International Journal of Multimedia and Ubiquitous Engineering
Vol.11, No.10 (2016), pp.285-296
http://dx.doi.org/10.14257/ijmue.2016.11.10.27

Application of a Hybrid Intelligent Optimization Algorithm in Cloud Computing Resources Scheduling

Zhang Chun-na[1] and Li Yi-ran[2]

[1]School of Software, University of Science and Technology Liaoning, Anshan Liaoning 114051, China;
[2]College of Applied Technology, University of Science and Technology Liaoning, Anshan Liaoning 114011, China
E-mail: zcn1979@163.com

Abstract

The resource scheduling imbalance is a multi-objective optimization problem in cloud computing environment, this paper introduced the particle swarm optimization algorithm in cloud computing, a simulated annealing ideas is proposed in view of the prematurity of the algorithm, on the premise of performance determination, the position of the particle is determined by the probability choice, which helps the particle to escape. In order to enhance the global searching ability of the particle, the algorithm is combined with the chaotic mechanism to improve the accuracy of the algorithm. The inertia weight is adjusted dynamically according to the current state of the particle, accordingly, at the same time to obtain the optimal solution to ensure the convergence. Analysis of the experimental results show that the improved algorithm has a significant improvement in the ability of optimization and convergence speed, compared with other algorithms, the benchmark functions comparison is better, the different resource task proportion spent the shortest time and load balancing is the highest.

***Keywords**: cloud computing; resource load; particle swarm; simulated annealing; convergence*

1. Introduction

The cloud computing is a new type of collaborative shared computing model, it is a combination of distributed computing, peer-to-peer computing, and grid computing. Virtualizes the resources, the system through a variety of deployment to meet the user's resource sharing, application hosting, service outsourcing and other requests [1-2]. With the continuous expansion of the scale of the Internet, a series of problems of resource sharing have been produced, the root cause is attributed to the imbalance of resource allocation. Therefore, the resource scheduling strategy is a very important link in the cloud computing environment.

At present, the scheduling strategy is divided into two kinds: static scheduling algorithm and dynamic scheduling algorithm. The former is mainly the pre-allocation strategy, complete the simple request - distribution process, it is easy to cause the waste of resources. In the cloud computing environment, Resources have characteristics such as dynamic, heterogeneous and so on, when starting a large-scale calculation, not only need to consider the load of the system, but also need the appropriate response time, so the static resource scheduling is not suitable, it is advisable to adopt the dynamic allocation method [3-6]. In summary, the cloud computing resource scheduling problem is actually a multi-objective optimization strategy under certain constraint conditions, the solution includes genetic algorithm, ant colony algorithm, leapfrog algorithm and particle swarm algorithm. Compared with other optimization algorithms, particle swarm optimization algorithm (PSO) has the advantages of simple structure, less parameter settings, no gradient information, and it has become the main stream algorithm of the current research on resource scheduling problem [7-10]. But the particle swarm optimization algorithm is over dependent on the characteristics of the optimal particle, which leads to the decrease of the late performance of some particles, which is easy to fall into the local optimum, and the convergence rate is decreased.

In order to ensure the accuracy and convergence of the particle swarm optimization algorithm, this paper presents an improved particle swarm optimization algorithm based on simulated annealing, which is called ISACPSO. The algorithm based on cloud computing environment resource scheduling analysis, established related resource scheduling objective function, and introduced the chaos mechanism to enhance the global search ability of the algorithm; at the same time, the position and velocity of the particles are updated by the simulated annealing and the adjustment of the inertia weight, in order to avoid premature phenomenon, and then improve the searching ability of the algorithm, and ensure the convergence.

2. The Cloud Computing Resource Scheduling Problem Description

The larger scale task in the cloud computing environment should be decomposed into several easy to handle and independent of each other small tasks, each small task provide certain information, the system adopt appropriate strategy to match with the corresponding resources, all small tasks completed, the whole task is over [11]. At present, the Map/Reduce proposed by Google is the most widely used programming model, it divides the whole process into two parts: Map and Reduce, in this way, a task includes several Map tasks as well as several Reduce tasks. In the Map implementation phase, the corresponding tasks are processed, and the calculated results are mapped to the Reduce task; in the Reduce implementation phase, the data from the Map phase is further processed and the results are obtained.

In cloud computing environment, the virtual resource allocation framework is defined as follows: set the overall task, that is, the total load is T, it is divided into several independent of each other sub load, using $T = \{t_1, t_2, \cdots t_n\}$ to represent. These loads need to run on the appropriate resources, set R to represent the resources, and the resource set is represented as $R = \{r_1, r_2, \cdots r_n\}$, provides a virtual resource only to support a load operation, set the set $E = \{e_1, e_2, \cdots e_n\}$ is a collection of physical devices. In this way, the resource scheduling problem in cloud computing environment is transformed into the solving process of TR matrix. The tr as all feasible solutions, the set TR matrix can be expressed as:

$$TR = \begin{bmatrix} tr_{1,1} & tr_{1,2} & \cdots & tr_{1,n} \\ tr_{2,1} & & & \\ \vdots & & \ddots & \\ tr_{m,1} & & & tr_{m,n} \end{bmatrix} \tag{1}$$

In the formula, $tr_{i,j}$ is the bearing relationship between the load t_i and the virtual resource r_k, which satisfies $\{tr \mid tr \in [0,1], \sum_{i=1}^{m} tr_{i,j} = 1\}$.

According to the relationship between t_i and r_i, set $TE(t_i, e_i)$ is consumed time of t_i performs on e_i, and the corresponding matrix can be expressed as:

$$TE = \begin{bmatrix} te_{1,1} & te_{1,2} & \cdots & te_{1,n} \\ te_{2,1} & & & \\ \vdots & & \ddots & \\ te_{m,1} & & & te_{m,n} \end{bmatrix} \tag{2}$$

In the formula, $te_{i,j}$ is the time that the i load performs on the j physical device. Here set a physical device e_j to perform the load t_i, remove the waiting time for the earliest start time is w_j, then the total execution time of the load t_i on the e_j can be defined as:

$$S(t_i) = w_j + TE(t_i, e_i) \tag{3}$$

For all load $\{t_1, t_2, \cdots, t_n\}$, the total execution time can be defined as:

$$S'(t) = \sum_{i=1}^{n} S(t_i) \tag{4}$$

This paper focuses on the study of time, so the load consumption time as the objective function should be satisfied to obtain a minimum value, defined as follows:

$$F(t) = \min \sum_{i=1}^{n} S(t_i) \tag{5}$$

3. Particle Swarm Optimization Algorithm

3.1. Basic Particle Swarm Optimization Algorithm

In the particle swarm algorithm, the search process of the particles is the iterative process of the algorithm, and the speed and position of the particle is updated in each iteration [12-14]. In Q dimension space, the particle is initialized to $\{x_1, x_2, \cdots x_n\}$, the position of the i particle can be expressed as $x_i = \{x_{i1}, x_{i2}, \cdots x_{iQ}\}$, the particle velocity is $v_i = \{v_{i1}, v_{i2}, \cdots v_{iQ}\}$, $i = 1, 2, \cdots, n$, the trajectory of the particle is random, in flight it closely follows the trajectory of the optimal particle, including the local optimal extreme value Pb and the global optimal extreme value Gb. Each iteration of the algorithm investigate the fitness function value to judge the advantages and disadvantages of the current particle, according to the actual situation, the speed and position of the particles are updated. Through constant judgment and renewal, get the best particle, that is, the optimal solution. The basic formula of the velocity and position of the particle is as follows:

$$v'_{iq} = \alpha v''_{iq} + \beta^0 (x_{Pb} - x_{iq}) + \beta^1 (x_{Gb} - x_{iq}) \tag{6}$$

$$x'_{iq} = x''_{iq} + v'_{iq} \tag{7}$$

In the formula, α is the inertia weight, which is used to balance the velocity relationship in the flight of the particle. β^0、β^1 is the correction factor, $\in [0,1]$, and the random distribution.

3.2. Chaos Mechanism

Chaos phenomenon is widespread in nature, it belongs to the category of nonlinear, which has the characteristics of ergodicity and regularity, It is sensitive to initial conditions, can be in a given region in accordance with their own laws to search all the internal state, and does not repeat [15-16]. In this way, can use the nature of chaos to optimize search, the search procedure is as follows:

(1) Define the initial region, set N dimensional initial state the vector $R_0 = (R_{01}, R_{02}, \cdots, R_{0N})$, the values in R_0 are adjacent to each other, and the difference is very small.

(2) The logistics equation is used to calculate the initial vector R_0, to generate a chaotic sequence $c_1, c_2, \cdots, c_n$. Here, after several iterations, the system will be completely in a state of chaos. Vector layer can be expressed as:

$$c_{i+1} = c_i(1 - c_{i-1})\lambda \tag{8}$$

In the formula, λ is an iterative control parameter.

(3)Set the space particle X_i, use the formula (8) to get a better position of X_i, as $X_i^{'}$.

$$X_i^{'} = r \cdot rnd \cdot c_j + X_i \tag{9}$$

In the formula, r is the active radius of the particle X_i, $rnd \in [-1,1]$, $j \in [0,n]$.

The main idea of the particle swarm algorithm based on chaotic mechanism is reflected in the following aspects: on the one hand, the position and velocity of particles are initialized by chaos sequence, because of the characteristics of ergodicity, it not only maintains the diversity of the particle, but also enhances the search ability of the particle; Furthermore, the chaotic state can make the particle motion is continuous.

Particle chaos initialization: the X_i in the formula (9) were given the initial value, and re-modify the velocity of particle swarm in the iteration.

$$\begin{aligned} v_{i,j}^{'}(t+1) = {} & a v^{''}(t) + b^0(t)(x_{lb}(t) - x_{i,j}(t)) \\ & + b^1(t)(x_{gb}(t) - x_{i,j}(t)) \end{aligned} \tag{10}$$

In the formula, a is the constant, $\in (0,1]$, b is the random number of normal distribution $N[0,1]$, $i \in [1,n]$, $j \in [1,m]$, n is the number of particles, m is the spatial dimension. For $v^{''}(t)$:

$$v^{''}(t) = \begin{cases} v_{i,j}(t) & q = 0 \\ N[0,1] \times d \times \tilde{v} & q = 1 \end{cases} \tag{11}$$

$$q = \begin{cases} 0 & f(x_{gb}(t-1)) > f(x_{gb}(t)) \\ 1 & f(x_{gb}(t)) = f(x_{gb}(t-1)) = \\ & \cdots = f(x_{gb}(t-5)) \end{cases} \tag{12}$$

In the formula, $\tilde{v} = v_{\max} \times c_i / 1.1$, $d = f(x_{gb}) - f(x_T)$, c_i is a new chaotic sequence, $f(x_{gb})$ is a satisfactory solution, $f(x_T)$ is the target solution.

3.3. Analysis of Inertia Weight

The introduction of chaos mechanism is to improve the accuracy of the algorithm, for the problem that the particle swarm optimization algorithm is easy to fall into local optimum, and the convergence rate is slow in late, this paper attempts to amend the inertia weight to achieve. In the process of particle evolution, each iteration will change its position and velocity. Among them, the velocity is used to measure the activity of the particles, and is also the key factor of the convergence of the algorithm, and the size of the inertia weight will directly affect the renewal of speed. When the inertia weight is larger, the particle flying speed is faster, and the global search ability is stronger; the inertia weight is small, the particle in the small range search ability can be reflected, the local search ability is stronger. Based on this, the inertia weight is modified in the design of particle swarm optimization algorithm, In the initial stage of the algorithm, the inertia weight is larger, so that the particles can fly at a faster speed in the global range, which is beneficial to the acquisition of Gb; in the latter part of the algorithm, the inertia weight is smaller, so that the particle can quickly get Pb in the local area, so as to enhance the global convergence of the algorithm. Therefore, the inertia weight is a gradual reduction process in the iterative algorithm, the paper is set to change with the cosine law, the related formula is as follows:

$$\theta = \theta_0 + (\theta_1 - \theta_0) \cdot \cos(\frac{s_0 \pi}{2 s_1}) \tag{13}$$

In the formula, θ_0、θ_1 is the minimum and maximum value of the inertia weight, $\theta \in [\theta_0、\theta_1]$, s_0、s_1 is the current iteration number. This setting guarantees that the process of falling speed is gradual, with the increase of iteration number, the influence of inertia weight on the speed is reduced, slow change of speed is more conducive to global to local search process, to ensure the stability of the algorithm.

3.4. Particle Performance Analysis

The introduction of chaos mechanism enhances the global searching ability of the algorithm, but does not improve the local search ability, the premature phenomenon of particle swarm optimization algorithm still exists, because the particle velocity is slow in the later period, some particles appear oscillation phenomenon, the algorithm is easy to fall into local optimum, and can't get the optimal solution. Here, if the successive generations of particles appear oscillation and stagnation, then determine the algorithm appears premature. This paper plan to deal with particles by means of variation, that is, to adjust the flight direction of the particles with performance degradation, to make them reverse flight and escape from the local space, and use the load gradient strategy to accelerate convergence, the formula is as follows:

$$-\nabla \psi(F_{Gb}^{'}) = \frac{\psi(F_{Gb}^{"}) - \psi(F_{Gb}^{'})}{F_{Gb}^{"} - F_{Gb}^{'}} \tag{14}$$

In the formula, $F_{Gb}^{'}$ is the best fitness value at present, $F_{Gb}^{"}$ is its former value.

The velocity and position correction formula of the variant particles are as follows:

$$v_{iq}^{'} = -\alpha v_{iq}^{"} + \beta^0 (x_{Pb} - x_{iq}) + \beta^1 (x_{Gb} - x_{iq}) \tag{15}$$

$$x_{iq}^{'} = x_{iq}^{"} - v_{iq}^{'} \tag{16}$$

3.5. Simulated Annealing

The idea of simulated annealing is derived from the principle of solid annealing, the solid temperature increases to the highest point, and then gradually cooling, when the temperature of the solid is increased, its internal particles gradually become disordered, the particles are gradually ordered with the decrease of temperature, and the temperature reaches the normal temperature, the internal energy is the smallest. In order to improve the global search ability of particles, so they can quickly escape from local optimum bound to get a better update position, algorithm intends to fusion the idea of simulated annealing.

According to the Metropolis criterion, the probability of the particle equilibrium in the temperature T is $e(\Delta E / kT)$. Among them, E is the internal energy of the reference when the temperature is T, ΔE is the amount of change, k is the Boltzmann constant. The change ratio of objective function in particle swarm optimization algorithm is ΔE, the control parameters t simulate temperature T, so as to get the optimization algorithm to solve the problem, the algorithm starts with the initial solution and the control parameter

t, simulated annealing temperature falling process, new solutions in the iteration produce continuously, the t value is smaller, the objective function value is constantly changing, accept or discard the current solution, until the end of the algorithm to obtain the optimal solution.

The idea of simulated annealing simulate the particle iterative optimization process as particle annealing process, the particle position update is determined by the probability choice, that is, the particle follows the optimal particle to arrive at a new position, compared to the fitness value, if the fitness value is better than the previous position, then move; otherwise, move according to the annealing probability $e(\Delta E / kT)$. This location update is controllable, and it can effectively avoid the premature.

4. The Cloud Computing Resource Scheduling Strategy Based on Improved Particle Swarm Optimization

4.1. Coding Strategy

Based on the characteristics of cloud computing resource scheduling, this paper intends to use decimal encoding rules, The particle encoding method can be expressed as $\{r_1, r_i, \cdots r_j, \cdots r_k\}$, the code length is determined by the number of sub tasks, where r_k is the k resources. This is a kind of coding rule which corresponds to the resources and tasks, namely a sub task corresponds to a resource. The specific allocation method is shown in the following table:

Table 1. Coding Strategy Table

Task code	Sub task code	Resource library number
1	1 2	1 3
2	3 4 5	2 3 1
3	6 7 8	4 2 1
4	9 10 11	3 4 2

In Table 1, the number of tasks is 4. Among them, 1, 2, 3, 4 task were divided into 2, 3, 3, 3 sub tasks, the corresponding resources are respectively {1 3}, {2 3 1}, {4 2 1}, {3 4 2}, then the corresponding codes are{1 3 2 3 1 4 2 1 3 4 2}.

4.2. Fitness Function and Algorithm Step

In the iteration of the algorithm, the position of the next generation particles is determined by the value of the fitness function. Cloud computing scheduling strategies need to consider two factors: time and cost. This paper focuses on the completion time of the task, that is, the higher the fitness value, the better the performance of the particle, the greater the possibility of the optimal solution, the corresponding fitness function is as follows:

$$f(t) = \frac{1}{\min \sum_{i=1}^{n} S(t_i)} \tag{17}$$

The following is the algorithm step:

Step1: Determine the scheduling objective function and the corresponding parameters of the task in cloud computing environment;

Step2: Initialize the particle swarm, determine the size of the population and the maximum number of iterations, determine the cluster center vector for each particle within the specified range of the data vector , and set the initial local and global optimal;

Step3: According to the fitness function calculate the fitness value, select the best position to initialize the particle;

Step4: The current state of the particle is determined by iteration, the reference formula (5), (6), and the velocity and position of the particles are updated by the simulated annealing;

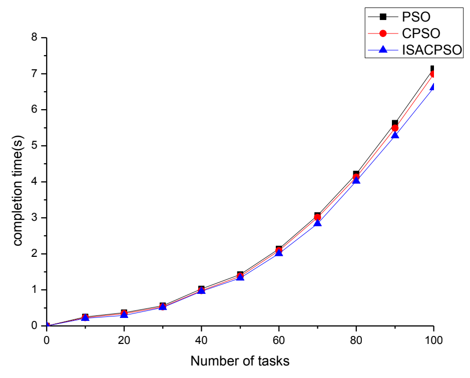
Step5: Iteration number plus 1, produce a new generation of particles;

Step6: Contrast to the termination conditions of algorithm to determine whether the conditions are met, if not meet return to step 4.

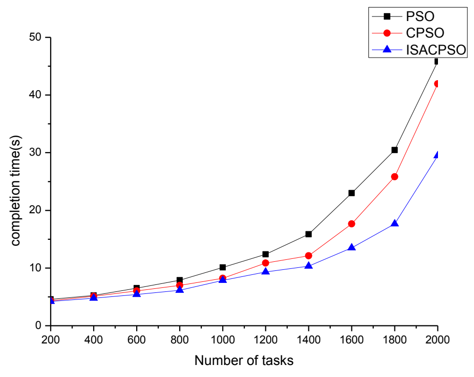
5. Experimental Analysis

The performance of the algorithm is analyzed by using the CloudSim platform, including the comparison of different task completion time, resource balance degree and convergence.

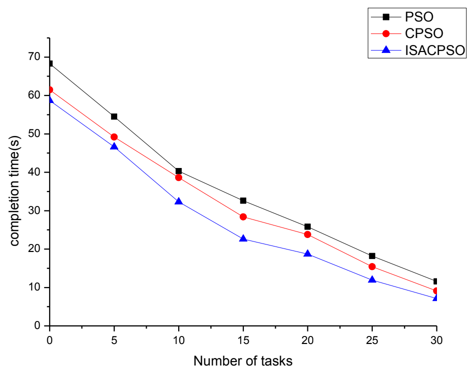
Three particle swarm optimization algorithms are used in the experiment, standard particle swarm optimization algorithm, PSO; particle swarm optimization with chaos, CPSO; improved particle swarm algorithm is proposed in this paper, ISACPSO. Setting the population size is 20, the inertia weight is 2, the maximum number of iteration is 400. Three group task scheduling test were compared, include: (1) 10-100 tasks paired with 10 resource device nodes; (2) 200-2000 tasks paired with 25 resource device nodes; (3) Fixed 3000 tasks paired with 30 resource device nodes. The comparison results of the three algorithms are shown below.



(a) Less Task Completion Time Ratio



(b) More Task Completion Time Ratio



(c) Completion Time Compared with Fixed Number of Nodes

Figure 1. Matching Contrast of Task and Node

According to the analysis of Figure 1, in the three group task scheduling, (a) and (b) is a dynamic allocation, (c) is a static allocation, among them, when the task is less, such as graph (a), the performance of the three algorithms is not very different, and the time-consuming is relatively short, with the increase of the number of tasks, the completion time of the algorithm is increased accordingly, however, the time required of ISACPSO is the shortest. When the number of tasks reaches 1600, the performance of the improved algorithm is more obvious. The number of tasks and resources in figure (c) are fixed, in the comparison of the three algorithms, when the resources of initial stage is less, the ISACPSO uses the least time, with the increase of the number of resources, the phenomenon of seize the resources can be alleviated, thanks to its global search and local escape ability to improve, the improved algorithm is still maintained advantage.

In order to investigate the resource utilization ratio of the algorithm, the resource load balancing index is used to analyze, the value is close to 1, the higher the resources utilization ratio. The following figure is the result of the comparison:

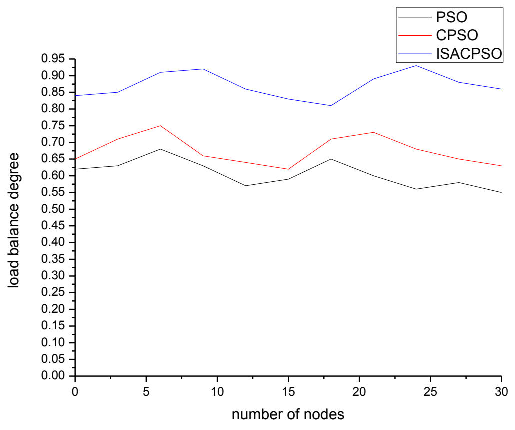
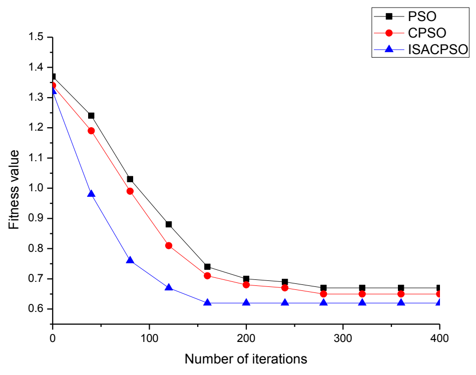


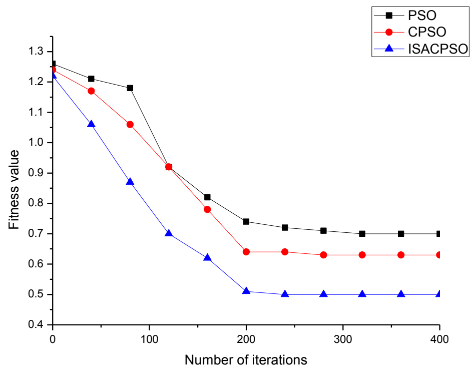
Figure 2. Comparison of Resource Balance Degree

From the comparison and analysis of the graph can know that the improved algorithm is better in resource load utilization ratio, the lowest is 0.84, the maximum is 0.93, and the curve is more smooth, in contrast, the degree of balance is hovering between 0.55-0.75 in the other two algorithm, which shows that the improved algorithm to perform task is more smooth.

In terms of convergence, 2 benchmark functions are used to analyze the three algorithms, and the test is random, the number is 100, take mean value.



(a) Comparison of *Schwefel* Convergence



(b) Comparison of *Rosenbrock* Convergence

Figure 3. Comparison of the Convergence of Benchmark Functions

From the analysis of Figure 3, the convergence performance of the two benchmark functions corresponding to the improved algorithm performed well, especially the advantage is more obvious in the middle and late stage. The main reason is that the algorithm is combined with the chaotic mechanism, the global search ability is improved, to deal with poor performance of the particles by the variation and the simulated annealing process, so that the particle speed is guaranteed, effective escape from the local optimal constraints, therefore, the performance of the algorithm is improved in the later period, and the convergence is guaranteed.

6. Conclusion

The resource scheduling is completed in the cloud computing environment, in view of the traditional particle swarm optimization algorithm is prone to premature phenomena, and to optimize it in this paper. The global searching ability of the algorithm is improved by the chaos mechanism, and the inertia weight of impact velocity is adjusted dynamically, which ensures the convergence speed; for the position and velocity of the particle, using simulated annealing to determine and update the position of particle, the poor performance particles were treated by variation. In this way, it can effectively improve the speed and accuracy of the algorithm.

References

[1] W. AL. Museelem and C. Li, "User privacy and security in cloud computing", International Journal of Security and its Applications, vol. 10, no. 2, **(2016)**, pp. 341-352.
[2] N. Farrukh and Q. Rizwan, "An Early Evaluation and Comparison of Three Private Cloud Computing Software Platforms", Journal of Computer Science and Technology, vol. 30, no. 3, **(2015)**, pp. 639-654.
[3] P. N. Sriram and H. J. Zygmunt, "A practical, secure, and verifiable cloud computing for mobile systems", Proceeding Computer Science, vol. 34, no. C, **(2014)**, pp. 474-483.
[4] J. P. McGlothlin and L. Khan, "Scalable queries for large datasets using cloud computing: a case study", Proceedings of the 15th Symposium on International Database Engineering & Applications, New York, USA, **(2011)**, pp. 8-16.
[5] G. V. Laszewski, F. Wang, H. Lee, H. Chen and G. C. Fox, "Accessing multiple clouds with cloudmesh", Proceedings of the 2014 ACM international workshop on Software-defined ecosystems, New York, USA, **(2014)**, pp. 21-28.
[6] G. Shivam and M. C. Subhas, "Compliance, network, security and the people related factors in cloud ERP implementation", International Journal of Communication Systems, vol. 29, no. 8, **(2016)**, pp. 1395-1419.
[7] A. B. Hashemi and M. R. Meybodi, "A note on the learning automata based algorithms for adaptive parameter selection in PSO", Applied Soft Computing Journal, vol. 11, no. 1, **(2011)**, pp. 689-705.
[8] M. G. Epitropakis, V. P. Plagianakos and M. N. Vrahatis, "Evolving cognitive and social experience in Particle Swarm Optimization through Differential Evolution: A hybrid approach", Information Sciences, vol. 216, no. 24, **(2012)**, pp. 50-92.
[9] B. Xu, Z. Peng, F. Xiao, G. A. Marce and J. P. Yu, "Dynamic deployment of virtual machines in cloud computing using multi-objective optimization", Soft Computing, vol. 19, no. 8, **(2015)**, pp. 2265-2273.
[10] P. B. Miranda and R. B. Prudêncio, "GEFPSO: A Framework for PSO Optimization based on Grammatical Evolution", Proceedings of the 2015 Annual Conference on Genetic and Evolutionary Computation, New York, USA, **(2015)**, pp. 1087-1094.
[11] L. Maya and L. Hyotaek, "Homomorphic encryption in mobile multi cloud computing", International Conference on Information Networking, Siem Reap, Cambodia, **(2015)**, pp. 493-497.
[12] A. E. Dor, M. Clerc and P. Siarry, "A multi-swarm PSO using charged particles in a partitioned search space for continuous optimization", Journal Computational Optimization and Applications, vol. 53, no. 1, **(2012)**, pp. 271-295.
[13] B. Tuli, S. Arindam, S. Bijan and S. K. Subir, "A new meta-heuristic PSO algorithm for resource constraint project scheduling problem", Advances in Intelligent Systems and Computing, vol. 202, no. 2, **(2012)**, pp. 381-392.
[14] O. A. Behrooz, M. Pooya and S. P. Masoud, "An improved PSO algorithm with a territorial diversity-preserving scheme and enhanced exploration-exploitation balance", Swarm and Evolutionary Computation, vol. 11, no. 2, **(2013)**, pp. 1-15.
[15] S. Mukhopadhyay and S. Banerjee, "Global optimization of an optical chaotic system by Chaotic Multi Swarm Particle Swarm Optimization", Expert Systems with Applications, vol. 39, no. 1, **(2012)**, pp. 917-924.
[16] M. Pluhacek, R. Senkerik, D. Davendra, Z. K. Oplatkova and I. Zelinka, "On the behavior and performance of chaos driven PSO algorithm with inertia weight", Computers & Mathematics with Applications, vol. 66, no. 2, **(2013)**, pp. 122-134.

Authors

C. N. Zhang, received the Master's degree in computer application technology from University of Science and Technology Liaoning, in 2007. Currently, she is a lecturer at School of Software Engineering at University of Science and Technology Liaoning. Her research interests include Distributed computing and data mining.

Y. R. Li, received the Master's degree in computer application technology from University of Science and Technology Liaoning, in 2008. Currently, he is a lecturer at School of applied technology college at University of Science and Technology Liaoning. His research interests include Distributed computing and data mining.