TELKOMNIKA, Vol.11, No.11, November 2013, pp. 6563~6569
e-ISSN: 2087-278X

A Resource Scheduling Strategy in Cloud Computing based on Multi-agent Genetic Algorithm

Wuxue Jiang[1,2], Jing Zhang*[1], Junhuai Li[1], Hui Hu[3]
[1]School of Computer Science and Engineering, Xi'an University of Technology, Xi'an 710048, Shaanxi, China
[2]Department of Computer Engineering, Dongguan Polytechnic, Dongguan 523808, Guangdong, China
[3]Department of Science and Research, Huizhou Univeresity, Huizhou 516007, Guangdong, China
*Corresponding author, e-mail: zhangjing@xaut.edu.cn

Abstract

Resource scheduling strategies in cloud computing are used either to improve system operating efficiency, or to improve user satisfaction. This paper presents an integrated scheduling strategy considering both resources credibility and user satisfaction. It takes user satisfaction as objective function and resources credibility as a part of the user satisfaction, and realizes optimal scheduling by using genetic algorithm. We integrate this scheduling strategy into Agent subsequently and propose cloud computing system architecture based on Multi-agent. The numerical results show that this scheduling strategy improves not only the system operating efficiency, but also the user satisfaction.

Keywords*: cloud computing, resources credibility, customer satisfaction, resources scheduling, multi-agent*

1. Introduction

Cloud computing system takes advantage of the tens of millions of idle computing resources on the Internet so that it can be more powerful than the centralized computing system. However, its resource scheduling faces challenges due to its massive resources, heterogeneous nature, and network communication delay.

Scheduling of resources is to assign the jobs submitted by users to appropriate resources reasonably to meet the needs of users and maximize system operating benefits. Scholars have put forward a series of scheduling strategies for the scheduling problems of Cloud Computing and distributed resources. [1, 2] propose a distributed resource scheduling strategy forecasting model based on ant colony algorithm to improve the dynamic and real-time performance of the distributed computing and real-time performance and effectiveness of resource scheduling. [3, 4] propose a balanced scheduling algorithm. The algorithm calculates job order values based on job dependencies, and adjusts the jobs according to that order values. As a result, key jobs will be finished as soon as possible, the wait time between jobs is reduced, and the job computing workflow execution time is finally reduced. It proves that the short job computing execution time of this algorithm in job management system has strong superiority. [5, 6] provide a distributed system scheduling model for the problem of weak dynamic regulatory capacity of fixed processing nodes distributed systems. The simulation experiments show that the algorithm has good dynamic control ability. It can reduce processor load and improve the delay of task processing as needed and make use of system resources more rationally. [7, 8] propose a task scheduling load balancing algorithm based on fair indexes. It deduces the task assignment method under multi-node conditions, and then improves the load balancing algorithm based on fair indexes with this method. [9, 10] induces a general model of load balancing scheduling on the basis of in-depth study of load balancing scheduling problem in distributed systems and conducts a detailed analysis of the various factors that affect load balancing.

For the current problems of cloud computing and distributed resource scheduling, this paper presents an integrated cloud computing resources scheduling strategy considering both resources credibility and user satisfaction base on the Multi-agent Genetic Algorithm. The

Received March 28, 2013; Revised June 20, 2013; Accepted July 20, 2013

strategy deals with both user satisfaction and resources credibility in the objective function. It integrates resource credibility into the function of user satisfaction and takes user satisfaction as optimization objective. Then it will calculate solutions with a genetic algorithm. By taking the above measures, the strategy can maximize the interests of users and computing efficiency. After integrating this scheduling strategy into Agents we propose a Multi-agent scheduling framework, which can complete resource scheduling in cloud computing with Multi-agent's excellent autonomy, learning ability and sociality.

2. Resource Credibility

Resource credibility is an index that reflects the processing capacity and reliability of the resources. It reflects the processing capacity and reliability of the resource during a period of time (from now to the future). With the model of resource credibility, the resources which have higher credibility are more likely to be called, while the resources which have lower credibility are less likely to be called or even have no chance to be called. So the load conditions of the resources will be more balanced. The credibility of the resource is dynamic and will vary with its performance, utilization, online rate and success rate of completed jobs. By introducing resource credibility into the scheduling strategy, the scheduling becomes dynamic and will change with resources. This is very important for distributed systems where resources change frequently.

Resources credibility is defined as follows:

$$r(t) = p(t) * i(t) * h(t) * v(t) \tag{1}$$

in which t is time, $r(t)$ is the credibility of the resources at the moment t, which consists of four parts: $p(t)$, $i(t)$, $h(t)$, $v(t)$. $p(t)$ is a performance function of resources. It reflects the processing capacity of the resources. For example, if the resource is a CPU, then it will be the CPU Clock Speed. $i(t)$ is the availability rate of resources. Same as $p(t)$, it reflects the processing capability of the resources. If the resource is CPU, then it will be (1 - CPU utilization). $h(t)$ is the online rate of resources, and it reflects the reliability of the resources. As distributed systems employ resources on the Internet, it is difficult to predict whether a resource is on-line or off-line at one moment. But we can obtain $h(t)$ by calculating the on-line rate of the resources. The higher the on-line rate is, the higher reliability the resource has, the higher credibility it has. $v(t)$ is the success rate of the resources to complete the job. After accepting jobs, the resources cannot always complete them successfully. Likewise, the success rate reflects the reliability of the resource.

3. User Satisfaction

Most scheduling algorithm studies only focus on optimizing overall system operating indicators, but they ignore the satisfaction from the perspective of an individual user. In fact, for different users, preferences of its operating time performance and economic costs are different. Take animation rendering system as an example, user A cares about rendering costs while user B is more concerned about the rendering time. Even for the same user, the requirements of job operating time and economic costs vary over time. When user A is facing an urgent case, the customer C wants to see the renderings as soon as possible and does not care about money. At this point, User A would prefer fast rendering performance; When user A is facing a non-emergency case, and the customer D has limited funds and needs A to control the budget, and he/she has no strict time requirement, we can tell that A will pay more attention to reducing costs; if the customer E asks user A to balance the economic costs and speed, obviously A will choose a balanced solution.

From this point of view, in order to meet the needs of different users, we need to design satisfaction functions according to their different requirements. And the objective function of the scheduling strategy should maximize the user's overall satisfaction. When the jobs of user i are assigned to resource j, the user satisfaction S_{ij} is calculated as follows:

$$s_{ij} = w_{it}\frac{r_j}{r_b} + w_{ie}\frac{c_b}{c_j} \tag{2}$$

$$w_{it} + w_{ie} = 1 \tag{3}$$

w_{it} and w_{ie} are the performance preference and cost preference respectively and their range is $[0,1]$. r_j is the credibility of resource j. The bigger r_j is, the bigger user satisfaction is. r_b is the mean resource credibility in the system history. c_j is the quotation of jth resource for the user i, and the lower c_j is, the bigger user satisfaction is. c_b is the mean quotation in the system history.

Depending on the user types, you can set different w_{it} and w_{ie}. For users who care more about performance, you can set w_{it} =0.8, w_{ie} =0.2 ; For users who care more about costs, you can set; For users who want a balancing system, you can set w_{it} =0.5, w_{ie} =0.2.

Credibility is an index that reflects the processing capacity and reliability of the resources.

4. Multi-agent System

Multi-agent technology is an important branch of artificial intelligence technology. Agent is an agency. It is a software entity that can complete its functions under certain circumstances continuously and spontaneously. It also can communicate with the relevant Agents and processes. Agent is autonomous. It can complete most of its functions and control its internal state without intervention of people or other Agents. Another important attribute of Agent is its social ability. It can take the initiative to interact with other Agents to achieve their goals. Agent also has the ability to learn. It can change with the environment adaptively.

Multi-agent system is a system composed of multiple Agents. It can solve complex problems which can not be solved by single Agent. Agents compete, coordinate and cooperate with each other to solve problems together.

Traditional scheduling methods, such as branch and bound method, dynamic programming and heuristic graph search algorithm, are mathematically perfect, but they can not adapt to complex production environments because they has made a lot of simplification of the real world. At the same time, lack of diversity has limited their scope of application. Scheduling strategy based on multi-agent technology can overcome the shortcomings of the above algorithms. The strategy relies on the swarm intelligence effect of Agents and has very good simulation of complex issues and processing power. At the same time it has powerful adaptivity and can change their status and learn relevant knowledge adaptively to achieve the expected effect of scheduling. As a result, it has very significant robustness.

The Multi-agent scheduling strategy presented in this paper contains the following Agents.

User Agent: Submit jobs to Job Response Agent and acts as the consumer of resources. It will occupy resources when the job application is successful and release resources after the job is completed.

Job Response Agent: It is the middle layer between the system and users. It is used to accept user-submitted job application and return results to the user after the job is completed.

Job Decomposition and Distribution Agent: Accept jobs from Job Response Agent, and decomposes jobs into the smallest particles and distribute the decomposed sub-jobs to Job Scheduling Agent.

Resource Search Agent: Search online resources in the network and register resources.

Job Scheduling Agent: Job Scheduling Agent will accept sub-jobs that has been decomposed into the smallest particles and select scheduling algorithm, and assign the sub-jobs to the Resource Agent.

Resource Agent: Send online messages to Resource Search Agents and select Resource Search Agent to determine the registration. Accept jobs from the Job Scheduling Agent and complete them. Detect its own status and send its own resources credibility to

Resource Search Agent. Submit job transfer signal to Job Monitoring Agent when the scheduling pours or job fails to apply for the transfer of jobs.

Job Monitoring and Coordination Agent: Monitor the status of the Resource Agent and re-allocate jobs after receiving the job transfer application signal sent by the Resource Agent.

The overall framework is shown in Figure 1:

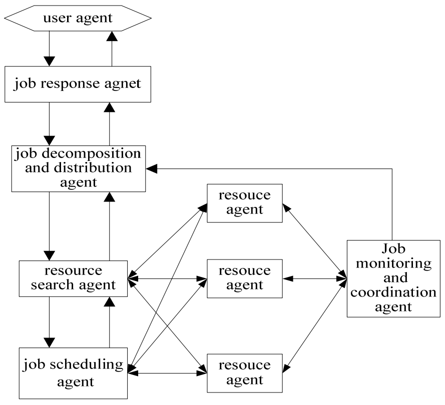

Figure 1. Multi-agent Scheduling Strategy Framework

5. Job-Resource Scheduling Strategy

Because of space constraints, we describe job resource scheduling strategy here only. Job resource scheduling strategy will be encapsulated in the Job Scheduling Agent. For the Job Agent, it has resources $1,2,......,m$ and credibility matrix $R=[r_1,r_2,......,r_m]^T$ of those resources and cost matrix $C=[c_1,c_2,......,c_n]^T$, and accepted jobs $1,2,......,n$ and jobs' satisfaction matrix $S=\begin{bmatrix} w_{1t} & w_{2t} & ... & w_{nt} \\ w_{1e} & w_{2e} & ... & w_{ne} \end{bmatrix}$. Its mission is to assign the accepted jobs $1,2,......,n$ to the resources $1,2,......,m$ respectively.

Since the scheduling problem is NP problem, it is often very difficult for traditional algorithms to obtain global optimal solutions. Genetic Algorithm stands out from most scheduling algorithms for its excellent global optimization capability. The main modules of the algorithm are: chromosome encoding, population initialization, genetic manipulation and fitness assessment. The flow of the algorithm is shown in Figure 2.

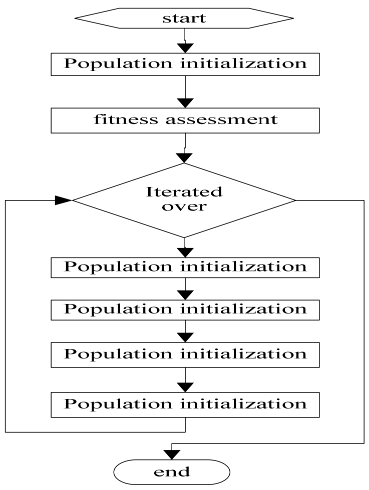
start
Population initialization
fitness assessment
Iterated over
Population initialization
Population initialization
Population initialization
Population initialization
end

Figure 2. Genetic Algorithm Flow Chart based on Resource Scheduling Strategy

5.1. Chromosome Encoding

The chromosome in this paper is a n-dimensional vector $D=[d_1 \quad d_2 \quad \cdots \quad d_n]$ (all elements are integers). $d_i (i=1\sim n)$ is an integer, it represents that the i th job is assigned to the resource d_i. d_i ranges in $1\sim m$. We can see that the vector D completely represent the allocation of resources for the current jobs $1, 2, ..., n$.

5.2. Population Initialization

After setting the size of the population: $popNum$, the algorithm will initialize randomly and generate $popNum$ chromosomes. And elements in each chromosome are also generated randomly, the number of elements ranges in $1\sim m$.

5.3. Fitness Function

After the initialization is complete, we need to assess the fitness of chromosome in the population. Rational fitness function is the key of the genetic algorithm because the fitness will guide the population evolutionary direction. Improper fitness function may lead to large deviation between population converged solutions and the optimal solutions.

As described in section 2, when the th job is assigned to the resource d_i, the user satisfaction s_{i,d_i} is calculated as follows:

$$s_{i,d_i} = w_{it}\frac{r_{d_i}}{r_b} + w_{ie}\frac{c_b}{c_{d_i}} \tag{4}$$

The same resource may be assigned to many jobs, as a result, multiple jobs may actually share the credibility of same resource. So, we need to adjust user satisfaction function according to the number of jobs assigned to each resource. If we set the number of jobs h_{d_i} assigned to resource d_i to the same value, then we can obtain the modified user satisfaction function:

$$s_{i,d_i} = w_{it}\frac{1}{h_{d_i}}\frac{r_{d_i}}{r_b} + w_{ie}\frac{c_b}{c_{d_i}} \tag{5}$$

The fitness evaluation function is as follows:

$$fitness = \sum_{i=1}^{n} s_{i,d_i} \tag{6}$$

5.4. Genetic Manipulation

The main operations of the genetic algorithm are: selection, hybridization and mutation. Selection is to select individual chromosome from the chromosome population to reproduce according to certain rules and preferences. The general rule is to choose those individual chromosomes who have better fitness. We adopt the roulette-based selection mechanism, in which the value, individual chromosome fitness/total fitness, acts as the individual chromosome's probability of being selected.

After selecting two or more chromosomes, Hybridization is to select a random bit-point in each chromosome and divide the chromosomes into two sections at that point. Subsequently, exchange the front or rear sections of two chromosomes to produce two new offspring chromosomes.

The mutation operation takes mutation rate as the probability and will regenerate some bit-code in one of the chromosomes randomly. Proper mutation rate can enhance the global optimization ability of the algorithm, however, if the mutation rate is too big, it will cause that the algorithm is difficult to converge.

6. Numerical Example

Simulation experiments are carried out in the CloudSim environment based on the JACK agent language. CloudSim is a simulation software of cloud computing which announced by Gridbus project in April 2009. CloudSim software framework consists of SimJava, GridSim, CloudSim, UserCode. CloudSim is an extensible simulation toolkit that enables modeling and simulation of cloud computing systems and application provisioning environments.

In this paper, the scheduling algorithm is implemented in CloudSim layer of the simulation platform, and the JACK agent simulation program is implemented in UserCode layer. The submitted job types and quantities are shown in Table 1.

Table 1. Job Type and Quantity

Job type	Quantity	Basic running time / s	Giving priority to performance	Giving priority to costs	Balanced Type
easy	3000	2	15%	70%	15%
medium	2000	5	33%	33%	34%
difficult	1000	10	70%	15%	15%

Seen from Table 1, job types are divided into three: easy, medium and difficult according to the basic running time of jobs (the basic time is the job running time at 1GHZ CPU). The composition of users is also different in each type of jobs. For example, in the easy job type, 70% of the jobs value costs, while 15% think more of performance and 15% think more of balanced type. This is because the running time of easy job is short, the user is more interested in their economic requirements. In the medium job type, performance, costs and balanced type account for approximate 1/3 each. Finally, for the difficult job type, because of its longer running time, users are more concerned about the performance. So 70% of the jobs value performance, while 15% think more of costs and 15% think more of balanced type.

The above jobs will not be released to the Multi-agent system at the same times. The system will release them to the Multi-agent randomly during a period of 1 hour.
Set the number of Resource Agent in the Multi-agent system to 10. The composition of Resource Agents is in Table 2.

Table 2. Agents of The Resources

Number of agents	Frequency / Hz	Running time
agent 1~3	500M	basic running time * 2
agent 4~7	1G	basic running time
agent 8~10	2G	basic running time / 2

Table 3. Average Response Time and User Satisfaction

Type of algorithm	Average response time/s	Overall user satisfaction
Multi-agent	5.42	1.68
Minimum load scheduling	5.74	1.59
Random scheduling	5.85	1.53

Among the three types of job shown in Table 1, medium type jobs are most objective and representative as they do not consider users' preferences. To compare this algorithm with random scheduling algorithm and minimum load scheduling algorithm, medium type jobs are taken as instance. The calculated average response time and user satisfaction is shown in Table 3.

From the above results, the Multi-agent's average response time and user overall satisfaction are shorter than that of Random scheduling algorithm and Minimum load scheduling algorithm, especially on the users' satisfaction, Multi-agent genetic algorithm than Random scheduling algorithm and Minimal load scheduling algorithm has greatly improved, and the increment between Multi-agent genetic algorithm and Random scheduling algorithm is 2.50 times of that between Minimum load scheduling algorithm and Random scheduling algorithm. Obviously, the user overall satisfaction of the Multi-agent genetic algorithm is superior to that of the other two algorithms.

7. Conclusion

An integrated assessment model considering both resource credibility and user satisfaction is established in this paper, and a resource scheduling strategy based on genetic algorithm is designed on the basis of this model. Then a cloud computing system resource scheduling architecture is proposed based on Multi-agent framework.

The numerical results show that this system enhances user satisfaction and reduces the average job access time. The operating efficiency of the cloud computing system is finally improved.

Acknowledgement

This work was supported by the National Natural Science Foundation of China (Grant No. 61172018), and by the National Natural Science Foundation of China (Grant No. 61172124), and by the Natural Science Foundation of Guangdong Province(Grant No. S2012010009865) and Guangdong Province Science and technology plan project (Grant No. 2012B010100040).

References

[1] Zhang Ying, Gao Min, Zhou Chi. Study on Cloud Resource Scheduling Problem Based on Logistics Park Distribution System. *Journal of Computational Intelligence and Electric Systems*. 2012; 1(1):122-125.

[2] Dewaki P, Valarmathi ML. Job Scheduling using Genetic Algorithm with Qos Satisfaction in Grid. *European Journal of Scientific Research*. 2012; 74(2): 272-285.

[3] Son Duy Dao, Kazem Abhary, Romeo Marian. Optimisation of Resource Scheduling in VCIM Systems Using Genetic Algorithm. *International Journal of Advanced Research in Artificial Intelligence*. 2012; 1(8): 49-56.

[4] Jianhua Gu, Jinhua Hu, Tianhai Zhao, Guofei Sun. A New Resource Scheduling Strategy Based on Genetic Algorithm in Cloud Computing Environment. *Journal of Computers.* 2012; 7(1): 42-52.

[5] Payal Singhal, Ravinder Singh, Pinky Rosemarry. An Improved Constraint Based Resource Scheduling Approach Using Job Grouping Strategy Grid Computing. *International Journal of Grid Computing & Application*. 2012; 3(4): 33-42.

[6] Buyya R, Ranjan R. Special Section: Federated Resource Management in Grid and Cloud Computing Systems. *Future Generation Computer Systems*. 2010; 26(8): 1189-1191.

[7] M Mezmaz, N Melab, Y Kessaci, et al. A parallel bi-objective hybrid metaheuristic for energy-aware scheduling for cloud computing systems. *Journal of Parallel Distribute Computing*. 2012; 71(1): 1497-1508.

[8] Sotomayor B, Montero RS, Llorente IM, et al. Virtual infrastructure management in private and hybrid clouds. *IEEE Internet Computing*. 2009; 12(5): 14-22.

[9] Veena Goswami, Sudhansu Shekhar Patra, GB Mund. Performance Analysis of Cloud Computing Centers for Bulk Services. *International Journal of Cloud Application and Computing*. 2012; 2(4): 13-26.

[10] RF Sun, ZW Zhao. Resource Scheduling Strategy Based on Cloud Computing. *Aeronautical Computing Technique.* 2010; 40(3): 103–105.