilize subnetting approach to subnet data center network. The left child under a middle-tier aggregation is assigned to the same class, and the right child is another class. Therefore, we can evaluate the number of inbound and outbound communication cost by calculating the classified packets. One thing should be mentioned, the outbound communication cost outweighs the inbound

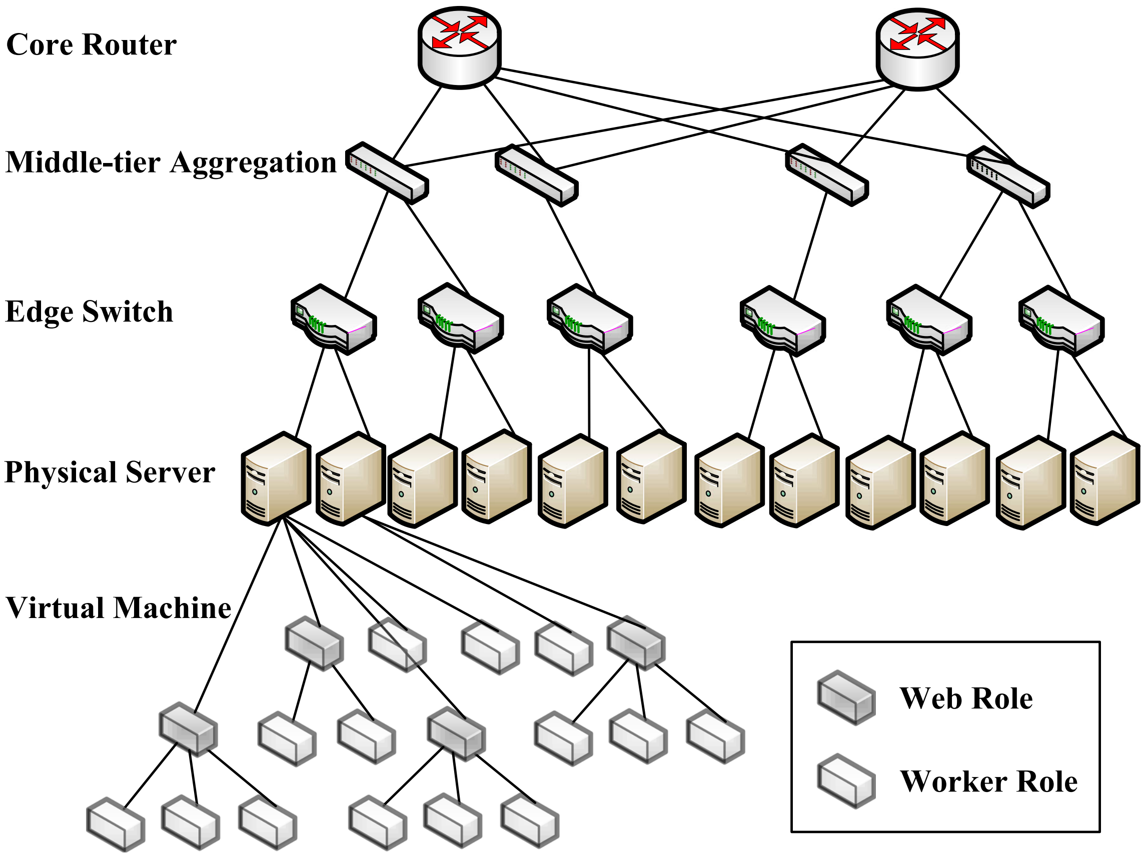

Fig. 2 The VM optimization result by Tree algorithm

communication cost. The third assumption is that the power consumption of each physical server is calculated. Moreover, the server can be shut down if the utility is lower than a specified threshold. Finally, we assume that the web role instance must be connected directly to a physical server. Unlike the web role instance, the worker role instance is able to connect with a physical server or a web role instance, which means that the worker role instance can link to the physical server through a web role instance.

The VM optimization result for the Tree algorithm is shown in Fig. 2. The Tree algorithm inclines the VM placement in the same physical server until the utility reaches the

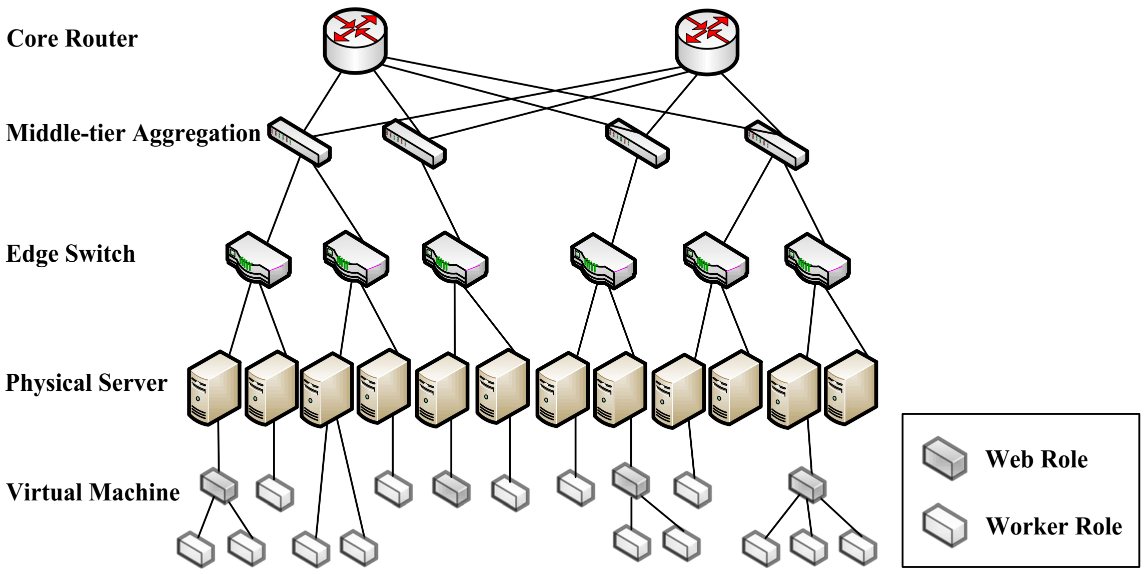

Fig. 3 The VM optimization result by Forest algorithm

Table 2 Simulation parameters

Variables/parameters	Value
Number of physical servers	30
Number of services	50
Number of web role instances	50
Number of worker role instances	150
Cloud infrastructure topology	Tree
Maximum power of one server	500 (W)
Server idle status threshold	30 (%)
Maximum utility rate	90 (%)

maximum utility δ. This implies that the depth-first-search method is applied in the Tree algorithm. We adopt the concept of service-oriented when deploy the different types of VMs. In order to decrease the unnecessary outbound communication cost, the proposed Tree algorithm utilizes depth-first-search approach to place the same service type of VMs into the same physical server. According to our assumptions the Tree algorithm achieves high power consumption efficiency with a lower communication cost. Furthermore, most of the idle servers are able to shut down to eliminate consuming power. However, an implicit problem of the Tree algorithm is that used servers are always in high utility. If the workloads of these servers increase, the server and system may crash due to overload. Therefore, the Tree algorithm requires a compatible migration mechanism.

The VM optimization result for the Forest algorithm is shown in Fig. 3. According to the breath-first-search method applied to the Forest algorithm, most physical servers are executed using few VM role instances. The Forest algorithm achieves better computation load balancing between physical servers, hence the probability of system failure is much lower than that for the Tree algorithm. Nevertheless, parts of VM role instances in the same service may execute between different physical servers. As stated before, the outbound communication cost of the Forest algorithm is greater than that for the Tree algorithm. The average server utility is low, meaning that most of the power consumption is wasted.

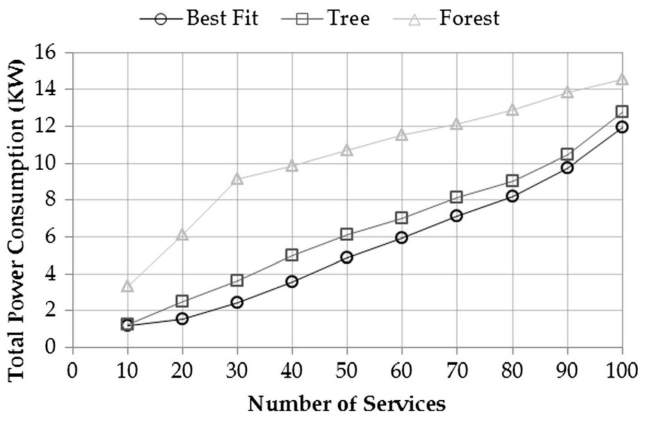

Fig. 5 Simulation results of total power consumption

4.2 Simulation results

To verify the performance and improvement we used Dev-C++ 5 to implement the proposed algorithms. Each result is repeatedly simulated five times to the average. The proposed algorithms are compared with the benchmark method named Best Fit algorithm, which represents the VM placement method without the service-oriented concept. The Best Fit algorithm always places the most suitable VM in the same physical server according to its' utility and deploys the VMs to a new physical server until the utility of previous server is full. In the simulation experiment the number of physical servers was 30. Each service includes at least one web role instance and one worker role instance. Therefore, we assume that the number of web roles is the same as the number of services and the worker role instances are always three times the number of web role instances. For example, if the number of services is 100, the number of web role instances is also 100 and the worker role instances number 300. There are 50 web role instances and 150 worker role instances in the general simulation case. The cloud infrastructure network topology is the

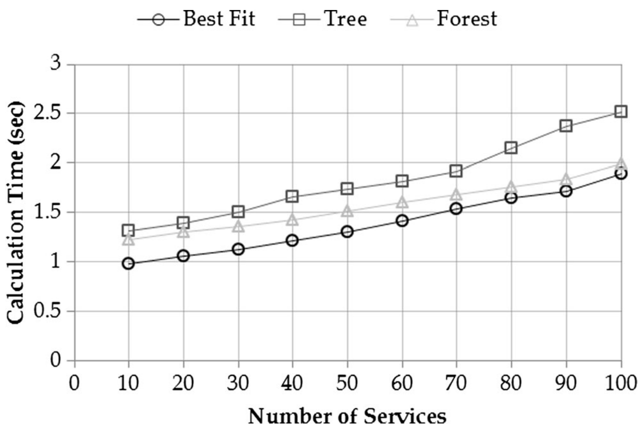

Fig. 4 Simulation results of calculation time

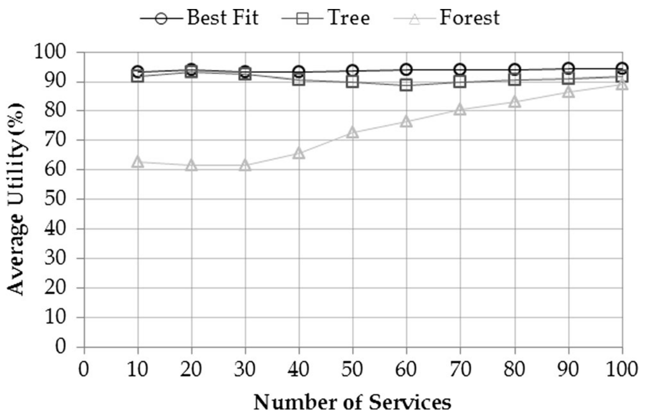

Fig. 6 Simulation results of average utility

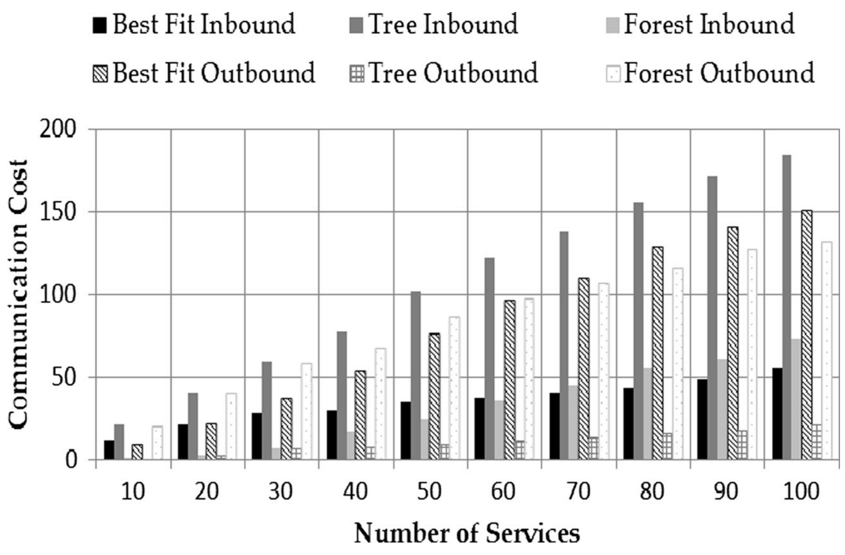

Fig. 7 Simulation results of communication cost

Tree structure and the VM placement structure is a tree structure. We assume that the maximum power of one physical server is 500 watts per hour, and the server idle status threshold is 300 watts per hour. One thing that should be noted is that if the power consumption of a server is lower than 30 %, the server is regarded as an idle server. The maximum utility rate represents that the highest utility rate of a physical server is defined as 90 %. All simulation parameters are shown in Table 2.

We increased the number of services to ten for each step from 10 to 100, and the numbers of web role instances with it. The worker role instances are then three times the number of web role instances. As Fig. 4 shows, the calculation time in both proposed algorithms and Best Fit algorithm are linear and approximately 1 to 2.5 s. Although the proposed algorithms spend more time than the Best Fit algorithm in the optimization procedure, the difference is smaller than 1 s and acceptable. According to this result, we can say that both proposed algorithms are appropriate for VM optimization and efficient even if the number of services is large. Originally, the computational cost of our formulated problem grows exponentially with the number of services. However, now the computational cost grows linearly with the number of services. Therefore, our proposed algorithm has an efficient calculation time on a NP-hard problem.

In [35] the authors pointed out the non-energy proportionality problem. A physical server consumes almost 50 % of the total energy usage when it is in the idle state. Therefore, we consider that the status of most physical servers should be power off in the beginning because it is unnecessary to switch on numerous servers when there are no service requests. A comparison of the total power consumption of both proposed algorithms is shown in Fig. 5. The total power consumption of the Tree and Forest algorithm is always higher than that of the Best Fit algorithm. The Best Fit algorithm fills up the active physical servers by fitting VMs as many as possible, hence its total power consumption is the lowest. Although the power consumption of Tree algorithm is higher than the Best Fit algorithm, the disparity is small. The Forest algorithm increases acutely when the number of services increases from 10 to 30 because most of the physical servers start up due to the VM placement. In other words, most of the physical servers are active when there are 30 service requests. However, the active servers are working at a low utility rate and waste much power when the server is in idle status. The Tree and Best Fit algorithms efficiently preserve power consumption.

In [36] the authors showed the low utilization server problem in the data center. The low utilization problem also implies a power efficiency problem. The average server utilization was reported to be 30 % even in the Google data center. The average physical server utility is shown in Fig. 6. Before discussing the results a definition of average utility is stated. We only calculate the utility rate of physical servers in which VM instances are executed. The result shows that the average utility of the Best Fit algorithm is always the highest, because it places the most appropriate VM instances based on the relation between VM utility and physical server. In the Tree algorithm VM instances with the same service have the higher priority for placement in the same physical server. Although the average utility of the Tree algorithm is slightly lower than the Best Fit algorithm, active servers are effectively utilized. Unlike the Tree algorithm the Forest algorithm awakens more physical servers because the breath-first-search method is applied and achieves better load balancing. The Best Fit algorithm achieves higher energy efficiency and maximizes the server utility. The Tree algorithm is similar. Both the Best Fit and Tree algorithms are more suitable for use in green data centers compared to the Forest algorithm.

The communication cost with different numbers of servers is shown in Fig. 7. It can be easily understood that higher

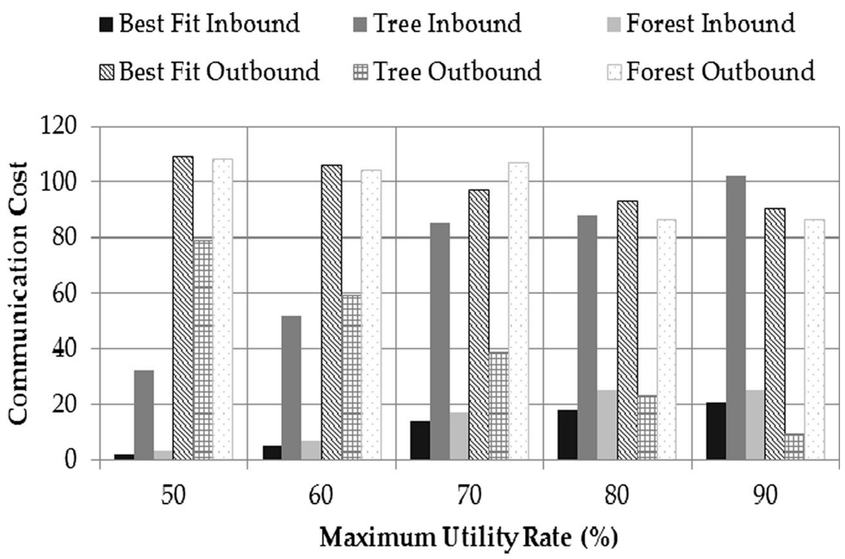

Fig. 8 Simulation results of communication cost with different utility rate

communication cost accompanies more services. In order to reduce the outbound communication cost for the same service, the proposed Tree algorithm always places VM role instances of the same service type in the same physical server, except when the server utility rate exceeds the maximum utility limitation. Moreover, regardless if the number of services is big or small, the outbound communication cost of the Tree algorithm is significantly lower than the Best Fit algorithm. Unlike the Tree algorithm, the Forest algorithm separates the VM instances for server load balancing. However, the outbound communication cost of the Forest algorithm is lower than that for the Best Fit algorithm when the number of services is greater than 70. It is very important that the Tree algorithm hugely eliminates unnecessary outbound communication cost in any scale network, while the Forest algorithm fits large scale networks such as the cloud environment or data center networks.

The utility rate stands for the benchmark of a physical server, hence a higher utility rate serves more VM instances. We adjusted the maximum utility rate limitation from 50 to 90 % and observed the communication cost from both algorithms, as shown in Fig. 8. In this scenario the number of physical servers and services were 30 and 50. The web and worker role instances were 50 and 150. If the maximum utility rate is reduced, the amount of outbound communication cost will increase because of the VM instances in the same service might be separated into different servers. However, the outbound communication cost of both proposed algorithms are always lower than that from the Best Fit algorithm. Regardless of the scenario, both proposed algorithms deployed VM instances within the same service type on a physical server as first priority. We summarize that both proposed algorithms reduce the outbound communication cost and agree with the higher utility rate. This also implies that both proposed algorithms provide energy efficiency for green cloud services.

5 Conclusions

This paper investigated the service-oriented VM placement problem in cloud data centers. We formulated the service-oriented VM placement optimization problem viz. SOVMPO problem based on ILP and proved it is a NP-hard problem by reducing it to the well-known traveling-salesman problem. Two heuristic algorithms, the Tree and Forest algorithms were proposed based on service-oriented VM placement and solving the SOVMPO problem in linear time. Lastly, both proposed algorithms were evaluated and compared with the Best Fit algorithm, standing in for non-service oriented VM placement in the simulation results.

For VM placement planning in current data center networks, green communication is a vital issue due to the three-tier architecture. We proposed the Tree and Forest algorithms to construct a network based on the service-oriented VM placement concept. Although the familiar Best Fit algorithm used the least power consumption and sustained the highest average utility in simulations, it led to enormously redundant outbound communications. Both proposed algorithms consumed slightly extra power consumption due to the service-oriented VM placement, but accomplished green communication with less outbound costs significantly. The Tree algorithm shuts down unnecessary physical servers to reserve power, while the Forest algorithm takes the computation loading into consideration during load balancing. The simulation results show that both proposed algorithms achieve less unnecessary outbound and more valuable inbound communication costs than the Best Fit algorithm. The Forest algorithm decreases 22 % outbound communication cost, and Tree algorithm decreases 92 % outbound communication cost significantly. We believe that our proposed algorithms are suitable for service-oriented VM placement in cloud data centers, particularly suited for large scale green data centers. We will investigate dynamic VM allocation with service-oriented concept in the future. In addition, the network architecture and topology in data center will be studied also.

Acknowledgments This research was partly funded by the Ministry of Science and Technology of the R.O.C. under grants MOST 103-2221-E-197-018 and 101-2221-E-197-008-MY3, National Nature Science Foundation of China under No. 61170296 & 60873241, and the Program for New Century Excellent Talents in University under Grant No. 291184.

References

1. Allred GR (1971) System/370 integrated evaluation under OS and DOS. In Proc. of the Spring Joint Computer Conference (SJCC)
2. Nelson M, Lim B.-H, Hutchins G (2005) Fast transparent migration for virtual machines. In Proc. of the USENIX Annual Technical Conference (ATC '05)
3. Wu T-Y, Chen C-Y, Kuo L-S, Lee W-T, Chao H-C (2012) Cloud-based image processing system with priority-based data distribution mechanism. Comput Commun 35(15):1809–1818
4. Gartner Inc., Available: http://www.gartner.com/technology/home.jsp Accessed 1 August 2014
5. Gartner identifies the top 10 strategic technologies for 2009, available: http://www.gartner.com/it/page.jsp?id=777212 Accessed 1 August 2014
6. Gartner identifies the top 10 strategic technologies for 2010. Available: http://www.gartner.com/it/page.jsp?id=1210613 Accessed 1 August 2014
7. Gartner identifies the top 10 strategic technologies for 2011. Available: http://www.gartner.com/it/page.jsp?id=1454221 Accessed 1 August 2014
8. Gartner identifies the top 10 strategic technologies for 2012. Available: http://www.gartner.com/it/page.jsp?id=1826214 Accessed 1 August 2014
9. Mishra M, Sahoo A (2011) On theory of VM placement anomalies in existing methodologies and their mitigation using a novel vector based approach. In Proc. of the IEEE International Conference on Cloud Computing (CLOUD)
10. Xu J, Fortes JAB (2010) Multi-objective virtual machine placement in virtualized data center environments. In Proc. of the 2010 IEEE/ACM Int'l Conference on Green Computing and Communications & Int'l Conference on Cyber, Physical and Social Computing (GREENCOM-CPSOM)
11. Wang M, Meng X, Zhang L (2011) Consolidating virtual machines with dynamic bandwidth demand in data centers. In Proc. of the IEEE International Conference on Computer Communications (INFOCOM)
12. Strunk A (2012) Costs of virtual machine live migration: a survey. In Proc. of the IEEE 8th World Congress on Services (SERVICES)
13. Liu H, Jin H, Liao X, Yu C, Xu C-Z (2011) Live virtual machine migration via asynchronous replication and state synchronization. IEEE Trans Parallel Distrib Syst 22(12):1986–1999
14. Verma A, Ahuja P, Neogi A (2008) pMapper: power and migration cost aware application placement in virtualized systems. In Proc. of the 9th ACM/IFIP/USENIX International Conference on Middleware (Middleware)
15. Castro H, Sotelo G, Diaz CO, Bouvry P (2011) Green flexible opportunistic computing with virtualization. In Proc. of the IEEE 11th International Conference on Computer and Information Technology (CIT)
16. Chaisiri S, Lee B-S, Niyato D (2012) Optimization of resource provisioning cost in cloud computing. IEEE Trans Serv Comput 5(2):164–177
17. Zhou L, Wang H (2013) Toward blind scheduling in mobile media cloud: fairness, simplicity, and asymptotic optimality. IEEE Trans Multimed 15(4):735–746
18. Zhou L, Yang Z, Rodrigues JJPC, Guizani M (2013) Exploring blind online scheduling for mobile cloud multimedia services. IEEE Wirel Commun 20(3):54–61
19. Yang K, Gu J, Zhao T, Sun G (2011) An optimized control strategy for load balancing based on live migration of virtual machine. In Proc. of the 6th Annual Chinagrid Conference (ChinaGrid)
20. Hu B, Lei Z, Lei Y, Xu D, Li Z (2011) A time-series based precopy approach for live migration of virtual machines. In Proc. of the IEEE 17th International Conference on Parallel and Distributed Systems (ICPADS)
21. Clark C, Fraser K, Hand S, Hansen JG (2005) Live migration of virtual machines. In Proc. of the 2nd Conference on Symposium on Networked Systems Design & Implementation (NSDI '05)
22. Lin RC-H, Liao H-J, Tung K-Y, Lin Y-C, Wu S-L (2012) Network traffic analysis with cloud platform. J Internet Technol 13(6):953–962
23. Horng M-F, Chen Y-T, Chung P-W, Shieh C-S, Pan J-S (2012) An adaptive approach to high-throughput inter-cloud data transmission based on fast TCP. J Internet Technol 13(6):971–980
24. Cisco Data Center 3.0, Available: http://www.bradreese.com/blog/cisco-data-center-3-0.pdf, 2009 Accessed 1 August 2014
25. Sridhar T (2009) Cloud computing—a primer. Internet Protocol J 12(3):1–31
26. Meng X, Pappas V, Zhang L (2010) Improving the scalability of data center networks with traffic-aware virtual machine placement. In Proc. of the IEEE International Conference on Computer Communications (INFOCOM)
27. Zhang Y, Ansari N (2013) On architecture design, congestion notification, TCP incast and power consumption in data centers. IEEE Commun Surv Tutorials 15(1):39–64
28. Hill Z, Li J, Mao M, Ruiz-Alvarez A, Humphrey M (2010) Early observations on the performance of windows azure. In Proc. of the 19th ACM International Symposium on High Performance Distributed Computing (HPDC)
29. Xu J, Tang J, Kwiat K, Zhang W, Xue G (2012) Survivable virtual infrastructure mapping in virtualized data centers. In Proc. of the IEEE 5th International Conference on Cloud Computing (CLOUD)
30. Shrivastava V, Zerfos P, Lee K-W, Jamjoom H, Liu Y-H, Banerjee S (2011) Application-aware virtual machine migration in data centers. In Proc. of the IEEE International Conference on Computer Communications (INFOCOM)
31. Song F, Huang D, Zhou H, You I (2012) Application-aware virtual machine placement in data centers. In Proc. of the International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS)
32. Greenberg A, Jain N, Kandula S, Kim C, Lahiri P, Maltz D, Patel P, Sengupta S (2009) VL2: a scalable and flexible data center network. In Proc. of the ACM SIGCOMM 2009 conference on Data communication
33. Guo C, Lu G, Li D, Wu H, Zhang X, Shi Y, Tian C, Zhang Y, Wu S (2009) BCube: a high performance, server-centric network architecture for modular data centers. In Proc. of the ACM SIGCOMM 2009 conference on Data communication
34. Garey MR, Johnson DS (1979) Computers and intractability a guide to the theory of NP-Completeness. W. H. Freeman, San Francisco
35. Barroso LA, Hölzle U (2007) The case for energy-proportional computing. IEEE Comput 40(12):33–37
36. Barroso LA, Hölzle U (2009) The datacenter as a computer: an introduction to the design of warehouse-scale machines. Morgan and Claypool Publishers