Computing and Informatics, Vol. 39, 2020, 83–104, doi: 10.31577/cai_2020_1-2_83

VIRTUAL MACHINE DEPLOYMENT STRATEGY BASED ON IMPROVED PSO IN CLOUD COMPUTING

Shanchen Pang, Dekun Dong, Shuyu Wang

College of Computer Science and Technology
China University of Petroleum
Qingdao, 266580, China
e-mail: pangsc@upc.edu.cn, z18070068@s.upc.edu.cn

Abstract. Energy consumption is an important cost driven by growth of computing power, thereby energy conservation has become one of the major problems faced by cloud system. How to maximize the utilization of physical machines, reduce the number of virtual machine migrations, and maintain load balance under the constraints of physical machine resource thresholds that is the effective way to implement energy saving in data center. In the paper, we propose a multi-objective physical model for virtual machine deployment. Then the improved multi-objective particle swarm optimization (TPSO) is applied to virtual machine deployment. Compared to other algorithms, the algorithm has better ergodicity into the initial stage, improves the optimization precision and optimization efficiency of the particle swarm. The experimental results based on CloudSim simulation platform show that the algorithm is effective at improving physical machine resource utilization, reducing resource waste, and improving system load balance.

Keywords: Cloud computing, Pareto optimal solution, particle swarm optimization algorithm, resource reservation, virtual machine deployment

Mathematics Subject Classification 2010: 68-W99, 68-M01

1 INTRODUCTION

Cloud computing is a distributed computing model that provides available, convenient and on-demand network access to shared resource pools (such as facilities, applications, storage devices, etc.). Resources as a service is a primary form of cloud

computing, mainly divided into infrastructure, platform and software. Through cloud computing, consumers can use or uninstall resources anytime, anywhere, which improves service quality and reduces operation costs.

The cloud data center uses the virtualization technology to construct computing resources, storage resources and network resources into a dynamic virtual resource pool. It uses virtual resource management technology to realize automatic deployment, dynamic expansion and on-demand allocation of cloud computing resources. Generalized virtualization includes virtual memory, virtual machines, storage virtualization, etc. We focus on virtualization of physical machines in this paper. As the number of cloud users increases, more virtual machine requests are added, what increases the pressure on physical machines. Thus cloud data center requires an effective virtual machine deployment strategy. The deployment problem is proved to be a non-deterministic polynomial problem, which meant we cannot find a precise solution. But with the help of some intelligent algorithms, some sub-optimal solutions could be found [26].

Kennedy and Eberhart developed Particle Swarm Optimization (PSO), which is considered a new swarm intelligence and evolutionary algorithm [7]. It has the advantages of fast search speed, simple implementation, high efficiency and so on, which has attracted the attention of the academic community. It shows its ascendancy in solving practical problems. However, swarm intelligence algorithms, such as ant colony algorithm, differential evolutionary algorithm, genetic algorithm have the problem of local optimization.

Existing virtual machine deployment strategies only consider resource utilization or virtual machine migration, and ignore the impact of load balancing on system performance. In the paper, we proposed an improved multi-objective particle swarm optimization algorithm (TPSO). The algorithm introduces chaotic strategy into the initial stage of the iterative process, thus the particle swarm distribution has better ergodicity and uniformity. In the medium term, random grouping strategy is used as the core process of the algorithm. Clustering is applied later, which increases the fineness of the end of the search and ensures convergence normally. Multi-objective selection in the updating process of particle swarm algorithm is a representative problem. Most of them add weight to the objective function and convert it into a single objective function. We set three objective functions, and seek the optimal solution of the algorithm by Technique for Order Preference by Similarity to an Ideal Solution method (TOPSIS). To improve physical machine resource utilization and save cloud system energy consumption, we apply the improved algorithm to virtual machine deployment, take physical machine resource utilization, virtual machine migration and load balance as the objective functions.

The rest of the paper is organized as follows. Section 2 discusses related work. Section 3 gives the resource waste model. Section 4 gives the improved algorithm design. The experiments and results are discussed in Section 5, and the Section 6 summarizes our work.

2 RELATED WORK

In recent years, PSO has become a focus of researchers due to its simple structure, few parameters and easy implementation of code, especially in image recognition, path planning and artificial intelligence. The theoretical research of this algorithm is mainly divided into two categories. The first category focuses on the topological structure, parameter optimization and population diversity of the algorithm. The second category mainly includes the combination of the algorithm and other intelligent algorithms.

Particle swarm optimization has performed advantages in solving practical problems and has a great development value and development space. However, due to the quick loss of diversity, it tends to fall into local optima. In order to enhance the performance of particle swarm, Wang et al. [23] proposed DNSPSO based on the enhanced diversity and neighborhood search, but the neighborhood radius may affect the effectiveness of the neighborhood search. For the traditional linear learning strategy, Zhao et al. [31] proposed a new position storage mechanism. The algorithm efficiently and quickly seeks high-quality solutions. To avoid premature convergence of the algorithm, Wang et al. [24] presented GOPSO by speeding up convergence and escaping local optimization, but generalized opposition-based learning performs badly on shifted and large scale problems. Cho et al. [1] presented a novel multimodal optimization algorithm, which used deterministic sampling to produce new particles during the optimization process. This paper did not address the case in which there are more local optima than could possibly be detected. In PSO, different inertia weight strategies can influence the performance of algorithm. Nickabadi et al. [13] summarized various inertial weight strategies and proposed a new adaptive inertia weight strategy based on the success rate of the particles. In order to improve the search efficiency in the complex problem spaces, Zhan et al. [29] proposed an orthogonal learning method (OL). Through orthogonal experimental design, the particle swarm can discover more useful information that exists in the best experience of history and the best experience of neighbors. Based on perturbing global best strategy, Zhao et al. [33] proposed an modified discrete immune optimization algorithm. Zhao et al. [32] guided the search strategy with multigroup information communication and sharing mechanism and multi-stage global disturbance in their paper, they considered the population diversity and selection pressure simultaneously. In the second category, classic intelligent algorithms include ant colony algorithm, genetic algorithm, and differential evolutionary algorithm [3, 4, 21, 16].

Virtual machine deployment is the process of assigning virtual machines to physical machines by the allocation strategy. Since the cloud infrastructure is completely virtual, the deployment of virtual machines becomes a core issue for cloud systems. To this end, a lot of research has been done on optimizing the deployment of virtual machines. Works [8, 28, 30, 2] focused on virtual machine deployment through particle swarm optimization, most of them focused on one or two optimization goals. Wilcox et al. [27] and Nguyen et al. [17] mainly took into account two goals based on

the genetic algorithm. Maurer et al. [12] proposed a VM deployment algorithm that used heuristic packing algorithms and forecasting techniques, they achieved the minimum number of physical machines and ensured a certain amount of service-level agreement. For reducing the unnecessary migrations, Sato et al. [18] used Auto Regressive Model to predict virtual machine usage, but it must be evaluated on the servers that the VMs are deployed on. For some performance bottlenecks with MapReduce, Shabeera and Madhu Kumar [20] and Li et al. [9] optimized MapReduce performance under the specified constraints. Wang et al. [25] balanced power consumption and quality of service by controlling the number of servers and virtual machine time sharing. Based on considering VM placement and data placement simultaneously, Shabeera et al. [19] proposed a meta-heuristic algorithm using ant colony algorithm. Song et al. [22] saved data center energy by effectively utilizing resource fragments, but they inevitably increased the number of VM migrations to further categorize the different intervals. Pang et al. [15] proposed a task-oriented resource allocation method based on ACO, it can reduce the power consumption of data center effectively on the premise of performance guarantee. Ma et al. [11] optimized time and power based on Genetic Algorithm, but the encoding process is complicated and the search speed is slow. Through estimation of distribution algorithm and genetic algorithm, Pang et al. [14] developed an EDA-GA hybrid scheduling algorithm, however, this paper does not consider the dynamics and uncertainty of the cloud computing environment. For example, the computing speed of virtual machines changes in real time.

3 RESOURCE WASTE MODEL

3.1 Problem Description

In the paper, we proposed a virtual machine deployment scheme based on particle swarm optimization. The scheme reduces the energy consumption of the cloud system by rationally deploying virtual machines. In the cloud system, virtual machines are divided into the deployed virtual machines and the newly requested virtual machines. The VM deployment framework is shown in Figure 1.

The virtual machine deployment solution proposed in this paper monitors the resource state of physical machines, including CPU, memory, and network bandwidth. Suppose we set n physical machines in the cloud environment. The threshold of the physical machine (PM) is represented by the triplet as $R_j = \{R_j^C, R_j^M, R_j^B\}$, where R_j^C represents the CPU threshold, R_j^M represents the memory threshold, R_j^B represents the bandwidth threshold. There are m virtual machines. Virtual machine (VM) requirements are represented by the triplet as $V_j = \{V_j^C, V_j^M, V_j^B\}$, where V_j^C represents the CPU requirement, V_j^M represents the memory requirement, V_j^B represents the bandwidth requirement. $m = m_r + m_s$, m_r and m_s indicate the number of newly requested virtual machines and deployed virtual machines, respectively. The matrix represents the deployment of the virtual machine. Each term in the matrix x_{ij} is 0 or 1. If $x_{ij} = 1$, representing the i^{th} VM is deployed on the j^{th} PM.

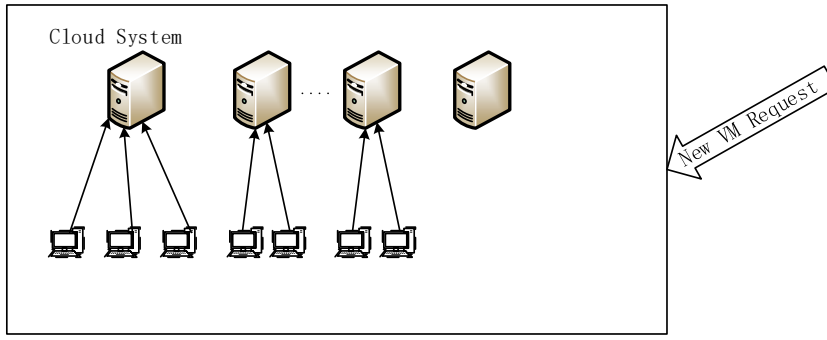

Figure 1. Deployment framework for VMs

If $x_{ij} = 0$, it is not deployed there. The matrix is expressed as follows:

$$\begin{bmatrix} x_{11} & x_{12} \dots & x_{1n} \\ x_{21} & x_{22} \dots & x_{2n} \\ \dots & & \\ x_{m1} & x_{m2} \dots & x_{mn} \end{bmatrix}. \tag{1}$$

3.2 Objective Functions and Constraint Functions

The objective functions are as follows:

$$\text{Max} f_1 = \left(\frac{\sum_{i=1}^{m} V_i^C}{\sum_{j=1}^{n} R_j^C} + \frac{\sum_{i=1}^{m} V_i^M}{\sum_{j=1}^{n} R_j^M} + \frac{\sum_{i=1}^{m} V_i^B}{\sum_{j=1}^{n} R_j^B} \right) /3, \tag{2}$$

$$\text{Min} f_2 = \sum_{i=1}^{m} m_i, \tag{3}$$

$$\text{Min} f_3 = \sqrt{\left(\sum_{j=1}^{n} (C_{juse} - C_{use})^2 + \sum_{j=1}^{n} (M_{juse} - M_{use})^2 + \sum_{j=1}^{n} (B_{juse} - B_{use})^2 \right) /n}. \tag{4}$$

The constraint functions are as follows:

$$\sum_{j=1}^{n} x_{ij} = 1, \tag{5}$$

$$\sum_{i=1}^{m} V_j^C \times x_{ij} < R_j^C, \tag{6}$$

$$\sum_{i=1}^{m} V_j^M \times x_{ij} < R_j^M, \tag{7}$$

$$\sum_{i=1}^{m} V_j^B \times x_{ij} < R_j^B. \tag{8}$$

Equation (2) represents virtual machine utilization. Equation (3) represents the number of migrations of the virtual machine. Equation (4) represents the load imbalance, where C_{use}, M_{use}, and B_{use} represent the average utilization of the system CPU, memory, and bandwidth, respectively. Correspondingly, C_{juse}, M_{juse} and B_{juse} respectively represent the CPU, memory and bandwidth utilization of the j^{th} physical machine. Equation (5) means that the same VM can only be deployed on one PM. Equations (6), (7), (8) indicate that the resource requirements (CPU, memory and bandwidth) of virtual machines on the same physical machine do not exceed its corresponding threshold.

3.3 Chaos Strategy

Chaos is a motion system synthesized when both the deterministic and random components of the system are clearly present, and it is a contradictory unity that exists objectively. Chaos optimization algorithm is a search algorithm that transforms variables from chaotic space to solution space. The algorithm has the advantages of global asymptotic convergence, easy to escape the local optimum, and fast convergence speed. To improve the ergodicity and uniformity, we use the chaos strategy to initialize the particle swarm.

3.4 Pareto Optimal Solution

Pareto optimality is an ideal state in the process of resource allocation. In some multi-objective models, Pareto optimality does not exist. We can only weigh each target to choose a solution from the Pareto front. In the data center, when we increase the utilization of physical machines, migrating virtual machine is inevitable, however, excessive virtual machine migration is bound to increased energy consumption. At the same time, load balance also interacts with them, which is obviously an optimal problem in Pareto. We use the TOPSIS method to optimize this problem in virtual machine dynamic deployment.

4 IMPROVED ALGORITHM DESIGN

4.1 Classic Particle Swarm Optimization

PSO belongs to one of swarm intelligence and evolutionary algorithms. The main idea of particle swarm optimization is that through multiple iterations, particles use their own experience and the experience of the group to gradually find the best solution to the problem. Unlike genetic algorithms, particle swarms do not use selection. Normally, all members of the group can survive from the beginning of the experiment to the end. Through multiple iterations and interactions between individuals, the optimal solution to the problem is obtained [5].

In each iteration, the best experience of each particle in the search space is called the individual extreme value pBest. The best experience of the population is the current global optimal solution of the entire particle swarm, called the global extremum gBest. All particles adjust themselves through individual optimal values and global optimal values. The formula is as follows:

$$V_{id}(t+1) = V_{id}(t) + r_1 \times c_1 \times (P_{id}(t) - X_{id}(t)) + r_2 \times c_2 \times (P_{gd}(t) - X_{id}(t)), \quad (9)$$

$$X_{id}(t+1) = V_{id}(t) + X_{id}(t). \quad (10)$$

The formula (9) is the particle velocity update formula, which contains three parts. The first part is the speed of the t^{th} generation, called the particle inertia speed part. The second part updates the speed through the individual optimal value, which is the individual cognitive part. The third part updates the speed through the group optimal value, which is the social cognitive part. Formula (10) is the particle position update formula.

4.2 Improved Multi-Objective Particle Swarm Optimization Algorithm Design

Kennedy claimed that particle swarms with large neighborhoods achieve better at solving simple problems, while particle swarms with small neighborhoods may implement better at solving complex problems [6]. In the paper, we improved the traditional PSO. Firstly, chaotic algorithm initializes the particle swarm. The first ninety percent of the iterative process uses a random grouping strategy, and then the last ten percent uses clustering convergence.

4.3 Chaos Mapping

By using the sensitivity and ergodicity of the initial value of chaotic mapping, a particle is randomly initialized, and then the initial value of multiple particles is obtained by chaotic mapping. Chaos mapping is used to expand the initial particle swarm, changes the extraction process of initial particle swarm. Tent mapping has simple structure and good traversal property. The mapping steps are as follows.

Tent mapping:

$$x_{n+1} = \begin{cases} 2x_n, & 0 \le x_n \le 0.5, \\ 2(1 - x_n), & 0.5 < x_n \le 1. \end{cases} \tag{11}$$

Step 1. Randomly initialize particle i, the speed is $V_{id} = (v_{i1}, v_{i2}, \ldots, v_{id})$ and the position is $X_{id} = (x_{i1}, x_{i2}, \ldots, x_{id})$.

Step 2. The particle i is iterated b times respectively according to Equation (11), that is, b initial particles' velocity vectors and position vectors are obtained.

Step 3. The b particles in Step 2 are still iterated c times according to Equation (11) to get the initial particle swarm p, $p = b \times c$, and get their velocity vectors and position vectors.

4.4 Inertia Weight

The value of the inertia weight affects the convergence speed and convergence accuracy. When w is large, the particle swarm tends to be globally optimized. On the contrast, if w is small, particle swarms tend to be locally optimized. According to the influence of w value on the search results, we use the linear decrement method to set the w value in this paper. As shown in the following formula:

$$w = w_{\max} - (w_{\max} - w_{\min}) \times \sqrt{t/\text{total\_}t} \tag{12}$$

where $w_{\max}$ is 0.95, $w_{\min}$ is 0.05, $total\_t$ represents the total number of iteration, and t represents the current number of iteration. The particle swarm search area is large in the early period. In the later period, the particle swarm switched from global search to local search as w slowly decreases, and the algorithm does not converge too fast.

4.5 Random Grouping and Clustering Convergence

Two particles are not sufficient to construct a good swarm, while five particles show better local search ability and reduce global search ability. Three particles achieve a balance between them, the swarms show better global search ability [10]. The iterative process of the particle swarm algorithm is first updated by a random grouping strategy. Three particles form a small group, and the particles are randomly reorganized every 5 generations. The steps for random grouping are shown in Algorithm 1.

In the later stage of the improved particle swarm optimization. The k-means clustering algorithm based on Euclidean distance is used to converge the particle swarms. The k values are successively reduced to 1, that is, they are merged into one particle swarm. After each cluster center is stabilized, the particle swarm is updated within each group. The cluster convergence steps are shown in Algorithm 2.

Algorithm 1 Random grouping iteration process

Require: m: Small group members; n: Small groups' number; D: Regrouping period; $total\_t$: The total number of iteration;
Ensure: swarm p

```
initial m = 3 and D = 5;
Initialize p = 3 × n particles by chaos strategy;
Grouping the population randomly;
for i = 1;0.9 × total_t do
    Update each small group by formula (9), (10);
    if mod(i, D) == 0 then
        Regroup the small group randomly;
    end if
end for
```

Algorithm 2 Cluster convergence iteration process

Require: e: Each swarm's population; D: Regrouping period;
Ensure: optimal particle

```
initial e = 6 and D = 5;
for i = 0.9 × total_t; total_t || k = 1;e = e + 6 do
    Select k = p/e particles as the initial centroid ;
    repeat
        Assign each point to the nearest centroid;
        Form k clusters;
        Recalculate the centroids of each cluster;
    until The center of mass does not change;
    Use the information in the group to update the velocity and position;
    if mod(i, D) == 0 then
        Clustering regroup the swarms randomly
    end if
end for
```

4.6 Pareto Optimal Solution Design

The TOPSIS method is an effective multi-index evaluation method. The basic principle is to sort by calculating the relative distance between each object and the best and worst solutions. In this paper, it is applied to particle selection with multi-objective optimal value in TPSO algorithm. The strategy is as follows.

The decision matrix is as follows:

$$v = \begin{bmatrix} v_{11}v_{12} & v_{13} \\ v_{21}v_{22} & v_{23} \\ \cdots & \\ v_{p1}v_{p2} & v_{p3} \end{bmatrix} \tag{13}$$

where v_{ij} represents the function value of the deployment strategy of i^{th} particle in the j^{th} objective function.

Step 1. Process the objective function to the same trend.

$$f_1^{'} = 1/f_1. \tag{14}$$

Step 2. Vector normalization normalizes the decision matrix V.

$$X_{ij} = v_{ij} / \sqrt{\sum_{i=1}^{p} v_{ij}^2} \quad (i = 1, 2, \ldots, p; j = 1, 2, 3). \tag{15}$$

Step 3. Determine the ideal and the negative ideal solution.

Ideal solution:

$$S^+ = \left\{ \left(\min_{1 \le i \le p} x_{ij} \right) \mid i = 1, 2, \ldots, p \right\} = \left\{ x_1^+, x_2^+, x_3^+ \right\}. \tag{16}$$

Negative ideal solution:

$$S^- = \left\{ \left(\max_{1 \le i \le p} x_{ij} \right) \mid i = 1, 2, \ldots, p \right\} = \left\{ x_1^-, x_2^-, x_3^- \right\}. \tag{17}$$

Step 4. Determine the distance of each solution to the actual solution.

$$S_i^+ = \sqrt{\sum_{j=1}^{3} \left(x_{ij} - x_i^+ \right)^2}, \tag{18}$$

$$S_i^- = \sqrt{\sum_{j=1}^{3} \left(x_{ij} - x_i^- \right)^2}. \tag{19}$$

Step 5. Calculate the relative closeness of each solution to the ideal solution.

$$C_i^* = S_i^- / (S_i^+ + S_i^-). \tag{20}$$

Step 6. Prioritize the scheme according to the size of C_i^*. The one with the highest relative closeness is the selected particle.

4.7 Virtual Machine Coding

The m virtual machines to be deployed are first coded from 1 to m. Then we obtain the correspondence between the virtual machines and the physical machines

through the TPSO algorithm. Finally, the virtual machines are deployed on the corresponding physical machines according to the mapping relationship. Thereby we achieve the optimization goal. For example, $X_{id} = (1, 5, 4, 2, 3, 6 \ldots)$ means: Virtual machines No. 1, 5, 4, 2, 3, and 6 are deployed on physical machines No. 1, 2, 3, 4, 5, and 6, respectively, and so on. The value of the particle position vector represents a deployment strategy of the virtual machine. The deployment scenarios for a VM resource is shown in Figure 2.

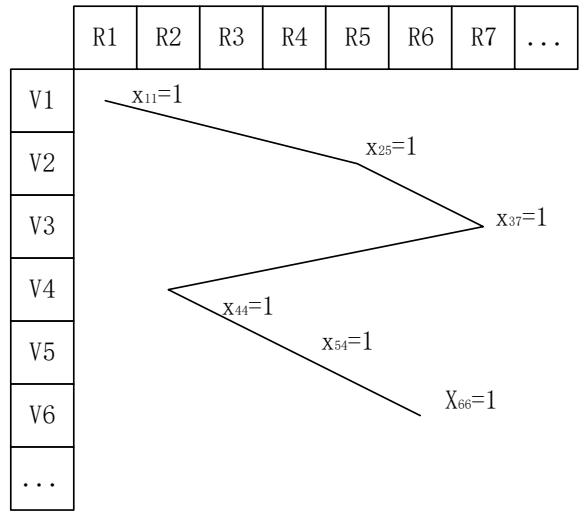

Figure 2. Deployment scenarios for a VM resource

4.8 Improved Algorithm Steps

Step 1. Set the number of particles, iterations, the learning factor, the inertia weight, the velocity threshold, and the position threshold.

Step 2. Initialize to generate p particles, and the chaotic mapping initializes the initial position and velocity of the particles in the population.

Step 3. Randomly group the particles. The adaptive values of the particles in each population are calculated in turn according to the objective function Formulas (2), (3), (4) under constraint conditions (5), (6), (7), (8).

Step 4. Record the individual extremum of every particle in each small group, calculate and record the global extremum of each small group by TOPSIS method.

Step 5. Update the particle's velocity with the individual extremum and the global extremum obtained by Step 4 and Equation (9). Update the particle's location by Equation (10).

Step 6. Regroup the swarms randomly every 5^{th} generation.

Step 7. Determine if the number of iterations reaches $0.9 \times total\_t$, and if it is reached, go to Step 8. Otherwise go to Step 3 to continue.

Step 8. Update particles position and speed through cluster convergence until the end of the iteration.

5 EXPERIMENT

CloudSim is a tool set for simulating cloud computing environment and evaluating resource scheduling algorithms. We select the CloudSim simulation platform for simulation experiments in this paper, and combine the physical model of cloud computing resource scheduling – the improved particle swarm algorithm with the resource model in CloudSim. Finally, we implement the TPSO algorithm with the inheritance classes in the basic CloudSim class.

The experimental environment of this paper was set as follows: compile environment JDK 1.8, compile software Eclipse 4.6, and simulate cloud computing environment CloudSim 3.0.

The experimental parameters were set as follows: 200 physical machines, 400 virtual machines. All physical machines were homogeneous: CPU with 10 processing units, 20 GB of memory, 100 M bandwidth. We divided 400 VMs into four types, and each type of request had 100 each. We set the request for 400 virtual machines to arrive randomly after each optimization. For different configuration of the algorithms, the requests arrived in the same order. CloudSim resource scheduling mechanism is shown in Figure 3.

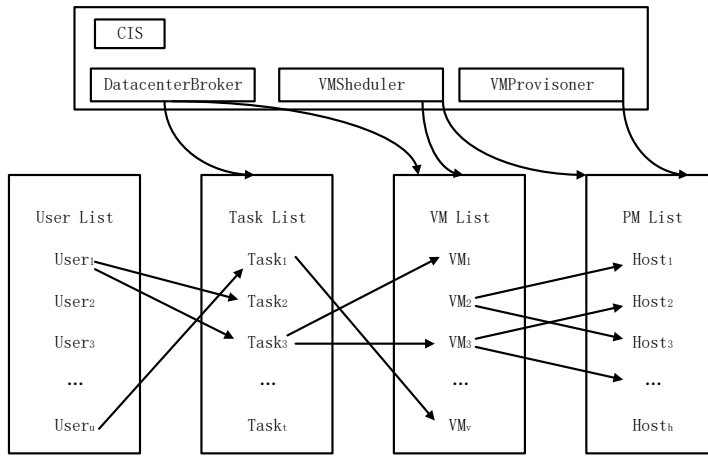

Figure 3. Resourse scheduling mechanism of CloudSim

We designed two experiments with CloudSim. The first experiment compares TPSO with single-objective PSO and DMS-PSO. The second experiment com-

pares the classic particle swarm optimization algorithm and the Round-Robin Algorithm (RR).

5.1 Experiment 1

In Experiment 1, we test the performance of the TPSO by six ten-dimensional benchmark functions. The equations are as follows:

1. Sphere Function
$$f(x) = \sum_{i=1}^{D} x_i^2$$
where $x \in [-5.12, 5.12]^D$.
2. Rosenbrock's Function
$$f(x) = \sum_{i=1}^{D-1} \left[100(x_i^2 - x_{i+1})^2 + (x_i - 1)^2\right]$$
where $x \in [-2.048, 2.048]^D$.
3. Ackley's Function
$$f(x) = -20 \exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} x_i^2}\right) - \exp\left(\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi x_i)\right) + 20 + e$$
where $x \in [-32.768, 32.768]^D$.
4. Griewank's Function
$$f(x) = \sum_{i=1}^{D} \frac{x_i^2}{4\,000} - \prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$$
where $x \in [-600, 600]^D$.
5. Rastrigin's Function
$$f(x) = \sum_{i=1}^{D} (x_i^2 - 10\cos(2\pi x_i) + 10)$$
where $x \in [-5.12, 5.12]^D$.
6. Weierstrass Function
$$f(x) = \sum_{i=1}^{D} \left(\sum_{k=0}^{k\ \max} \left[a^k \cos(2\pi b^k (x_i + 0.5))\right]\right) - D \sum_{k=0}^{k\ \max} \left[a^k \cos(2\pi b^k \cdot 0.5)\right]$$
where $a = 0.5$, $b = 3$, $k_{\max} = 20$, $x \in [-0.5, 0.5]^D$.

We use functions 1–6 to test the algorithm's optimization ability, global search ability, convergence speed, and whether it is liable to fall into a local optimum. The experimental results are as follows.

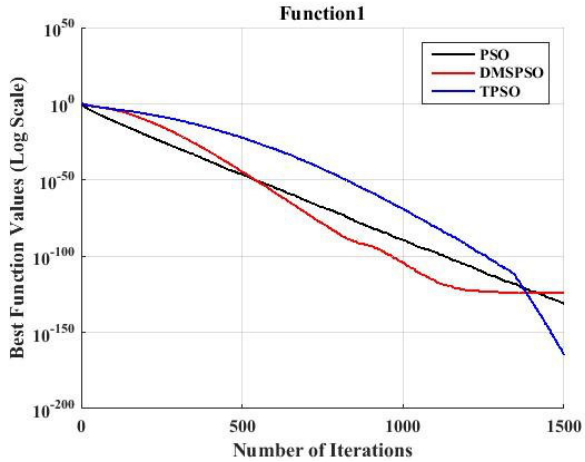

Figure 4. Results achieved under function1

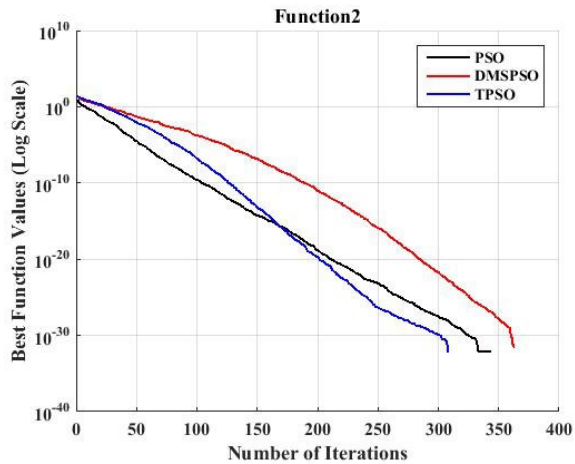

Figure 5. Results achieved under function2

As shown in Figures 4, 5, 6, 7, 8, 9, the six functions are based on experimental results under different iterations. The particle group by chaotic initialization has ergodicity and uniformity. In the early stage, groups constantly exchange information among small groups through random groupings. The late clustering convergence

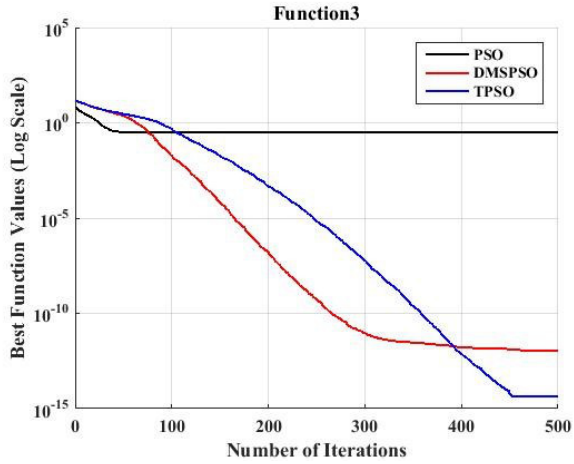

Figure 6. Results achieved under function3

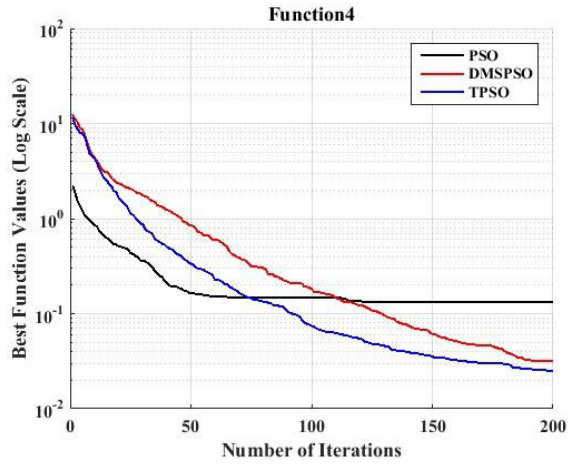

Figure 7. Results achieved under function4

makes the particle group more directional. Experiments show that the improved method TPSO is compared with two single-objective algorithms DMS-PSO and PSO, which is feasible and effective.

5.2 Experiment 2

In the second experiment, we tested and analyzed the performance of the multi-objective optimization algorithm TPSO for virtual machine resource deployment.

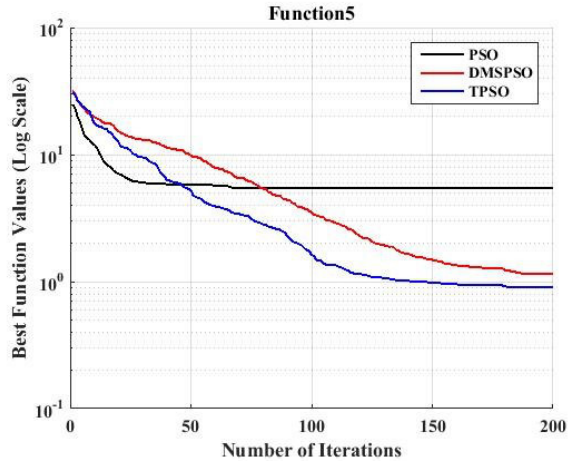

Figure 8. Results achieved under function5

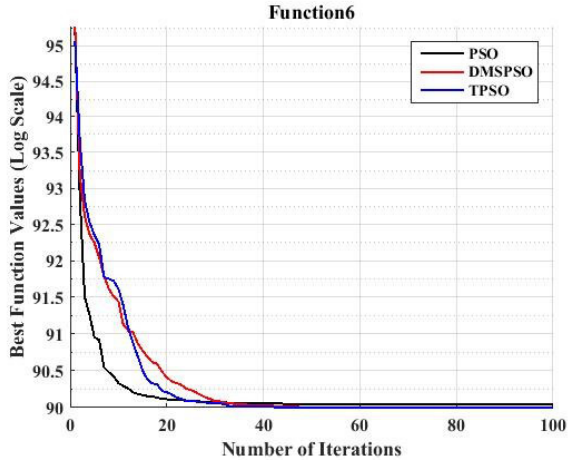

Figure 9. Results achieved under function6

We compared the multi-objective optimization algorithm TPSO with the traditional optimization algorithm PSO and the Round-Robin algorithm RR. The comparison was made from three aspects: resource utilization rate f1, virtual machine migration number f2 and load balance rate f3. Among them, the PSO had the same attention to the three goals, each accounting for 1/3. On the CloudSim platform, simulation experiments were carried out to record relevant data, and the experimental results were compared and analyzed. Finally, we obtained the experimental results distribution of the three algorithms.

Figure 10 shows the resource utilization of the three algorithms under different service requests. It proves that our proposed algorithm TPSO is effective and the resource utilization is higher than the other two algorithms. Figure 11 shows the times of virtual machine migrations. The virtual machine is migrated 220 times under PSO, and the virtual machine is migrated 160 times under TPSO. The proposed algorithm has fewer virtual machine migrations than PSO algorithm. Figure 12 shows the load imbalance for three algorithms at different iterations. It can be found from Figure 12 that the load imbalance of the TPSO algorithm is smaller than the other two algorithms, indicating that the property of the TPSO algorithm in load balance is better than other two algorithms. This is because the RR algorithm has poor dynamic adaptability, its efficiency is relatively low. The PSO algorithm lacks a resource selection mechanism and is liable to fall into local optimality. The population of TPSO algorithm has better ergodicity and uniformity.

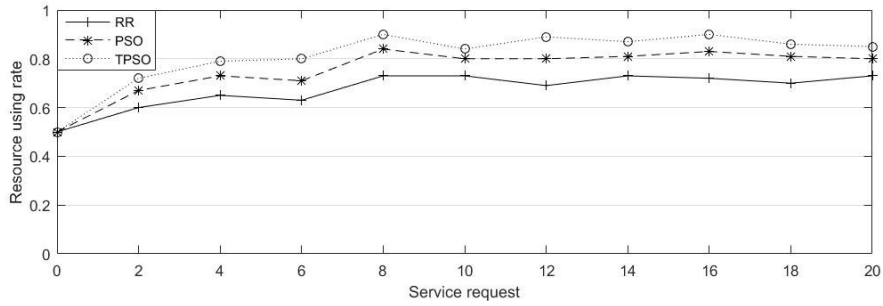

Figure 10. Resource using rate under different algorithms

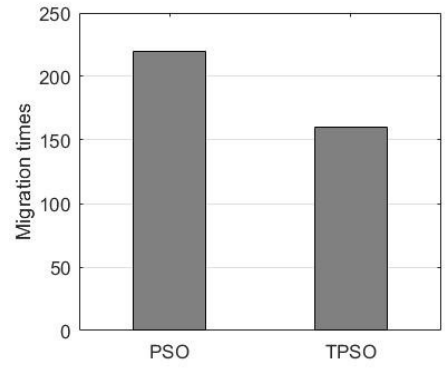

Figure 11. Migration times under different algorithms

The results of Experiment 2 show that the initial particle swarm distribution is more ergodic and uniform, maintaining the diversity of the population. Random

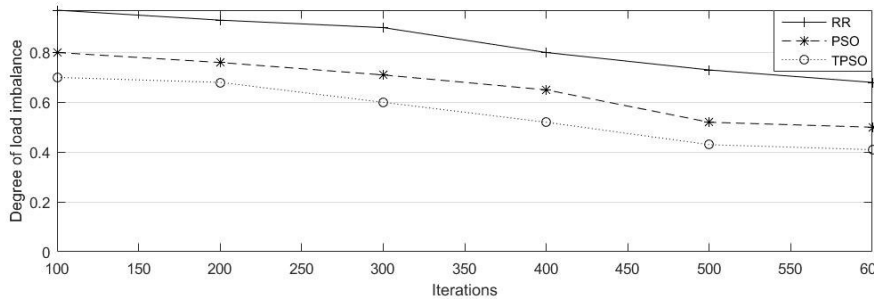

Figure 12. Load imbalance under different algorithms

grouping strategy is used in the medium term. These particles are frequently regrouped to exchange information among groups. Later clustering refined search. With keeping stability, the TOPSIS method selected the Pareto optimal solution. The Pareto optimal solution has better distribution and convergence. Compared to other two algorithms, TPSO has achieved better results in resource utilization, migration times and load balance.

6 CONCLUSIONS

Cloud computing improves the data center resource utilization through virtualization technology. As a key technology of cloud computing, virtual machine deployment algorithm has important research significance. The existing virtual machine deployment strategy only considers maximizing resource utilization and virtual machine migration, ignoring the impact of load balance on system performance. In the paper, we propose an improved multi-objective particle swarm optimization to balance the three goals (resource utilization, virtual machine migration, and load balance) to optimize data center performance.

The TPSO algorithm firstly initializes the particle swarm by chaos, making the population distribution have ergodicity and uniformity. We use the small group iterative update in the middle stage, these small groups are frequently regrouped to exchange information among groups, and later use clustering to converge the particle swarm algorithm, increasing the refinement of the late search. We designed two experiments: Experiment 1 compares the improved method TPSO with single-objective algorithms DMS-PSO and PSO, and verifies the effectiveness of the improved method. Experiment 2 compares TPSO with RR and PSO algorithm. The experimental results show that the algorithm balances the three objectives of resource utilization, virtual machine migration and load balance, and optimizes data center performance.

Acknowledgments

This work was supported by National Natural Science Foundation of China (Nos. 61672033, 61873280, 61873281, 61972416, 61672248, 61902430), National Key Research and Development Project (No. 2018YFC1406204), Key Research and Development Program of Shandong Province (No. 2019GGX101067), Natural Science Foundation of Shandong Province (No. ZR2019MF012), Taishan Scholars Fund (No. ZX20190157), Independent Innovation Research Project (No. 18CX02152A), Fundamental Research Funds for the Central Universities (No. 19CX02028A).

REFERENCES

[1] Cho, H.—Kim, D.—Olivera, F.—Guikema, S. D.: Enhanced Speciation in Particle Swarm Optimization for Multi-Modal Problems. European Journal of Operational Research, Vol. 213, 2011, No. 1, pp. 15–23, doi: 10.1016/j.ejor.2011.02.026.

[2] Dashti, S. E.—Rahmani, A. M.: Dynamic VMS Placement for Energy Efficiency by PSO in Cloud Computing. Journal of Experimental and Theoretical Artificial Intelligence, Vol. 28, 2016, No. 1-2, pp. 97–112, doi: 10.1080/0952813X.2015.1020519.

[3] Garg, H.: A Hybrid PSO-GA Algorithm for Constrained Optimization Problems. Applied Mathematics and Computation, Vol. 274, 2016, pp. 292–305, doi: 10.1016/j.amc.2015.11.001.

[4] Higashi, N.—Iba, H.: Particle Swarm Optimization with Gaussian Mutation. Proceedings of the 2003 IEEE Swarm Intelligence Symposium (SIS '03), 2003, pp. 72–79, doi: 10.1109/SIS.2003.1202250.

[5] Kennedy, J.: Particle Swarm Optimization In: Sammut, C., Webb, G. I. (Eds.): Encyclopedia of Machine Learning. Springer, Boston, MA, 2011, pp. 760–766, doi: 10.1007/978-0-387-30164-8_630.

[6] Kennedy, J.: Small Worlds and Mega-Minds: Effects of Neighborhood Topology on Particle Swarm Performance. Proceedings of the 1999 Congress on Evolutionary Computation (CEC '99), IEEE, 1999, Vol. 3, pp. 1931–1938, doi: 10.1109/CEC.1999.785509.

[7] Kennedy, J.—Eberhart, R.: Particle Swarm Optimization. Proceedings of International Conference on Neural Networks (ICNN '95), IEEE, 1995, Vol. 4, pp. 1942–1948, doi: 10.1109/ICNN.1995.488968.

[8] Kumar, D.—Raza, Z.: A PSO Based VM Resource Scheduling Model for Cloud Computing. 2015 IEEE International Conference on Computational Intelligence and Communication Technology, 2015, pp. 213–219, doi: 10.1109/CICT.2015.35.

[9] Li, M.—Subhraveti, D.—Butt, A. R.—Khasymski, A.—Sarkar, P.: CAM: A Topology Aware Minimum Cost Flow Based Resource Manager for MapReduce Applications in the Cloud. Proceedings of the 21st International Symposium on High-Performance Parallel and Distributed Computing (HPDC '12), 2012, pp. 211–222, doi: 10.1145/2287076.2287110.

[10] Liang, J. J.—Suganthan, P. N.: Dynamic Multi-Swarm Particle Swarm Optimizer with Local Search. 2005 IEEE Congress on Evolutionary Computation (CEC 2005), 2005, Vol. 1, pp. 522–528, doi: 10.1109/CEC.2005.1554727.

[11] Ma, T.—Pang, S.—Zhang, W.—Hao, S.: Virtual Machine Based on Genetic Algorithm Used in Time and Power Oriented Cloud Computing Task Scheduling. Intelligent Automation and Soft Computing, Vol. 25, 2019, No. 3, pp. 605–613.

[12] Maurer, M.—Emeakaroha, V. C.—Brandic, I.—Altmann, J.: Cost-Benefit Analysis of an SLA Mapping Approach for Defining Standardized Cloud Computing Goods. Future Generation Computer Systems, Vol. 28, 2012, No. 1, pp. 39–47, doi: 10.1016/j.future.2011.05.023.

[13] Nickabadi, A.—Ebadzadeh, M. M.—Safabakhsh, R.: A Novel Particle Swarm Optimization Algorithm with Adaptive Inertia Weight. Applied Soft Computing, Vol. 11, 2011, No. 4, pp. 3658–3670, doi: 10.1016/j.asoc.2011.01.037.

[14] Pang, S.—Li, W.—He, H.—Shan, Z.—Wang, X.: An EDA-GA Hybrid Algorithm for Multi-Objective Task Scheduling in Cloud Computing. IEEE Access, Vol. 7, 2019, pp. 146379–146389, doi: 10.1109/ACCESS.2019.2946216.

[15] Pang, S.—Zhang, W.—Ma, T.—Gao, Q.: Ant Colony Optimization Algorithm to Dynamic Energy Management in Cloud Data Center. Mathematical Problems in Engineering, Vol. 2017, 2017, Art. No. 4810514, 10 pp., doi: 10.1155/2017/4810514.

[16] Pant, M.—Thangaraj, R.—Abraham, A.: DE-PSO: A New Hybrid Meta-Heuristic for Solving Global Optimization Problems. New Mathematics and Natural Computation, Vol. 7, 2011, No. 03, pp. 363–381, doi: 10.1142/S1793005711001986.

[17] Quang-Hung, N.—Nien, P. D.—Nam, N. H.—Tuong, N. H.—Thoai, N.: A Genetic Algorithm for Power-Aware Virtual Machine Allocation in Private Cloud. In: Mustofa, K., Neuhold, E. J., Tjoa, A. M., Weippl, E., You, I. (Eds.): Information and Communication Technology (ICT EurAsia 2013). Springer, Berlin, Heidelberg, Lecture Notes in Computer Science, Vol. 7804, 2013, pp. 183–191, doi: 10.1007/978-3-642-36818-9_19.

[18] Sato, K.—Samejima, M.—Komoda, N.: Dynamic Optimization of Virtual Machine Placement by Resource Usage Prediction. 2013 11th IEEE International Conference on Industrial Informatics (INDIN), 2013, pp. 86–91, doi: 10.1109/INDIN.2013.6622863.

[19] Shabeera, T. P.—Madhu Kumar, S. D.—Salam, S. M.—Krishnan, K. M.: Optimizing VM Allocation and Data Placement for Data-Intensive Applications in Cloud Using ACO Metaheuristic Algorithm. Engineering Science and Technology, an International Journal, Vol. 20, 2017, No. 2, pp. 616–628, doi: 10.1016/j.jestch.2016.11.006.

[20] Shabeera, T. P.—Madhu Kumar, S. D.: Optimising Virtual Machine Allocation in MapReduce Cloud for Improved Data Locality. International Journal of Big Data Intelligence, Vol. 2, 2015, No. 1, pp. 2–8, doi: 10.1504/IJBDI.2015.067563.

[21] Shelokar, P. S.—Siarry, P.—Jayaraman, V. K.—Kulkarni, B. D.: Particle Swarm and Ant Colony Algorithms Hybridized for Improved Continuous Optimization. Applied Mathematics and Computation, Vol. 188, 2007, No. 1, pp. 129–142, doi: 10.1016/j.amc.2006.09.098.

[22] Song, T.—Wang, Y.—Li, G.—Pang, S.: Server Consolidation Energy-Saving Algorithm Based on Resource Reservation and Resource Allocation Strategy. IEEE Access, Vol. 7, 2019, pp. 171452–171460, doi: 10.1109/ACCESS.2019.2954903.

[23] Wang, H.—Sun, H.—Li, C.—Rahnamayan, S.—Pan, J.-S.: Diversity Enhanced Particle Swarm Optimization with Neighborhood Search. Information Sciences, Vol. 223, 2013, pp. 119–135, doi: 10.1016/j.ins.2012.10.012.

[24] Wang, H.—Wu, Z.—Rahnamayan, S.—Liu, Y.—Ventresca, M.: Enhancing Particle Swarm Optimization Using Generalized Opposition-Based Learning. Information Sciences, Vol. 181, 2011, No. 20, pp. 4699–4714, doi: 10.1016/j.ins.2011.03.016.

[25] Wang, J.—Huang, C.—Liu, Q.—He, K.—Wang, J.—Li, P.—Jia, X.: An Optimization VM Deployment for Maximizing Energy Utility in Cloud Environment. In: Sun, X. et al. (Eds.): Algorithms and Architectures for Parallel Processing (ICA3PP 2014). Springer, Cham, Lecture Notes in Computer Science, Vol. 8630, 2014, pp. 400–414, doi: 10.1007/978-3-319-11197-1_31.

[26] Wigderson, A.: P, NP and Mathematics – A Computational Complexity Perspective. Proceedings of the 2006 International Congress of Mathematicians, Madrid, 2006. EMS Publishing House, Zurich, 2007, pp. 665–712.

[27] Wilcox, D.—McNabb, A.—Seppi, K.: Solving Virtual Machine Packing with a Reordering Grouping Genetic Algorithm. 2011 IEEE Congress of Evolutionary Computation (CEC), 2011, pp. 362–369, doi: 10.1109/CEC.2011.5949641.

[28] Xu, B.—Peng, Z.—Xiao, F.—Gates, A. M.—Yu, J.-P.: Dynamic Deployment of Virtual Machines in Cloud Computing Using Multi-Objective Optimization. Soft Computing, Vol. 19, 2015, No. 8, pp. 2265–2273, doi: 10.1007/s00500-014-1406-6.

[29] Zhan, Z.-H.—Zhang, J.—Li, Y.—Shi, Y.-H.: Orthogonal Learning Particle Swarm Optimization. IEEE Transactions on Evolutionary Computation, Vol. 15, 2010, No. 6, pp. 832–847, doi: 10.1109/TEVC.2010.2052054.

[30] Zhao, G.: Cost-Aware Scheduling Algorithm Based on PSO in Cloud Computing Environment. International Journal of Grid and Distributed Computing, Vol. 7, 2014, No. 1, pp. 33–42, doi: 10.14257/ijgdc.2014.7.1.04.

[31] Zhao, X.—Lin, W.—Zhang, Q.: Enhanced Particle Swarm Optimization Based on Principal Component Analysis and Line Search. Applied Mathematics and Computation, Vol. 229, 2014, pp. 440–456, doi: 10.1016/j.amc.2013.12.068.

[32] Zhao, X.—Liu, Z.—Yang, X.: A Multi-Swarm Cooperative Multistage Perturbation Guiding Particle Swarm Optimizer. Applied Soft Computing, Vol. 22, 2014, pp. 77–93, doi: 10.1016/j.asoc.2014.04.042.

[33] Zhao, X.—Song, B.—Huang, P.—Wen, Z.—Weng, J.—Fan, Y.: An Improved Discrete Immune Optimization Algorithm Based on PSO for QoS-Driven Web Service Composition. Applied Soft Computing, Vol. 12, 2012, No. 8, pp. 2208–2216, doi: 10.1016/j.asoc.2012.03.040.

Shanchen Pang received his graduation degree from the Tongji University of Computer Software and Theory, Shanghai, China, in 2008. He is Professor in the China University of Petroleum, Qingdao, China. His current research interests include theory and application of Petri net, service computing, trusted computing.

Dekun Dong graduated from the Shandong University of Science and Technology and got his Bachelor degree in engineering and management. Currently, he is pursuing his Master's degree in the China University of Petroleum. His research area is cloud computing.

Shuyu Wang received her graduation degree from the Shandong Women's University, Jinan, China, in computer science and technology in 2018. Currently, she is pursuing her Master's degree in the China University of Petroleum, Qingdao, China. Her current research interests include cloud computing and workflow scheduling.