Improved MOPSO Algorithm Based on Map-Reduce Model in Cloud Resource Scheduling

Heng-Wei ZHANG[1], Kan NIU[1]*, Jin-Dong WANG[1], Na WANG[1]

[1]Zhengzhou Institute of Information Science and Technology, Zhengzhou 450001, China

State Key Laboratory of Mathematical Engineering and Advanced Computing, Zhengzhou 450001, China

niukan_nk@sina.com

Received 9 November 2015; Revised 23 November 2015 ; Accepted 8 January 2016

Abstract: A multi-objective resource scheduling model with quality of service (QoS) restriction was built to improve the computing efficiency of Map-Reduce resource scheduling. The model considered the scheduling problem of both Map and Reduce phase and a chaotic multi-objective particle swarm algorithm was proposed to solve the model. The information entropy theory was used to maintain nondomination solution set by the algorithm so as to retain the diversity of solution and the uniformity of distribution. By Sigma methods to achieve fast convergence, chaotic disturbance mechanism was introduced to improve the diversity of the population and the ability of global optimization algorithm, which can avoid the algorithm from falling into local extremism. The experiments show that the number of iteration in the algorithm obtaining solutions is little and nondomination solutions distribute equably. In solving Map-Reduce resource scheduling problems, it indicates that the astringency and the diversity of solution set of this algorithm are better than the traditional multi-objective particle swarm algorithm.

Keywords: cloud computing; resource scheduling; map-reduce model; particle swarm algorithm; chaotic disturbance

1 Introduction

Cloud computing is a new computing model developed by grid computing, parallel computing and peer-to-peer (P2P) technology. Cloud computing adopts virtualization technology that links a large amount of computing resources, storage resources and software resources together to form a large-scale shared virtual resource pool and provides cloud services for the user or application system[1-3]. The user can access the needed services at any time and place in cloud computing mode. So reasonable, efficient resource scheduling is the key factor affecting the quality of cloud services and is a research emphasis, difficulty in the field of cloud computing. Map-Reduce is a cloud computing model[4-5] proposed by Google. Although Google has published a large number of open codes[6] for models, some details for the models have not been published so far. Because of the implementation being simple and utilizing the advantages of cloud resources efficiently, it receives the extensive attention of academia and discussion.

Map-Reduce resource scheduling means that how to assign a large number of the tasks to the corresponding virtual computing resources and satisfy the requirement of mission complete time, cost, etc to the utmost extent in the Map and Reduce two stages. On the one hand, Map-Reduce resource scheduling problem has been proved to be an NP-Hard problem. On the other hand, because cloud computing reduces the access resource threshold and combines with a huge number of network resources to form a huge candidate resource pool, it greatly increases the solution space scale of the problem. At the same time, the demand for resources in the cloud computing environment model puts forward the serious challenge to the efficiency of resource scheduling. Therefore traditional algorithms such as FIFO scheduling algorithm, fair scheduling algorithm, computing power scheduling algorithm have the high time complexity and do not apply to the Map-Reduce resource scheduling. In recent years, more and more intelligent algorithms are applied in the cloud resource scheduling problem, but the research achievements are few in the Map-Reduce resource scheduling. Literature [7] used the ant colony algorithm to design the Map-Reduce scheduling scheme. It effectively reduced the task response time. Literature [8] used the genetic algorithm to reduce the total completion time and the average completion time. But none of the above literatures respectively consider scheduling process of Map phase and Reduce phase. The algorithms practicability of the above literatures had limitations. Literature [9] considered Map and Reduce scheduling processes. But it was difficult to coordinate the objective function value when the multi-objective optimization linear weighting transformed a single objective function. It did not produce multiple feasible solutions. The users had no choice. The algorithm of the Literature [9] did not consider the QoS constraints and had limitations in use. Literature [10] turned the resource scheduling problem into multi-objective optimization problem and improved the multi-objective genetic algorithm based on methods in the field of self-adaption. The algorithm was used to solve the network resource scheduling problems. The convergence speed was slow and the quality of the optimization solution needed to be improved because it adopted the more complex genetic algorithm. Multi-objective particle swarm optimization (MOPSO) algorithm is applied to the multi-objective optimization problems. The algorithm has the advantages of parallel computing, colony optimization, easy implementation and fast search.

But the main disadvantages in the application of MOPSO algorithm are: nondomination solution of distribution uniformity in the solution space is poorer, and the diversity of solution is lack. The global optimization ability is poor.

For this reason, based on Map-Reduce model, this article puts forward a resource scheduling method based on chaos multi-objective particle swarm optimization (CMOPSO) algorithm. Resource scheduling problem is converted to the multi-objective optimization problem with the QoS constraint. Through the optimization of multiple target parameters at the same time, eventually produce a set of satisfied constraint conditions of nondomination solutions. CMOPSO algorithm can effectively avoid falling into local optimum and has better global optimization ability compared with the algorithm MOPSO algorithm. And in the aspect of convergence and the diversity of solution set are improved, better able to resolve resource scheduling problem under the cloud computing environment.

2 The Resource Scheduling Model Based on Map-Reduce

Map-Reduce model is used in parallel computing of the large-scale data sets (greater than 1 TB).The Map-Reduce model of the structure of the framework as shown in Fig. 1, there are three characters[11] that are the user, the Master, the resource node in the model. The Master is the Master control program system, responsible for metadata organization, task scheduling, load balancing and fault tolerance, etc. Resource nodes receive tasks from the Master to do data processing and calculation. The user needs to realize the Map, Reduce functions and to control calculation. Map-Reduce operation process is as follows: the user submits a task that is divided into M subtasks of Map. Firstly, assign to the Map stage of resource nodes and produce R results to the local disk cache. Secondly, the results of the Map stage left as R Reduce subtasks that are submitted to Reduce phase of resource nodes to execute. Finally, the got final result is returned to the user. Master program realizes to schedule the M of Map subtasks and the R of Reduce subtasks in idle resource nodes.

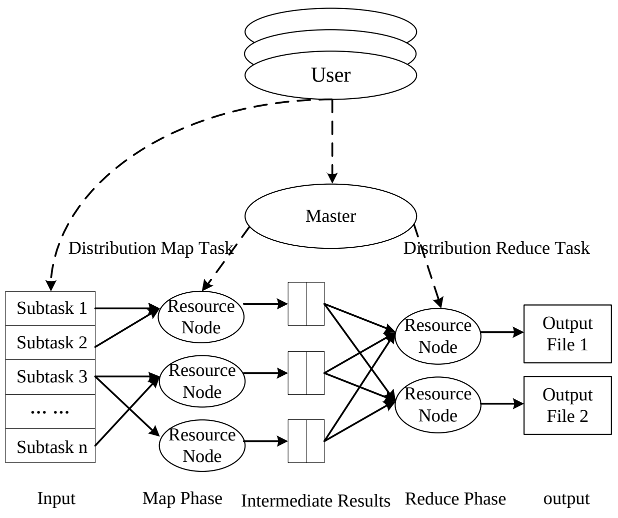


Fig. 1. Map-Reduce model structure

In reference [12], Map-Reduce resource scheduling is expressed as the following four groups:

$$M = (U, S, F, \theta)$$

Among them, the U is task set that composes of n user tasks; S is made up of m a resource nodes collection. The F is the cloud resource scheduling model of objective function. The θ is the intelligent optimization algorithms that solve the scheduling model. Scheduling model of the specific characteristics can be described as follows:

(1) The task set $U = (t_1, t_2, \cdots, t_n)$ is consist of n tasks submitted by the user. Among them the first t_i task can be broken up into $M(i)$ Map subtasks and $R(i)$ Reduce subtasks. So the U has sn subtasks in total. $sn = \sum_{i=1}^{n} (M(i) + R(i))$.

(2) The resource collection $S = (s_1, s_2, \cdots, s_m)$ is composed of m resources. Each resource is responsible for handling assigned to the resources on the subtasks.

(3) Using known predicting the execution time (ETC) matrix to record the task execution time, $ETC(i, j)$ is the execution time of the first i subtask on first j resource.

(4) With RCU array showing that the cost of each resource per unit time, $ETC(s)$ represents the first s resource cost per unit time.

You can get the execution time of the task t_i as the following type:

$$Time(t_i) = \max_{k=1}^{M(i)}(\sum_{u=1}^{Nom} WM(k,u)) + \max_{g=1}^{R(i)}(\sum_{v=1}^{Nor} WR(g,v)) \tag{1}$$

The *Nom* is arranged in the resource S to calculate the positions of the queue of the first *k* Map subtask. In front of it, resource S of the calculation queue has (*Nom-1*) subtasks. The WM (*k, u*) is the resource S by computing queue of the order to complete the first u subtask of the time from matrix ETC. When *u= Nom*, $WM(k,u) = WM(k,k)$ represents that resource S completes the first k subtask of the time. The $\sum_{k=1}^{Nom} WM(k,u)$ is the whole time that resources complete the first *k* Map subtask according to the calculation of the queue order. In the same way, the *Nor* represents that the first *g* Reduce subtask is arranged in the calculation of a resource in the queue and the $\sum_{g=1}^{Nor} WR(g,v)$ represents the whole time that the resources complete the first *g* Reduce subtask according to the calculation of the queue order.

Based on the above analysis, the whole task set is completed finally when the longest task *t'* is completed in the task set *U*. Therefore, all total tasks of the task set of the completion time T_1 is equal to the task *t'* of the execution time. It is:

$$T_1 = \max_{i=1}^{n}[\max_{k=1}^{M(i)}(\sum_{u=1}^{Nom} WM(k,u)) + \max_{g=1}^{R(i)}(\sum_{v=1}^{Nor} WR(g,v))] \tag{2}$$

The average task completion time T is:

$$T_2 = \frac{1}{n}T_1 = \frac{1}{n}(\max_{i=1}^{n} Time(t_i)) \tag{3}$$

It can be seen that when resources are full, with the increase of the task set *U*, the total time T_1 of task completion can not increase. The average completion time T_2 decreases instead even less than the shortest task execution time. It embodies the advantages of cloud computing.

The first *S* completes to assign to the resources needed for all the subtasks in cost for:

$$cost(s) = \sum_{i=1}^{Num(s)} ECT(i,s) \times RCT(s) \tag{4}$$

Among them, the *Mum*(s) represents the number of subtasks allocated for the resource *S*.

The total cost of all the tasks completed is as follows:

$$SC = \sum_{s=1}^{m} cost(s) \tag{5}$$

Cloud resource scheduling goal is for optimal matching task queue and resource nodes, to meet the specified QoS constraints, making more goals to achieve optimal resource scheduling scheme, belongs to the multi-objective optimization problem with constraints. This paper will make the total time T_1 of the tasks completed, the average time T_2 of the tasks completed, the total cost *SC* of the whole tasks completed to three goals and hopes to get the smallest T_1 、T_2 、*SC*. Resource reliability *Dep*, credibility *Rep* as two constraints, suggests that resources scheduling required by the minimum reliability and credibility. A multi-objective resource constrained scheduling model can be described as:

$$\begin{cases} MinT_1, \quad MinT_2, \quad MinSC \\ s.t. \quad Dep(s) \geq Dep_0 \\ \qquad\quad Rep(s) \geq Rep_0 \end{cases} \tag{6}$$

On the one hand, as a result of the practical application, many different kinds of optimization objectives and constraint condition, they can be as a multi-objective optimization problem of objective function or constraint condition, so that the model can be replicated in the objective function and constraint conditions.

On the other hand, due to different target having many characteristics, such as not commensurability and the contradiction between, may cause conflicts between the objective function, not satisfying multiple target optimization conditions at the same time. Therefore, the result of the multi-objective optimization problem is not a single solution, but subject to the constraints of a set of solutions[13].

Based on the above analysis, the Map-Reduce scheduling problem is two stages, the multi-objective optimization problem with constraints. So the scheduling algorithm not only should consider two stages process but also to the QoS constraints under the premise of multi-objective optimization at the same time. Therefore, based on MOPSO algorithm, according to the Map-Reduce scheduling problem, it puts forward CMOPSO algorithm for solving the problem.

3 Chaos Multi-objective Particle Swarm Optimization Algorithm

Particle swarm optimization (PSO) algorithm[13] is put forward earliest by James Kenndy and Russell Eberhart by the further study of birds community grazing. Because optimization algorithm has the advantages of parallel computing, colony optimization, easy implementation and fast search, it gets the wide attention of scholars both at home and abroad. Particle swarm optimization algorithm has been widely applied in many fields. MOPSO algorithm is put forward when the PSO algorithm is applied to the multi-objective optimization problems. The algorithm has set the particle position, velocity and adaptive value of three basic attributes. Each particle's position is on behalf of a feasible solution to the problem. The velocity of particles determines the location of the direction and rate of change, and adaptive value is determined by the objective function. Compare the merits of the particles according to the size of the adaptive value and screen out nondomination solution set from the initial particle swarm according to dominate the relationship. The basic flow of the algorithm is shown in Fig. 2.

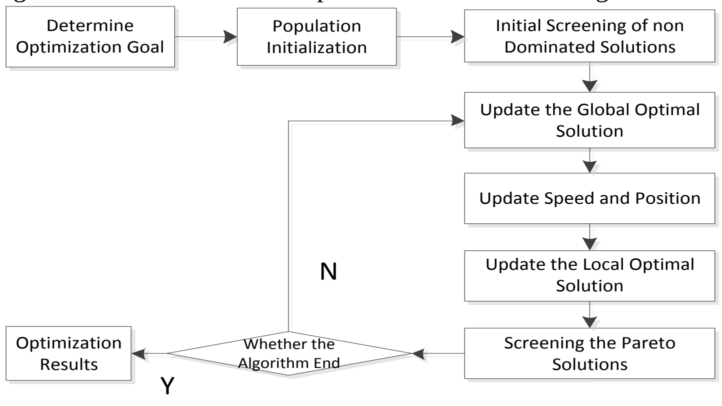

Determine Optimization Goal
Population Initialization
Initial Screening of non Dominated Solutions
Update the Global Optimal Solution
Update Speed and Position
Update the Local Optimal Solution
Screening the Pareto Solutions
Whether the Algorithm End
N
Y
Optimization Results

Fig. 2. Basic multi-objective particle swarms optimization process

The main disadvantages in the application of MOPSO algorithm are: nondomination solution of distribution uniformity in the solution space is poorer, and the diversity of solution is lack. The global optimization ability is poor. Aiming at these problems, the researchers improved MOPSO algorithm from different angles. Wei-Xing Zhang[13] improved MOPSO algorithm based on multiple population dynamics. The improved algorithm can carry on the fast track to transform Pareto frontier and improve the distribution of the optimal solution set. Sheng-Yu Pei, Yong-Quan Zhou[14] proposed to use chaotic mutation method to jump out of local optimal solution and enhanced the capacity of MOPSO algorithm of global optimization. But it did not give the calculation method of the fitness of particle population and used the roulette method to choose the optimal individual. The accuracy and efficiency needed to improve. The use time and concrete realization of chaotic disturbance were not given at the same time. The above documents are research foundation which the chaotic multi-objective particle swarm optimization algorithm was put forward.

The contributions of CMOPSO algorithm lie in: (1) the method of information entropy is used to calculate the particle fitness. (2) Use the Sigma iteration method to determine the optimal global position of the particle and improve the convergence speed. (3) The design decision for carrying out the chaos disturbance conditions and concrete form, help get rid of the bondage of the optimum local population.

3.1 The Initialization Particles

The particle's position variables are on behalf of the resource scheduling solution. In the Map-Reduce resource scheduling model, with *n* tasks, *m* resources, including task t_i is divided into $M(i)$ Map subtasks and $R(i)$ Reduce subtasks. To Map subtasks of the scheduling, scheme x_i can be expressed as $(x_{i1},\cdots,x_{im},\cdots,x_{iM})$, including x_{im} $(1\le m\le M,\ M=\sum_{i=1}^{n}M(i))$ being the first t_i mission of the first *j* Map subtasks number of resources. On all the Map subtasks sequence encoding, each code can be expressed as follows:

$$m[i,j]=j+\sum_{k=1}^{i-1}M(k) \tag{7}$$

Type of $m[i, j]$ is for the first t_i task of the first *j* Map of subtasks serial number. For example, there are five resources, three missions. The first task is divided into three Map subtasks, and the latter two tasks are divided into two Map subtasks. The number of subtasks is 7. The (2,3,5,1,4,3,1) is a solution of scheduling model and is the first task of the first Map subtasks performed on No.2 resource. The second task of the first Map of subtasks serial number is 4. It executes in the No.1 resource. The rest may be deduced by analogy, seen in Table 1.

Table 1. Code sample

Task Sequence Number	Subtasks Sequence Number	Executing Subtasks Number of Resources
1	1	2
	2	3
	3	5
2	4	1
	5	4
3	6	3
	7	1

The same method for Reduce of subtasks coding, Reduce of subtasks scheduling scheme can be expressed as $(x_{i1},\cdots,x_{ir},\cdots,x_{iR})$. Among them, $x_{ir}(1\le r\le R,\ R=\sum_{i=1}^{n}R(i))$ says the first t_i mission of the first j Reduce of subtasks resource number.

By the above method, the position of each particle in an n dimension vector to describe, for $n=sn=\sum_{i=1}^{n}(\ M(i)+R(i)\ )$, it is on behalf of sn subtasks. Variable scope for each dimension is the integer in [1, m], representing m resources, using the integer in [1, m] to number each resource. Assuming that the vector $\boldsymbol{x_i}=(x_{i1},\cdots x_{ij},\cdots,x_{in})$ for the case of first i particle's position, it represents the first i kind of resource scheduling scheme, including x_{ij} being first j subtask selecting Number x_{ij} of resource in sn subtasks.

Particle swarm initialization quality directly affects the algorithm of the effect. For the algorithm, the distribution of particles in the solution space after initialization is the more uniform and the result is better. The chaotic sequence has the typical characteristics of nonlinear and randomness. The chaotic sequence has been widely used in the optimization problem[15]. In this paper, using chaotic sequence, use a logistic function to structure chaotic variables and initialize the particle swarm.

$$\beta_{(k+1)j}=\mu\beta_{kj}(1-\beta_{kj}),k=1,2\cdots,\beta_j\in(0,1) \quad (8)$$

In type, the k being numbers of function iteration, β_{ij} is chaotic variables, on behalf of the particles in the first j dimension position of attribute values. When $\mu\ge 4$, the sequence constructed by Logistic equation is the chaotic sequence. Initialize m dimension random number $\beta_{ij}=Rand(\ )\ (j=1,2,...,m)$ in the interval [0,1]. For particle swarm of size N, complete $(N-1)$ times iteration based on the formula (8), namely $\beta_{(i+1)j}=\mu\beta_{ij}(1-\beta_{ij}),i\in[1,N-1],j\in[1,m]$, then use the formula (9) to map the β_{ij} as the particle's position, and take up the integer to get the position variable of the integer value.

$$x_{ij}=\lceil 1+(n_j-1)\beta_{ij}\rceil \quad (9)$$

3.2 Update the Particle Velocity and Position

In the algorithm of the iterative process, in the first t iteration the particle's velocity updating formula is as shown in (10) and position update formula is as shown in (11).

$$v_{ij}^{t+1}=wv_{ij}^{t}+c_1r_1(P_{ij}^{t}-x_{ij}^{t})+c_2r_2(P_{gj}^{t}-x_{ij}^{t}) \quad (10)$$

$$x_{ij}^{t+1}=\left\lceil\left|(x_{ij}^{t}+v_{ij}^{t+1})\%n_j\right|\right\rceil \quad (11)$$

The w is weight. The c_1, c_2 are the study variables and are usually 2. The r_1, r_2 are the random numbers on [0,1] interval. Set the maximum particle velocity component in different dimensions is $v_{\max}\ (v_{\max}>0)$. When $v_{ij}^{t}<-v_{max}$, $v_{ij}^{t}=v_{max}$ and $v_{ij}^{t}<-v_{max}$, $v_{ij}^{t}=-v_{max}$. Using formula (11) to update location, the location of each dimension variable first gets remainder for n_j, and then take absolute value to guarantee an non-negative result, finally rounding up.

3.3 Update and Maintain the Noncontrol Solution Set

In this paper, keep the diversity of the nondomination solution through the calculation of information entropy of particles. Assuming that F_i is the fitness of particle x_i, then the calculation steps are as follows:
Every particle of information entropy $H_{i,j}$ is

$$H_{i,j}=-P\times\ln P\ ,\ (i=1,2,...,N_s) \quad (12)$$

The N_s is a dominant solution concentration of the particle number. Get the degree of similarity between different particles $\gamma_{i,j}$ according to the information entropy of particles.

$$\gamma_{i,j}=1/(1+H_{i,j}) \quad (13)$$

The particle x_i is not in control of the density of the solution concentration $D_i = m / N_s$. The m represents the number of particles for the similar degree $\gamma_{i,j} \geq 0.9$ of particle x_i in the nondomination solution, then particle x_i of the fitness F_i is:

$$F_i = 1/D_i = N_s / m \tag{14}$$

By the calculation process, the number of similar individuals is inversely proportional to the individual fitness. If the solution set is in the descending order according to the fitness of particles, according to the preset threshold to remove the sorting of the particles, can guarantee the uniformity of the solution concentration of the particle distribution and the diversity of solution.

3.4 Choose the Optimal Position

In algorithm during operation, each iteration needs to determine the global optimal and local optimal location. This paper adopts the Sigma method proposed by Mostaghim[16] to determine the optimal global position of the particle. The method can effective improve the convergence rate of population.

Hypothesis optimization problem with two objective function f_1 and f_2, particle x_i in solution space, the two objective functions in the two-dimensional space of vector-valued are $(f_{1,i}, f_{2,i})$ and define that the particle of the Sigma value is α_i.

$$\alpha_i = \frac{f_{1,i}^2 - f_{2,i}^2}{f_{1,i}^2 + f_{2,i}^2} \tag{15}$$

When to extend multi-dimensional solution space, the Sigma value of α_i is one which has an element vector C_m^2. Among them, the m means that there are m optimization goals in the optimization. Every element of the vector $\overrightarrow{\alpha_i}$ in the solution space of all the possible combinations between any two coordinates is calculated according to the formula (16), such as the three objective functions corresponding to the three-dimensional space of f_1, f_1 and f_3. The Sigma value of $\overrightarrow{\alpha_i}$ can be expressed as follows:

$$\overrightarrow{\alpha_i} = \begin{pmatrix} f_{1,i}^2 - f_{2,i}^2 \\ f_{2,i}^2 - f_{3,i}^2 \\ f_{3,i}^2 - f_{1,i}^2 \end{pmatrix} / (f_{1,i}^2 + f_{2,i}^2 + f_{3,i}^2) \tag{16}$$

Determine the optimal global position of particle x_i consists of two steps: First, calculate the control all the particles in the solution of the Sigma value $\overrightarrow{\alpha_j}$ by the above method. Then calculate the particle x_i corresponding to Sigma value of $\overrightarrow{\alpha_j}$. Calculate $\overrightarrow{\alpha_i}$ and all particles of nondomination solution of the corresponding the Sigma value in the solution space distance, then the smallest distance of nondomination solution is the optimal global position. In the double objective corresponding two-dimensional space, the optimal global location to determine the particle in a population situation is shown in Fig. 3.

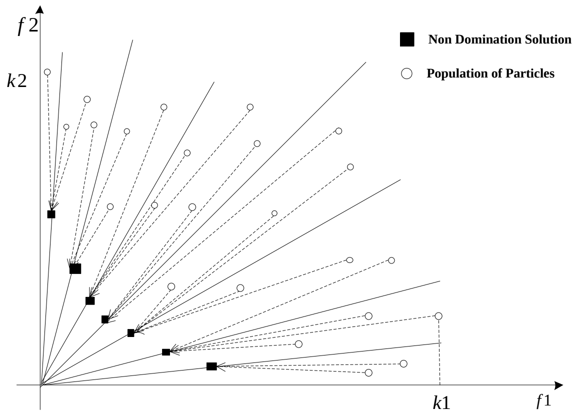


Fig. 3. Based on the sigma method to find the global optimal location map

When the objective function corresponds to the range of values at the difference, need to do each dimension in the solution space normalization processing. Assume that the range of values of the objective function to $[0, K_1]$, the corresponding function values for $f_1(x_i)$, then particle has normalized processing as follows:

$$f_1'(x_i) = f_1(x_i) / K_1 \tag{17}$$

Determine the local optimum position of the particles with domination relations. If the current position of the particles to P_i has domination relationship, use the current position instead of P_i . If P_i for the current location has dominated relations, the P_i remains unchanged. If these two domination relationships are not established, refer as the local optimum position of the particles from the current position and P_i in either.

3.5 Chaos Disturbance

When the diversity of nondomination solution set is poorer, the algorithm is easy to fall into the optimum local and lose a good global search capability. Therefore, use chaos theory through the chaos disturbance to help get rid of the bondage of local optimum population and improve the global optimization ability of the algorithm. Specific implementation process is as follows:

(1) Measure $\mathrm{div}(t)$ species diversity by the method of literature [17] proposed, and through the detection of $\mathrm{div}(t)$ is beyond the preset threshold to determine whether the population to a halt.

$$\mathrm{div}(t) = \frac{1}{N} \times \frac{1}{R} \times \sum_{i=1}^{n} \sqrt{\sum_{j=1}^{D} (x_{ij}^{t} - \overline{x_{j}^{t}})^2} \tag{18}$$

The t means population number of iterations. The N represents the number of particles in the population. The maximum radius R represents the search space. The x_{ij}^{t} says the first i particle on the first j dimension values when t iterations. The $\overline{x_j^t}$ says the population in the first j dimension on average. The diversity of threshold value is defined as $\eta \times div$, where $\eta = 0.4$. The theoretical calculation and experiment show that the population tends to halt[17].

(2) Will not dominate the descending order according to the fitness and take $N/2$ particles before to do the chaotic disturbance. Formulas (8) are used to get the $\beta_{ij}^{(h)}(h = 1, 2, \cdots H)$. The maximum of chaotic iterations is H.

(3) Through the chaos disturbance deviation $Bias(c_{ij})$ to control particle disturbance intensity, will affect the control of chaotic disturbance within a reasonable range.

$$Bias(c_{ij}) = \lceil \delta(H - h) Rand \rceil \tag{19}$$

Among them, the σ is the scale factor and h represents the current number of iterations and $Rand(\ )$ is to define the random numbers on [0,1] interval.

(4) Chaos perturbation is calculated according to the formula (20) load sequence after the particle's new position variables in every dimension. Every nondomination particle neighborhood produces H neighborhood particles $c_i^{(h)*} = (c_{i1}^{(h)*}, c_{i2}^{(h)*}, \cdots c_{im}^{(h)*}), (h = 1, 2, \cdots H)$. The c_i^* is known as newly generated particles after chaos disturbance from a selected the dominant particles. The result of *N/2* newly generated particles replaces with the new particle random population of particles.

$$c_{ij}^{(h)*} = c_{ij} - Bias(c_{ij}) + \lceil 2 Bias(c_{ij}) \beta_{ij}^{(h)} \rceil \tag{20}$$

After the above steps, given the particles in the solution space, by using chaos disturbance on the neighboring domain endogenous into many new particles, effectively enhance population diversity and help get rid of the bondage of the optimum local population, improving the global optimization ability of the algorithm.

3.6 Chaos Multi-objective Particle Swarm Optimization Algorithm Process

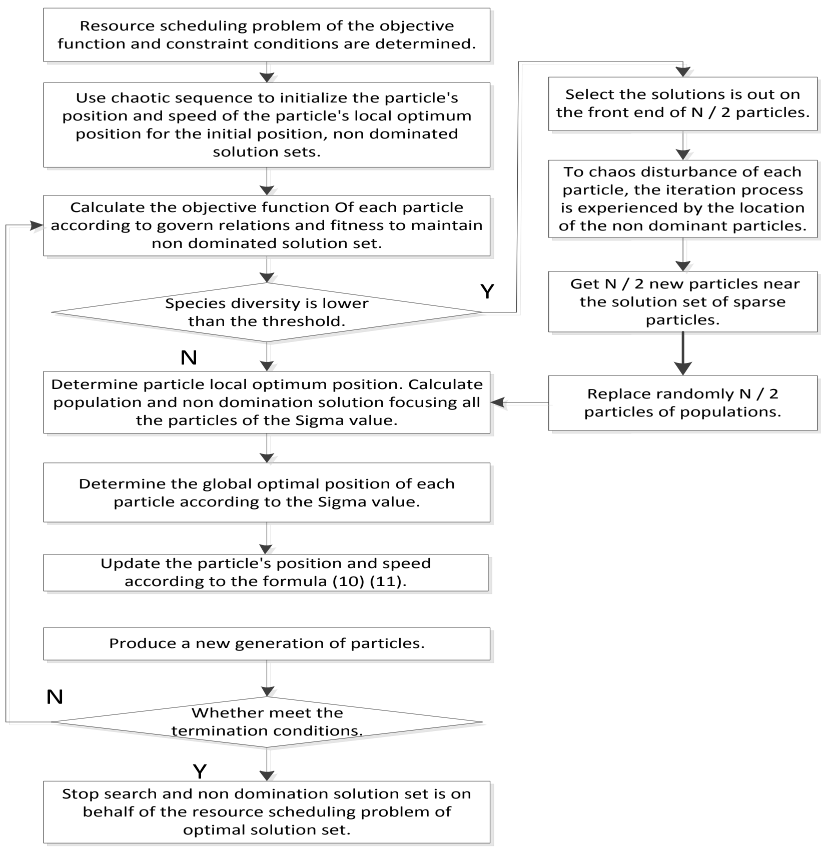

Resource scheduling problem of the objective function and constraint conditions are determined.
Use chaotic sequence to initialize the particle's position and speed of the particle's local optimum position for the initial position, non dominated solution sets.
Calculate the objective function Of each particle according to govern relations and fitness to maintain non dominated solution set.
Species diversity is lower than the threshold.
Y
N
Select the solutions is out on the front end of N / 2 particles.
To chaos disturbance of each particle, the iteration process is experienced by the location of the non dominant particles.
Get N / 2 new particles near the solution set of sparse particles.
Replace randomly N / 2 particles of populations.
Determine particle local optimum position. Calculate population and non domination solution focusing all the particles of the Sigma value.
Determine the global optimal position of each particle according to the Sigma value.
Update the particle's position and speed according to the formula (10) (11).
Produce a new generation of particles.
N
Whether meet the termination conditions.
Y
Stop search and non domination solution set is on behalf of the resource scheduling problem of optimal solution set.

Fig. 4. Algorithm process

4 Simulation Experiment and Analysis

4.1 Design of Experiment

This paper adopts cloud computing simulation platform CloudSim[18] in simulation experiment by extending the CloudSim DatacenterBroker. The simulation uses the proposed CMOPSO algorithm and traditional MOPSO algorithm comparison experiment with computer configuration for Core2 processors, 2 GB of memory, Windows XP operating system, algorithm implementation under Matlab2012. Resource scheduling model of minimum reliability R_0=0.1, the minimum credit = 2. CMOPSO algorithm parameter is set as follows, speed weight $N = 50$, population size $w = 0.75$, nondomination solution set $N_S = 50$, the chaos maximum iterative times H=10, proportional variable $\delta = 0.9$. Assume that the number of the tasks is 50. Each task is divided into 20 Map subtasks and 10 Reduce subtasks. Use China electronics association hosted the 2015 session of the Chinese WEB Service competition (China WEB Service Cup 2015) provided the test data of generator produces the experimental test set[19]. Due to the test set by the concept of number (CN) and mix Solution (Solution Depth) of the influence of two parameters, this paper experiment sets CN = 3000, Solution Depth = 6. Respectively test CMOPSO and MOPSO algorithm convergence speed and nondomination solution of distribution through the experiment.

4.2 Algorithm Convergence Contrast Experiment

Set some resources for 30. Fig. 5 shows two iterative convergence process of the algorithms. You can see that the two algorithms have convergence to the optimal solution of the trend. The iteration after 50 times, both algorithms achieve convergence completely. CMOPSO algorithm has done in 35 generation algorithm convergence, and MOPSO algorithm has done in 44 generation algorithm convergence. CMOPSO algorithm is obviously better than MOPSO algorithm on the convergence speed.

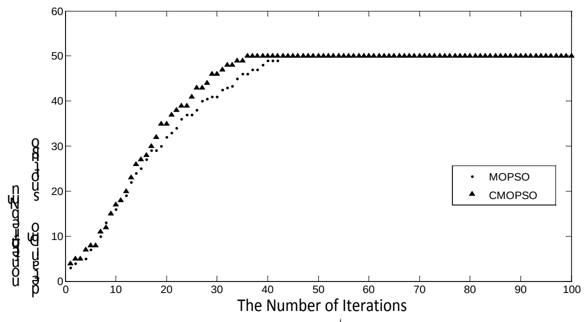


Fig. 5. Comparison of iteration convergence

To set the number of resources for 50, 70, 100 and 70, experimenting respectively, the results are shown in Table 2.

Table 2. Algorithm convergence under different number of resources

	Number of Resources	Number of Iterations Required for Convergence Algorithm	
		MOPSO Algorithm	CMOPSO Algorithm
Experiment 1	30	44	35
Experiment 2	50	54	41
Experiment 3	70	65	46
Experiment 4	100	78	53
Experiment 5	120	93	61

From Fig. 6, you can see that with the increase of a number of resources, to solve the space greatens, and two algorithms to achieve convergence increase the number of times. But CMOPSO algorithm of growth is slow and is better than the MOPSO algorithm convergence speed in all experiments.

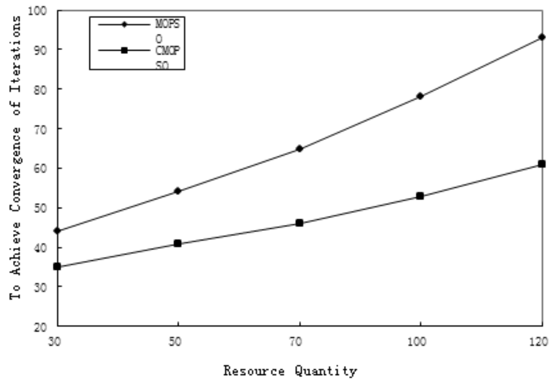
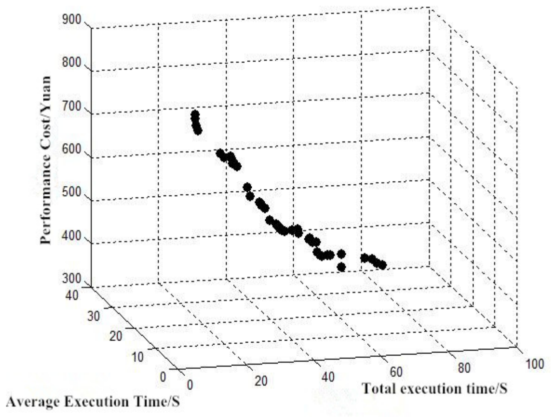


Fig. 6. Different resources size of the algorithm convergence speed

4.3 Distribution of Nondomination Solutions Contrast Test Algorithm

Set the number of resources for 30, Fig. 7 and Fig. 8 are two kinds of the distribution of nondomination solution algorithm. Seen by comparing, the chaos of multi-objective particle swarm optimization to search the solutions is the distribution uniformity of better, more diversity.



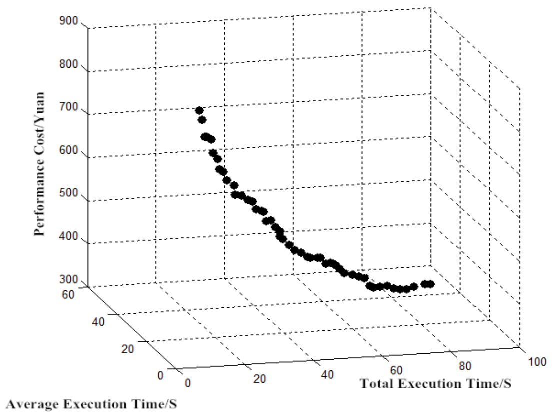


Fig. 7. MOPSO algorithm of distribution of domination solution

Fig. 8. CMOPSO algorithm of distribution of non-nondomination solution

To quantitatively evaluate the distribution uniformity of solutions, in literature [15], the definition of nondomination solutions of spacing S and maximum spread range D, the specific meaning of parameters in the formula can be found in the references.

$$S=\sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(\bar{d}-d_i)^2} \quad , \quad D=\sqrt{\sum_{p=1}^{3}(\max_{i=1}^{n} f_p^i-\min_{i=1}^{n} f_p^i)^2}$$

When the resource quantity respectively for 30, 50, 70, 100, 120, the experiment can get the largest spacing S and maximum spread range D of CMOPSO and MOPSO algorithm, the results are as shown in Table 3. It can be seen that CMOPSO algorithm gets the maximum distance smaller between the nondomination solution and the solution set of the whole spreading scope bigger.

Table 3. Algorithm convergence under different number of resources

	Number of Re-sources	Largest Spacing S		Maximum Dispersion Scope D	
		MOPSO	CMOPSO	MOPSO	CMOPSO
Experiment 1	30	0.213	0.098	851	1133
Experiment 2	50	0.301	0.071	877	1221
Experiment 3	70	0.195	0.082	1643	2012
Experiment 4	100	0.502	0.176	1462	1873
Experiment 5	120	0.275	0.184	1496	1520

Through the above analysis shows that chaotic multi-objective particle swarm optimization algorithm is global optimization ability stronger, the nondomination solution set the speed of convergence faster. Get better optimization solution of the diversity and distribution more uniform and can provide users with constraint conditions satisfied of the nondomination solution set. It is advantageous for the user according to the application requirements and practical limit to choose the appropriate optimization solution.

5 Conclusion

In order to achieve efficient resource scheduling in cloud computing environment and guarantee the quality of cloud services effectively, this article proposes a solution Based on chaos multi-objective particle swarm optimization algorithm. This solution is based on the Map-Reduce model and convert the resource scheduling problem to the constrained multi-objective optimization problems which with QoS. Algorithm keep the set of particle diversity by the calculation of information entropy and use the Sigma method to speed up the algorithm convergence speed. Algorithm can also generate new particles near the solution space of the sparse particles by introducing chaos disturbance mechanism and guide the population space evolution to advantage, in order to avoid the algorithm entering the local optimum. Experiments show that CMOPSO algorithm's convergence speed and the diversity of the nondomination solutions are significantly improved by compared with MOPSO algorithm. It is able to achieve more efficient scheduling in the cloud resources.

The resource scheduling which based on the cloud computing is divided into matching between tasks and virtual machine and the mapping between the virtual machine and host. This paper only considers the scheduling between tasks and virtual machines, so how to efficiently solve the problem of the virtual machine and host mapping is the key point of further research.

Acknowledgement

This work is supported by the National Nature Science Foundation of China under Grant No. 61303074 and No. 61309013.

References

[1] R. Buyya, et al., “Cloud computing and emerging IT platforms: vision, hype, and reality for delivering computing as the 5th utility,” *Future Generation Computer Systems*, Vol. 25, No. 6, pp. 599-616, 2009.
[2] M. Armbrust, et al., “A view of cloud computing,” *Communications of the ACM*, Vol. 53, No. 4, pp. 50-58, 2010.
[3] J. Luo, et at., “Cloud computing: architecture and key technologies,” *Journal on Communications*, Vol. 32, No. 7, pp. 3-21, 2011.
[4] P. Wang, “Key technology and application example of cloud computing,“ in *Proceeding of Science Press*, pp. 11-13, 2010.
[5] J. Dean, S. GHEMAWAT, "MapReduce: simplified data processing on large cluster,” in Proceeding *of the 6th Conference on Symposium on Operating System Design and Implementation*, pp. 137-150, 2004.
[6] What are some promising open-source alternatives to Hadoop MapReduce for map/reduce?: https://www.quora.com/What-are-some-promising-open-source-alternatives-to-Hadoop-MapReduce-for-map-reduce.
[7] C. Zhang et al., “Task allocation based on ant colony optimization in cloud computing,” Journal of Computer Applications, Vol. 32, No. 5, pp. 1418-1420, 2012.
[8] J. LI, J. Peng, “Task scheduling algorithm based on improved genetic algorithm in cloud computing environment,” *Journal of Computer Applications*, Vol. 31, No. 1, pp. 184-186, 2011.
[9] L. Sun, et al., “Service Selection of Network Simulation Task Community Based on Improved Particle Swarm Optimization Algorithm,” *Acta Armamentarii*, Vol. 33, No. 11, pp. 1393-1403, 2012.
[10] J. Liang, B. Xu, Y. Ge, “Map-Reduce job scheduling algorithm based on improved shuffled frog leaping strategy,” *Application Research of Computers*, Vol. 30, No.7, pp. 1999-2002, 2013.
[11] L. Lu, “Research of job scheduling algorithms under cloud computing environment,” *Nanjing University of Science and Technology*, 2013.
[12] D. Sun, et al., “Optimizing multi- dimensional QoS cloud resource scheduling by immune clonal with preference,” *Acta Electronica Sinica*, Vol. 39, No. 8, pp. 1824- 1831, 2011.
[13] W.-X. Zhang, “Dynamic multi-objective optimization algorithm of research and application based on particle swarm optimization algorithm,” *Zhengzhou University*, 2013.
[14] S.-Y. Pei，Y.-Q. Zhou, “A multi-objective particle swarm optimization algorithm based on the chaotic mutation,” *Journal of Shandong University(Natural Science)*, Vol. 45, No. 7, pp. 18-23, 2010.
[15] J. Zheng, “Multi-objective evolution algorithm and its applications,” in *Proceeding of Science Press*, pp. 4-6, 2007.
[16] S. Mostaghim S, J. Teich, “Strategies for finding good local guides in multi-objective particle swarm optimization,” in *Proceeding of the 2003 IEEE Swarm Intelligence Symposium Indianapolis*, p.. 26-33, 2003.
[17] F. Yi-Qiu, W. Fei, G. Jun-Wei, “A task scheduling algorithm based on load balancing in cloud computing,” in *Proceedings of the 2th International Conference on Web Information Systems and Mining*, pp. 156-162, 2010.
[18] The Cloud Lab. Cloudsim: http://www.cloudbus.org/cloudsim.
[19] 2015 China Web Service Cup. Experiment Data Lab: http://debs.ict.ac.cn/2015.
[20] J. Song, et al., “An energy-efficiency optimized resource ratio model for MapReduce,” *Chinese Journal of Computers*, Vol. 38, No. 1, pp. 59-73, 2015.
[21] Y.-H. Zhong, “Based on cloud computing key technology research and application of virtualization,” *Beijing university of posts and telecommunications*, 2014.
[22] Y. Ai-Ping, W. Can-Jun, “Virtual machine deployment strategy based on improved genetic algorithm in cloud computing environment,” *Journal of Computer Applications*, Vol. 34, No. 2, pp. 357-359, 2014.
[23] H.-L. Xue, “Analysis of Cloud Computing Architecture and Its Core Technology,” *Intelligent Computer and Applications*, Vol. 4, pp. 63-67, 2014.