An Ant Colony Optimization Fuzzy Clustering Task Scheduling Algorithm in Mobile Edge Computing

Jianwei Liu[1], Xianglin Wei[2](✉), Tongxiang Wang[1], and Junwei Wang[1]

[1] Army Engineering University of PLA, Nanjing 210007, China
[2] Nanjing Telecommunication Technology Research Institute, Nanjing 210007, China
wei_xianglin@163.com

Abstract. Mobile edge computing has always been a key issue in the development of the mobile Internet and the Internet of things, how to efficiently schedule tasks has gradually become the focus of mobile edge computing research. Task scheduling problem belongs to the NP-hard optimization problem. Many traditional heuristic algorithms are applied to deal with the task scheduling problem. For improving the problem that ant colony algorithm has slow convergence speed, an ant colony optimization fuzzy clustering algorithm is proposed in this paper. In this algorithm, the fuzzy clustering algorithm is used to reduce the search space range in order to reduce the complexity of the scheduling algorithm and the number of iterations. And the optimal solution of the scheduling is found using the strong global search ability of ant colony algorithm. The simulation results show that the performance of the ant colony optimization fuzzy clustering algorithm is better than that of the First-Come-First-Served algorithm and the traditional ant colony optimization algorithm.

Keywords: Mobile edge computing · Task scheduling · Ant colony optimization algorithm · Fuzzy clustering algorithm

1 Introduction

With the rapid development and popularization of mobile Internet and the Internet of things (IoT), various applications such as augmented reality (AR) and virtual reality (VR) has greatly expanded the functions of mobile devices. Meanwhile, IoT businesses such as smart city, environmental monitoring, and intelligent agriculture are emerging. However, due to the limited storage, computing and perceived capabilities of mobile devices, mobile applications face many challenges in quality of service (QoS), mobility management and security. In this case, a new computing paradigm, i.e. Mobile Edge Computing (MEC), has been paid much attention by researchers due to its wide employment in several scenarios, such as IoT and Vehicular Network.

A typical mobile edge computing architecture consists of MEC servers and mobile devices [1, 2]. Compared to mobile devices, MEC servers are deployed at the edge of the network and have more available resources, which can process some computation-intensive, storage-intensive or latency tolerant tasks. And mobile devices can reduce their own computing load and energy consumption by offloading some tasks to MEC servers. Therefore, it is very important to design an efficient task scheduling strategy under MEC architecture. Previous works have investigated heuristic algorithms to deal with scheduling problem in MEC such as genetic algorithm and ant colony optimization (ACO) algorithm. However, traditional heuristic algorithms have some shortcomings such as high complexity, long iteration time and local optimum. When a large number of tasks need to be processed, the traditional scheduling algorithm is used to iterate for too long and cannot meet the requirements of tasks' deadline, which affecting mobile device user's quality of experience (QoE). Moreover, When the traditional task scheduling algorithm selects the resource, its target is the total service resource, which does not consider the characteristics of the resource itself and the preferences of the user or task, which leads to the large cost of the resource selection and the blind interest. Therefore, it is important to find a suitable method to reduce the resource selection range, thereby reducing the task scheduling spending.

Inspired by the above motivations, an ant colony optimization and fuzzy clustering (ACOFC) algorithm is proposed in this paper. Firstly, according to the tasks' preference for resources, the fuzzy clustering algorithm is used to divide the resources in order to reduce the space of the resource search and the complexity of the algorithm. Afterwards, the ACO algorithm is used to find the optimal scheduling solution in the divided spaces. At last, a series of numerical tests are conducted to illustrate the performance of the ACOFC algorithm is better than that of the First-Come-First-Served algorithm (FCFS) and the ACO algorithm.

The rest of paper is organized as follow. Section 2 overviews related studies on task scheduling. the system model is presented and, the joint scheduling problem is formulated in Sect. 3. The calculations of collision probability and access delay are discussed in Sect. 4. Section 5 presents a series of numerical test. Finally, the contributions of this paper are concluded in Sect. 6.

2 Related Work

In the previous work, Wei et al. formulated the joint scheduling problem as a 0-1 knapsack problem for the deadline-constraint tasks and the ACO-based method was employed to maximize the profits for the service providers [3]. In [4], They put forward a scheduling algorithm based on MAX–MIN Ant System (MMAS) considering multi-dimensional resources. Then, the load balancing of mobile devices was considered when scheduling the results [5]. Later, Wang et al. modified the algorithm while taking the types of tasks with deadline-constraint into consideration [6].

Although amounts of heuristic traditional scheduling algorithms have been proposed to scheduling problems with different targets, how to improve the traditional algorithm to solve scheduling problem is rarely studied in the MEC system. In [9], Wang et al. proposed an improved genetic algorithm method where the min-min and max-min algorithm are used to boost the search efficiency. For improving ACO algorithm's shortcomings in task scheduling, a task scheduling algorithm based on simulated annealing ant colony Algorithm is put forward in [10]. In this algorithm, the local optimal solution is constructed by ACO algorithm, the strong local search ability of simulated annealing algorithm (SA) is utilized to avoid algorithm trapped in local optimum at a certain probability.

3 System Model and Problem Formulation

3.1 MEC System Model

Before introducing the scheduling problem, our system model is presented in Fig. 1. The service providers, including MEC servers and mobile devices, can run the tasks concurrently. MEC servers are attached to the access point of the mobile devices to achieve low response latency, and mobile devices can offload tasks to the mobile servers. Both mobile devices and MEC servers have limited resources. MEC servers have more available resources than mobile device so that the processing time for task on MEC servers is shorter. Assume that each task cannot be further partitioned. The general working process is as follow:

- Each mobile device sends offloading requests to the decision maker in MEC servers by the wireless access network.

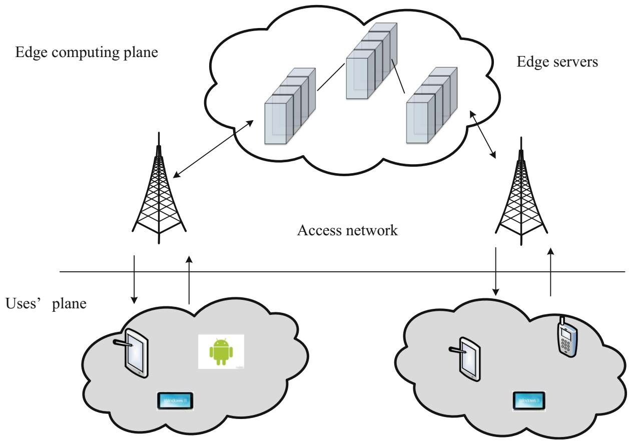

Fig. 1. System model

- According to the collected offloading requests of all the mobile devices and the property of each task, the decision maker executes the scheduling algorithm to decide where should the tasks be processed (in the mobile device locally or in the MEC server).
- The mobile devices offload the chosen tasks to the MEC servers.
- The MEC servers handle the tasks received and send their results to the mobile devices.

3.2 Problem Formulation

Assume that the resource dimension is three dimensions the resource for each MEC server i can be represented as a vector $\overrightarrow{c}_i = (c_i^{comp}, c_i^{stor}, c_i^{comm})$, where c_i^{comp}, c_i^{stor} and c_i^{comm} represents computing resources, storage resources and communication resources respectively. The set of tasks to be scheduled is represented as $I = \{1, 2, \ldots, n\}$, the value of each task in I is expressed as $(p_1, p_2, \ldots, p_n)$. And when the task i is running on the MEC server j, the resource that it consumes is represented as a vector $\overrightarrow{r_{ij}} = (r_{ij}^{comp}, r_{ij}^{stor}, r_{ij}^{comm})$. Assuming that all the task scheduling requests reach the decision maker at the same time and each task can only be offloaded to one MEC server. Under the constraints of the resources of each MEC server, the scheduling goal is to maximize the total profits, namely:

$$\text{Maximize} \sum_{j=1}^{n} \sum_{i=1}^{m} p_j \times x_{ij} \tag{1}$$

$$\text{Subject to:} \sum_{j=1}^{n} x_{ij} \leq 1, \; i = 1, 2, \ldots, m \tag{2}$$

$$\sum_{i=1}^{m} \overrightarrow{r_{ij}} \times x_{ij} \leq \overrightarrow{c_i}, \; j = 1, 2, \ldots, n \tag{3}$$

$$x_{ij} \in \{0, 1\}, \; i = 1, 2, \ldots, m; j = 1, 2, \ldots, n \tag{4}$$

From the above scheduling problem, we can see that this is a three-dimensional 0-1knapsack problem, and is NP-hard.

4 Tasks Scheduling Algorithm

The simple ACO algorithm has the advantages of parallelism, synergism and positive feedback, but when solving the complex NP-hard problem of task scheduling, the algorithm has some shortcomings such as poor local search ability, easy to fall into the local optimal solution and slow convergence speed.

For solving this NP-hard scheduling problem, we propose an ACOFC algorithm, which can be widely utilized to solve this kind of combinational optimization problem. At first, fuzzy clustering algorithm is used to divide the tasks to be scheduled can be divided into three categories by tasks' preference for resources: computational task set, bandwidth task set, and storage task set. Similarly, MEC servers can be divided into three categories: computational server set, bandwidth server set and storage server set. By using fuzzy clustering algorithm, the tasks and the MEC servers are divided respectively, which reduces the spatial space of the search and the complexity of the algorithm. Afterwards, Because of the strong global search ability of ACO algorithm, the ACO algorithm is used to find the optimal scheduling solution in the space of task set and MEC server set with the same resource preference.

4.1 Resource Fuzzy Clustering

Fuzzy c-means clustering method (FCM) is a data clustering method based on the optimization of objective function. Given the data set $X = \{x_1, x_2, \ldots, x_n\}$, the objective of FCM algorithm is to find a fuzzy partition cluster $U_{c \times n}$ of the data set, that is, divide the original data into c fuzzy groups, and minimize the non-similarity index between each group of samples and its clustering center. The general definition of clustering objective function is as follows.

$$J(U, c_1, \cdots, c_c) = \sum_{i=1}^{c} J_i = \sum_{i=1}^{c} \sum_{j}^{n} u_{ij}^{a} d_{ij}^{2} \tag{5}$$

where U_{ij} represents the membership degree of data j to fuzzy group i, the value of U_{ij} is between 0 and 1. c_i is the clustering center of fuzzy group i, and $d_{ij} = ||c_i - x_j||$ is the Euclidean distance between the i clustering center and the j data point. a $\in [1, \infty)$ is a weighted index.

4.2 ACO Tasks Scheduling Algorithm

After fuzzy clustering of tasks and MEC servers, the task set and MEC server set, which have the same resource preferences, are respectively taken as tasks to be scheduled and service providers for ACO algorithm.

The basic principle of ACO algorithm is to release pheromones on the path of the natural ants, and the following the concentration of the remaining pheromones in the path of the left pheromones is higher, and the higher the concentration of the remaining pheromones, the greater the probability of selecting the path, which gradually converts to the process of the optimal solution of the entire company.

Local Heuristic Information and Path Selection.
In the ACO algorithm, the path can be determined by pheromone concentration and local heuristic values. Resource consumption is an important factor influencing the

scheduling goals. Therefore, the local heuristic value can be determined by the remaining resources. This means that the corresponding path of a host with a larger remaining resource has a higher priority. ACOFC algorithm considers the task profit and the resource consumption of the task, and defines the local heuristic information as

$$\eta_i(t) = \frac{h_i}{\sum_{j=1}^{m} \left| \frac{\overrightarrow{c_i}}{\overrightarrow{c_i} - \overrightarrow{r_{ij}}} \right|} \tag{6}$$

where h_i represents the profit of task i. And the probability that ant q selects the next task i is

$$P_{ij}^{q} = \begin{cases} \frac{\left[\tau_{ij}(t)\right]^{\alpha}\left[\eta_{ij}(t)\right]^{\beta}}{\sum_{k \in allowedq(t)} [\tau_{ik}(t)]^{\alpha}[\eta_{ik}(t)]^{\beta}} \\ 0, \qquad otherwise \end{cases} \tag{7}$$

where α and β are the weighted factors of the pheromone and heuristic value. $\tau_{ij}(t)$ presents the pheromone concentration on the path from i to j at time t.

Pheromone Update.
After each cycle of ACO algorithm, the pheromone needs to be updated according to the partial solutions. At first, the pheromone applied on the link would decline at a certain rate according to the volatile characteristics of pheromone. Besides, the solutions will increase its value for the ants. Assume that $\tau_{ij}(t)$ will be updated to be $\tau_{ij}(t+1)$ at time t +1.

$$\tau_{ij}(t+1) = (1-\rho)\tau_{ij}(t) + \Delta\tau_{ij}(t) \tag{8}$$

where ρ represents the evaporation rate of the ant. $\Delta\tau_i(t)$ is the increment of the pheromone, which can be determined according to the partial solution. The calculation of $\Delta\tau_j(t)$ is:

$$\Delta\tau_{ij}(t) = \sum_{q=1}^{Q} \Delta\tau_{ij}^{q}(t) \tag{9}$$

where Q is the ants' number. $\Delta\tau_{ij}^{q}(t)$ is determined by the partial solution at time t:

4.3 Algorithm Description

Algorithm: ACOFCalgorithm

```
1: For tasks and MEC servers
2: Set the number of clusters c, the maximum number of iterations T, accuracy of convergence
   $\varepsilon$ and initialize the membership matrix $U_0$
3:     If the current iteration number $t \le T$ and $\|U_t - U_{t-1}\| \le \varepsilon$
4:         Calculate the distance between the centroid and the data point;
5:         Calculate the FCM objective function;
6:         Update the membership matrix $U_t$;
7:     End
8: End
9: According to the obtained membership matrix, the task and MEC servers are divided into
   different categories
10: For each group of the task set and MEC server set which have the same resource
        preference.
11:        For each cycle of ACO
12:            For each ant
13:                Determine the available links between tasks and MEC servers according to the
                   resource's consumption;
14:                Place pheromone on paths based on the path's priority;
15:                If there is at least one path
16:                    Calculate local heuristic value for each path;
17:                    Choose the path with highest selection probability;
18:                    Add the path to the partial solution;
19:                    Update the paths set and remaining resource of hosts;
20:                End
21:            Calculate the profit of the partial solution;
22:            Update the best value of the solution;
23:            Update the pheromone of each path;
24:        End
25:    End
    End
26: Return the total optimal value.
```

5 Numerical Tests

In this section, we conduct a series of numerical tests to analyze the ACOFC algorithm by MATLAB simulator. In order to better evaluate the performance of the ACO algorithm, two typical scheduling algorithms, i.e. FCFS algorithm and ACO algorithm are utilized to compare with our proposed algorithm under different experimental settings.

In this simulation, the dimension of the resource is three, which represents computing resources, communication resources, and storage resources. There are *m* MEC servers and *n* tasks in the MEC system. Assume that these tasks arrive at the same time and each task has a large demand for a certain resource. The task's consumption of three-dimensional resources is in the range [a_1, a_2] on a MEC server. And the three-dimensional resource capacity of MEC servers is evenly distributed within the interval [a_3, a_4]. The profit of the task is subject to uniform distribution [a_5, a_6] within the interval. The number of loops is set to 5. The main parameters employed in the simulation process are illustrated in Table 1.

Table 1. Related parameters

Parameters	Value	Parameters	Value
α	1	a_2	50
β	1	a_3	50
n	100	a_4	100
m	20	a_5	10
ρ	0.3	a_6	40
a_1	10		

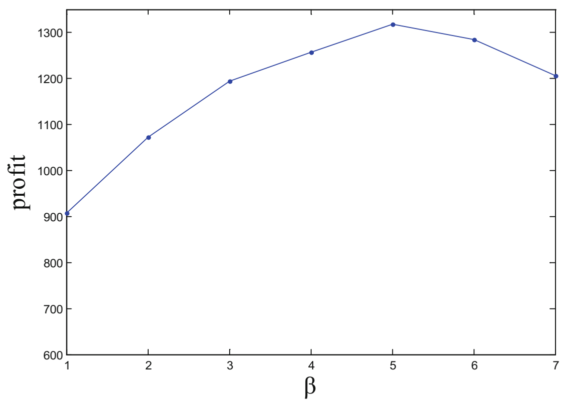

Fig. 2. Comparison of total profit for different β

Figure 2 shows the impact of β, which is defined as the local heuristic value in the path section process, on the result of ACOFC algorithm. The *x*-coordinates are the β, and the *y*-coordinates are the total profit. As β increases, the total profit increases with it linearly. While total profit is little reduced when β is larger than 5.

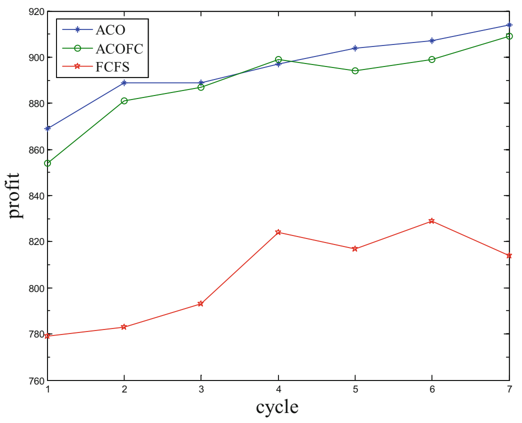

Fig. 3. The total profit varies with cycle for ACO, ACPFC and FCFS

The performance of ACOFC algorithm is illustrated in Fig. 3. by comparing with ACO algorithm and FCFS algorithm. the *x*-coordinate represents the cycles and the *y*-coordinate represents the total profit. From the figure, we can see that the total profit of ACOFC algorithm is close to that of ACO algorithm.

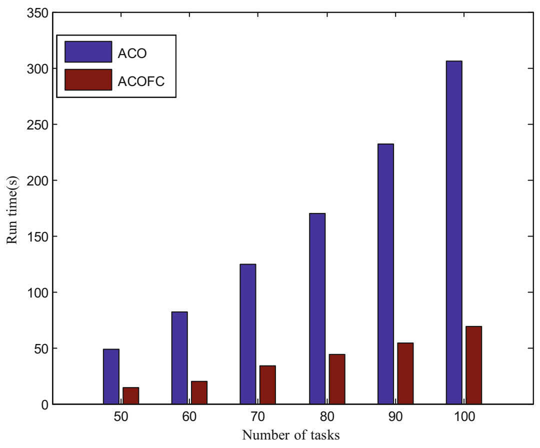

Fig. 4. Comparison of run time for different number of tasks

In Fig. 4, the *x*-coordinates are the number of tasks, and the y-coordinates are the run time. We compare the run time of ACPFC algorithm with that of ACO algorithm under different number of tasks. We can see that the running time of ACOFC algorithm is much shorter than ACO algorithm, especially in the case of a large number of tasks to be scheduled

6 Conclusion

In this paper, by combining the ACO algorithm and the fuzzy clustering algorithm, we have proposed an ant colony optimization fuzzy clustering algorithm to deal with a NP-hard scheduling problem. At first, according to the tasks' preference for resources, fuzzy clustering algorithm is used to divide the resources in order to reduce the space of the resource search and the complexity of the algorithm. Afterwards, the ACO algorithm is used to find the optimal scheduling solution in the divided spaces. At last, a series of numerical tests are conducted to illustrate the performance of the ant colony optimization fuzzy clustering algorithm is better than that of the FCFS algorithm and the ACO algorithm.

References

1. Mao, Y., You, C., Zhang, J., et al.: A survey on mobile edge computing: the communication perspective. IEEE Commun. Surv. Tutor. **19**(4), 2322–2358 (2017)
2. Abbas, N., Zhang, Y., Taherkordi, A., et al.: Mobile edge computing: a survey. IEEE Internet Things J. **PP**(99), 1 (2017)
3. Wei, X., Fan, J., Lu, Z., et al.: Bio-inspired application scheduling algorithm for mobile cloud computing. In: Fourth International Conference on Emerging Intelligent Data and Web Technologies, pp. 690–695. IEEE Computer Society (2013)
4. Wei, X., Fan, J., Wang, T., et al.: Efficient application scheduling in mobile cloud computing based on MAX—MIN ant system. Soft. Comput. **20**(7), 2611–2625 (2016)
5. Wei, X., Fan, J., Lu, Z., Ding, K.: Application scheduling in mobile cloud computing with load balancing. J. Appl. Math. **2013**(3), 337–366 (2013)
6. Wang, T., Wei, X., Tang, C., et al.: Efficient multi-tasks scheduling algorithm in mobile cloud computing with time constraints. Peer-to-Peer Netw. Appl. **5**, 1–15 (2017)
7. Ravi, A., Peddoju, S.K.: Mobility managed energy efficient Android mobile devices using cloudlet. In: Students' Technology Symposium, pp. 402–407. IEEE (2014)
8. Deng, J., Wang, Y., Dong, Z.: Dynamic trajectory pattern mining facing location prediction. Appl. Res. Comput. **34**(10), 2984–2988 (2017). (In Chinese)
9. Wang, T., Liu, Z., Chen, Y., et al.: Load balancing task scheduling based on genetic algorithm in cloud computing. In: Control Conference. IEEE (2016)
10. Hao-Rong, Z., Ping-Hua, C., Jian-Bin, X., et al.: Task scheduling algorithm based on simulated annealing ant colony algorithm in cloud computing environment. J. Guangdong Univ. Technol. **31**, 77–82 (2014)