

Support vector machine approach for virtual machine migration in cloud data center

Fan-Hsun Tseng · Xiaojiao Chen · Li-Der Chou ·
Han-Chieh Chao · Shiping Chen

Received: 23 December 2013 / Revised: 3 March 2014 / Accepted: 7 May 2014 /
Published online: 19 June 2014
© Springer Science+Business Media New York 2014

Abstract The social media services are popular with Internet services today, such as Facebook, YouTube, Plurk and Twitter. However, the enormous interactions among human beings also result in highly computational costs. The requested resources and demands of some specific social media services are changing severely, and the virtual machines (VMs) exhaust the computing resource of physical machine (PM). Thus this will lead to VM migration. Many researchers investigate how to stabilize the average utilization of virtual machines and physical machines in cloud data center. In this paper, we formulated the VM

F. -H. Tseng · L. -D. Chou
Department of Computer Science and Information Engineering, National Central University,
No. 300, Jhongda Rd., Jhongli, Taoyuan County 32001, Taiwan, Republic of China

F. -H. Tseng
e-mail: fhsteng@ieee.org

L. -D. Chou
e-mail: cld@csie.ncu.edu.tw

X. J. Chen
School of Optical-Electrical and Computer Engineering, University of Shanghai for Science
and Technology, No. 516, Jun Gong Rd., Shanghai, 200093, Peoples's Republic of China
e-mail: clkjh1210@163.com

H. -C. Chao (✉)
Department of Electronic Engineering & Department of Computer Science and Information
Engineering, National I-Lan University, No. 1, Sec. 1, Shen-Lung Rd., I-Lan, 26047, Taiwan, Republic
of China
e-mail: hcc@niu.edu.tw

H. -C. Chao
Department of Electrical Engineering, National Dong Hwa University, No. 1, Sec. 2, Da Hsueh Rd.,
Shoufeng, Hualien 97401, Taiwan

S. P. Chen
School of Optical-Electrical and Computer Engineering & Network Center, University of Shanghai for
Science and Technology, No. 516, Jun Gong Rd., Shanghai, 200093, Peoples's Republic of China
e-mail: chensp@usst.edu.cn

migration problem in cloud data center based on mixed integer linear programming (MILP). Then, the VM allocation algorithm was proposed to allocate the VMs among the PMs, which is based on the Support Vector Machine (SVM). According to the training process during a specific time, the minimum numbers of VM migration and maximum resource utilization of PMs were accomplished. As the allocation case and simulation results showed, we achieved the stable and low-cost for social media services in cloud data center.

Keywords Social media service · Load balance · Support vector machine · Mixed integer linear programming · Cloud data center

1 Introduction

Social media was defined by Andreas M. Kaplan and Michael Haenlein [13], and it refers to the interaction among human beings. They defined social media as various Internet-based applications that allow users create and share the user-generated content [6]. People create some events and exchange their thoughts in the virtual social communities and worlds. For instance, bloggers may write their works and creations to blog, and share information without face to face conversation [11]. In order to serve and manage the enormous social media services, the well-balanced VM placement and allocation is definitely required in cloud data center.

The evolution of cloud computing not only changes the business model [27] and system architecture [1], but also leads the Internet services to a novel direction [18]. The various resources in the resource pool should be elastic and scalable [16], which is different from the consolidated resource in traditional computing method. Therefore, the researchers start to investigate how to improve the performance [5] and allocate the resource elastically [21]. There are many advantages of virtualization technique, including efficient resource utilization [8], easy management [25], reduced energy consumption [15], simultaneous resource monitoring [29] and so forth. However, it also brings some problems, such as the attack problem [10], performance overhead [22], unnecessary VM migration [24] and so on.

It is clear that the centralized task assignment is infeasible in cloud environment [23]. The researchers in [26] divide the cloud infrastructure into three layers, which are request management level, service management level and service execution level. In this paper, the same three-tier architecture for cloud data center is considered. On the other hand, the tree structure is generally applied in data center. The leaves are various VMs, and the parents are several PMs. If the resource requirement of VMs exceeds the remaining resource in PMs, part of VMs should be migrated to other PMs. The valid VM migration is able to reduce the resource and energy wastage [9]. An efficient and easy method of load balancing is migrating some VMs from the busy PM to the idle PM.

The remainder of this paper is organized as follows. In Section 2, we compare the current researches about VM migration and load balance schemes in cloud data center. The Section 3 defines the solved VM migration problem based on Mixed Integer Linear Programming (MILP), and states the strategy for classifying VMs. Besides, the proposed VM allocation algorithm is also depicted and explained in Section 3. In Section 4, we discuss the simulation results of this work. The conclusion from this research is drawn in Section 5.

2 Related works

There are many researches investigating how to balance payoff between cost and efficiency during VM migration. Most of them study balancing the workload of homogeneous nodes. But it is unfeasible in current cloud data center, the dynamic and heterogeneous systems are used to provide on-demand resources and services [23]. To find out the lower load node for migration, parts of researchers put forward the scheduler to choose the appropriate PM and set a threshold to VM. Once the PM exceed in its utilization, it will recall the schedule and choose a suitable VM to migrate [20]. However, they usually neglect the migration cost during migration process.

The files in data center are created and deleted dynamically. The file chunks are not distributed uniformly in virtual machines [7], hence it leads the imbalanced workload in the entire system. This implies that the task usage is different, and the change between each node is also diverse. Based on the above-mentioned phenomenon, the VM migration is needed and unavoidable. Therefore, some researchers consider that the virtualization technique should be the first step in cloud computing [30].

Some literatures on the VM load balance have been proposed, such as [20], [30] and [14]. In [20], the authors proposed a combined algorithm that focus on the server workload and migration efficiency. The numbers of virtual machines and some migration criteria were studied. The VMs were migrated for the minimum power consumption, and the underutilized PMs were shut down. The energy cost and cooling cost in data center were economized. The authors [30] proposed Multi-agent Genetic Algorithm (MAGA) based on the Genetic Algorithm (GA) and multi-agent technique, and derived it to a mathematical problem from their proposed model. It not only solved the high dimensional function optimization, but also adjusted the parameters in large-scale network to achieve the load balance feature.

Here are some related works to ours, such as [12] and [28]. The authors in [28] detected the resource and the change of hotspot, and found the hotspot that needs to be migrated. It migrated the VMs through remapping the resources of VMs. The most similar work with ours is [12]. According to historical data and current state of VMs, the authors proposed their GA algorithm to put forward a scheduling strategy on load balancing of VM resources. By computing the influence of VMs, they selected the least-effective solution to prevent dynamic migration. However, they didnt consider the stability of VMs. Since the resource in cloud environment is elastic and changeable, the computing states in VMs are differential. It will lead to low utilization of PMs in data center. In this work, we not only consider the stability of the VMs, and also investigate the utilization of the VMs. According to the difference of VMs stability and utilization, we propose an algorithm for allocating VMs stably and steady.

3 Problem definition and proposed algorithm

3.1 Problem definition

The VM migration problem is defined and formulated based on Mix Integer Linear Programming (MILP), viz the Minimization VM Migration (MVMM) problem in this work. The used variables and notations in MVMM problem are listed and defined as Table 1.

Let there are N VMs in M PMs, and the utilization of VM n and PM m are U_n and U_m . Assume that the capacity of the PM m is C_m and the minimum resource demand of the VM

Table 1 Definition of the important notations

Notations	Definition
\mathbb{N}_{PM}	Set of PMs in cloud infrastructure
$ \mathbb{N}_{PM} = M$	Number of PMs in cloud infrastructure
\mathbb{N}_{VM}	Set of VMs in PM
$ \mathbb{N}_{VM} = N$	Number of VM in PM
C_m	The capacity of PM m
U_m	The utilization of PM m
U_n	The utilization of VM n
δ_n	The minimum resource demand for VM n
T	A time interval
$a_{m,n}$	The VM n migrates from PM m
Z_N^M	The number of migration VMs in PMs

n is δ_n . Then, we consider the utilization of PM m . When the utilization of VM n or PM m is higher than the capacity of PM m , the VM n must be migrated from PM m to other PM, where

$$a_{m,n} = \begin{cases} 1, & \text{if } U_n > C_m \vee U_m > C_m. \\ 0, & \text{otherwise.} \end{cases} \tag{1}$$

The VM needs resource to execute tasks. We consider the U_m and U_n including the common resources, such as CPU, storage and bandwidth utilization. According to above statements, we formulate our MVMM problem as the minimum numbers of VM migration in a time interval T , which is modeled as the following.

Minimize

$$Z_N^M = T \left(\sum_{m=1}^M \sum_{n=1}^N a_{m,n} \right). \tag{2}$$

subject to

$$a_{m,n} \in \{0,1\}, \text{ for } \forall n \in \mathbb{N}_{VM}, \forall m \in \mathbb{N}_{PM}, \tag{3}$$

$$Z_N^M \geq 0, \text{ for } N \in \mathbb{N}_{VM}, M \in \mathbb{N}_{PM}, \tag{4}$$

$$U_m \leq 1, \text{ for } \forall m \in \mathbb{N}_{PM}, \tag{5}$$

$$U_n \leq 1, \text{ for } \forall n \in \mathbb{N}_{VM}, \tag{6}$$

$$\delta_n \leq C_m \leq, \text{ for } \forall n \in \mathbb{N}_{VM}, \forall m \in \mathbb{N}_{PM}, \tag{7}$$

$$T \in \mathbb{Q}. \tag{8}$$

The objective function (2) minimizes the numbers of VM migration in PMs within a time interval T . Constraint (3) states that the decision of VM migration for \mathbb{N}_{VM} and \mathbb{N}_{PM} is a binary, which represents the VM should be migrated or not. Constraint (4) ensures that the

numbers of VM migration is a positive integer. In constraints (5) and (6), the utilization of PM m and VM n is less than or equal to one. They are associated with one PM and one VM. Constraint (7) ensures the VM resource requirement should not exceed the PM capacity. Constraint (8) states that the time interval is a rational number.

3.2 VM classification strategy

In this work, the VMs are categorized and classified according to types. We utilize the concept of Support Vector Machine (SVM) [2] to decide the VM types. The concept of SVM could be applied to many research fields, such as anomaly detection [4], appliance recognition [17], and so on. We use the open source tool called LIBSVM [3] to implement the VM classification model.

The space classification of SVM is shown as Fig. 1. It is a well-known approach for data classifying and machine learning. The classifier is trained by the existing categories, and sorts the data by category and type. In SVM, the concept of classification boundary is adopted to demarcate these classification boundaries. Through data training, SVM finds the boundaries between these classifications, which are liner-called liner division and curve-called nonlinear division.

In this work, two main factors influence the types of VMs. One is the average utilization of VMs, and the other one is the stability of VMs. The stability refers to the resource demand during a time period. In other words, the VM is regarded as unstable if its resource usage is frequently changed. Based on these two factors, the VMs are able to classify into four types. We can utilize SVM to discover the boundaries.

We name the four VM types A, B, C and D. The type A VMs are changeable so that the remained resource may be insufficient. It is unstable and needs high resource utilization. The type B VMs are also unstable but need less resource than type A VM. Unlike type A and B, the type C and D VMs are stable which means the resource requirement is steady. The type C VM is stable but needs high resource utilization, and the type D VM requires less resource allocation. Since the boundaries between these four VM types are unknown, the SVM tool is applied to classify them.

The dataflow of proposed classification method is shown as Fig. 2. In order to achieve the decision model, we must provide the training data for SVM. It follows the rule of training data and obtains the decision model. We execute the VMs in advance, and select the data and training data. The original data should be transformed to vector form through hash function,

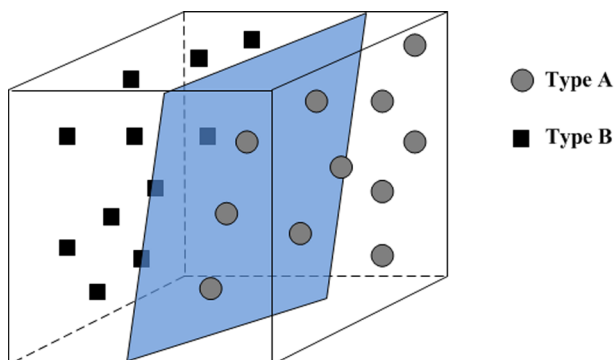


Fig. 1 The classification capability of SVM

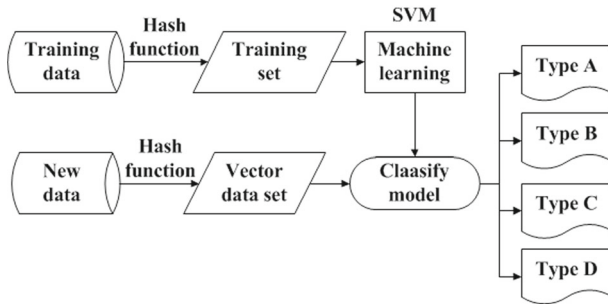


Fig. 2 The classification dataflow in general SVM

and then the training set is acquired. Through machine learning, the SVM gets decision model of four VM types. Finally, the incoming data can be classified and categorized by the decision model.

3.3 VM allocation algorithm

After the categories of VMs obtained, we propose a VM allocation algorithm to distribute them based on their categories. For minimizing the probability of VM migration, we allot the VMs with the remained resource of PM. Unlike the existing algorithms, we not only consider the average utilization of VMs but also the stability. It will decrease the redundant migration costs.

The notations and functions in proposed algorithm are clarified as follows. The \mathbb{N}_{PM} stands for the set of PMs. Then, we assume that all PMs have the same capacity C_m . The E_x is defined as the type of VM. $\mathbb{N}_{PMY} = (N_{PM}, U_{PM})$ used to record the PM which contains the Y type VMs only, and its utilization is U_{PM} . The $f_a(\mathbb{N}_{PM}, E_X, C_m)$ is used to allot all VMs of type X under the PMs alone. It generates the list \mathbb{N}_{PMX} and the list S_X of VMs selected in type X. Our allocation policy is that the VMs utilization must not exceed the PM capacity C_m . This function goes to allocate E_X alone under the PMs. Another function $f_b(\mathbb{N}_{PMY}, E_X, C_m)$ is used to allocate the type X VMs E_X under the PMs \mathbb{N}_{PMY} , which had been allocated to the Y type VM. It produced a list S_X recording the selected X type VMs. If the list S_X is null, it implies that there is no enough resource for the VM type X or the VMs of type X are allocated completely. This function allocates VMs of one type with another type VMs.

We introduce and explain the proposed VM Allocation algorithm herein. From line 2 to line 6, all type A VMs are placed in the PMs. The same procedure with above, the type B VMs are allocated to PMs in line 7 to line 11. From line 12 to line 15, we allot type D VMs to the PMs which allocated to type A VMs in line 2 to line 6, until there is no more resource for type D VMs or there is no type D VMs. From line 16 to line 18, the type C VMs are allocated to the PMs which type B VMs placed before. If the PMs for type C VMs are use up, the type C VMs are allocated to other PMs in line 19 to 24, until all VMs of type C are allocated. From line 25 to line 29, if the VMs of type D is not allocate yet, we allocate the remained type D VMs to PMs. These PMs were allocated for the type C VMs before. Lastly, the unallocated VMs of type D are placed to other PMs in line 30 to line 33. It ensures all of the VMs are allocated.

4 Experiment results

4.1 VM classification model

We utilize a well-known open source, viz. LIBSVM [19] to implement our classification model. The LIBSVM is an SVM library, and the newest version is 3.17 released on April 1, 2013. Firstly, we provide the training data for LIBSVM. The original data should be transferred to SVM format before using training function. In order to avoid the range of features is within the reasonable value, and it can be executed in the kernel function. We scale the training data for LIBSVM tool. The process from training data to scale set is shown as Fig. 3.

To obtain the training data, we provide some data about the VMs during a specific running time. The label in training data stands for the species of VMs, and it could be filled in or empty. We classify the total VMs into four kinds. Therefore the SVM categorizes the VMs in training process, and fill in 1 to 4. The c-u(tx), m-u(tx), s-u(tx) and b-u(tx) are the feature values of VMs. They represent the CPU utilization, memory utilization, storage utilization and bandwidth utilization at time tx, respectively. In this paper, we choose 600 records at five time stamps in a unit time as the training data. In the training set, hash function used to transform the training data into SVM format. The scale function converts the training set to the data within a certain range. It is also called scale set, and the scale range is from -1 to 1. Finally, the obtained parameters are used to train the scale set.

For the better training data, we need to find the best configuration for training parameter. During the selection process, the function repeatedly executes to obtain the optimal parameters for VM classification. The parameter is produced and drawn by gnuplot. It is

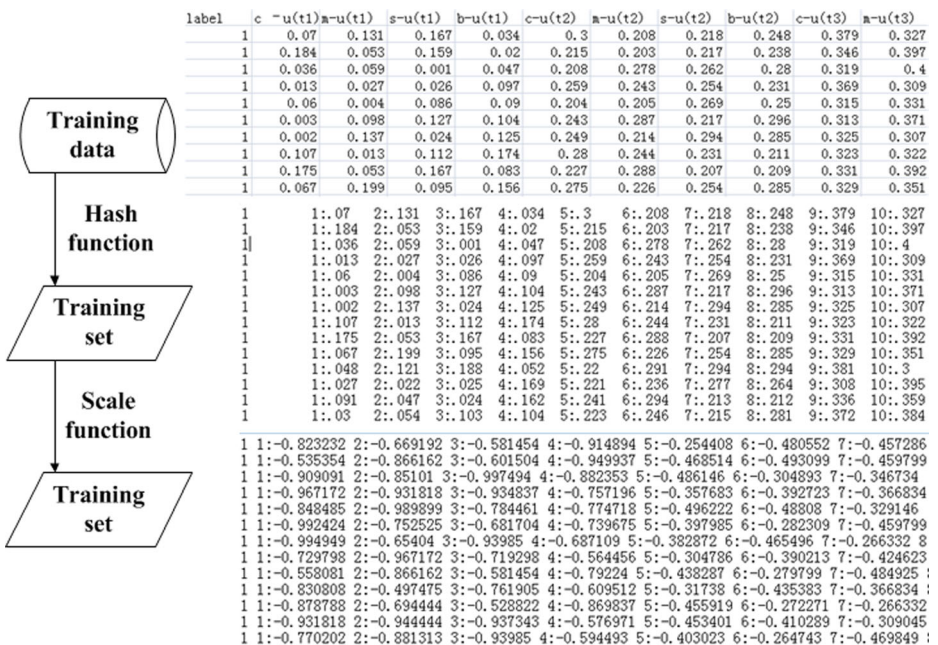


Fig. 3 The process from training data to training set

shown as Fig. 4. According to cross validation, the optimal parameter for VM classification is acquired and found.

Algorithm: VM Allocation ($E_A, E_B, E_C, E_D, \mathbb{N}_{VM}, C_m$)

Input: \mathbb{N}_{PM} : the set of PM_s
 E_A : the set of VM_s of type A
 E_B : the set of VM_s of type B
 E_C : the set of VM_s of type C
 E_D : the set of VM_s of type D
 C_m : the capacity of PM_s

```

01  $S_{PM} = \phi, S_B = \phi, S_C = \phi, S_D = \phi$ 
02 repeat
03  $f_a(\mathbb{N}_{PM}, E_A, C_m) = \{\mathbb{N}_{PMA}, S_A\}$ 
04  $\mathbb{N}_{PM} = \mathbb{N}_{PM} \setminus \mathbb{N}_{PMA}$ 
05  $E_A = E_A \setminus S_A$ 
06 until  $E_A = \phi$ 
07 repeat
08  $f_a(\mathbb{N}_{PM}, E_B, C_m) = \{\mathbb{N}_{PMB}, S_B\}$ 
09  $\mathbb{N}_{PM} = \mathbb{N}_{PM} \setminus \mathbb{N}_{PMB}$ 
10  $E_B = E_B \setminus S_B$ 
11 until  $E_B = \phi$ 
12 repeat
13  $f_b(\mathbb{N}_{PMA}, E_D, C_m) = \{S_D\}$ 
14  $E_D = E_D \setminus S_D$ 
15 until  $f_b(S_{PMA}, E_D, C_m) = \phi$ 
16  $f_b(\mathbb{N}_{PMB}, E_C, C_m) = \{S_C\}$ 
17  $E_C = E_C \setminus S_C$ 
18 until  $f_b(S_{PMA}, E_C, C_m) = \phi$ 
19 if  $E_C \neq \phi$ 
20 repeat
21  $f_a(\mathbb{N}_{PM}, E_C, C_m) = \{\mathbb{N}_{PMC}, S_C\}$ 
22  $\mathbb{N}_{PM} = \mathbb{N}_{PM} \setminus \mathbb{N}_{PMC}$ 
23  $E_C = E_C \setminus S_C$ 
24 until  $E_C = \phi$ 
25 if  $E_D \neq \phi$ 
26 repeat
27  $f_b(\mathbb{N}_{PMC}, E_D, C_m) = \{S_D\}$ 
28  $E_D = E_D \setminus S_D$ 
29 until  $f_b(S_{PMC}, E_D, C_m) = \phi$ 
30 else if
31 repeat
32  $f_a(\mathbb{N}_{PM}, E_D, C_m) = \{\mathbb{N}_{PMC}, S_D\}$ 
33 until  $E_D \neq \phi$ 

```

When the training process finished, we utilize the machine learning function in LIBSVM to classify VMs. The training data achieves the decision model for classifying VMs. The iteration of machine learning in training procedure and the decision model generation is as shown as Fig. 5.

4.2 Planning case

We have acquired the decision model for VM classification. The decision model should be coped with the proposed VM Allocation algorithm to minimize the number of VM migration. The Best Fit algorithm and First Come First Serve (FCFS) algorithm are compared to

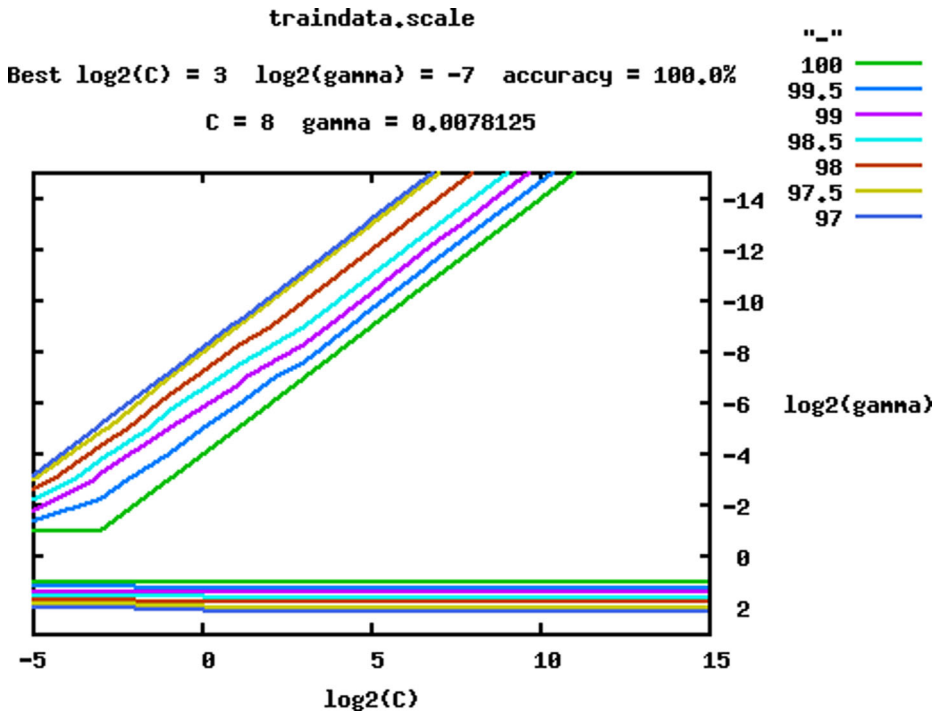


Fig. 4 The process for producing parameter

our algorithm with the planning case and simulation. The object of Best Fit algorithm is try to utilize the PM resource as much as possible, and it will allocate the closest VM to PM in terms of requested demand. The FCFS algorithm distributes VMs according to the incoming order of VMs. Before discussion on the planning case, there are three assumptions should be assumed.

1. The hardware standard and computing resource of all PMs are all the same, including the allocated capacity.
2. In the simulation, we do not define the required bandwidth and storage of all PMs, and the VMs vice versa.
3. We assume that all the related communication effects are consistent, therefore we only consider the cost of VM migration.

The planning condition is shown as Fig. 6. We take 6 PMs and 10 VMs into consideration, and the capacity of each PM is 0.9. In order to compare the difference between three algorithms, we observe and record the utilization of VMs during two unit times. The current requirement and maximum requirement within two unit times are shown as Table 2.

In VM allocation algorithm, we utilize the decision model which achieved by LIBSVM to classify the VM types and acquire the maximum requirement during training time. In the planning case, the training time is 1 unit time. The types and maximum requirements of each VM are shown as Table 3. The results of three algorithms are shown as Figs. 7, 8 and 9.

The best fit algorithm maximizes the utilization of PMs, and allocates the VM that has the maximum requirement to PM. Firstly, it sorts all VMs based on their demand in descending

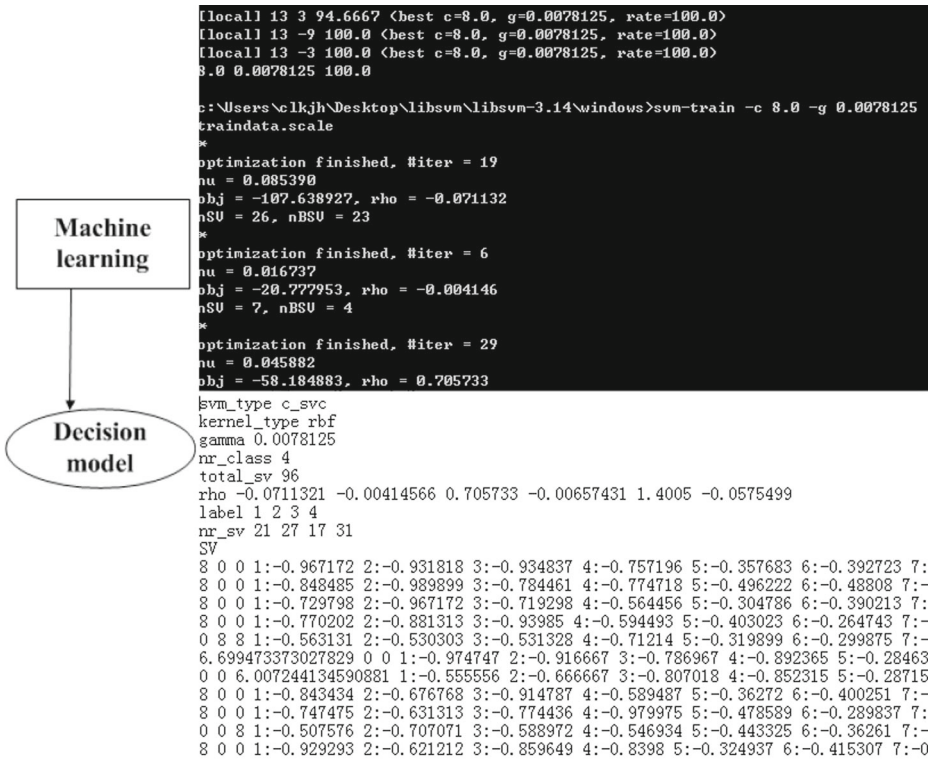


Fig. 5 The iterative training process

order, which is VM5, VM9, VM10, VM3, VM6, VM7, VM1, VM4, VM2, VM8. Then, the VM5 is allocated to PM1. Due to the limitation of PMs capacity, we have to allocate the VM9 to PM2. Moreover, the demand of VM10 is less than the remaining resource of PM1, hence the VM10 is placed to PM1. When there is no remaining resource in PM1, we continue allocating the remaining VMs to PM2. The rest of PMs are continually allocated to PMs with the same method.

In the FCFS algorithm, the VMs are allocated to PMs based on the incoming order of VMs. The incoming sequence of VMs is same as their numeric. Firstly, since the summation

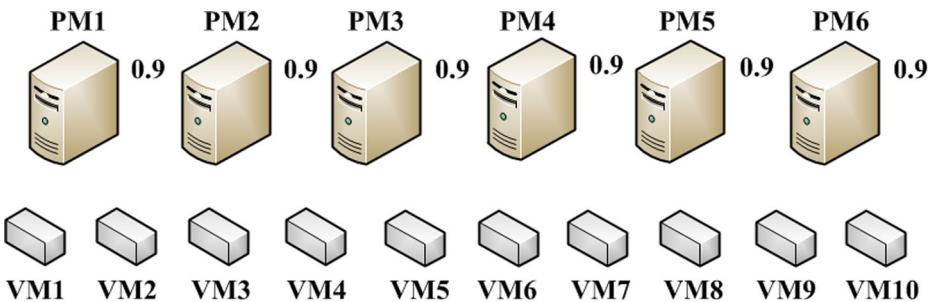


Fig. 6 Planning condition

Table 2 The requirement of VMs

The ID of VMs	The current requirement	The max. requirement during two unit times
VM1	0.15	0.39
VM2	0.14	0.59
VM3	0.21	0.20
VM4	0.15	0.33
VM5	0.53	0.51
VM6	0.17	0.60
VM7	0.17	0.15
VM8	0.13	0.39
VM9	0.52	0.51
VM10	0.22	0.21

of requirement is less than the capacity of PM1, we allocate the VM1, VM2, VM3 and VM4 to PM1. Then, the VM5, VM6 and VM7 are allocated to PM2. Finally, the VM8, VM9, VM10 are allocated to the PM2 according to their incoming order and the PM capacity limitation.

In the proposed VM allocation algorithm, we allocate the VMs according their types, which are judged with the maximum requirement during the training time. Firstly, we respectively allocate the VMs of type A (VM2 and VM6 with red label in Fig. 9) and the VMs of type B (VM1, VM4 and VM8 with blue label in Fig. 9) to the PM1, PM2, PM3, PM4 and PM5. Then we distribute the VMs of type D (VM3 and VM7 in this case) to PM1 and PM2. In this phase, we allocate the VMs of type C (VM9 and VM5 with green label in Fig. 9) to PM3 and PM4. At last we distribute the remaining VMs (VM10 in this case) to PM5, which has enough resource for placing VM10. One thing should be mentioned here, the assignment of VMs is based on the maximum requirement during training time, rather than the current requirement.

The comparison of planning result for three algorithms is shown as Table 4. In the planning case, the migration cost is caused when a specific VM migrates to another PM. The number of migration during two unit times for best fit algorithm and FCFS algorithms is

Table 3 The types of VMs and maximum requirement during training time

The ID of VMs	The VM types	The max. requirement during training time
VM1	B	0.37
VM2	A	0.57
VM3	D	0.20
VM4	B	0.35
VM5	C	0.55
VM6	A	0.60
VM7	D	0.17
VM8	B	0.39
VM9	C	0.51
VM10	D	0.21

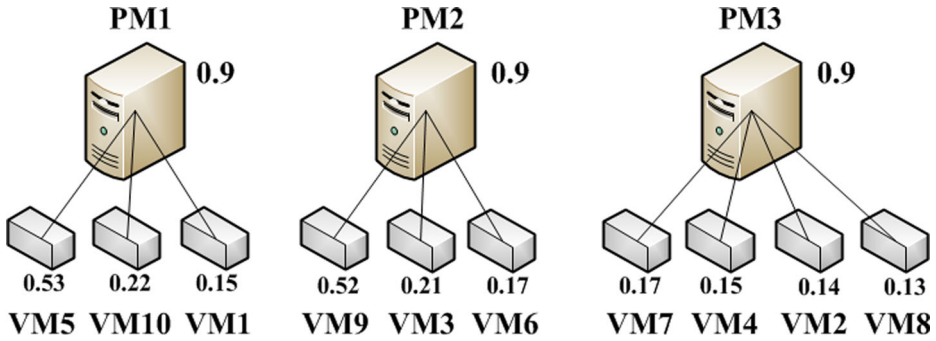


Fig. 7 The planning result of best fit algorithm

4 and 5 times, while proposed VM allocation algorithms is 2 times. Moreover, the total cost is calculated that the VM migrates from one PM to another PM through the routers and switches. In other words, it is based on routing length of VM migration. The total cost of best fit and FCFS algorithm is 6 and 9 times, and the proposed VM allocation algorithm keeps 2 times. According to this planning case, we consider that the VM allocation algorithm achieves the minimum number of VM migration and the lowest total cost.

4.3 Simulation results

The network structure in simulation is similar to [26]. We focus on the migration time and cost, including the insufficient resource of PM and VM. The used simulation parameters are shown as Table 5.

For the better explanation of proposed algorithm, we additionally add a benchmark setting of Allocation algorithm in simulation, viz Benchmark. It evaluates the influence on the proportion of four types VM in Allocation algorithm. All of the simulation parameters in Benchmark and Allocation algorithm are same except the proportion of VM types. The proportion of VM types in Benchmark is 1:1:1:1, and it is 2:3:2:3 in Allocation algorithm.

In Fig. 10, the number of VM migration of four algorithms during 10 unit time is shown. The number of VM migration in a time period is calculated and recorded at each unit time. The proposed Allocation algorithm may not predict all the max requirement, because it

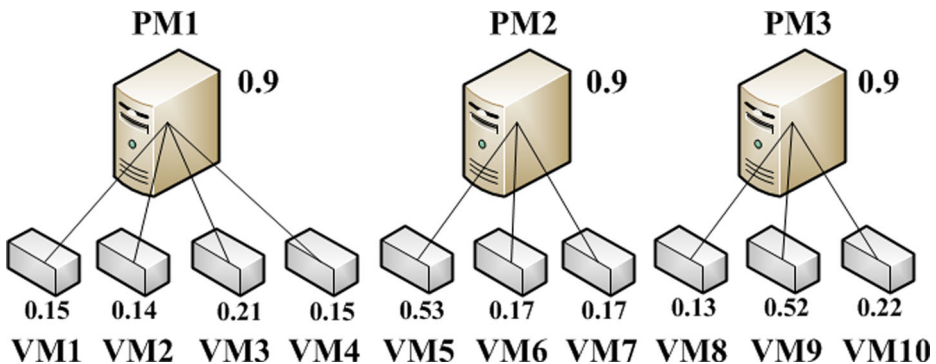


Fig. 8 The planning result of FCFS algorithm

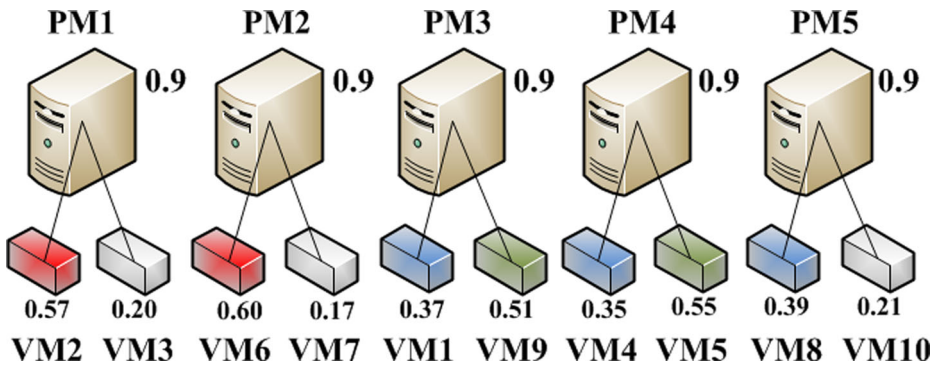


Fig. 9 The planning result of the proposed VM Allocation algorithm

just obtain the max value during the training course. The resource requirement of VMs may exceed the remained resource, so that the VM migration might be happened. Once the VM request appears, the FCFS and best fit algorithm allocate VMs to PMs immediately. It means that both algorithms do not consider the changeability and stability of VMs. As the result, their VM migration numbers are severely higher than the proposed Allocation algorithm. The VM migration number of Benchmark and Allocation algorithm are much lower than best fit and FCFS algorithm. Both of them consider the maximum resource requirement through the training process, hence not only the migration times is quite low also the variability. The varition of Benchmark and Allocation algorithm is lower than the FCFS and best fit. Moreover, the Allocation algorithm is always slightly less than the Benchmark due to the uniform distribution on VM types. It means the Allocation algorithm can reduce migration times effectively, and the proportion of four types VMs does not highly influence on the result.

The total cost of four VM allocation methods is shown as Fig. 11. First of all, the definition of total cost should be defined and clarified. We only focus on the cost of VM migration. The cost is decided by the routing overhead, and caused by the notification of VM migration between two PMs. The proposed Allocation algorithm achieves the lower VM migration probability so that the amount of notification is also lower. Therefore, the total cost of proposed algorithm is much less than the FCFS and best fit algorithm.

Because of the PM number is sufficient in this simulation, some PMs are awake to supply service for VMs and some are asleep. We denominate the awake PMs as active PMs, the result is shown in Fig. 12. The active PMs of Allocation algorithm is more than the best fit algorithm and FCFS algorithm. Although the active PMs are more than other algorithms in the beginning, we understand that the gap between them is decreasing with time increasing due to frequent VM migration. Both FCFS and best fit algorithm do not consider the characteristic of VMs, they only consider the remaining resource of PMs and the resource demand of VMs. Therefore, the unstable VMs may migrate frequently in FCFS and best fit

Table 4 Planning results

	Number of migration during two unit times	Total Cost
Best Fit	4	6
FCFS	5	9
Allocation	2	2

Table 5 Simulation parameters

Parameters	Value
Number of PMs	50
Number of VMs	100
Number of VMs of typeA	20
Number of VMs of typeB	30
Number of VMs of typeC	20
Number of VMs of typeD	30
Capacity of PMs	90 %
Simulation time	10 (unit times)

algorithm. On the other hand, the active PM number of the proposed algorithm is higher than other algorithms. We consider that this weakness is acceptable. Because the numbers of active PM will increase with time in other two algorithms, but the proposed algorithm provides the better VM allocation result. It minimizes the probability of VM migration and the migration cost. In brief summary, we conclude that the proposed Allocation algorithm is a better method that balances the computing loading of VMs in cloud environments.

Then, we investigate the influence on number of VMs and migration times. The simulation parameters are same with Table 5. except the increasing number of VMs, and the number of PMs is fixed. The propotion of four types of VMs is 2:3:2:3 in this simulation. The results are shown as Figs. 13, 14 and 15.

The number of VM Migration with varying number of VMs is shown as Fig. 13. When the number of VMs is increasing, the propability of VM migration also increases. Both best fit and FCFS algorithm suffered from the unnecessary VM migration when the number of VMs increased, best fit algorithm especially. It is encouraged that the proposed VM

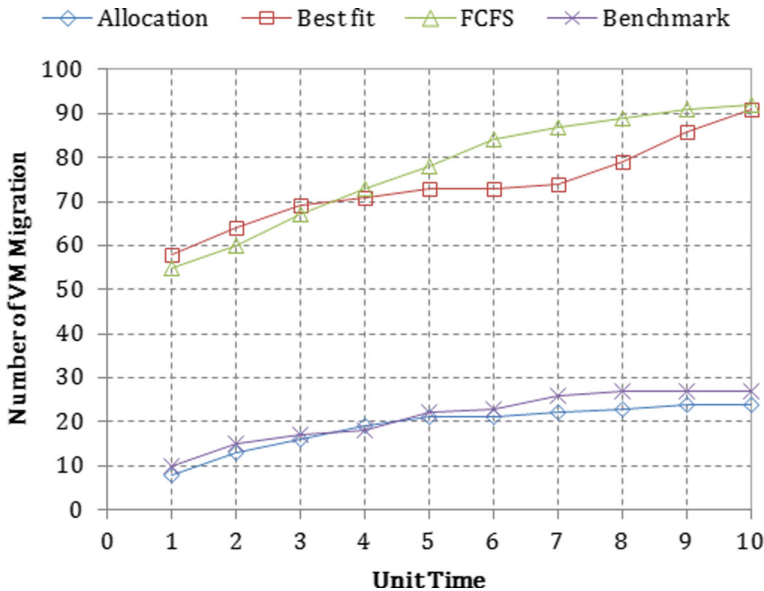


Fig. 10 Simulation results of VM migration with varying time unit

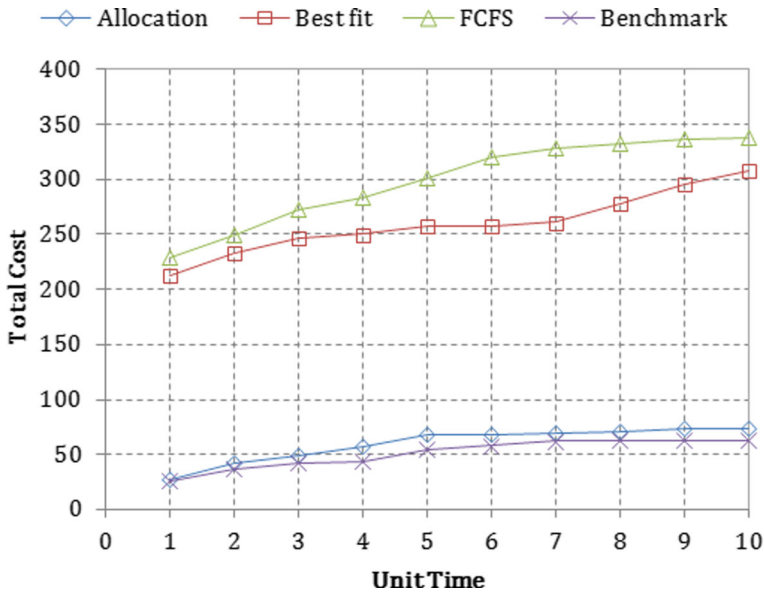


Fig. 11 Simulation results of total cost with varying time unit

Allocation algorithm keeps the lowest VM migration no matter what VM numbers. The total cost with varying number of VMs is shown as Fig. 14, and it has the similar tendency of the VM migration times. When the the number of VMs is increasing, the total cost is also increasing due to VM migration. It is obvious that the proposed VM Allocation algorithm is lower than best fit and FCFS algorithm. The Fig. 15 shows the influence of VM numbers on active PMs. The comprehensive result that the higher number of VMs leads to more active PMs. The proposed algorithm intends to balance the utilization between active PMs, thereby

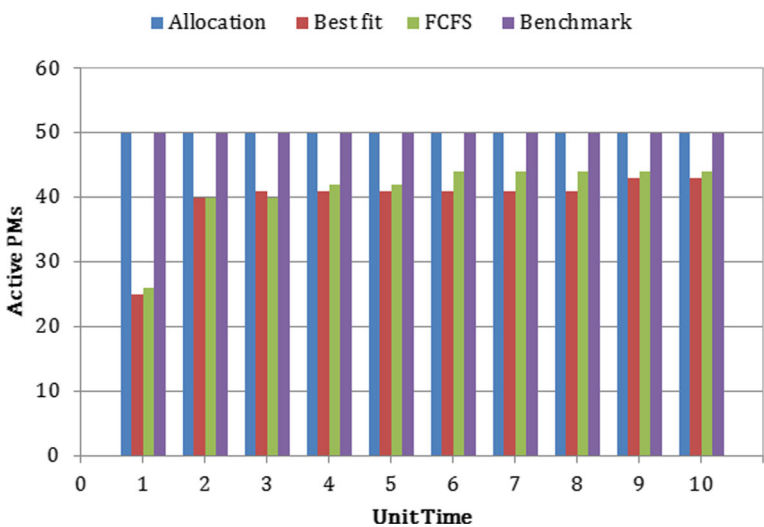


Fig. 12 Simulation results of active PMs with varying time unit

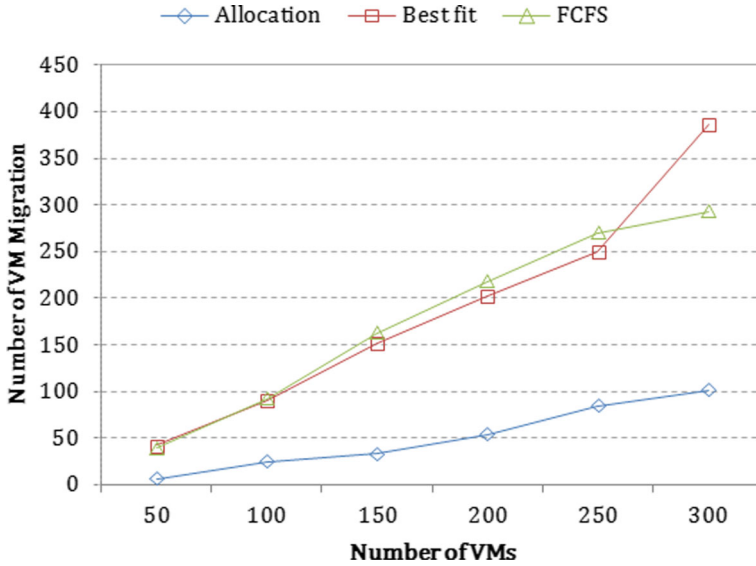


Fig. 13 Simulation results of VM migration with different VM number

it is always higher than other two algorithms. However, we consider that it is acceptable due to the significant improvement on VM migration times and total cost. In summary, the proposed VM Allocation algorithm efficiently decrease the unnecessary VM migrations with the slightly additional PMs.

We examine the influence of different PM capacity, and the simulation results are shown as Figs. 16, 17 and 18. The numbers of VM and PM are same and fixed at 100. The proportion of four VM types is still 2:3:2:3. The number of VM migration is shown in Fig. 16.

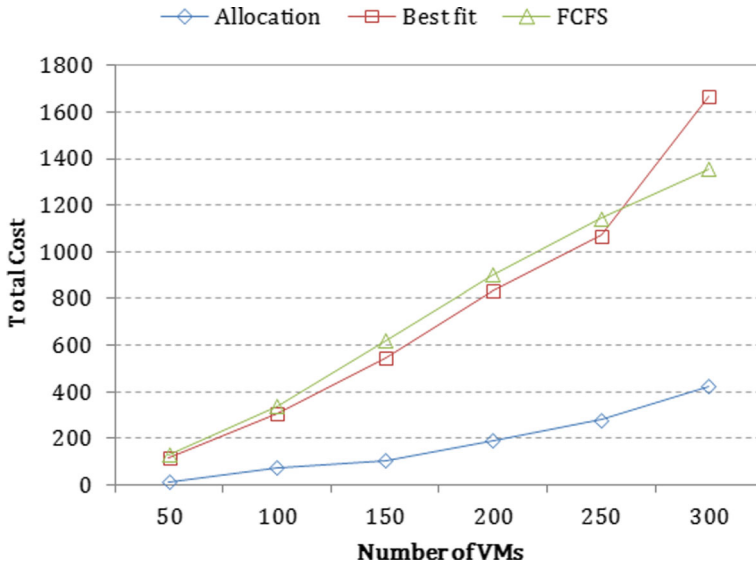


Fig. 14 Simulation results of total cost with different VM number

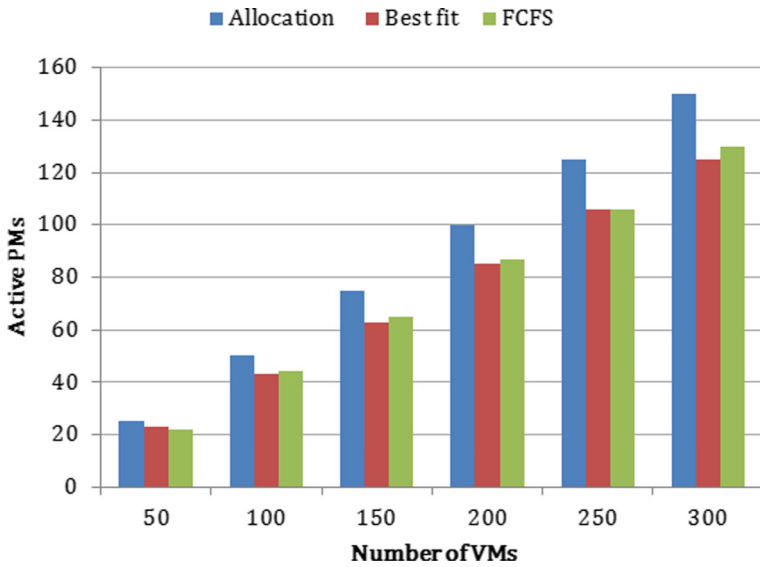


Fig. 15 Simulation results of active PMs with different VM number

and the total cost is shown Fig. 17. No matter what kind of PM capacity, the proposed algorithm is always better than other algorithms. When the capacity of PM is set to 0.8, all of the three algorithms are downside on VM migration numbers and total cost. We found that the value of VM utilization in our simulation case is suitable for the higher PM capacity. In other words, each PM serves more VMs when the capacity value increases from 0.7 to 0.8. The number of active PMs is shown as Fig. 18. The proposed VM Allocation algorithm

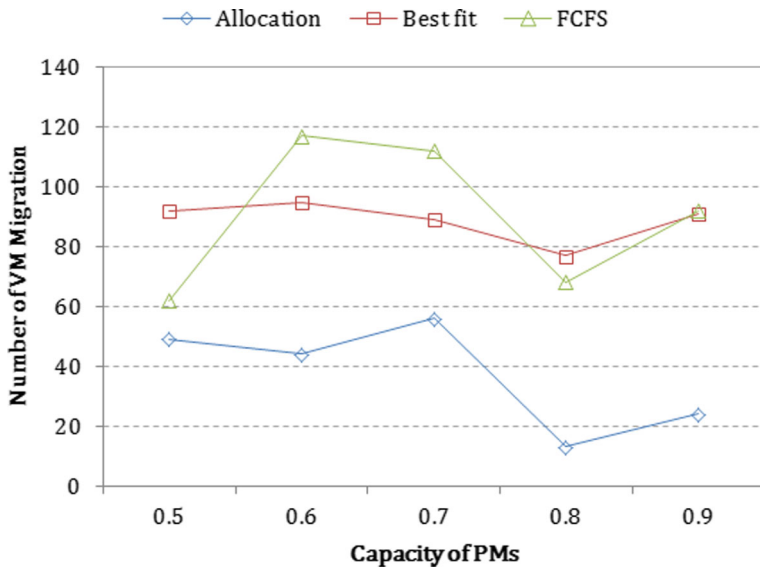


Fig. 16 Simulation results of VM migration with different PM capacity

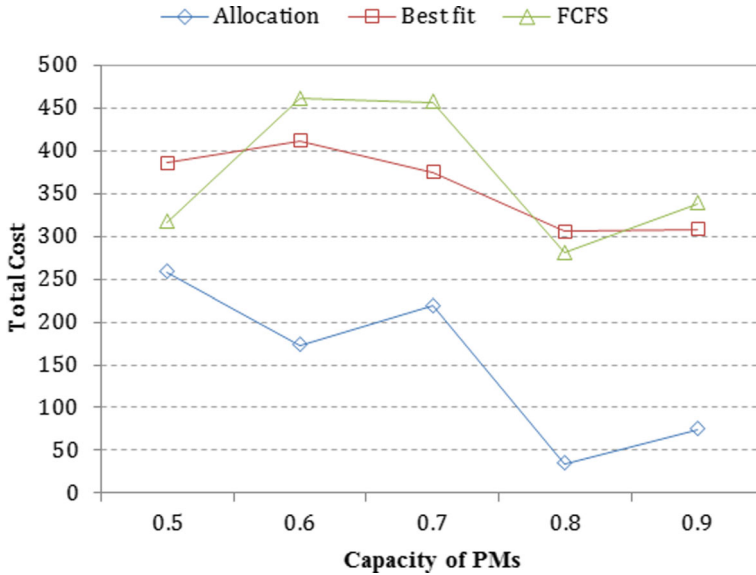


Fig. 17 Simulation results of total cost with different PM capacity

activates less PMs when the capacity of PM is lower than 0.8, but higher than other two algorithms when it is 0.9. Because the proposed algorithm dedicate to balance the loading of PMs, it does not allocate the VMs to a PM if they are possible to migrate from this PM. In other words, the proposed algorithm considers the variance in VM utilization, rather the VM current requirement. We found that the capacity of PMs effects the allocation results defintely, and the proposed algorithm is better than other two algorithms in overall setting.

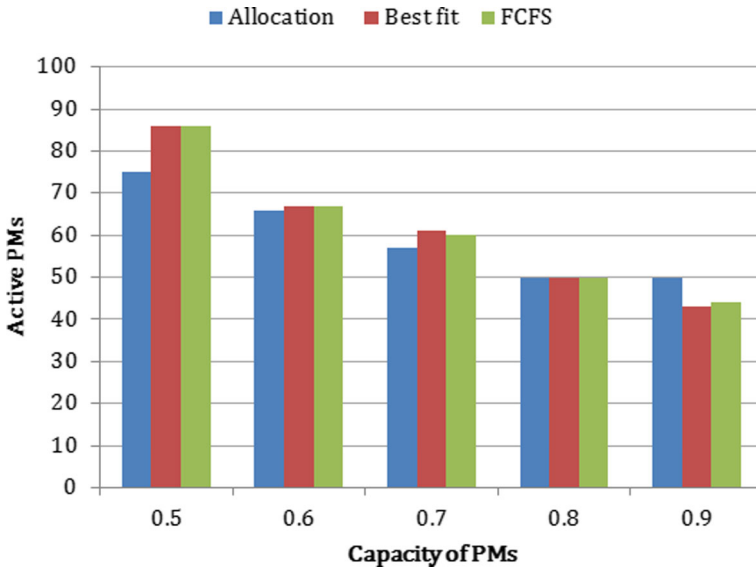


Fig. 18 Simulation results of active PMs with different PM capacity

5 Conclusions

The social media services should be established with the well-defined infrastructure. In this paper, the VM migration problem in cloud data center is formulated based on mixed integer linear programming, and the VM Allocation algorithm is proposed to construct a stable, robust, balanced network. Moreover, we particularly focus on the VM migration. The proposed algorithm not only achieves the lower number of VM migration and total cost, but also balances the utilization between PMs. Furthermore, the proposed algorithm is verified in the simulation results, we believe that the constructed VMs and PMs are suitable for large scale data center.

Acknowledgments This research was supported by the National Science Council (NSC) of the Taiwan under grants NSC 101-2221-E-197-008-MY. This research was also partly funded by the National Natural Science Foundation of China (NSFC) under Grant 61170277, the Innovation Program of Shanghai Municipal Education Commission under Grant 12zz137, the First-class Discipline Construction Project of Shanghai under Grant S1201YLXK, and the Innovation Fund Project for Graduate Student of Shanghai under Grant JWCXSL1202.

References

1. Armbrust M, Fox A, Griffith R, Joseph AD, Katz R, Konwinski A, Lee G, Patterson D, Rabkin A, Stoica I, Zaharia M (2010) A view of cloud computing. *ACM Commun* 53(4):50–58
2. Boser BE, Guyon IM, Vapnik VN (1992) A training algorithm for optimal margin classifiers. In Proc. of the ACM 5th Annual Workshop on Computational Learning Theory
3. Chang C-C, Lin C-J (2011) LIBSVM: a library for support vector machines. *ACM Trans Intell Syst Technol* 2(3):1–39
4. Chen C-Y, Chang K-D, Chao H-C (2011) Transaction-pattern-based anomaly detection algorithm for IP multimedia subsystem. *IEEE Trans Inf Forensic Secur* 6(1):152–161
5. Cheng R-S, Shih M-Y, Yang C-C (2010) Performance evaluation of threshold-based control mechanism for vegas TCP in heterogeneous cloud networks. *Int J Internet Protocol Technol* 5(4):202–209
6. Chou L-D, Chang Y-J, Yang J-Y, Peng W-S (2011) Knowledge management system for social network services. *J Internet Technol* 12(1):139–151
7. Chung H-Y, Chang C-W, Hsiao H-C, Chao Y-C (2012) The load rebalancing problem in distributed file systems. In Proc. of the IEEE International Conference on Cluster Computing
8. Chung W-C, Lin Y-H, Lai K-C, Li K-C, Chung Y-C (2012) A Self-adaptive resource index and discovery system in distributed computing environments. *Int J Ad Hoc Ubiquit Comput* 10(2):74–83
9. Clark C, Fraser K, Hand S, Hansen JG (2005) Live migration of virtual machines. In Proc. of the 2nd USENIX Conference on Networked Systems Design and Implementation
10. Feng Z, Bai B, Zhao B, Su J (2012) Redball: throttling shrew attack in cloud data center networks. *J Internet Technol* 13(4):667–680
11. Hsu H-H, Chen Y-F, Lin C-Y, Hsieh C-W, Shih TK (2012) Emotion attention to friends on social networking services. *J Internet Technol* 13(6):936–970
12. Hu J, Gu J, Sun G, Zhao T (2010) A scheduling strategy on load balancing of virtual machine resources in cloud computing environment. In Proc. of the 3rd International Symposium on Parallel Architectures, Algorithms and Programming
13. Kaplan AM, Haenlein M (2010) Users of the world, unite! The challenges and opportunities of Social Media. *Bus Horiz* 53(1):59–68
14. Khiyaita A, Zbakh M, El Bakkali H, El Kettani D (2012) Load blancing cloud computing: state of art. In Proc. of the National Days of Network Security and Systems:106–109
15. Kim HS, Shih DI, Yu YJ, Eom H, Yeom HY (2012) Systematic approach of using power save mode for cloud data processing services. *Int J Ad Hoc Ubiquit Comput* 10(2):63–73
16. Knig B, Alcaraz CJM, Kirschnick J (2012) Elastic monitoring framework for cloud infrastructures. *IET Commun* 6(10):1306–1315

17. Lai Y-X, Lai C-F, Huang Y-M, Chao H-C (2013) Multi-appliance recognition system with hybrid SVM/GMM classifier in ubiquitous smart home. *Inf Sci* 231:39–55
18. Lai YX, Lai CF, Hu CC, Chao HC, Huang YM (2011) A personalized mobile IPTV system with seamless video reconstruction algorithm in cloud networks. *Int J Commun Syst* 24(10):1375–1387
19. LIBSVM, A Library for Support Vector Machines, Available: <http://www.csie.ntu.edu.tw/~cjlin/libsvm/>. Accessed 15 January 2014
20. Ma F, Liu F, Liu Z (2012) Distributed load balancing allocation of virtual machine in cloud data center. In Proc. of the IEEE 3rd International Conference on Software Engineering and Service Science
21. Mahajan K, Makroo A, Dahiya D (2013) Round robin with server affinity: a VM load balancing algorithm for cloud based infrastructure. *J Inf Process Syst* 9(3):379–394
22. Ostermann S, Iosup A, Yigitbasi N, Prodan R, Fahringer T, Eperna D (2010) A performance analysis of EC2 cloud computing services for scientific computing. In Proc. of the 1st International Conference on Cloud Computing
23. Randles M, Lamb D, Taleb-Bendiab A (2010) A comparative study into distributed load balancing algorithms for cloud computing. In Proc. of the IEEE 24th International Conference on Advanced Information Networking and Applications Workshops
24. Strunk A (2012) Costs of virtual machine live migration: a survey. In Proc. of the IEEE 8th World Congress on Services
25. Wang L, Chen D, Zhao J, Tao J (2012) Resource management of distributed virtual machines. *I J Ad Hoc Ubiquit Comput* 10(2):96–111
26. Wang S-C, Yan K-Q, Liao W-P, Wang S-S (2010) Towards a load balancing in a three-level cloud computing network. In Proc. of the IEEE 3rd International Conference on Computer Science and Information Technology
27. Weng MM, Shih TK, Hung JC (2013) A personal tutoring mechanism based on the cloud environmen. *J Converg* 4(3):37–44
28. Wood T, Shenoy P, Venkataramani A, Yousif M (2007) Black-box and gray-box strategies for virtual machine migration. In Proc. of the 4th USENIX Conference on Networked Systems Design and Implementation
29. Xie X, Jiang H, Jin H, Cao W, Yuan P, Yang L (2012) Metis: a profiling toolkit based on the virtualization of hardware performance counters
30. Zhu K, Song H, Liu L, Gao J, Cheng G (2011) Hybrid genetic algorithm for cloud computing applications. In Proc. of the IEEE Asia-Pacific Services Computing Conference



Fan-Hsun Tseng received his B.S. degree in Electronic Engineering from National Ilan University, Taiwan, in 2008; the Masters degree in Computer Science and Information Engineering from National Ilan University, Taiwan, in 2010. He is currently pursuing his Ph.D. degree in the department of Computer Science and Information Engineering at National Central University, Taiwan. He is a student member of IEEE. His research interests include software defined radio, sensor network applications, cloud computing, WiMAX and LTE-Advanced network.



Xiaojiao Chen received her B.S. degree in Computer Science and Technology from University of Shanghai for Science and Technology, China, in 2007. She is currently pursuing her Master degree in the School of Optical-Electrical and Computer Engineering at University of Shanghai for Science and Technology. Her research interests include cloud computing, virtualization and peer-to-peer network and data mining.



Li-Der Chou received the M.S. degree and the Ph.D. degree in electronic engineering from National Taiwan University of Science and Technology, Taipei, Taiwan in 1991 and 1995 respectively. He is currently a Distinguished Professor at the Department of Computer Science and Information Engineering, National Central University, Taoyuan, Taiwan, where he was also the Director of the Multimedia Communications and Services Division, Computer Center. He is the author or coauthor of more than 100 journal and conference papers in the area of computer networks, and owns 2 U.S. and 13 Taiwan invention patents. His research interests include vehicular networks, network management, broadband wireless networks, and Internet services. Dr. Chou served as the Principal Investigator of an interdisciplinary project in the application of information and communication technologies to families with children with developmental disabilities and individuals with cognitive impairments. Dr. Chou has been invited to join scores of technical program committees of international conferences. He received the Award of Excellence on IPv6 International Appli-Contest 2005 held in Japan, and the Excellent Paper Award on 2010 International Conference on Ubiquitous and Future Networks (ICUFN 2010) held in Korea. Dr. Chou won the Gold Medal Award and the Silver Medal Award in the Invention Contest of 2010 Taipei International Invention Show & Technomart (INST 2010). He also won the Silver Medal Award in the Invention Contest of INST 2009. Dr. Chou was nominated for 2010 National Invention and Creation Award of Taiwan and received the Silver Medal Award in the 2011 British Invention Show held in London, UK. He is a member of the IEEE Communication Society.



Han-Chieh Chao is a joint appointed Full Professor of the Department Computer Science & Information Engineering and Electronic Engineering of National Ilan University, I-Lan, Taiwan (NIU). He is serving as the President since August 2010 for NIU as well. He was the Director of the Computer Center for Ministry of Education Taiwan from September 2008 to July 2010. His research interests include High Speed Networks, Wireless Networks, IPv6 based Networks, Digital Creative Arts, e-Government and Digital Divide. He received his MS and Ph.D. degrees in Electrical Engineering from Purdue University in 1989 and 1993 respectively. He has authored or co-authored 4 books and has published about 400 refereed professional research papers. He has completed more than 100 MSEE thesis students and 4 PhD students. Dr. Chao has been invited frequently to give talks at national and international conferences and research organizations. Dr. Chao is the Editor-in-Chief for IET Networks, Journal of Internet Technology, International Journal of Internet Protocol Technology and International Journal of Ad Hoc and Ubiquitous Computing. Dr. Chao has served as the guest editors for Mobile Networking and Applications (ACM MONET), IEEE JSAC, IEEE Communications Magazine, IEEE Systems Journal, Computer Communications, IEE Proceedings Communications, the Computer Journal, Telecommunication Systems, Wireless Personal Communications, and Wireless Communications & Mobile Computing. Dr. Chao is an IEEE senior member and a Fellow of IET (IEE).



Shiping Chen received the B.S. degree in electrical engineering from JiangXi University of China in 1984. He received the M.S. and PH.D degrees in computer science from the Institute of Computing Technology of the Chinese Academy of Sciences and Fudan University in 1990 and 2006, respectively. He joined the University of Shanghai for Science and Technology in 1990 and is currently a full professor in the School of Optical-Electrical and Computer Engineering. He is also the director of the network center of the university. His research interests include peer-to-peer network, network communications, cloud computing and database systems.