Software-Defined Task Scheduling Strategy Towards Edge Computing

Yue Guo, Junfeng Hou, Heng Wang, Changjin Li(✉), Hongjun Zhang, Guangxu Zhou, and Xudong Zhao

Xuchang Cigarette Factory, China Tobacco Henan Industrial Co., Ltd., Xuchang 461000, Henan, China

Abstract. Data processing has posed new challenges to transmission bandwidth and computing load under the existing cloud computing architectures. In this paper, a software-defined task cooperative scheduling structure is proposed towards edge computing. Specifically, a clustering algorithm based on two-way selection between idle users and overloaded users is firstly designed, which combines the historical information of idle users and the interest similarity of overloaded users to form the stable cooperative clusters. Then, a sub-task partitioning algorithm based on the optimal delay is presented to achieve the overall optimal delay with the guarantee that sub-tasks are completed simultaneously. Numerical results show that the proposed strategy is not only able to save the data transmission bandwidth significantly, but also achieve the optimal delay while ensuring the stability of cooperative clusters.

Keywords: Edge computing · Task scheduling · Software-defined

1 Introduction

Cloud computing can greatly improve the efficiency of data processing by offloading computation-intensive tasks to cloud servers with rich storage and computing resources [1]. However, the geographical and logical long-distance characteristics of users and cloud servers may lead to higher delay and error probability in communication. In order to solve this problem, mobile edge computing (MEC) is considered as a potential solution [2, 3]. MEC can reduce task completion delay effectively through migrating computing tasks from the cloud data center to the devices at the edge of network, like domiciliary WiFi routers and gateways. However, the thousands of billions access users and sensor nodes and the diversity of emerging task types in the future network scenario would result in the restriction on the tasks migrated to mobile edge servers due to MEC resources. Edge servers with more idle resources contribute to reduce computing delay, but may suffer from long communication delay due to unstable factors of communication links.

Researchers have proposed numerous solutions to this problem. Kong et al. [4], based on the characteristics of moving edge calculation, established a fine-grained task division model of directed acyclic graph, analyzed the problem of minimum energy

J. Xiong et al. (Eds.): MobiMedia 2021, LNICST 394, pp. 197–211, 2021.
https://doi.org/10.1007/978-3-030-89814-4_15

consumption and proposed a genetic algorithm to find the optimal solution, and finally obtained the migration decision results of sub-tasks. Considering the general three-layer fog computing network architecture and the mobility patterns of users, Wang et al. [5] jointly optimized the offloading decision and resource allocation, which was modeled as a mixed integer nonlinear programming problem. Then, the GCFSA selection algorithm and distributed resource optimization algorithm was proposed to solve the task offloading and resource allocation, respectively. Aiming to minimize delay with the constraints on power, a one-dimensional searching algorithm was proposed in [6] to obtain the optimal task scheduling strategy, where Markov decision process was utilized to solve the inevitable dual-time scale stochastic optimization problem in task scheduling strategy. In the process of pursuing high task execution efficiency, these methods only pay attention to the scheduling of tasks among servers and the allocation of resources among users, and lack the development of potential computing resources at the user level. The exponential increase in the number of users leads to a sharp increase in base station load. However, not every user equipment needs to offload tasks to edge servers. Reasonably arranging idle users and exploring the huge computing resources at the user level can effectively reduce the load on the edge server side and avoid uplink congestion. Therefore, how to reasonably distribute and schedule tasks according to the users' social relationship and resource distribution is the key issue in this paper.

In response to the above problems, this paper proposes a user cooperative computing task scheduling strategy based on the software-defined network framework in edge computing. Firstly, a cooperative cluster is established for each overloaded user, and idle users select a cluster to join according to the strength of the social relationship with the overloaded users in each cluster. Then, in-cluster tasks is scheduled, where resource weights of idle users are ranked followed by determining the task scheduling order according to the resource ranking, and an optimization equation is established. Finally, select the appropriate cooperative computing user through traversing the number of users who have joined the cooperative computing and calculate the task segmentation vector.

2 System Architecture and Network Model

The system architecture of this paper is divided into two layers, i.e. the terminal equipment layer at the bottom and the edge layer at the top, as shown in Fig. 1. The terminal equipment layer mainly includes the smart terminals used in people's daily life, such as mobile phones, tablet. Some computation-intensive tasks generated by the aforementioned smart terminals cannot be accomplished locally. Therefore, it is necessary to formulate a reasonable scheduling strategy for them to migrate the tasks to the suitable place for completing the tasks while meeting the task delay requirements. The other equipment of the terminal equipment layer are composed of idle terminal users who have no task to process. At certain moments, their CPU is in an idle state, which can provide computing resources for end users whose CPU is overloaded.

The edge layer mainly consists of macro base stations equipped with edge servers which contains SDN control module, user information management module (UIMM) and user fairness scheduling module (UFSM). UIMM is responsible for recording the basic information of users in the coverage area, mainly including the remaining amount

of resources of idle users, the attribute information of the requested tasks, the social attributes of all user terminals and the spectrum resource situation in the coverage area. On the other hand, UFSM is responsible for executing the corresponding algorithm under the task requirements of the corresponding terminals according to the statistical information collected by UIMM to obtain the scheduling strategy achieving the optimal target indicator in the entire area. SDN control module forwards the scheduling policy results of all tasks in this area to the corresponding bottom terminals through the control plane according to the results of UFSM operation.

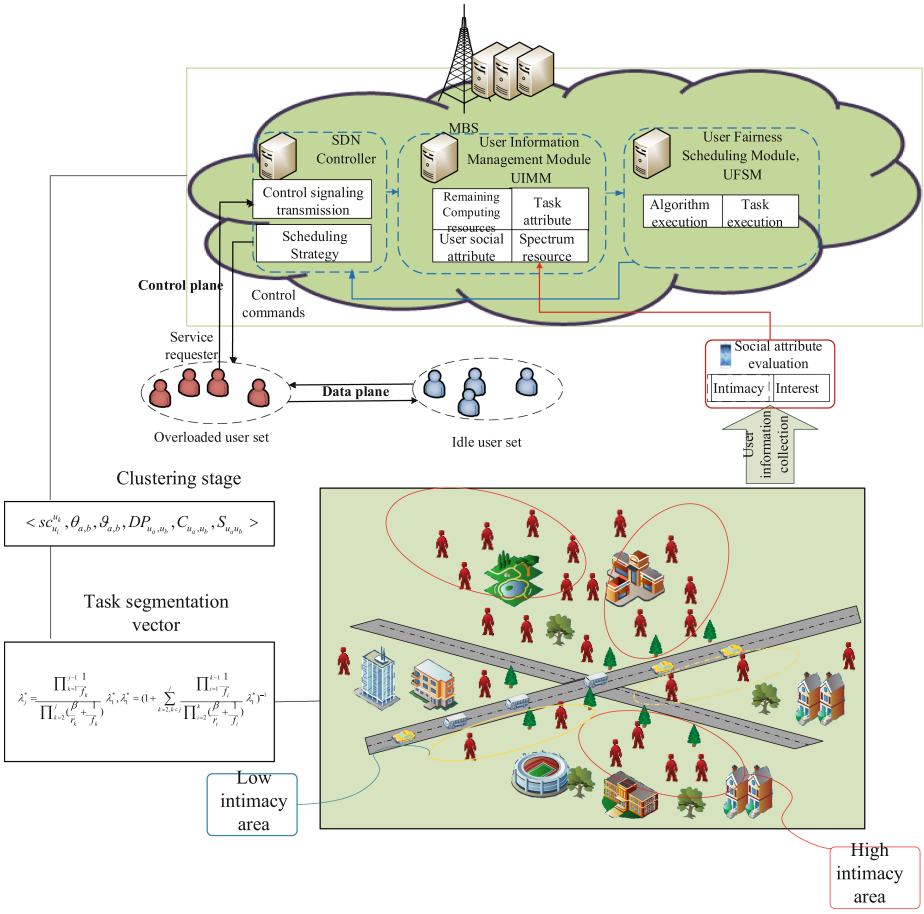

Fig. 1. System architecture

The terminal device layer is composed of multiple single-antenna users in this paper. Multiple end users are not only the generator of the tasks, but also the participants to assist in the execution of tasks. Denote the sets of terminals generating tasks and executing tasks as $M = \{1, 2, ..., m\}$ and $N = \{1, 2, ..., n\}$, respectively. The terminals generating tasks can select appropriate multiple idle users since a user is allowed to communicate

with multiple users, and offload tasks to the selected users for execution through the wireless communication links.

In the scenario designed in this paper, servers execute a clustering algorithm based on the historical information of overloaded users and idle users. Thus, terminals with similar social attributes would be clustered into a cluster to perform the subsequent in-cluster task scheduling process. Assume that the overloaded user i and N_i idle users are clustered into a cluster, user i has a computation-intensive task to be completed, denoted as $T_i(s_i, c_i, t_i)$, where s_i indicates the amount of input data of this task, c_i is the number of CPU cycles required to complete this task, and t_i is the task delay constraint. Note that the above information is also recorded in UIMM module of edge servers. Suppose that tasks can be divided into multiple sub-tasks and assigned to multiple idle users. For simplicity, it is assumed that the task granularity has arbitrary precision and there is no overlap between sub-tasks. Define the size of the data offloaded to the idle user j as $\lambda_j s_i$, where $\lambda_j \in [0, 1]$ indicates the task amount allocated to user j, and let $\lambda = [\lambda_1, \lambda_2, ..., \lambda_j]$ be the task allocation vector. This paper uses a time-sequential offloading method to transmit the sub-tasks one after another as these sub-tasks are independent. Note that the delay caused by the backhaul links can be ignored since the output data size of each sub-task is much smaller than the input data.

3 Collaborative Cluster Formation Algorithm Based on Social Attributes

An in-cell user clustering algorithm is introduced in this section. Specifically, the willingness, revenue, and similarity of both idle users and overloaded users are quantitatively considered to form a cluster according to the recorded user information, and then two-way selection is executed. In-cluster users perform the tasks of overloaded users through cooperative computing. The formation of the cooperative cluster is divided into two steps: the first step is to select candidate cooperative nodes for overloaded users based on the historical cooperative information of idle users, the feedback evaluation mechanism and cooperative benefits; in the second step, idle users perform inverse selection based on the strength of their social relationship with overloaded users, and voluntarily select the user clusters they want to join.

3.1 Rating Feedback Mechanism Based on Historical Information

It is the basic demand of overloaded users to select suitable and high-reputation users for auxiliary computation. This section evaluates the reputation of users based on the service history information of each user, and on this basis, proposes a rating feedback mechanism, which is used as the criterion for users to choose to join the cooperative cluster.

Suppose user i in the cell has assisted other users in completing tasks in history, and record the total z_i times assists as $F = \{hc_1, hc_2, ..., hc_{z_i}\}$. Every time a user has been assisted, the assisted user will conduct a service evaluation on user i, and the evaluation feedback score is recorded as sc_i^k. The higher the score, the better the service quality

provided by the user, and the corresponding score is directly uploaded to the UIMM module.

The level of the evaluation score depends on the task completion efficiency, which is manifested in the difference between the delay of completing the task and the original required delay of the task. This paper quantifies the evaluation score in the open interval of (0, 1). Specifically, it is quantified as 0.5 points when the actual task completion delay is equal to the task delay constraint; the score is required to increase from 0.5 to less than 1 when user i helps k complete the task at a faster speed; otherwise, user i would get a relative low score of 0–0.5. In order to meet the above requirements, the specific evaluation formula is shown in formula (1):

$$sc_{u_i}^{u_k} = \frac{2^{\frac{(t_k - to_{u_i,u_k})}{t_k}}}{2} \tag{1}$$

Where $sc_{u_i}^{u_k}$ represents the score obtained by user i assisting user k in completing the task, to_{u_i,u_k} represents the time it takes for user i to assist user k to complete the task, t_k represents the delay constraint of task k. Because some idle users have strong social relations and willingness to cooperate, but often have less resources to provide, they may get a low score. In order to weaken the impact of this situation on the user's score, this paper updates the user's score as shown in Eq. (2):

$$SCO_{u_i}^{u_k} = \frac{\sum_{u_i \in uc_i} sc_{u_i}^{u_k} \cdot \frac{c_{u_i}^{u_k}}{c_i}}{\sum_{u_i \in uc_i} sc_{u_i}^{u_k}} \tag{2}$$

Where uc_i represents the set of users assisted by user i. At the same time, in order to dynamically update the trust score of each user, the evaluation score of each time is processed by an exponential attenuation mechanism. The specific operation is shown in formula (3):

$$SCO_{u_i}^{u_k *} = \sum_{i=1}^{k} e^{1-i} \cdot SCO_{u_i}^{u_k} \tag{3}$$

To sum up, this mechanism can ensure that the score is true and effective to feedback the service status that users can provide.

3.2 Evaluation of Collaboration Benefits

In this paper, the parametric collaboration rate of return is defined to measure the revenue brought by users when they provide computing resources. The specific quantization process is as follows.

For a group of users a and b, if a is an overloaded user, that is, its own computing resources cannot meet the task requirements, and b is an idle user, then the task distribution between them can be divided into two parts: 1) User b is relatively idle; 2) The user is also overloaded. The computational resource relationship corresponding to the above two cases is as follows:

$$\theta_{a,b} = \begin{cases} \frac{f_b^l - \frac{c_b}{t_b}}{\frac{c_a}{t_a} - f_a^l}, & \frac{c_a}{t_a} > f_a^l \text{ and } \frac{c_b}{t_b} < f_b^l \\ 0, & \frac{c_a}{t_a} > f_a^l \text{ and } \frac{c_b}{t_b} > f_b^l \end{cases} \tag{4}$$

When $\theta_{a,b}$ is 0, it means that user b cannot provide effective assistance to user a; when $\theta_{a,b} > 1$, it means that user b's resources can meet the resources that user a lacks. When $0 < \theta_{a,b} < 1$, it means that user b alone cannot meet the resource needs of user a, and the assistance of more users is needed at this time.

In addition, the communication range between users is also another factor that measures the cost of collaborative computing. If two users are not within the communication range of each other, no matter how appropriate their computing resources are, they cannot provide assistance to each other. According to the requirements of D2D communication, a reliable connection can be established only when the distance between users a and b meets certain requirements. Therefore, parameter $\vartheta_{a,b}$ is defined in this paper to measure whether reliable communication is conducted between users. The specific definition is shown in formula (5):

$$\vartheta_{a,b} = sel(\gamma_{u_a,u_b}, \gamma_{u_a}^{th}) \tag{5}$$

Among them, $sel(a, b)$ is a selection function, when the parameter $a \geq b$, take a, and when $a < b$, take 0. Where γ_{u_a,u_b} represents the SNR formula of the channel between users a and b, which depends on the additive white Gaussian noise interference and the interference from the multiplexed cellular users to it. The formula is shown in formula (6):

$$\gamma_{u_a,u_b} = \frac{\left|h_{u_a,u_b}\right|^2 P_{u_a}}{\left|h_{m,u_a}\right|^2 P_m + \sigma^2} \tag{6}$$

$\gamma_{u_a}^{th}$ represents the SNR threshold that meets the requirements of the user's direct communication range. The specific calculation formula is shown in Eq. (7).

$$\gamma_{u_a,u_b}^{th} = 2^{\frac{s_a}{W(t_a - \frac{c_a - t_a f_a^l}{f_b^l - \frac{s_b}{t_b}})}} - 1 \tag{7}$$

To sum up, $\vartheta_{a,b}$ describes whether direct communication can be conducted between two user terminals. When they can communicate with each other, the larger $\vartheta_{a,b}$ indicates that the link is more conducive to communication. According to parameter $\vartheta_{a,b}$ and the collaborative yield rate between users, this paper defines the cost of collaborative computing between users as shown in Eq. (8):

$$DP_{u_a,u_b} = \frac{\vartheta_{a,b}}{\theta_{a,b}} \tag{8}$$

It can be seen from the above formula that in the case of meeting the requirements of direct communication, the higher the rate of return of collaboration, the lower the cost of collaboration between users.

3.3 Social Attribute Driven Bidirectional Selection Clustering Algorithm

The social relationship between users can be measured from two aspects. One is the social relationship affected by intimacy, and the other is the social relationship affected by interest similarity. This section will measure the strength of social relationships between users from these two aspects.

Intimacy Relationship

The definition of intimacy $C_{a,b}$ represents the strength of the social relationship between user terminals a and b under the influence of intimacy. The larger the value, the stronger the intimacy between them, the more likely they are to trust each other and assist in calculations. Intimacy $C_{a,b}$ is affected by different scene factors. The degree to which users with different social identities are affected by different factors shows obvious differences. In this paper, the intimacy between users is defined as the sum of the product of weight factor and intimacy factor. The weight factor reflects the degree of influence of the scene on the user's intimacy, specifically the ratio of the average interaction interval of the user to time T in the time period T, as shown in Eq. (9):

$$Q^i_{u_a,u_b} = \frac{\sum_{p=1}^{\mathrm{ct}} \mathrm{STA}_{p+1} - \mathrm{EDT}_p}{T \cdot \mathrm{ct}} \tag{9}$$

Where $Q^i_{u_a,u_b}$ represents the weight factor of the i-th influencing factor between users a and b, and represents the importance of the influencing factor. ct represents the number of interactions between two users in time period T, STA_{p+1} represents the start time of $p+1$ interaction, and EDT_p represents the end time of p interaction.

The total intimacy between users a and b can be calculated as shown in Eq. (10):

$$C_{u_a,u_b} = \sum_{i=1}^{n} Q^i_{u_a,u_b} \cdot \zeta^i_{u_a,u_b} \tag{10}$$

Where $\zeta^i_{u_a,u_b}$ represents the closeness of users a and b corresponding to influencing factor i. In this paper, interaction frequency is used to measure intimacy under this factor. The interaction frequency between each pair of user terminals can be quantified, and the specific formula is shown in (11):

$$\zeta^i_{u_a,u_b} = \frac{\alpha_{u_a,u_b}}{\sum_{b \in U} \alpha_{u_a,u_b}} \times \frac{\eta_{u_a,u_b}}{\sum_{b \in U} \eta_{u_a,u_b}} \tag{11}$$

Where α_{u_a,u_b} represents the number of content sharing between users a and b, and η_{u_a,u_b} represents the average time for content sharing between users a and b. In summary, the total intimacy between users a and b is recorded as:

$$C_{u_a,u_b} = \sum_{i=1}^{n} Q^i_{u_a,u_b} \cdot \zeta^i_{u_a,u_b} = \sum_{i=1}^{n} \frac{\sum_{p=1}^{\mathrm{ct}} \mathrm{STA}_{p+1} - \mathrm{EDT}_p}{T \cdot \mathrm{ct}} \cdot \frac{\alpha_{u_a,u_b}}{\sum_{b \in U} \alpha_{u_a,u_b}} \times \frac{\eta_{u_a,u_b}}{\sum_{b \in U} \eta_{u_a,u_b}} \tag{12}$$

Interest Similarity Perception

Assuming that there are m users in this area, denoted as $U = \{u_1, u_2, ..., u_m\}$, there are a total of v points of interest. Since users' interest in different locations is affected by time period, in order to better measure the degree of interest, this paper divides the time period of a day into several time periods of corresponding size, denoted as $\{\chi_1, \chi_2, ..., \chi_w\}$, and there are w time periods in total. Each user's interest in the corresponding place at time χ_w is denoted as $I^v_{u_i}(\chi_w)$. This value is determined by the number of visits and the length of the time period between the corresponding time and the previous time. The formula is as (13):

$$I^v_{u_i}(\chi_w) = \frac{\chi_w - \chi_{w-1}}{\phi_w} \tag{13}$$

where χ_w and χ_{w-1} denote the length of time between the current moment and the previous statistical moment, respectively, and ϕ_w denotes the number of visits to the location within the time range. The interest matrix M is defined as a $v \times w$ matrix. Each row of the matrix corresponds to a different location, and each column represents a different time period.

After obtaining the interest matrix of all users, the similarity of interest of different users in the places of interest can be calculated based on the interest matrix. Firstly, the overall error of the degree of interest between two end users is calculated according to the definition of covariance, as shown in Eq. (14):

$$\mathrm{cov}(u_a, u_b) = \frac{\sum_{i=1}^{w} I^v_{u_a}(\chi_i) \times I^v_{u_b}(\chi_i)}{w} - \frac{\sum_{i=1}^{w} I^v_{u_a}(\chi_i) \sum_{i=1}^{w} I^v_{u_b}(\chi_i)}{w^2} \tag{14}$$

Since different individual users manifest variability for different locations of interest, in order to measure the similarity of interest among different users, it is necessary to know the fluctuation of the difference of each user's interest in different locations. Calculate the square of the difference between the interest degree of each end-user at different points of interest and the average interest degree, and take the average of the squares to measure the degree of fluctuation of interest of different users in each interest point. The formula is shown in (15):

$$D(u_a) = \sqrt{\frac{1}{v} \cdot \sum_{i=1}^{v} (I^v_{u_a}(\chi_i) - \sum_{i=1}^{v} I^v_{u_a}(\chi_i)/v)^2} \tag{15}$$

Finally, combining formulas (14) and (15), the interest similarity is derived by calculating the ratio of the difference in interest between users and the product of the fluctuation of users' respective interests, and the formula is shown in Eq. (16):

$$\rho(u_a, u_b) = \frac{\frac{\sum_{i=1}^{w} I_{u_a}^{v}(\chi_i) \times I_{u_b}^{v}(\chi_i)}{w} - \frac{\sum_{i=1}^{w} I_{u_a}^{v}(\chi_i) \sum_{i=1}^{w} I_{u_b}^{v}(\chi_i)}{w^2}}{\sqrt{\frac{1}{v} \cdot \sum_{i=1}^{v} (I_{u_a}^{v}(\chi_i) - \sum_{i=1}^{v} I_{u_a}^{v}(\chi_i)/v)^2} \times \sqrt{\frac{1}{v} \cdot \sum_{i=1}^{v} (I_{u_b}^{v}(\chi_i) - \sum_{i=1}^{v} I_{u_b}^{v}(\chi_i)/v)^2}} \tag{16}$$

$$|\rho(u_a, u_b)| \leq 1$$

From the above equation, the interest similarity takes values within the range of absolute values less than or equal to 1 when $\rho(u_a, u_b) = 0$ indicates that users a, b have almost no points of common interest.

In summary, the social relationship between users in the edge computing network needs to be considered from two aspects. The higher the social intensity between users, the higher the probability that both parties will perform assisted computing. Therefore, considering the intimacy relationship and interest similarity between user terminals, the strength of social relationship between them is shown in Eq. (17), where τ denotes the weighting factor.

$$S_{u_a u_b} = \tau \cdot C_{u_a, u_b} + (1 - \tau) \cdot \rho(u_a, u_b) \tag{17}$$

From Eqs. (12) and (16), it can be seen that in different scenarios, the strength of social relationships can be determined by a combination of several factors such as intimacy relationship and interest similarity. Therefore, according to the weight proportion of various influencing factors, the importance degree of the two types of social relations in different scenarios can be represented, i.e., the weight factor τ is shown in Eq. (18), where Q_{u_a, u_b}^{i} is the weight factor of the i-th influencing factor between users a, b, and $\rho(u_a, u_b)$ is the interest similarity of users a, b.

$$\tau = \frac{\sum_{i=1}^{n} Q_{u_a, u_b}^{i}}{\sum_{i=1}^{n} Q_{u_a, u_b}^{i} + \sum_{a=1}^{N} \sum_{b=1}^{N} |\rho(u_a, u_b)|} \tag{18}$$

4 Collaborative Scheduling Strategy in Cluster

The optimization problem can be established as shown in Eq. (19):

$$
\begin{aligned}
&p1: \min_{\boldsymbol{\lambda}} T(\boldsymbol{\lambda}) \\
&s.t. \sum_{j=1}^{N_i} \lambda_j = 1 \ (a) \\
&0 \leq \lambda_j \leq 1 \ (b)
\end{aligned} \tag{19}
$$

The two constraints ensure that there is no overlap in the offloading of subtasks. The delay for each subtask is a non-convex problem due to the sequential transfer of tasks. In this paper, a heuristic algorithm is proposed to achieve a suboptimal solution of the non-convex problem. The specific process is divided into three steps.

4.1 Idle User Selection

In this paper, we mainly consider the link quality and computation rate, and make the selection by the collaboration factor that will define the candidates. The collaboration factor is defined as:

$$
w_j = \frac{\beta s_i}{r_{i,j}} + \frac{c_i}{f_j} \tag{20}
$$

This represents the delay of offloading the total task to user j.

4.2 Task Scheduling

Intuitively, minimizing the maximum value of a set of variables is balancing these variables. The purpose of task scheduling is to balance the execution delay of each selected idle user. Therefore, the scheduling policy is to balance the latency of each selected user. To balance the latency between idle users is shown in Eq. (21):

$$
\begin{aligned}
&\frac{c_i \lambda_1}{f_1} = (\frac{\beta s_i}{r_{i,1}} + \frac{c_i}{f_2})\lambda_2 \\
&\frac{c_i \lambda_2}{f_2} = (\frac{\beta s_i}{r_{i,2}} + \frac{c_i}{f_3})\lambda_3 \\
&..., \\
&\frac{c_i \lambda_{j-1}}{f_{j-1}} = (\frac{\beta s_i}{r_{i,j-1}} + \frac{c_i}{f_j})\lambda_j
\end{aligned} \tag{21}
$$

According to the set of Eqs. (21), for a given j idle users, the optimal task assignment decision is shown in Eq. (22):

$$
\lambda_j^* = \frac{\prod_{k=1}^{j-1} \frac{1}{f_k}}{\prod_{k=2}^{j} (\frac{\beta}{r_k} + \frac{1}{f_k})} \lambda_1^* \tag{22}
$$

$$\lambda_1^* = (1 + \sum_{k=2,k<j}^{j} \frac{\prod_{i=1}^{k-1} \frac{1}{f_i}}{\prod_{i=2}^{k} (\frac{\beta}{r_i} + \frac{1}{f_i})})^{-1} \tag{23}$$

More idle users can help reduce the computational latency of the overall task, but can lead to a higher risk of offload failure. Therefore, we must find the optimal number of selected idle users by traversing from 1 to j. The number of users that minimizes the overall task latency is the optimal number. The corresponding task assignment strategy is the optimal decision. Table 4.3 summarizes the heuristic algorithm, which has a worst-case complexity of $O(j^3)$.

5 Numerical Results and Analysis

The performance of the proposed algorithm is evaluated by using MATLAB in this section. In the simulation environment, a single-cell multi-user model is adopted to measure the collaboration scores of users according to their historical information; the channel model is established as a Gaussian channel to measure the efficiency of task migration between users, thereby calculating the benefits of collaboration between users. Finally, the implementation of the clustering algorithm is verified according to the trajectory information and social information of the participants of the InfoCom06 conference, where the specific track information includes the start and end time of interaction between users and the number of interactions. Meanwhile, users' social relationship information can also be obtained from their activity trajectories to measure their closeness and similarity of interests. In terms of parameters, the configuration of the transmission link is based on the LTE system, and the channel bandwidth is 10 MHz. To simplify the problem, the noise spectral density is set as −174 dBm/Hz. The user's CPU rate ranges from 10×10^9 cycles per second to 100×10^9 cycles per second. And the task data size is set to 50–100 mbits to represent computationally intensive tasks. The main simulation parameters are shown in Table 1.

Table 1. Simulation parameter settings

Parameter	Parameter value
Transmission bandwidth	5 MHz
D2D user transmission power	15 dbm
User history information score	[0,1]
Noise spectral density	−174 dbm/Hz
Number of edge users	[10,100]
CPU cycles required for the task	$[1,50]\times10^8$ cycles
User computing power	$[10,100]\times10^9$ cycles/s
Task data size	[50,100] MBits
Task delay	[0,100] ms
Channel fading factor	10^{-4}

The proposed MDGTSM heuristic algorithm is compared with two typical task scheduling mechanisms. The first is based on the cooperative task computing mechanism of nodes in the fog wireless access network [10] (Cooperative Task Computing, CTC). The CTC algorithm splits the subtasks to multiple candidate terminals in parallel, and meets the ultra-low latency requirements by combining the settlement capacity of multiple powerful fog wireless access nodes and the near-field communication at the edge; The second is a task scheduling algorithm based on the task calculation time prediction algorithm [11] (Prediction of Tasks Computation Time Algorithm, PCA), which uses principal component analysis (PCA) and reduces the expected time of calculation (ETC) matrix to greatly increase the delay and reduce the computation and complexity. Moreover, it is also compared with two baseline schemes: (1) one-to-one offloading scheme: offload the whole task to the user with the lowest weight among idle users; (2) all offloading scheme: all idle users are involved Task execution.

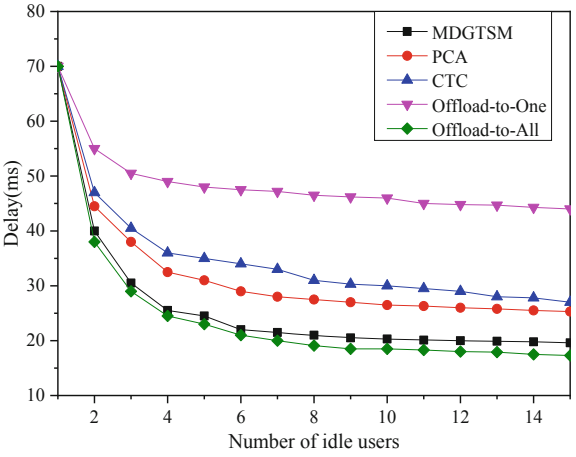

Fig. 2. Task delay with different number of idle users

Figure 2 shows the relationship between task delay and the number of idle users with different algorithms. As shown in the figure, the algorithm proposed in this paper achieves lower latency than the CTC algorithm and the PCA algorithm. Specifically, the algorithm proposed in this paper has similar performance to all offloading schemes, while the CTC algorithm has the highest latency.

Figure 3 depicts the relationship between the number of idle users and the probability of unloading failure. We can see that the offloading failure probability of the MDGTSM algorithm proposed in this paper is very close to that of the PCA algorithm, both of which are less than the failure probability of all offloading schemes but higher than that of one-to-one offloading. The CTC algorithm shows poor performance in this regard, because it tends to allocate more resource blocks for users with lower data rates.

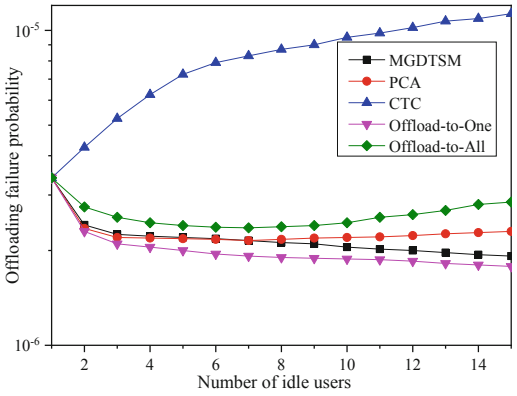

Fig. 3. Offloading failure probability with different number of idle users

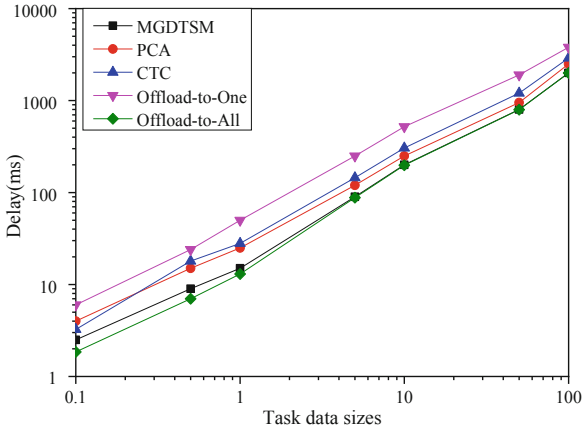

Fig. 4. Task delay with different task data sizes

Figures 4 and 5 describe the performance of the algorithm with different task data sizes with the total number of idle users set to 15. Obviously, the waiting time and the probability of offloading failure increase as the task data size increases. Specifically, as shown in Fig. 4, the MGDTSM algorithm and the PCA algorithm have better performance than the CTC algorithm. The CTC algorithm has the longest time delay among the three algorithms, while the MGDTSM and the PCA algorithm have similar waiting time delay and are both close to that of all offloading schemes. In Fig. 5, the CTC algorithm shows the worst offloading failure probability, while the performance of the other algorithms is very close to each other, especially at 50 Mbits.

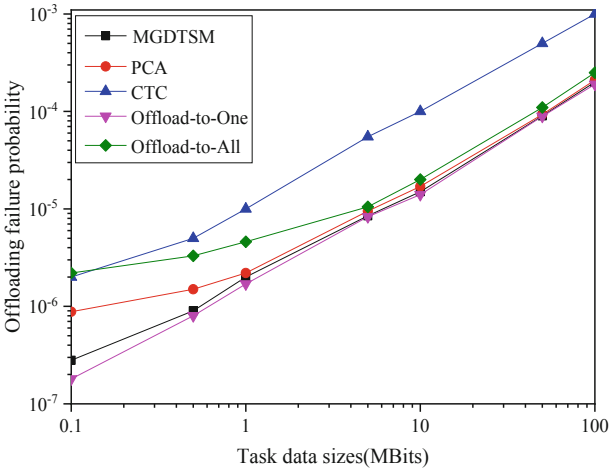

Fig. 5. Offloading failure probability with different task data sizes

6 Conclusion

In this paper, a task scheduling strategy for edge computing is presented. Firstly, the overloaded user makes positive selection according to the idle user's historical collaboration record and resource distribution. At the same time, idle users will also make reverse selection according to the social attributes of overloaded users to form a stable collaboration cluster. Secondly, the overload user in the cluster selects the appropriate user for unloading according to the link state and calculation rate of the idle user. The task is divided into subtasks which are offloaded in a reasonable sequence to ensure that the whole task is completed with low latency. Finally, the results show that the proposed strategy can effectively reduce the task completion delay, avoid the channel congestion, and optimize the service quality.

References

1. Armbrust, M., Fox, A., Griffith, R., et al.: A view of cloud computing. Commun. ACM **53**(4), 50–58 (2010)
2. Zhang, Q., Fitzek, F.H.P.: Mission critical IoT communication in 5g. In: Atanasovski, V., Leon-Garcia, A. (eds.) FABULOUS 2015. LNICSSITE, vol. 159, pp. 35–41. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-27072-2_5
3. Weisong, S., Jie, C., Quan, Z., et al.: Edge computing: vision and challenges. IEEE Internet Things J. **3**(5), 637–646 (2016)
4. Kong, Y., Zhang, Y., Wang, Y., et al.: Energy saving strategy for task migration based on genetic algorithm. In: 2018 International Conference on Networking and Network Applications (NaNA), pp. 330–336. IEEE Press, Piscataway (2018)
5. Dongyu, W., Zhaolin, L., Xiaoxiang, W., et al.: Mobility-aware task offloading and migration schemes in fog computing networks. IEEE Access **7**, 43356–43368 (2019)
6. Zeng Deze, G., Lin, G.S., et al.: Joint optimization of task scheduling and image placement in fog computing supported software-defined embedded system. IEEE Trans. Comput. **65**(12), 3702–3712 (2016)

7. Chen, X., Lei, J., Wenzhong, L., et al.: Efficient multi-user computation offloading for mobile-edge cloud computing. IEEE/ACM Trans. Netw. **24**(5), 2795–2808 (2015)
8. Wang, C., Yu, F.R., Liang, C., et al.: Joint computation offloading and interference management in wireless cellular networks with mobile edge computing. IEEE Trans. Vehicul. Technol. **66**(8), 7432–7445 (2017)
9. Wang, X., Zhang, Y., Leung, V.C.M., et al.: D2D big data: content deliveries over wireless device-to-device sharing in large scale mobile networks. IEEE Wireless Commun. **25**(1), 32–38 (2018)
10. Chiu, T.-C., Chung, W.-H., Pang, A.-C., et al.: Ultra-low latency service provision in 5g fog-radio access networks. In: 2016 IEEE 27th Annual International Symposium on Personal. Indoor, and Mobile Radio Communications (PIMRC), pp. 1–6. IEEE Press, Piscataway (2016)
11. Al-Maytami, B.A., Fan, P., Hussain, A., et al.: A task scheduling algorithm with improved makespan based on prediction of tasks computation time algorithm for cloud computing. IEEE Access **7**, 160916–160926 (2019)