Computer Communications 160 (2020) 759–768

Contents lists available at ScienceDirect

Computer Communications

journal homepage: www.elsevier.com/locate/comcom

A joint optimization scheme for task offloading and resource allocation based on edge computing in 5G communication networks

Shi Yang

School of Information Engineering, Changchun University of Finance and Economics, Changchun, Jilin, 130122, China

ARTICLE INFO

Keywords:
5G communication network
Edge computing
Task offloading
Resource allocation
Joint optimization of time delay and energy consumption
Device-to-Device (D2D)

ABSTRACT

With the development of 5G communication networks and the popularization of intelligent terminals, the computing resource intensive characteristic of various new applications poses a severe challenge to the task processing ability of intelligent terminals. In order to improve the efficiency of task processing, a joint optimization scheme for task offloading and resource allocation based on edge computing in 5G communication networks is proposed. Firstly, combining edge computing and Device-to-Device communication technologies, we propose three modes for processing computationally intensive tasks based on multi-user network system model for 5G edge networks, including local computing, fog node computing, edge node computing. Then, the corresponding time delay model, task execution model and offloading energy consumption computing model are constructed for these three computing modes. Finally, the problem of computing task offloading is transformed into a joint optimization problem of time delay and energy consumption, including optimization problems such as CPU frequency, offloading decision, transmission bandwidth allocation and power allocation of offloading users. Besides, the interior point method is utilized to solve this problem. Simulation platform is used to demonstrate the performance of our proposed scheme. The experimental results show that the scheme can effectively reduce the time delay and energy consumption of terminal tasks, which improves the efficiency of task processing and the experience quality of end users.

1. Introduction

While mobile communication promotes social development, it also brings severe challenges to the current mobile communication networks [1]. On the one hand, the explosive growth of mobile data and the repeated transmission of abundant popular contents have brought great pressure to mobile communication networks. This can cause network congestion and longer network transmission delays easily. On the other hand, many emerging network services and applications have increasingly demanded computing resources. This also brings higher requirements on the networks and terminal equipment [2,3]. Compared with 4G network, 5G will be a new network architecture, providing peak rate above 10 Gbps, better mobile performance, millisecond delay and ultra-high-density connection. However, it is difficult to break through the performance bottlenecks such as bandwidth bottleneck and delay jitter of bearer network. Many services will be terminated at the network edge after the introduction of edge computing. However, facing with that the number of users and computing tasks continues to increase, the limited computing capabilities of edge nodes still do not meet user needs well. How to combine 5G communication networks on the edge nodes with limited resources to ensure the service quality of end users has become a hot issue in current research [4,5].

The 5G mobile communication network uses network virtualization technologies to uniformly abstract network infrastructure resources and divides them into virtual network resources, which provides specific network services and functions by unified orchestration [6–8]. Deploying computing resources in 5G mobile communication networks is of great significance for satisfying the growing demand for computing processing capabilities of terminal equipment. With the rapid development of mobile Internet and Internet of Things, the applications in terminal equipment have become more and more complex, and the requirements for computing processing capabilities have become higher and higher correspondingly [9,10]. However, since the computing resources of terminal equipment are usually limited, it is often inadequate to handle computation-intensive tasks, which results in longer processing times. Therefore, in the case of limited computing resources, an effective solution is very important, which can not only ensure the quality of user experience, but also make full and reasonable use of resources.

The introduction of related work is in Section 2. The system model and problem modeling are introduced in Section 3. The proposed joint optimization scheme is introduced in Section 4. The experience and result analysis are written in Section 5. The conclusion and prospects are written in Section 6.

E-mail address: teacheryangshi@126.com.

https://doi.org/10.1016/j.comcom.2020.07.008
Received 18 March 2020; Received in revised form 3 July 2020; Accepted 6 July 2020
Available online 7 July 2020

2. Related works

Mobile Edge Computing (MEC) technology can provide users with differentiated and customized network services. The basic idea is to migrate cloud computing platforms (including computing, storage and network resources) to the edge of mobile networks, and to provide high-quality network services at the edge of mobile networks near end users [11]. MEC reduces the end-to-end latency of content distribution and service delivery by strengthening deep integration with traditional mobile communication networks, Internet and Internet of Things. Therefore, user experience and system performance of mobile communication networks are improved [12–14]. Especially when mobile terminals are overloaded with computing, if a large amount of tasks are offloaded to edge nodes and remote cloud nodes, it can effectively relieve the load pressure on edge nodes. This can also save the resource consumption and task processing time of user equipment [15]. Efficient computation of tasks is achieved in this collaborative computing mode [16,17]. In order to obtain the optimal offloading strategy, reference [18] considers different requirements of the task in terms of communication and computing in the single-user scenario. It defines a new variable called computational energy efficiency, which is used for communication computing scheduling and to solve the problem of computational offloading with time delay constraints. Reference [19] proposed a special type of data partition-oriented application. They designed such applications to be partially offloaded, considering the optimization of transmission power, thereby minimizing the energy consumption of mobile users. However, there is a big difference between the single-user scenario and the actual multi-user scenario, which makes these offloading strategies problematic in practical applications [20,21].

Aiming at the multi-user problem in real scenarios, how to make optimal offloading and resource allocation decisions considering the resource allocation of edge nodes is a difficult problem to be solved. In reference [22], a game theory approach was used to obtain an optimal offloading decision in a multi-user cloud computing environment. It studied the optimization of resource allocation in a multi-radio channel environment [23]. However, the effect of edge node resource allocation on computing offload is ignored. In reference [24], joint optimization of resource allocation and offloading decision-making was performed to satisfy user delay constraints while saving energy consumption. However, in order to simplify the calculation, the energy consumption of mobile equipment is set to a constant value, which ignores the impact of time changes on user energy consumption [25].

It can be seen that in the multi-user and multi-task scenario where the edge cloud and remote cloud are combined, there are still many key issues to be resolved in terms of service quality assurance for end users. Among them, how to achieve efficient computational offloading and optimal resource allocation is one of the key research issues in this field. Therefore, based on the above analysis, a joint optimization scheme for task offloading and resource allocation of 5G communication networks based on edge computing is proposed comprehensively considering the task characteristics and system availability status. This scheme can effectively reduce terminal task execution time delay and energy consumption, which significantly improves service quality of users. The main innovations of our proposed scheme are summarized as follows:

(1) Aiming at the problem that it is difficult to perform intensive tasks in 5G communication networks, a distributed computing task processing model is established to reduce network delay. The model contains three computing modes, namely local computing mode, fog node computing mode and edge node computing mode. And user equipment may select the optimal mode for processing computing tasks according to the characteristics of computing tasks and system states.

(2) The performance of different computing modes is different, and the pros and cons are not clear. Thus, corresponding time delay models, task execution models and offload energy consumption computing models are constructed for the three computing modes. We also uses D2D communication technologies to offload computing tasks between user equipment, which further reduces the burden on fronthaul links and transmission delays.

(3) The proposed scheme considers the impact of different time delay requirements and resource allocation on the performance of dense networks. It transforms the problem of computing task offloading into a joint optimization problem of time delay and energy consumption. Furthermore, corresponding optimization algorithms are designed to reduce time delay and system energy consumption.

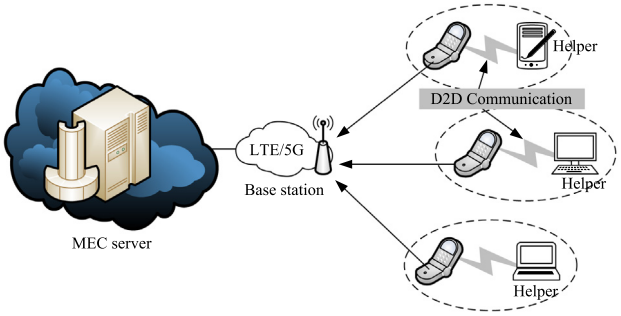

Fig. 1. 5G edge network diagram.

3. System model and problem modeling

A multi-user network system model for 5G edge networks is shown in Fig. 1. In this network model, it is assumed that there is a base station and mobile terminals, and the base station is connected to MEC server in a side-by-side manner. Therefore, mobile terminals can access MEC server by the 5G wireless network. In addition, it is assumed that each mobile terminal has a fixed fog computing node, named Helper node. The Helper node usually has certain computing resources, such as mobile computing equipment and personal computers. Thus, it can be used to assist mobile terminals in computing tasks [26]. In particular, the communication interaction between mobile terminals and Helper nodes uses D2D communication technologies.

Due to the limitations of mobile terminals in computing power and battery energy, offloading computing tasks is an important way to improve the energy efficiency and computing performance of mobile terminals. However, considering the limited computing resources in the Helper (that is, the fog computing equipment), it is difficult to satisfy the performance requirements only by offloading computing tasks to the Helper. Consequently, for computation-intensive tasks, consider three processing methods, as shown in Fig. 2. The first method is to execute in mobile terminals with limited computing resources, that is locally. The second is to offload computing tasks to the Helper for execution by D2D communication technologies. Helpers usually have idle but limited computing resources. Finally, the third is to offload computing tasks to MEC server for execution over the cellular network. The MEC server usually has enough computing resources to complete the tasks.

In addition, the framework and process of computing task offloading need to be further explained. As shown in Fig. 2, the computing task offloading framework can be divided into two parts: the control plane and the data plane. The control plane mainly includes a computational task unload controller, which can make computing task offloading decisions. Besides, the computational task unload controller can also sense the status of the queue in Helper, the status of computing task arrival and the status of idle computing resources in real time [27]. The data plane is mainly used for data buffering, transmission and offloading. It mainly includes the task queue buffer and the task data transmission part in mobile terminals. Therefore, the computational task unload controller can make offloading decisions based on the network status [28].

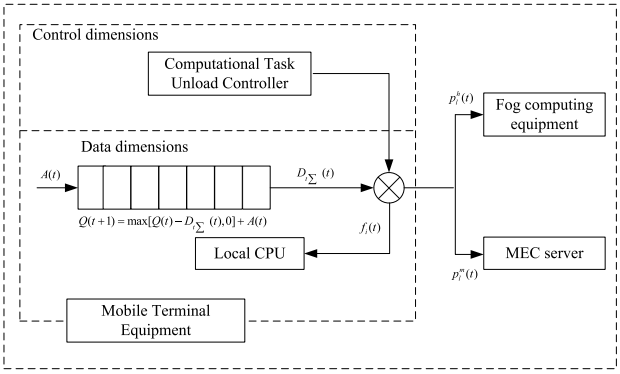

Fig. 2. Computing task offloading framework.

3.1. Computing task and task queue model

Assume that time is slotted and the length of each slot is T, the set of slots is denoted as $\Gamma = \{1, 2, \ldots, T\}$. In each time slot t, the computing task arrives at the mobile terminal i can be represented by a random variable $A_i(t)$. At the same time, it can be further assumed that each $A_i(t)$ is independent and identically distributed, that is $IE\left[A_i(t)\right] = \lambda_i, i \in N$, where $A_i(t) \in \left[A_{i,\min}, A_{i,\max}\right]$. It needs to be further emphasized that the proposed method focuses on computationally intensive tasks, which require a large amount of computing resources. On the one hand, processing these computationally intensive tasks in a mobile terminal results in long computation time. On the other hand, it also causes an increase in energy consumption, since increasing the frequency of local CPU significantly increases energy consumption [29].

Mobile terminals have limited battery capacity and computing power. Therefore, offloading these computing tasks is of great significance for reducing the energy consumption of terminal equipment. Through the computing task offloading, in each time slot t, the computing tasks of mobile terminals can be executed on the local CPU, Helper equipment and MEC server. Thus, $D_{i,l}(t)$ (bits) is used to represent the amount of computing tasks performed on the local CPU. Use $D_{i,h}(t)$ (bits) to represent the amount of computing tasks performed in Helper equipment. $D_{i,m}(t)$ (bits) represents the amount of computing tasks performed in MEC server.

Although the Helper equipment has idle computing resources, the computing resources in Helper equipment are usually limited. And the amount of idle computing resources in each Helper equipment is usually different. In addition, the idle computing resources in Helper equipment can be represented by the number of CPU cycles that can be allocated. In order to reduce the model complexity, a simplified model is used to represent the idle computing resources in Helper equipment. That is, the amount of computing tasks that the Helper equipment can handle in time slot t is used to represent idle computing resources. Generally, the more CPU cycles that can be allocated in Helper equipment, the larger amount of computing tasks that can be used for processing. Thus, it is reasonable to represent the idle computing resources of Helper equipment with the amount of computing tasks that can be processed [30]. Let $O_{i,h}(t)$ denote idle computing resources in Helper $O_{i,h}(t)$ and follow the constraint $O_{i,h}(t) \in \left[O_{i,h}^{\min}, O_{i,h}^{\max}\right]$. On the other hand, it is assumed that MEC server has sufficient computing resources to handle these tasks. It is assumed that mobile terminals can sense the workload state of Helper equipment. Usually state awareness requires some communication overhead, but since the communication overhead is very small, it can be ignored.

In addition, tasks that arrive but have not yet been executed are placed in a buffer queue. The length of this queue can be expressed as $Q(t) \cong \left[Q_1(t), Q_2(t), \ldots, Q_N(t)\right]$, where $Q(0) = 0$. For the queue $Q_i(t)$ in mobile terminal i, the following formula is followed:

$$Q_i(t+1) = \max\left\{Q_i(t) - D_{i,\sum(t)}, 0\right\} + A_i(t) \tag{1}$$

Here, $D_{i,\sum(t)} = D_{i,l}(t) + D_{i,h}(t) + D_{i,m}(t)$ is the amount of computing tasks leaving the queue. Besides, the queue length usually reflects the time delay and longer queues usually result in higher delays [31].

3.2. Time delay models for different modes

Time delay models are built for different modes, such as local computing, fog node computing and edge node computing. The specific expressions are as follows:

(1) Time delay model in local computing mode: In this mode, the processing time delay of the computing task $T_{m,n}$ only includes the processing time delay of user equipment. It is expressed as follows:

$$D_{m,n}^{L} = \frac{\mu_{L,m} B_{m,n}}{f_{L,m}} \tag{2}$$

In the formula, $\mu_{L,m}$ is the number of cycles that the CPU of user equipment U_m needs to process a 1-bit computing task. $B_{m,n}$ is the size of computing task $T_{m,n}$, and $f_{L,m}$ is the CPU operating frequency of user equipment U_m.

(2) Time delay model of fog node computing mode: The time delay of this mode consists of the transmission delay of access link and the processing time delay of fog node F_T. It is expressed as follows:

$$D_{m,n}^{F} = D_{m,n}^{F,1} + D_{m,n}^{F,2} = \frac{B_{m,n}}{\lambda_m \log_2\left(1 + \frac{|h_m|^2 d_m^{-\alpha} P_m}{\lambda_m \delta_0}\right)} + \frac{\mu_F B_{m,n}}{f_F} \tag{3}$$

In the formula, λ_m is the transmission bandwidth allocated to user equipment U_m, and h_m is the Rayleigh fading factor of access link. d_m is the distance between user equipment U_m and fog node F_T, and α is the road loss index. P_m is the transmission power of user equipment U_m, and δ_0 is the noise power spectral density of access channel. μ_F is the number of cycles that the CPU needs to run for fog node F_T to process a 1-bit computing task. f_F is the CPU operating frequency of fog node F_T.

(3) Time delay model of edge node computing mode: The time delay of this mode is determined by the access link transmission delay, the return link transmission delay and the processing time delay of edge computing center. The specific calculation is as follows:

$$\begin{aligned} D_{m,n}^{C} &= D_{m,n}^{C,1} + D_{m,n}^{C,2} + D_{m,n}^{C,3} \\ &= \frac{B_{m,n}}{\lambda_m \log_2\left(1 + \frac{|h_m|^2 d_m^{-\alpha} P_m}{\lambda_m \delta_0}\right)} + \frac{B_{m,n}}{\lambda_T \log_2\left(1 + \frac{|h_T|^2 d_T^{-\alpha} P_T}{\lambda_T \delta_0}\right)} + \frac{\mu_C B_{m,n}}{f_C} \end{aligned} \tag{4}$$

In the formula, λ_T, h_T are the transmission width and Rayleigh fading factor of the return link respectively. P_T is the transmitting power of F_T, μ_C is the edge computing center, and S_C is the number of cycles that the CPU needs to run for a 1-bit computing task. f_C is the CPU operating frequency of edge computing center.

3.3. Computing task execution model and computing offloading energy consumption model

3.3.1. Different computing task execution models

Computing task execution models include local execution model, fog computing equipment execution model and MEC execution models. The details are as follows:

(1) Local execution model. It is assumed that the amount of 1-bit computing tasks on mobile terminal i requires L_i CPU cycles. This value usually depends on the type of application and can be obtained from offline measurements [32]. Let $f_i(t)$ be the CPU cycle frequency

of mobile terminal i in the tth time slot. In addition, set a CPU cycle frequency constraint, that is the maximum CPU cycle frequency, denoted as $f_i(t) \le f_{i,\max}$. Therefore, the amount of computing tasks performed locally by mobile terminal i in time slot t can be expressed as:

$$D_{i,l}(t) = \tau f_i(t) L_i^{-1} \tag{5}$$

The local energy consumption of the ith mobile terminal in a time slot can be expressed as:

$$P_{i,l}(t) = \tau \alpha f_i^3(t) \tag{6}$$

where α is a parameter determined by CPU model.

(2) Fog computing equipment execution model, that is the task execution model when computing tasks are unloaded into the fog computing equipment Helper. In this model, computing tasks can be offloaded to Helper for processing and execution by D2D communication technologies. For simplicity, it is assumed that Helper has idle but limited computing resources to handle the computing tasks of mobile terminals. And it is assumed that the processing time delay of Helper is negligible [33]. Thus, the amount of tasks that mobile terminal i offloads to Helper equipment in time slot t is:

$$D_{i,h}(t) = r_i^h(t)\tau = \rho_i^h(t) W \tau \log_2\left(1 + \frac{p_i^h(t) H_i^h(t)}{\rho_i^h(t) W N_0}\right) \tag{7}$$

where due to the limited computing resources of Helper, $D_{i,h}(t)$ follows the constraint $D_{i,h}(t) \le O_{i,h}(t)$. The energy consumption transmitted by mobile terminal i to Helper in a time slot can be expressed as:

$$P_{i,h}(t) = p_i^h(t)\tau \tag{8}$$

(3) MEC execution model, that is the execution model of computing tasks offloaded to MEC server. In this model, computing tasks can be offloaded to a MEC server via a cellular network for task execution. For simplicity, it is assumed that MEC server has sufficient computing resources to process N different applications in parallel. And the processing time delay of MEC server is negligible. At time t, the amount of tasks that mobile terminal i offloads to MEC server can be expressed as:

$$D_{i,m}(t) = r_i^m(t)\tau = \rho_i^m(t) W \tau \log_2\left(1 + \frac{p_i^m(t) H_i^m(t)}{\rho_i^m(t) W N_0}\right) \tag{9}$$

In a time slot, the transmission energy consumption of mobile equipment i offloading the computing task to MEC server can be expressed as:

$$P_{i,m}(t) = p_i^m(t)\tau \tag{10}$$

3.3.2. Computing offloading energy consumption model

As described above, the total energy consumption of mobile terminals mainly includes local execution energy consumption and transmission energy consumption. Besides, the transmission energy consumption mainly includes the transmission energy consumption offloaded to Helper and the transmission energy consumption offloaded to MEC server. Therefore, the energy consumption of mobile terminal i in a time slot can be defined as:

$$P_i(t) \cong P_{i,l}(t) + P_{i,h}(t) + P_{i,m}(t) \tag{11}$$

Then the total energy consumption of all mobile terminal equipment in a time slot can be expressed as:

$$P(t) = \sum_{i=1}^{N} P_i(t) \tag{12}$$

Furthermore, it is assumed that all mobile terminals share the common bandwidth of WHz in an orthogonal manner. And each mobile terminal i can be allocated a certain bandwidth resource, denoted as $\rho_i^{(x)} W$. It is further assumed that the channel gain in the allocated bandwidth of each mobile terminal is constant (i.e., flat fading in allocated spectrum) [34]. The transmission rate from mobile terminal i to the base station and its Helper can be expressed as:

$$r_i^x(t) = \begin{cases} \rho_i^x(t) W \log_2\left(1 + \dfrac{p_i^x(t) H_i^x(t)}{\rho_i^x(t) W N_0}\right), & \rho_i^x(t) > 0 \\ 0, & \text{otherwise} \end{cases}$$
$$H_i^x(t) = h_i(t)\, go \left(d_0/d_i^x\right)^{\theta} \tag{13}$$

In the formula, $x \in \{h, m\}$, $\rho_i^x(t) \in [0,1]$ and h are Helper, m is MEC server, and $p_i(t)$ is the transmission power of mobile terminals. W is the system bandwidth and N_0 is the power spectral density of additive white Gaussian noise. go is the path loss constant, d_0 is the reference distance, and θ is the path loss index. d_i is the distance from mobile equipment i to the Helper/MEC server.

3.4. Problem modeling

It can be known from the above that the energy consumption of mobile terminals mainly includes computing energy consumption and task transmission energy consumption. Therefore, based on the above system model, the average time energy consumption of mobile terminals can be defined as:

$$\overline{P}_i = \lim_{T\to\infty} \frac{1}{T} IE\left[\sum_{t=0}^{T-1} P_i(t)\right], i \in N \tag{14}$$

Then the average time energy consumption of N mobile terminals can be expressed as:

$$\overline{P} = \lim_{T\to\infty} \frac{1}{T} IE\left[\sum_{t=0}^{T-1}\sum_{i=1}^{N} P_i(t)\right], i \in N \tag{15}$$

According to Little's Law, the execution delay is proportional to the average queue length of task buffer. Thus, if mobile terminals greedily time delay the offloading of computing tasks to save energy consumption, the average queue length of task buffer will continuously increase. This can cause large network delays and poor end-user quality of experience. Therefore, when deciding to offload computing tasks, it is necessary to balance energy consumption and time delay [35]. The average queue length of task buffer is used as a measure of execution latency, that is:

$$\overline{Q}_i = \lim_{T\to\infty} \frac{1}{T} IE\left[\sum_{t=0}^{T-1} Q_i(t)\right], i \in N \tag{16}$$

where $\overline{Q}_i$ is the average queue length of task buffer. Since all queues are required to have a stable average rate, in each time slot t, mobile terminals can make an online computing offloading decision. The goal is to minimize time-averaged energy consumption under the constraints of queue stability [36]. Therefore, the problem of minimizing energy consumption can be expressed as:

$$\begin{aligned} &P_1: \min_{f(t),P_h(t),P_m(t)} \overline{P} \\ &s.t. \\ &C1: 0 \le f_i(t) \le f_{i,\max}, i \in N, t \in T \\ &C2: 0 \le p_i^h(t) \le p_{i,\max}^h, i \in N, t \in T \\ &C3: 0 \le p_i^m(t) \le p_i^m(t), i \in N, t \in T \\ &C4: \sum_{i=1}^{N} \left(\rho_i^h(t) + \rho_i^m(t)\right) \le 1, i \in N \\ &C5: D_{i,h}(t) \le O_{i,h}(t), i \in N \\ &C6: Q_i(t) \text{ is a queue with a stable average rate.} \end{aligned} \tag{17}$$

where $f(t) = \left[f_1(t), f_2(t), \ldots, f_N(t)\right]$ and $C1$ are the cycle frequency limits of CPU. $C2, C3$ are the transmission power limits for offloading tasks to Helper and MEC server respectively. $C4$ is the bandwidth limit, $C5$ is the computing resource limit of fog computing equipment, and $C6$ is the stability limit of the network system.

3.5. Problem description for joint optimization of time delay and energy consumption

According to the analysis of time delay performance, it can be known that both the computing resources and the wireless communication resources have an impact on time delay performance. Specifically, the more computing resources (CPU operating frequency) allocated to each computing node (user equipment, fog computing node and edge computing center), the shorter processing task processing time delay. The more communication resources (transmission bandwidth and transmission power) allocated to each computing node, the shorter transmission delay of computing tasks [37,38]. If only the optimization of time delay index is considered, it is necessary to allocate as many computing resources and communication resources as possible to each computing node. However, the more computing and communication resources allocated to each computing node, the more energy the system consumes. In order to achieve a balance between time delay performance and energy consumption, this paper studies the joint optimization of time delay and energy consumption.

The latency and energy consumption performance of this system are jointly determined by the offloading strategies of multiple computing tasks, and the computing/communication and cache capabilities of network nodes.

(1) Offloading decision constraints: In the introduction of system model, we can know that any computing task is either processed by user equipment or transmitted to F_T/S_C for processing. Thus, the uninstallation decision needs to satisfy the following constraints:

$$x_{m,n}^L + x_{m,n}^F + x_{m,n}^C = 1, x_{m,n}^L, x_{m,n}^F, x_{m,n}^C \in \{0,1\} \tag{18}$$

where $x_{m,n}^L$ is a local computing mode indicator, and $x_{m,n}^L = 1$ indicates that the computing task is handled by user equipment. Similar to $x_{m,n}^L$, $x_{m,n}^F$ and $x_{m,n}^C$ are indicators for fog node computing mode and cloud node computing mode respectively. The above formula indicates that each computing task can only be processed by one mode.

(2) Constraints on communication resources: In communication resources, the main focus is on transmission power and spectrum resources. The fog computing node F_T and all user equipment are subject to independent transmission power limits, that is

$$P_T^{\min} \le P_T \le P_T^{\max}, P_m^{\min} \le P_m \le P_m^{\max},\ m = 1, \ldots, M \tag{19}$$

where $P_T^{\min}$ and $P_T^{\max}$ are the minimum and maximum values of P_T, $P_m^{\min}$ and $P_m^{\max}$ are the minimum and maximum values of P_m respectively.

(3) Computing and cache resource constraints: Using dynamic voltage frequency adjustment technology, the CPU operating frequency can be changed by adjusting the chip voltage. Therefore, the CPU operating frequencies of user equipment U_m, fog computing node F_T and edge computing center S_C need to satisfy the following constraints:

$$f_{L,m}^{\min} \le f_{L,m} \le f_{L,m}^{\max}, f_F^{\min} \le f_F \le f_F^{\max}, f_C^{\min} \le f_C \le f_C^{\max} \tag{20}$$

where $f_{L,m}^{\min}$ and $f_{L,m}^{\max}$ are the minimum and maximum values of $f_{L,m}$, $f_F^{\min}$ and $f_F^{\max}$ are the minimum and maximum values of f_F, $f_C^{\min}$ and $f_C^{\max}$ are the minimum and maximum values of f_C respectively.

To minimize the total processing cost of computing tasks within the system. The optimization problems established are as follows:

$$\min_{x_{m,n}^L, x_{m,n}^F, x_{m,n}^C, P_T, P_m, \lambda_m, f_{L,m}, f_F, f_C} Q = \sum_{n=1}^{N} \sum_{m=1}^{M} C_{m,n} \tag{21}$$

where Q is the total processing cost of computing tasks in the system.

4. Proposed joint optimization scheme of time delay and energy consumption

The above optimization problem is decoupled into four independent sub-problems: CPU frequency optimization problem, offloading decision optimization problem, transmission bandwidth allocation optimization problem, and power allocation optimization problem of offloading users. For each sub-problem, we utilize a suitable algorithm to solve it and iteratively solves each sub-problem in turn until convergence.

4.1. CPU frequency optimization

The optimization of CPU frequency has nothing to do with the allocation strategy of communication resources. To minimize the total processing cost of computing tasks in this system, it is equivalent to solving the following problems:

$$\begin{aligned} &\min_{f_i} Q_{1,i}\left(f_i\right) = \frac{\theta_D \mu_i}{f_i} + \theta_E \eta_i f_i^2 \\ &s.t.\ f_i^{\min} \le f_i \le f_i^{\max},\ i \in \left\{L_1, L_2, \ldots, L_M, F, C\right\} \end{aligned} \tag{22}$$

By taking the second derivative of $Q_{1,i}\left(f_i\right)$, that is:

$$\nabla_{f_i}^2 Q_{1,i}\left(f_i\right) = \frac{2\theta_D \mu_L}{f_i^2} + 2\theta_E \eta_i \ge 0 \tag{23}$$

Therefore, the CPU frequency optimization problem is a convex optimization problem. By solving equation $\nabla_{f_i} Q_{1,i}\left(f_i\right) = 0$, the optimal solution can be obtained:

$$f_i^{opt} = \sqrt[3]{\frac{\theta_D \mu_i}{2\theta_E \eta_i}} \tag{24}$$

The closed-form solution for CPU frequency optimization is:

$$f_i^* = \begin{cases} f_i^{\min}, f_i^{\min} > f_i^{opt} \\ f_i^{opt}, f_i^{\min} \le f_i^{opt} \le f_i^{\max} \\ f_i^{\max}, f_i^{\max} < f_i^{opt} \end{cases} \tag{25}$$

4.2. Bandwidth allocation optimization

The transmission bandwidth allocation strategy only affects the fog node computing mode and edge node computing mode. When offloading decision, the power control of each network node and CPU operating frequency are determined, in order to minimize the total system cost, it is equivalent to solving the following optimization problem:

$$\begin{aligned} &\min_{\lambda_1, \ldots, \lambda_M} Q_2\left(\lambda_1, \ldots, \lambda_M\right) = \sum_{m=1}^{M} \frac{D_m\left(\theta_D + \theta_E P_m\right)}{\lambda_m \log_2\left(1 + \frac{|h_m|^2 d_m^{-\alpha} P_m}{\lambda_m \delta_0}\right)} \\ &s.t. \sum_{m=1}^{M} \lambda_m \le \lambda, \lambda_m \ge 0 \end{aligned} \tag{26}$$

where $D_m = \sum_{n=1}^{N} B_{m,n}\left(x_{m,n}^F + x_{m,n}^C\right)$.

The objective function $Q_2\left(\lambda_1, \ldots, \lambda_M\right)$ given in the above formula is convex with respect to $\lambda_1, \ldots, \lambda_M$. Using some existing convex optimization algorithms, such as the interior point algorithm, the global optimal solution to this problem can be obtained.

4.3. Offloading decision optimization

Let $A = \left[a_1, a_2, \ldots, a_N\right]$ be the offloading decisions for all users. In the initial state, it is assumed that the offloading decision is an all-one matrix, that is, all users choose to offload the computing of tasks, which is denoted as A^0. Coordinate descent requires multiple iterations to find the optimal value. At the $l-1\,(l = 1, 2, \ldots)$ iteration, let A^{l-1} represent the offloading decision at this time. Then $V\left(A^{l-1}\right)$ represents the optimal value of the formula when offloading decision is A^{l-1}. The coordinate descent method changes the offloading decision of only one user at a time during one iteration. Let Q_n^l denote the gain of the system at the lth iteration, which can be expressed as:

$$Q_n^l = V\left(A^{l-1}\right) - V\left(A^{l-1}(n)\right) \tag{27}$$

where $A^{l-1}(n)$ is the offloading decision after user n changes state. The specific update rules are as follows:

$$A^{l-1}(n) = \left[a_1^{l-1}, a_2^{l-1}, \ldots, a_n^{l-1} \oplus 1, \ldots, a_N^{l-1}\right] \tag{28}$$

The coordinate descent method will continuously optimize in the direction of a user variable a_n to find the local optimal value of objective function. Through multiple iterations, the algorithm can reach convergence. In the lth iteration, the offloading decision is A^l. If computing return $Q^l_{n_l^*} > 0$, the current offloading decision is updated to $A^l = A^{l-1}\left(n_l^*\right)$, where $n_l^* = \arg\max_{n=1,\ldots,N} Q_n^l$ means the user who gains the most from changing offloading decision.

It can be seen from the above formula that the offloading decision is closely related to sub channel allocation and the users' transmit power. Thus, reasonable sub channel allocation and power allocation during each iteration will help to get a better offloading decision.

4.4. Power allocation optimization for offloaded users

Under the specific offloading decision and resource allocation results, the initial optimization problem is transformed into an optimal power solution problem with time delay and power constraints:

$$\begin{aligned} P1: \quad & \min Z(P) = \sum_{n=1}^{N_C} \frac{d_n \sum_{k\in K} c_n^k p_n^k}{\sum_{k\in K} c_n^k B \log_2\left(1 + SINR_n^k\right)} + P_n^i \cdot \frac{w_n}{f^c} \\ s.t. \quad & T_n^t + T_n^c \le T_n^{\max}, \forall n \in N_c \\ & \sum_{k\in K} p_n^k \le P_{\max}, \forall n \in N_c \\ & p_n^k \ge 0, \forall n \in N_c \end{aligned} \tag{29}$$

It can be seen from the above formula that this is a problem of minimizing energy consumption based on power optimization. Since it represents the time delay constraint of users, problem $P1$ can be transformed into a power minimization problem under the time delay constraint. And the optimization problem is not a convex function problem. The problem is transformed by the method of variable substitution, and then transformed into a convex optimization problem. If $p_n^k = e^{S_n^r}$, it will be transformed into the following form:

$$\begin{aligned} P2: \quad & \min_{S_n^k, r_n^k, R_n} \sum_{n=1}^{N_c} \sum_{k\in K} e^{S_n^k} \\ s.t. \quad & R_n \ge \frac{d_n}{T_n^{\max} f^c - w_n}, \forall n \in N_c \\ & \sum_{k\in K} e^{S_n^k} \le P_{\max}, \forall n \in N_c \\ & r_n^k \le B \log_2\left(1 + \frac{h_{n,n}^k e^{S_n^k}}{\omega_0 + \sum_{m=1,m\ne n}^{N_c} h_{m,n}^k e^{S_m^k}}\right), \forall n \in N_c \\ & R_n = \sum_{k\in K} r_n^k, \forall n \in N_c \end{aligned} \tag{30}$$

In the above problem, the equation relationship of equation is changed into an inequality constraint. The purpose of this change is to convert problem $P2$ into a convex problem. And this change will not affect the optimality of solving problem. This is because the data rate of user n on sub channel k cannot be less than $B\log_2\left(1 + h_{n,n}^k e^{S_n^k} / \omega_o + \sum_{m=1,m\ne n}^{N_c} h_{m,n}^k e^{S_m^k}\right)$ in the optimality.

Under high signal-to-interference and noise ratio conditions, $P2$ is a convex optimization problem. The optimal power allocation results can be solved using the interior point method.

4.5. Joint optimization algorithm for time delay and energy consumption

Based on the above analysis, an iterative optimization algorithm is designed to solve the joint optimization problem of time delay and energy consumption, as shown in Algorithm 1.

Algorithm 1 Joint optimization algorithm of time delay and energy consumption

1. Initialization: $x_{m,n}^L(0)$, $x_{m,n}^F(0)$, $x_{m,n}^C(0)$, $P_T(0)$, $P_m(0), \lambda_m(0)$, $f_{L,m}(0)$, $f_F(0)$, $f_C(0)$, $Q(0)$, maximum iterations L and convergence threshold ε;

2. Repeat: in the l iteration, $1 \le l \le L$;

3. Update $f_{L,m}(l)$, $f_F(l)$, $f_C(l)$, $m = 1, 2, \cdots, M$ according to equation (24);

4. Use the interior point method to solve optimization problems, update $\lambda_m(l)$, $m = 1, 2, \cdots, M$;

5. Initialization $p_n^{(0)}, p_n^{\min} \le p_n^{(0)} \le p_n^{\max}, R_n^{(0)} = \sum_{n\in N_c} r_n^{(0)}$, maximum iterations K and convergence threshold ζ;

5.1 Repeat: In the k iteration, $1 \le k \le K$;

5.2 Update p_n^k according to equation (28);

5.3 Update R_n according to equation (29);

5.4 Termination: $k \ge K$ or $\left|R_n - \frac{d_n f^c}{T_n^{\max} f^c - w_n}\right| \le \zeta$;

6. Solve the relaxation problem shown in equation (26) according to (27);

6.1. **If** $Q_n^l = V\left(A^{l-1}\right) - V\left(A^{l-1}(n)\right) > 0$

$A^l = A^{l-1}\left(n_l^*\right)$

$A^{l-1}(n) = \left[a_1^{l-1}, a_2^{l-1}, \cdots, a_n^{l-1} \oplus 1, \cdots, a_N^{l-1}\right]$

6.2 ELSE return 6.1;

6.3 Optimal offloading decision $A = [a_1, a_2, \cdots, a_N]$;

7. Update $Q(l)$ according to formula (20);

8. **Termination:** $l > L$ or $\left|Q(l) - Q(l-1)\right| \le \varepsilon$;

In Algorithm 1, the optimization results of CPU frequency, transmission bandwidth allocation, transmission power control and offloading decision of each computing task are sequentially updated until the maximum number of iterations is reached or the results converge.

Since the solutions of all sub problems are optimal, the total cost Q decreases as the number of iterations increases. In addition, Q has a lower bound of 0, which guarantees that Algorithm 1 can converge to a stable solution. Otherwise, Q will keep falling, which will contradict its existence of lower bound 0. Consequently, the joint time delay and energy optimization algorithm can converge to a stable optimal solution by a limited number of iterations.

5. Experiment and result analysis

In order to evaluate the performance of our proposed algorithm, this paper analyzes the convergence, total energy consumption and total

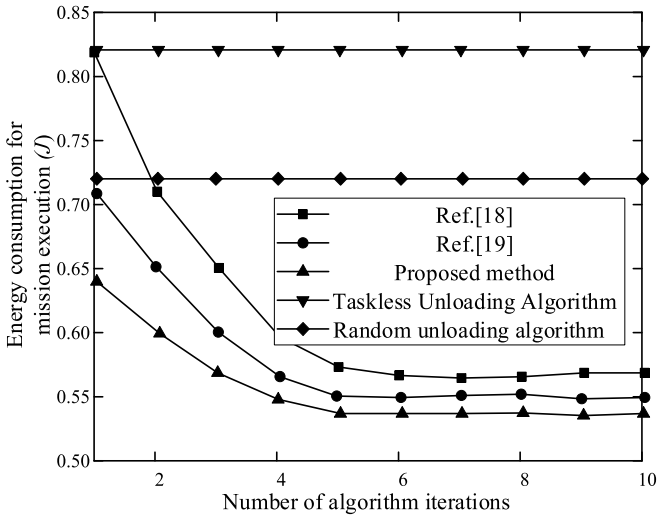

Fig. 3. The relationship between task execution energy consumption and algorithm iteration times.

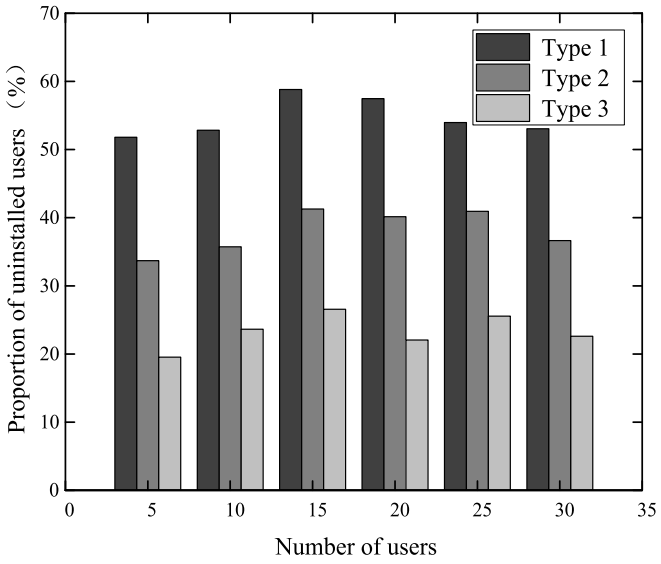

Fig. 4. Proportion of users offloaded under different time delay constraints.

time delay performance of the algorithm. We consider that there are N mobile terminals around the MEC server, and each mobile terminal has fog computing equipment. The distance between mobile terminals and MEC server is set to 200 ~ 400 m; the distance between mobile terminals and fog computing equipment is set to 10 ~ 30 m. The parameter settings in the simulation are as follows:

The computing task size follows a normal distribution with a mean value of 200 M bits and a variance of 200 M bits. The buffer capacity of fog computing node F_T is $B_F = 4G$ bits. The local user equipment, fog computing node and edge computing center S_C need to perform a 1-bit computing task, and the number of CPU revolutions required by them is $\mu_L = \mu_F = \mu_C = 737.5$ revolutions/bit. The effective conversion capacity of U_m, F_T, S_C is set to $\mu_{L,m} = 2 \times 10^{-26}, \eta_F = 7 \times 10^{-28}, \eta_C = 5 \times 10^{-29}$ respectively. The maximum and minimum values of CPU frequency of U_m are $f_{L,m}^{\min} = 1$ GHz and $f_{L,m}^{\max} = 5$ GHz respectively. Similarly, the maximum and minimum values of CPU frequencies of F_T and S_C are $f_F^{\min} = 5$ GHz, $f_F^{\max} = 10$ GHz, $f_C^{\min} = 15$ GHz and $f_C^{\max} = 25$ GHz respectively. In addition, the total bandwidth of access link $\lambda = 1200$ MHz and the return link bandwidth $\lambda_T = 1500$ MHz. The Rayleigh fading parameters of the access link and the return link are $\sigma_m = \sigma_T = 1$, the path loss factor is $\alpha = 4$. The distance between U_m and F_T is $d_m = 200$ m, and the distance between F_T and S_C is $d_T = 2000$ m. Noise power spectral density $\delta_0 = -174$ dBm/Hz. The maximum and minimum values of U_m transmission power are $P_m^{\min} = 20$ dBm and $P_m^{\max} = 30$ dBm respectively; the maximum and minimum values of F_T transmission power are $P_T^{\min} = 30$ dBm and $P_T^{\max} = 40$ dBm respectively.

5.1. Iterative analysis

The convergence times of the proposed scheme are critical to the energy consumption of system, so simulation experiments are performed on it. The results are shown in Fig. 3, which includes two benchmark algorithms (taskless unloading algorithm and random unloading algorithm). Comparison algorithm uses the methods in reference [18] and reference [19].

It can be seen from Fig. 3 that as the number of algorithm iterations increases, the task execution energy consumption tends to converge in fewer times. Comparing the task execution energy consumption of different methods, it can be seen that the task execution energy consumption of our proposed method is relatively small. Since the proposed method can adapt to different channel noises, it will not reduce users' transmission rate, which will lead to reduced task execution time delay and energy consumption. The algorithms in reference [18] and reference [19] are affected by a certain signal noise. Therefore, the transmission rate is affected and the energy consumption is large. The taskless unloading algorithm and random unloading algorithm are both non-iterative algorithms, and the task execution energy consumption does not change with the number of iterations.

5.2. Total number of offloaded users and total system energy consumption with different time delay constraints

Considering the impact of simulation parameter value range on the performance of our proposed algorithm, three different types of time delay constraint value ranges are specified: type 1 = 1~3 s, type 2 = 1~4 s, type 3 = 1~5 s. The system performance of proposed algorithm is analyzed under different time delay values. The proportion of offloaded users under different time delay constraints is shown in Fig. 4.

It can be seen from Fig. 4 that when the time delay constraint range is relatively small, more users choose to offload tasks to MEC computing. As the value of time delay constraint range increases, fewer users choose to offload. Since the smaller the time delay constraint range, the more delay-sensitive users are. When latency is more sensitive, the local computing CPU consumes more energy and consumes more energy. Offloading is often better than local computing at this time, so the higher the percentage of offloading. As the range of time delay constraints increases, the proportion of delay-sensitive users decreases. The greater the time delay constraint, the smaller the local CPU consumption, and the smaller the corresponding local computing energy consumption, users are more inclined to compute locally.

Furthermore, the comparison results of the total system energy consumption of proposed algorithm under different time delay constraints are shown in Fig. 5.

It can be seen from Fig. 5 that the total energy consumption of system is decreasing with the increase of time delay constraint range. In conjunction with the analysis in Fig. 4, since as the range of time delay constraints increases, more and more users choose local computing. The greater the time delay constraint, the more energy is saved in local computing. In addition, when the range of the time delay constraint is small, there are many delay-sensitive users. For these users, since more wireless resources are allocated during offloading, co-frequency interference based on frequency reuse will be more serious. This will cause an increase in system energy consumption. Therefore, as the scope of time delay constraint increases, the total energy consumption of the system becomes smaller and smaller.

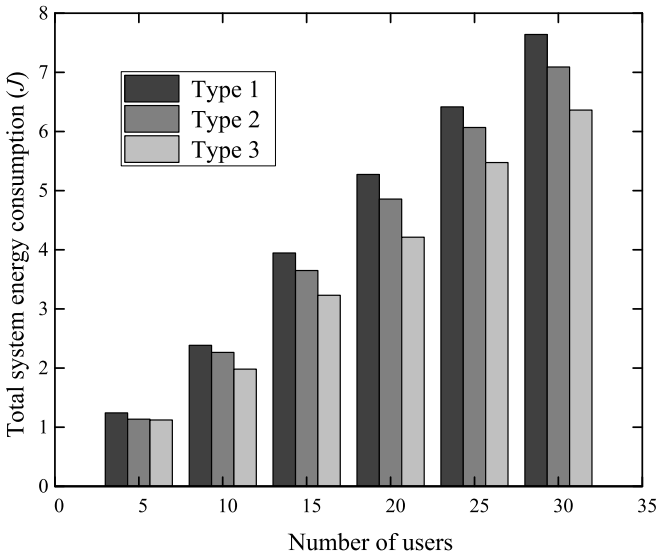

Fig. 5. Total energy consumption of the system under different time delay constraints.

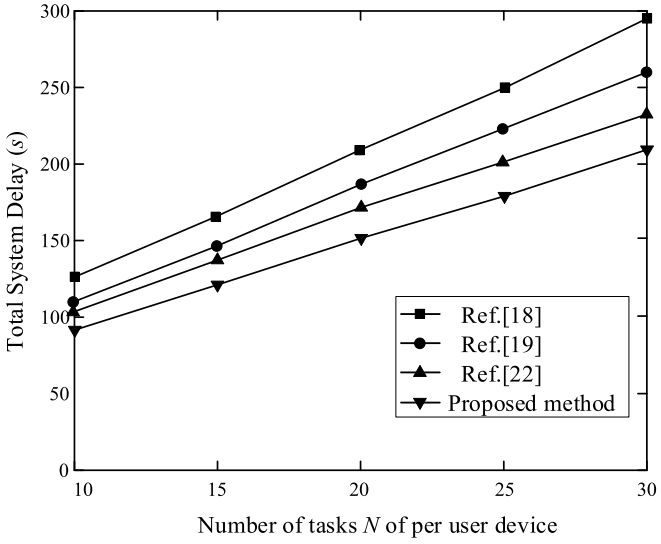

Fig. 6. The relationship between total system time delay and computing task.

5.3. Total system latency for different numbers of users/computing tasks

Considering the impact of different numbers of users and computing tasks on the total system time delay, simulation experiments are performed on two factors. When the number of users is fixed, the three comparison algorithms selected are: reference [18] random algorithm. The algorithm satisfied the buffer capacity limit of fog computing nodes, randomly selects the computing mode of each task, randomly allocates computing and wireless communication resources. Reference [19] jointly optimized offloading decisions and computing resources without considering the optimization of wireless communication resources. Reference [22] jointly optimized the offloading decision and wireless communication resources without considering the optimization of computing resources. The simulation results of the relationship between total system time delay and computing tasks are shown in Fig. 6.

As can be seen from Fig. 6, the overall time delay performance of our proposed algorithm is still optimal compared with the three existing comparison algorithms. When $M = 5$ and $N = 30$, the total time delay of proposed algorithm is reduced by 19% and 10% compared to reference [19] and reference [22] respectively. Since the computing and wireless communication resources in the system are jointly optimized, references [19] and reference [22] only optimize part of the system resources.

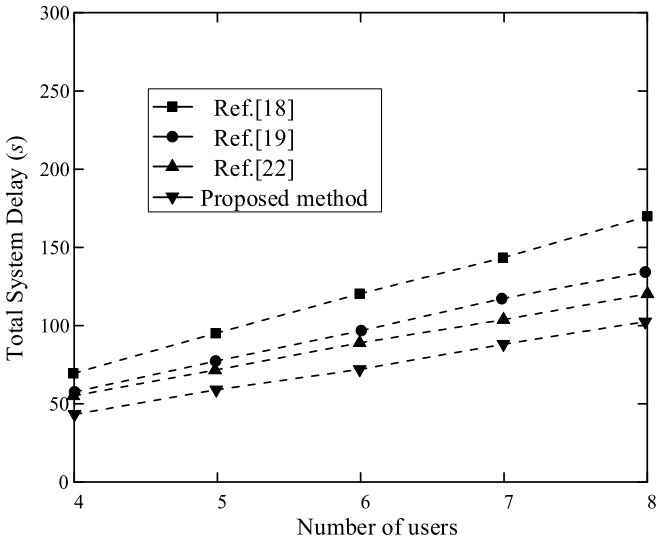

Fig. 7. The relationship between total system time delay and computing task.

Similarly, the simulation results of the relationship between the total system time delay and the number of users are shown in Fig. 7.

As can be seen from Fig. 7, under certain computing tasks, the total time delay increases with the number of users. However, the total time delay of proposed algorithm is relatively small compared with other algorithms. When the number of users increases, the algorithm will optimize the allocation of resources and offload certain network tasks, thereby ensuring that important calculations are performed to reduce the time delay.

5.4. The total system energy consumption for different input data sizes/number of users

Considering the impact of input data size on system performance, when the number of users is $N = 15$, the proposed algorithm and three algorithms (local computing algorithm, uninstall all algorithm, and the method in reference [24]) change with the input data., as shown in Fig. 8.

It can be seen from Fig. 8 that as the input data increases, the total energy consumption is increasing. Among them, the growth trend of all local computing is the most obvious, and the proposed algorithm has better performance than other algorithms. The reason is that as the input data increases, CPU cycles required for computing tasks also increase, the energy consumption of local computing will increase exponentially. Compared with other algorithms, the proposed algorithm jointly optimizes users' offloading decision and resource allocation. It can minimize the total energy consumption of the system, so it can get better system performance. Moreover, it can be seen that as the input data continues to increase, eventually all users tend to offload all to MEC computing. The reason is that as the input data increases, CPU cycles required for calculations increase. At this time, the local computing energy consumption is high, and users will be more inclined to offload computing tasks.

In addition, the proposed algorithm and three algorithms (local computing algorithm, uninstall all algorithm, and the method in reference [24]) are shown in Fig. 9 as the total number of users increases as the number of users increases.

As can be seen from Fig. 9, the proposed algorithm has greatly improved performance compared with local computing algorithm and

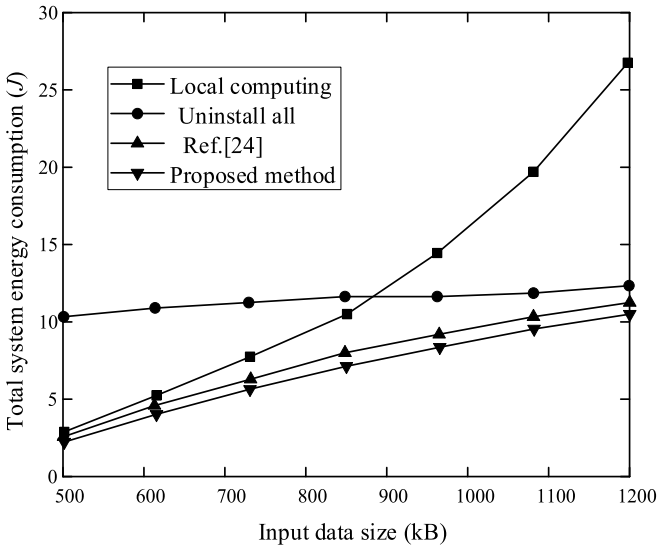

Fig. 8. Relationship between system energy consumption and input data size.

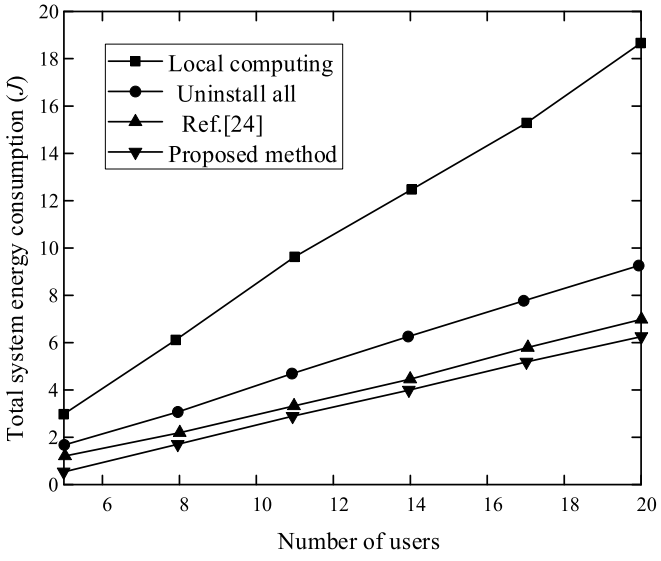

Fig. 9. Relationship between system energy consumption and number of users.

uninstall all algorithm. And compared with the algorithm in reference [24], the proposed algorithm can also get better performance. The reason is that the proposed algorithm formulates a joint optimization task offloading scheme, joint optimization offloading decision and resource allocation. The algorithm in reference [24] and the proposed algorithm both optimize offloading decisions and resource allocation. The proposed algorithm jointly optimizes channel allocation and power allocation in the optimization of offloading decision. The algorithm in reference [24] focuses on optimizing channel allocation. Besides, the optimization goal of the proposed algorithm is the total energy consumption, and then the power is optimized on the basis of channel allocation. It helps achieve the goal of reducing energy consumption and improving system performance. Thus, our proposed algorithm has better performance than the algorithm in reference [24].

6. Conclusion

With the arrival of the Internet of Things industry and the 5G era, the explosive growth of various smart devices will cause the future network to face huge data traffic shocks. MEC provides users with IT and cloud computing services at the edge of networks, which makes it possible for user equipment with limited resources to run computation-intensive applications. Therefore, this paper proposes a joint optimization scheme for task offloading and resource allocation based on edge computing in 5G communication networks, which makes optimal task offloading decision and resource allocation scheme based on the network environment in dense networks to improve the overall performance of networks. The proposed scheme combines edge computing and D2D communication technologies based on multi-user network system model for 5G edge networks, three modes for processing computationally intensive tasks are developed, including local computing, fog node computing, edge node computing. The corresponding time delay model, task execution model and offloading energy consumption computing model are constructed. In addition, the problem of computing task offloading is transformed into a joint optimization problem of time delay and energy consumption, the interior point method is used to solve this problem, and then we develop a corresponding optimization algorithm. Simulation platform is used to demonstrate the performance of our proposed scheme, including the impact of different user numbers/data sizes on system energy consumption and time delay. Finally, the experimental results show that the scheme can achieve better latency performance with controlling the energy consumption of system, which improves the performance of 5G mobile communication networks and the experience quality of end users.

The proposed method constructs a joint optimization problem of time delay and energy consumption, only considers the total processing time delay of all computing tasks in the system, and does not consider the execution order of computing tasks. In the future work, the execution order of computing tasks can be studied when the time delay sensitivity of each computing task in the system is greatly different. Different latency requirements of different computing tasks are satisfied, which further improves the experience quality of users. Moreover, with the continuous development and maturity of artificial intelligence theories and technologies, they are applied to 5G mobile communication networks, which can improve system performance and intelligence. For example, in the field of mobile communication network computing, the use of deep reinforcement learning to improve the decision-making level of computing resource allocation and computing offloading is also a hot topic in future work.

CRediT authorship contribution statement

Shi Yang: Conceptualization, Methodology, Software, Data curation, Funding acquisition, Validation, Project administration, Supervision, Formal analysis, Writing - original draft, Writing-review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] G.G. Haftu, Information communications technology and economic growth in Sub-Saharan Africa: A panel data approach, Telecommun. Policy 43 (1) (2019) 88–99.

[2] D. Kim, S. Kim, Network slicing as enablers for 5g services: state of the art and challenges for mobile industry, Telecommunication Systems 71 (3) (2019) 517–527.

[3] K. Mekki, E. Bajic, F. Chaxel, et al., A comparative study of LPWAN technologies for large-scale iot deployment, ICT Express 5 (1) (2019) 1–7.

[4] C.F. Liu, M. Bennis, M. Debbah, et al., Dynamic task offloading and resource allocation for ultra-reliable low-latency edge computing, IEEE Trans. Commun. 67 (6) (2019) 4132–4150.

[5] Lianyong Qi, Wanchun Dou, Chunhua Hu, Yuming Zhou, Jiguo Yu, A context-aware service evaluation approach over big data for cloud applications, IEEE Trans. Cloud Comput. (2015) http://dx.doi.org/10.1109/TCC.2015.2511764.
[6] Y. He, J. Ren, G. Yu, et al., D2d communications meet mobile edge computing for enhanced computation capacity in cellular networks, IEEE Trans. Wireless Commun. 18 (3) (2019) 1750–1763.
[7] K. Zhang, Y. Zhu, S. Leng, et al., Deep learning empowered task offloading for mobile edge computing in urban informatics, IEEE Internet Things J. 6 (5) (2019) 7635–7647.
[8] E. El Haber, T.M. Nguyen, C. Assi, Joint optimization of computational cost and devices energy for task offloading in multi-tier edge-clouds, IEEE Trans. Commun. 67 (5) (2019) 3407–3421.
[9] B. Gu, Z. Zhou, Task offloading in vehicular mobile edge computing: A matching-theoretic framework, IEEE Veh. Technol. Mag. 14 (3) (2019) 100–106.
[10] H. Xing, L. Liu, J. Xu, et al., Joint task assignment and resource allocation for D2D-enabled mobile-edge computing, IEEE Trans. Commun. 67 (6) (2019) 4193–4207.
[11] Lianyong Qi, Xiaokang Wang, Xiaolong Xu, Wanchun Dou, Shancang Li, Privacy-aware cross-platform service recommendation based on enhanced locality-sensitive hashing, IEEE Trans. Netw. Sci. Eng. (2020) http://dx.doi.org/10.1109/TNSE.2020.2969489.
[12] Z. Ding, J. Xu, O.A. Dobre, et al., Joint power and time allocation for NOMA–MEC offloading, IEEE Trans. Veh. Technol. 68 (6) (2019) 6207–6211.
[13] Z. Zhang, W. Zhang, F.H. Tseng, Satellite mobile edge computing: Improving qos of high-speed satellite-terrestrial networks using edge computing techniques, IEEE Network 33 (1) (2019) 70–76.
[14] D. Xu, Q. Li, H. Zhu, Energy-saving computation offloading by joint data compression and resource allocation for mobile-edge computing, IEEE Commun. Lett. 23 (4) (2019) 704–707.
[15] P. Yuan, Y. Cai, X. Huang, et al., Collaboration improves the capacity of mobile edge computing, IEEE Internet Things J. 6 (6) (2019) 10610–10619.
[16] J. Feng, Q. Pei, F.R. Yu, et al., Computation offloading and resource allocation for wireless powered mobile edge computing with latency constraint, IEEE Wirel. Commun. Lett. 8 (5) (2019) 1320–1323.
[17] Lianyong Qi, Wanchun Dou, Wenping Wang, Guangshun Li, Hairong Yu, Shaohua Wan, Dynamic mobile crowdsourcing selection for electricity load forecasting, IEEE Access 6 (2018) 46926–46937.
[18] L. Hu, Y. Tian, J. Yang, et al., Ready player one: UAV-clustering-based multi-task offloading for vehicular VR/AR gaming, IEEE Network 33 (3) (2019) 42–48.
[19] R. Ranji, A.M. Mansoor, A.A. Sani, EEDOS: An energy-efficient and delay-aware offloading scheme based on device to device collaboration in mobile edge computing, Telecommun. Syst. 73 (2) (2020) 171–182.
[20] Q. Jia, R. Xie, Q. Tang, et al., Energy-efficient computation offloading in 5G cellular networks with edge computing and D2D communications, IET Commun. 13 (8) (2019) 1122–1130.
[21] R. Dong, C. She, W. Hardjawana, et al., Deep learning for hybrid 5G services in mobile edge computing systems: Learn from a digital twin, IEEE Trans. Wireless Commun. 18 (10) (2019) 4692–4707.
[22] W. Ni, H. Tian, X. Lyu, et al., Service-dependent task offloading for multiuser mobile edge computing system, Electron. Lett. 55 (15) (2019) 839–841.
[23] L. Huang, X. Feng, C. Zhang, et al., Deep reinforcement learning-based joint task offloading and bandwidth allocation for multi-user mobile edge computing, Digit. Commun. Netw. 5 (1) (2019) 10–17.
[24] J. Zheng, L. Gao, H. Wang, et al., Joint downlink and uplink edge computing offloading in ultra-dense hetnets, Mob. Netw. Appl. 24 (5) (2019) 1452–1460.
[25] M.S. Elbamby, C. Perfecto, C.F. Liu, et al., Wireless edge computing with latency and reliability guarantees, Proc. IEEE 107 (8) (2019) 1717–1737.
[26] B. Cao, L. Zhang, Y. Li, et al., Intelligent offloading in multi-access edge computing: A state-of-the-art review and framework, IEEE Commun. Mag. 57 (3) (2019) 56–62.
[27] Z. Kuang, L. Li, J. Gao, et al., Partial offloading scheduling and power allocation for mobile edge computing systems, IEEE Internet Things J. 6 (4) (2019) 6774–6785.
[28] S. Misra, N. Saha, Detour: dynamic task offloading in software-defined fog for IoT applications, IEEE J. Sel. Areas Commun. 37 (5) (2019) 1159–1166.
[29] A. Asheralieva, D. Niyato, Hierarchical game-theoretic and reinforcement learning framework for computational offloading in UAV-enabled mobile edge computing networks with multiple service providers, IEEE Internet Things J. 6 (5) (2019) 8753–8769.
[30] M.A. Messous, S.M. Senouci, H. Sedjelmaci, et al., A game theory based efficient computation offloading in an UAV network, IEEE Trans. Veh. Technol. 68 (5) (2019) 4964–4974.
[31] M. Hu, L. Zhuang, D. Wu, et al., Learning driven computation offloading for asymmetrically informed edge computing, IEEE Trans. Parallel Distrib. Syst. 30 (8) (2019) 1802–1815.
[32] X. Qiu, L. Liu, W. Chen, et al., Online deep reinforcement learning for computation offloading in blockchain-empowered mobile edge computing, IEEE Trans. Veh. Technol. 68 (8) (2019) 8050–8062.
[33] J. Zhao, Q. Li, Y. Gong, et al., Computation offloading and resource allocation for cloud assisted mobile edge computing in vehicular networks, IEEE Trans. Veh. Technol. 68 (8) (2019) 7944–7956.
[34] Q.D. La, M.V. Ngo, T.Q. Dinh, et al., Enabling intelligence in fog computing to achieve energy and latency reduction, Digit. Commun. Netw. 5 (1) (2019) 3–9.
[35] Y. Zhang, J. Lopez, Z. Wang, Mobile edge computing for vehicular networks [from the guest editors], IEEE Veh. Technol. Mag. 14 (1) (2019) 27–108.
[36] Z. Zhang, Z. Hong, W. Chen, et al., Joint computation offloading and coin loaning for blockchain-empowered mobile-edge computing, IEEE Internet Things J. 6 (6) (2019) 9934–9950.
[37] Z. Zhou, H. Liao, X. Zhao, et al., Reliable task offloading for vehicular fog computing under information asymmetry and information uncertainty, IEEE Trans. Veh. Technol. 68 (9) (2019) 8322–8335.
[38] X. Wang, Y. Han, C. Wang, et al., In-edge ai: Intelligentizing mobile edge computing, caching and communication by federated learning, IEEE Network 33 (5) (2019) 156–165.

Shi Yang, has got his Master Degree of Computer Software and Theory, Associate Professor. He has graduated from Changchun University of Science and Technology in 2012. He is currently working at Changchun University of Finance and Economics. His research interests include cloud computing, internet of things and edge computing.