Hindawi Publishing Corporation
International Journal of Distributed Sensor Networks
Volume 2015, Article ID 679170, 14 pages
http://dx.doi.org/10.1155/2015/679170

Research Article

MTAD: A Multitarget Heuristic Algorithm for Virtual Machine Placement

Lei Chen,[1] Jing Zhang,[1] Lijun Cai,[2] Rui Li,[2] Tingqin He,[2] and Tao Meng[2]

[1] College of Electrical and Information Engineering, Hunan University, Changsha, Hunan 410082, China
[2] College of Computer Science and Electronic Engineering, Hunan University, Changsha, Hunan 410082, China

Correspondence should be addressed to Lijun Cai; ljcai@hnu.edu.cn

Received 20 August 2014; Accepted 11 October 2014

Academic Editor: Yu Gu

Cloud data centers are facing increasingly virtual machine (VM) placement problems, such as high energy consumption, imbalanced utilization of multidimension resource, and high resource wastage rate. In order to solve the virtual machine placement problems in large scale, three algorithms are proposed. Firstly, we propose a physical machine (PM) classification algorithm by analyzing pseudotime complexity and find out an important factor (the number of physical hosts) that affects the efficiency, which improves running efficiency through reduction number of physical hosts; secondly, we present a VM placement optimization model using multitarget heuristic algorithm and figure out the positive and negative vectors of three goals using matrix transformation so as to provide the mapping of VMs to hosts by comparing distance with positive and negative vectors such that the energy consumption is saved, resources wastage of occupied PM is lowered, multidimension resource utilization is optimized, and the running time is shortened. Finally, we consider the poor placement efficiency problem of large-scale virtual serial requests and design a concurrent VM classification algorithm using the K-means method. Simulation experiments validate the performance of the algorithm in four aspects, including placement efficiency, resources utilization balance rate, wastage rate, and energy consumption.

1. Introduction

Cloud data center is a new development trend of Internet data center [1] that is a combination of data center and cloud computing. Cloud data centers provide varieties for on-demand computing services consisting of three service models including software as a service (SaaS), platform as a service (PaaS), and infrastructure as a service (IaaS) [2] for users, which is called data center revolution. Currently, data centers benefit a lot from virtualization technology and it has become one of the core technologies of the cloud data center. However, the placement problems of VMs on PMs have always been a huge challenge at the cloud data center.

VM placement problems [1, 3] in cloud data center are a physical resources mapping process of VMs to PMs according to the reasonable allocation rules. The whole processing services are facing many decision objectives and other constraints, which is a complex combinatorial optimization placement problem and it has no optimal mathematical solution in theory. Heuristic algorithm or approximate algorithm is commonly regarded as the useful choice to solve the VM placement. In recent years, a great number of works have been devoted by many scholars to optimize the VM placement algorithm and they have made some great achievements. However, the management of cloud data center is still facing many problems, such as the VM placement efficiency, energy consumption of data center, the balance of the multidimension resources of the physical machine hosts, and the efficiency of serial placement under large-scale virtual requests.

Most traditional VM placement algorithms only consider a single target such as physical host resource utilization or energy consumption, while the motivation of this paper is to put forward new heuristic algorithms for improving the VMs placement efficiency, enhancing virtual machine placement performance, increasing the balance rate of dimension physical resources, and reducing the resource wastage rate of physical host under heterogeneous cloud data center environment. For considering the physical hosts resource utilization, we hope to improve the equilibrium degree of multidimensional resources and reduce resource waste rate of physical host. For considering energy consumption, we need to select the physical hosts which have low energy consumption but strong computing capacity, thus accelerating placement time.

The main contributions can be summarized as follows: (1) we propose an ISPMC (iterative self-organizing physical machine classification) based on ISODATA (iterative self-organizing data analysis techniques algorithm) [4] which improves VM placement efficiency of large Internet data center using the idea of reduction dimensionality. The physical servers can be divided into K classes by ISPMC algorithm according to the number of the PM's CPU and memory, which ensures that the similar physical machines will be divided into similar class. The algorithm improves the virtual machine placement efficiency and reduces the PMs' scanning number by actual mapping of virtual machine requests to available physical machines. (2) We design a multitarget heuristic decision algorithm: MTAD (multitarget approach decision). The algorithm considers multiple objectives at the same time and uses approximate approach to calculate the positive and negative ideal vectors of three objectives. The MTAD realizes the optimal placement of VM by comparing realistic and positive and negative ideal solution approach degree. (3) We propose a k-means algorithm for balancing resource rate, which is called RBRC (resource balance rate classification) for considering the problem of the large-scale VM serial placement efficiency. RBRC algorithm sorts all virtual machine requests to queue in nonincreasing order in terms of resource balanced degree and divided the virtual machine requests into k classes according to the request number and balancing degree to improve the placement efficiency. In addition, if the different dimensions VM resources have a better balance degree, the mapping workload will be a balance status, such that improving the PMs' utilization rate.

The rest of this paper is organized as follows. Section 2 presents related work of the VM placement problem. Section 3 presents VM placement model; in Section 4, we propose three algorithms for optimizing VM placement: SPMC, MTAD, and RBRC, and Section 5 evaluates the performance of the algorithm. Finally, Section 6 summarizes the work and looks forward to the next work.

2. Related Work

Currently, the majority of existing VM placement algorithms in cloud data center can be divided into two categories according to the placement algorithm in terms of server object: (1) the placement algorithm based on the server-side level. It works by using a variety of advanced technologies to improve the power consumption or performance. (2) The placement algorithms based on cluster level mainly consider the global optimization and coordination of the VM placement aiming at using various strategies or theories to reduce the energy consumption and improve performance. For goals, the existing virtual machine placement algorithm can be allocated into two categories: (1) a virtual machine placement algorithm based on energy consumption and emissions reduction which aims to save energy and reduce CO_2 emissions so as to achieve "green" cloud data center. The placement algorithms mainly concentrated on server level, such as dynamic voltage FM technology DVFS and new cooling technique. (2) The second category is a virtual machine placement algorithm based on resource utilization; the purpose is to maximize the physical resource utilization for ensuring application of quality of service (QoS) and meeting the needs of application service-level constraints. We focus on the cluster level on the VM placement algorithm. The current common VM placement algorithms include bin packing stochastic [5], integer programming [6], constraint programming [7, 8], random greedy algorithm [9], simulated annealing [10], genetic algorithm [11, 12], and ant colony optimization algorithm. At present, most of these algorithms are trying to solve the imbalanced use of multidimension resources utilization problem, high energy consumption problem, hyperbole resources wastage problem, and other issues.

The resource utilization rate of PM's different dimensions refers to the imbalanced utilization between different dimensions resources (CPU and memory) in the process of virtual machine placement. Sun et al. [14] build up a two-objective optimization model and propose a virtual machine placement algorithm based on matrix transformation. Two-objective optimization model takes the balance of different dimensions in minimizing number of physical machines and resource utilization into consideration. The two-objective optimization model firstly sets up corresponding virtual machine requests queue matrix, cluster matrix, and the corresponding initialization placement matrix by considering virtual machine placement algorithm based on matrix transformation and then looks for the best result meeting the two goals by the corresponding matrix transformation. We reduce the dimension of physical machine resources by using the concept of vector distance. However, the model does not take the heterogeneous of physical servers into account and complexity of solving balancing rate based on vector distance. Wang et al. [15] put forward a multiobjective genetic evolution VM placement algorithm. Firstly, the algorithm constructs the model by considering multiobjective such as balancing utilization rate of PM's different dimensions or maximizing resource usage and then considers the average inequality and position constraint conditions to solve the virtual machine placement problem using the genetic evolutionary algorithm by way of heuristic iterative mutation virtualized allocation. However, the resource usage algorithm is poor and has low time efficiency with a large number of VM requests. In [16], Lu et al. propose a two-stage virtual machine consolidation placement algorithm. In the first stage, they use a polynomial approximate load balance scheme PTAS and adjust the different VM placement and then migration and integration to maximize the maximum resource utilization. In the second stage, they balance the load differences of physical servers using queuing model and reduce the imbalance of different dimensions resources. The two-stage algorithm is the objective process of virtual machine migration and integration, but the adjustment of dynamic resource takes a long time and has poor flexibility.

Energy saving [3, 17] is another main developing direction of VM placement algorithm. Shi et al. [18] put forward a kind of vector packing algorithm; PMs can be seen as a box and

VMs can be seen as items into box, using the greedy thought to allocate virtual machine placement with the minimum possible physical machine and shut down the other unused physical machine so as to reduce the number of physical machines used to achieve the purpose of saving energy. However, the algorithm does not take heterogeneous physical servers constraints, and greedy placement methods cannot achieve the optimal global solution. In [19], Wu et al. firstly attempt to solve the VM placement problem by using the simulated annealing to achieve the energy saving. On this basis, Dhingra and Paul [20] use optimized simulated annealing for VM placement process, looking for energy consumption by random iteration better physical host servers. It ensures virtual machine's SLA server degree, increases the resources utilization, and reduces the energy consumption. Tang and Pan [13] propose a virtual machine placement algorithm based on genetic algorithm. Such algorithms consider the energy consumption of data center's physical host and the energy consumption of network communication by generating a random initial population and then use multiple gene mutations to look for minimum energy consumption of physical hosts and network communication. However, the algorithm just considers one network structure and has high computational complexity and low allocating efficiency. In [21], Song et al. propose a large-scale convex optimization algorithm for virtual machine placement. The algorithm improves the virtual machine placement by using the theory of convex optimization to convert the VM placement problem into multiobjective optimization problems according to the actual data center network architecture.

Currently, most optimization virtual machine placement algorithms are converting multiple allocation objectives into several single-objectives and rarely simultaneously optimize multiple targets. Therefore, most of them just obtain a local solution rather than the global optimization solution.

3. Virtual Machine Placement Model

3.1. Notation Used in Virtual Machine Placement. Before giving the multitarget approximating virtual machine placement model, some notations used in subsequent section are listed in Notations section.

Notations used in virtual machine placement are shown in Notations section.

3.2. A Multiobjective Approximating Virtual Machine Placement Model. The VM placement process in multiobjective approach way is as follows. According to the multitarget decision-making (wastage rate, balance rate, energy consumption, etc.), a series of multidimensions (CPU, memory, etc.) of random VM requests conduct periodic optimization of the decision-making process to determine the physical placement programs. Multiobjective approximating virtual machine placement model considers the following objectives:

(1) improve placement algorithm efficiency;

(2) balance physical servers load, reduce wastage rate of cluster resources, and improve resource utilization;

(3) optimize resource balance rate on different dimensions;

(4) reduce the number of high-consumption physical machines used and improve energy-efficiency and scalability of data centers.

3.2.1. Improve Placement Algorithm Efficiency. Optimal placement algorithm efficiency and reducing placement time. VM placement is NP-hard problem, which cannot be solved in polynomial time but in a pseudopolynomial time.

Theorem 1. *Pseudopolynomial time of virtual machine placement problem is $O(n^2 \cdot O(Z_{pro}^{\max}))$; the time required for the maximum target solution is $O(Z_{pro}^{\max})$.*

Assumptions 1. We have the following

The set of virtual machines request is

$$\{v_1\,(\text{cpu}, \text{mem})\,, v_2\,(\cdots)\,, \ldots, v_n\,(\cdots)\}\,. \tag{1}$$

The set of physical host servers is

$$\{p_1\,(\text{cpu}, \text{mem})\,, p_2\,(\cdots)\,, \ldots, p_m\,(\cdots)\}\,. \tag{2}$$

The objective function set of virtual machines placement is $\{f_1(x), f_2(x), \ldots, f_l(x)\}$.

Proof. The time complexities of virtual machine placement problem can be expressed as follows.

Firstly, scan the virtual machine requests i, $i = 1 \rightarrow n$, one by one; traversal matched physical machines according to decision-making objective function j, $j = 1 \rightarrow m$; solve decision-making optimization objective $f_k(x)$, $k = 1 \rightarrow l$, for physical machines j one by one.

Let $O(Z_{\text{pro}}^{\max})$ be the maximum time required for solving the objective function, $O(Z_{\text{pro}}^{\max}) = \max\{O(f_1(x)), \ldots, O(f_l(x))\}$.

So, the maximum time complexity of virtual machine is allocated as follows: $O(n * m * O(Z_{\text{pro}}^{\max})) = O(n^2 O(Z_{\text{pro}}^{\max}))$; n is the dimension number of the physical host.

According to Theorem 1, we can solve the objective function itself from the beginning and reduce the used number of physical servers to improve the VM placement algorithm efficiency. □

3.2.2. Balance Physical Servers Load, Reduce Wastage Rate of Cluster Resources, and Improve Resource Utilization. Wastage rate Waste_{p_j} refers to the average ratio value between different dimension remaining resources and the whole physical server, as shown in (3). Cluster resource wastage rate refers to the average value of different dimensions resources wastage rate on physical machine servers which carries virtual machines. Consider

$$\text{Waste Rate} = \begin{cases} \text{Waste}_{P_j}^{\text{Cpu}} = \dfrac{p_j^{T\text{cpu}}}{p_j^{\text{cpu}}} \\[2mm] \text{Waste}_{P_j}^{\text{Mem}} = \dfrac{p_j^{T\text{mem}}}{p_j^{\text{mem}}} \\[2mm] \text{Waste}_{p_j} = \dfrac{\left(\text{Waste}_{P_j}^{\text{Cpu}} + \text{Waste}_{P_j}^{\text{Mem}}\right)}{2} \\[2mm] \text{Waste}_{P} = \displaystyle\sum_{j=0}^{m} \text{Waste}_{p_j}. \end{cases} \tag{3}$$

Reduce wastage rate of cluster resources; on one hand, we need to improve resource utilization of single physical machines; on the other hand, we need to strengthen the resource balance rate on different dimensions of physical servers. Optimized cluster resource wastage rate can be expressed as

$$\min f_1(x) = \min \text{Waste}_P = \sum_{j=0}^{m} \frac{\left(p_j^{T\text{cpu}}/p_j^{\text{cpu}} + p_j^{T\text{mem}}/p_j^{\text{mem}}\right)}{2}. \tag{4}$$

3.2.3. Optimize Resource Utilization Rate on Different Dimensions. Considering multidimensional PM's resources (CPU, memory, storage, etc.), resource utilization rate on different dimensions should be taken into account with placement virtual machines, ensuring similarity load among multidimensional resources, to maximize improving resource utilization of physical servers. Resource placement balancing comparison is shown in Figure 1.

As Figure 1(a) shows, PM's CPU utilization has been up to 90%, with only 22% of memory utilization. The virtual machine requests include (CPU and memory) resources due to full capacity utilization of the PM's CPU which cause PM not to carry more virtual machines, and 68% of the memory resources are idle wasted. Figure 1(b) shows that CPU and memory resources are up to 90% utilization; all dimensions resources have a balanced utilization, and PM's different dimensions reached the best benefit. As time goes by, the virtual machines demise; CPU and memory utilization can maintain a good balance, which ensures the full use of different dimensions resources and maximizes PM resource utilization. However, due to the randomness of the virtual machines requests and randomness of the demand for resources in each dimension, which could not ensure accuracy balancing of different dimension resources. Therefore, the more the balanced utilization of each dimension resources is, the more PM's resources are fully used.

In order to ensure balance utilization of PM's different dimensions resources, the physical server must try to host a similar virtual machine request with their remaining resources, so we use vector angle to measure the similarity between virtual machines and physical server. Without considering the physical storage attributes, we handle virtual machine requests as a two-dimensional vector $\vec{v}\,\langle \text{CPU}, \text{MEM}\rangle$. The remaining resources of physical hosts can be seen as a two-dimensional vector $\vec{p}\,\langle \text{CPU}, \text{MEM}\rangle$. Just make sure that the two vectors are mutually parallel and keep resources relatively balancing after physical servers hosting virtual machines. Therefore, we use vector angle to measure impact on virtual machine requests to the physical machine resources, the smaller the angle is, the better balancing of different dimension resources will be after the physical machines have hosted virtual machines. Vector angle formula is as follows:

$$\cos\langle \vec{v}, \vec{p}\rangle = \frac{\vec{v}\cdot\vec{p}}{|\vec{v}|\cdot|\vec{p}|}, \tag{5}$$

$$\vec{v}\cdot\vec{p} = v^{\text{cpu}}\left(p^{\text{cpu}} - p^{T\text{cpu}}\right) + v^{\text{mem}}\left(p^{\text{mem}} - p^{T\text{mem}}\right), \tag{6}$$

$$|\vec{v}|^2 = \vec{v}\cdot\vec{v} = \left(v^{\text{cpu}}\right)^2 + \left(v^{\text{mem}}\right)^2, \tag{7}$$

$$|\vec{p}|^2 = \vec{p}\cdot\vec{p} = \left(p^{\text{cpu}} - p^{T\text{cpu}}\right)^2 + \left(p^{\text{mem}} - p^{T\text{mem}}\right)^2. \tag{8}$$

Therefore, in order to optimize resource balance rate of physical machines, we need to minimize the vector angle between virtual machines and physical machines; objectives are as follows:

$$\min f_2(x) = \sum_{v_i\in V}\sum_{p_j\in P} \cos\left\langle v_i, p_j\right\rangle. \tag{9}$$

3.2.4. Saving the Energy Efficiency. Physical hosts in cloud data center present heterogeneous structure, usually composed by a variety of different structures physical hosts. If we only reduce the PM usage number for saving energy consumption, usually it is inefficient because small amounts of high-performance energy PMs may cause far greater energy consumption than multiple low-power physical hosts. Therefore, multiobjective approach model of this paper for energy saving includes two aspects as follows.

(1) Reducing the Number of High Energy Physics Hosts. In virtual machine placement process, the multitarget approach models will choose smaller power consumption physical host among the physical hosts that meet virtual machine request. Placement principles are as follows: as far as possible to assign virtual machines to physical host that already carried other virtual machines to achieve multiple virtual machines sharing physical host; when a new physical host is needed to open, try to select the low power consumption physical host. Therefore, energy savings objective function can be defined as follows:

$$\min f_3(x) = \sum_{j\in P_{\text{used}}} p_j^{\text{energy}}. \tag{10}$$

(2) Extend the Physical Host Opening and Closing Cycle. Due to the randomness of the virtual machine requests and randomness of life cycle, the frequent switching physical hosts will inevitably lead to additional energy consumption and cause long switching machine cycle, seriously affecting the placement efficiency and performance of virtual machines. Therefore, the hosts need a certain degree of extending the

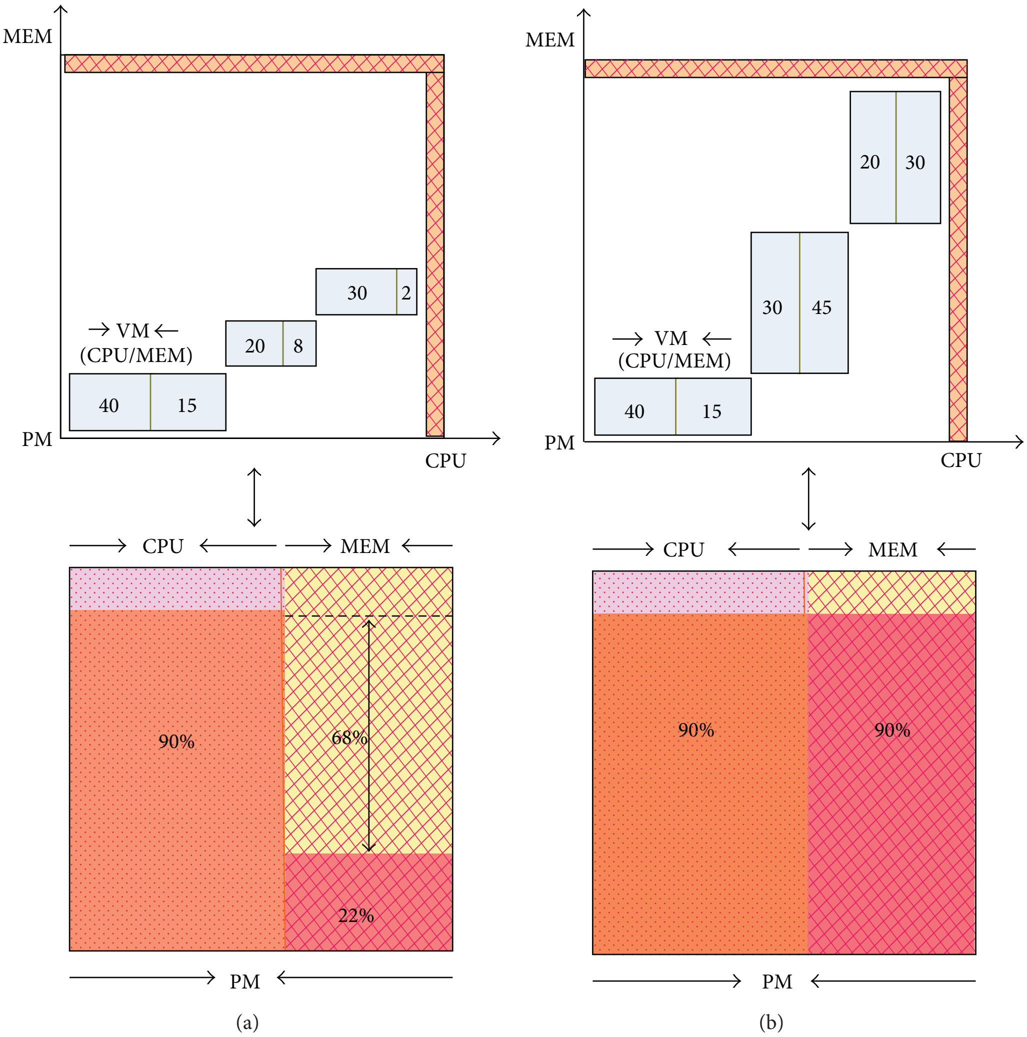


Figure 1: PM multidimensional resource balance.

switching cycles. The model proposes a solution by using a fixed waiting period. We the expert database to determine the physical host's closing period T as a fixed waiting time so as to avoid frequent switching the servers. During the wait period, the physical host is idle, waiting for the arrival and placement of virtual requests.

3.2.5. The Objective Function. Therefore, the objective function of multiobjective approach virtual machine placement model is as follows:

$$\begin{aligned}\min f(x) &= \alpha * f_1(x) + \beta * f_{12}(x) + \gamma * f_3(x) \\ &= \alpha * \left[\sum_{j=0}^{m} \frac{\left(p_j^{T\text{cpu}}/p_j^{\text{cpu}} + p_j^{T\text{mem}}/p_j^{\text{mem}}\right)}{2}\right] \\ &\quad + \beta * \left[\sum_{v_i \in V} \sum_{p_j \in P} \cos\left\langle v_i, p_j\right\rangle\right] \\ &\quad + \gamma * \left[\sum_{v_i \in V} \sum_{j \in P} p_j^{\text{energy}}\right].\end{aligned} \tag{11}$$

4. A Multitarget Heuristic Algorithm for Virtual Machine Placement: MTAD

4.1. Physical Host Classification Algorithm (ISPMC). The VM placement problem in cloud data center is NP-hard problem; it cannot be solved in polynomial time. From Theorem 1, we know that pseudopolynomial time of virtual machine placement problem is $O(n^2 \cdot O(Z_{\text{pro}}^{\max}))$, and the time complexity of the maximum target of all solutions is $O(Z_{\text{pro}}^{\max})$. In case of not considering objective function solving time, we must reduce the number of physical hosts n to enhance the efficiency of placement algorithm; the smaller n is, the faster virtual machine placement is.

The main function of ISPMC classification algorithm is to classify the heterogeneous PM hosts according to the cluster resource types. So divide the large number of physical hosts into K set that have similar structure. When the virtual machine is allocated, we use a single physical set as target groups so as to reduce the number of placement algorithm for computing the physical host n and accelerate virtual machine requests placement rate.

```
Input:
(1) Read expert parameter database, get pre-classification number $U$.
(2) The set of physical hosts $P\{p_i, p_i \in P\}$.
Output: The category set of physical hosts
  For Iterations $I$ do
    (1) According to the Euclidean distance to get the nearest neighbor clustering, calculate the cluster domain-related information, the cluster center and the average distance between the category and the global average distance
    For Initial cluster do
      IF $\overline{D}_j > \overline{D}$ and $N_k > 2(\theta_n + 1)$ then stop splitting,
        jump out of loop.
      ELSE IF $K \le U/2$ then split, jump out of loop.
      ELSE IF iterations are even times or $U \ge K \ge U/2$
        then merge, jump out of loop.
      ELSE IF iterations reach $\mathbf{I}$ times, the last iteration,
        then $\theta_c = 0$, then merge, jump out of loop.
      End IF
    End For
  End For
```

Algorithm 1: PM classification algorithm: ISPMC.

Due to the heterogeneous characteristics of the cloud data center physical host, the classification parameters of physical host will be determined by actual number of physical machines Type U and other information, and dynamically construct parameters expert database that managed by the system maintenance people. Also, the number of classification U in expert library has great subjective and arbitrary, which may cause to lower actual classification performance. Therefore, we propose a physical host classification algorithm ISPMC based on the ISODATA clustering algorithm. Compared with ISODATA, ISPMC algorithm focuses on the following optimization.

(1) To achieve PM classification, we use the number of physical host CPU and memory as clustering properties and calculate coordinates in ISPMC algorithm. The amount of memory is enormous magnitude units which need to be reduced correspond, we can divided it by 512 as a clustering criterion.

(2) Compared to the ISODATA algorithm, ISPMC algorithm reduces the input initial parameters according to the actual situation of physical hosts. The reducing initial parameters are as the following: (1) initialize the cluster center. Because the ISPMC algorithm parameters expert database has already stored species quantity U of physical host, ISPMC algorithm can randomly obtain different types hosts to composite initialize cluster centers. (2) Reduce the complexity of the sample standard deviation calculation according to threshold θ_s. In ISODATA algorithm, it uses distance between the auxiliary samples to judge whether cluster needs splitting according to the fluctuation degree (standard deviation) of sample and the cluster center; as we know, the number of samples is huge, multidimensional standard deviation calculation is time consuming, and the effectiveness is low, so ISPMC algorithm combined with the actual physical host classification requirements, using the distance between the hosts to determine whether a cluster classification is needed, shielding the complexity of the sample standard deviation calculation, optimizes the sample standard deviation of the complex calculation.

(3) Adjust the splitting standard of ISODATA. When the number of clusters is two times larger than the predicted number of U, ISODATA algorithm will not conduct data division; for PM classification, the actual classification of the data should not exceed two times the predicted number. SPMC algorithm adjusts the splitting of the original ISODATA algorithm standard $2U$ and ensures ISPMC classification number K satisfying $U/2 < K < 2U$. The optimizing of ISPMC classification algorithm can help to solve virtual machines placement problem.

(4) ISPMC algorithm improves the classification time by abandoning standard deviation calculation for all clusters between samples and classification standards, greatly reduces the classification of computing time, and improves the classification efficiency.

Table 1 shows the initial parameters of ISPMC algorithm which includes four stages, through repeatedly self-organization iteration to achieve physical host classification.

Pseudocode for ISPMC algorithm is shown in Algorithm 1.

Stage 1: *Initialize the Environment and Calculate Clustering Information.*

Step 1. Initialize the physical environment, read expert parameter database to obtain preclassification number U, input physical host set $P\{p_i, p_i \in P\}$, and generate the initial cluster centers $Z\{z_1, \ldots, z_k\}$; CPU and memory are two-dimensional properties of the physical hosts; U is equal to the preclassification number K.

Table 1: ISPMC placement algorithm parameters.

Vars.	Description
U	The expected number of (physical classification number) classification
θ_n	The minimum number of physical machines in each category. If the physical machine number is less than it, then it is not a classification
θ_c	The minimum distance between the two clusters. If the number is smaller than it, merge the two clusters
L	The merger standard maximum clusters number of each iteration
I	Maximum number of iterations

Step 2. According to the Euclidean distance of initial cluster centers, classify the physical host.

Step 3. According to (12), correct each cluster domain center z_k; by (13), calculate the distance between various cluster domain center physical hosts and cluster center field $\overline{D}_k$; calculate the maximum between various cluster domain center physical hosts and cluster center field component $\text{Differ}_k^{\max}$ such as the maximum between CPU of various cluster domain center physical hosts and CPU of cluster center field component $\text{Differ}_k^{\text{cpu}}$ and memory maximum $\text{Differ}_k^{\text{mem}}$; by (14), calculate the total average distance between physical host and corresponding cluster center as

$$z_k = \frac{1}{N_k} \sum_{p \in z_k} p, \quad k = 1, 2, \ldots, K, \tag{12}$$

$$\overline{D}_k = \frac{1}{N_k} \sum_{p \in z_k} \left\| p - z_k \right\|, \quad k = 1, 2, \ldots, K, \tag{13}$$

$$\overline{D} = \frac{1}{N} \sum_{k=1}^{K} N_k \overline{D}_k. \tag{14}$$

Stage 2: Splitting Determination and Merging Operations.

Step 4. Cluster splitting determination, merger and iteration.

(1) If the number of iterations has been reached i times: the last iteration, then $\theta_c = 0$; go to Step 6.

(2) If the host number in cluster $N_k < \theta_n$, stop the classification, $k = k - 1$; go to Step 2.

(3) If $K \leq U/2$, that is, half of the clusters center number is less than or equal to the predicted value, go to Step 5 and split the existing clustering process.

(4) If the number is an even number of times of the iteration or $U \geq K \geq U/2$, then there is no splitting; go to Step 6. Otherwise, go to Step 5. Iteration to an even number is for fair dealing the merger and split operations.

(5) If it is the last iteration, the algorithm ends; otherwise, if it is changing the input parameters, go to Step 1; if not, go to Step 2.

Stage 3: Cluster Splitting.

Step 5. Judge whether the cluster meets one of the following two conditions.

(1) $\overline{D}_j > \overline{D}$ and $N_k > 2(\theta_n + 1)$, such that total number of z_k classification samples exceeds the specified value more than double.

(2) $K \leq U/2$.

If this is true, split z_k into two new cluster centers z_k^+ and z_k^-, $K = k + 1$. Each corresponding component of the cluster centers in z_k^+ plus $\text{Differ}_k^{\max}$; each corresponding component of z_k^- is equal to the cluster centers' component minus $\text{Differ}_k^{\max}$ and finishes splitting operations; go to Step 2. Otherwise, go to Step 4.

Stage 4: Cluster Merging.

Step 6. According to formula (15), calculate the distance of all cluster centers as follows:

$$D_{ij} = \left\| z_i - z_j \right\|, \quad i = 1, 2, \ldots, K - 1; \; j = i + 1, \ldots, K, \tag{15}$$

$$z_k^* = \frac{1}{N_{i_k} + N_{j_k}} \left[N_{i_k} z_{i_k} + N_{j_k} z_{j_k} \right], \quad k = 1, 2, \ldots, L. \tag{16}$$

Step 7. Compare D_{ij} with θ_c in ascending order by cluster distance to form a set $\{D_{i_1 j_1}, D_{i_2 j_2}, \ldots, D_{i_L j_L}\}$; that is, $D_{i_1 j_1} < D_{i_2 j_2} < \cdots < D_{i_L j_L}$.

Step 8. According to formula (16), merge the two cluster centers z_{i_k} and z_{j_k} when the distance is $D_{i_k j_k}$ and then get new center z_k^*. The two merged cluster centers vectors were, respectively, divided by the number of clustering domain weighted samples; ensure z_k^* as a real averaging vector.

4.2. A Multitarget Heuristic Algorithm for Virtual Machine Placement: MTAD. In Section 3, we present a multitarget heuristic virtual machine placement model; MTAD algorithm includes three objectives: resource wastage rate, different dimension resource utilization rate of physical hosts, and reducing the energy consumption, using approximate approximation method to sort all solutions, select the fit highest multiattribute physical host as the mapping entity, and complete the virtual machine placement. The basic idea of MTAD algorithm is based on the resources of the virtual machine requests; select hosts which meet the conditions of physical host and solve the three dimensions targets by forming a raw data matrix. According to the different sizes of three targets data, normalize the original matrix to get a normalized matrix and work out the best and worst schemes that have the maximum positive closeness and minimum negative closeness.

Pseudocode for MTAD algorithm is shown in Algorithm 2.

The basic steps of MTAD algorithm are as follows.

Step 1. Traverse the virtual machine placement request queue $V\{v_1, \ldots, v_n\}$.

```
Input: (1) Virtual Machine request set $V\{v_1, \dots, v_n\}$
       (2) Input the physical host set $P\{p_i, p_i \in P\}$.
Output: The set of virtual machines mapping
For virtual machine requests set do
  (1) According to the virtual machine requests, select the proper physical hosts
  (2) Form the original decision matrix according to the three properties of the target
  (3) Normalize the original decision matrix to form a matrix of normalized
  (4) According to the attributes weights to form the judgment matrix.
  (5) Looking for the positive and negative ideal solution of multi-attribute
  (6) Calculate closeness between attributes and ideal solution to determine the final placement
End For
```

Algorithm 2: MTAD: multitarget heuristic algorithm for virtual machine placement.

Step 2. Select the physical hosts set $P^i\{p_1, \dots, p_m\}$ which meets the virtual machine requests.

Step 3. According to the three decision attributes of multitarget approach model, for physical hosts set $P^i\{p_1, \dots, p_m\}$, use the objective attribute function to solve the property values x_{ij} and form an initial judgment matrix as follows:

$$J\text{matrix} = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1n} \\ x_{21} & x_{22} & \cdots & x_{2n} \\ x_{31} & x_{32} & \cdots & x_{3n} \end{bmatrix}. \tag{17}$$

Step 4. Because the attribute values may have different units, the original decision matrix needs to be normalized according to formula (19); form a normalized matrix $J\text{matrix}'$ as follows:

$$J\text{matrix}' = \begin{bmatrix} x'_{11} & x'_{12} & \cdots & x'_{1n} \\ x'_{21} & x'_{22} & \cdots & x'_{2n} \\ x'_{31} & x'_{32} & \cdots & x'_{3n} \end{bmatrix}, \tag{18}$$

where

$$x'_{ij} = \frac{x_{ij}}{\sqrt{\sum_{j=1}^{n} x_{ij}^2}}, \quad i = 1, 2, 3. \tag{19}$$

Step 5. Form a weighted judgment matrix Z according to the target attributes weights as

$$\begin{aligned} Z = J\text{matrix}'B &= \begin{bmatrix} w_1 & 0 & 0 \\ 0 & w_2 & 0 \\ 0 & 0 & w_3 \end{bmatrix} \begin{bmatrix} x'_{11} & x'_{12} & \cdots & x'_{1n} \\ x'_{21} & x'_{22} & \cdots & x'_{2n} \\ x'_{31} & x'_{32} & \cdots & x'_{3n} \end{bmatrix} \\ &= \begin{bmatrix} f_{11} & f_{12} & \cdots & f_{1n} \\ f_{21} & f_{22} & \cdots & f_{2n} \\ f_{31} & f_{32} & \cdots & f_{3n} \end{bmatrix}, \end{aligned} \tag{20}$$

where $w_1 + w_2 + w_3 = 1$.

Step 6. Get the positive ideal solution and negative ideal solution which are used for evaluating targets according to the weighted comparison matrix Z as follows:

(1) positive ideal solution:

$$f_i^* = \max\left(f_{ij}\right), \quad i \in 1, 2, 3, \tag{21}$$

(2) negative ideal solution:

$$f_i' = \min\left(f_{ij}\right), \quad i \in 1, 2, 3. \tag{22}$$

Step 7. Calculate the Euclidean distance between the ideal solution values of positive and negative solution as

$$\begin{aligned} S_j^* &= \sqrt{\sum_{i=1}^{3} \left(f_{ij} - f_i^*\right)^2}, \quad j \in 1, \dots, n, \\ S_j' &= \sqrt{\sum_{i=1}^{3} \left(f_{ij} - f_i'\right)^2}, \quad j \in 1, \dots, n. \end{aligned} \tag{23}$$

Step 8. Calculate the relative closeness of each target as

$$C_j^* = \frac{S_j'}{\left(S_j^* + S_j'\right)}, \quad j = 1, 2, \dots, n. \tag{24}$$

Step 9. Use the relative closeness C_j^* size to sort and get a final decision and solve the next virtual machine.

4.3. Virtual Machine Classification Algorithm Based on Balancing Rate: RBRC.

With the continuous development of cloud computing technology and the expanding size of the cloud data center, the virtual machines concurrent placement requests are becoming increasingly huge. Large-scale virtual machine placement requests have brought unprecedented challenges to the traditional serial placement algorithm. So, we propose a K-means virtual machine classification algorithm (RBRC) based on balancing utilization

```
Input: Virtual Machine request Set $V\{v_1, \ldots, v_n\}$
Output: $K$ virtual machine classification set
   IF number of $V$ is bigger than $K$ then
    (1) Convert the virtual requests into two- dimensional
   vector $\vec{v}\,\langle \mathrm{CPU}, \mathrm{MEM}\rangle$, with vector $\langle 1, 1\rangle$, Calculate
   angle $|\vec{v}|$
    (2) Ascend order $|\vec{v}|$, select $K$ initial point as the center
point in stepwise way.
    While $K$ clusters have changed do
      (3) re-clustering according to the Euclidean distance
      (4) Calculate the center point of each cluster.
    End While
   End IF
```

Algorithm 3: Virtual machine classification algorithm based on balancing rate: RBRC.

rate; the algorithm ensures the balancing rate degree of virtual machine requests and dynamically divides the virtual machine requests into K-class according to the K-class physical host partition achieved from ISPMC algorithm so as to improve the efficiency of virtual machines placement and load balancing between different classified physical hosts.

Definition 2 (K-means). Input parameter k, divide the set of n objects into K clusters, ensure within the clusters having high similarity, such that the clusters having low similarity.

Definition 3 (Euclidean distance). Euclidean distance is defined as follows:

$$d(i, j) = \sqrt{\left(\left|x_{i1} - x_{j1}\right|^2 + \left|x_{i2} - x_{j2}\right|^2 + \cdots + \left|x_{ip} - x_{jp}\right|^2\right)}, \qquad (25)$$

where $i = (x_{i1}, x_{i2}, \ldots, x_{ip})$ and $j = (x_{j1}, x_{j2}, \ldots, x_{jp})$ are the two p-dimension data objects.

RBRC algorithm process is as follows.

Step 1. Input the virtual machine requests. Set $V\{v_1, \ldots, v_n\}$ and judge whether the number of V is less than or equal to K; if true, end algorithm; otherwise, go to Step 2.

Step 2. Convert the virtual requests into two-dimensional vector $\vec{v}\ \langle \mathrm{CPU}, \mathrm{MEM}\rangle$, according to formula (5) and vector $\langle 1, 1\rangle$. Calculate angle value $|\vec{v}|$.

Step 3. Ascend order $|\vec{v}|$ and select K initial point as the center point in stepwise way.

Step 4. Loop from Step 5 to Step 6 until the cycles do not change in each cluster anymore.

Step 5. Traverse V and perform neighbor clustering according to the virtual machine requests and Euclidean distance of K center point (25) to form k clusters.

Step 6. Recalculate the center vector of each cluster; each component of the vector is the average value of all objects' component in cluster.

Pseudocode for RBRC algorithm is shown in Algorithm 3.

5. Experimental Simulation

In this paper, we use cloud computing platform CloudSim3.5 [22] as a simulation tool to compare ISPMC, MTAD, and RBRC algorithms with several VM placement algorithms and verify the placement efficiency of ISPMC and RBRC algorithms. At the same time, we evaluate the performance of the MTAD algorithm by considering placement efficiency, resource wastage rate, multidimension resources balance rate, and physical machine energy consumption; simulation results are illustrated; the MTAD algorithm has better performance than other algorithms.

5.1. Simulation Environment. In the CloudSim platform, physical machine requests and virtual machine placement are generated in the random way. We design multiple classes, such as the data center, host, VM, and DataCenterBroker, and implement the simulation of the VM and PM. We optimize CloudSim simulation so as to submit repeatedly virtual machine allocation requests in multibatch way by using the multithread. Therefore, we can simulate the placement of virtual machine requests on more reality environment (because real virtual machine placement is a dynamic change process, physical machine hosts may have already loaded some virtual machine requests). Strategies generated for the physical and virtual machine placement are listed as follows.

5.1.1. Physical Machine. Random generation is adopted to define class DataCenterCharacteristics for generating the corresponding DataCenter and main physical machine hosts. To approach more approximately real circumstances, four types of physical hosts are generated to simulate heterogeneous environment, as shown in Table 2.

Again, random strategy is applied under the condition of four-type physical machines to generate multiple physical host machines. Machines of the first type are equipped with ordinary and larger amount of parameters. Similarly, host machines have smaller amount when they are more highly equipped. Main random generation lists are given in Table 3.

Table 2: Parameters of physical host machine.

Type	CPU cores	Memory (G)	Power (w)
G1	2	4	220
G2	6	8	260
G3	8	14	300
G4	16	24	380

Table 3: Random generation lists of physical hosts.

Amount	Type-G1	Type-G2	Type-G3	Type-G4
800	350	200	150	100
1200	550	350	200	100
2000	900	550	300	250
3500	1600	900	600	400
5000	2300	1200	1000	500
7500	3500	2000	1200	800

Table 4: The description of simulation algorithms.

Indicator	Algorithm
Gr [9]	Comparing VM placement algorithms of on-demand cloud computing using greedy algorithm
Sa [10]	Resource allocation in cloud computing area using simulated annealing algorithm
Ga [13]	A hybrid genetic algorithm for the energy efficient virtual machine placement problem in data centers
MTAD	A multitarget heuristic algorithm for virtual machine placement
ISPMC	An iterative self-organizing physical machine classification algorithm
RBRC	A K-means virtual machine classification algorithm

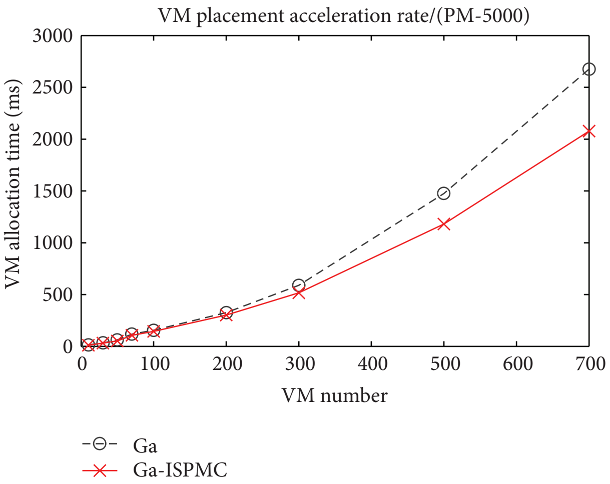


Figure 2: The acceleration rate on genetic algorithm (a).

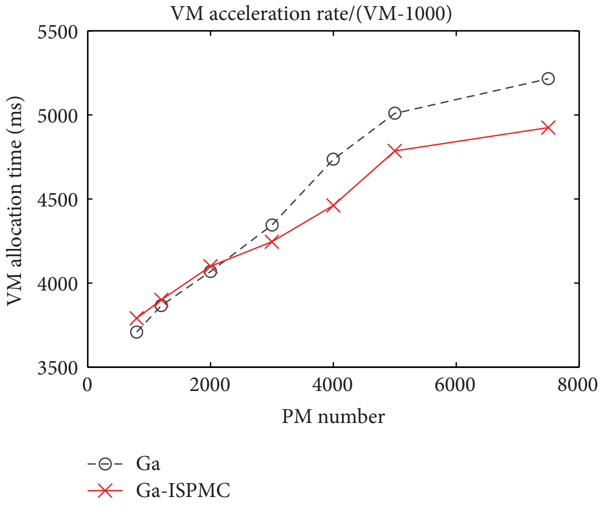


Figure 3: The acceleration rate on genetic algorithm (b).

5.1.2. Virtual Machine Placement Requests. Random strategy is used for the second time to generate the placement queue of VM requests based on the number of physical hosts generated so as to form VM queue that meets CloudSim and DataCenterBroker. In this study, random parameters of placement requests were chosen from 10 to approximately 3500, where CPU cores were generated randomly from 1 to 6 and memory from 1 $*$ 512 M to approximately 15 $*$ 512 M. Memory amount in each generation is equal to multiple integers of 512 M.

5.2. Simulation Result. On CloudSim, many demonstrations were given for ISPMC, MTAD, and RBRC algorithms. Experiments simulation and performance analysis were shown in Table 4, where placement efficiency, wastage rate, balance rate, and energy consumption were taken as the performance indexes. Table 4 depicts diagram of the six algorithms.

5.2.1. Simulation Results of ISPMC Algorithm. The main purpose of ISPMC algorithm is to classify the physical hosts and narrow the scanning dimension of physical host machines. On basis of genetic revolution placement, the experiment has compared the placement efficiency difference by using ISPMC and analyzed its performance. In Figures 2 and 3, it can be clearly known that ISPMC further accelerates the placement efficiency rate and improves placement performance. Figure 2 displays the acceleration condition of ISPMC genetic placement algorithm when the number of host machines is 5000 and placement requests are ranging from 10 to 700. With the number of virtual machine requests being increased, it is more obvious for ISPMC to improve the placement efficiency. In Figure 3, we fixed the number of virtual machine requests to verify the acceleration performance of ISPMC placement efficiency by changing the number of the physical hosts. As seen from Figure 3, when the numbers of physical hosts become larger, ISPMC can better accelerate the placement efficiency. Through the classification of ISPMC algorithm, we reduced the physical host dimension and shortened virtual machine placement time; however, when the numbers of physical hosts become smaller, the acceleration performance of ISPMC is a bit poorer than that of the genetic algorithm. Because physical hosts dimension is too small, if it is classified again, physical host dimension decreased slowly. Besides, the ISPMC algorithm itself needs

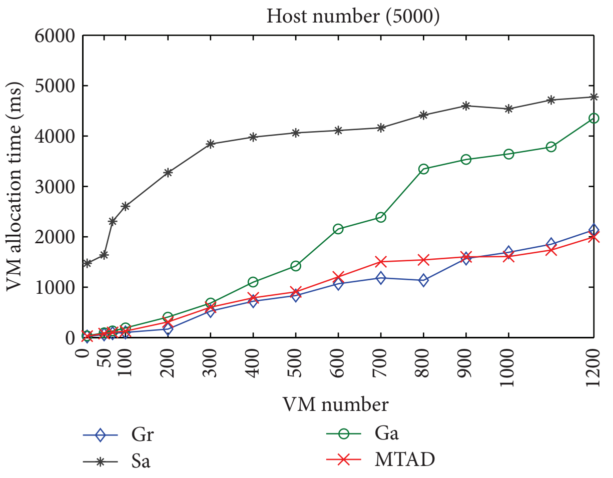


FIGURE 4: The running time.

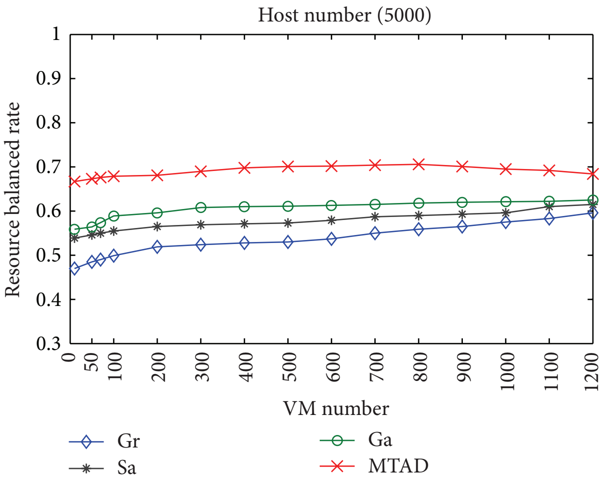


FIGURE 5: Resource balanced rate.

classification for some time. Therefore, when the physical host number is small, the acceleration performance of ISPMC algorithm is poor. In addition, the ISPMC algorithm is to classify the physical hosts so that virtual machine placement requests will relatively concentrate on mapping on the same type of physical host, which promotes regional balanced performance and provides convenience for conservation and management of energy consumption.

5.2.2. Simulation Results of MTAD Algorithm. The experiment verifies MTAD algorithm in many aspects. We verify performance between MTAD algorithm and several other algorithms including algorithm placement efficiency, wastage rate, resource balance rate, and energy consumption.

Figure 4 shows the placement time difference of various algorithms. Placement efficiency of MTAD algorithm is better than that of other algorithms. As shown in Figure 4, with the increase of the virtual machine requests, placement time increases gradually. Time efficiency of MTAD algorithm is between greedy algorithm and genetic algorithm; the simulated annealing algorithm is the worst, because the greedy algorithm only needs to randomly select the large resource in physical host set and has the faster time. In the physical hosts, MTAD algorithm is required to extract positive and negative solutions, which meet multiple objectives (properties), and then approximately chooses a suitable physical host. MTAD algorithm is slightly worse than the greedy algorithm over time aspect. When using genetic algorithm and simulated annealing, which need many time iterations in the initial solutions to obtain more optimal physical host, the longer iteration is, the longer time consumption is. In addition, with the number of virtual machine requests being increased, greedy algorithm needs to traverse longer scope, and time consumption becomes longer; with the number of virtual machine requests being increased, MTAD algorithm supported by ISPMC will exceed greedy algorithm.

The resource utilization rate of PM's different dimensions (balance rate) has been verified in Figure 5. Utilization balance rate refers to angle value between CPU and memory utilization and reflects difference degree on different dimensions of physical hosts. Balance rate is higher and the available extended resources of different dimensions are greater, which increase overall resource utilization of physical hosts. As seen from Figure 5, the resource utilization of MTAD algorithm is optimal; Ga algorithm takes second place, and greedy algorithm is the worst. When carrying out virtual machine placement, firstly, MTAD algorithm traverses physical hosts set to calculate the physical hosts set which meets virtual machine requests by using the approach method. Then, we get the approximate positive and negative solutions in the physical host set according to multiple targets restricted. Finally, we find the suitable physical hosts through the global approach degree of multiple targets. We fully consider virtual machine placement agent that affects PM's different dimensions resources balance rate when MTAD algorithm carries the virtual machine placement and choose smaller difference balance rate as assignment target and therefore improve the overall balance level of PM hosts. Ga intelligent placement algorithm uses self-organizing repeated iterative; it also considers local resource balancing degree. Greedy algorithm does not consider different dimensions resources balance rate, so its efficiency is the worst.

The resource wastage rate refers to wastage degree average value of different dimensions resources for virtual machines hosted by physical machine (in this paper, we only consider CPU and memory), and it is another measure index of the resource utilization rate. Figure 6 compares various algorithm differences of a PM's resource wastage rate. As known from Figure 6, with the VM requests being increased, physical host resource wastage rate gradually decreased and stabilized. Among the algorithms, resource wastage rate of MTAD algorithm is the lowest, Ga algorithm and Sa algorithm are approximate, and greedy algorithm is the worst. From Figure 6, it is not difficult to know that resource wastage rate of MTAD algorithm is optimal. Firstly, the MTAD considers the current balance rate approximation degree of virtual machine requests and physical host, the closer of the degree is, the more balancing of the physical host resource utilization will

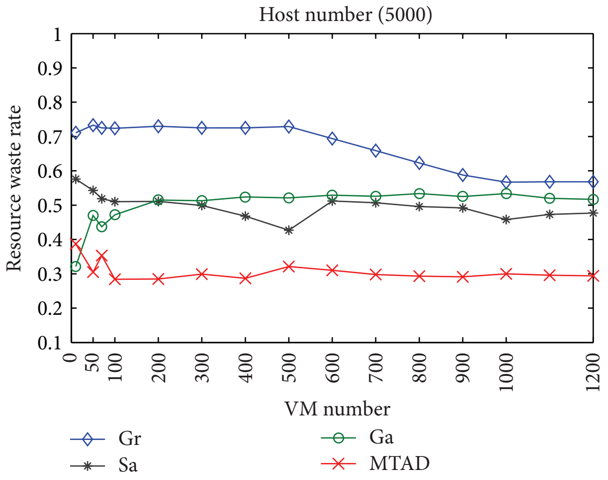


Figure 6: Resource waste rate.

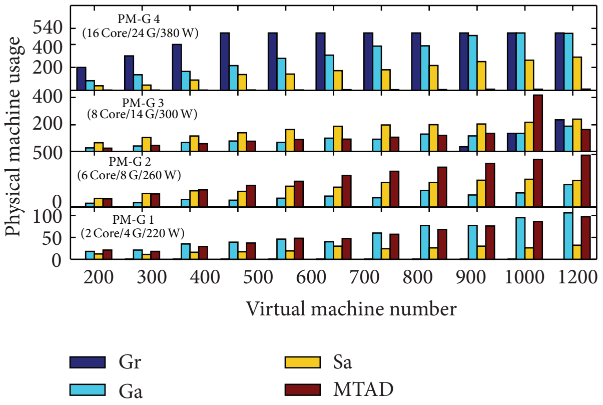


Figure 7: Physical machine usage.

be; the available degree of different dimensions resources becomes larger, and the PM's wastage rate will be smaller; secondly, use ISPMC algorithm to classify virtual machine placement; virtual machines will relatively concentrate on a classification physical host so as to enhance the physical host area utilization rate. However, through repeated iteration, Ga algorithm and Sa algorithm, which take the whole physical machine set as the placement area for improving the virtual machine placement efficiency, yet the overall resource rate is weaker than that of MTAD algorithm. However, the greedy algorithm does not consider any optimization for virtual machine placement, so the resource waste rate is the highest.

Figure 7 shows the energy consumption performance of different placement algorithms. Due to heterogeneous structure of data center physical hosts, it is not proper if we simply compare energy performance according to the physical hosts used number. We consider the used number of different type of physical hosts under the condition that the size of physical hosts is fixed; different virtual machine placement requests are carried out so as to statistical the quantity of different type physical hosts. From Figure 7, we can see that four algorithms placement 200~1200 virtual machine requests in 5000 physical hosts. The used number of four different physical hosts can be seen from left to right as follows.

(1) The first class physical host: with the number of virtual machine placement requests being increased, the used number of physical hosts of GA algorithm and MTAD algorithm is relatively larger; Ga algorithm is slightly higher than MTAD algorithm. Compared with two previous algorithms, the used number of Sa algorithm is smaller and the greedy algorithm does not have any physical host. (2) The second class physical host: a large number of the second class physical hosts are used by MTAD algorithm, followed by Sa algorithm, Ga algorithm, and the GR algorithm. The GR algorithm also does not use any second host. (3) The third class host: among the four kinds of algorithms, Sa algorithm and MTAD algorithm used more physical hosts; Sa algorithm and Gr algorithm are less. (4) The fourth class host: with the number of virtual machine requests being increased, greedy algorithm used almost all of the fourth class physical hosts; Ga algorithm is the second, and Sa algorithm is the medium. MTAD algorithm almost does not use any fourth class physical host. From Figure 7, we summarize that when MTAD algorithm carries the virtual machine requests, it tends to use lower energy consumption (low configuration) physical hosts. Sa algorithm and Ga algorithm are the random balancing placement, while greedy algorithm is likely to use the high energy consumption physical host (high configuration).

5.2.3. Simulation Results of RBRC Algorithm. With the continuous development of the cloud computing model, the number of virtual machine placement will be gradually expanded; the current serial placement will inevitably become a bottleneck of virtual machine placement algorithm. We present RBRC algorithm that aims at solving the above problem; the main idea is to use classification thought to classify resource demand similar to virtual machine based on large-scale virtual machine requests and achieve the concurrent placement way. We take the demand balance degree of different dimensions resources as core classification idea. According to ISPMC algorithm, physical hosts will be divided into K class by using K-means classification. Figure 8 is an assigning time efficiency acceleration diagram of RBRC algorithm.

As shown in Figure 8, with the number of virtual machines being increased, RBRC algorithm can greatly improve the placement efficiency of MTAD algorithm. Because of RBRC classification algorithm concurrent placement virtual machine requests, the algorithm greatly reduces the whole placement time and improves the performance. However, when the quantity of virtual machine placement is few, the classification degree of RBRC is relatively small; the used time is almost the same and does not change too much.

6. Conclusion

We put forward three optimal algorithms for virtual machine placement by analyzing the related work and defects of virtual machine placement algorithms in cloud data center. (1) ISPMC is a physical hosts classification algorithm. Taking the heterogeneous nature of the physical host constraints, we classify the physical hosts by clustering so as to reduce the dimensions of the physical hosts and optimize the placement

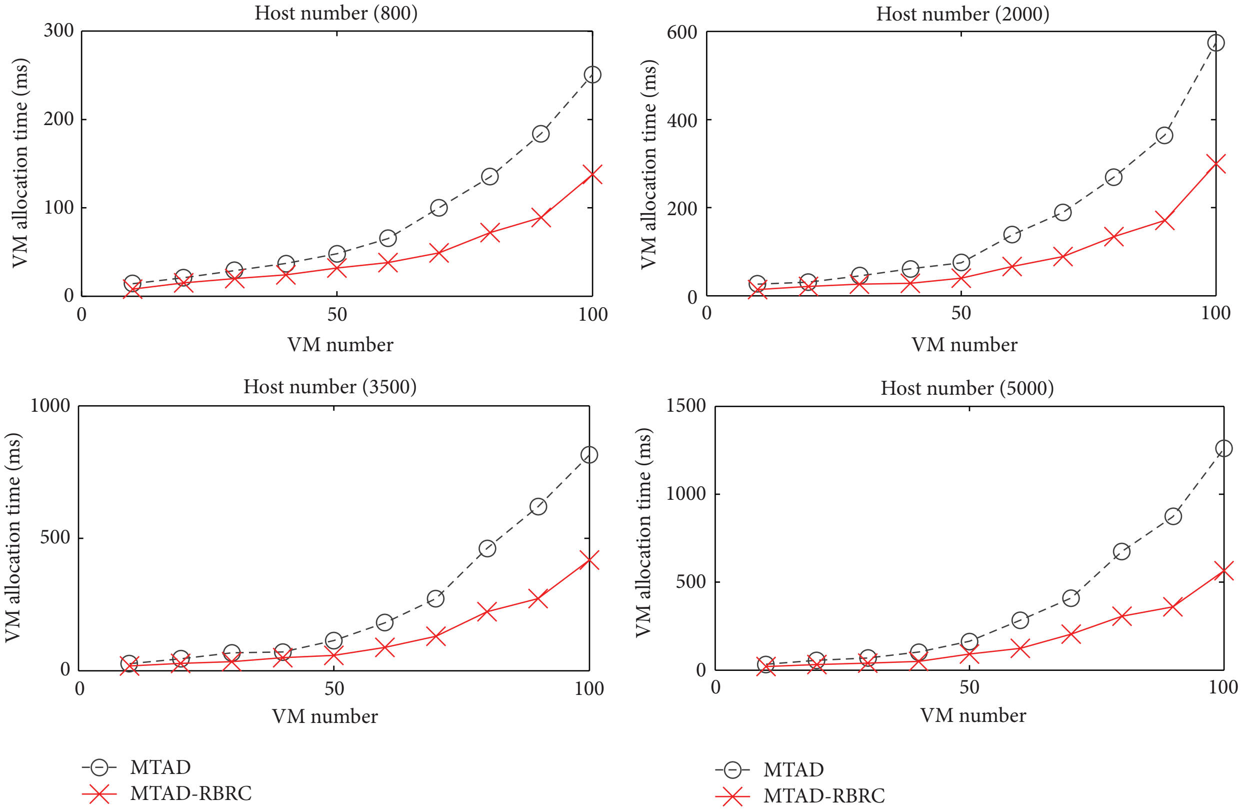


Figure 8: Placement time efficiency of RBRC.

efficiency of virtual machine requests. (2) MTAD is a multi-target heuristic virtual machine placement algorithm. MTAD considers virtual machine placement problems, such as "different dimensions resources imbalanced," "high wastage rate," and "high energy consumption." We use multiobjective optimization theory, traverse the set of physical hosts, and get the multiple objectives positive and negative ideal solutions. Thus, we can obtain a reasonable placement solution by comparison of approach degree between multiple objectives and positive and negative ideal solutions. MTAD algorithm enhances the balance rate of different PM's dimensions resources and the overall resources utilization; it is effective to request heterogeneous physical hosts with lower energy consumption. (3) There is a VM classification algorithm RBRC based on K-means. We divided the virtual machines into several similar demand groups and used concurrent placement model to achieve the goal of rapidly allocating virtual machines. The main target of RBRC algorithm is to solve low efficiency problem of large-scale virtual machine placement in serial way.

All three kinds of optimization algorithms focus on the physical machine hosts and have no considering of the network factors and other special applications (QoS quality, etc.), which will be the future work.

Notations

V: Random set of virtual machine requests
P: Set of data center physical servers
P_{used}: Set of virtual machines hosted by physical servers
V_i^{cpu}: CPU demand for virtual machine i, $v_i \in V$
V_i^{mem}: Memory demand for virtual machine i, $v_i \in V$
p_j^{cpu}: The total capacity of the physical host CPU of j, $p_j \in P$
p_j^{mem}: The total capacity of the physical host memory of j, $p_j \in P$
p_j^{energy}: Power consumption per unit time of the physical host (W)
$p_j^{T\text{cpu}}$: The total used number of physical host CPU of j, $p_j \in P$
$p_j^{T\text{mem}}$: The total used number of physical host memory of j, $p_j \in P$
V_{pj}: Set of virtual machines hosted by physical host p_j, $p_j \in P$
$p_j^{v\text{count}}$: The number of virtual machines on a physical server j hosted, $p_j \in P$.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

This work was supported by the National Key Technology Support Program of China Grant no. 2012BAH09B02 and the Key Science and Technology Project of Changsha City Grant no. K1204006-11-1.

References

[1] M. F. Bari, R. Boutaba, R. Esteves et al., "Data center network virtualization: a survey," *IEEE Communications Surveys and Tutorials*, vol. 15, no. 2, pp. 909–928, 2013.

[2] T. Dillon, C. Wu, and E. Chang, "Cloud computing: issues and challenges," in *Proceedings of the 24th IEEE International Conference on Advanced Information Networking and Applications (AINA '10)*, pp. 27–33, Perth, Australia, April 2010.

[3] R. Ranjana and J. Raja, "A survey on power aware virtual machine placement strategies in a cloud data center," in *Proceedings of the IEEE International Conference on Green Computing, Communication and Conservation of Energy (ICGCE '13)*, pp. 747–752, 2013.

[4] E. Vintrou, D. Ienco, A. Begue, and M. Teisseire, "Data mining, a promising tool for large-area cropland mapping," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 6, no. 5, pp. 2132–2138, 2013.

[5] N. Bobroff, A. Kochut, and K. Beaty, "Dynamic placement of virtual machines for managing SLA violations," in *Proceedings of the 10th IFIP/IEEE International Symposium on Integrated Network Management (IM '07)*, pp. 119–128, Munich, Germany, May 2007.

[6] S. Chaisiri, B.-S. Lee, and D. Niyato, "Optimal virtual machine placement across multiple cloud providers," in *Proceeding sof the IEEE Asia-Pacific Services Computing Conference (APSCC '09)*, pp. 103–110, IEEE, December 2009.

[7] H. N. Van, F. D. Tran, and J.-M. Menaud, "Autonomic virtual resource management for service hosting platforms," in *Proceedings of the ICSE Workshop on Software Engineering Challenges of Cloud Computing (CLOUD '09)*, pp. 1–8, IEEE Computer Society, May 2009.

[8] X. Meng, V. Pappas, and L. Zhang, "Improving the scalability of data center networks with traffic-aware virtual machine placement," in *Proceedings of the IEEE Conference on Computer Communications (IEEE INFOCOM '10)*, pp. 1–9, San Diego, Calif, USA, March 2010.

[9] K. Mills, J. Filliben, and C. Dabrowski, "Comparing VM-placement algorithms for on-demand clouds," in *Proceedings of the 3rd IEEE International Conference on Cloud Computing Technology and Science (CloudCom '11)*, pp. 91–98, Athens, Ga, USA, December 2011.

[10] D. Pandit, S. Chattopadhyay, M. Chattopadhyay, and N. Chaki, "Resource allocation in cloud using simulated annealing," in *Proceedings of the Applications and Innovations in Mobile Computing (AIMoC '14)*, pp. 21–27, Kolkata, India, March 2014.

[11] H. Nakada, T. Hirofuchi, H. Ogawa et al., "Toward virtual machine packing optimization based on genetic algorithm," in *Distributed Computing, Artificial Intelligence, Bioinformatics, Soft Computing, and Ambient Assisted Living*, pp. 651–654, Springer, Berlin, Germany, 2009.

[12] S. Agrawal, S. K. Bose, and S. Sundarrajan, "Grouping genetic algorithm for solving the server consolidation problem with conflicts," in *Proceedings of the 1st ACM/SIGEVO Summit on Genetic and Evolutionary Computation (GEC '09)*, pp. 1–8, June 2009.

[13] M. Tang and S. Pan, "A hybrid genetic algorithm for the energy-efficient virtual machine placement problem in data centers," *Neural Processing Letters*, 2014.

[14] M. Sun, W. Gu, X. Zhang, H. Shi, and W. Zhang, "A matrix transformation algorithm for virtual machine placement in cloud," in *Proceedings of the 12th IEEE International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom '13)*, pp. 1778–1783, July 2013.

[15] S. Wang, H. Gu, and G. Wu, "A new approach to multi-objective virtual machine placement in virtualized data center," in *Proceedings of the 8th IEEE International Conference on Networking, Architecture and Storage (NAS '13)*, pp. 331–335, July 2013.

[16] L. Lu, H. Zhang, E. Smirni, G. Jiang, and K. Yoshihira, "Predictive VM consolidation on multiple resources: beyond load balancing," in *Proceedings of the IEEE/ACM 21st International Symposium on Quality of Service (IWQoS '13)*, pp. 83–92, Montreal, Canada, June 2013.

[17] B. Wadhwa and A. Verma, "Energy saving approaches for green cloud computing: a review," in *Proceedings of the Recent Advances in Engineering and Computational Sciences (RAECS '14)*, pp. 1–6, Chandigarh, India, March 2014.

[18] L. Shi, J. Furlong, and R. Wang, "Empirical evaluation of vector bin packing algorithms for energy efficient data centers," in *Proceedings of the IEEE Symposium on Computers and Communications (ISCC '13)*, Split, Croatia, July 2013.

[19] Y. Wu, M. Tang, and W. Fraser, "A simulated annealing algorithm for energy efficient virtual machine placement," in *Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics (SMC '12)*, pp. 1245–1250, Seoul, South Korea, October 2012.

[20] A. Dhingra and S. Paul, "Green cloud: heuristic based BFO technique to optimize resource allocation," *Indian Journal of Science and Technology*, vol. 7, no. 5, pp. 685–691, 2014.

[21] F. Song, D. Huang, H. Zhou, H. Zhang, and I. You, "An Optimization-Based Scheme for Efficient Virtual Machine Placement," *International Journal of Parallel Programming*, vol. 42, no. 5, pp. 853–872, 2014.

[22] R. N. Calheiros, R. Ranjan, A. Beloglazov, A. F. C. de Rose, and R. Buyya, "CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms," *Software—Practice & Experience*, vol. 41, no. 1, pp. 23–50, 2011.

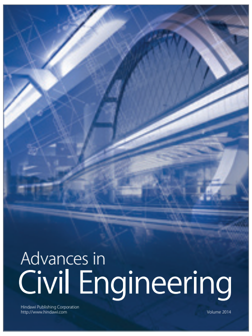
Journal of
Engineering

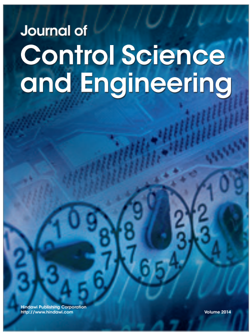
The Scientific World Journal

International Journal of
Rotating Machinery

Journal of
Sensors

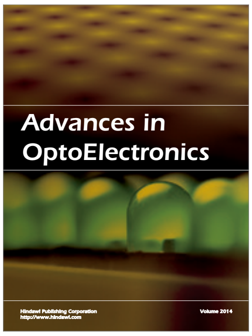
International Journal of
Distributed Sensor Networks


Advances in
Civil Engineering


Journal of
Control Science and Engineering

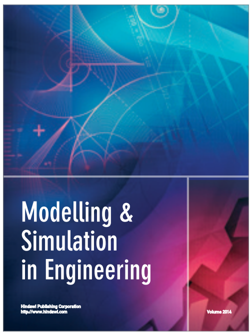
Journal of
Robotics

Journal of
Electrical and Computer Engineering

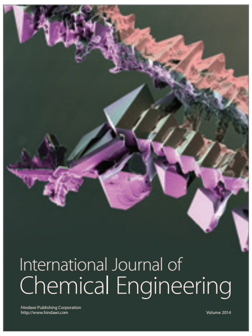

Advances in
OptoElectronics

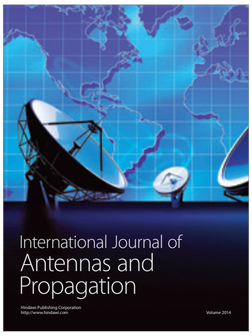
VLSI Design

International Journal of
Navigation and Observation

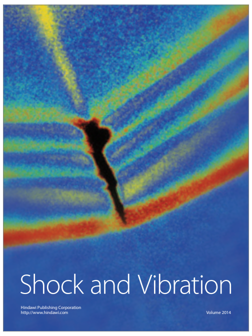

Modelling & Simulation in Engineering

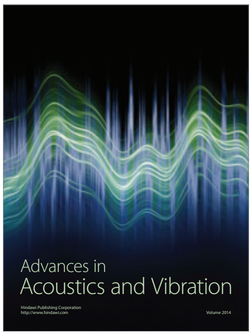
International Journal of
Aerospace Engineering


International Journal of
Chemical Engineering


International Journal of
Antennas and Propagation

Active and Passive Electronic Components


Shock and Vibration


Advances in
Acoustics and Vibration