IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received May 7, 2019, accepted July 8, 2019, date of publication July 16, 2019, date of current version August 14, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2929075

Task Caching, Offloading, and Resource Allocation in D2D-Aided Fog Computing Networks

YANWEN LAN, XIAOXIANG WANG, DONGYU WANG, ZHAOLIN LIU, AND YIBO ZHANG
Key Laboratory of Universal Wireless Communication, Ministry of Education, Beijing University of Posts and Telecommunications, Beijing 100876, China
Corresponding author: Xiaoxiang Wang (cpwang@bupt.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61701038, and in part by the Fundamental Research Funds for the Central Universities.

ABSTRACT In this paper, we investigate the allocation of resource in D2D-aided Fog computing system with multiple mobile user equipments (MUEs). We consider each MUE has a request for task from a task library and needs to make a decision on task performing with a selection of three processing modes which include local mode, fog offloading mode, and cloud offloading mode. Two scenarios are considered in this paper, which mean task caching and its optimization in off-peak time, task offloading, and its optimization in immediate time. In particular, task caching refers to cache the completed task application and its related data. In the first scenario, to maximize the average utility of MUEs, a task caching optimization problem is formulated with stochastic theory and is solved by a GA-based task caching algorithm. In the second scenario, to maximize the total utility of system, the task offloading and resource optimization problem is formulated as a mixed integer nonlinear programming problem (MINLP) with a joint consideration of the MUE allocation policy, task offloading policy, and computational resource allocation policy. Due to the nonconvex of the problem, we transform it into multi-MUEs association problem (MMAP) and mixed Fog/Cloud task offloading optimization problem (MFCOOP). The former problem is solved by a Gini coefficient-based MUEs allocation algorithm which can select the most proper MUEs who contribute more to the total utility. The task offloading optimization problem is proved as a potential game and solved by a distributed algorithm with Lagrange multiplier. At last, the simulations show the effectiveness of the proposed scheme with the comparison of other baseline schemes.

INDEX TERMS Fog computing, computation offloading, resource allocation, cache, D2D, potential game.

I. INTRODUCTION

In recent years, the world has witnessed a growing number of intelligent devices and the accompanied wireless data traffic [1], [2]. It is foreseen that the mobile data traffic will increase even more significantly due to the development of the novel sophisticated applications, such as face recognition, interactive gaming and augmented reality [3]. These emerging applications and services need not only extensive computing capabilities and vast battery energy, but also high data rate. However, from the users' point of view, the computing capability of equipments are constrained, which has a serious impact on the delay performance and operational costs of services in fifth generation (5G) wireless networks.

To overcome such disadvantages, a new paradigm Fog computing network is proposed which provides cloud services at the edge of the network [4]. Fog computing is defined as a scenario where a huge number of heterogeneous ubiquitous and decentralized devices communicate and potentially cooperate among them to perform tasks without the intervention of third parties [5], [6]. With the help of Fog computing, MUEs no longer need to offload all of their tasks (e.g, high quality video streaming, mobile gaming, etc.) to the central and remote cloud, which enable their requirement to be satisfied at anytime and anywhere. By deploying numerous Fog notes (FNs) in the edge network, MUEs can offload their tasks to one of Fog servers or the cloud server, which can not only reduce the backbone traffic, but also decrease the latency for delay sensitive services [7], [8].

The models of Mobile edge computing and Fog computing are seemed similar, but actually have many differences in the

The associate editor coordinating the review of this manuscript and approving it for publication was Shree Krishna Sharma.

methods for monitoring, processing, and conveying data. The FCNs are heterogeneous based on different kinds of elements including routers, switches, access points, IoT gateways and so on [9]. This architecture enables data collection, processing and storage at the local area network which achieves less latency in comparison to the cloud. Edge computing refers to data processing at the edge of a network close to the data source within the Radio Access. The MEC nodes or servers are usually co-located with the Radio Network Controller which is close to mobile subscribers and are more independent in decision making. This results in offering context aware application and services with ultra-low latency and high-bandwidth requirements.

In mobile Cloud computing (MCC), significant contribution has been achieved [10]. Cloud server is supposed to have sufficient computing capability, but incurs long roundtrip corresponding transmit delay. In the mobile Fog computing system, the computing units are deployed on the side close to MUEs, which can largely save the transmit delay. However, due to the limited computiontal power in the Fog server, sometimes the quality of service (QoS) of each application cannot be guaranteed, in which situation efficient allocation of Fog computing resource should be considered [11].

Up to now, many precious works have been done in the scenario of Fog computing. To the best of our knowledge, their works can be classified into following aspects.

- Content offloading. The content caching is known as caching the popular contents in the caching entities (users equipments, Fog notes, etc.) located at edge network, with which the delay and energy consumption of end users who requesting content would be largely decreased. The researches in such aspect mainly concentrated on how to design the caching strategies [12]–[18]. Some works optimize the caching policy with a jointly consideration of user mobility [13], user social relationship [14] and D2D sharing [17], [18].
- Task offloading. The main concern of task offloading strategy is what, how and where to offload MUEs' tasks with the current network conditions. Various works have been done in order to achieve an optimal offloading policy [19]–[28]. The purposes of these works are to decrease the signal overhead [19], to maximize the total utility [20] and to decrease the serving delay or energy consumption [21]–[28] of MUEs with QoS requirements guaranteed.
- Resource allocation. As the channel conditions and the requested tasks of MUEs are heterogeneity, a joint resource allocation strategy includes channel allocation policy, transmit power allocation policy, and computational resource allocation policy are critical to the ultimate QoS of MUEs [29]–[34]. For this problem, the researchers have proposed various optimization strategies, such as the allocation radio resource [29]–[31], computing resources [32]–[34]and so on.

From now on, although the benefits of caching have received much attention on Fog computing networks, but the main concern of caching in most works is traffic offloading, which is not suitable for the hybrid services having a large data size and needing quantities of computation resource for processing (e.g., scenes rendering task in VR). Moreover, the potential advantages of D2D communication technology are still not totally explored in the scenario of Fog computing.

As MUEs have the caching ability which enables to cache related data of tasks requested, more gains can be achieved by adopting D2D. The advantages for applying task caching in D2D-aided Fog networks mainly lie in two points: firstly, when MUEs request the task they have cached, the computation and transmission can be omitted, which can bring local gains by saving computational energy and delay. Secondly, combined with D2D, lots of MUEs' requests can be satisfied by the cache of other adjacent MUEs thanks to a distributed task caching, which can provide a good chance of data sharing. By task caching, requests are more likely to be satisfied by the MUEs' local cache or other MUEs nearby. Thus the resource consumption and the serving delay can be largely decreased.

Inspired by the concept of task caching proposed in [35], we further investigate the benefits of task caching for the D2D-aided Fog computing networks. Task caching refers to the caching of completed task applications and their related data in MUEs' local caching entities which they have been processed. In this paper, we propose a caching enabled task offloading and resource allocation scheme in D2D-aided Fog compuing networks. The task caching policy, MUE association policy, task offloading policy and resource allocation policy are jointly considered and optimized. Our goal is maximizing the average utility of MUEs in terms of serving delay and process energy consumption.

There are two scenarios considered in this paper, which mean task caching in D2D networks and task offloading and its optimization in D2D-aided Fog networks. We consider these two scenarios separately as caching is occurred in off-peak time while the task offloading should be handled timely. Moreover, the task caching and the task offloading are happened and optimized in different periods. The task caching refers caching the task requested before based on the history or preference of requesting, in which time, future request is unknown and not happened. The decision of task caching policy is based on the preference of MUE, the attributes of requested tasks and the other chance of D2D sharing. The task offloading and its optimization is based on the status of task caching, current channel condition, etc. In addition, the purpose of task caching is to improve the performance when user requesting, in which time the future requests and the conditions of request users are both unknown.

The contribution of this paper are listed as follows.

1) For the scenario which means task caching and its optimization, we propose a novel task caching scheme in D2D-aided Fog wireless networks. Specially, by adopting

the stochastic theory, the average utility of MUEs with random caching probability is formulated.To improve the gains bring by local caching and D2D sharing, a task caching optimization problem is formulated. This problem is nonlinear programming problem and non convex. In order to solve it, a near-optimal task caching algorithm based on GA is proposed.

2) For the scenario of task offloading, a task offloading optimization problem is formulated with the purpose of maximizing the utility of system. In this problem, the MUEs association policy, offloading selection policy and computational resource allocation policy are taken into account in the problem.

3) Due to the NP-hard properties of the task offloading optimization problem, we decompose it into multi-MUEs association problem (MMAP) and mixed Fog/Cloud task offloading and and optimization problem (MFCOOP). On the one hand, with the consideration of preventing multiple-FN matching conflict, an Gini coefficient-based algorithm is proposed which can effectively select the MUEs who are most important to the total utility in each FNs. On the other hand, a distributed task offloading algorithm is proposed to solve the MFCOOP. Thus the optimal offloading policy and resource allocation strategy would be obtained.

4) We investigate the performance of the proposed scheme through extensive numerical experiments. Simulation results show that the proposed scheme outperforms other schemes.

The paper is organized as follows. The system model is described in Section II. The task caching problem and its solution are given in Section III. Section IV provides the formulation of task offloading and resource optimization problem, meanwhile, the solution to the problem is also given. Simulation results are presented in Section V. Finally, Section VI concludes this paper.

II. SYSTEM MODEL

In this section, we introduce an D2D-aided Fog computing system model with a hierarchical computing structure which consists of a set of MUEs and FNs. In such system, the FNs can provide MUEs seamless access and abundance of computing resources in their close proximity, while the cloud is seen as a supplement to FNs as it has sufficient computing resource. Each MUE has a choice to offload its tasks to FNs, remote cloud, or perform them locally. Fig. 1 illustrates an instant of such Fog computing system.

A. NETWORK MODEL

We assume there is M FNs in the system, denoted by $\mathcal{M} = (1, 2, \cdots, M)$. We further assume that there is K MUEs denoted by $\mathcal{K} = (1, 2, \cdots, K)$. Each FN is installed with a Fog server and connects to the remote Cloud via wired optical fibers. Assume the remote Cloud has sufficient computing resources, while the computing ability of each Fog server is limited. We model the Cloud server as a large number of virtual machines with each has a dedicated processing capacity of f_0 (in cycles per time unit). Similarly and without loss of generality, we model each Fog note as a virtual machine with a processing power of f_0 as same as that in the cloud server (in cycles per time unit) like the work in [9]. The servers help MUEs for task computing. If there is more than one MUEs accessed in a same FN, the processing power of the associated FN will be shared.

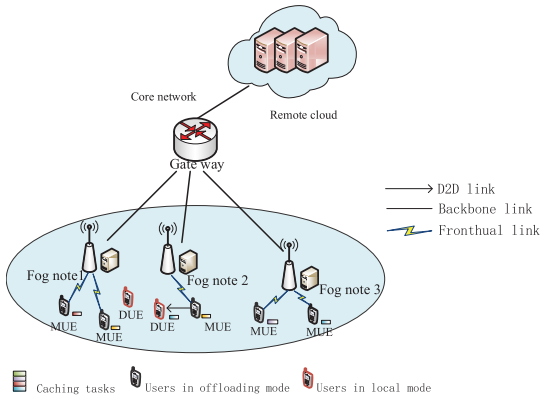


FIGURE 1. The considered D2D-aided fog computing system.

We denote $a_{i,m} \in \{0, 1\}, \forall i \in \mathcal{K}, \forall m \in \mathcal{M}$ as the association decision of MUE i. Specifically, $a_{i,m} = 1$ indicates that the MUE i is associated FN m, which means MUE i can offload its tasks to the FN m or relay them to the Cloud through FN m. Specially, if $\{a_{i,m} = 0\}_{\forall m \in \mathcal{M}}$, MUE i is not associated with offloading mode, where three cases will be occurred: i) The requested MUE has cached the required task. ii) The request can be satisfied by other MUEs who have cached the task by D2D. iii) It has to perform its task locally if it cannot be served by its local cache or other MUEs.

Assume MUEs are equipped with multi-RAT and may access more than one FN but only be served by a server in Fog or Cloud during task processing. We define a profile of the offloading decision as $\mathcal{Y} = \{y_1, y_2, \cdots, y_K\}$. Specially $y_i = 0$ means the MUE i is associated with Fog computing mode, otherwise, Cloud computing mode will be applied.

B. TASK CACHING MODEL

Assume there is a task library consists of N computation tasks denoted by $\mathcal{N} = \{1, 2, \cdots, N\}$. For heterogeneous computing tasks, we define $L_n(D_n, S_n, \theta_n)$ as the task of n, where D_n denotes total computational resource required by task L_n, which is presented as the number of cycles for processing a unit bit of data. Further more, S_n shows the data size in bits of task L_n. Different from many of works, we consider the transmission of task results cannot be ignored which is more practical as the results of many of the services have a certain amount of data. The ratio of data size after computation to data size before computation for L_n is denoted by θ_n.

Each MUE will random cache its task results(e.g.,task related data) after the completion of task processing

according a carefully designed caching policy. We denote the caching probability distribution by $\mathcal{Q} = \{Q_1, Q_2, \cdots, Q_N\}$, where Q_N is the caching probability for the result of task N. In our setting, each MUE cache task or request task once while different MUEs can request the same task based on their preferences. In this study, the requested MUEs are called requesters while the MUEs who have cached the task needed by others are called responders. In many scenarios that task requests are highly concentrated in the spatial domain and asynchronously or synchronously repeated in the time domain, storing computation results for future reuse can greatly reduce the computation burden and latency. For instance, in augmented reality subscriptions for better viewing experience in museums, a processed augmented reality output may be simultaneously or asynchronously used by visitors in the same place [28]. When the task is processed and obtained by requesting MUEs, the task processing results should not be abandoned as they are useful for the future requests. Consider that the requested task would be asked again in the furfure time or be used by other D2D MUEs nearby. In this paper, we consider a random task caching policy because a deterministic caching schemes is difficult for the management and updating of the caching results as the idle capacity of MUEs and the size of results are heterogeneous. What's more, consider the future requests are random, the random cache caching policy is more able to achieve a high diversity caching placement which can improve the chance of D2D sharing and bring a potential gain for task offloading.

The process of task caching and offloading is as follows. A requester firstly requests a task according its personal preference. If the task has been cached on the its local cache, it can be directly satisfied without consuming computational resource. Otherwise, if there are at least one responder nearby, an optimal D2D transmission link will be established which enables the related results subsequently delivered with a cost of transmission. If neither of its local cache nor the adjacent MUEs can serve the request, Fog offloading mode or Cloud offloading mode will be adopted.

For simplicity, we assume that the requests of MUEs follow the same Zipf distribution $\mathcal{P} = (P_1, P_2, \cdots, P_N)$, in which the popularity is ranked in descending order. The distribution of many Internet services was proven to follow Zipf's law. Similar to Internet services, the distribution of computing services also follows Zipf's law [28]. As the assumption in [28], [35], in this paper, we assume the popularity of tasks is follow a Zipf's law. The popularity of tasks can be calculated by

$$P_n = \frac{n^{-\beta}}{\sum_{j=1}^{N} j^{-r}}, \quad \forall n \in \mathcal{N} \tag{1}$$

where exponent β is the popularity distribution parameter which reflects the skewness of popularity.

C. COMMUNICATION MODEL

There are two communication modes include D2D communication and celullar communication. In this paper, all the MUEs occupie the orthogonal spectrum in both D2D link and cellular link and the cell reuses the bandwidth resource of another cell. That ensure there are no interfere between MUEs no matter which mode they are associated with in a same cell.

1) COMMUNICATIONS IN D2D NETWORKS

The direct discovery strategy is considered in this paper [36]. UE devices participated in the device discovery process to periodically transmit/receive discovery signals synchronously. In a Device discovery period, a requester will transmit discovery signals that may be detected by other UE devices. The information in the discovery signals should include identity and application-related information (e.g., cache state). The corresponding responder who have cached the requested task would response to the discovery signal. The requester will establish a connection with the most proper responder with a maximal downlink signal strength.

In the cache-enabled D2D networks, if a requester cannot be satisfied by their local caches, it should ask from other MUEs by D2D communications. Specially, for a D2D link between requester i and its responder j, the transmit rate can be calculated by

$$r(x_{i,j}) = Blog(1 + \frac{p_j x_{i,j}{}^{-a} g_{i,j}^{d2d}}{\sigma^2 + \sum\limits_{z \in K, z \neq i} I_{z,j}}) \tag{2}$$

where p_j denotes the transmit power of MUEs j, $x_{i,j}$ is the distance between requester i its responder j, a represents the path loss exponent, $g_{i,j}^{d2d}$ means the small-scale fading coefficient, σ^2 is the noise power, I is the interfere come from other concurrent transmit links occupying the same channel.In this paper, we assume all of MUEs have the same transmit power(e.g.,$p_i = p_j, \forall i, j \in \mathcal{K}$). For convenience, we use p_u to denote the transmit power of MUEs.

According to Formula (2), the time consumption of for the delivery the results of task n from the responder j can be calculated by

$$t_{i,j,n} = \frac{\theta_n S_n}{r(x_{i,j})} \tag{3}$$

We assume the distribution of MUEs follows the Poisson distribution with a parameter of λ. According to the feature of the Poisson process, the distribution of requesters for task n and its responders are modeled as two mutually independent homogeneous Poisson point processes (PPPs). As mentioned before, eacch MUE have the chance to become a requester or responder which is determined by their caching state and other neaby MUEs' requesting state. In instance, all of MUEs have a probability to cache the result of task n that can be the potential responders. Thus, we can get that the distribution of task n is following the Poisson distribution with the parameter of $Q_n\lambda$.

For task n, considering the geographical locations of MUEs, the distance between a requester and its nearest responder cannot be directly got, while the probability density function of the association distance can be obtained according to the works in [37]

$$f(x_n, Q_n) = 2\pi x_n \lambda Q_n e^{-\lambda Q_n \pi {x_n}^2} \tag{4}$$

where x_n means the distance between a requester and its nearest responder of task n.

Assume MUE i is a requester of task n and MUE j is the nearest responder, by replacing the parameter $x_{i,j}$ by x_n in Formula (2) and integrating Formula (2)-(4), the average transmit rate for the results of task n can be represented as

$$\overline{r}_n(Q_n) = \int_0^R r(x_n) f(x_n, Q_n) dx_n \tag{5}$$

where R is maximum communication distance of D2D comunication.

The average D2D comunication time of task n' results for a typical MUE i can be caculated by

$$t_{i,n}^D = \frac{\theta_n S_n}{\overline{r}_n(Q_n)} \tag{6}$$

The corresponding average energy consumption for a requester i can be got by

$$E_{i,n}^D = p_u t_{i,n}^D \tag{7}$$

2) COMMUNICATIONS IN CELLULAR NETWORKS

As mentioned before, if a requester cannot find its responders nearby, offloading mode should be adopted. For a requester in offloading mode, the task uploading in uplink and result delivery in downlink should both be considered.

The transmit rate of requester i associated with the Fog note m can be calculated by

$$R_{i,m}^u = B \log(1 + \frac{p_u l_{i,m}^{-a} g_{i,m}^u}{\sigma^2 + \sum_{o=1, o\neq m}^{M} \sum_{j=1, j\neq i}^{K} p_u l_{j,m}^{-a} g_{j,m}^u}) \tag{8}$$

where $l_{i,m}$ denotes the distance between UE i and the Fog note m, $g_{i,m}^u$ is the channel gain of uplink.

Thus, the uploading delay of task n for MUE i can be calculated by

$$t_{i,m,n}^u = \frac{S_n}{R_{i,m}^u} \tag{9}$$

The corresponding energy consumption in uplink can be presented as

$$E_{i,m,n}^u = p_u t_{i,m,n}^u \tag{10}$$

Smilarlly, we can get its downlink rate, which can be calculated by

$$R_{m,i}^d = B \log(1 + \frac{p_m l_{i,m}^{-a} g_{i,m}^d}{\sigma^2 + \sum_{o=1, l\neq m}^{M} \sum_{j=1, j\neq i}^{K} p_o l_{i,o}^{-a} g_{i,o}^d}) \tag{11}$$

where p_m is the transmit power of FN m, $g_{i,m}^d$ is the channel gain of the offloading downlink.

The delivery delay and its corresponding energy consumption can be respectively represented as

$$t_{m,i,n}^d = \frac{\theta_n S_n}{R_{m,i}^d} \tag{12}$$

$$E_{m,i,n}^d = p_m t_{m,i,n}^d \tag{13}$$

D. COMPUTING MODEL

For a requester i cannot find its responders nearby, one of the computing modes will be associated which include local computing mode, Fog computing mode and Cloud computing mode.

1) LOCAL COMPUTING

Let w_i denotes the the computational power of MUE i, the computation execution time of task n by local computing can be calculated by

$$t_{i,n}^l = \frac{D_n}{w_i} \tag{14}$$

We use the same model of energy consumption as in [38]. The energy consumption of computing locally can be represented as

$$E_{i,n}^l = \kappa_i D_n (w_i)^2 \tag{15}$$

where κ_i denotes the energy effective switched capacitance of MUE i relying on the its chip architecture.

2) FOG COMPUTING

The computation execution time of FN m for the task n requesting by MUE i can be calculated by

$$t_{i,m,n}^f = \frac{D_n}{f_{i,m}} \tag{16}$$

where $f_{i,m}$ represents the allocated commputational resource in MEC server for MUE i who connects to FN m.

The energy consumption of computing in FN m can be represented as

$$E_{i,m,n}^{cpt} = \kappa_{server} D_n (f_{i,m})^2 \tag{17}$$

where κ_{server} denotes the energy effective switched capacitance of Fog servers.

3) CLOUD COMPUTING

Similar to Fog computing, for a typical MUE i, the computation execution time of task n be processed in cloud can be calculated by

$$t_{i,n}^c = \frac{D_n}{f_0} \tag{18}$$

The corresponding energy consumption for computing can be represented as

$$E_{i,n}^{cpt} = \kappa_{server} D_n (f_0)^2 \tag{19}$$

TABLE 1. Some parameter notations used in this paper.

Symbol	Definition
M	The number of FNs
K	The number of MUEs
N	The number of tasks in task library
K_T	The maximum associated number of MUEs in each FNs
$\mathcal{M}$	The set of FNs
$\mathcal{K}$	The set of MUEs
$\mathcal{K}_m$	The set of MUEs in FN m
$\mathcal{K}_m^*$	The associated MUEs in FN m
$\mathcal{N}$	The library of tasks
$\mathcal{Y}$	The offloading strategy set of MUEs
θ_n	The ratio of task results' size to tasks' size for n
P_n	The request probability of task n
Q_n	The caching probability of task n
p_u	The transmit power of MUEs
p_m	The transmit power of FN m
$\overline{r}_n$	D2D average transmit rate for the results of task n
t_n^D	D2D average transmit delay for the results of task n
$t_{i,m,n}^u$	The uploading time of MUE i to FN m for task n
$t_{i,m,n}^d$	the downlink delay from FN m to MUE i for task n
$t_{i,n}^l$	The Local computing time by MUE i for task n
$t_{i,n}^c$	The Cloud computing time of MUE i for task n
$t_{i,m,n}^f$	The Fog computing time of MUE i in FN m for task n
ρ^t	The revenue coefficient per unit of saved delay
ρ^e	The revenue coefficient per unit of saved energy
$E_{i,n}^D$	The average energy consumption for getting the results of task n in D2D
$E_{i,n}^l$	The energy consumption of MUE i for task n processing locally
$E_{i,m,n}^c pt$	The energy consumption of MUE i for task n processing in FN m
$E_{i,n}^c pt$	The energy consumption of MUE i for task n processing in Cloud
$R_{i,m}$	The transmit rate of MUE i associated with Fog note m
$E_{i,m,n}^u$	The energy comsuption of transmition in uplink from MUE i to FN m for task n
U^o	The total utility of system for MUEs in offloading mode
U^{ch}	The average utility of MUEs in local mode

It should be mentioned that the energy effective switched capacitance in Fog servers and the cloud server are seemed as equally in this paper.

III. TASK CACHING PROBLEM AND ITS NEAR-OPTIMAL SOLUTION

In this section, we introduce the formation of task caching optimization problem. It should be noticed that task caching is carried out in off peak time(e.g., midnight) when the number of requests and the load of networks are both small, which ensure the caching placement policy and the caching updating policy be conveied and implemeted efficiently and accurately. The caching policy and the cache updating policy should well designed according to the analysis of MUEs' performance during long-term statistics. Considering the fact that the different properties and performance criterions of delay and energy of tasks, We use the subtraction between system revenues and costs for delay decreasing and energy saving as the system utility function.

A. PROBLEM FORMATION

Comparing to the task offloading and local processing, task caching and sharing in cached-enabled D2D networks can largely save the energy consumption and the delay for computing and the transmission of task-realted data. The aim of caching scheme optimization is to maximize the total benefit gained by the MUEs' local cache and contents sharing.

In order to evaluate the benefits of saving time and energy for the future requests in D2D networks, it is necessary to obtain the expectation of caching gain in random case as the future requests and caching state is random. According the state of task caching in D2D networks, there are two cases which mean self-satisfaction, D2D satisfaction.

Case I: self-satisfaction

If the request task has already been processed and has been cached in the local cache of a requester, the task result would directly satisfy the requester without any additional computational cost.

The average utility of task caching bring by local caching can be expressed as

$$U_{ch}^l \triangleq \left\{ \sum_{i=1}^{K} \sum_{n=1}^{N} P_n Q_n (\rho^t t_{i,n}^l + \rho^e E_{i,n}^l) \right\} / K \tag{20}$$

In Formula (20), $(P_n Q_n)$ represents the probability of each MUE who requests task n that can be satisfied by its local cache. $(\rho^t t_{i,n}^l + \rho^e E_{i,n}^l)$ means the the utility by preventing processing task locally in the case of self-satisfaction.

Case II: D2D-satisfaction

If the request task has not been cached in the requester's local cache but can be got from its responder, a D2D transmission link will be established and the task result would be directly delivered with the cost of transmission.

The average utility of task caching bring by D2D sharing can be expressed as

$$U_{ch}^d \triangleq \left\{ \sum_{i=1}^{K} \sum_{n=1}^{N} P_n (1 - Q_n) \left[\rho^t (t_{i,n}^l - t_{i,n}^D) + \rho^e (E_{i,n}^l - E_{i,n}^D) \right] \right\} / K \tag{21}$$

where ρ^t denotes revenue coefficient per unit of saved delay compared with local computing. ρ^e is the revenue coefficient per unit of saved energy compared with local computing.

In Formula (21), $[P_n(1 - Q_n)]$ represents the probability of each MUE who requests task n that can't be satisfied by its local cache. $[(\rho^t (t_{i,n}^l - t_{i,n}^D) + \rho^e (E_{i,local} - E_{i,n}^D)]$ means the utility obtained by decreasing processing time and saving energy compared with local processing in the case of D2D-satisfaction.

In consequence, the optimization problem of task caching in our scenario is shown as

$$\begin{aligned} &max\ U^{ch} = U_{ch}^l + U_{ch}^d \\ &s.t.\ C1: Q_n \geq 0, \forall n \in \mathcal{N} \\ &\quad C2: \sum_{n=1}^{N} Q_n \leq 1, \forall n \in \mathcal{N} \end{aligned} \tag{22}$$

Constraints C1 means all of tasks in the task library have a possibility to be cached by MUEs. C2 limits the cumulative cache probability of all tasks in the library to 1.

By integrating the Formula (5) - (7), we can get a reformation of Formula (21) as

$$U_{ch}^{d} = \sum_{i=1}^{K}\sum_{n=1}^{N} P_n(1-Q_n)\left[\rho^t t_{i,n}^{l} - \frac{\rho^t \theta_n S_n}{\overline{r}_n(Q_n)} + \rho^e(E_{i,n}^{l} - \frac{\rho^e p_u \theta_n S_n}{\overline{r}_n(Q_n)})\right] \quad (23)$$

The Formula (23) indicates the utility bing by D2D sharing. From the observations of Formula (23), the utility function of D2D sharing is nonlinear about variable P_n ($\forall n \in \mathcal{N}$). Therefore the problem of (22) is nonlinear and noconvex, which is difficult to find an optimal solution in polynomial time.

B. NEAR-OPTIMAL TASK CACHING OPTIMIZATION

In this section, we give the solution to the task caching optimization problem. As mentioned before, it is hard to obtain the optimum equilibrium solution to problem (22) in polynomial time. Thus, genetic algorithm(GA) is adopted in order to obtain a sub-optimal solution.

Genetic algorithm is a heuristic search method to approximate the optimal solution, which is inspired by Darwin's natural evolution theory. The process of the inheritance algorithm embodies the ''natural selection theory'' in evolution theory, in which the most suitable individual will be chosen to breed the next generation. The algorithm is widely used in machine learning, combinatorial optimization and intelligent computing.

Similarly, in the genetic algorithm, each solution is seemed as an individual which is coded into a binary ''Gene string''. Whether a solution will be selected or not in each ''generation'' is based on its fitness value. A higher the fitness value means a higher chance of the individual will survive and reproduce. After a process of selection, gene crossing, Gene mutation, these selected individuals will form a new population. The crossover mechanism exchanges some bits in the Gene strings according to a defined crossover probability. The mutation maintains the diversity in the population by altering bits of strings randomly, which prevent the algorithm from falling into a local optimum. The details of the near-optimal solution based on GA are presented in Algorithm 1.

In the algorithm, the fitness metric is defined as the objective function of (22). The caching probability for each task is encoded into binary strings of the length of $g_e = \lceil \log_2 j \rceil$, where j is the caching probability accuracy rate expressed in percentage. In this paper, we set j as 100 with the consideration of the complexity. As there are N tasks in the task library, the length of an individual in the gene pool is represented as $\sum_{i=1}^{i=N} \lceil \log_2 j \rceil$.

At the beginning of the algorithm, a population of M individuals will be randomly generated. Then after a number

Algorithm 1 Near-Optimal Task Caching Algorithm Based On GA

Input: $\mathcal{K}, \mathcal{N}, \lambda, \beta, P_{cr}, P_{mu}, t_{max}$

Output:

The sub-optimal caching distribution vector $\mathcal{P}^* = (P_i^*, P_2^*, \cdots, P_N^*)$

1: Initialize a $M \times N$ matrix randomly as the first population;
2: **for** $t = 1$ to $t = t_{max}$ **do**
3: Selection with the roulette wheel selection scheme;
4: Crossover with the probability P_{cr};
5: Mutation with the probability P_{mu};
6: Remove the individuals not satisfying the requirements C1, C2;
7: Calculate the object value according to (22) and record the best individuals;
8: **end for**
9: Select the best individual $Q^* = (Q_i^*, Q_2^*, \cdots, Q_N^*)$ as the near-optimal solution and output;

generation of evolutions, the best individuals will be obtained, which means the optimal solution. In each generation of evolutions, selection, crossover, and mutation will be implemented and a new population will be produced for future evoluting.

In the selection process, the roulette wheel selection scheme is adopted to implement proportionate selection. the roulette wheel selection scheme easure the probability of each individual being selected is proportional to its fitness value. The process are as follows:

1) Calculate the fitness of each individual in each population.

2) Calculate the probability that each individual is chosen to be inherited into the next generation of population by $P(x) = \frac{U^{ch}(x)}{\sum_{y=1}^{M} U^{ch}(y)}$, where $U^{ch}(x)$ means the fittness value for individual x.

3) Calculate the cumulative probability of each individual.

4) produce a random value from zero to one and compare it with the cumulative probability for each individual, if the cumulative probability is the larger one, the individual will be chosed.

The crossover probability and the mutation probability are denoted as P_{cr} and P_{mu} respectively. The two values are adpated upated according the difference among fitness values in each population. In our algorithm, we simplly set $P_{cr} = 1/(U_{max}^{ch} - U_{aver}^{ch})$ and $P_{mu} = 1/(U_{max}^{ch} - U_{aver}^{ch})$ in each population, where U_{max}^{ch} means the maximum fittness value of individuals before current iteration, U_{aver}^{ch} presents the average fittness value in each population.

In the new population, due to the constraints C1 and C2, the individuals which are not satisfied the two constraints will be abandoned, and the left individual with the best fitness will be recorded. Then the new generation of population will

be served for the succeeding iteration until the repeat time reaches. At last, select the best individual with the maximum fitness value of the recorded individuals and decode it into decimal numeral system. The sub-optimized caching probability distribution vector obtained represented as $\mathcal{Q}^* = (Q_1^*, Q_2^*, \cdots, Q_N^*)$.

IV. TASK OFFLOADING OPTIMIZATION PROBLEM AND SOLUTION

If a requester has not cached the task it need nor can find a responder nearby, task offloading would be needed to relay the task to Fog notes or Cloud. Like the utility function of task caching, the utility for task offloading is seemed as the benefits of delay saving and energy saving compared to the local computing.

A. PROBLEM FORMATION

We firstly discuss the utility of offloading to FNs. For MUEs who have been associated with fog offloading mode, the time consumption includes three parts: offloading delay, delivery delay and computing delay. The toc N for MUE i assoiated with FN m can be presented as

$$t_{i,m,n}^{fog} = t_{i,m,n}^{u} + t_{m,i,n}^{d} + t_{i,m,n}^{f} \tag{24}$$

Similarly, the total energy consumption includes: offloading energy, computing energy and delivery energy, which is presented as

$$E_{i,m,n}^{fog} = E_{i,m,n}^{u} + E_{i,m,n}^{cpt} + E_{m,i,n}^{d} \tag{25}$$

The utility for Fog offloading can be caculated by

$$U_i^{fog} \triangleq \rho^t(t_{i,n}^{l} - t_{i,m,n}^{f}) + \rho^e(E_{i,n}^{l} - E_{i,m,n}^{fog}), \quad \forall m \in \mathcal{M}, \forall n \in \mathcal{N} \tag{26}$$

For the MUE i with Cloud offloading mode, the uploading delay includes fornthual delay from i to its relay FN m denotes by $t_{i,m,n}^{u}$, the backhual delay from FN m to the remote cloud server denotes by $t_{m,o}^{u}$. The delivery delay in downlink includes two parts in uplink, denote by $t_{m,i}^{d}$ and $t_{o,m}^{d}$ separately. We let T_c denote the round-trip delay between FNs to Cloud (e,g, $T_c = t_{m,o}^{u} + t_{o,m}^{d}$).

The total time consumption for requesder i associated to FN m for processing task n in cloud offloading mode can be caculated by

$$t_{i,m,n}^{cloud} = t_{i,m,n}^{u} + t_{m,i,n}^{d} + t_{i,n}^{c} + T_c \tag{27}$$

The energy consumption can be represented as

$$E_{i,m,n}^{cloud} = E_{i,m,n}^{u} + E_{m,i,n}^{d} + E_{i,n}^{cpt} \tag{28}$$

According to the definition of utility function, the utility cloud offloading can be calculated by

$$U_i^{cloud} \triangleq \rho^t(t_{i,n}^{l} - t_{i,m,n}^{cloud}) + \rho^e(E_{i,n}^{l} - E_{i,m,n}^{cloud}), \quad \forall m \in \mathcal{M}, \forall n \in \mathcal{N} \tag{29}$$

The task offloading optimization problem can be represented as

$$\begin{aligned} U^0(A, Y) = & \sum_{i \in K} \sum_{m \in M} a_{i,m} \left[y_i^m U_i^{fog} + (1 - y_i^m) U_i^{cloud} \right] \\ & s.t.\ C1' : f_{i,m} \in (0, f_0) \forall i \in K, \quad \forall m \in M \\ & C2' : \sum_{i}^{K} f_{i,m} \leq f_0, \quad \forall i \in K, \forall m \in M \\ & C3' : y_i^m \in \{0, 1\}, \forall i \in K, \quad \forall m \in M \\ & C4' : \sum_{i}^{K} y_i^m \leq K_T, \quad \forall i \in K, \forall m \in M \\ & C5' : a_{i,m} \in \{0, 1\}, \quad \forall i \in K, \forall m \in M \end{aligned} \tag{30}$$

Constraints C1' ensures the allocated computatul capacity of each FN for all the tasks is not less than zero. Constraints C2' means the allocated computational resource for the MUEs who associated with it cannot exceed its maximum computational power. Constraints C3 means there is only two offloading mode to be chosen, cloud offloading mode and Fog offloading mode. C4' states that the number of MUEs accessed in each FN should be less than the maximum accessible number.Constraints C5' is proposed to guarantee each MUE cannot be associated with more than one FN at the same time.

From the observation of task offloading optimization problem (30), we can see that $a_{i,m}$ is binary resulting in the non-convexity of objective function and feasible sets. The problem is a mixed discrete and non-convex optimization problem, which is challenging to find the global optimum. Moreover, a joint consideration of MUE association and offloading mode selection makes centralized algorithm commonly a high computation complexity, which is not practical.

B. DISTRUBUTED TASK OFFLOADING OPTIMIZATION

Due to the nonconvex property of problem (30), We reformulate it by decomposing it into two sub problems, which are named multi-MUEs association optimization problem (MMAP) and mixed Fog-Cloud task offloading and optimization problem (MFCOOP). We designed a Gini coefficien-based near-optimal MUE allocation algorithm to solve MMAP. Then based on the results of MUE association, we will give an optimal solution to MFCOOP. From the combination of these two solutions, a suboptimal solution to task offloading problem will be obtained.

1) GINI COEFFICIENT-BASED MMOP OPTIMIZATION

In order to solve MMAP, We will adopt **Gini coefficient** and design a gini coefficient-based MUE association algorithm. The **Gini coefficient** can effectively obtain the set of MUEs that most contributed to total utility for each FN, which was verified in the work [39]. We modify their study and further apply it to multi-user and multi-base station matching scenario in which matching conflict is ubiquitousness as each MUEs may access more than one FN. The Gini coefficient,

between 0 and 1, is major for assessment on regional income inequality. A smaller Gini Coefficient means a more equality of the utility distribution (i.e., there is more MUEs who contribute to the majority of total obtained utility), and vice versa. With the matching policy, the MUEs who are abandoned by all FNs will be associated with local computing mode.

Due to the resource for each requester is not allocated, their total utility cannot be directly obtained. In order to calculate the Gini coefficient, we introduce an income function to reflect the total utility with adequate resources for each requester. The income function is defined as follows

$$\Gamma_{i,m,n} = \frac{\rho^t t_i^{local}}{t_{i,m,n}^T + \widetilde{t}_{i,m}(f_0)} + \frac{\rho^e E_{i,n}^l}{E_{i,m,n}^T + \widetilde{E}_{i,m,n}^{cpt}(f_0)}, \quad \forall i \in \mathcal{K}, \forall m \in \mathcal{M} \forall n \in \mathcal{N} \tag{31}$$

where $t_{i,m}^T = t_{i,m,n}^u + t_{m,i,n}^d$ and $E_{i,m,n}^T = E_{i,m,n}^u + E_{i,m,n}^d$, denote the total transmission delay and total transmission energy consumption respectively. $\widetilde{t}_{i,m,n}(f_0) = \frac{D_n}{f_{i,m}}$ and $\widetilde{E}_{i,m,n}^{cpt}(f_0) = \kappa_{server} D_n(f_0)^2$ respectively denote the computational delay and computational energy consumption when all the computational resource are allocated to MUE i.

For each FN, we calculate the income of the accessible MUEs according to the Income Function, then sort the income in ascending order (e.g., $\Gamma_1' \leq \Gamma_2' \leq \cdots \leq \Gamma_{\mathcal{K}_m}'$) with the ordered set of MUEs, $\{\mathcal{K}_{m,|K_m|}, \mathcal{K}_{m,|K_m-1|}, \cdots, \mathcal{K}_{m,1}\}$.

According to [23], the Gini Coefficient and the number of MUEs contributed more to the total utility can be defined as

$$Gini_m = 1 - \frac{1}{|\mathcal{K}_m|}(1 + 2\sum_{i=1}^{|\mathcal{K}_m|-1} b_i), \quad \forall i \in \mathcal{K}_m, \forall m \in \mathcal{M} \tag{32}$$

$$K_m^* = \min(\frac{1}{Gini_m} + \gamma_i(|\mathcal{K}_m| - \lceil \frac{1}{Gini_m} \rceil), |\mathcal{K}_m|) \tag{33}$$

where b_i can be expressed as

$$b_i = \frac{\sum_{j=1}^{i} \Gamma_j}{\sum_{j=1}^{\mathcal{K}_m} \Gamma_j}, \forall i \in \mathcal{K}_m, \forall m \in \mathcal{M} \tag{34}$$

and γ_i is the modify weight factor of $|\mathcal{K}_m| - \lceil \frac{1}{Gini_m} \rceil$, which can be calculated by

$$\gamma_i = \frac{\min(\frac{f_0}{\arg max\{D_n\}}, |\mathcal{K}_m|, K_{max})}{|\mathcal{K}_m|} \tag{35}$$

where K_{max} is the maximum number of accessible MUEs in each FN.

Then we can get the selection decision for each FN.

$$\mathcal{K}_m^* = \{\mathcal{K}_{m,|K_m|}, \mathcal{K}_{m,|K_m-1|}, \cdots, \mathcal{K}_{m,|K_m+1-K*_m|}\}, \quad \forall i \in K_m, \forall m \in \mathcal{M} \tag{36}$$

where $\mathcal{K}_{m,|K_m|}$ means the $|K_m|th$ MUE in the order set of MUEs in m.

There exit the situation that one MUE is associated with multiple FNs.

The mach conflict occurred if the following condition was met

$$\mathcal{K}_m^* \cap \mathcal{K}_v^* \neq \emptyset, \quad \forall m, v \in M, \ m \neq v \tag{37}$$

If an MUE i has been associated with more than one FN, in order to help it chose the most proper FN, we only keep its associate policy which can help it reach the maximum of income, (e,g, $a_{i,v} = 1$). Then abolish all other associated policy $\{a_{i,m}\}_{m \in \mathcal{M}, m \neq n} = 0$. The process of eliminating conflicting which can be expressed as

$$m^* = \arg\max_m \Gamma_{i,m,n}, \forall i \in \mathcal{K}_m \forall i \in \mathcal{N} \quad \{a_{i,m}\}_{\forall m \in \mathcal{M} \setminus m^*} = 0 \tag{38}$$

After an operation of conflict eliminating, the current optimal association will be reserved. The MUE mache policy in the FN of i is unchanged while the associate number of MUEs in other FNs decrease and should be supplemented.

For an FN whose associated MUE has been grabbed, the abandoned MUE with the maximal income in the abandoned User group would be reselected.

The Gini-coefficient based muti-MUE muti-FN matching algorithm is shown is algorithm 2.

By the Gini coefficient based association algorithm, the MUEs who make major contributions to system utility will be associated with most proper FNs, while the other MUEs who are abandoned by all the accessible FNs will select local computing mode.

2) MCFOOP GAME AND OPTIMIZATION

Due to the existence of competition for resources in FNs, the utility in an FN will be influence by the number of access MUEs, the required computing resources, some tasks should be relayed to the cloud server to ensure the maximization of total utility.

In order to determine which tasks should be offloaded to Fog server or remote Cloud, we formulate the interactions between the MUE users as a strategic game and propose an algorithm that can obtain the NE.

We define game $\mathfrak{g}_\mathfrak{m} = (\mathcal{K}_m, \prod_{i \in \mathcal{K}_m} y_i^m, \{U_i^o\}_{i \in \mathcal{K}_m})$, where $\mathcal{K}_m$ is the set of players in FN m, Y_i^m is the feasible strategy space of player i. In the game, the each MUE is one player and selecting the Fog mode or Cloud mode in order to maximize its own QoE (e.g,U_i^o) in response to the other MUEs' strategies which represent its utility achieved from offloading.

For a computation offloading strategy Y^m in FN m, define matrix $Y_{-i} = (y_1^m, y_2^m, \cdots, y_{i-1}^m, y_{i+1}^m, \cdots, y_{K_m}^m)$ as the offloading strategies of all MUEs except i. According to the definition of Game Theory, the best response strategy for each MUE can be expressed as

$$y_i^m* = \arg\max U^o(y_i^m, y_{-i}^m) \quad s.t.\ y_i^m = \{0, 1\}, \forall i \in \mathcal{K}_m \tag{39}$$

Algorithm 2 Geni-Coefficient Based Multi-MUEs Association Algorithm

Input:

The set of MUEs in each FN, $\{\mathcal{K}_m\}_{m\in\mathcal{M}}$;
The request set of MUEs $\mathcal{R} = \{R_1, R_2, \cdots, R_K\}, \forall i \in \mathcal{K}$;
The set of MUEs $\mathcal{K}$;
ρ_t, ρ_e, f_0, w_i

Output:

The matching policy $\{A_m^*\}_{\forall m\in\mathcal{M}}$;

Set *Conflict* = 1
for $m \to \mathcal{M}$ **do**
  Caculate the K_m^* and the corresponding income $\{\Gamma_{i,m,n}\}_{\forall i\in\mathcal{K}_m}$ according to (31)-(35);
  Update the MUE matching set $\mathcal{K}_m^*$ according to (36);
end for
while conflict **do**
  for $i \underset{M}{\to} K$ **do**
    if $\sum_{m=1}^{M} a_{i,m} > 1$ **then**
      Set conflict = 1
      Update the association policy for i;
      Reselect the abondened MUE according to the income order, update allociation set except FN m^*;
    else
      Set conflict = 0;
    end if
  end for
end while

Definition 1: An offloading strategy $y^*_{i\,\forall i\in\mathcal{K}}$ is an NE of game $\mathfrak{g}$ if no player can further to improve the QoE by unilaterally altering its strategy, i.e. for all MUEs in the game

$$U_i^o(y_i^m, Y_{-i}^m) \geq U_i^o(y_i^{m'}, Y_{-i}), \quad \forall i \in \mathcal{K}_m.$$

We next show that there exists an NE for game $\mathfrak{g}$ using the potential games.

Definition 2: A game is called a potential game if it exists a potential function Q^o such that for every player $\forall i \in \mathcal{K}$ and offloading vectors y_i^m and $y_i^{m'}$ for all MUEs

$$U_i^o(y_i^m, Y_{-i}^m) - U_i^o(y_i^{m'}, Y_{-i}) = Q_i^o(y_i^m, Y_{-i}^m) - Q_i^o(y_i^{m'}, Y_{-i}).$$

The Nash has self-stability properties which make the MUEs in the game derive mutually satisfactory solution at the equilibrium. At the equilibrium, no one can improve their utility by changing the offloading policy since the MUEs are selfish to act in their own interests in the non-cooperative offloading problem.

Proposition 1: The following function is a potential function and $\{\mathfrak{g}_m\}_{m\in\mathcal{M}}$ are potential games for all MUEs

$$Q^o(y_i^m, Y_{-i}^m) = a_{i,m}(1 - y_i^m)\left(\sum_{j\in\mathcal{K}_m, j\neq i} U_j^{fog} + U_i^{cloud}\right) + a_{i,m}y_i^m \sum_{i=1}^{K} U_i^{fog}, \forall i \in \mathcal{K}_m, \forall m \in \mathcal{M} \quad (40)$$

Proof: As for an MUE i associated with FN m, it can choose FN server ($y_i^m = 1$) or cloud server ($y_i^m = 1$) for task computing. By substituting the value of y_i^m into the Formula 39, we have

$$Q(1, Y_{-i}) = \sum_{i=1}^{K} a_{i,m} U_i^{fog} \quad (41)$$

$$Q(0, Y_{-i}) = a_{i,m} \sum_{j=1, j\neq i}^{K} U_j^{fog} + U_i^{cloud} \quad (42)$$

By subtracting the Formula (41) and (42), we can achieve that

$$Q(1, Y_{-i}) - Q(0, Y_{-i}) = a_{i,m}(U_i^{fog} - U_i^{cloud}) \quad (43)$$

According to the Formula (30), the subtraction of system utility obtained by the MUEs associated with m in Fog computing mode or cloud computing mode can be calculated by

$$U^o(1, Y_{-i}) - U^o(0, Y_{-i}) = a_{i,m}(U_i^{fog} - U_i^{cloud}) \quad (44)$$

By comparing the fomular (35) and (34), we have

$$U^o(1, Y_{-i}) - U^o(0, Y_{-i}) = Q(1, Y_{-i}) - Q(0, Y_{-i}) \quad (45)$$

According to the definition 2, the games $\{\mathfrak{g}_m\}_{m\in\mathcal{M}}$ are potential games. The proposition is concluded and there is at least one pure-strategy NE. ■

Next we will give the optimal solution to problem (30). For a certain offloading strategy $\{\overline{y_i^m}\}_{i\in\mathcal{K}}$, the objective function of (30) become function of $\{f_{i,m}\}_{i\in\mathcal{K}}$ and the objective function can be expressed as follows

$$U^o = \sum_{i\in\mathcal{K}}\sum_{m\in M} a_{i,m}\left[\overline{y_i^m} U_i^{fog}(f_{i,m}) + (1 - \overline{y_i^m})U_i^{cloud}\right] \quad (46)$$

Let $z_{i,m} = -u_{i,m}$, the problem of (30) can be rewritten by

$$\min_F Z(F) = \sum_{i=1}^{K} z_{i,m}$$
$$s.t. \quad C1' - C4' \quad (47)$$

The second derivative of U^o for $f_{i,m}$ can be calculated by

$$\frac{\partial^2 Z_{i,m}}{\partial^2 f_{i,m}} = \frac{2a_{i,m}y_i^m \rho^t \sigma_i}{(f_{i,m})^3} + 2a_{i,m}y_i^m \rho^e k \geq 0, \quad \forall i \in \mathcal{K}_m, \forall m \in \mathcal{M} \quad (48)$$

Derived from (48), Z is a convex function with respect to $\{f_{i,m}\}_{i\in\mathcal{K}_m, \forall m\in\mathcal{M}}$.

We can solve the optimization problem by applying the Karush-KuhnTucker (KKT) conditions. The Lagrangian function of problem of (48) can be expressed as

$$L = \sum_{i=1}^{K} Z_{i,m} + u\left(\sum_{i}^{K} f_{i,m} - f_0\right) \quad (49)$$

Algorithm 3 Distributed Task Offloading Algorithm

Input: $\{\mathcal{K}_m\}_{m\in\mathcal{M}}, N, u, g_{max}$
Output: ${\mathcal{F}_m}^*{}_{\forall m\in\mathcal{M}}$;

initialize u, δ g_{max},
for $m \in \mathcal{M}$ **do**
  Set $Y_o^m = (1, 1, \cdots, 1)$, $Y^m = (0, 0, \cdots, 0)$;
  while $Y_o^m \neq Y^m$ **do**
    $Y^m = Y_0^m$
    Set $y_i^m = 1$;
    if $g \leq= g_{max}$ **then**
      for $i \in \mathcal{K}_m$ **do**
        if $Y_0^m = \{1, 1, \cdots, 1\}$ **then**
          Compute the utility U_i^{cloud} according to (29);
        end if
        Calculate f_i^m and U_i^{fog} according to (48) and (26);
      end for
    end if
    Update u by (50)-(52) and g++;
    if $U_i^{fog} < U_i^{cloud}$ **then**
      Change the offloading policy of i, $y_i^m = 0$;
    else
      Change the offloading policy of i, $y_i^m = 1$;
    end if
    Update the offloading pllicy to Y_0^m;
  end while
end for
Output the optimized policy ${\mathcal{F}_m}^*{}_{\forall m\in\mathcal{M}}$

For $\forall i \in K$, the KKT conditions can be presented are as follows

$$\frac{\partial L}{\partial f_{i,m}} = -\frac{\overline{y_i^m}\rho^t\sigma_i}{(f_{i,m})^2} + 2\overline{y_i^m}\rho^e k f_{i,m} + u = 0 \quad (50)$$

$$u\left(\sum_i^K f_{i,m} - f_0\right) = 0 \quad (51)$$

Combine with the conditions (50) (51), the Lagrange multipliers update as below.

$$u(t+1) = \left[u(t) + \delta(t)\left(\sum_i^K f_{i,m} - f_0\right)\right]^+ \quad (52)$$

where t is the current times of iteration, $\delta(t)$ represents the step of t th iteration. By utilizing the KKT conditions, the optimal resource allocation solution can be found.

The solution Algorithm is shown in algorithm 3. In the algorithm 3, we initially set the offloading strategy $A_m = (1, 1, \cdots, 1)$ in each FN which means all MUEs are associated with Fog offloading mode for task processing.

V. SIMULATION RESULTS AND DISCUSSIONS

In this section, we use computer simulation software MATLAB to evaluate the performance of our algorithms.

TABLE 2. Summary of the simulation parameters.

Parameters	Values
Trasnmit power of MUEs (p_u)	0.1 mw
Trasnmit power of FNs (p_m)	0.2 mw
Revenue coefficient per unit of energy (ρ^e)	0.016 /J
Revenue coefficient per unit of saving delay (ρ^t)	0.5 /s
The number of tasks in the library (N)	100
The number of FNs (M)	5 - 30
The number of MUEs (K)	250
The maximum number of accessible MUEs in each FN (K_{max})	10

A. SIMULATION SETUP

The simulation parameters are described as follows. We consider the system consists of 250 UEs, 30 FNs. In the simulation, The FNs are located uniform distributed in a 600 × 600 m^2 area. The wireless bandwidth for each MUE in downlink and uplink is set to 1 MHz. According to the wireless channel model for cellular radio environment, we set the path loss factor $a = 3$. The background noise is $\sigma^2 = -100\ dbm$. For each MUE, we assume that the CPU clock speed is 900 MHz. Without loss of generality, we assume that computation resource of Fog server and the cloud server are equally set to 4 GHz [21]. The Data size of tasks S_i is randomly distributed between 10 MB and 30 MB. Meanwhile, the required computational resource for each task is randomly distributed between 1 Ghz and 6 Ghz The other main parameters in the simulations are summarized in table 2.

B. PERFORMANCE AND DISCUSSIONS

In the simulations, the utility of MUEs and system are all presented in units. The performance of task caching and task offloading are all disscussed.

1) PERFORMANCE OF TASK CACHING

To evaluate the impact of different parameters of task caching, we next implement the simulations with two parameters, the average required computational resource of tasks D_{aver}, the distribution parameter of MUEs λ.

We introduce three random-based caching strategies and two determined strategies:

- Popularity-based random caching strategie**(pop-CS)**, which means all MUEs random select tasks according to the popularity distribution of tasks(e.g., $\mathcal{Q} = \mathcal{P}$).

Uniform-based

- random caching strategie**(unif-CS)**, which means all MUEs random select tasks to cache according to an uniform probability distribution. In other words, each task have a equal probability to be cached on each MUE.
- Random-based caching strategie**(rand-CS)**, which means the MUEs random cache tasks in a random probability distribution. In this strategie, the probability of each task be cached on each MUE is random value.
- Greedy-based caching strategie**(greedy-cs)**, which means to optimaze the caching placement of MUEs throuth a amount of iterations. In each iteration the task

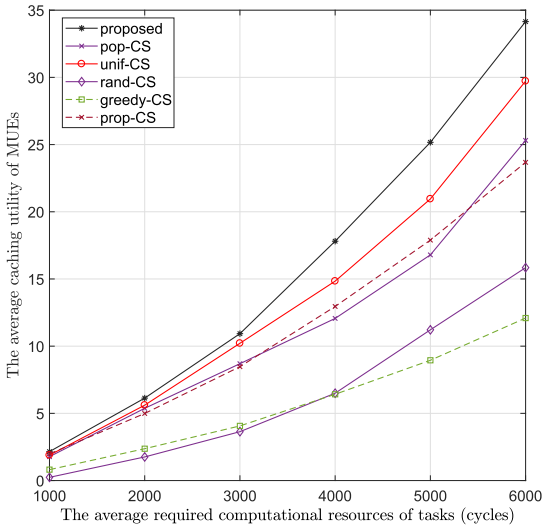


FIGURE 2. The average caching utility over different average required computational resource of tasks.

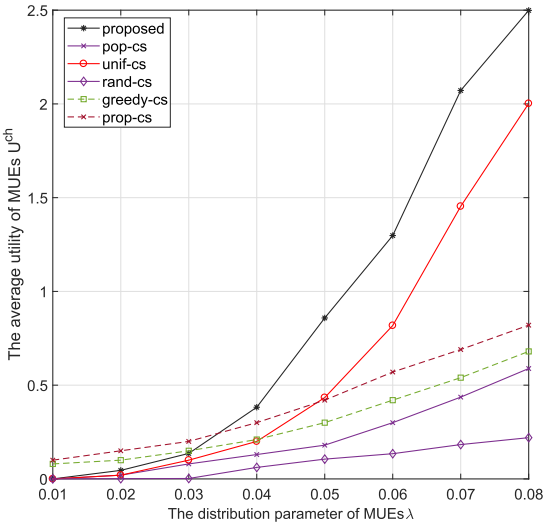


FIGURE 3. The average utility over differennt distrubution parameter λ.

which obtains the maximum utility will be determined to be cached in one of MUE. This strategie ensure to obtain a optimal solution in each iteration.

- proportional placement strategie(**prop-CS**). A certain proportion of MUEs will chose the most popular task to cache, the other MUEs will random select one of the left tasks in the task library to cache. The proportion is set to the optimal value.

We first present the utility of the proposed GA-based caching strategie with different values of parameter D_{aver}. In Fig.2, in order to show the influence of average required computational resource of tasks, we fixed the MUE distribution density to 0.1 and change D_{aver}. The average size of tasks is followed a uniform distribution between $10MB$ to $20MB$ and the popularity parameter β is set to 0.8. As shown is Fig.2, the average utility of MUEs increases with the increasing of D_{aver}. This is due to the fact that when D_{aver}increases, more computational resource will be needed for processing which result in increase of energy consumption and time delay of task computing in local computing mode while the cost of the task result delivery is not influenced as the energy consumption of computing is omitted. Thus average utility of MUEs increases. Morever, compared with some other caching schemes, the proposed GA-based caching strategie have best performance while the greedy-based caching strategie obtains the worst performance. We think it is becasue the determined caching strategie decrease the chance of D2D sharing compared with most random-based caching strategies.

From Fig. 3, we can see that average caching utility of MUEs increases with the MUEs distribution parameter λ. It is because the larger of distribution parameter of MUEs, the denser the MUEs are. Thus, it is more likely for MUEs to get the desire task data by D2D sharing from other MUEs nearby. To evaluate the caching strategy, we compare the proposed GA-based caching strategie in this paper with popularity-based caching strategie, the uniform probability caching algorithm and the random probability algorithm. The Fig. 3 shows that the performance of the proposed strategie is outstanding than other strategies.

2) EFFECT OF TASK OFFLOADING

The proposed task offloading and resource allocation algorithms in this paper are compared with three methods.

- The proposed **GAORA** strategy, which means the combination of Gini coefficient based MUE association policy, distribute task offloading and optimal resource allocation policy.
- **Cloud offloading** strategy, which stands for all MUEs who can access the network are associated with Cloud computing mode. In this strategy, each MUE will select nearest FN until the current accessable number of FNs is exceed to the maximum accessable number.
- **Fog offloading** strategy, which stands for all MUEs who can access the network are associated with Fog computing mode. The computational resource in each FNs is optimally allocated. In this strategy, each MUE will select nearest FN until the current accessable number of FNs is exceed to the maximum accessable number.
- **Greedy-based offloading** strategy denotes , optimaze the offloading policy and computational resource of MUEs throuth a large amount of iterations. In each iteration oplicy of the task offloading and resource allocation which obtains the maximum utility will be determined. This strategie ensure to obtain a current optimal solution in each iteration.
- **ϵ-Nash offlaoding** strategy denotes all the offloading MUEs chose a better offloading policy until the difference of changing offloading policy is small than the deviation ϵ [21]. In this strategy, each MUE will select nearest FN until the current accessable number of FNs is

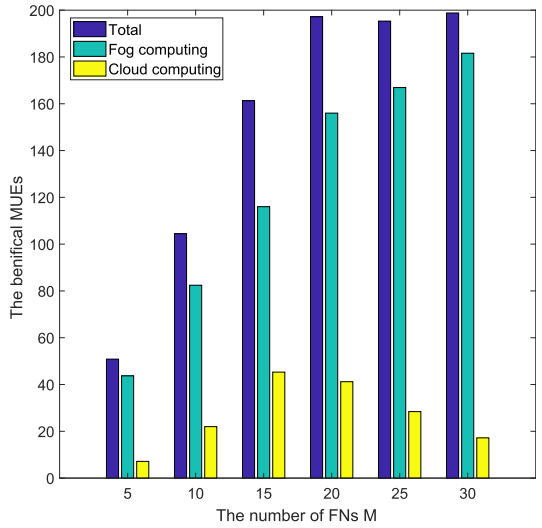


FIGURE 4. The beneficial number of MUEs over the mumber of FNs.

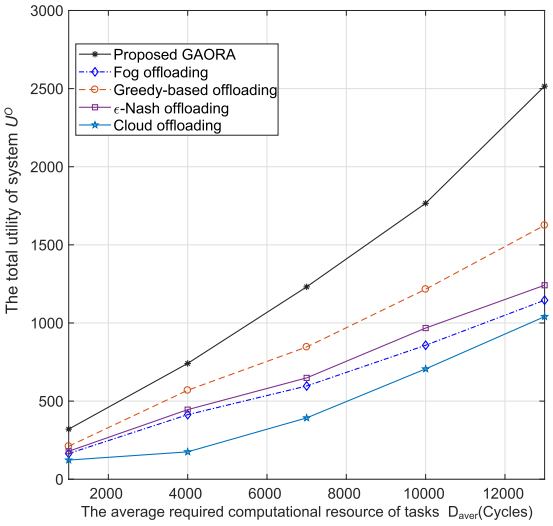


FIGURE 5. The total utility of system over the average required computational resource of tasks.

exceed to the maximum accessable number. Moreover, the computational resource in each FN is equally shared.

Fig.4 shows the number of beneficial users for different number of FNs with the scheme GAORA proposed in this paper. The default setting is $K = 250$, $D_{aver} = 6Ghz$, $S_{aver} = 20\ MB$. As can be observed from Fig.4, as the number of FNs increases, the number of beneficial MUEs increases. It is because the increasing number of FNs enables more MUEs to access the proper FNs for task offloading, and more MUEs have chances to enjoy sufficient computational resource in Fog servers or cloud server, which result in the increasing of their utility by decreasing of the computing delay. According to the Fig. 4, the beneficial number of MUEs in Cloud computing mode increases at the beginning but decreases later. We analyze the reason may be that as the number of FNs increase from 5 to 15, more MUEs will access the network through FNs. In this time, as the competition among MUEs are fierce, some MUEs have to associated with Cloud offloading mode to ensure the total utility are maximized. As the number of FNs increase from 15 to 30, there are more computational resource can be provided by FNs, so it is sufficient to make the MUEs in Fog computing mode obtain a higher utility compared offloading their tasks to Cloud server with long roundtrip delay and energy consumption, more MUEs will chose Fog computing mode to increase their utility, especially When the number of FNs increase to 30, all MUEs can access the network and the computational resource in FNs are reached to the peak value.

We now study the number of beneficial users that offload their computation tasks. In Fig. 5 - Fig. 7, the number of FNs is set to 5.

The influence of average required computational resource of tasks to the total utility is illustrated is Fig. 5. We fix the average data size of tasks to $10MB$. As we can see from Fig. 5, as the average required computational resource of tasks increase, the total utility in all schemes increase too. It is

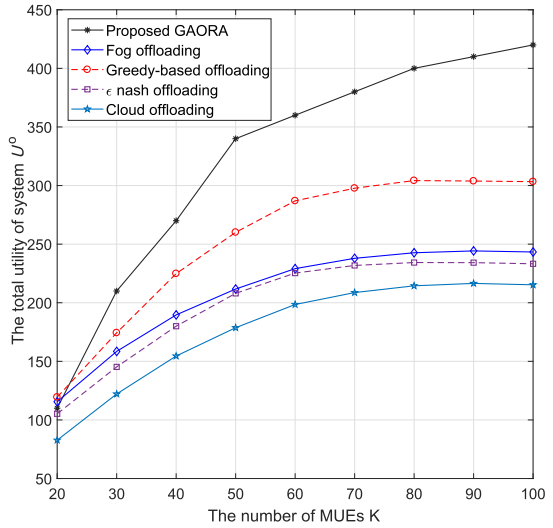


FIGURE 6. The total utility of system over the number of MUEs.

obviously that more required computational resource means more energy cost and larger delay for MUEs with local computing. Computing by Cloud and Fog have greater advantages versus Local computing in saving delay. By the way, although excess energy consumption is needed, but the total energy consumption of Fog or Cloud computing is not too difference compared with the local computing. It is because the CPU in Fog server and Cloud server aways have a the better energy effective switched capacitance compared with that in MUEs.

Fig. 6 illustrates the influence of the number of MUEs to the total utility. As shown in Fig. 6, the total utility have a tendency to rise with the increasing number of MUEs for all algorithms because computation resources can be used for task processing in FNs is more. Compare the proposed distributed GRORA scheme with other algorithms, the performance of GRORA have the best performance. The reason

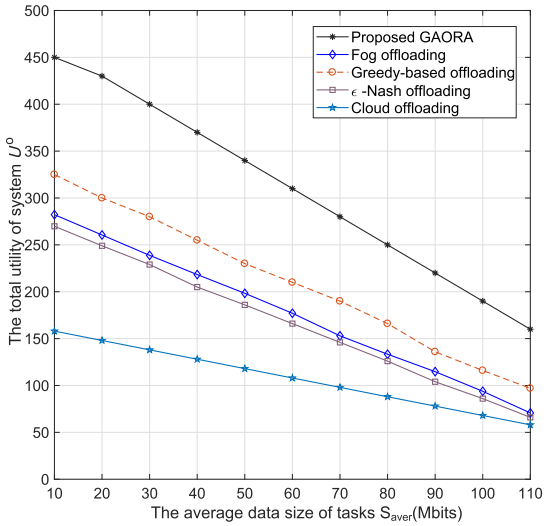


FIGURE 7. The total utility over average task size S_{aver}.

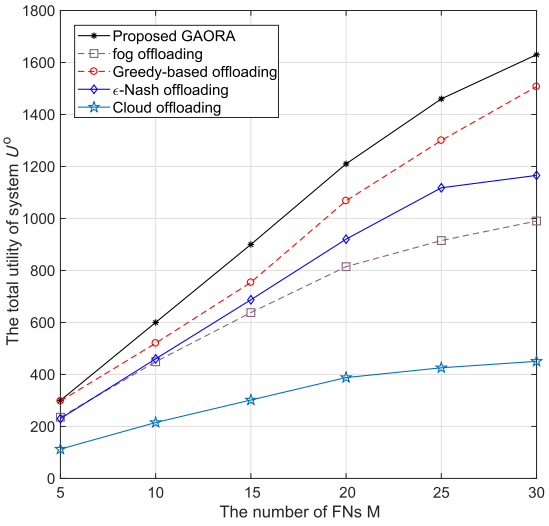


FIGURE 8. The total utility over the number of FNs.

is that the resource allocation in FNs is optimized in GRORA which can further improve the utility in each FN. According to the Fig.6, we can see that the scheme Cloud computing has the worst performance compared with other schemes. It is due to the long roundtrip of delay and energy consumption for Cloud offloading compared with Fog computing.

Figs. 7 illustrates the influence of the average of data size of tasks to the total utility. In this part, the average required computional resource is fixed to 2 *Ghz*, and the average data size of tasks in changed from 10 *MB* to 110 *MB*. As shown in Figs. 7, the total utility decrease with the increasing of data size for all algorithms. The reason is that as the data size of tasks increases, more transmit delay will be suffer which results in the decrease of total utility. Moreover, by the comparison of the proposed distributed GRORA algorithm with other algorithms, the proposed algorithm have the best performance. According to the Fig.6, we can see that the scheme Cloud offloading has the worst performance compared with other schemes.

We investigate the impact of number of FNs in Fig. 8. From Fig. 8, we observe that as the number of FNs increase, the total utility increase too. It is because With the increase number of FNs, more MUEs can be associated to their proper FNs with a good channel condition, moreover the available computing resources in FN server increase too which result in a smaller transmission delay and computational delay. Thus the total utility increase. By the comparison of the proposed distributed GAORA algorithm with other algorithms, the proposed algorithm has the best performance.

VI. CONCLUSION

In this paper, we investigated the allocation of resource in caching-enabled Fog computing system with multiple mobile user equipments (MUEs). We assumed MUEs should make the decision on task performing with a selection of three task processing modes including local mode, Fog offloading mode and Cloud offloading mode. Two scenarios were considered in this paper, task caching and its optimization in off-peak time, task offloading and its optimization in immediate time. For the first scenario, to maximize the average of utility, a task caching optimization problem was formulated with stochastic theory and was solved by a GA-based task caching algorithm. For the second scenario, to maximize the total utility, the task offloading and resource optimization problem was formulated as a mixed integer nonlinear programming (MINLP) which jointly considers the MUE allocation policy, task offloading policy, the computational resource allocation policy. We transform it into multi-MUEs association problem (MMAP) and mixed Fog-Cloud offloading and optimization problem (MFCOOP) and solved them by a Gini coefficient-based MUEs allocation algorithm and a distributed algorithm based on Lagrange multiplier respectively. At last, simulations show the effectiveness of the proposed scheme with the comparison of other baseline schemes.

REFERENCES

[1] M. Peng, Y. Sun, X. Li, Z. Mao, and C. Wang, "Recent advances in cloud radio access networks: System architectures, key techniques, and open issues," *IEEE Commun. Surveys Tuts.*, vol. 18, no. 3, pp. 2282–2308, 3rd Quart., 2016.

[2] M. Peng, C. Wang, J. Li, H. Xiang, and V. Lau, "Recent advances in underlay heterogeneous networks: Interference control, resource allocation, and self-organization," *IEEE Commun. Surveys Tuts.*, vol. 17, no. 2, pp. 700–729, 2nd Quart., 2015.

[3] E. Bastug, M. Bennis, M. Medard, and M. Debbah, "Toward interconnected virtual reality: Opportunities, challenges, and enablers," *IEEE Commun. Mag.*, vol. 55, no. 6, pp. 110–117, Jun. 2017.

[4] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, "Fog computing and its role in the Internet of Things," in *Proc. 1st Ed. MCC Workshop Mobile Cloud Comput. (MCC)*, Helsinki, Finland, Feb. 2012, pp. 13–16.

[5] L. M. Vaquero and L. Rodero-Merino, "Finding your way in the fog: Towards a comprehensive definition of fog computing," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 44, no. 5, pp. 27–32, Oct. 2014.

[6] M. Peng and K. Zhang, "Recent advances in fog radio access networks: Performance analysis and radio resource allocation," *IEEE Access*, vol. 4, pp. 5003–5009, 2016.

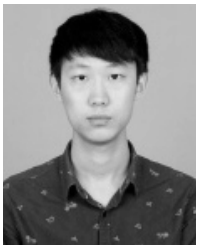
[7] F. Jalali, K. Hinton, R. Ayre, T. Alpcan, and R. S. Tucker, "Fog computing may help to save energy in cloud computing," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 5, pp. 1728–1739, May 2016.
[8] R. Deng, R. Lu, C. Lai, and T. H. Luan, "Towards power consumption-delay tradeoff by workload allocation in cloud-fog computing," in *Proc. IEEE Int. Conf. Commun. (ICC)*, London, U.K.:, Jun. 2015, pp. 3909–3914.
[9] H. Xiang, W. Zhou, M. Daneshmand, and M. Peng, "Network slicing in fog radio access networks: Issues and challenges," *IEEE Commun. Mag.*, vol. 55, no. 12, pp. 110–116, Dec. 2017.
[10] B. Zhou, A. V. Dastjerdi, R. N. Calheiros, S. N. Srirama, and R. Buyya, "mCloud: A context-aware offloading framework for heterogeneous mobile cloud," *IEEE Trans. Services Comput.*, vol. 10, no. 5, pp. 797–810, Sep./Oct. 2017.
[11] C. Mouradian, D. Naboulsi, S. Yangui, R. H. Glitho, M. J. Morrow, and P. A. Polakos, "A comprehensive survey on fog computing: State-of-the-art and research challenges," *IEEE Commun. Surveys Tuts.*, vol. 20, no. 1, pp. 416–464, 1st Quart., 2018.
[12] Y. Niu, Y. Liu, Y. Li, Z. Zhong, B. Ai, and P. Hui, "Mobility-aware caching scheduling for fog computing in mmWave band," *IEEE Access*, vol. 6, pp. 69358–69370, 2018.
[13] S. M. Azimi, O. Simeone, A. Sengupta, and R. Tandon, "Online edge caching and wireless delivery in fog-aided networks with dynamic content popularity," *IEEE J. Sel. Areas Commun.*, vol. 36, no. 6, pp. 1189–1202, Jun. 2018.
[14] Z. Su, Q. Xu, J. Luo, H. Pu, Y. Peng, and R. Lu, "A secure content caching scheme for disaster backup in fog computing enabled mobile social networks," *IEEE Trans. Ind. Informat.*, vol. 14, no. 10, pp. 4579–4589, Oct. 2018.
[15] F. Xu and M. Tao, "Fundamental limits of decentralized caching in Fog-RANs with wireless fronthaul," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)* Vail, CO, USA, Jun. 2018, pp. 1430–1434.
[16] Y. Wei, F. R. Yu, M. Song, and Z. Han, "Joint optimization of caching, computing, and radio resources for fog-enabled IoT using natural actor-critic deep reinforcement learning," *IEEE Internet Things J.* vol. 6, no. 2, pp. 2061–2073, Apr. 2019.
[17] R. Karasik, O. Simeone, and S. Shamai, "Fundamental latency limits for D2D-Aided content delivery in fog wireless networks," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Vail, CO, USA, Jun. 2018, pp. 2461–2465.
[18] S. Yan, M. Peng, M. A. Abana, and W. Wang, "An evolutionary game for user access mode selection in fog radio access networks," *IEEE Access*, vol. 5, pp. 2200–2210, 2017.
[19] S. Jošilo and G. Dán, "Decentralized algorithm for randomized task allocation in fog computing systems," *IEEE/ACM Trans. Netw.*, vol. 27, no. 1, pp. 85–97, Feb. 2019.
[20] Z. Wei and H. Jiang, "Optimal offloading in fog computing systems with non-orthogonal multiple access," *IEEE Access*, vol. 6, pp. 49767–49778, 2018.
[21] H. Shah-Mansouri and V. W. S. Wong, "Hierarchical fog-cloud computing for IoT systems: A computation offloading game," *IEEE Internet Things J.*, vol. 5, no. 4, pp. 3246–3257, Aug. 2018.
[22] Y. Jiang and D. H. K. Tsang, "Delay-aware task offloading in shared fog networks," *IEEE Internet Things J.*, vol. 5, no. 6, pp. 4945–4956, Dec. 2018.
[23] Y. Yang, K. Wang, G. Zhang, X. Chen, X. Luo, and M. Zhou, "MEETS: Maximal energy efficient task scheduling in homogeneous fog networks," *IEEE Internet Things J.*, vol. 5, no. 5, pp. 4076–4087, Oct. 2018.
[24] Y. Liu, F. R. Yu, X. Li, H. Ji, and V. C. M. Leung, "Distributed resource allocation and computation offloading in fog and cloud networks with non-orthogonal multiple access," *IEEE Trans. Veh. Technol.*, vol. 67, no. 12, pp. 12137–12151, Dec. 2018.
[25] L. Liu, Z. Chang, X. Guo, S. Mao, and T. Ristaniemi, "Multiobjective optimization for computation offloading in fog computing," *IEEE Internet Things J.*, vol. 5, no. 1, pp. 283–294, Feb. 2018.
[26] J. Du, L. Zhao, J. Feng, and X. Chu, "Computation offloading and resource allocation in mixed fog/cloud computing systems with min-max fairness guarantee," *IEEE Trans. Commun.*, vol. 66, no. 4, pp. 1594–1608, Apr. 2018.
[27] R. Deng, R. Lu, C. Lai, T. H. Luan, and H. Liang, "Optimal workload allocation in fog-cloud computing toward balanced delay and power consumption," *IEEE Internet Things J.*, vol. 3, no. 6, pp. 1171–1181, Dec. 2016.
[28] L. Liu, Z. Chang, and X. Guo, "Socially aware dynamic computation offloading scheme for fog computing system with energy harvesting devices," *IEEE Internet Things J.*, vol. 5, no. 3, pp. 1869–1879, Jun. 2018.
[29] Y. Sun, M. Peng, S. Mao, and S. Yan, "Hierarchical radio resource allocation for network slicing in fog radio access networks," *IEEE Trans. Veh. Technol.*, vol. 68, no. 4, pp. 3866–3881, Apr. 2019.
[30] G. M. S. Rahman, M. Peng, K. Zhang, and S. Chen, "Radio resource allocation for achieving ultra-low latency in fog radio access networks," *IEEE Access*, vol. 6, pp. 17442–17454, 2018.
[31] G. Li, J. Wu, J. Li, K. Wang, and T. Ye, "Service popularity-based smart resources partitioning for fog computing-enabled industrial Internet of things," *IEEE Trans. Ind. Informat.*, vol. 14, no. 10, pp. 4702–4711, Oct. 2018.
[32] H. Zhang, Y. Xiao, S. Bu, D. Niyato, F. R. Yu, and Z. Han, "Computing resource allocation in three-tier IoT fog networks: A joint optimization approach combining Stackelberg game and matching," *IEEE Internet Things J.*, vol. 4, no. 5, pp. 1204–1215, Oct. 2017.
[33] X. Wang, Z. Ning, and L. Wang, "Offloading in Internet of vehicles: A fog-enabled real-time traffic management system," *IEEE Trans. Ind. Informat.*, vol. 14, no. 10, pp. 4568–4578, Oct. 2018.
[34] Y. Sun, M. Peng, and S. Mao, "Deep reinforcement learning-based mode selection and resource management for green fog radio access networks," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 1960–1971, Apr. 2019.
[35] Y. Hao, M. Chen, L. Hu, M. S. Hossain, and A. Ghoneim, "Energy efficient task caching and offloading for mobile edge computing," *IEEE Access*, vol. 6, pp. 11365–11373, 2018.
[36] X. Lin, J. Andrews, A. Ghosh, and R. Ratasuk, "An overview of 3GPP device-to-device proximity services," *IEEE Commun. Mag.*, vol. 52, no. 4, pp. 40–48, Apr. 2014.
[37] Y. Long, D. Wu, Y. Cai, and J. Qu, "Joint cache policy and transmit power for cache-enabled D2D networks," *IET Commun.*, vol. 11, no. 16, pp. 2498–2506, 2017.
[38] J. Zhang, W. Xia, F. Yan, and L. Shen, "Joint computation offloading and resource allocation optimization in heterogeneous networks with mobile edge computing," *IEEE Access*, vol. 6, pp. 19324–19337, 2018.
[39] P. Zhao, H. Tian, C. Qin, and G. Nie, "Energy-saving offloading by jointly allocating radio and computational resources for mobile edge computing," *IEEE Access*, vol. 5, pp. 11255–11268, 2017.

YANWEN LAN received the B.S. degree in communications engineering and the M.E. degree in electrical and communications engineering from Henan University, in 2013 and 2016, respectively. He is currently pursuing the M.S. degree with the Beijing University of Posts and Telecommunications (BUPT). His research interests include fog computing, mobile edge computing, and cache-enabled heterogeneous networks.

XIAOXIANG WANG received the B.S. degree in physics from Qufu Normal University, Qufu, China, in 1991, the M.S. degree in information engineering from East China Normal University, Shanghai, China, in 1994, and the Ph.D. degree in electronic engineering from the Beijing Institute of Technology, Beijing, China, in 1998. In 1998, she joined the School of Information and Communication Engineering, Beijing University of Posts and Telecommunications. From August 2010 to February 2011, she was a Visiting Fellow with the Department of Electrical and Computer Engineering, North Carolina State University, Raleigh. Her research interests include communications theory and signal processing, with specific interests in cooperative communications, multiple-input-multiple-output systems, multimedia broadcast/multicast service systems, and resource allocation.

DONGYU WANG received the B.S. and M.S. degrees from Tianjin Polytechnic University, China, in 2008 and 2011, respectively, and the Ph.D. degree from the Beijing University of Posts and Telecommunications, China, in 2014. From September 2014 to June 2016, he held a post Ph.D. position with the Department of Biomedical Engineering, Chinese PLA General Hospital, Beijing.

In September 2016, he joined the School of Information and Communication Engineering, Beijing University of Posts and Telecommunications. His research interests include device-to-device communication, multimedia broadcast/multicast service systems, resource allocation, theory, and signal processing, especially cooperative communications and mobile edge computing.



ZHAOLIN LIU received the B.S. degree in communications engineering from the University of Electronic Science and Technology of China, in 2017. He is currently pursuing the M.E. degree with the Beijing University of Posts and Telecommunications (BUPT). His research interests include about edge computing, including mobile edge computing and fog computing.

YIBO ZHANG received the B.S. degree from Henan University, Kaifeng, China, in 2013, and the M.S. degree from Henan Normal University, Xinxiang, China, in 2015. He is currently pursuing the Ph.D. degree with the Key Laboratory of Universal Wireless Communications, Ministry of Education, Beijing University of Posts and Telecommunications, Beijing, China. His research interests include nonorthogonal multiple access, wireless multicast, and cooperative communication.