Cloud Computing Task Scheduling Algorithm Based On Improved Genetic Algorithm

Fang Yiqiu[1], Xiao Xia[1], Ge Junwei[1]
1. College of Computer Science and Technology, Chongqing University of Posts and Telecommunications
Chongqing, China
fangyq@cqupt.edu.cn, 1228107169@qq.com, gejw@cqupt.edu.cn

***Abstract*—Because of the continuous promotion of cloud computing applications, the demand for data processing in cloud computing is increasing. Users have higher requirements for the service quality of cloud computing, the high efficiency of cloud computing task scheduling algorithm plays a key role in the cloud computing. How to scheduling the computing resources efficiently, all tasks can be completed in the least time and cost is an important issue in cloud computing research. In this paper, a method of initial optimization on the crossover mutation probability of adaptive genetic algorithm (AGA) using binary coded chromosomes is proposed. Through experiments, the improved adaptive genetic algorithm is compared with the adaptive genetic algorithm (AGA) and the standard genetic algorithm (SGA). The experimental results show that the improved algorithm is an effective cloud computing task scheduling algorithm.**

Keywords—cloud computing; genetic algorithm; task scheduling

I. INTRODUCTION

Cloud computing is a mode of increasing, using and delivering related services based on the Internet. It usually providing dynamic, scalable and virtualized resources through the Internet. The National Institute of Standards and Technology (NIST) defines the cloud computing as a pay-for-use model that provides usable, convenient and on-demand network access. The resource in the configurable computing resource pools include networks, servers, storage, applications and services, which can be provided quickly with little management effort or little interaction by the service providers. With the development of cloud computing technology, the scale of cloud computing is getting larger and larger, and more and more tasks need to be dealt with. How to allocate and schedule these tasks effectively affects the overall efficiency and service quality of cloud computing. So the task scheduling algorithm of cloud computing has become an important issue in cloud computing research. For now, there are many studies on cloud computing scheduling algorithms, but these studies have some shortcomings.

II. RELATED WORK

The scheduling algorithms in cloud computing include the traditional task scheduling algorithms and Heuristic Task scheduling algorithms. The traditional scheduling algorithms include Min-Min algorithm, Max-Min algorithm, Sufferage algorithm and so on. Heuristic algorithms include genetic algorithm, ant colony algorithm, particle swarm optimization and other intelligent algorithms. As we know there is still premature phenomenon in genetic algorithm, which means that when the population evolves to the middle and late stages of the algorithm, the diversity of the population has been destroyed, and the search of the algorithm is stagnant, so that the algorithm falls into local optimum solution and cannot get the global optimum solution. The premature phenomenon affects the global convergence and computational performance of the genetic algorithm. [1] In addition, the current research on genetic algorithm in cloud computing task scheduling mainly has the following problems: the objective of optimization just only the task scheduling time [2], it uses the genetic algorithm to find the scheduling results which has the shortest total task time, or also the average task time of the scheduling results is also shorter. In fact, this method only aims at execution time, without considering other constraints. On the basis of optimizing the time of tasks, Reference [3] also proposes that the operation cost should be optimized at the same time. Although this method adds the cost, the cost is a function related to the time, so this method is actually about the optimization of the time of tasks. When research more into this field, there are more and more objectives for algorithm optimization. Multidimensional constrained task scheduling [4] is a multi-dimensional task scheduling algorithm that mainly considers time, cost, CPU, memory, load balancing and bandwidth constraints. This method has many objectives, first, many objectives can be transformed into functions of total task time. Secondly, many objectives have little impact on scheduling efficiency. For this problem, weight allocation for multi-objective has become a solution, [5] but if the weight allocation is not reasonable, it will seriously affect the quality of cloud computing services, so the determination of weight has become a more complex problem.

Based on the research of genetic algorithm and cloud computing task scheduling, combined with the above research results and problems, this paper proposes an improved scheme with the total time of task and load balancing as the optimization objective. According to the simulation results on cloudsim, it can be concluded that the algorithm can get better solution set than the previous scheme in the case of large amount of tasks and more computing resources.

978-1-5386-6243-4/19/$31.00 ©2019 IEEE

III. Implementation of Optimized AGA in Cloud Computing Task Scheduling

Cloud computing task scheduling is the process of assigning different tasks to appropriate computing resources. Assuming that there are M tasks and N computing resources, the total allocation results are N^M. Scheduling algorithm needs to find the optimal allocation results from all the results.

A. Coding and Decoding of Chromosomes

Genetic algorithms can encode chromosomes in two ways: binary coding and floating point number coding. Although binary coding has the defects of continuous function discretization mapping error and individual coding string may not meet the accuracy requirements in a short time, but the search ability of binary coding is stronger than the floating point number coding. The difference became more pronounced as the population size increased. [6] Therefore, the chromosome encoding in this paper adopts binary encoding, uses two-dimensional array for storage, the rows represent tasks and columns represent computing resources.

The value of each genetic locus is 0 or 1, Chromosome[i][j]=1, means the ith task to be performed on the jth computing resource, such as i=7, j=1,means the seventh task to be performed on the first computing resource. If the following chromosome {10001000110001010100100} is generated, it can be known by decoding, that the first task is executed on the first computing resource, the second task is executed on the second computing resource so on and so forth.

TABLE I. The Chromosome Example

tasks	computing resource		
	1th	*2th*	*3th*
1th	1	0	0
2th	0	1	0
3th	0	0	1
4th	1	0	0
5th	0	1	0
6th	1	0	0
7th	1	0	0

The time that each computing resources execute all the assignments can be calculated by the chromosomes and the ETC (Expected Time to Compute) matrix [7] (ETC [i] [j] means that the time the ith task working on the jth computing resources needed to complete). And the time to complete all the tasks is the maximum value of time calculated above. assumes that the number of computing resources for n, number of jobs for m. Then the total time to complete the scheduling is:

$$\text{ScheduleTime} = \max_{0\le j\le n}\left\{\sum_{i=0}^{m}\text{ETC}[i][j]\times\text{Chromosome}[i][j]\right\} \quad (1)$$

B. The Generation of Initial Population

The initial population use the random generation. Each time the population generated some of the specific patterns need to be eliminated. The existence of these specific patterns (the same task is repeatedly scheduled to be executed on multiple computing resources) will affect the search efficiency.

C. The Function of Fitness

The objective of task scheduling optimization in this paper is not only to minimize the total task time, but also to balance the load of each computing resources. The load balancing is measured by the standard deviation of the time it takes each computing resources to complete all tasks on that computing resource. The task running time of each computing resource is:

$$\text{VMtime} = \sum_{i=0}^{m}\text{ETC}[i][j]\times\text{Chromosome}[i][j] \quad (2)$$

And the standard deviation of the running time of tasks on each computing resource is:

$$\text{Load} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(\text{VMtime}_i - \frac{\sum_{j=1}^{n}\text{Vmtime}_i}{n}\right)^2} \quad (3)$$

The smaller the standard deviation, the closer the task running time on each computing resource is to the average task running time on each computing resource, the more balanced the load will be. So the Function of Fitness is:

$$F_i = \frac{1}{\text{ScheduleTime}+\text{Load}} \quad (4)$$

It means the scheduling the individual represents for has the minimum total time to complete the task and the balanced load on each computing resource, the higher the fitness value, the easier to be choose.

D. Genetic Operations

1) The Probability of Selection

Selection operation is the basic way of genetic algorithm to realize the good gene transmission. In the improved algorithm, roulette selection is used in the selection operation. And the probability of each individual being selected is proportional to the value of fitness function. The selection probability is calculated by fitness function (4).

$$P_j = \frac{F_j}{\sum_{j=0}^{n}F_j} \quad (5)$$

When selecting the next generation of individuals, the probability of each individual being selected is calculated by (5). Through such selection, there are individuals with the shortest total task time and balanced load in the population, while maintaining the genetic diversity of the population.

2) Crossover and Mutation

Crossover is the main search operator in genetic algorithm, which imitates the recombination process of sexual reproduction in nature, inherits the original good genes to the next generation of individuals, and generates new individuals with better gene structure. Mutation can expand the new search space and maintain the diversity of the population when the population converges locally.

If the crossover probability of genetic algorithm is high, the genetic model with high fitness may be destroyed. If the crossover probability is small, the search for the optimal solution will be slow. Similarly, for the probability of variation, if the probability of variation is too high, some excellent genetic models will be more likely to be destroyed. When the probability of variation is higher, the genetic algorithm will degenerate into

a random search algorithm. If the probability of variation is small, then the algorithm search process will become very slow. The determination of crossover and mutation probability is a tedious work, and the probability of finding is not necessarily the optimal probability. Therefore, Srinvivas proposed an adaptive genetic algorithm, which automatically increases the probability of crossover and mutation when the fitness of the population tends to be consistent, and automatically reduces the probability of crossover and mutation when the fitness is scattered. At the same time, individuals whose fitness is higher than the average fitness should be given a lower probability of crossover mutation to enter the next generation. For individuals whose fitness is lower than the average fitness, a large probability of crossover mutation should be given to eliminate them as soon as possible.

In the adaptive genetic algorithm, crossover (P_c) and mutation (P_m) probability is adjusted according to the following formula:

$$P_c=\begin{cases}\frac{k_1(f_{max}-f')}{f_{max}-f_{avg}}, & f\geq f_{avg}\\ k_2, & f<f_{avg}\end{cases} \quad (6)$$

$$P_m=\begin{cases}\frac{k_3(f_{max}-f)}{f_{max}-f_{avg}}, & f\geq f_{avg}\\ k_4, & f<f_{avg}\end{cases} \quad (7)$$

Where the fmax represents the maximum fitness value in each generation population, the favg represents the average fitness value in each generation population, f' represents the fitness value of the larger one of the two individuals to be crossed, f represents the fitness value of the individual to be changed, k1, k2, k3, k4 take the value of the interval (0,1). [8]

Through (6) and (7), we can find that when the fitness value is closer to the maximum fitness value in the population, the probability of crossover and mutation is smaller. When the fitness value is equal to the maximum fitness value in the current population, the probability of crossover and mutation becomes 0. This improvement is more appropriate in the later stage of evolution, but in the early stage of evolution, the individuals close to the maximum fitness value in the population almost do not cross and mutate, but the individuals with the maximum fitness at this time might not be the global optimal solution, which easily leads to the algorithm search falling into the local optimal situation. In view of the problems at the beginning of the evolution, the Kuang's work[9] put forward by the cosine of the improved adaptive genetic algorithm and Yu's work[10] using custom discriminant to determine whether a group in the immature convergence. According to different conditions using macro or micro processing method to set the crossover probability and mutation probability method. After research and combining the IAGA algorithm improvement by Zhang [11], get a better improve adaptive formula. The improved formula is as follows:

$$P_c=\begin{cases}\frac{k_1(f_{max}-f')+k_2^n}{f_{max}-f_{avg}+k_2^n}, & f\geq f_{avg}\\ k_2, & f<f_{avg}\end{cases} \quad (8)$$

$$P_m=\begin{cases}\frac{k_3(f_{max}-f)+k_4^n}{f_{max}-f_{avg}+k_4^n}, & f\geq f_{avg}\\ k_4, & f<f_{avg}\end{cases} \quad (9)$$

Where the n represents the current evolution generation. So it can improve the early evolutionary population to the fitness values of individuals almost never happen in the crossover and mutation, and by the end of evolutionary k_2^n and k_4^n will tend to be 0, the algorithm become adaptive genetic algorithm again, and the advantages will be kept.

IV. Results and Analysis

A. Improved algorithm test on Schaffer F6 function

The function has an infinite number of local maximum points, which only one is the globally maximum point, the point is (0,0), and the maximum value is F(0,0) =1. The value of the independent variable is x∈ (-10,10) , y∈ (-10,10) . The Schaffer's function has a circular ridge around the maximum peak, and their values are all 0.990283, which is easy to stop at this local maximum.

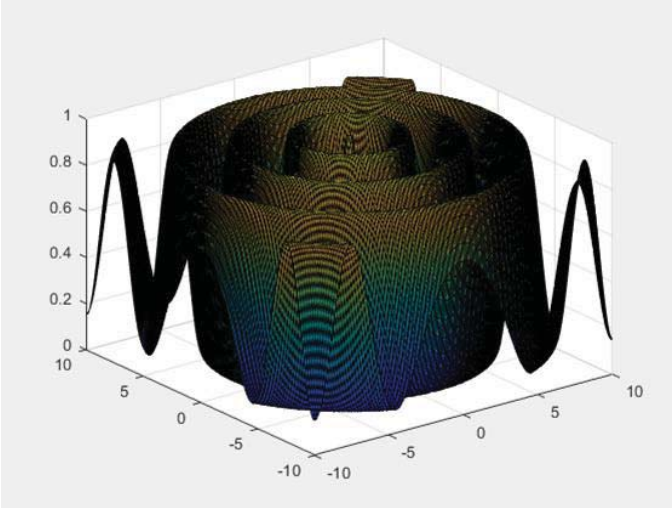

Fig. 1. Schaffer F6 function

$$F(x,\ y)=0.5-\frac{\sin^2\sqrt{x^2+y^2}-0.5}{(1+0.001\ (x^2+y^2))^2} \quad (10)$$

In this paper, the algorithm is tested under the environment of Windows 7, MATLAB 2009. This is the test result (Fig. 2).

Fig. 2 shows the algorithm's performance comparison from Kuang's work and Yu's work, AGA and the improved AGA from this paper average fitness under the test function. This paper's improved AGA has an average fitness that is closer to the optimal fitness than Kuang's work and Yu's work, and also convergence faster than the others, so indicating that the population of this paper's improved AGA has more excellent solutions than the algorithm from Kuang's work and Yu's work, and showing strong adaptive performance. So the new algorithm has a strong optimization ability.

B. *Simulation Cloud Environment Test*

In this paper, CloudSim platform [12] was used to simulate the cloud computing environment, and under the same conditions, SGA, AGA and improved AGA by this paper were compared in the experiment. In CloudSim, the task size, that is, the number of instructions to be executed, represented by length and the computing power of the virtual machine, that is, the number of instructions to be executed per unit time, measured by MIPS, the time required to complete a task on a virtual machine is represented by the ratio of length to MIPS.

It can be seen from the test function results that the algorithm has been convergent at the evolution generation about 1000, and there is no need for more generation in searching the optimal solution. The maximum evolution generation is set as 1000. If there is no significant change in the task completion time of successive 50 generations, the algorithm will be terminated ahead of schedule. Set the number of computing resources are 5, the population size is 100, and other parameters from

TABLE II. PARAMETERS

algorithm	parameter	value
SGA	scale	100
	P_c	0.7
	P_m	0.1
AGA	scale	100
	K_1	0.9
	K_2	0.9
	K_3	0.6
	K_4	0.6
ImprovedAGA	scale	100
	K_1	0.9
	K_2	0.9
	K_3	0.9
	K_4	0.9

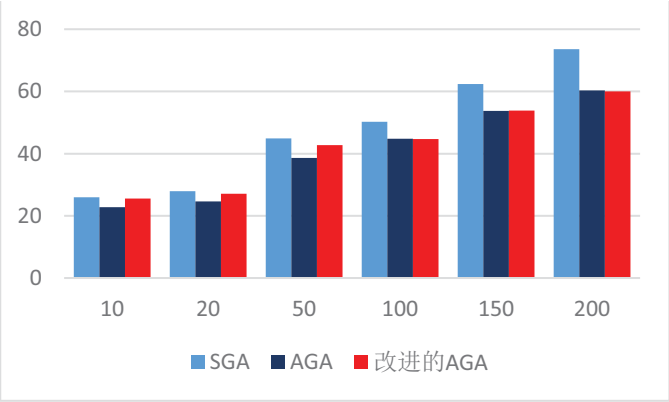

Fig. 3. Task numbers(x) and time (y)

Based on the observation of fig. 3, we can conclude that AGA's efficiency is higher than SGA. With the increase of tasks, AGA always takes less time to complete the task than SGA, and the gap becomes more obvious when the tasks increased. The improved AGA algorithm in this paper is not as efficient as AGA algorithm when the task amount is relatively small (when the Chromosome length is relatively short). When the task amount increases to a large value, the advantages of binary coding are reflected, and the advantages of our improved algorithm are gradually reflected.

V. CONCLUSION

With the emergence of cloud computing, the resource allocation and scheduling of cloud data center has become an important factor determining the efficiency of cloud computing, and the scheduling of cloud resources has become a research hotspot at present. On the basis of genetic algorithm, this paper quantifies the satisfaction of resource scheduling by improving the execution process of genetic algorithm, aiming at the completion time of scheduling tasks and load balancing, and forms a set of cloud resource scheduling research scheme, and uses CloudSim platform to simulate the cloud computing environment for example analysis. Experimental results show that the improved genetic algorithm in this paper has a better effect on cloud resource scheduling, achieves a more reasonable task scheduling, and produces an ideal task scheduling result.

The future work is to explore the maximum number of tasks that binary coding can be used. Because when the number of tasks increases to a certain value, due to the excessive length of chromosome, encoding and decoding will consume a lot of time and space, so the efficiency of the algorithm will be seriously reduced. In the further work, it will give the recommendations for the certain value.

REFERENCES

[1] Xiong Weiqing, wei Ping. Research on early maturity of genetic algorithm [J]. Computer application and research, 2001 (9) : 12-14.

[2] Li Jianfeng, Peng Jian. Task scheduling algorithm based on improved genetic algorithm in cloud computing environment [J]. Computer application, 2011,31 (1) : 184-186.

[3] Zhu Zongbin, Du Zhongjun. Cloud computing task scheduling algorithm based on improved GA [J]. Computer engineering and application, 2013,49 (5) : 77-80.

[4] Li Chao, Dai Bingrong et al. Research on multi-dimensional constrained task scheduling based on improved genetic algorithm in cloud computing environment [J]. Small computer system, 2017,38 (9) : 1945-1949.

[5] Huang Chao, Hu Demin, Yu Xing. Application of multi-objective genetic algorithm in cloud computing task scheduling [J]. Information technology, 2014,05:130-134.

[6] Wang Xiaoping, Cao Liming. Genetic algorithm [M]. Xi 'an: xi 'an jiaotong university press,2002.

[7] Braun T D, Siegel H J, Beck N, et al.A comparison of elevenstatic heuristics for mapping a class of independent tasksonto heterogeneous distributed computing systems[J].Journalof Parallel and Distributed Computing, 2001, 61 (6) : 810-837.

[8] Srinivas M, Patnaik L M.Adaptive probabilities of crossover and mutation in genetic algorithms[J].IEEE Trans on SMC, 1944, 24 (4) : 656-667.

[9] Kuang Hangyu, Jin Jing, Su Yong. Improvement of crossover mutation operator of adaptive genetic algorithm [J]. Computer engineering and application, 2006, 42(12):93-96.

[10] Yu Guangshuai, Yu Xianwei. An improved adaptive genetic algorithm [J]. Mathematical practice and understanding, 2015, 45(19):259-264.

[11] Zhang Guoqiang, Peng Xiaoming. Improvement and application of adaptive genetic algorithm [J]. Ship electronic engineering, 2010, 30(1):83-84.

[12] Calheiros R N, Ranjan R, de Rose C A F, et al.CloudSim : a novel framework for modeling and simulation of cloudcomputing infrastructures and services[EB/OL].[2009-09-03].http : //www.cloudbus.org/reports/CloudSim-ICPP2009.pdf.

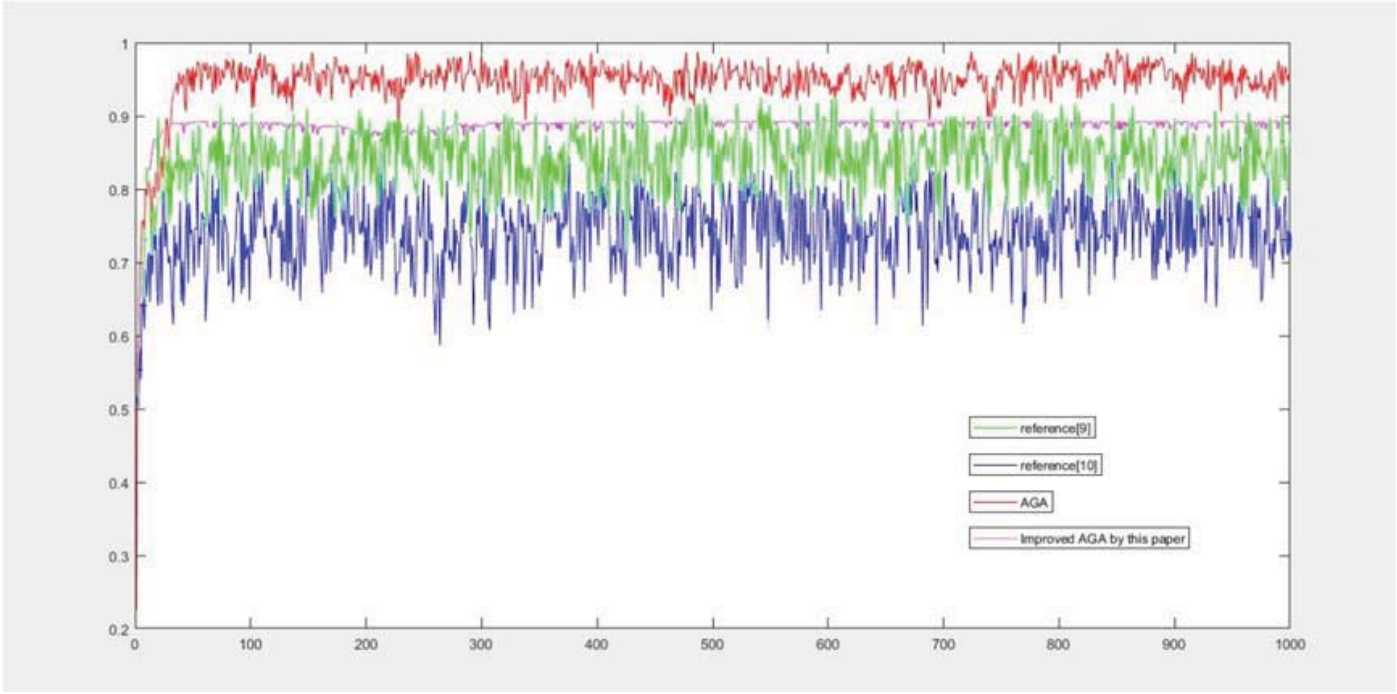

Fig. 2. The relationship between evolution generation (x) and the optimal fitness value (y) per generation