International Journal of Grid and Distributed Computing
Vol. 9, No. 9 (2016), pp.269-280
http://dx.doi.org/10.14257/ijgdc.2016.9.9.23

Application of PSO Algorithm Based on Improved Accelerating Convergence in Task Scheduling of Cloud Computing Environment

Zhulin Li[1,2*], Cuirong Wang[3], Haiyan Lv[4] and Tongyu Xu[5]

1College of information science and engineering, Northeastern University, Shenyang, Liaoning, China
2Modern educational technology center, Shenyang Agricultural University, Shenyang, Liaoning, China
3College of computer and communication engineering, Northeastern University at Qinhuangdao, Qinhuangdao, Hebei, China
4College of forestry, Shenyang Agricultural University, Shenyang, Liaoning, China
5College of information and electrical engineering, Shenyang Agricultural University, Shenyang, Liaoning, China
Corresponding author's email: zhulin@syau.edu.cn

Abstract

Hadoop uses a reliable, efficient and scalable way to process data. It provides a good solution for dealing with big data. The task scheduler is the core component of Hadoop, and it is responsible for the managing and allocating the cluster resources. Therefore, scheduling algorithm directly affects the overall performance of Hadoop platform and utilization of cluster resource. Based on this, the improved accelerate particle swarm algorithm (IAPSO) is introduced to the cloud environment, and to solve the cloud task scheduling problem in this article. When we use particle swarm algorithm for task scheduling, the tasks are considered as particles, the resource pool is seen as the search space, and the process of finding the optimal solution is considered as a process of task scheduling. If all the sub tasks find the appropriate resources, then stop the iteration and allocate sub asks to the resource nodes. Finally, we simulate the experiment by using CloudSim software. When a single type of task is committed, our algorithm and the other three algorithms can also be used to complete the task scheduling process, and our algorithm is more efficient. But in practice, the cloud computing environment is facing multiuser, and the types of tasks are also varied. With the increase in the number of tasks, the advantage of the other three algorithms decreases gradually, but algorithm in this paper has been exhibited higher efficiency. In addition, with the increase of the number of nodes, task completed time of the algorithm in this paper is significantly less than the other three algorithms, and it has a steady downward trend. Therefore, IAPSO algorithm which is proposed in this paper is applied to solve task scheduling problem in the cloud environment, and it can effectively improve the efficiency of task scheduling.

***Keywords**: cloud computing, Hadoop, particle swarm optimization, accelerating convergence, task scheduling*

1. Introduction

With the development of computer technology, people are more and more required to compute capacity, storage capacity, transmission capacity and interactive ability of computer, and the traditional computing model cannot fully meet the needs of users at the present stage. Cloud computing is a new application mode of network, and it is also a kind of new service based on the resource supply model. Hadoop is an important platform for cloud computing. It uses a reliable, efficient and scalable way to process data, and it provides a good solution for dealing with big data. The task scheduler is the core component of Hadoop, and it is responsible for the managing and allocating the cluster resources. Therefore, the scheduling algorithm directly affects the overall performance of Hadoop platform and utilization of cluster resource [1-3].

A good task scheduling strategy of cloud computing should be able to adapt to different cloud computing environment, as well as the number of tasks and the nature. Particle swarm optimization algorithm is very suitable to solve the problem of cloud computing task scheduling. But the traditional PSO algorithm also has many deficiencies, scholars have improved the traditional particle swarm algorithm in these years. In order to improve the diversity of particles, and expand the search space, Bergh proposes the PSO multi-start algorithm. His algorithm have retained the historical best particle and have initialized all the particles [4]. A particle swarm optimization algorithm with dynamic inertia weight is proposed by Jiao. In his algorithm, the greater the number of iterations, the smaller the inertia weight [5]. In the literature [6], the chaos algorithm is introduced into the particle swarm algorithm. Based on the traditional PSO algorithm, Feng Liangliang proposes an improved algorithm. They add a fitness function to consider the total task completion time and the cost of task [7]. Shen Kaitao uses natural number encoding to represent the position of the particle, and puts forward the adjustment method of an adaptive inertia weight factor [8]. Li Jingmei proposes a heterogeneous multiprocessor task scheduling algorithm based on particle swarm optimization. The algorithm uses the integer matrix to encode the particle, and defines the exchange operation to update the particle. They realize the mapping from the search space to the discrete space [9]. In addition to the PSO algorithm, the scholars using simulated annealing (SA) and genetic algorithm (GA) in the field of task scheduling [10-12]. GA uses the idea of biological evolution to solve the optimization problem through the competition strategy of survival of the fittest. Vincenzo introduces a kind of grid computing task scheduling algorithm based on genetic algorithm [13]. Abraham introduces the application of simulated annealing algorithm in grid computing task scheduling [14]. Although GA has a strong global search performance, it is easy to produce premature convergence problem in practical application, and the search efficiency is low during anaphase. SA algorithm is originated from statistical physics method, and it is first introduced by Kirkpatric. SA algorithm has a good local search ability, and it has a strong dependence on the parameters.

To sum up, the improved accelerate particle swarm algorithm (IAPSO) is introduced to the cloud environment, and to solve the cloud task scheduling problem. When we use particle swarm algorithm for task scheduling, the tasks are considered as particles, the resource pool is seen as the search space, and the process of finding the optimal solution is considered as a process of task scheduling. If all the sub tasks find the appropriate resources, then stop the iteration and allocate sub asks to the resource nodes. Finally, we simulate the experiment by using CloudSim software.

2. Task Scheduling in Hadoop Architecture

At present, the cloud computing environment is mostly based on Hadoop. Hadoop can run the application with a large number of hardware cluster. It provides a stable set of interfaces for an application. In this way, a distributed system with high reliability and high expansibility is constructed. Hadoop has advantages of high scalability, open source, high efficiency and high reliability. These advantages prompted it become the main choice to build the cloud computing environment. Hadoop mainly includes two sub projects which are Hadoop distributed file system (HDFS) and MapReduce calculation model.

2. 1. Hadoop Distributed File System

Compared with the existing distributed file system, HDFS has three characteristics. The first is that the hardware failure is regarded as the norm rather than the exception, so it has a high fault tolerance. The second is the HDFS using the data stream access technology, so it has a high data throughput rate. The third is that HDFS is well suited to be deployed on the cheap resource cluster. HDFS uses the typical master-slave structure. The master is composed of the NameNode and JobTracker, and slaves is composed of the TaskTracker and DataNodes. A HDFS cluster usually contains a NameNode and multiple DataNodes. The NameNode is responsible for managing client access master server, monitoring requests and processing the requests. It plays a role of arbiter in cluster. Datanodes is responsible for storing data and sending requests to NameNode, JobTracker is responsible for global scheduling of tasks, and TaskTracker is responsible for the implementing the task. Hadoop is mainly contains two subsystems, they are Hadoop distributed file system and MapReduce model. The overall deployment structure of the Hadoop cluster is shown in Figure 1

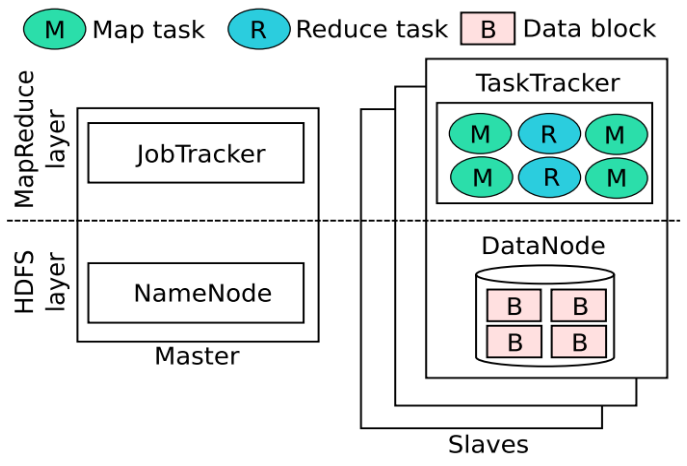

Figure 1. The Overall Deployment Structure of the Hadoop Cluster

2.2. Mapreduce Model

MapReduce is a distributed programming model, it expresses the large distributed computing to the serial computation of the set of key/value which is abbreviated to KV. With the help of the computer cluster, MapReduce performs tasks which are distributed to the node computers in the cluster. A MapReduce calculation consists of two main stages, the Map phase and the Reduce phase. The input of the calculation is a KV data set. There are five steps in the operation of MapReduce, as shown in Figure 2.

Step1: Input file.

Divide the input files, and then start the multiple backup of the user program on the cluster.

Step2: Allocate files to multiple worker for parallel execution.

Select one backup from the program backup as master, and the rest as slaves. Then, the task is divided into Map tasks and Reduce tasks. There are M Map tasks and R Reduce tasks. Master allocates a Map task (or a Reduce task) to the idle Slaves. Slaves read the required data, extract the KV, and transmit the data to the user. Then, the user defines the Map function.

Step3: Write the middle file.

Periodically write the KV which are in the cache to disk. Using block function, Slaves divide the data into R blocks, and transmit the location information to Master.

Step4: Perform the Reduce task.

Based on the location information data received from the Master, the Slaves transfer the data from the remote connection. After sorting the intermediate data, Slaves put keys together that have the same value. Then, it execute Reduce function, and write the final results.

Step5: Output the final results.

Master wake up the user program and return the results of the operation.

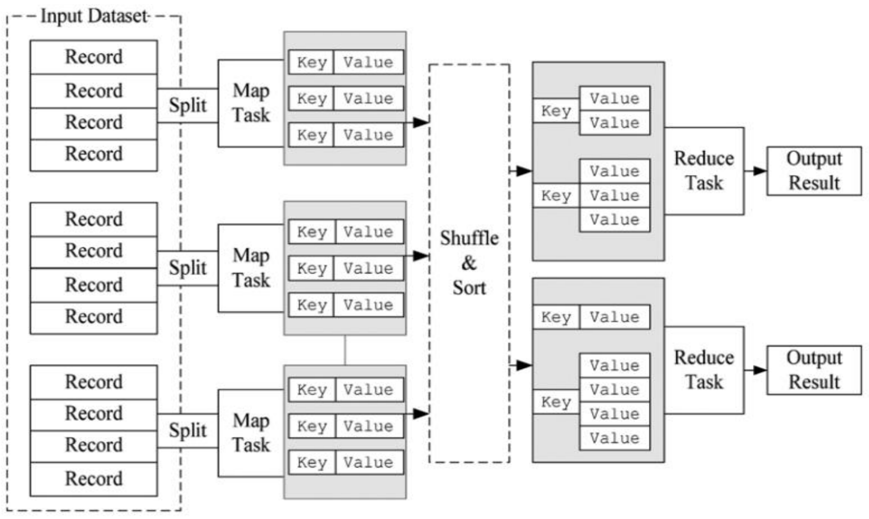

Figure 2. Step Diagram of MapReduce

3. The Standard PSO Algorithm

3.1. The Architecture of PSO

Particle swarm optimization algorithm is a kind of intelligent optimization method proposed by Kennedy and Eberhart[15]. The algorithm has the advantages of simple modeling, fast convergence, easy to implement, and so on. Therefore, it is not only has achieved remarkable results in solving combinatorial optimization problems, neural network training, pattern classification, fuzzy system control and other traditional optimization problem[16-17], but also has been widely used in the field of communication, sensor networks, optimal path, resource allocation, molecular biology research [18-20].

Next, we introduce the basic idea of the standard PSO algorithm and the process of finding the optimal solution of the particle. Firstly, initializing a group of particles that set the initial speed and position to each particle. Then, calculating each particle's fitness value by using the fitness function, so as to determine the virtues or defect degree of the current position of the particle. Secondly, comparing the virtues or defect degree of the current position of the particle and the best position of the particle. If the current position is better than the best position of the particle, the current position of the particle is set to the optimal position of the particle. Third, to determine whether the global optimal position of population is the best of the best position of all particles. If it is not, modify the global optimal position of population. Finally, updating the speed and position of each particle to determine if the end condition is satisfied. If satisfied, the algorithm ends. If not satisfied, continue to repeat the process of iteration. The execution process is shown in Figure 3.

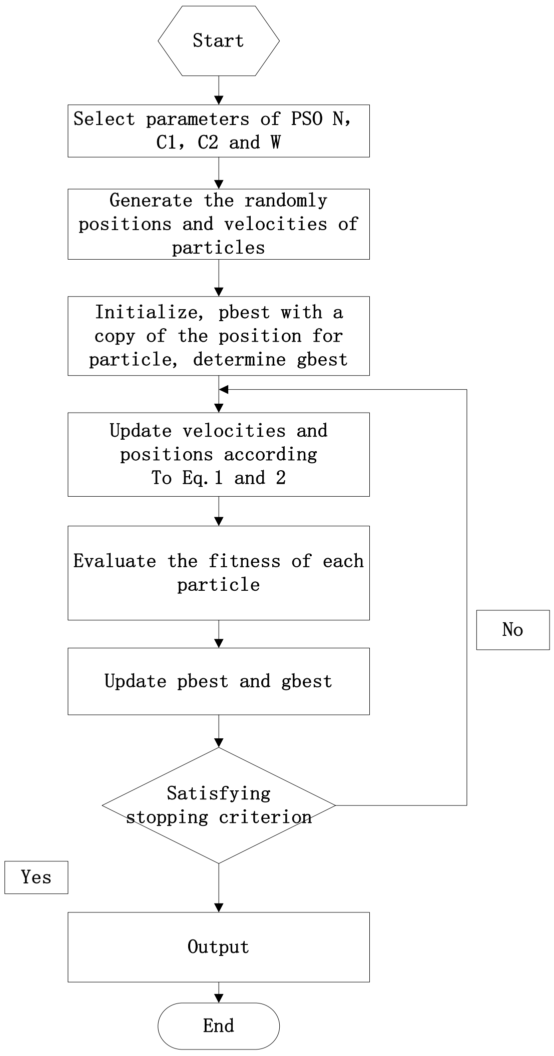
Start
Select parameters of PSO N, C1, C2 and W
Generate the randomly positions and velocities of particles
Initialize, pbest with a copy of the position for particle, determine gbest
Update velocities and positions according To Eq.1 and 2
Evaluate the fitness of each particle
Update pbest and gbest
Satisfying stopping criterion
No
Yes
Output
End

Figure 3. The Execution Process of Standard PSO Algorithm

3.2. The Basic Formula of PSO and its Improved Form

There are m particles that form a group in a D-dimensional search space, and the particle swarm optimization algorithm can be described as follow. The position of the i th particle is indicated by the $X_i=(x_{i1},x_{i2},\cdots,x_{id})$, $(i=1,2,\cdots,m)$, the $V_i=(v_{i1},v_{i2},\cdots,v_{id})$ represents the speed of the i th particle, the p_{best} , $i=(p_{i1},p_{i2},\cdots,p_{id})$ is the optimal location for the i th particle, and the $g_{best}=(g_1,g_2,\cdots,g_d)$ is the optimal location for all particles in the group. To follow these two optimal values, the particle is updated by the formula (1) and (2) respectively, and the speed and position of the particles are updated to meet the conditions of the end of the iteration.

$$v_{id}^{k+1}=\omega^k v_{id}^k+c_1 r_1(p_{id}^k-x_{id}^k)+c_2 r_2(g_d^k-x_{id}^k) \tag{1}$$

$$x_{id}^{k+1}=x_{id}^k+v_{id}^k \tag{2}$$

Among them, c_1 and c_2 are learning factors, r_1 and r_2 are random numbers in the interval of (0,1). The ω is the inertia weight, which plays an important role in coordinating the global and local search ability of particles. Larger ω is helpful for the global search of the particles, the smaller ω is helpful for the local search of the particles. The standard PSO algorithm has its own limitations.

Such as the algorithm implementation process and parameter values have a greater relationship; when the algorithm is applied to the high dimensional complex optimization problems, it is easy to occur premature convergence; the convergence speed of the algorithm is obviously slow when the algorithm is near or in the optimal solution. The convergence speed of PSO algorithm is faster during prophase of iteration. But during anaphase of iteration, when the algorithm converges to a local minimum, the local search speed becomes slow. The accelerated particle swarm optimization algorithm can effectively solve this problem.

Next, we introduce an improved particle swarm optimization algorithm(APSO). The accelerated particle swarm optimization uses only the global best position to update the velocity v_{id}^{k}, and it is due to the fact that individual best is used to increase the diversity in the quality solutions. Therefore, the velocity of the i th particle is updated as follows:

$$v_{id}^{k+1} = v_{id}^{k} + c_1\varepsilon + c_2(g_d^k - x_{id}^k) \tag{3}$$

Where, ε is a random value uniformly distributed in the range of 0-1. The position of the i th particle is update as follow equation:

$$x_{id}^{k+1} = (1-c_2)x_{id}^{k} + c_1\varepsilon + c_2 g_d^k \tag{4}$$

The APSO uses 2 parameters which are c_1 and c_2 as described in the above paragraph. On the basis of the APSO algorithm in this article, we introduce the third parameter variable which is θ. This variable can combine with the other two variables to form a new solution of c_1 and c_2. The function $c_1(\theta)$ is used to control the particles exploration of the search space. The mathematical expression of $c_1(\theta)$ is as follows:

$$c_1(\theta) = c_{1\max} - \frac{c_{1\max} - c_{1\min}}{\theta_{\max}}\theta \tag{5}$$

Where, the $\theta_{\max}$ is the maximum iteration number.

In addition, the function $c_2(\theta)$ is considered as an increasing function which is expressed as follow:

$$c_2(\theta) = c_{2\max} - (c_{2\max} - c_{2\min})\tan(\frac{\Pi}{2} \times \frac{\theta}{\theta_{\max}}) \tag{6}$$

Where, the $c_{2\min}$ and $c_{2\max}$ are respectively minimal and maximal values of $c_2(\theta)$ at first and last iterations. The function $c_2(\theta)$ make balance between global and local explorations during the search process.

With formula 5 and 6, we can update the position equation and the velocity equation of the particle as follows:

$$v_{id}^{k+1} = v_{id}^{k} + c_1(\theta)\varepsilon + c_2(\theta)(g_d^k - x_{id}^k) \tag{7}$$

$$x_{id}^{k+1} = (1-c_2(\theta))x_{id}^{k} + c_1(\theta)\varepsilon + c_2(\theta)g_d^k \tag{8}$$

4. The Cloud Task Scheduling Model Based on Hadoop is Constructed by Using PSO Algorithm

In this paper, the improved accelerate particle swarm algorithm (IAPSO) is introduced to the cloud environment, and to solve the cloud task scheduling problem. For example, we use particle swarm algorithm for task scheduling, tasks are considered as particles, the resource pool is seen as the search space, and the process of finding the optimal solution is considered as a process of task scheduling. In order to set the initial position and initial velocity of task in the resource pool, we can iteratively update the velocity and position of particles, and form a mapping relationship between tasks and resources.

Based on improved acceleration PSO algorithm in a cloud computing environment, the specific steps of cloud task scheduling can be divided into the following five steps:

Step1: Receive the tasks submitted by the user, and divides each task into several sub tasks.

Step2: Give the initial location and velocity information for each sub task in the resource pool. The current position of sub tasks is its best position. According to the value of the fitness function, we can get the global optimal position of the initial time.

Step3: Use the formula 7 to update the velocity information for each sub task.

Step4: Use the formula 8 to update the position information for each sub task.

Step5: Determine whether to meet the end conditions. If all the sub tasks find the appropriate resources, then stop the iteration and allocate sub asks to the resource nodes. If the number of iterations is more than the maximum number, then end the search, Re-initialize the particle swarm, and repeat step 2, 3 and 4. If the above two conditions are not satisfied, then continue iteration, and repeat step 3 and 4.

5. The Simulation and Result Analysis

Application services in the cloud computing has a complex configuration, composition, structure, and deployment requirements. In different systems and user requirements, use the same way to achieve the evaluate cloud allocation strategy, application engineering mode and resource execution mode is difficult. In order to overcome this challenge, the cloud computing simulation software (CloudSim) is a scalable simulation toolkit that can be used for modeling and simulating the cloud computing system and application configuration. The Cloudsim toolkit supports modeling behavior and system components, such as data center, virtual machine, resource allocation and task scheduling methods.

5.1 Task Type Setting

Through the Cloudlet class to create a cloud task, we set each task of the number, length, the size of the input and output, and create the task contains a list of 12 different types. The specific type is shown in table 1.

Table 1. The Cloud Task Type

type	Length	input size	output size
A	500	200	250
B	1000	800	400
C	2000	1500	600
D	3500	3000	2000
E	4500	3000	1000
F	5000	4500	2000
G	6500	4500	3000
H	7000	6000	4000
I	8000	7500	5000
J	9000	8000	6500
K	10000	10000	6000

In this paper, we create a list of 12 different types of cloud tasks by using the above types. We use the MapReduce model to divide each task into several sub tasks. Next, we begin a follow-up experiment.

5.2. The Simulation

According to the above experimental data, this paper uses GA algorithm, PSO algorithm, simulation annealing algorithm and the improved accelerated particle swarm algorithm to complete the process of task scheduling. The parameters of the four algorithms are set as follows.

GA algorithm: initial population number is 100, crossover probability is 0.8, mutation probability is 0.05, and the iteration number is set to 500.

SA algorithm: the initial temperature is 50, the cooling step is 1, cooling coefficient is 0.5, the step is 5 before cooling, the number of iterations is set to 500.

PSO algorithm: initial population number is 100, $c_1 = c_2 = 2$, the number of iterations is 500.

IAPSO algorithm: the number of initial population is 100, each iteration produces 50 new particles. According to the formula (5) and (6), we can obtain the value $c_1(\theta)$ and $c_2(\theta)$, and the number of iterations set to 500.

Experiment A:

In the cloud task table, the number of the tasks are 50, and the four algorithms run with the same type of cloud task. This experiment summits a small number of cloud tasks that only have a single task type.

Experiment B:

In the cloud task table, the number of the tasks are 200, and the four algorithms run with the same type of cloud task. This experiment summits a large number of cloud tasks that only have a single task type.

Experiment C:

In the cloud task table, the number of the tasks are 200, and the four algorithms run with the different types of cloud task. This experiment summits a large number of cloud tasks that consists of 12 different types in Table 1.

The experimental results are shown in Figure 4,5 and 6

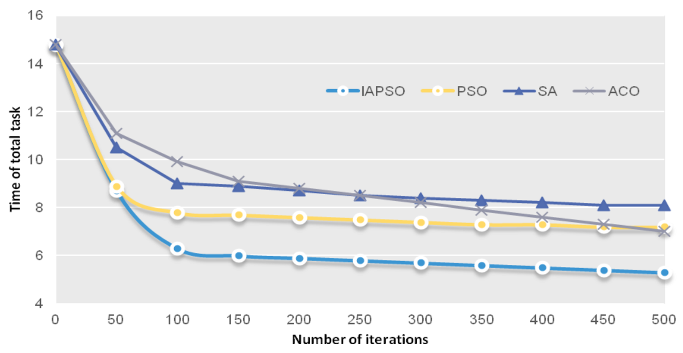

Figure 4. Experiment A

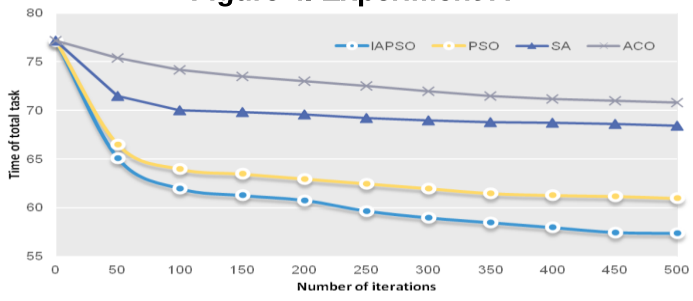

Figure 5. Experiment B

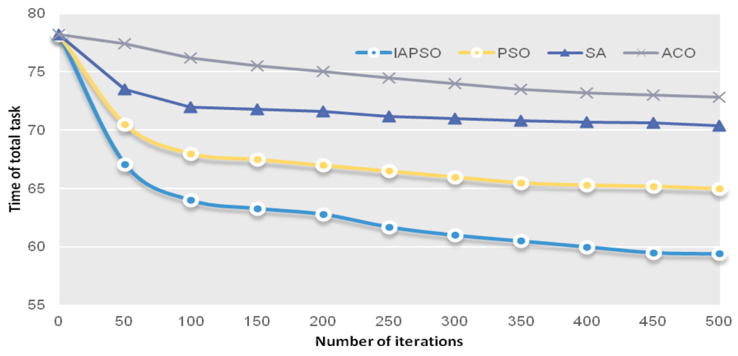

Figure 6. Experiment C

From experimental results of Figure 4, 5 and 6, we can see that the task scheduling optimization effect based on the improved accelerated PSO algorithm is significantly better than the rest of the three kinds of algorithm, the total task time of IAPSO is less than the other three algorithms. In Figure 4, when the number of iterations is less than 100, the PSO algorithm is better than the ACO algorithm. However, when the number of iterations is greater than 400, the ACO algorithm is better than the PSO algorithm. This is caused by the characteristics of PSO, which has the characteristics of fast convergence during prophase and lack of local search capability during anaphase. In Figure 4, when a single type of task is committed, our algorithm and the other three algorithms can also be used to complete the task scheduling process, and our algorithm is more efficient. But in practice, the cloud computing environment is facing multiuser, and the types of tasks are also varied. Through the comparison of Figure 4 and Figure 5, we know that when the tasks are various types, the total task time of all the algorithms will increase. From Figure 6 we can see that with the increase in the number of tasks, the advantage of the other three algorithms decreases gradually, and algorithm in this paper has been exhibited higher efficiency. At this point, we can find that the other three algorithms cannot adapt to the multiuser and multitype environment.

Experiment D:

In the different number of Hadoop nodes, the experiment 3's total task time is shown in Figure 5:

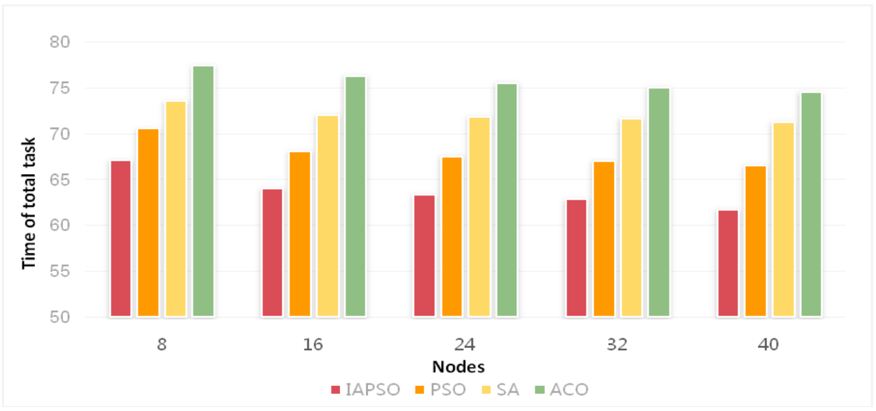

Figure 7. Experiment D

Because of the time consuming of communication, when we use a computer to perform a task, four models based on the MapRedue algorithm are more time consuming. However, with the increasing number of computers, we can see the task completion time is significantly decreased. When we use the 6 or 8 computers, each of them have the 8 Hadoop nodes, the task completion time is basically stable. This is related to the input size of the task. When the number of computer already meet the task allocation of MapReduce, increase the computer had no significant effect on the total completion time.

6. Conclusion

In this article, the IAPSO algorithm is introduced to the cloud environment, and to solve the cloud task scheduling problem. When we use particle swarm algorithm for task scheduling, the tasks are considered as particles, the resource pool is seen as the search space, and the process of finding the optimal solution is considered as a process of task scheduling.

The conclusion of this paper is as follows. When a single type of task is committed, our algorithm and the other three algorithms can also be used to complete the task scheduling process, and our algorithm is more efficient. But in practice, the cloud computing environment is facing multiuser, and the types of tasks are also varied. With the increase in the number of tasks, the advantage of the other three algorithms decreases gradually, and algorithm in this paper has been exhibited higher efficiency. In addition, with the increase in the number of nodes, task completed time of the algorithm in this paper is significantly less than the other three algorithms, and it has a steady downward trend. Therefore, IAPSO algorithm which is proposed in this paper is applied to solve task scheduling problem in the cloud environment, and it can effectively improve the efficiency of task scheduling.

Acknowledgments

Liaoning Information Platform of Rural Science and Technology Service and the University of Agricultural Technology Extension Demonstration of Research. Science and Technology Projects of Liaoning Province. 2014104017.

References

[1] Liu Peng. Cloud computing [M]. Beijing: Publishing House of electronics industry, 2007.
[2] Wu Hao. Task scheduling algorithm in the cloud environment[D]. Jiangsu: Nanjing University of Posts and Telecommunications, 2013.
[3] Shi Hengliang. The research on cloud computing task scheduling[D]. Jiangsu: Nanjing University of Science and Technology, 2012.
[4] BERGH F. An Analysis of Particle Swarm Optimizers[D]. Department of Computer Science, University of Pretoria, South Africa, 2006.118-123.
[5] JIAO B, LIAN Z G, GU X S. A dynamic inertia weight particle swarm optimization algorithm[J]. Chaos, Solitons & Fractals, 2008, 37(3):698-705.
[6] MENG H J, ZHENG P, WU R Y, et al. A hybrid particle swarm algorithm with embedded chaotic search[C]. Proceedings of the 2004IEEE Conference on Cybernetics and Intelligent Systems. Singapore, 2004. 367-371.
[7] Feng Liangliang, Zhang Tao, Jia Zhenhong. Cloud computing environment based on particle swarm optimization task scheduling algorithm [J]. Computer engineering, 2013, 39 (5): 183-186.
[8] Shen Kaitao, Hu de min. Based on cloud computing and improved discrete particle swarm optimization task scheduling[J]. Computer measurement and control, 2012, 20 (11) 3070-3072.
[9] Zhang Bo, Li Jingmei. A task scheduling algorithm of particle swarm optimization of heterogeneous multiprocessor[J]. Mini micro systems, 2013, 34 (5): 1154-1157.
[10] Wei Zhicheng, Yang Lianxiang, Zhou jiliu. Genetic algorithm optimization neural network topology and weight [J]. Journal of Guangxi Normal University: Natural Science Edition, 2003, 21 (S1): 62-65.
[11] Zhang Liyi, Wang Hongbin. Multi user detection of neural network based on genetic algorithm optimization [J]. Computer Engineering, 2011 (7): 207-209.
[12] Zhou Li, Huang Suzhen. Application of hybrid genetic algorithm based on simulated annealing of [J]. Computer, 2005 (9): 72-76.
[13] VINCENZOD M, MILILOTTIM. Sub-optimal scheduling in a grid using genetic algorithm [J]. Parallel Computing, 2004, 30(5/6): 553-565.
[14] ABRCHAM A, BUYYA R. Nature's heuristics for scheduling jobs on computational grids [C]. Advanced Computing and Communications. Cochin, India: Springer, 2000.
[15] KENNEDY J, EBERHART R C. Particle swarm optimization[C]. Proc of the First IEEE International Conference on Neural Networks. Perth, Australia: IEEE Press, 1995. 1942-1948.

[16] MODARES H, ALFI A, NAGHIBI-SISTANI M B. Parameter estimation of bilinear systems based on an adaptive particle swarm optimization[J]. Engineering Applications of Artificial Intelligence, 2010, 23(7): 1105-1111.
[17] KARAKUZU C. Parameter tuning of fuzzy sliding mode controller using particle swarm optimization[J]. International Journal of Innovative Computing, Information and Control, 2010, 6(10):4755-4770.
[18] KULKARNI R V, VENAYAGAMOORTHY G K. Bio-inspired algorithms for autonomous deployment and localization of sensor nodes[J].IEEE Transactions on Systems, Man, and Cybernetics, 2010, 40(6):663-675.
[19] ZHANG W, LIU J, NIU Y Q. Quantitative prediction of MHC-II binding affinity using particle swarm optimization[J]. Artificial Intelligence in Medicine, 2010, 50(2):127-132.
[20] GHEITANCHI S, ALI F, STIPIDIS E. Particle swarm optimization for adaptive resource allocation in communication networks[J]. EURASIP Journal on Wireless Communications and Networking, 2010. 1-13.

Authors

Zhulin Li, He was born in 1976. He received the M.S. degree from college of information science and engineering, Northeastern University, Shenyang, China. Now he is a Senior Experimentalist, and his research interests include task scheduling of cloud computing environment, secure programming.

Cuirong Wang, She was born in 1963. She received the Ph.D. degree from the school of information science and engineering, Northeastern University, Shenyang, China in July 2004. Now, she is a professor and a research manager of the wireless sensor network and next generation network technology group. Her research interests include Routing Protocol, Network Security and Wireless Sensor Networks.

Haiyan Lv, She was born in 1979. She received Ph.D. degree from Chinese Academy of Agricultural Sciences. Now, she is a lecturer, and her main research fields are Remote Sensing and Geographical Information System.

Tongyu Xu, He was born in 1967. He received the Ph.D. degree from the college of information and electrical engineering, Shenyang Agricultural University, Shenyang, China in June 2006. Now he is a professor and a doctoral supervisor. His research interests is Agricultural Internet of things.