D2D-Based Multi-relay-Assisted Computation Offloading in Edge Computing Network

Xuan Zhao, Song Zhang, Bowen Liu, Xutong Jiang, and Wanchun Dou(✉)

State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210023, China
{xzhao,songzhang,liubw,jiangxutong}@smail.nju.edu.cn, douwc@nju.edu.cn

Abstract. With the rapid development of edge computing, a large number of compute-intensive tasks are offloaded to the edge server, and computation offloading strategy has become a hot research topic. Because the deployment of edge servers is still in its infancy, there would be some service blind area. Through D2D technology, mobile devices in the blind area can obtain edge services with the help of relays. However, the existing methods usually select single relay to transmit tasks, and seldom consider the sensitivity of processing delay for compute-intensive tasks. When the available bandwidth of single relay is not enough to complete the data transmission in a limited time, it will produce a large delay, which seriously affects the quality of user experience. In view of this challenge, a D2D-based multi-relay-assisted computation offloading method is proposed. Technically, multiple mobile devices in the available area of edge service are used as the relays. Based on D2D technology, mobile devices in the blind area use the relays to transmit the computing task to the edge server to improve the application computing efficiency. A large number of experiments with real-world datasets have proved the feasibility and effectiveness of our method.

Keywords: Compute-intensive task · Computation offloading · Edge computing · D2D · Multi-relay

1 Introduction

With the rapid development of electronic technology and the large-scale popularization of mobile devices, complex applications to achieve various functions have sprung up. Compute-intensive applications characterized by high demand for computing resources, such as AR/VR, online games and so on, have received extensive attention [1,2]. However, due to the lack of computing resources and high energy consumption of mobile devices, the promotion of compute-intensive applications is facing great challenges [3]. Although traditional cloud computing can provide nearly unlimited computing resources for these applications, due to

H. Gao and X. Wang (Eds.): CollaborateCom 2021, LNICST 407, pp. 33–49, 2021.
https://doi.org/10.1007/978-3-030-92638-0_3

the time delay caused by physical distance and complex network links, some delay-sensitive compute-intensive tasks cannot be effectively handled [4,5].

Edge computing is a new computing model proposed in recent years. As a supplement to cloud computing, edge computing places part of the computing and storage resources on the edge of the network close to users [6]. This makes the computing tasks that need to be offloaded to the cloud can be directly processed at the edge of the network, which greatly reduces the task processing time and meets the demand of delay sensitive applications for low latency [7]. Therefore, the problem of computing offloading in edge computing environment has become the focus of industry and academia [8,9].

Because the deployment of edge devices is still in its infancy, there are a lot of edge service blind areas in real life. If the mobile devices in these blind areas want to use the computing and storage resources of the edge server, they can only use other mobile devices in the service scope of the edge server to assist the transmission. By establishing a direct network connection between devices, D2D technology can help mobile devices and relays in the blind area to establish a D2D transmission link and obtain edge service. At present, a lot of work has been done based on D2D technology [9–12].

However, the existing methods usually select single relay to transmit tasks, and seldom consider the sensitivity of processing delay for compute-intensive tasks. When the available bandwidth of single relay is not enough to complete the data transmission in a limited time, it will cause a large network delay at the application end, which seriously affects the quality of user experience [13]. Therefore, it is necessary to select multiple relays among the candidate relays and select the most reasonable computing task to be offloaded to the edge server for processing, so as to minimize the energy consumption of mobile devices and task execution time.

In view of this challenge, a D2D-Based multi-Relay-Assisted Computation Offloading method (named DRACO) is proposed. In this method, multiple mobile devices in the available area of edge service are used as the relays, and the mobile devices in the blind area use the relays to transmit the task data, and then offload the compute-intensive task to the edge server to improve the performance of compute-intensive tasks. A large number of experiments using real-world datasets are conducted and the results proved the feasibility and effectiveness of our method. The main contributions are summarized as follows.

- We study the task processing in the local and in th edge server and incentive mechanism for the relays, and build the task processing edge system model.
- We fully consider the energy consumption of system devices, the revenue of relays and task processing time, and design a multi-relay selection method to select the best relay set.
- A large number of experiments using real-world datasets are conducted and the results proved that our method significantly improved the satisfaction of helper devices while ensuring video quality of requester devices.

The remainder of the paper is organized as follows. In Sect. 2, a motivating example is introduced. In Sect. 3, we present the system model and problem

formulation of our computation offloading scenario. Section 4 describes our offloading method DRACO. Experimental results are presented in Sect. 5 to demonstrate the performance of our method, followed by Sect. 6 summarizing related work. Finally, Sect. 7 concludes the paper.

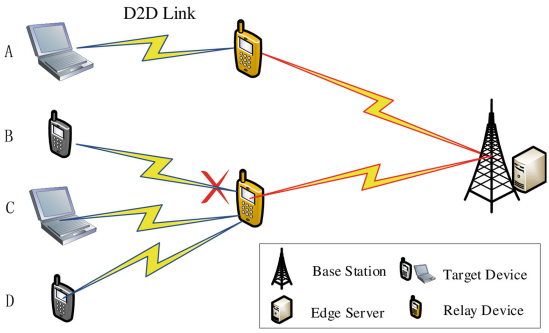


Fig. 1. Single relay cannot transmit multiple tasks at the same time.

2 Motivating Example

This section takes online games as an example to explore the motivation of our method. Large online games need high bandwidth to support the normal operation, especially AR/VR games. These games need to be rendered in real-time and the results are fed back to mobile users. At this time, the mobile device will continue to collect the data of the surrounding environment and transmit the data to the edge server for processing through the wireless network.

When the mobile device is in the blind area, it needs to use the relay to assist data transmission. However, AR/VR games receive and process the surrounding environment data in real-time. Therefore, it needs sufficient bandwidth resources to ensure the real-time completion of computing tasks. However, as a mobile device, the relay not only needs to assist the target device to transmit data, but also needs to process its local tasks, or transmit the local tasks to the edge server for processing. Therefore, it is very likely that a single relay can not complete the huge data transmission of compute-intensive tasks within the specified time, which will lead to task timeout.

Moreover, in the actual edge computing environment, there may be more than one mobile device in the blind area at the same time. As shown in Fig. 1, there are mobile devices A, B, C and D in the blind area, all of which need to transmit the computing task through the relay. Mobile devices B, C and D choose the same relay. At this time, since the compute-intensive task being executed by device B is a VR/AR game, it has a higher bandwidth requirement. However, the bandwidth provided by the relay is difficult to meet the needs of the task in device B. Therefore, the relay selection for device B is unreasonable. Multiple relays can provide more bandwidth resources.

In view of the above challenges, a D2D-based multi-relay-assisted computation offloading method is proposed. As shown in Fig. 2, when the target device in the blind area needs to offload the compute-intensive task to the edge server, the mobile device that can directly communicate with the edge server will be selected as the relay. However, the bandwidth resources provided by a single relay may not be able to complete the data transmission of compute-intensive applications. Therefore, the target device can select multiple relays to assist the target device in task transmission at the same time. Moreover, considering the transmission overhead and data allocation overhead that may be brought by relays, we only select relays for one hop distance assisted transmission, that is, target device –> relay(s) –> edge server. The relay will not transmit the data to other ralays for help.

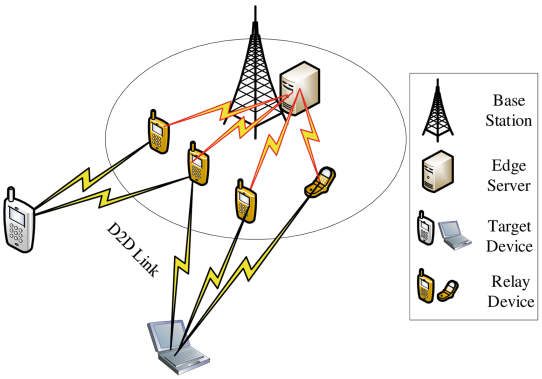


Fig. 2. D2D-based multi-relay-assisted computation offloading.

3 System Model and Problem Formulation

3.1 Local Model

In D2D computation offloading supporting compute-intensive tasks, the target device can select multiple relays to assist it in transmitting task data. In the process of assisting the target device to transmit data, the relays not only provide bandwidth resources but also consume the energy of target devices in the local.

The compute-intensive task to be processed in the target device is recorded as $m_i = (d_i, r_i, D_i)$. d_i is the amount of raw data of the m_i, r_i represents the CPU processing cycle required by m_i. D_i represents the expected completion time of m_i. The target device will divide the task m_i, part of which are offloaded to the edge server and others are executed locally. We use $\alpha_i \in [0, 1]$ to represent the proportion of task m_i that is offloaded to the edge server.

The time consumed by the local execution is not only affected by the computing power of the local device, but also related to the calculation backlog of the local device. For task m_i, the part to be executed locally is: $\mathcal{A}_i = (1 - \alpha_i)r_i$.

The computing tasks run at a fixed interval t_{in} continuously arrive at the target device. If the target device is limited by the computing performance of the local device and cannot be processed in time, these tasks will be overstocked locally, affecting the execution time of the tasks arriving later. When task m_i arrives, the target device's calculation backlog Q_i is:

$$Q_i = \begin{cases} 0, & i = 0 \\ \frac{max Q_{i-1} + \mathcal{A}_{i-1} - f_D t_{in}, 0}{f_D}, & i > 0 \end{cases} \tag{1}$$

Here, f^D is the computing power of the target device. The locally execution time

$$t_i^L = (1 - \alpha_i)\frac{r_i}{f^D} + Q_i \tag{2}$$

3.2 Edge Model

In the edge computing network, each base station is equipped with an edge server. Mobile devices need to go through the base station to access the edge server. Time of task m_i executed on the edge server is:

$$t_i^E = \alpha_i \frac{r_i}{\sum_k x_k^E f_k^E} \tag{3}$$

where f_k^E represents the computing power of the edge server accessed through base station k. $x_k^E = 1$ indicates that the edge server accessed through base station k is selected, otherwise, $x_k^E = 0$.

The relay set is $D^r = \{D_1, D_2, \cdots, D_j, \cdots, D_N\}$. For each relay, $D_j = \{\varphi_j, b_j, n_j\}$ where φ_j is the cost per unit time of using relay j to transmit compute-intensive tasks; b_j represents the bandwidth owned by relay j; n_j represents the number of tasks currently being transmitted by relay j.

When relay j is used to transmit task m_i, the transmission rate of relay j is

$$s_j = \frac{b_j}{n_j + 1} log_2(1 + \frac{p_j H_j}{\sigma^2}) \tag{4}$$

Here, p_j is the transmission power of relay j, H_j is the channel gain between the relay and the base station, and σ is the noise power. $\frac{b_j}{n_j+1}$ indicates that the tasks transmitted by the relay evenly divide the bandwidth of the relay. The transmission time for task m_i by relay j is

$$t_{i,j} = \frac{d_{i,j}}{s_j} + \frac{d_{i,j}}{s_{i,j}} \tag{5}$$

where $d_{i,j}$ denotes the amount of data of compute-intensive task m_i transmitted on relay j, and $\sum_{j=1}^{N} d_{i,j} = d_i$. $N = |D^r|$ indicates the number of relays in the relay set. $s_{i,j}$ is the transmission rate between the target device and relay j,

$$s_{i,j} = \frac{b_i}{N + 1} log_2(1 + \frac{p_i H_{i,j}}{\sigma^2}) \tag{6}$$

Here, b_i and p_i are the bandwidth and transmission power of the target device. $H_{i,j}$ is the channel gain between the target device and the relay.

The relay sets the transmission time of all relay to form the relay transmission time set $T^r = t_{i,1}, t_{i,2}, \cdots, t_{i,N}$. Therefore, the uplink transmission time for the target device to offload the task is: $t_i^r = \max\{T^r\}$. At the same time, we also need to consider the current backlog of transmission data in the transmission of compute-intensive tasks. This paper takes δ_i represents the transmission backlog, which is defined as when task m_i arrive, it takes time to wait for transmission, which is related to $i-1$ and previous tasks. Based on this, the time for task transmission can be calculated when the target device offloads the task:

$$t_i^T = x_i((1-y)\frac{d_i}{s_i} + yt_i^r + \delta_i) \tag{7}$$

where x_i is 0(if $\alpha_i = 0$) or 1(if $\alpha_i > 0$). y indicates whether a relay is selected. If the target device can complete the transmission without a relay, $y = 0$; Otherwise, $y = 1$. s_i is the transmission rate to the base station and can be calculated by $s_i = b_i \log_2(1 + \frac{p_i H_i}{\sigma^2})$ where H_i is the channel gain between the target device and the base station. It should be noted that when task m_i after reaching the target device, δ_i is a constant.

3.3 Incentive Model

As a relay, mobile devices not only occupy bandwidth resources, but also consume power. Therefore, in order to encourage more mobile devices to join the transmission process, the incentive model is considered for the transmission of relays. During the offloading, the relay can benefit from the transmission:

$$E^R = x_i \sum_{j=1}^{N} \varphi_j \frac{d_{i,j}}{s_j} \tag{8}$$

where φ_j is the cost per unit time when using relay j to transmit.

There is also a payment for the usage of edge servers. The cost for renting the computing resources is:

$$E^E = \varphi_k t_i^E \tag{9}$$

where φ_k is the cost of unit time when the task is offloaded to the edge server through base station k. The overall cost for task processing is $E^R + E^C$.

3.4 Computation Offloading Model

In computation offloading, the task m_i in the target device is to be processed execute in parallel locally and at the edge. Therefore, the task completion time

$$t_i = \max\{t_i^E + t_i^T, t_i^L\} \tag{10}$$

The energy consumed by the target device can be divided into two parts: 1) the energy consumed by local computing; 2) the energy consumed by transmitting data. Therefore, the local energy consumption for m_i can be calculated by

$$E^C = \alpha_i \varphi_i r_i + x_i p_i \frac{d_i}{S_i} \tag{11}$$

where φ_i is the energy consumed by local processing in one cycling time. p_i represents the energy consumed by task transmission.

We use μ_i and ψ_i to represent sensitivity of user to the task m_i and power consumption, respectively. Then the target device offloading utility for task m_i can be calculated as

$$U_i(\alpha_i, d_{i,j}, x_k^E) = \mu_i(D_i - t_i) - E^R - E^E - \psi_i E^C \tag{12}$$

Therefore, the optimization problems can be obtained

$$\max_{(x\in\{0,1\},y\in\{0,1\},0<d_{i,j}<d_i,x_k^E\in\{0,1\})} U_i(\alpha_i, d_{i,j}, x_k^E) \tag{13a}$$

$$\textbf{s.t.} \quad dist(j) \leq l(0 < j \leq N) \tag{13b}$$

$$\sum_{j=1}^{N} d_{i,j} = d_i \tag{13c}$$

$$\sum_{k} x_k^E = 1 \tag{13d}$$

Formula (13b) indicates that the distance between the target device and the relay j cannot exceed the maximum distance l of D2D communication. Formula (13c) represents collaborative transmission for the compute-intensive tasks by all relays. Formula (13d) ensures that only one edge server will be selected.

4 D2D-Based Multi-relay-Assisted Computation Offloading Method

4.1 Relay Selection Algorithm

When the target device is in the blind area, it needs to establish a D2D connection with the relay that can access the edge server to obtain the service of the edge server. Therefore, the target device needs to first confirm whether it is in the blind area, that is, whether it is within the service coverage of the edge server. This paper considers the case of edge server damage or downtime. In this case, although the mobile device can connect with the base station, the edge server equipped with the base station is difficult to provide external services. In this paper, base stations are denoted as $K = \{k_1, k_2, \cdots, k_m\}$. Algorithm 1 gives the specific process.

Algorithm 1 outputs the relay information ω. Specifically, the target device will traverse the base station set K. If the target device can connect to the base station, it will further test whether the edge server is available. If no edge server is available, return $y = 1$ and the relay set D. If the target device finds an available edge server, return $y = 0$ and $D = \emptyset$. At this time, the target device can directly access the edge server through the base station.

The target device obtains the relay set D through Algorithm 1. The target device will select one or more mobile devices according to the bandwidth and

Algorithm 1. Candidate relay sets construction

Require: Target device, relay set, base station set
Ensure: $\omega = \{y, D\}$
$y = 1$
for $k \in K$ **do**
  if Target device can connect to the base station k **then**
    if The edge server connected to base station k is available **then**
      $y = 0$
if $y == 0$ **then**
  $D = \emptyset$
else
  The target device obtains the information D_i of the mobile device that can establish a D2D connection
  Construct relay set $D = \{D_1, D_2, \cdots, D_M\}$
return ω

price of each relay to form the final relay set D_r, and $D_r \in D$. The target device will allocate the amount of data to each relay in the relay set, which is related to the utility that the target device can obtain.

We allocate the amount of data according to the scheme of minimum transmission time. The average speed of data transmission through relay j can be reached by

$$\bar{v_j} = \frac{d_{i,j}}{\frac{d_{i,j}}{s_j} + \frac{d_{i,j}}{s_{i,j}}} = \frac{s_j s_{i,j}}{s_j + s_{i,j}} \tag{14}$$

where $d_{i,j} = \frac{\bar{v_j}}{\sum_{k=1}^{N} \bar{v_k}} d_i$. In this way, the profit for relay j is

$$E^R = \sum_{j=1}^{N} \varphi_j \frac{d_{i,j}}{\bar{v_j}} = \sum_{j=1}^{N} \varphi_j \frac{d_i}{\sum_{k=1}^{N} \bar{v_k}} \tag{15}$$

If the target device adjusts the devices in the relay set and the amount of data transmitted, it needs to meet the requirement that the utility value improvement obtained by the target device after adjustment is greater than the cost of adjustment:

$$\Delta u^r = \mu_i \Delta t_i^T - \Delta E^R > 0 \tag{16}$$

where Δt_i^T represents the difference between the transmission time after policy adjustment and that before policy adjustment; ΔE^R represents the difference between the revenue of the relay after policy adjustment and that before policy adjustment.

Based on the strategy of transmission delay minimization, the data allocation $d_{i,j}$ of each relay can be obtained, but also need to get the final relay set D_r. Algorithm 2 gives an greedy-based iterative strategy. Based on this strategy, Algorithm 3 gives the relays selection algorithm. In Algorithm 3, D^i is used to represent the data allocation set: $D^i = \{d_{i,1}, d_{i,2}, \cdots, d_{i,N}\}$.

Algorithm 2. Greedy-based relay selection algorithm

Require: D, current utility u^r, transmission time t_i^T
Ensure: Relay set D_k
$\Delta u^r = -\infty, D^r = D, D_k = \emptyset$
for $D_i \in D^r$ **do**
  Calculate the utility improvement Δu_{temp}^r and time cost t_{temp}^T based on u^r
  if $\Delta u_{temp}^r > \Delta u^r$ **then**
    $D_k = D_i$; $t_i^T = t_{temp}^T$; $\Delta u^r = \Delta u_{temp}^r$
  if $\Delta u_{temp}^r == \Delta u^r$ and $t_{temp}^T < t_i^T$ **then**
    $D_k = D_i$; $t_i^T = t_{temp}^T$; $\Delta u^r = \Delta u_{temp}^r$
return D_k

Algorithm 3 gives the steps of selecting a relay set. The relay set selection algorithm allocates the transmission data intending to minimize the transmission time, and then selects the relay set based on the utility value. Initially, the algorithm takes the whole set of relays to be selected as the set of relays to calculate the transmission data allocation corresponding to the minimum transmission time. After that, the algorithm will remove the mobile devices with the least utility improvement in the set of relays to be selected in a greedy way every iteration, and calculate again until the set is empty. Finally, the target device selects the set with the largest utility value as the relay set.

Algorithm 3. Relay Set Selection Algorithm

Require: y, candidate relay set D
Ensure: Information of relay set $r = \{y, D^{max}, D^i, t_i^T\}$
$t_i^T = +\infty$; $D^r = D$
if $y == 0$ **then**
  $t_i^T = \frac{d_i}{s_i}$; $D^r = \emptyset$; $D^i = \emptyset$
else
  while $D^r \neq \emptyset$ **do**
    for $D_i \in D^r$ **do**
      $d_{i,j} = \frac{\bar{v_j}}{\sum_{k=1}^{N} \bar{v_k}} d_i$
    $t_{temp}^T = \frac{d_{i,j}}{\bar{v_j}}$; $E^R = \sum_{j=1}^{N} \varphi_j \frac{d_{i,j}}{\bar{v_j}}$
    if $D^r == D$ **or** $\mu_i \Delta t_i^T > \Delta E^R$ **then**
      $t_i^T = t_{temp}^T$; $D^{max} = D^r$; record D^i; $E_{temp}^R = E^R$
    else
      if $\mu_i \Delta t_i^T == \Delta E^R$ and $t_{temp}^T < t_i^T$ **then**
        $t_i^T = t_{temp}^T$; $D^{max} = D^r$; record D^i; $E_{temp}^R = E^R$
    Get D_k by Algorithm 2
    $D^r = D^r - D_k$
return r

4.2 D2D-Based Multi-relay-Assisted Computation Offloading

The construction method of candidate relay set is given in the previous. In this way, the target device can get the relay set D^r, as well as the transmission data amount D^i allocated by each relay. Considering the intensive deployment of edge servers, relays may be in the service coverage of multiple edge servers at the same time. Therefore, the target device also needs to determine the proportion of data uploaded to the edge server α_i and edge server K.

According to the connection between the relay to be selected and the edge server, the relay to be selected is divided first. For the set D of relays to be selected, it is divided into k subsets d according to the connection between the relays to be selected and the edge server $D_1^E, D_2^E, \cdots, D_K^E$. For the edge server equipped on the base station k, All mobile devices in D_k^E are accessible. K is the number of base stations that can be accessed by the selected relay. This chapter assumes that each base station is equipped with only one edge server, so K is also the number of edge servers. After partition, the set of relays to be selected meets the requirement of $\bigcup_{k=1}^{K} D_k^K = D$. Moreover, for any two subsets $D_{k_1}^K, D_{k_2}^K$ The intersection of the two is not necessarily empty.

After the division, the combination set $D^E = \{\{1, D_1^E\}, \{2, D_2^E\}, \cdots, \{k, D_k^E\}, \cdots, \{K, D_K^E\}\}$. $\{k, D_k^E\}\}$ represents the base station k and the set of relays to be selected that can access the edge server equipped on the base station k. To obtain the offloading strategy, the target device can traverse D^E to calculate the maximum utility that the target device can obtain by offloading the compute-intensive task to the edge server in D2D mode under the current situation. In this case, the edge server to be offloaded by the target device is determined, and the relay set and the transmission data assigned to each relay can be obtained by using Algorithm 3. Formula (12) is transformed into

$$f(\alpha_i) = \mu_i t_i^D - \mu_i t_i - E^E - E^R - \psi_i E^C \tag{17}$$

After determining the relay, C is a constant. We define

$$f_1(\alpha_i) = \mu_i t_i^D - \mu_i (t_i^E + t_i^{T'}) - E^E - E^{R'} - \psi_i E^{C'} (t_i^E + t_i^T \geq t_i^L) \tag{18}$$

$$f_2(\alpha_i) = \mu_i t_i^D - \mu_i (t_i^E + t_i^{T'}) - E^E - E^{R'} - \psi_i E^{C'} \quad (t_i^E + t_i^T \leq t_i^L) \tag{19}$$

$$f_3(\alpha_i) = \mu_i t_i^D - \mu_i (t_i^E + t_i^T) - E^E - E^{R'} - \psi_i E^C \quad (t_i^E + t_i^T \leq t_i^L) \tag{20}$$

where $t_i^{T'} = (1-y)\frac{d_i}{s_i} + y t_i^r$, $E^{R'} = \sum_{j=1}^{N} \varphi_j \frac{d_{i,j}}{s_j}$, $E^{C'} = \alpha_i \varphi_i r_i + p_i \frac{d_i}{S_i}$. Then we can get

$$\max f(\alpha_i) = \max \left(\max f_1(\alpha_i), \max f_2(\alpha_i), f_3(0)\right) \tag{21}$$

The utility maximization problem is transformed into the maximum problem of $f_1(\alpha_i)$, $f_2(\alpha_i)$ and $f_3(0)$. Moreover, from the definition of function, $f_1(\alpha_i)$ and $f_2(\alpha_i)$ are continuous. Furthermore, we decompose the above maximum problem into two subproblems: (1) If $t_i^E + t_i^T \geq t_i^L$, find the maximum value of $f_1(\alpha_i)$; (2) If $t_i^E + t_i^T \leq t_i^L$, find the maximum value of $f_2(\alpha_i)$. In the following section,

we will discuss the maximum problem in these two cases in turn. For ease of expression, we set $\alpha_1 = (Q_i + \frac{r_i}{f^D} - t_i^T)/r_i(\frac{1}{f_k^E} + \frac{1}{f^D})$.

For subproblem (1), we first calculate

$$f_1'(\alpha_i) = \frac{df_1(\alpha_i)}{d\alpha_i} = -\mu_i \frac{r_i}{f_k^E} - \varphi_k \frac{r_i}{f_k^E} - \psi_i \varphi_i r_i \tag{22}$$

From the definition of each variable in the formula, we can see that $f_1'(\alpha_i) \leq 0$. As $t_i^E + t_i^T \geq t_i^L$, $\alpha_i \geq \alpha_1$. If $\alpha_1 \in (0, 1]$, $\max f_1(\alpha_i) = f_1(\alpha_1)$.

For subproblem (2), we first calculate

$$f_2'(\alpha_i) = \frac{df_2(\alpha_i)}{d\alpha_i} = \mu_i \frac{r_i}{f^D} - \varphi_k \frac{r_i}{f_k^E} - \psi_i \varphi_i r_i \tag{23}$$

By $t_i^E + t_i^T \leq t_i^L$, we can get $\alpha_i \leq \alpha_1$. If $f_2'(\alpha_i) \geq 0$, $\max f_2(\alpha_i) = f_2(\alpha_1)$. If $f_2'(\alpha_i) < 0$, $\max f_2(\alpha_i) = f_2(0)$.

Through the above process, we can get the maximum value of $f_1(\alpha_i)$ and $f_2(\alpha_i)$ and further get the utility value $f(\alpha_i)$. Based on this process, a D2D-based multi-relay-assisted offloading strategy can be obtained. Algorithm 4 shows the process of solving the problem.

Algorithm 4. D2D-Based Multi-Relay-Assisted Computation Offloading

Require: m_i, μ_i, f^D, y_{final}, D
Ensure: Offloading strategy $\Omega = \{\alpha_{final}, D_{final}, k_{final}\}$
1: $u_{final} = -\infty$, $k_{final} = \emptyset$, task processing time t_i
2: **if** $y_{final} == 0$ **then**
3: $D^E = \{k, \emptyset\}$
4: **else**
5: D^E is obtained by dividing the candidate relays set
6: **for** $D_k^E \in D^E$ **do**
7: Execute Algorithm 3 to obtain relay set D_i and data allocation set D^i
8: Obtain the currently available base station k
9: $\alpha = \frac{Q_i + \frac{r_i}{f^D} - t_i^T}{r_i(\frac{1}{f_k^E} + \frac{1}{f^D})}$
10: Calculate the maximal value u of $f(\alpha_i)$ and task processing delay t
11: **if** $u_{final} < u$ or ($u_{final} = u$ and $t_i < t$) **then**
12: $u_{final} = u$, $t_i = t$, $\alpha_{final} = \alpha$, $D_{final} = D_i$
13: **return** Ω

Algorithm 4 will traverse the relay partition set, and each time, it will select the current local optimal solution based on maximum utility value, and then obtain the global optimal strategy Ω. The strategy includes task m_i's offloading ratio α_{final}, relay set D_{final}, base station k_{final}.

4.3 Complexity Analysis

In Sect 4.1, we first give the algorithm for constructing the candidate relay sets. The algorithm will traverse the surrounding base station that may establish communication with the target device, so the complexity is $O(|K|)$. Greedy-based device selection algorithm is a part of the relay set selection algorithm. The algorithm will traverse the candidate relay set, so the complexity is $O(|D|)$. The relay set selection algorithm also traverses the candidate relay set. In each round of traversal, greedy-based device selection algorithm also needs to select the devices that are not in the relay set, so the complexity is $O(|K| + |D|^2)$.

In Sect 4.2, we present a computation offloading algorithm to support compute-intensive tasks. The algorithm will first execute the construction algorithm to obtain the candidate relay sets. After that, the algorithm will divide the candidate relay sets. Based on the divided set D^E, the algorithm will find the final offloading strategy. During the execution of the algorithm, it will traverse the base station set, and the complexity of the algorithm can be obtained as $O(|K| + |D| + |D|^2|K|)$.

5 Performance Evaluation

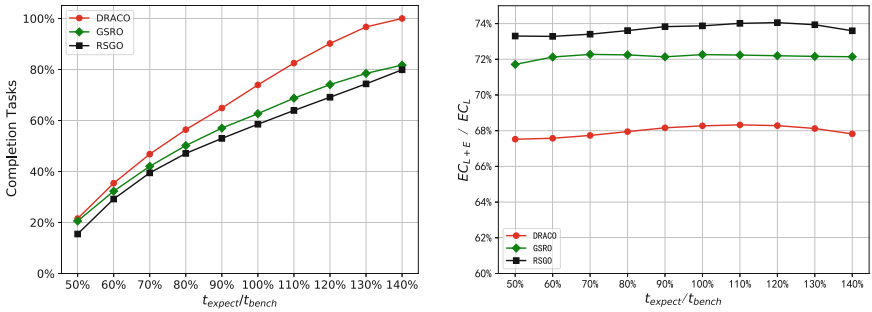
5.1 Simulation Setting

We select the Australian base station distribution dataset [14] which is derived from the radio communication license dataset issued by the Australian Communications and Media Authority. Twenty edge servers are chosen and the coverage radius of each edge server is randomly set to [450, 750]. Within the coverage of each edge server, there will be 10 randomly generated mobile devices for transmission assistance.

This experiment uses the following indicators to evaluate the performance of the three methods: (1) the proportion of tasks completed before the expected completion time; (2) The energy consumption of mobile devices accounts for the proportion of local execution energy consumption.

The task data transmission time in $[\frac{r_i}{f^D}, \frac{1.5r_i}{f^D}]$. We set $t_{bench} = \frac{r_i}{f^D} + \frac{2d_i}{S_i}$, and $50\% - 140\%$ of t_{bench} are expected completion time by tasks. The arrival of the task obeys Poisson distribution, and $\lambda = 1$. This experiment will first calculate the index value of each target device in turn, and finally calculate the average value of 10 target devices.

We compare our method with the following two methods: (1) Random Single Relay-Assisted Offloading (RSRO): it will randomly select a device from the set of relays as the relay. The target device will determine the offloading ratio according to the strategy of maximizing utility value. (2) Greedy-Based Single Relay-Assisted Offloading (GSRO): it will sort the devices and select the device with the highest transmission rate as the relay. The target device will determine the offload ratio according to the strategy of minimizing execution time.



(a) Completion tasks ratio

(b) Energy consumption ratio

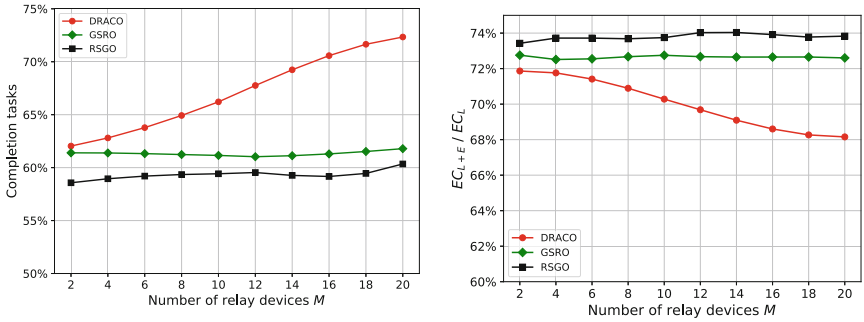
Fig. 3. Impact of the expected completion time.

5.2 Experimental Results

We first change the expected completion time. Figure 3(a) shows the number of tasks completed before the expected completion time of the user. With the increasing expected completion time, the performance of DRACO, GSRO and RSRO are constantly improving, but the number of tasks that the method DRACO has completed is always the most. Figure 3(b) shows the ratio of energy consumption of considering to un-considering computation offloading. The energy consumed by local devices using different methods is not related to the expected time change of task users, but the energy consumption of method DRACO is always the lowest of the three methods. As can be seen from Figs. 3(a) and 3(b), the performance of DRACO is always better than the other two methods, because a single relay may not be able to complete task processing before the user expected completion time.

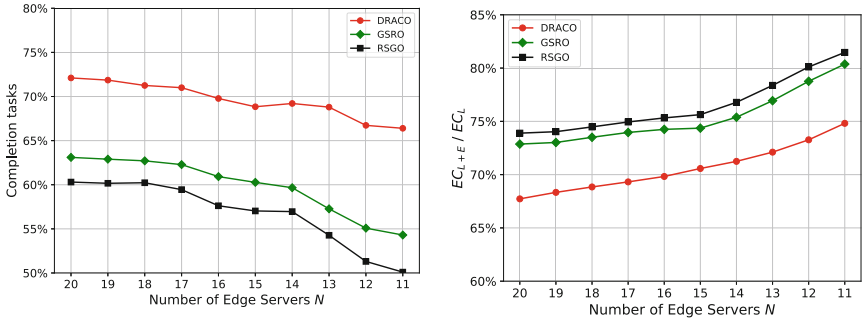
Then we change the number of relays to be selected from 2 to 20. Figure 4(a) shows that the proportion of completion tasks before the expected completion time. With the increasing number of relays to be selected, the number of tasks completed before the expected completion time with DRACO is gradually increasing. However, the number of tasks completed by GSRO and RSRO before the expected completion time has little change. The performance of DRACO is always the best. Figure 4(b) shows the ratio of energy consumption of considering to un-considering computation offloading. The energy consumption of DRACO is decreasing and remains the lowest at all times, but the energy consumption of GSRO and RSRO is unchanged. This is because GSRO and RSRO can only select one relay, so the improvement of the number of relays will not improve the performance of the algorithm. However, DRACO can use multiple relays for parallel data transmission at the same time, which greatly improves the performance of compute-intensive tasks.

Finally, we test the impact of the number of available edge servers. Figure 5(a) shows the completion tasks before the expected completion time. As the number of available edge servers is decreasing, the performance of DRACO, GSRO and



(a) Completion tasks ratio (b) Energy consumption ratio

Fig. 4. Impact of the number of relays.



(a) Completion tasks ratio (b) Energy consumption ratio

Fig. 5. Impact of the number of edge servers.

RSRO is decreasing, and the number of tasks completed before the expected completion time decreases. But the performance of DRACO is always the best. Figure 5(b) shows the ratio of energy consumption of considering to unconsidering computation offloading. With the decrease of the number of available edge servers, the energy consumption of DRACO, GSRO and RSRO for target devices is increasing, but the energy consumption of DRACO is always the lowest among the three methods. The reason for the above phenomenon is that with the decrease of the number of available edge servers, the single relay assisted offloading method is difficult to support more compute-intensive tasks, while the DRACO with multi-relay strategy can. Moreover, the continuous reduction of the number of available edge servers makes the target devices put more tasks on the local execution, so the energy consumption of mobile devices will continue to increase. However, DRACO is still superior to the other two methods.

6 Related Work and Comparison Analysis

Low latency, high bandwidth, close to users and other advantages make edge computing receive a lot of attention and research. With the rapid development of edge computing, more and more service providers try to offload large-scale computing tasks to edge servers. Computing offload strategy has become a research hotspot. Wang et al. [15] introduced a dynamic edge computing model and conducted the first study on robust task offloading which is tolerant to h server failures. Deng et al. [16] designed a user-centered joint optimization offloading scheme to minimize the weighted cost of time delay, energy consumption and price under the constraint of satisfying the advanced personalized needs of users. Song et al. [17] proposed a scheme in which tasks are offloaded to servers with the aim of maximizing a reward within a limited power budget, server processing capacities, and wireless network coverage. Yan et al. [18] proposed an MEC service pricing scheme to coordinate with the service caching decisions and control wireless devices' task offloading behavior in a cellular network. Based on the available bandwidth of heterogeneous edge severs and the location of mobile devices, Yang et al. [19] formulated an optimal offloading node selection strategy as a Markov decision process (MDP), and solved by employing the value iteration algorithm (VIA). Zhan et al. [7] investigated the problem of offloading decision and resource allocation among multiple users served by one base station to achieve the optimal system-wide user utility, which is defined as a trade-off between task latency and energy consumption. However, these methods seldom consider the devices in blind area that cannot communicate with the edge server directly. We use D2D technology to solve this problem by using multiple devices as relays. If the devices in blind area want to use edge services, they could offload computing tasks to edge servers through relay devices.

D2D technology improves the communication efficiency through the direct connection between devices, and also provides a new idea for data transmission. The establishment of efficient D2D connection plays an important role in the calculation of offloading efficiency. Sun et al. [20] studied device-to-device enabled traffic offloading scheme by employing non-orthogonal multiple access (NOMA) and unlicensed access technologies to maximize the capacity of the D2D network by optimizing sub-channel assignment and power control while guaranteeing the capacity of NOMA-based cellular links and the WiFi system. Pan et al. [21] investigated the caching strategy to maximize the D2D offloading gain with the comprehensive consideration of user collaborative characteristics as well as the physical transmission conditions. Feng et al. [22] proposed a device-to-device communications-assisted traffic offloading scheme to improve the amount of traffic offloaded from cellular to WiFi in integrated cellular and WiFi networks. Peng et al. [23] discussed joint multi-user cooperative partial offloading, transmission scheduling and computation allocating for device-to-device underlay mobile edge computing. Ko et al. [12] proposed a distributed device-to-device offloading system (DDOS) in which a task owner opportunistically broadcasts an offloading request that includes its mobility level and task completion deadline. After receiving the request, mobile devices in the vicinity of the task owner

employ a constraint stochastic game to decide, in a distributed manner, whether to accept the request or not. However, these methods rarely consider that single relay may not be able to meet the response time requirements of compute-intensive tasks. We fully consider the available bandwidth and choose multiple relays to complete the transmission and ensure that the application is completed in the effective time.

7 Conclusion

In order to support the computation offloading of compute-intensive tasks in blind area, this paper proposes a computation offloading method, DRACO. Specifically, DRACO will select multiple mobile devices that can communicate with edge servers as relays, and target devices will establish D2D connection with these devices in turn. The compute-intensive tasks in blind area are offloaded to edge servers by parallel transmission of multiple relays. A large number of experiments with the real-world datasets have proved the feasibility and effectiveness of our method.

Acknowledgments. This work was supported in part by the National Key Research and Development Program of China (No. 2020YFB1707600), the Key Research and Development Program of Jiangsu Province (No. BE2019104), and the Collaborative Innovation Center of Novel Software Technology and Industrialization, Nanjing University.

References

1. Josilo, S., Dán, G.: Selfish decentralized computation offloading for mobile cloud computing in dense wireless networks. IEEE Trans. Mob. Comput. **18**(1), 207–220 (2019)
2. Josilo, S., Dán, G.: Wireless and computing resource allocation for selfish computation offloading in edge computing. In: 2019 IEEE Conference on Computer Communications, INFOCOM 2019, pp. 2467–2475. IEEE (2019)
3. Sun, H., Wang, J., Peng, H., Song, L., Qin, M.: Delay constraint energy efficient cooperative offloading in MEC for IoT. In: Collaborative Computing: Networking, Applications and Worksharing, pp. 671–685 (2021)
4. Liang, Y., Ge, J., Zhang, S., Wu, J., Tang, Z., Luo, B.: A utility-based optimization framework for edge service entity caching. IEEE Trans. Parallel Distributed Syst. **30**(11), 2384–2395 (2019)
5. Malik, R., Vu, M.: Energy-efficient computation offloading in delay-constrained massive MIMO enabled edge network using data partitioning. IEEE Trans. Wireless Commun. **19**(10), 6977–6991 (2020)
6. Peng, Q., He, Q., Xia, Y., Wu, C., Wang, S.: Collaborative workflow scheduling over MANET, a user position prediction-based approach. In: Gao, H., Wang, X., Yin, Y., Iqbal, M. (eds.) CollaborateCom 2018. LNICSSITE, vol. 268, pp. 33–52. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-12981-1_3
7. Zhan, W., Luo, C., Min, G., Wang, C., Zhu, Q., Duan, H.: Mobility-aware multi-user offloading optimization for mobile edge computing. IEEE Trans. Veh. Technol. **69**(3), 3341–3356 (2020)

8. Liu, L., Chang, Z., Guo, X., Mao, S., Ristaniemi, T.: Multiobjective optimization for computation offloading in fog computing. IEEE Internet Things J. **5**(1), 283–294 (2018)
9. Yang, Y., Long, C., Wu, J., Peng, S., Li, B.: D2D-enabled mobile-edge computation offloading for multi-user IoT network. IEEE Internet Things J. **8**, 12490–12504 (2021)
10. Zhang, X., Zhu, Q.: D2D offloading for statistical QoS provisionings over 5G multimedia mobile wireless networks. In: IEEE INFOCOM 2019 - IEEE Conference on Computer Communications, pp. 82–90 (2019)
11. Saleem, U., Liu, Y., Jangsher, S., Tao, X., Li, Y.: Latency minimization for D2D-enabled partial computation offloading in mobile edge computing. IEEE Trans. Veh. Technol. **69**(4), 4472–4486 (2020)
12. Ko, H., Pack, S.: Distributed device-to-device offloading system: design and performance optimization. IEEE Trans. Mob. Comput. **20**, 2949–2960 (2020)
13. Lagar-Cavilla, H.A., Tolia, N., de Lara, E., Satyanarayanan, M., O'Hallaron, D.: Interactive resource-intensive applications made easy. In: Cerqueira, R., Campbell, R.H. (eds.) Middleware 2007. LNCS, vol. 4834, pp. 143–163. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-76778-7_8
14. Lai, P., et al.: Optimal edge user allocation in edge computing with variable sized vector bin packing. In: Pahl, C., Vukovic, M., Yin, J., Yu, Q. (eds.) ICSOC 2018. LNCS, vol. 11236, pp. 230–245. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-03596-9_15
15. Wang, H., Xu, H., Huang, H., Chen, M., Chen, S.: Robust task offloading in dynamic edge computing. IEEE Trans. Mob. Comput., 1 (2021)
16. Deng, X., Sun, Z., Li, D., Luo, J., Wan, S.: User-centric computation offloading for edge computing. IEEE Internet Things J. **8**, 12559–12568 (2021)
17. Song, M., Lee, Y., Kim, K.: Reward-oriented task offloading under limited edge server power for multi-access edge computing. IEEE Internet Things J. **8**, 13425–13438 (2021)
18. Yan, J., Bi, S., Duan, L., Zhang, Y.J.A.: Pricing-driven service caching and task offloading in mobile edge computing. IEEE Trans. Wireless Commun. **20**, 4495–4512 (2021)
19. Yang, G., Hou, L., He, X., He, D., Chan, S., Guizani, M.: Offloading time optimization via Markov decision process in mobile-edge computing. IEEE Internet Things J. **8**(4), 2483–2493 (2021)
20. Sun, M., Xu, X., Tao, X., Zhang, P., Leung, V.C.M.: NOMA-based D2D-enabled traffic offloading for 5G and beyond networks employing licensed and unlicensed access. IEEE Trans. Wireless Commun. **19**, 4109–4124 (2020)
21. Pan, Y., Pan, C., Yang, Z., Chen, M., Wang, J.: A caching strategy towards maximal D2D assisted offloading gain. IEEE Trans. Mob. Comput. **19**(11), 2489–2504 (2020)
22. Feng, B., Zhang, C., Liu, J., Fang, Y.: D2D communications-assisted traffic offloading in integrated cellular-WiFi networks. IEEE Internet Things J. **6**(5), 8670–8680 (2019)
23. Peng, J., Qiu, H., Cai, J., Xu, W., Wang, J.: D2D-assisted multi-user cooperative partial offloading, transmission scheduling and computation allocating for MEC. IEEE Trans. Wireless Commun. **20**, 4858–4873 (2021)