EasyChair Preprint

№ 4015

An Improved Ant Colony Optimization Job Scheduling Algorithm in Fog Computing

Chao Yin, Tongfang Li, Xiaoping Qu, Sihao Yuan and Yan Liu

August 9, 2020

An Improved Ant Colony Optimization Job Scheduling Algorithm in Fog Computing

Chao Yin [a], Tongfang Li*[a], Xiaoping Qu[a], Sihao Yuan and Yan Liu[a]
[a]Department of Information Science and Technology, Jiujiang University, Jiujiang, China 332005

ABSTRACT

With the application of internet of things technology, fog computing has provided computing and storage services near the bottom network. It can solve the problem of rapid response and bandwidth consumption of delay sensitive applications at the edge of local network as a highly virtualized platform. However, in the case of large scale service requests, if the job scheduling problem cannot be effectively solved, it will increase service delay, reduce resource utilization and user satisfaction. In this paper, we have improved the basic ant colony optimization (ACO) and developed a new job scheduling strategy improved ant colony optimization named IACO. IACO can assign to the resource with the lowest total cost of all selected tasks, which is always determined by calculating the total cost value of the task on the resource. We have finished some experiments by imitating the foraging process of multiple ants and repeated it iteratively. The optimal task scheduling sequence can be obtained through the different pheromone concentration left by ants. Experimental results show that IACO scheduling algorithm is better than ACO in the total cost, completion time and economic cost.

Keywords: Fog Computing, Job Schedule, IACO

1. INTRODUCTION

With the rapid development of internet of things technology in recent years [1], millions of intelligent devices are connected to each other and exchange data through the internet. However, the data center is far away from the user terminal in the multi-level structure of internet of things, so that user messages usually need to go through multiple hops to reach the data center [2]. It is not easy to move data from the edge to big data centers when more and more data content requires higher bandwidth to transmit. In addition, unpredictable delays can destroy the user experience of time-sensitive applications, including human-computer interfaces, emergency services and real-time games. It is difficult to deploy applications with high real-time requirements in the cloud server since the distance between the cloud server and the user terminal is too high. Fog computing is proposed to extend the traditional cloud computing paradigm to the edge of the network as a new service computing mode [3].

Located between cloud computing and the internet of things, fog computing is characterized by low latency, wide geographical distribution, and high mobility [4]. The traditional network computing processes, computes and stores data at the center of the network, while the fog computing deals with all the things at the edge of the network [5]. Due to the dynamic and uncertain nature of resources, as well as the high variability and unpredictability environment, reasonable resource scheduling and allocation is particularly important in order to meet the needs of different users better [7].

In fog computing, resource scheduling faces a lot of problem, such as the total cost, completion time, economic cost and so on. ACO is an effective solution, which is simulated the evolution mechanism of natural organisms. It is proposed as a novel group bionic intelligent algorithm, which can imitate the group behavior of real ants in the process of foraging. Ants will release a substance called pheromone on their way. The more ants choose a path, the higher the concentration of pheromone on it. The change of pheromone concentration can get the shortest path, which can be used to explain the process of optimization. In this paper, we update the method about the pheromone to achieve global optimization. We have developed IACO to form a new pheromone generated by random deviation disturbance, and update the pheromone to get a better path. This method can solve the resource scheduling problem in fog computing.

The contributions of this paper are described as follows:

--We have established a resource scheduling model, which can minimize the time, cost of user's job completion and optimize the load balance of the system.

--We have proposed IACO to update the method of the pheromone, which can find a match between tasks and resources to minimize the total cost of users and providers.

--The experimental results verify that IACO can provide better job scheduling in fog computing.

The rest of this paper is organized as follows. Section 2 is related works. Section 3 introduces the resource scheduling model. Section 4 introduces the IACO. The experimental result and evaluation are described in Section 5. Section 6 is the conclusion.

2. RELATED WORKS

Job scheduling is an important research direction of fog computing. At present, some scholars have proposed some fog computing resource scheduling algorithm to achieve different scheduling goals. There are algorithms to reduce execution time, reduce energy consumption, balance delay, which usually transform the job scheduling problem into a constrained optimization problem [8] [9]. People adopt the corresponding scheduling strategy to solve it so as to optimize different scheduling objectives.

Smart mobile devices are popular at present, but there is usually a lot of limitation. The current usage cannot satisfy complex user requests, which will greatly reduce the user experience. In order to solve these problems, some high-energy computing tasks which cannot be handled by terminals need to be transferred to the cloud for processing to reduce the energy consumption and cost of terminals. However, long network delay will be caused due to the distance from the cloud. At the same time, a large amount of data transmission will increase the network load. In order to solve this problem, paper [10] adopts markov decision process method, which will schedule the calculation task based on the queuing state of the task buffer, the execution state of the local processing unit and the state of the transmission unit. Paper [11] studies how to schedule tasks to heterogeneous clouds and how computing resources in edge clouds are allocated to users. An optimization algorithm is proposed to meet the task delay requirement. When the task is only offloaded to the edge cloud, the computing resources are allocated to the task with loose delay limit. In heterogeneous cloud scenarios, edge clouds allocate resources to tasks with strict delay boundaries, while tasks with loose delay boundaries are scheduled to remote clouds. In order to meet resource and delay sensitive requirements and allocate cloud resources for heavy computing tasks, paper [12] proposed a task scheduling algorithm based on fog area and cloud. Scheduling problem is described as an integer programming problem and solved by heuristic algorithm.

With the rapid progress of mobile information technology, many new mobile applications have emerged, such as face recognition, voice recognition, interactive games and augmented reality. These mobile apps consume a lot of energy during operation, which puts a lot of pressure on resource-constrained mobile devices to meet the demands of these apps. We can migrate these mobile applications to edge servers for data storage, which reduces the energy consumption on mobile devices to some extent. In addition, how to achieve the best task migration is a key issue we need to consider. In view of this problem, paper [13] proposes an optimal task unloading transfer scheme that considers the power consumption in mobile edge computing, which aims to minimize the power consumption of mobile devices. In this paper, the concept of conditional power consumption is firstly proposed. The conditional power consumption of mobile devices is analyzed, and an optimal task unloading transfer model is designed based on the above analysis. Simulation studies show that the optimal task unloading and transferring mechanism can find the appropriate virtual resource to perform the task and complete the task quickly, which can save more energy and have good performance. Paper [14] proposed the problem of energy consumption minimization in mobile edge cloud computing. KKT condition is used to solve this problem. The paper has proposed a request uninstall transfer scheme, which is determined by the energy consumption and bandwidth capacity of each slot.

In fog computing networks, some applications have strict delay and energy consumption requirements. For example, the internet of vehicles has high delay requirements, and online games on mobile terminals have energy consumption requirements. How to balance the delay and energy consumption is a challenging problem in the fog network. To solve this problem, a new task scheduling algorithm was proposed [15-17]. A cross-layer analysis framework is proposed, which considers actual local execution, task unloading, wireless transmission, incentive constraints, user traffic, and queue models. In addition, a limited control parameter is described as the tradeoff between service latency and energy consumption. A Delay Energy Balanced Task Scheduling（DEBTS) algorithm was proposed by Lyapunov optimization technique to minimize the total energy consumption and reduce the average service delay. Simulation results show that the proposed algorithm can achieve better delay and energy consumption performance compared with local execution and traditional task scheduling algorithms. Paper [18] proposes a new fog computing model, which contains remote

cloud nodes and local cloud nodes and is connected to the wireless access infrastructure. An offloading transfer strategy was also proposed to consider task execution, energy consumption and other costs. Experimental results show that the proposed scheme can reduce the execution time and energy consumption better. Paper [19] proposed a joint resource allocation and coordinated unloading transfer algorithm, which made full use of the advantages of Cloud Radio Access Network (C-RAN) and fog computing. Based on the maximum transmission delay tolerance constraints, front transfer and backhaul capacity limits, the energy cost was minimized and the optimal computational resource allocation was obtained.

3. RESOURCE SCHEDULING MODEL

There are different resources as a resource pool in the fog computing data center. This resource pool is made up of virtual resources, which provide different cpu processing power, memory, bandwidth, storage and other computing and storage resources. Each of virtual resources has a different price. In principle, the better the performance, the higher the price. This paper focuses on the matching of tasks and resources, that is, which resources are allocated for tasks. The purpose of job scheduling is to find a match between tasks and resources to minimize the total cost to users and service providers, including job completion time, cost and the load balance.

Suppose there are *m* resources in the data center, and the user submits *j* jobs, which are divided into *n* tasks. The following assumptions are made for the scheduling model [20-21]:

(1) Tasks are independent of each other.

(2) A task can only be assigned to one resource.

(3) The number of tasks is more than the number of resources.

(4) The usage of each resource can be obtained in real time, and the execution time of tasks can be estimated.

Each task has some information, including: each user task has a unique flag named as CloudLetid, number of processors named as Num_of_Cpu, file size submitted to data center named as InputSize, task length named as Length, output file size named as OutputSize.

One or more resources can be opened in a data center, including: each resource has a unique ID Re_id, the size of resource image named as ImageSize, single processor processing power named as Capacity, and resource price named as Price.

The execution time and cost of each task on each resource can be estimated as Esttime and Estcost, respectively. Both of them are n*m matrices, which are expressed in formulas (1) and (2).

$$Esttime = \begin{bmatrix} time_{0,0} & \cdots & \cdots & time_{0,m-1} \\ time_{1,0} & \cdots & \cdots & time_{1,m-1} \\ \vdots & \cdots & time_{i,j} & \vdots \\ \vdots & \cdots & \ddots & \vdots \\ time_{n-1,0} & \cdots & \cdots & time_{n-1,m-1} \end{bmatrix} \tag{1}$$

$$Estcost = \begin{bmatrix} Estcost_{0,0} & \cdots & \cdots & Estcost_{0,m-1} \\ Estcost_{1,0} & \cdots & \cdots & Estcost_{1,m-1} \\ \vdots & \ddots & Estcost_{i,j} & \vdots \\ \vdots & \ddots & \ddots & \vdots \\ Estcost_{n-1,0} & \cdots & \cdots & Estcost_{n-1,m-1} \end{bmatrix} \tag{2}$$

In the matrix Esttime, $Time_{i,j}$ is the execution time of task i on resource j. The calculation method is shown in formula (3).

$$Time_{i,j} = Cputime_{i,j} + Transtime_{i,j} \qquad i \in [0, n-1], i \in [0, m-1] \tag{3}$$

$Cputime_{i,j}$ is CPU execution time, $Transtime_{i,j}$ is file transfer time, both of which are calculated according to formula (4) and (5).

$$Cputime_{i,j} = Length_i / (CpuNum_i \times Capacity_j) \tag{4}$$

$$Transtime_{i,j} = (InputSize_i + OutputSize_i) / Bw_j \tag{5}$$

In the matrix Estcost, $Estcost_{i,j}$ means the cost of task i on resource j, which includes CPU cost, memory cost, hard disk cost, and bandwidth cost.

Rsctime is the total execution time of resource j after all tasks are completed, as shown in formula (6). The number of tasks assigned to the resources is shown as s.

$$Rsctime_j = \sum_{d=0}^{s-1} Time_{d,j} \tag{6}$$

All tasks are executed on virtual resources concurrently so that the total completion time of all tasks is the maximum execution time of all resources, which is described as Completetime. The task scheduling scheme is described as I. Expressed by formula (11).

$$Completetime(I) = \max_{j=0}^{m-1}(Rsctime_j) \tag{11}$$

From the cost matrix Estcost, we can get that the cost to complete all tasks is the sum of all $Estcost_{i,j}$, such as the formula (12).

$$\mathrm{Cos}\,t(I) = \sum_{i=0}^{n-1}\sum_{j=0}^{m-1} Est\ \mathrm{cos}\,t_{i,j} \tag{12}$$

According to the number of tasks of the resource cannot be good as a standard to evaluate the load balance of the resource, because the performance of the resource is not the same. The good performance of the virtual resources should be assigned more tasks. Therefore, the execution time of the resource will be used as the evaluation of load balancing. The calculation method is shown as formula (13) and (14).

$$Loadbalence(I)\sqrt{\frac{\sum_{j=0}^{m-1}(Rsctime_j - \overline{Rsctime})^2}{m}} \tag{13}$$

$$\overline{Rsctime} = \frac{\sum_{j=0}^{m-1} Rstime_j}{m} \tag{14}$$

Assuming that the failure rate of resource j is subject to an exponential distribution with parameter j. When task i is executed by time t, the reliability of resource is $Relib_{i,j}$ can be expressed as formula (15).

$$\mathrm{Re}lib_{i,j} = e^{-\lambda_j i} \qquad i \in [0, n-1], j \in [0, m-1] \tag{15}$$

$$\mathrm{Re}\,lib = \prod_{i-0}^{n-1} \mathrm{Re}\,lib_{i,j} \tag{16}$$

In conclusion, the total cost of the tasks includes completion time, cost, load balancing standard deviation and reliability requirements. We have used the weighted method to change multi-objective optimization into single-objective optimization. The total cost is calculated as formula (17). The optimization goal of scheduling algorithm is to minimize the total cost.

$$\begin{aligned} &All\,\mathrm{Cos}\,t = a * Completetime + b * \mathrm{Cos}\,t + c * Loadbalance + d * \ln \mathrm{Re}\,lib \\ &s.t. \quad a + b + c + d = 1 \\ &\qquad a, b, c, d \in [0, 1] \end{aligned} \tag{17}$$

In formula (17), a, b, c and d are the weights of the completion time of the activity, the economic cost of the activity, the load balancing degree and the reliability respectively. The larger the corresponding value is, the more attention is paid to the target.

4. IMPROVED ANT COLONY OPTIMIZATION ALGORITHM

4.1 The main idea of IACO

Ant colony optimization (ACO) is a new generation of heuristic bionic intelligent algorithm proposed by several scholars in 1991. It imitates the real ant colony behavior in the process of foraging. Ants release a kind of substance called pheromone on their way. The more ants choose a path, the higher the concentration of pheromone on it, which makes the process of ant colony searching for path a positive feedback process. The ants will exchange information and cooperate with each other in an organized way because of the change of the concentration of the substance. They can find the shortest path from the nest to the food source finally. The above process can be used to explain the process of artificial ant colony optimization. There are three core contents in the process. The first one is the selection method. The more pheromones the path is, the more likely it is to be selected. The second one is update pheromone mode. The shorter the path is, the faster the pheromones on it will increase. The last one is the cooperation method. The information element is used as the intermediary between individuals to exchange information. Pheromones play a decisive role in the whole process.

Differential evolution algorithm (DEA) is a new group bionic intelligent algorithm proposed by Rainer storm and Kenneth price. It simulates the evolution mechanism of natural organisms on the basis of genetic algorithm and other evolutionary ideas. This algorithm has the characteristics of preserving individual optimal solution and sharing information among individuals in the population. It allows individuals in the population to cooperate and compete to solve the optimization problem, including generating initial population, mutation, crossover and selection. The first operation is to initialize the population since each individual is equivalent to a possible solution in the population. Then we should take an individual as the target individual, and select two individuals in the population to subtract randomly. The third operation is to add the difference vector and another random individual to get the variant individual. We cross the variant individual with the target individual to get the crossover individual according to the principle of selecting the best. Comparing the cross individuals with the original target individuals, we can get new individuals at last.

For ant colony optimization algorithm, pheromone is the soul of the algorithm. We use mutation operator, crossover operator and selection operator to form a new pheromone generated by random deviation disturbance to update the pheromone in the IACO algorithm to get a better path. The main idea of this algorithm is to get the optimal order of task scheduling by using the IACO algorithm. The task is scheduled to a certain resource, which is determined by calculating the total cost value of the task on the resource. The task is always allocated to the resource with the lowest total cost, which refers to the total cost of all currently selected tasks. By simulating the process of ants' foraging, the algorithm can find the task scheduling sequence and match resources. The optimal task scheduling sequence can be obtained by making multiple ants repeat the feeding process with different pheromone concentration left by ants. Because the goal of the job scheduling model is to minimize the total cost of scheduling all tasks to the corresponding resources, these costs take into

account the user's QoS (time and cost) and the degree of system load balancing concerned by the fog service provider. Therefore, this paper uses the total cost of tasks to update pheromones. For example, the higher the concentration of pheromones is on the search path, the lower the total cost of task scheduling sequence is.

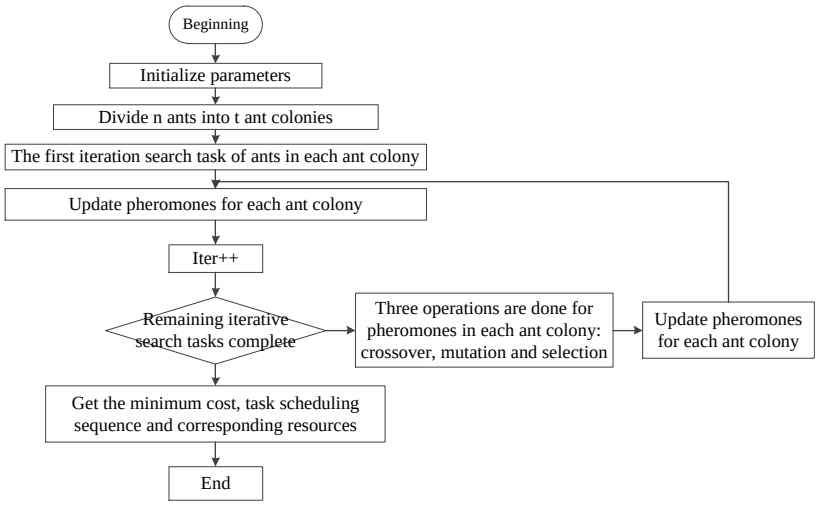

Figure 1. Scheduling flow chart of algorithm IACO.

Combining job scheduling model and IACO algorithm, the process of job scheduling in fog computing environment can be obtained.

Step 1: Initialize parameters and the maximum number of iterations, which are related to ant colony optimization algorithm and differential evolution algorithm. At the same time, we initialize the pheromone concentration between the two tasks to a constant.

Step 2: Divide m ants into t groups and place each ant in the task randomly.

Step 3: For the first iteration, let each ant search task in each ant colony group. Roulette is used to select the next task and the resource with the lowest total cost at that time, until all tasks are matched. Then we update the information prime number group of each ant colony group.

Step 4: For the other iterations, the pheromone in each ant colony group is operated by mutation operator, crossover operator and selection operator. Among them, mutation operator and crossover operator are used to form a new pheromone array. The selection operator is used to update pheromones in each ant colony by selecting new and old pheromones and minimizing the total cost.

Step 5: Repeat step 4 until the remaining iterations are completed.

Step 6: Output the minimum total cost, task order and corresponding resource order.

4.2 The Parameter of Pheromone

For each ant colony, there is a two-dimensional array representing pheromones between two tasks. If the number of tasks is n, the two-dimensional array is n*n. There are t pheromone arrays in t ant colony, which are recorded as t_i, i = 1, 2, 3 t. For each ant colony, the following pheromone transformation operations are performed:

$$v_i = \tau_{r1} + F \times (\tau_{r2} - \tau_{r3}) \tag{18}$$

The first parameter is mutation operator. According to formula (18), the mutation pheromone v_i is generated for the current pheromone. Among them, r1, r2 and r3 are random numbers of [0, T). They are not equal and cannot be equal to i. The minimum value of T is 4. F is a constant factor with a value of [0, 2], which is used to control the importance of the difference between two pheromones. The second parameter is crossover operator. We use crossover operation to generate a new pheromone array U_i, whose representation is shown in formula (19) (20). Among them, $T_{i,}^{j,k}$ represents

the pheromone concentration between task j and task k in group i ant colony, and $V_{i,}^{j,k}$ represents the pheromone concentration between task j and task k in group i ant colony after mutation operator. Rdcr is a random value between [0,1] and CR is a constant with a value of [0,1] cross operation. The larger the value is, the greater the probability of cross operation. Rdk is a random integer between [0, n-1] to ensure that at least one element in the new pheromone array U_i comes from the element in the variant pheromone array V_i. Otherwise, the new pheromone array will not have any changes, which will weaken the exchange of pheromone arrays in the ant colony.

$$u_i = \begin{bmatrix} u_i^{0,0} & u_i^{0,1} & \cdots & u_i^{0,n-1} \\ u_i^{1,0} & u_i^{1,1} & \cdots & u_i^{1,n-1} \\ \vdots & \vdots & \ddots & u_i^{j,n-1} \\ u_i^{n-1,0} & u_i^{n-1,1} & \cdots & u_i^{n-1,n-1} \end{bmatrix} \tag{19}$$

$$u_i^{j,k} = \begin{cases} \tau_i^{j,k}, if & rdcr \le CR \quad or \quad rdk = k \\ v_i^{j,k}, if & rdcr > CR \quad or \quad rdk \ne k \end{cases} \tag{20}$$

The last parameter is selection operator. We select the pheromone as shown in formula (21) and use the obtained pheromone as the legacy pheromone in updating the global pheromone O_i. Among them, Mincost1 is the minimum total cost obtained by selecting tasks according to the probability obtained by using the new pheromone array U_i generated by the crossover operator. Mincost0 is the minimum cost obtained by selecting tasks according to the probability obtained by the pheromone array T_i at the beginning.

$$o_i = \begin{cases} u_i, \quad if & MinCost1 < Min\cos t0 \\ \tau_i \quad if & MinCost1 \ge Min\cos t0 \end{cases} \tag{21}$$

4.3 Task search

Task search means that an ant uses roulette algorithm to select the next task. The total cost will be calculated by combining with the scheduling model, and the resource of the lowest total cost will be selected at the same time. The resulting task scheduling sequence and corresponding resource sequence are a job scheduling solution. Roulette algorithm is based on task state transition probability. The state transition probability $P_{i,j}$ means The state transition probability from task i to task j. It is a general form in the definition of ant colony algorithm shown as formula (22).

$$p_{i,j} = \begin{cases} \dfrac{\left[\tau_{i,j}\right]^{\alpha}\left[\eta_{i,j}\right]^{\beta}}{\sum_{d \in AllowT}\left[\tau_{i,j}\right]^{\alpha}\left[\eta_{i,j}\right]^{\beta}}, & if \quad j \in AllowT \\ 0, & else \end{cases} \tag{22}$$

$T_{i,j}$ is the pheromone concentration between task i and task j. $N_{i,j}$ is the heuristic function, which reflects the expectation of task i when choosing the next task. In order to make the selection of task j with low total cost and short completion time, it will be shown as formula (23):

$$N_{i,j} = 1/ (r * CmpleteTime_j) \tag{23}$$

While r is the adjustment factor of task completion time, and $CmpleteTime_j$ is the completion time of task j, including waiting time and execution time, which is the minimum 8completion time estimated by calculation. Parameter α is the information heuristic parameter, representing the importance of pheromones. Parameter β is the expectation heuristic parameter, representing the importance of task completion time. Parameter AllowT is a set of tasks that have not been selected, which is often said in ant colony algorithm except other tasks in the table. The roulette algorithm can be realized by using this probability, so that the next task can be selected until all tasks are selected. When a task is selected, all resource should be traversed. The total cost of the task running on it should be calculated, and the resource with the lowest cost should be matched with this task.

The pseudo code of the task search algorithm is shown in algorithm 1.

Algorithm 1. Pseudo code of task search algorithm

Output: Pheromone array trail between tasks of the ith ant colony

Current minimum estimated completion time of each task

Input: The order of task, the order of corresponding resource

Procedure: Search(Trial,i)

In [0,n)select a random task ,select a vm that make minimal cost;

taskCount=1;

While(TaskCount<N)// Using roulette algorithm to get the next task

selectTask←selectNextTask(Trial,i);

mincost (vmlist, select task); // Get the virtual machine with the lowest current cost

taskCount++;

Update the state and cost of the task;

Procedure: selectNextTask(Trial,i)

sumPro=0.0; // Calculate the transfer probability of task I and the sum of the probabilities of all unselected tasks sumpro

for(i=0;i<n++i)

if(task i not selected)

according to formula (3-24), get the possibility of i :p(i);

sumPro +=p(i);

else

p(i)=0;

// Select task randomly according to the wheel

selectTask=-1;

if(sumPro>0)

tmpPro=rand(0.0,sumPro);

for(i=0;i<n++i)

if(task i not selected)

tmpPro -= p(i);

if(tmpPro<0.0)

selectTask=i;

break;

if(selectTask==-1)

selec←tTahsek first one of nonselected tasks ;

get selectTask;

5. EXPERIMENTAL RESULTS

5.1 Experimental setup

In order to prove the effectiveness of the method proposed in this chapter, we use matlab experimental platform to analyze the experimental results to verify the correctness, effectiveness and superiority of the scheduling strategy based on IACO algorithm. The experimental simulation results of the two scheduling algorithms are obtained by setting the same experimental environment configuration. We will compare ACO algorithm with IACO algorithm in the total cost, task completion time, task economic cost and load balance degree. We set ant number as 60, group number as 6, iteration number as 100.

5.2 Performance comparison and analysis

The time value is generally lower than the economic cost value, and it is easy to see the effect by setting the time weight to a high point. Therefore, the cost parameter is set as follows: a = 0.7, b = 0.3, c = 0.0, d = 0.0.

The iterative performance of the intelligent optimization algorithm makes the execution result uncertain, which may make the job completion time of IACO greater than that of ACO. Therefore, take 20-100 tasks and execute 5 for each algorithm. The total cost, task completion time and average economic cost were calculated and then compared, as shown in Figure 2, 3 and 4.

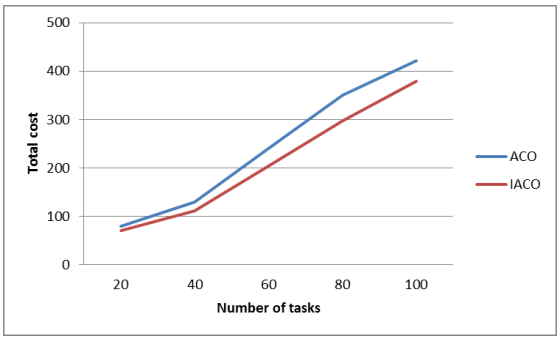

Figure 2. The total cost of ACO and IACO with different number of tasks.

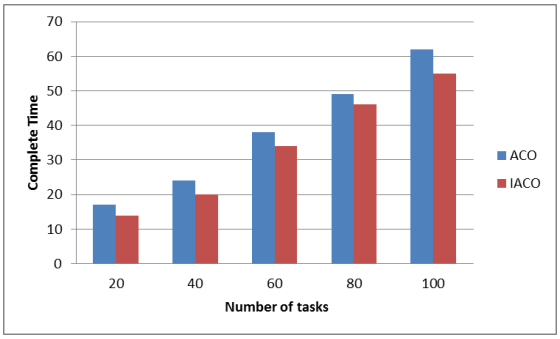

Figure 3. The complete time of ACO and IACO with different number of tasks.

There are five groups of data of job completion time obtained from the experiment, which can be seen as follows: the completion time obtained by ACO is shorter than that obtained by IACO. However, the average result of five times shows that from 20 to 100 tasks, IACO is slightly better than ACO. At the same time, the total cost and economic cost are reduced, which means the effectiveness and certain superiority of the algorithm.

In addition to focusing on users' QoS, system performance, such as load balancing, resource utilization, reliability and so on, should also be considered. When the system performance is good, it can provide users with continuous and reliable services. We set IACO with cost parameters of a = 0.7, b = 0.3, c = 0, d = 0, and ACOI with parameters of a = 0.3, b = 0.1, c = 0.5, d = 0, respectively. Each task number will be executed 5 times to calculate the average value of load balance. The result is shown as Figure 5.

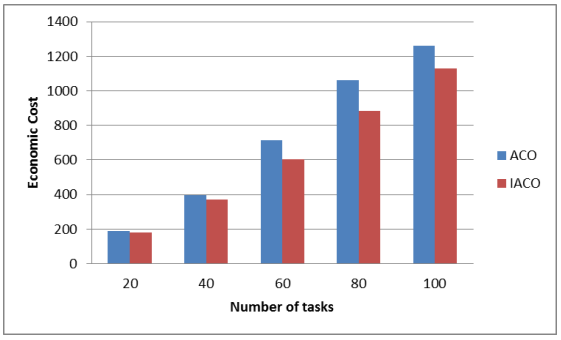

Figure 4. The economic cost of ACO and IACO with different number of tasks.

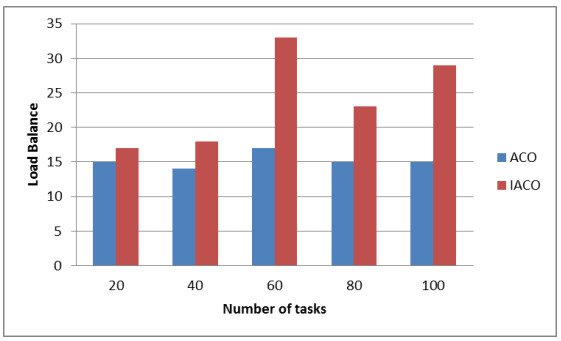

Figure 5. The load balance of ACO and IACO with different number of tasks.

In terms of load balancing, this paper uses the standard deviation of virtual machine running time to measure the degree of load balancing. When the load balance weight factor is 0.5, the load balance degree of the system is better than that when the load weight factor is 0. When the load weight factor is the same as 0.5, the load balance effect of IACO is better than that of ACO. Considering the load balance of the system from the aspect of job scheduling, it is likely to reduce the number of migration in the later stage. From the above analysis, we can see that this algorithm has a certain reference value for cloud service providers to improve system performance.

From Figure 2 to 5, it can be found that the total cost, completion time and economic cost of IACO are lower than those of ACO. However, the degree of load balance, total execution time and reliability may not be improved, because there is no corresponding weight factor control, which has certain randomness. Due to the higher unit price of resources with better performance, this paper pays attention to the economic cost while the economic cost index is reduced. At the same time, the improvement of operation execution time is predictable. Through the above experimental simulation comparison, the correctness and effectiveness of the job scheduling method proposed in this paper are verified from the perspective of users and cloud service providers.

6. CONCLUSIONS

At present, fog computing has become a research hotspot at home and abroad. Among them, the effectiveness of job scheduling strategy affects the user experience, the operation of fog computing data center and the economic benefits of service providers. Therefore, how to schedule user job requests to available resources is very important and meaningful.

From the perspective of users and fog service providers, aiming at minimizing the total cost, a job scheduling algorithm named IACO in fog computing environment is proposed. This algorithm not only achieves this goal, but also takes into account the quality of service (QoS) of users, such as the completion time and cost of jobs, and the load balancing of virtual resources in fog computing. We have verified the effectiveness of this job scheduling strategy on MATLAB platform.

This paper puts forward the corresponding algorithm and strategy for the research of job scheduling in fog computing environment. However, further research and exploration are needed in the future. It can be considered that there is a sequential dependency between tasks or requests, and the network communication overhead involved in the

relationship. The relevant excellent and mature technologies can be applied to job scheduling. At the same time, it also can be used to reduce the energy consumption of fog computing data center and improve the profit of service providers.

ACKNOWLEDGEMENT

This work was supported in part by the National Natural Science Foundation of China under Grant 61662038.

REFERENCES

[1] Lei, Y., Cai, Z, "Dynamic scaling of virtual clusters with bandwidth guarantee in cloud datacenters," In Proceedings of the IEEE INFOCOM, 2016.
[2] Armbrust, M., Fox, A., Griffith, R., Joseph, A. D., Katz, R., Konwinski, A., Lee, G., Patterson, D., Rabkin, A. and Stoica, I., "A view of cloud computing," Int. J. Comput. Technol, 50-58(2013).
[3] Lei, Y., Shen, H., Cai, Z., Ling, L., Pu, C., Lei, Y., Shen, H., Cai, Z., Ling, L. and Pu, C. "Towards bandwidth guarantee for virtual clusters under demand uncertainty in multi-tenant clouds," IEEE Trans. Parallel Distributed System, 450-465(2018).
[4] Lei, Y., Shen, H., Sapra, K., Lin, Y. and Cai, Z, "CoRE: Cooperative end-to-end traffic redundancy elimination for reducing cloud bandwidth cost," IEEE Trans. Parallel Distributed System, 446-461(2017).
[5] Al-Fuqaha, A., Guizani, M., Mohammadi, M., Aledhari, M. and Ayyash, M, "Internet of things: A survey on enabling technologies, protocols and applications, " IEEE Commun. Surv. Tutor., 2347-2376(2015).
[6] Shahid, S., Ismawati, J. N., Shamshul, B., et al, "Sentiment Analysis of Big Data: Methods, Applications, and Open Challenges, " IEEE Access, 37807-37827(2018).
[7] Ahsam, U., Bais, A, "A Review on Big Data Analysis and Internet of Things," IEEE International Conference on MobileAd Hoc and Sensor Systems, 325-330(2016).
[8] Sumi, L, Ranga, V, "Sensor Enabled Internet of Things for Smart Cities," Fourth International Conference on Parallel, Distributed and Grid Computing, 295-300(2016).
[9] Sarkar, S., Misra, S, "Theoretical Modelling of Fog Computing: A Green Computing Paradigm to Support IoT Applications," The Institution of Engineering and Technology Networks, 23-29(2016).
[10] Aazam, M., Huh, E. N, "Fog Computing and Smart Gateway Based Communication for Cloud of Things," IEEE Future Internet of Things and Cloud, 27-29(2014).
[11] Liu, J., Mao, Y., Zhang, J., et al, "Delay-Optimal Computation Task Scheduling for Mobile-Edge Computing Systems," IEE Symposium on Information Theory, 1451-1455(2016).
[12] Zhao, T., Zhou, S., Guo, X., et al, "Tasks Scheduling and Resource allocation in Heterogeneous Cloud for Delay-bounded Mobile Edge Computing," IEEE International Conference on Communications, 1-7(2017).
[13] Hoang, D., Dang, T, D, "FBRC: Optimization of Task Scheduling in Fog-based Region and Cloud," IEEE Trustcom/BigDataSE/ICESS, 1109-1114(2017).
[14] Li, T., Wu, M., Zhao, M, "Consumption Considered Optimal Scheme for Task Offloading in Mobile Edge Computing," IEEE International Conference on Telecommunications, 1-6 (2016).
[15] Tao, X., Ota, K., Dong, M., et al, "Performance Guaranteed Computation Offloading for Mobile-Edge Cloud Computing," IEEE Wireless Communications Letters, 774-777(2017).
[16] Yang, Y., Zhao, S., Zhang, W., et al, " DEBTS: Delay Energy Balanced Task Scheduling in Homogeneous Fog Networks," IEEE Internet of Things Journal, 1-1(2018).
[17] Zhu, Q., Si, B., Yang, F., et al, "Task Offloading Decision in Fog Computing System," China Communications, 59-68(2017).
[18] Liang, K., Zhao, L., Zhao, X., et al, "Joint Resource Allocation and Coordinated Computation Offloading for Fog Radio Access Networks," China Communications, 136-144(2016).
[19] Al-Fuqaha, A., Guizani, M., Mohammadi, M., et al, "Internet of Things: A Survey on Enabling Technologies, Protocols, and Applications," IEEE Communications Surveys and Tutorials, 2347-2376(2015).
[20] Hoang, D., Dang, T. D, "FBRC: Optimization of Task Scheduling in Fog-based Region and Cloud," IEEE Trustcom/BigDataSE/ICESS, 1109-1114(2017).
[21] Tao, X., Ota, K., Dong, M, et al, "Performance Guaranteed Computation Offloading for Mobile-Edge Cloud Computing," IEEE Wireless Communications Letters, 774-777(2017).