# Mean Field Graph Based D2D Collaboration and Offloading Pricing in Mobile Edge Computing

Xiong Wang, *Member, IEEE, ACM*, Jiancheng Ye, *Senior Member, IEEE, Member, ACM*, and John C. S. Lui, *Fellow, IEEE, ACM*

***Abstract*— Mobile edge computing (MEC) facilitates computation offloading to edge server and task processing via device-to-device (D2D) collaboration. Existing works mainly focus on centralized network-assisted offloading solutions, which are unscalable to collaborations among massive users. In this paper, we propose a *joint framework* of decentralized D2D collaboration and task offloading for MEC systems with large populations. Specifically, we utilize the power of two choices for D2D collaboration, which enables users to assist each other in a decentralized manner. Due to short-range D2D communication and user movements, we formulate a *mean field model* on a finite-degree and dynamic graph to analyze the collaboration state evolution. We derive the existence, uniqueness and convergence of the state stationary point to provide a *tractable collaboration performance*. Complementing this D2D collaboration, we further build a *Stackelberg game* to model users' task offloading, where the provider, managing many servers, is the leader to determine service prices, while users are followers to make offloading decisions. By embedding Stackelberg game into Lyapunov optimization, we develop an *online* offloading and pricing scheme, which can optimize servers' service utility or fairness, and users' system cost simultaneously. Extensive evaluations show that D2D collaboration can mitigate users' workloads by 73.8% and fair pricing can promote servers' utility fairness by 15.87%.**

***Index Terms*— Mobile edge computing, decentralized D2D collaboration, mean field graph, task offloading, dynamic pricing.**

## I. Introduction

IN RECENT years, we have witnessed a rapid growth of data generated from the network edge, especially with the enormous popularity of mobile devices [1]. Many intelligent mobile applications, such as interactive gaming, face recognition and natural language processing, are emerging which typically demand intensive computation and low latency. In general, mobile devices have constrained resources, while remote-resided cloud server suffers from high latency due to long-haul transmissions. To support the compute-intensive yet delay-sensitive applications, mobile edge computing (MEC) is recognized as a new paradigm to push cloud frontier close to the edge for such service requirements [2].

At a high level, MEC enables mobile users to offload tasks to edge servers endowed with computing functionalities. Compared to the cloud datacenter, deployed edge servers basically have limited computing capacity, making them difficult to accommodate huge amount of tasks since over 90% of the data will be stored and processed at the network edge [3]. Under this scenario, exploiting *collaboration among users* is a promising complementary approach to task offloading for easing the strain on servers [4]. Device-to-device (D2D) communication (e.g., via Bluetooth, Wi-Fi Direct or LTE-Direct) is more energy-saving while less time-consuming [5], thereby providing a low-latency service for users when they collaboratively process tasks via D2D links [6]. Specifically, heavily-loaded users can seek immediate assistance from lightly-loaded ones within proximity, and hence the average task delay is expected to decrease significantly.

Despite the advantage of D2D collaboration, task offloading to edge servers is still indispensable for MEC as high latency is witnessed if compute-intensive tasks are handled solely by resource-constrained mobile devices. Along with the offloading, mobile users are charged service prices by the provider who manages edge servers for providing computing service [2], [7], [8]. Needless to say, setting proper prices is critical, as excessively low prices are insufficient to compensate for the servers' operation costs, whereas unduly high prices will cause a decrease in user demands of task offloading and further increase the delay in task execution. Therefore, a reasonable pricing scheme is required to incentivize task offloading while also bringing benefits to the service provider.

Various efforts have been dedicated to investigate D2D collaboration in MEC [4], [6], [9]. Benefiting from users' mutual assistance, D2D collaboration can improve the system's energy efficiency and delay performance. In these works, edge servers mainly serve as the central coordinator to aid the collaborative task processing, whereas the potential benefit of task offloading was not explored. Few following works further

Manuscript received 12 November 2022; revised 25 April 2023; accepted 12 June 2023; approved by IEEE/ACM TRANSACTIONS ON NETWORKING Editor R. Vaze. Date of publication 14 July 2023; date of current version 16 February 2024. This work was supported in part by NSFC under Grant 62202185, in part by the Hubei Provincial Natural Science Foundation of China under Grant 2022CFB611, in part by the Research Grants Council (RGC) under Grant GRF 14215722, and in part by The Chinese University of Hong Kong (CUHK) under Grant 6905407. An earlier version of this paper was presented in the proceedings of IEEE/ACM IWQoS 2021 [DOI: 10.1109/IWQOS52092.2021.9521271]. *(Corresponding author: Jiancheng Ye.)*

Xiong Wang is with the National Engineering Research Center for Big Data Technology and System, Services Computing Technology and System Laboratory/Cluster and Grid Computing Laboratory, School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan 430074, China (e-mail: xiongwang@hust.edu.cn).

Jiancheng Ye is with the Network Technology Laboratory and the Hong Kong Research Center, Huawei, Hong Kong (e-mail: yejiancheng@huawei.com).

John C. S. Lui is with the Department of Computer Science and Engineering, The Chinese University of Hong Kong, Hong Kong (e-mail: cslui@cse.cuhk.edu.hk).

This article has supplementary downloadable material available at https://doi.org/10.1109/TNET.2023.3288558, provided by the authors.

Digital Object Identifier 10.1109/TNET.2023.3288558

incorporate D2D collaboration into task offloading to take advantage of the computing capacity in both mobile devices and edge servers [10], [11]. However, these works mostly concentrate on the centralized offloading and/or collaboration, with a restrictive assumption of time-invariant D2D links in order to achieve a tractable analysis. When there are a large population of moving mobile users, which is often the case in D2D collaboration, *how to characterize a decentralized collaboration and develop an efficient offloading for "dynamic" MEC system* still remains unresolved. To answer this critical question, researchers are faced with the following challenges.

First, due to short-range D2D communication, collaboration is mainly among nearby users, thus leading to a graph structure formed by spatially distributed mobile users. A decentralized collaboration scheme should encompass both *static and dynamic connectivity setting* when considering user movements, which however is theoretically challenging. It brings new modeling requirements for D2D collaboration to achieve rigorous theoretical guarantee as well as good empirical performance. Second, task offloading is influenced by service prices set by the provider, whereas pricing scheme also depends on the strategic offloading behaviors of mobile users. Their *mutual-dependency* raises difficulty in the optimal offloading and pricing design. Third, service utility of different edge servers is unbalanced, making it essential to maintain their *utility fairness* rather than maximizing the utilitarian profit only. Fourth, D2D collaboration is *intertwined* with task offloading due to various task executions including collaborative execution and offloaded execution. This demands incorporating decentralized collaboration into determining appropriate proportions of tasks to be offloaded and to be processed locally so as to reduce the execution latency while enhancing energy efficiency.

In this paper, we propose a *joint D2D collaboration and offloading pricing* for MEC systems with massive users. We first utilize the power of two (Po2) choices for *decentralized collaboration* among mobile users, where each user randomly polls a neighbor within its D2D range and forwards a task if the polled neighbor has a lighter workload. We develop a novel *mean field model on graph* to analyze this D2D collaboration, through which we can characterize the state evolution of MEC system in both static and dynamic situations. By incorporating the steady state of D2D collaboration, we further formulate a Stackelberg game to model users' task offloading. Specifically, users are followers in making their offloading decisions, while the provider is the leader in determining dynamic prices based on *Lyapunov optimization* for providing computing service. In particular, the provider aims to optimize both the utilitarian service utility, and the egalitarian proportional fairness to achieve a fair utility distribution. As a result, we consider the intertwined collaboration and offloading processes in order to collectively enable an efficient task execution. This paper has the following main contributions:

- We develop a joint D2D collaboration and task offloading framework which facilitates users to collaboratively process tasks in a *decentralized manner* and offload computation to local edge servers. Our framework targets the real-life large-scale MEC system so as to fully unleash the potentials of widely-distributed mobile devices and edge servers' capacity, thereby mitigating the execution latency meanwhile improving the energy efficiency.
- We propose a novel mean field model on static and dynamic graphs to characterize D2D collaboration, so that we can analyze the stochastic state evolution by *deterministic ordinary differential equations* (ODEs). We rigorously prove the existence and uniqueness of mean field stationary point to provide a *theoretically tractable performance* for D2D collaboration. To the best of our knowledge, this is the first work that conducts a thorough analysis of mean field model on *finite-degree and dynamic graphs*.
- We design an *online offloading and pricing scheme* using a Lyapunov optimization framework to determine the optimal offloading and pricing decisions over time. By embedding a Stackelberg game into the online decision making, we can minimize users' system cost under their stringent task delay requirements, while maximizing servers' *long-term* utilities and fairness simultaneously.
- We carry out extensive evaluations for the D2D collaboration and offloading pricing. Results show that we can mitigate users' workloads by $73.8\%$, and improve servers' fairness by $15.87\%$ with only $2.55\%$ utility reduction.

## II. Related Work

In this paper, we explore the D2D collaboration and task offloading in MEC systems with many users. Next, we briefly survey the related works.

**Collaborative MEC**. Emerging MEC offers new possibility for intelligent mobile applications [1]. As single edge server has limited computing capacity, collaborative MEC is effective to accommodate more computation [12]. Li et al. propose an online learning aided collaborative mechanism allowing edge servers to offload tasks to each other considering trust, delay and multi-hop transmission [13]. These works mainly focus on the collaboration among edge servers, instead of, among mobile users. To explore how users can help each other, Pu et al. study an incentive-aware task offloading among users via D2D links [6], but offloading to edge servers is not considered. He et al. further incorporate D2D collaboration and task offloading to edge servers for enhancing the computation capacity [11]. However, existing works rarely comprehensively investigate D2D collaboration and task offloading as a whole. Besides, they often concentrate on the centralized collaboration for finite, often small-scale, users which makes their methods hard to be extended to MEC systems with massive users.

**Mean field model**. Mean field model is a promising method to characterize complex interactions in large-scale systems [14], which studies the asymptotic system performance by representing the state transition using an ODE, and then shows the system convergence to ODE when agent number increases largely. Narasimha et al. apply mean field model to design distributed MAC protocols in ultra-dense multichannel wireless networks [15], and Li et al. explore the mean field in content streaming among co-located peer devices

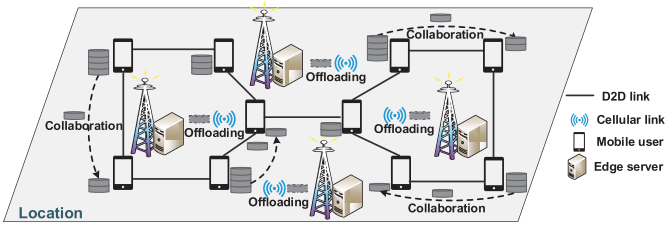


Fig. 1. Snapshot of the MEC system.

for enhancing streaming quality of service [16]. A multi-user task offloading scheme to edge servers is proposed in [17], where authors use a non-cooperative mean field game to minimize the computation cost. Mitzenmacher uses mean field to analyze the power of $d$ choices in randomized load balancing, where $d$ queues are randomly sampled and a task will join the shortest queue [18]. Results show that even $d$ is a small value, such as $d = 2$, the average sojourn time still decreases dramatically. Later on, Gast investigates the power of two choices on finite-degree graphs, but only simulation results are provided [19]. Budhiraja et al. show that the power of $d$ choices on graphs can still be analyzed via mean field model, as long as each vertex has infinite degree [20]. Graphon mean field studies the game on graph structure, especially exploring the reaction-diffusion models on dense graphs [21]. Nevertheless, for D2D collaboration, the degree of a user (vertex) will not scale with the total number of users due to short-range D2D communication, i.e., users actually have finitely many neighbors. Another strand of researches using mean field model on graphs focus on epidemic processes in networks [22]. These works usually assume an uncorrelated network, and use an ODE system to represent the state evolution, whereas the convergence to mean field model is often not proved [23]. All existing literatures mainly consider static graph structures when analyzing mean field model, while the dynamic case is often ignored. Therefore, previous studies have not yet rigorously explored the mean field model on finite-degree or dynamic graphs.

**Service pricing**. Pricing scheme design is important to a service provider when providing specific service for end users [24]. Regarding MEC, Zhao et al. propose a pricing scheme to charge mobile users when they offload computation via access points [7]. A Stackelberg game is built to make decisions for both edge servers (leaders) and mobile users (followers) in [25]. However, these works only consider the case of finitely many users, and often adopt heterogeneous prices among users which makes them too complicated to be implemented in real-life applications.

## III. System Model

We consider a MEC system with a *large population* of users $\mathcal{N} = \{1, 2, \ldots, N\}$, and a set of edge servers $\mathcal{M} = \{1, 2, \ldots, M\}$ managed by a provider [2]. As shown in Fig. 1, users' generated tasks can be offloaded to servers via cellular network, or processed by their collaboration via D2D link

### A. System Overview

Due to short-range D2D communication, collaboration is mainly among users within proximity, and we model this collaboration network as a connected graph $\mathcal{G} = \{\mathcal{N}, \mathcal{E}\}$ with $\mathcal{E}$ denoting the D2D links. The graph $\mathcal{G}$ will change dynamically when users move around, which is to be analyzed together with the static case later. Considering the uneven distribution of moving users within the radio coverage of different edge servers, the empirical frequency or probability of users' offloaded tasks processed by server $j$ is denoted as $f_j$ where $\sum_{j=1}^{M} f_j = 1$. Along with task offloading, a service price $p_j$ is charged by the provider for serving the offloading request at edge server $j$, with $p_j$ often *remaining fixed* for a long period, e.g., weekly or monthly basis [24]. In return, users strategically choose to offload tasks with probability $x \in [0, 1]$, while handling the rest via D2D collaboration.

A task is typically characterized by the amount of required computation (CPU cycles) and the input data size [27]. By convention, task generation of each user follows a rate-$\lambda$ Poisson process, where the computation amount of a task obeys an exponential distribution with normalized unit mean value, and the data size follows a potential distribution with an average value of $B$ [28]. The normalized computation service rates of mobile devices and edge server $j$ are $\mu$ and $\gamma_j$, respectively, with $\lambda < \mu$ to keep the MEC system stable, i.e., users are able to handle their tasks alone while offloading to servers will shorten the latency. Moreover, edge servers can process the offloaded tasks in parallel because they have more powerful computing capabilities than mobile devices [29].

### B. D2D Collaboration

We use the Po2 choices for decentralized collaboration [18]. Let $Q_u(t)$ be the number of tasks, or workload, of user $u \in \mathcal{N}$ at time $t$. For Po2, when a task is generated by $u$ and not offloaded, $u$ randomly polls a neighbor, say $v$, and forwards the task to $v$ if $Q_u(t) > Q_v(t)$; otherwise the task joins $Q_u(t)$ with ties being broken arbitrarily. Denote $d_u$ as the number of $u$'s neighbors, also known as its degree in graph $\mathcal{G}$. The degree $d_u$ is distributed in a *finite degree set* $\mathcal{K} = \{k_{\min}, \ldots, k_{\max}\}$ due to the short-range D2D communication. In particular, the D2D network specified by graph $\mathcal{G}$ is considered to be *uncorrelated* [22], i.e., the probability $\Pr(k'|k)$ that a user with degree $k$ has a link to a neighbor with degree $k'$ satisfies:

$$\Pr(k'|k) = \frac{k' \Pr(k')}{\overline{k}}, \tag{1}$$

where $\Pr(k)$ is the probability that a user has degree $k$ and $\overline{k} = \sum_{k \in \mathcal{K}} \Pr(k) k$ is the expected degree. Eq. (1) means the degree distribution of neighboring users is independent.

For $k \in \mathcal{K}, i \geq 0$, we define $q_{k,i}(t)$ and $s_{k,i}(t)$ as:

$$q_{k,i}(t) = \frac{\sum_{u \in \mathcal{N}} \mathbf{1}_{d_u = k, Q_u(t) = i}}{\sum_{u \in \mathcal{N}} \mathbf{1}_{d_u = k}}, \tag{2}$$

$$s_{k,i}(t) = \sum\nolimits_{j \geq i} q_{k,j}(t). \tag{3}$$

Physically, $q_{k,i}(t)$ or $s_{k,i}(t)$ can be regarded as the probability that a user with degree $k$ holds $i$ or at least $i$ tasks, respectively.

Besides, $q_{k,i}(t) = s_{k,i}(t) - s_{k,i+1}(t)$ and $s_{k,0}(t) = 1, \forall k \in \mathcal{K}$. We say $s_{k,1}(t)$ is the *busy probability* as it implies the case where a $k$-degree user has a non-empty workload. Denote $\boldsymbol{s}(t) = \{s_{k,i}(t)\}$ as the system state, that is the workload distribution of the MEC system which also indicates the probability of how many tasks a mobile user locally maintains. This way, the state evolution of $\boldsymbol{s}(t)$ represents the changes in workload distribution during users' D2D collaboration. Hereinafter, we will use $q_{k,i}, s_{k,i}, Q_u$ and $\boldsymbol{s}$ without index $t$ if there is no confusion.

Our objective is to illustrate that D2D collaboration can effectively mitigate users' workloads, by characterizing the stationary point $\boldsymbol{s}^*$ for state evolution when the user number $N \to \infty$ and $\dot{\boldsymbol{s}}^* = \mathbf{0}$ (MEC system is stable). Given the graph collaboration structure, we devise a new analysis framework as existing Po2 methods are no longer applicable.

### C. Task Offloading

Upon generating a task, users offload it to edge servers with probability $x$, and are charged with a price $p_j$ if processed by server $j$. Since each $p_j$ is fixed for a long time, the offloading and pricing decisions are made in terms of discrete time slot $\{0, 1, \ldots, n, \ldots\}$, say at a time interval of every week. The probability $x[n]$ and price $p_j[n]$ are constant in each slot, and the system is regarded stable. Hence, we can leverage the *stationary point* $\boldsymbol{s}^*$ in offloading and pricing schemes design.

*1) Offloading Constraint:* Together with D2D collaboration, users also collaboratively decide the offloading probability $x[n]$ in each time slot, which in fact only relies on $\boldsymbol{s}^*$.

**Task delay**. If a task is offloaded to edge servers, the expected task delay includes the transmission time and server processing time, that is $d_o = \sum_{j=1}^{M} x[n] f_j (\frac{B}{r_j} + \frac{1}{\gamma_j})$ where $r_j$ denotes the data rate of cellular link to server $j$ and 1 is the normalized unit computation required by a task. With probability $x_c[n] = 1 - x[n]$, a task will be processed via D2D collaboration, then the delay amounts to the queueing time, which is $d_q = x_c[n] \frac{\sum_i \sum_k \Pr(k) s^*_{k,i}}{x_c[n]\lambda} = \frac{\sum_i \sum_k \Pr(k) s^*_{k,i}}{\lambda}$ based on Little's law. Here, the transmission delay of fast and short-range D2D communication is negligibly small compared to $d_o$ and $d_q$ [5]. Therefore, the average task delay $d(x[n])$ is:

$$d(x[n]) = d_o + d_q. \tag{4}$$

Note that $s^*_{k,i}$ depends on $x_c[n]$, or the probability $x[n]$.

**Collaboration fairness**. Due to heterogeneous number of neighbors, users have *unbalanced contributions* in D2D collaboration, i.e., busy probability $s^*_{k,1}$ varies over degree $k$. When deciding the probability $x[n]$, collaboration fairness requires that the gap between the highest and lowest $s^*_{k,1}$ should not be too large to prevent the "free-riding" scenario.

*2) Pricing Constraint:* The offloading probability $x[n]$ of mobile users is affected by service prices $\boldsymbol{p}[n] = \{p_j[n] : j \in \mathcal{M}\}$ set by the provider. In general, setting high prices will restrain user demands of task offloading, or low $x[n]$, whereas positing low prices will lead to *overloaded situations* at edge servers because of too many offloaded tasks. As such, the provider will adaptively choose $\boldsymbol{p}[n]$ for compensating servers' operation costs and avoiding being overloaded. Note that users will not impose a charge on each other since they are supposed to "jointly" process the tasks during D2D collaboration.

*3) Problem Formulation:* We now formulate the system cost of users and the service utility of servers in task offloading.

**Users' system cost**. Since D2D communication is energy efficient, the system cost of a user mainly consists of the charged fee, processing cost and offloading transmission cost. Formally, the charged fee is the payment to edge servers for task offloading, which is $x[n]\lambda \sum_{j=1}^{M} f_j p_j[n]$. When processing a task, it needs an average $\frac{1}{\mu}$ time, and hence the energy consumption is $\rho_c^m \frac{1}{\mu}$ where $\rho_c^m$ is the energy cost per CPU cycle for computation in a mobile device [27]. The busy probability that a user holds at least one task is $s_1^* \triangleq \sum_{k \in \mathcal{K}} \Pr(k) s^*_{k,1}$ from Eq. (3), so that we have the processing cost as $s_1^* \frac{\rho_c^m}{\mu}$. Finally, the transmission cost is $x[n]\lambda\rho_t^m \sum_{j=1}^{M} f_j \frac{B}{r_j}$ where $\rho_t^m$ is the unit cost for transmitting cellular traffic. Overall, the system cost $c[n]$ in time slot $n$ is:

$$c[n] = x[n]\lambda \sum_{j=1}^{M} f_j p_j[n] + s_1^* \frac{\rho_c^m}{\mu} + x[n]\lambda\rho_t^m \sum_{j=1}^{M} f_j \frac{B}{r_j}. \tag{5}$$

**Server's service utility**. On the side of server $j$, its profit in time slot $n$ is $x[n]\lambda f_j p_j[n]$, and processing cost is $x[n]\lambda f_j \frac{\rho_j}{\gamma_j}$ with $\rho_j$ representing the energy cost per CPU cycle. Then, the service utility $u_j[n]$ is acquired:

$$u_j[n] = x[n]\lambda f_j p_j[n] - x[n]\lambda f_j \frac{\rho_j}{\gamma_j}. \tag{6}$$

Moreover, denote $\overline{u}_j = \lim_{T\to\infty} \frac{1}{T} \sum_{n=0}^{T-1} \mathbb{E}[u_j[n]]$ as the time average service utility.

**Stackelberg game**. Given prices $\boldsymbol{p}[n]$, users aim to reduce their system cost by deciding probability $x[n]$, subject to constraints of task delay and collaboration fairness:

$$\min_{x[n]} \quad c[n] \tag{7}$$
$$\text{s.t.} \quad d(x[n]) \le \overline{d} \tag{7a}$$
$$\max\{s^*_{k,1}\} - \min\{s^*_{k,1}\} \le \overline{s}. \tag{7b}$$

As for the provider, its objective is to optimize the long-term service value $\sum_{j=1}^{M} g(\overline{u}_j)$ via dynamically setting service prices $\boldsymbol{p}[n]$. The function $g(\cdot)$ has two forms: 1) $g(\overline{u}_j) = \overline{u}_j$, which means optimizing the service value is equivalent to utilitarian utility maximization; 2) $g(\overline{u}_j) = \log(1 + \beta\overline{u}_j)$, which further considers the proportional fairness among edge servers when maximizing the service value [30]. Overall the Dynamic Service Pricing of Value Maximization (DSP-VM) is formulated:

$$\max_{\boldsymbol{p}[n]} \quad \sum\nolimits_{j=1}^{M} g(\overline{u}_j) \tag{8}$$
$$\text{s.t.} \quad \lim_{T\to\infty} \frac{1}{T} \sum\nolimits_{n=0}^{T-1} \mathbb{E}[x[n]\lambda f_j] \le \overline{x}_j, \quad \forall j \in \mathcal{M} \tag{8a}$$
$$p_j[n] \in (0, p_u], \quad \forall j \in \mathcal{M}, \tag{8b}$$

where the inequality in Eq. (8a) is the overloaded constraint and $p_u$ denotes the highest price users could accept.

In time slot $n$, the provider first chooses prices $\boldsymbol{p}[n]$, and then users react via the offloading decision $x[n]$, which is modeled as a Stackelberg game. At a high level, mobile users

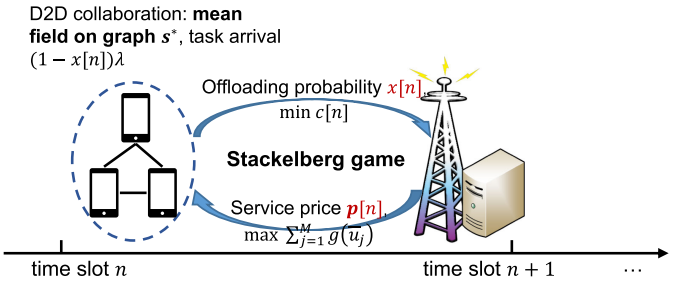


Fig. 2. Task offloading overview.

TABLE I
NOTATION

| Notation | Description |
|---|---|
| $N, M$ | # of users, # of edge servers |
| $\mathcal{G}=\{\mathcal{N},\mathcal{E}\}$ | D2D network, user set, D2D links |
| $f_j, p_j, r_j$ | processing probability, service price, cellular data rate |
| $x, \lambda$ | offloading probability, task generation rate |
| $B$ | average data size of tasks |
| $\mu, \gamma_j$ | service rates of mobile device, edge server |
| $t, n$ | time index, time slot index |
| $Q_u(t)$ | user workload |
| $\mathcal{K}, \overline{k}$ | user degree set, expected degree |
| $\Pr(k'\|k), \Pr(k)$ | conditioned degree probability, degree probability |
| $q_{k,i}(t), s_{k,i}(t)$ | probability of workload at a $k$-degree user |
| $d(x[n]), d_o, d_q$ | task delay, offloading delay, queuing delay |
| $\rho_c^m/\rho_j, \rho_t^m$ | user/server energy cost, user transmission cost |
| $c[n], u_j[n]$ | user system cost, server service utility |
| $\overline{d}, \overline{s}$ | delay constraint, fairness constraint |
| $g(\cdot), p_u$ | server objective, highest accepted price |
| $x_l^*, x_u^*, x_l', x_u'$ | feasible ranges of offloading probability |

will collectively decide the offloading probability to reduce their average system cost while respecting the delay and fairness constraints, and the provider aims to set optimal prices for service value maximization. Note that optimizing the cost and value is intertwined with D2D collaboration, as shown in Fig. 2. More concretely, different $x[n]$ will lead to different user's system cost and system state $\boldsymbol{s}^*$, thus coupling the mean field model and Stackelberg game. Also, we have to design an offloading and pricing scheme to simultaneously maximize the service value and minimize the system cost online. The main notations are listed in Table I.

## IV. Mean Field D2D Collaboration

In this section, we formulate a mean field model on graph to analyze D2D collaboration. Specifically, we will derive the state evolution by allowing the user number $N$ to approach infinity. Since users may move around, we consider the decentralized collaboration on both static and dynamic graphs to encompass the case of *time-varying D2D links*.

### A. Collaboration on Static Graph

Static graph implies that D2D links are time-invariant, i.e., the graph $\mathcal{G}$ remains unchanged throughout the collaboration. To characterize state $\boldsymbol{s}$, we explore the transition of each $s_{k,i}$ from the perspective of a particular $k$-degree user $u$, as users are asymptotically independent when $N \to \infty$ according to the propagation of chaos [31], [32], [33].

*1) State Evolution:* Considering $s_{k,i}$ represents the workload distribution of $k$-degree users, then the transition events for the Markov chain include the following three instances:

- The number of tasks is $Q_u = i-1$, and $u$ generates a task which stays at $u$, so the state transits from $s_{k,i-1}$ to $s_{k,i}$.
- The number of tasks is $Q_u = i-1$, and $u$ receives a task sent from a neighbor, then $s_{k,i-1}$ transits to $s_{k,i}$.
- The number of tasks is $Q_u = i$, and $u$ processes a task locally. Hence, the state changes from $s_{k,i}$ to $s_{k,i-1}$.

Considering that each user is stochastically independent due to propagation of chaos, we next characterize the transition probability of every instance above. For the first instance, the polled neighbor by $u$ must have no fewer tasks, which occurs with probability $q_{k,i-1}\sum_{k'\in\mathcal{K}}\Pr(k'|k)s_{k',i} + q_{k,i-1}\frac{1}{2}\sum_{k'\in\mathcal{K}}\Pr(k'|k)q_{k',i-1}$. The first term $q_{k,i-1}\sum_{k'\in\mathcal{K}}\Pr(k'|k)s_{k',i}$ means $Q_u = i-1$ and the polled neighbor has at least $i$ tasks. The second term $q_{k,i-1}\frac{1}{2}\sum_{k'\in\mathcal{K}}\Pr(k'|k)q_{k',i-1}$ denotes tie breaking, namely the polled user also holds $i-1$ tasks. As for the second instance that $u$ receives a task from a neighbor, its probability is $kq_{k,i-1}\sum_{k'\in\mathcal{K}}\Pr(k'|k)s_{k',i}\frac{1}{k'} + kq_{k,i-1}\frac{1}{2}\sum_{k'\in\mathcal{K}}\Pr(k'|k)q_{k',i-1}\frac{1}{k'}$, where $kq_{k,i-1}$ is because $u$ has $k$ neighbors and $\frac{1}{k'}$ implies that a $k'$-degree neighbor randomly polls $u$ in Po2 choices (D2D collaboration) with probability $\frac{1}{k'}$. Similarly, the two terms represent situations where $u$ has fewer tasks and tie breaks, respectively. The probability of the last instance is simply $q_{k,i}$. Combining task generation rate $\lambda$, offloading probability $x$ and service rate of a mobile device $\mu$, the state evolution is specified:

$$\begin{aligned}\dot{s}_{k,i} = &-\mu q_{k,i} + x_c\lambda q_{k,i-1}\sum\nolimits_{k'\in\mathcal{K}}\Pr(k'|k)\Big(s_{k',i}+\frac{1}{2}q_{k',i-1}\Big)\\ &+ kx_c\lambda q_{k,i-1}\sum\nolimits_{k'\in\mathcal{K}}\frac{1}{k'}\Pr(k'|k)\Big(s_{k',i}+\frac{1}{2}q_{k',i-1}\Big),\end{aligned} \tag{9}$$

where $x_c = 1-x$ is the probability that a task is processed via D2D collaboration and $x$ would be $x[n]$ if in time slot $n$.

*2) ODE System:* Remember that $q_{k,i} = s_{k,i} - s_{k,i+1}$ and $\mathcal{G}$ is an uncorrelated network. Based on Eq. (1), we have $\Pr(k'|k) = \frac{k'\Pr(k')}{\overline{k}}$. Therefore, for $i>0$, Eq. (9) is simplified:

$$\begin{aligned}\dot{s}_{k,i} = &-\mu(s_{k,i}-s_{k,i+1}) + x_c\lambda(s_{k,i-1}-s_{k,i})\\ &\times\Big[\frac{1}{2}\sum\nolimits_{k'\in\mathcal{K}}\frac{k'+k}{\overline{k}}\Pr(k')(s_{k',i-1}+s_{k',i})\Big].\end{aligned} \tag{10}$$

Besides, $s_{k,0} = 1$ according to Eq. (3). Define the drift function $\boldsymbol{F}(\boldsymbol{s}) = \{F_{k,i}(\boldsymbol{s})\}$, where $F_{k,0}(\boldsymbol{s}) = 0$ and $F_{k,i}(\boldsymbol{s}) = \dot{s}_{k,i}, \forall i>0$. We have the following form:

$$\dot{\boldsymbol{s}} = \boldsymbol{F}(\boldsymbol{s}). \tag{11}$$

The *deterministic ODE system* of Eq. (10) corresponds to the *mean field model for our characterized D2D collaboration*.

### B. Collaboration on Dynamic Graph

As users may move around, their neighbors within D2D communication range also change accordingly, resulting in a time-varying graph structure. Specifically, we leverage the model in [34] and [35] to capture this dynamic feature.

*1) Dynamic Graph Model:* In a dynamic graph, each user has a fixed expected degree $k$ drawn from a finite set $\mathcal{K} = \{k_{\min}, \ldots, k_{\max}\}$ with probability $\Pr(k)$. Value of expected degree implies the willingness of a user to collaborate. Hence, a D2D link between two users is established based on their expected degrees and spatial distance. In this regard, the number of D2D neighbors, or realized degree, of a user follows certain distribution conditioned on its expected degree $k$, which is specified as a Poisson distribution with the mean value being $k$ in [35]. Due to user mobility, graph $\mathcal{G}(t)$ is dynamic with a time-varying edge set $\mathcal{E}(t)$, or changing realized degrees. Similar to the static case, the D2D network $\mathcal{G}(t)$ formed by realized degrees is considered uncorrelated. Sometimes, there is a temporal correlation between new graph $\mathcal{G}(t)$ and its previous structure $\mathcal{G}(t'), t' < t$. We should emphasize that as long as $\mathcal{G}(t)$ is uncorrelated, our later mean field result will hold, i.e., temporal correlation will not affect our analysis.

*2) State Evolution:* Define $q_{k,i}, s_{k,i}$ as Eqs. (2)-(3), whereas $k$ now denotes the *expected degree*. Akin to the static graph, transitions of state $s_{k,i}$ on dynamic graph also entail three instances. The main difference lies in the probability $\Pr(k'|k)$, a user has expected degree $k$ and its neighbor has expected degree $k'$, is no longer the expression in Eq. (1). Instead, we use the conditional distribution of the realized degree to help compute this $\Pr(k'|k)$. For compactness, we elucidate the details of deriving the state evolution in Appendix A as shown in the supplementary material, and only present the final result here. We find out that the evolution of $s_{k,i}$ is exactly Eq. (10), i.e., *the mean field models on static and dynamic graphs are unified by the same ODE system.* Hereinafter, we will rely on Eq. (10) to analyze D2D collaboration in both static and dynamic scenarios as a whole.

### C. Mean Field Model on Finite-Degree Graphs

There have been many efforts devoted to mean field model on graphs, while existing works mostly focus on complete and infinite-degree graphs [18], [20], or apply the mean field analysis without theoretical guarantees [19], [22]. Therefore, our work has two novel contributions in the mean field aspect. First, we extend current mean field model on graphs to both *static and dynamic graphs with finite yet heterogeneous degrees* as long as they are uncorrelated. Second, we also provide *rigorous proofs* (in the next section), along with extensive evaluations to show the effectiveness of our mean field model.

Now we discuss a special case where users have a homogeneous degree, that is the degree set $\mathcal{K} = \{k\}$ and $\Pr(k) = 1$. Consider that the graph $\mathcal{G}$ is connected and uncorrelated. The mean field model becomes (irrelevant to degree $k$ indeed):

$$\dot{s}_{k,i} = x_c \lambda (s_{k,i-1}^2 - s_{k,i}^2) - \mu (s_{k,i} - s_{k,i+1}). \tag{12}$$

This in fact degenerates to the classical Po2 result [18].

Our main objective is to obtain the stationary point $\boldsymbol{s}^*$ for state evolution, such that $\boldsymbol{F}(\boldsymbol{s}^*) = \mathbf{0}$. To this end, we will derive the existence and uniqueness of the stationary point, as well as showing that the system state from *any initial point* will eventually converge to this stationary point.

## V. Stationary Point for State Evolution

In this section, we specify the stationary point for the mean field model to obtain the steady state of D2D collaboration.

### A. Stationary Point

Since drift at the stationary point is 0, the MEC system is statistically stable, and hence we can achieve a tractable collaboration performance. For this reason, the existence and uniqueness issues of stationary point need to be explored.

*1) Existence of Stationary Point:* We first show there exists a stationary point $\boldsymbol{s}^*$ for mean field model. Considering $\boldsymbol{F}(\boldsymbol{s}^*) = \mathbf{0}$, we attain the existence by proving the ODE system of Eq. (10) has a fixed point.

*Theorem 1: There exists a stationary point $\boldsymbol{s}^*$ for the mean field model.*

See Appendix B in the supplementary material for the proof. Though $\boldsymbol{s}^*$ exists, we still can not use $\boldsymbol{s}^*$ to directly represent the steady system state, as there may be multiple stationary points or the state $\boldsymbol{s}(t)$ may not converge to $\boldsymbol{s}^*$. This requires to further address the issues of unique stationary point and convergence of state $\boldsymbol{s}(t)$.

*2) Uniqueness and Convergence:* Due to the graph structure, mean field model also depends on the degree distribution, which makes it difficult to derive the unique stationary point and state convergence. To circumvent this problem, we will handle the uniqueness and convergence issues alternatively.

**Coordinate-wise dominance**. The state evolution of $s_{k,i}(t)$ is identified by the ODE of Eq. (10). To obtain the state convergence, we have to figure out how the initial values $\boldsymbol{s}(0)$ would influence the state $\boldsymbol{s}(t)$ at later time $t$. Define coordinate-wise dominance $\boldsymbol{s} \succeq \hat{\boldsymbol{s}}$ if $s_{k,i} \geq \hat{s}_{k,i}, \forall k \in \mathcal{K}, i \geq 0$. The lemma below states that the dominance at any time $t$ is consistent with that of the initial values.

*Lemma 1: Let $\boldsymbol{s}(t)$ and $\hat{\boldsymbol{s}}(t)$ be the solutions to the ODE system of Eq. (10) at time $t$ with the initial values being $\boldsymbol{s}(0)$ and $\hat{\boldsymbol{s}}(0)$, respectively. If $\boldsymbol{s}(0) \succeq \hat{\boldsymbol{s}}(0)$, then $\boldsymbol{s}(t) \succeq \hat{\boldsymbol{s}}(t)$.*

See Appendix C-A in the supplementary material for the proof. With the dominance consistency, we demonstrate that every trajectory of the state converges to the stationary point in an appropriate metric.

**Exponential convergence rate**. To show convergence, we need to find a Lyapunov function $\phi(\mathbf{s})$ which satisfies: 1) $\phi(\mathbf{s})$ relates to the distance between $\boldsymbol{s}$ and $\boldsymbol{s}^*$; 2) $\phi(\mathbf{s})$ is strictly decreasing, except at $\boldsymbol{s}^*$. Here, $\phi(\mathbf{s})$ is constructed:

$$\phi(\mathbf{s}) = \min_{\boldsymbol{s}^* \in \mathcal{S}^*} \sum\nolimits_{i \geq 0} \frac{|\sum_{k \in \mathcal{K}} \Pr(k)(s_{k,i} - s_{k,i}^*)|}{2^i}, \tag{13}$$

where $\mathcal{S}^*$ is the stationary point set. To simplify $\phi(\mathbf{s})$, we denote $s_i = \sum_{k \in \mathcal{K}} \Pr(k) s_{k,i}$ and $s_{(k),i} = \sum_{k \in \mathcal{K}} \Pr(k) k s_{k,i}$. In line with Eq. (10), the evolution of $s_i$ is rewritten as:

$$\dot{s}_i = \frac{x_c \lambda}{\bar{k}} \left( s_{i-1} s_{(k),i-1} - s_i s_{(k),i} \right) - \mu (s_i - s_{i+1}). \tag{14}$$

Using $s_i$, we have $\phi(\mathbf{s}) = \min_{\boldsymbol{s}^* \in \mathcal{S}^*} \sum_{i \geq 0} \frac{|s_i - s_i^*|}{2^i}$. The convergence of state $\boldsymbol{s}$ is derived by showing $\phi(\mathbf{s}) \to 0$.

*Lemma 2: If initial points $\boldsymbol{s}(0) \succeq \boldsymbol{s}^*, \forall \boldsymbol{s}^* \in \mathcal{S}^*$ or $\boldsymbol{s}^* \succeq \boldsymbol{s}(0), \forall \boldsymbol{s}^* \in \mathcal{S}^*$, $\boldsymbol{s}(t)$ converges to $\mathcal{S}^*$ with exponential rate.*

See Appendix C-B in the supplementary material for the proof. The exponential convergence rate in Lemma 2 is derived under certain initial conditions. However, if there is only one stationary point, global convergence from any initial point will be naturally obtained. This is because when we have a unique stationary point $\boldsymbol{s}^*$, $\boldsymbol{s}(t)$ always converges to $\boldsymbol{s}^*$ if $\boldsymbol{s}(0) \succeq \boldsymbol{s}^*$ or $\boldsymbol{s}^* \succeq \boldsymbol{s}(0)$. The dominance consistency then ensures the convergence of $\boldsymbol{s}(t)$ to $\boldsymbol{s}^*$ from other initial values $\boldsymbol{s}(0)$, because we can always choose two extreme points $\boldsymbol{s}'(0), \hat{\boldsymbol{s}}(0)$ to encompass $\boldsymbol{s}(0)$ and meanwhile both $\boldsymbol{s}'(t), \hat{\boldsymbol{s}}(t)$ will converge to the same $\boldsymbol{s}^*$ exponentially fast.

**Unique stationary point**. Now we show that our mean field model has a unique stationary point, which will be proved by combining the dominance consistency property and the exponential convergence result.

*Theorem 2: There is a unique stationary point $\boldsymbol{s}^*$ for the mean field model.*

See Appendix C-C in the supplementary material for the proof. Lemma 2 and Theorem 2 imply the result as stated in the following theorem.

*Theorem 3: The state evolution of D2D collaboration globally converges to a unique stationary point with exponential rate.*

### B. Influence of Heterogeneous Degrees

In minimizing the system cost, one constraint is the collaboration fairness in Eq. (7b) pertaining to heterogeneous degrees. Concretely, a user with larger degree receives more tasks in D2D collaboration, and has a higher probability to forward tasks out. This effect is presented below.

*Theorem 4: At the stationary point $\boldsymbol{s}^*$, larger-degree users tend to have heavier workloads: $s^*_{k,i} \geq s^*_{k',i}, \forall i \geq 0$ if $k > k'$.*

Please refer to Appendix D in the supplementary material for the proof. This theorem reveals that users with heterogeneous degrees have *uneven workloads* because larger-degree users generally have higher contributions in D2D collaboration. Note that $s^*_{k,1}$ indicates the busy probability of a $k$-degree user with processing tasks. To avoid small-degree users from free-riding large-degree neighbors, we bound the gap between $\max\{s^*_{k,1}\}$ and $\min\{s^*_{k,1}\}$ in the system cost minimization so as to ensure the collaboration fairness, where $\max\{s^*_{k,1}\}$ and $\min\{s^*_{k,1}\}$ now become $s^*_{k_{\max},i}$ and $s^*_{k_{\min},i}$, respectively, from Theorem 4. Fig. 3 compares typical values of $s^*_{k_{\max},i}$ and $s^*_{k_{\min},i}$ for some basic understanding.

### C. Convergence to Mean Field Model

The mean field model makes use of a deterministic ODE system to analyze D2D collaboration in a stochastic MEC system. In the following, we show that the original stochastic $N$-user system will converge to the deterministic mean field model when $N$ is large, namely the ODE system of Eq. (10) is accurate in describing the state evolution.

Consistent with Eq. (3), we denote $\boldsymbol{s}^{(N)}(t)$ as the state of $N$-user system. Our goal is to demonstrate that $\boldsymbol{s}^{(N)}(t) \to \boldsymbol{s}(t)$ when $N \to \infty$. To this end, we first prove that the drift function $\boldsymbol{F}(\boldsymbol{s})$ in Eq. (11) is Lipschitz continuous.

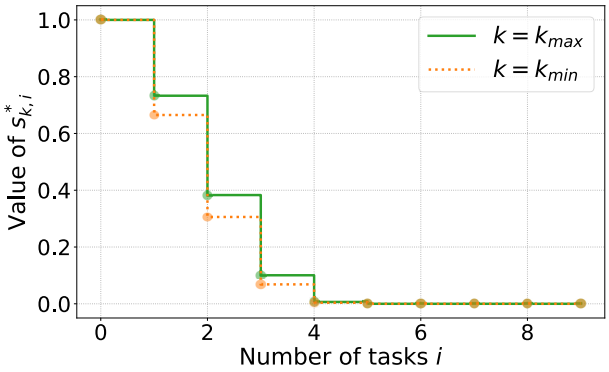


Fig. 3. Illustration of workload distribution.

*Lemma 3: The drift function $\boldsymbol{F}(\boldsymbol{s})$ is $||\cdot||_\infty$-Lipschitz as there is a constant $C = 3x_c\lambda\left(1 + \frac{k_{\max}}{\bar{k}}\right) + 2\mu$ such that for any $\boldsymbol{s}, \hat{\boldsymbol{s}}$:*

$$||\boldsymbol{F}(\boldsymbol{s}) - \boldsymbol{F}(\hat{\boldsymbol{s}})||_\infty \leq C||\boldsymbol{s} - \hat{\boldsymbol{s}}||_\infty. \tag{15}$$

See Appendix E-A in the supplementary material for the proof. With Lemma 3, we can claim the convergence of $N$-user system to the mean field model based on the Kurtz's theorem (Theorem 2.2 in [37]).

*Theorem 5: Fix a time $t^*$. When the number of users $N \to \infty$, the state $\boldsymbol{s}^{(N)}(t)$ converges to $\boldsymbol{s}(t)$ in the mean field model of Eq. (10) almost surely (a.s.) if they start from the same initial points.*

$$\lim_{N\to\infty} \sup_{t\in[0,t^*]} ||\boldsymbol{s}^{(N)}(t) - \boldsymbol{s}(t)||_\infty = 0, \text{a.s.} \tag{16}$$

Please refer to Appendix E-B in the supplementary material for the detailed proof. Theorem 5 ensures that the mean field model is effective for large population $N$. On this basis, one can obtain the existence and uniqueness of stationary point to characterize the steady state of the MEC system and show the power of D2D collaboration.

Compared to previous works [9], [10], [11], our mean field model not only enables characterizing large-scale D2D collaboration, but also ensures the system convergence to a steady state when users make offloading decisions in a decentralized manner.

### D. Discussion of Stationary Point

*1) Relation to Classical Po2:* The mean field models on static and dynamic graphs are unified by the ODE system of Eq. (10). When users have a homogeneous degree, Eq. (10) boils down to the classical Po2 of Eq. (12), which is independent of the degree and has a closed-form stationary point [18]:

$$\pi^*_i = \left(\frac{x_c\lambda}{\mu}\right)^{2^i-1}. \tag{17}$$

For a more general graph of heterogeneous degrees, an explicit expression for stationary point $\boldsymbol{s}^*$ is not available. Nevertheless, we can use the traditional Po2 as a bound for $\boldsymbol{s}^*$. Define two ratios:

$$\delta_1 = \frac{k_{\max}}{\bar{k}}, \ \delta_2 = \frac{k_{\min}}{\bar{k}}. \tag{18}$$

Here, $\delta_1 > 1$ and $\delta_2 < 1$. Also, let $\frac{1+\delta_1}{2}\frac{x_c\lambda}{\mu} < 1$ to guarantee the system stability.

*Corollary 1: For the mean field model of Eq. (10), if* $\frac{1+\delta_1}{2}\frac{x_c\lambda}{\mu} < 1$, *then* $s^*_{k,i}, \forall k, i$ *has an upper bound*:

$$s^*_{k,i} \leq \left(\frac{1+\delta_1}{2}\frac{x_c\lambda}{\mu}\right)^{2^i-1}, \tag{19}$$

*and a lower bound*:

$$s^*_{k,i} \geq \left(\frac{1+\delta_2}{2}\frac{x_c\lambda}{\mu}\right)^{2^i-1}. \tag{20}$$

See Appendix F-A in the supplementary material for the proof. From this corollary, if the degree distribution is slightly heterogeneous, i.e., both the ratios $\delta_1$ and $\delta_2$ in Eq. (18) are close to 1, the gap between upper and lower bounds will be small, thereby leading to an accurate estimation of $s^*_{k,i}$ with closed-form expressions.

*2) Busy Probability:* The probability that a user is busy with processing task, $s^*_1 = \sum_{k\in\mathcal{K}} \Pr(k) s^*_{k,1}$, is critical in computing the system cost in Eq. (5). The corollary below provides the value of $s^*_1$ with the proof being presented in Appendix F-B in the supplementary material.

*Corollary 2: The busy probability is* $s^*_1 = \frac{x_c\lambda}{\mu}$.

*3) Workload Distribution Relation:* In the end of this section, we illustrate the relation between $s^*_i$ and $s^*_{i-1}$. Based on Eq. (17), traditional Po2 satisfies $\pi^*_i = \frac{x_c\lambda}{\mu}\left(\pi^*_{i-1}\right)^2$. As for our mean field model on graph, similar conclusion is attained. Specifically, from Eq. (14), we know that the stationary point satisfies the following condition:

$$\frac{x_c\lambda}{\overline{k}}\left(s^*_{i-1}s^*_{(k),i-1} - s^*_i s^*_{(k),i}\right) - \mu(s^*_i - s^*_{i+1}) = 0. \tag{21}$$

*Corollary 3: For any* $i \geq 1$, *we have* $s^*_i = \frac{x_c\lambda}{\overline{k}\mu} s^*_{i-1} s^*_{(k),i-1}$.

See Appendix F-C in the supplementary material for the proof. Compared to the classical Po2, the graph structure causes the difference between $\pi^*_i = \frac{x_c\lambda}{\mu}\pi^*_{i-1} \times \pi^*_{i-1}$ and $s^*_i = \frac{x_c\lambda}{\mu} s^*_{i-1} \times \frac{s^*_{(k),i-1}}{\overline{k}}$.

## VI. Online Offloading And Pricing of Server Utility Maximization

D2D collaboration can reduce the workloads of users. However, due to constrained resources of mobile devices, offloading a portion of tasks to more powerful edge servers is still essential to further mitigate task execution delay. Along with the offloading, there is a price charged by the provider who manages edge servers, so that users have to balance how many tasks should be offloaded and how many should be processed collaboratively. The task offloading process is modeled as a Stackelberg game, where the provider is the leader in setting service prices which remain fixed for a long time, and users are followers in deciding the offloading probability. The provider aims to optimize the service value, including the service utility and proportional fairness depending on the form of $g(\cdot)$. This section is devoted to tackling the utility maximization.

For utility maximization, we have $g(\overline{u}_j) = \overline{u}_j$ in Eq. (8). The provider will solve DSP-VM with the objective being the cumulative time average utility $\lim_{T\to\infty}\frac{1}{T}\sum_{n=0}^{T-1}\sum_{j=1}^{M}\mathbb{E}[u_j[n]]$, by deciding the optimal prices $\boldsymbol{p}[n]$ in each time slot $n$. In the meantime, mobile users aim to reduce their system cost while maintaining satisfactory task delay and collaboration fairness through determining the offloading probability $x[n]$, which is described in Eq. (7). To achieve these two goals, *Lyapunov optimization is leveraged to maximize the long-term utility, with each time slot amounting to a Stackelberg game to minimize the system cost.*

### A. Optimal Task Offloading

In time slot $n$, assume the provider has declared the service prices $\boldsymbol{p}[n]$. Users then collectively determine the offloading decision $x[n]$ to minimize their system cost $c[n]$. Combining Eq. (5) and the busy probability in Corollary 2, we have:

$$c[n] = x[n]\lambda\sum_{j=1}^{M} f_j p_j[n] + (1-x[n])\lambda\frac{\rho^m_c}{\mu^2} + x[n]\lambda\rho^m_t\sum_{j=1}^{M} f_j\frac{B}{r_j}. \tag{22}$$

*1) Critical Points:* Users will derive the optimal probability $x[n]$ to minimize the system cost $c[n]$ by respecting the delay and fairness constraints. In particular, changes of $x[n]$ influence both the task delay and collaboration fairness as different portions of tasks will be processed by edge servers and D2D collaboration. Therefore, we need to first characterize the feasible range when both constraints are satisfied. Specifically, task delay $d(x[n])$ in Eq. (4) is composed of two parts. The first part is the transmission delay and completion time of task offloading, that is $d_o = \sum_{j=1}^{M} x[n] f_j\left(\frac{B}{r_j} + \frac{1}{\gamma_j}\right)$. The second part corresponds to the sojourn time of D2D collaboration $d_q = \frac{\sum_i\sum_k \Pr(k) s^*_{k,i}}{\lambda}$. If $x[n]$ increases from 0 to 1, $d_o$ will monotonically increase whereas $d_q$ will monotonically decrease. Hence, there exist a lower bound $x^*_l$ and an upper bound $x^*_u$ such that when $x[n] \in [x^*_l, x^*_u]$ the delay constraint is fulfilled, where critical points $x^*_l, x^*_u$ satisfy:

$$d\left(x[n] = x^*_l\right) = d\left(x[n] = x^*_u\right) = \overline{d}. \tag{23}$$

Note that the stationary point $\boldsymbol{s}^*$ is dependent on $x[n]$. We will compute each $s^*_{k,i}$ numerically given the value of $x[n]$, say $x[n] = x^*_l, x^*_u$, since there is no closed-form solution.

Let us discuss the collaboration fairness constraint, which is $s^*_{k_{\max},1} - s^*_{k_{\min},1} \leq \overline{s}$ according to Theorem 4. The trend of the gap $s^*_{k_{\max},1} - s^*_{k_{\min},1}$ over $x[n]$ is not obvious. Nevertheless, if $x[n]$ approaches 1, i.e., users offload all their tasks to the edge servers, both $s^*_{k_{\max},1}$ and $s^*_{k_{\min},1}$ will be 0, and then their gap will be 0. When we push $x[n]$ approaching 0, that is users do not offload but only collaborate, all users will be heavily loaded, and hence $s^*_{k_{\max},1}$ and $s^*_{k_{\min},1}$ will be close to $\frac{\lambda}{\mu}$, with the gap being very small. As a result, we can characterize two feasible regions for $x[n]$: $[0, x'_l] \cup [x'_u, 1]$. If jointly considering the delay and fairness constraints, Fig. 4 provides a typical illustration of the feasible region for the offloading decision $x[n]$. In this paper, we assume that $[x^*_l, x^*_u] \cap ([0, x'_l] \cup [x'_u, 1]) \neq \emptyset$, namely $x[n]$ has a feasible solution. For any feasible region, we denote $x_l$ and $x_u$ as its lower and upper boundary points, respectively, which hinge on the values of $x^*_l, x^*_u, x'_l, x'_u$, as also displayed in Fig. 4.

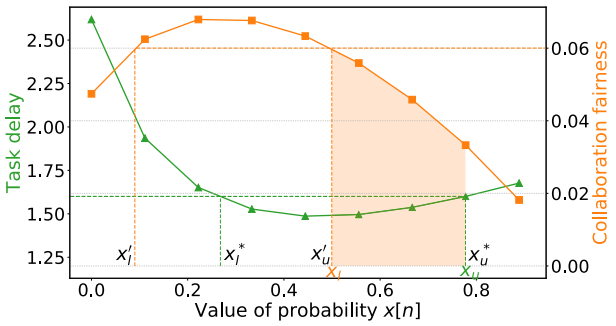


Fig. 4. Feasible region for offloading decision.

*2) Threshold Based Offloading Decision:* To minimize the system cost in Eq. (22), which is linear in the offloading probability when provided the price information $\boldsymbol{p}[n]$, the optimal probability $x[n]$ for users is specified by a *threshold based decision*:

$$x[n]=\begin{cases} x_u \text{ if } \dfrac{\rho_c^m}{\mu^2}\geq \displaystyle\sum_{j=1}^{M} f_j p_j[n]+\rho_t^m \sum_{j=1}^{M} f_j \frac{B}{r_j}, \\ x_l \text{otherwise}. \end{cases} \tag{24}$$

The intuition behind Eq. (24) is now explained. If the provider sets excessively high prices such that the cost of offloading task is greater than the cost of processing task collaboratively, users will offload as few tasks as possible to reduce their system cost, and vice versa. With this reacted offloading decision $x[n]$, the provider in turn determines the optimal prices $\boldsymbol{p}[n]$ in each time slot to maximize the cumulative utility of all edge servers. Note that the critical points $x_l, x_u$ are altered by $\boldsymbol{p}[n]$ in that prices will affect the offloading decision. Recall from the overloaded constraint in Eq. (8a), it implies that $\overline{x}_j \in [x_l\lambda f_j, x_u\lambda f_j], \forall j \in \mathcal{M}$.

Knowledgeable readers may ask how massive users reach a consensus on the same optimal offloading probability, which may need costly information exchange. Actually, each user can first obtain the steady system state through the mean field model, and then he could locally derive Eq. (24) without extra inter-user communications. Since users can benefit from D2D collaboration, they have the incentive to collectively reduce the average system cost, rather than only minimizing their own.

### B. Dynamic Service Pricing

The provider will judiciously choose the prices over time to maximize the long-term service utility only using observed interacted results. Considering this, we employ an online Lyapunov optimization framework to attain the optimal utility. For ease of exposition, denote $U[n] := \sum_{j=1}^{M} u_j[n]$ as the cumulative utility of all edge servers in time slot $n$.

*1) Lyapunov Based Utility Optimization:* **Drift-minus-utility**. In view of the overloaded constraint in DSP-VM, we define a virtual queue $X_j[n]$ for edge server $j$ which buffers the virtual amounts of offloaded tasks. Here, we use the prefix "virtual" to denote that tasks are not actually offloaded from users, but rather, to reflect the requirement of the overloaded constraint. Consistent with this queue definition, tasks will enter into the queue with arrival rate $x[n]\lambda f_j$ where $x[n]$ is users' offloading probability, and will leave the queue with departure rate $\overline{x}_j$. Therefore, we have the following dynamics for the virtual queue $X_j[n]$:

$$X_j[n+1]=\max(X_j[n]+x[n]\lambda f_j-\overline{x}_j, 0). \tag{25}$$

Let $\boldsymbol{X}[n]=\{X_j[n]:\forall j\in\mathcal{M}\}$ be the queue set. Based on Eq. (25), we construct Lyapunov function as $L(\boldsymbol{X}[n])=\sum_{j=1}^{M}\frac{1}{2}X_j^2[n]$, and compute Lyapunov drift which is the change of Lyapunov function from one time slot to the next:

$$\begin{aligned}\Delta(\boldsymbol{X}[n]) &= \mathbb{E}[L(\boldsymbol{X}[n])-L(\boldsymbol{X}[n])|\boldsymbol{X}[n]] \\ &= \mathbb{E}\Big[\sum_{j=1}^{M}\frac{1}{2}X_j^2[n+1]-\sum_{j=1}^{M}\frac{1}{2}X_j^2[n]\Big|\boldsymbol{X}[n]\Big]. \end{aligned}\tag{26}$$

The expectation is taken over the randomness in task generation, offloading decision and pricing scheme. By minimizing Lyapunov drift $\Delta(\boldsymbol{X}[n])$, one can drive the queue backlog to a small value so as to maintain each $X_j[n]$ rate stable, $\lim_{n\to\infty}\frac{X_j[n]}{n}=0$, with probability 1. From the queue stability theorem (Theorem 4.1 in [38]), a queue $X_j[n]$ is stable if and only if the arrival rate is no larger than the departure rate, i.e., $\lim_{T\to\infty}\frac{1}{T}\sum_{n=0}^{T-1}\mathbb{E}[x[n]\lambda f_j]\leq\overline{x}_j$, and hence the overloaded constraint is satisfied. Furthermore, we define drift-minus-utility $\Delta(\boldsymbol{X}[n])-V\sum_{j=1}^{M}\mathbb{E}[u_j[n]|\boldsymbol{X}[n]]$, where $V$ is the importance weight on the utility term. Minimizing drift-minus-utility will *simultaneously push the queue backlog to a small value and maximize the utility* as well [38].

**Bound of drift-minus-utility**. In time slot $n$, the queue value $\boldsymbol{X}[n]$ is known in advance. Besides, the offloading decision $x[n]$ is given in Eq. (24), which is also expressed as $x(\boldsymbol{p}[n])$ to explicitly indicate its dependence on prices $\boldsymbol{p}[n]$.

*Lemma 4: Drift-minus-utility* $\Delta(\boldsymbol{X}[n])-V\mathbb{E}[U[n]|\boldsymbol{X}[n]]$ *satisfies the following condition*:

$$\begin{aligned}&\Delta(\boldsymbol{X}[n])-V\sum_{j=1}^{M}\mathbb{E}[u_j[n]|\boldsymbol{X}[n]] \\ &\leq \mathbb{E}\Big[\sum_{j=1}^{M}X_j[n](x(\boldsymbol{p}[n])\lambda f_j-\overline{x}_j)|\boldsymbol{X}[n]\Big] \\ &\quad-\mathbb{E}\Big[\sum_{j=1}^{M}\big(Vx(\boldsymbol{p}[n])\lambda f_j p_j[n]-Vx(\boldsymbol{p}[n])\lambda f_j\frac{\rho_j}{\gamma_j}\big)\Big|\boldsymbol{X}[n]\Big]+D, \end{aligned}\tag{27}$$

*where* $D=\sum_{j=1}^{M}\max\big(\frac{1}{2}(x_l\lambda f_j-\overline{x}_j)^2, \frac{1}{2}(x_u\lambda f_j-\overline{x}_j)^2\big)$.

See Appendix G-A in the supplementary material for the proof. Lemma 4 allows us to use a *simple-form bound* rather than the original complex drift-minus-utility in deriving the optimal price. As $\boldsymbol{X}[n]$ is a priori knowledge in each time slot $n$ and $D$ is a constant, the remaining terms in Eq. (27) precisely correspond to the overloaded constraint and the service utility, respectively.

*2) Optimal Service Price:* In every time slot $n$, we minimize the bound of drift-minus-utility with $x(\boldsymbol{p}[n])$ in Eq. (24). For simplicity, denote $\Xi(\boldsymbol{p}[n])=\sum_{j=1}^{M}\big[X_j[n](x(\boldsymbol{p}[n])\lambda f_j-\overline{x}_j)-Vx(\boldsymbol{p}[n])\lambda f_j p[n]+Vx(\boldsymbol{p}[n])\lambda f_j\frac{\rho_j}{\gamma_j}\big]$

$$\min_{\boldsymbol{p}[n]}\ \Xi(\boldsymbol{p}[n]) \tag{28}$$
$$\text{s.t. } p_j[n]\in(0,p_u],\quad \forall j\in\mathcal{M}. \tag{28a}$$

**Algorithm 1** Online Offloading And Pricing

**Input:** Processing probability $\{f_j\}$, task generation rate $\lambda$, service rates $\mu, \{\gamma_j\}$, unit cost parameters $\rho_c^m, \rho_t^m, \{\rho_j\}$
**Output:** Optimal probability and prices

```
Initialize virtual queue {X_j[0]};
for time slot n = 0, 1, ... do
    Users perform D2D collaboration;
    for server j ∈ M do
        Determine the prices based on Eq. (29);
    Users determine the offloading probability x[n] based on Eq. (24);
    for server j ∈ M do
        Update queue X_j[n+1] based on Eq. (25);
return offloading probability and service prices;
```

First, suppose that $\sum_{j=1}^M f_j p_j[n] \le \frac{\rho_c^m}{\mu^2} - \rho_t^m \sum_{j=1}^M f_j \frac{B}{r_j}$, then $x(\boldsymbol{p}[n]) = x_u$, thus the objective of Eq. (28) is $\Xi_1(\boldsymbol{p}[n]) = \sum_{j=1}^M \left[X_j[n]\left(x_u\lambda f_j - \overline{x}_j\right) - V x_u \lambda f_j p_j[n] + V x_u \lambda f_j \frac{\rho_j}{\gamma_j}\right]$. In order to minimize $\Xi_1(\boldsymbol{p}[n])$, the provider can set prices to any feasible combinations such that $\sum_{j=1}^M f_j p_j[n] = \frac{\rho_c^m}{\mu^2} - \rho_t^m \sum_{j=1}^M f_j \frac{B}{r_j}$, and we adopt $p_j[n] = \frac{\rho_c^m}{\mu^2} - \rho_t^m \sum_{j=1}^M f_j \frac{B}{r_j}, \forall j \in \mathcal{M}$ for a typical example. As a result, we have $\Xi_1(\boldsymbol{p}[n]) = \sum_{j=1}^M X_j[n](x_u\lambda f_j - \overline{x}_j) - V x_u \lambda(\frac{\rho_c^m}{\mu^2} - \rho_t^m \sum_{j=1}^M f_j \frac{B}{r_j}) + V x_u \lambda \sum_{j=1}^M f_j \frac{\rho_j}{\gamma_j}$. Following this line, assume that $\sum_{j=1}^M f_j p_j[n] > \frac{\rho_c^m}{\mu^2} - \rho_t^m \sum_{j=1}^M f_j \frac{B}{r_j}$, and the objective becomes $\Xi_2(\boldsymbol{p}[n]) = \sum_{j=1}^M \left[X_j[n](x_l\lambda f_j - \overline{x}_j) - V x_l \lambda f_j p_j[n] + V x_l \lambda f_j \frac{\rho_j}{\gamma_j}\right]$, thereby each $p_j[n], \forall j \in \mathcal{M}$ ought to be $p_u$, and also $\Xi_2(\boldsymbol{p}[n]) = \sum_{j=1}^M X_j[n](x_l\lambda f_j - \overline{x}_j) - V x_l \lambda p_u + V x_l \lambda \sum_{j=1}^M f_j \frac{\rho_j}{\gamma_j}$. In general, we compare the two values $\Xi_1(\boldsymbol{p}[n])$ and $\Xi_2(\boldsymbol{p}[n])$ to determine $\boldsymbol{p}[n]$. Let $X^* = \frac{V x_u \left(\frac{\rho_c^m}{\mu^2} - \rho_t^m \sum_{j=1}^M f_j \frac{B}{r_j}\right) - V x_l p_u}{x_u - x_l} - V \sum_{j=1}^M f_j \frac{\rho_j}{\gamma_j}$, then for each server $j$:

$$p_j[n] = \begin{cases} \dfrac{\rho_c^m}{\mu^2} - \rho_t^m \displaystyle\sum_{j=1}^M f_j \frac{B}{r_j} & \text{if } \displaystyle\sum_{j=1}^M f_j X_j[n] \le X^*, \\ p_u & \text{otherwise.} \end{cases} \tag{29}$$

Note that Eq. (29) is actually a homogeneous pricing scheme, namely the provider implements a uniform price across all edge servers. After setting $\boldsymbol{p}[n]$ abide to Eq. (29), users then make offloading decision $x[n]$ based on Eq. (24), and the queue $X_j[n]$ is updated following Eq. (25). A brief description of the online offloading the dynamic pricing is shown in Algorithm 1.

### C. Performance of Offloading and Pricing Scheme

Now we provide the performance analysis of our task offloading and service pricing. In particular, we demonstrate that an asymptotically optimal utility is obtained and each queue backlog $X_j[n]$ has a constant upper bound.

*Theorem 6: Suppose the initial queue backlog $X_j[0] = 0, \forall j \in \mathcal{M}$. For any importance weight $V > 0$, the task offloading and service pricing satisfy the following properties.*

*a) Each queue backlog $X_j[n]$ in any time slot $n$ is bounded*:

$$X_j[n] \le \frac{V x_u \left(\frac{\rho_c^m}{\mu^2} - \rho_t^m \sum_{j=1}^M f_j \frac{B}{r_j}\right) - V x_l p_u}{f_j(x_u - x_l)} - V \sum_{m=1}^M \frac{f_m}{f_j}\frac{\rho_m}{\gamma_m} + x_u \lambda f_j - \overline{x}_j. \tag{30}$$

*b) Denote $U^*$ as the optimal cumulative time average utility for DSP-VM, then the achieved utility satisfies*:

$$\lim_{T\to\infty} \frac{1}{T} \sum_{n=0}^{T-1} \sum_{j=1}^M \mathbb{E}[u_j[n]] \ge U^* - \frac{D}{V}. \tag{31}$$

Please refer to Appendix G-B in the supplementary material for the proof. Theorem 6 unveils an $(O(V), O(1/V))$ tradeoff between the queue backlog and the service utility. Specifically, as the importance weight $V$ increases, the queue backlog also increases as fast as the order of $O(V)$, while the time average utility approaches the theoretical optimum within an $O(1/V)$ gap. In addition, because each $X_j[n]$ is upper bounded by a finite value given in Eq. (30), and then $X_j[n]$ is rate stable, that is the overloaded constraint will hold asymptotically.

## VII. Online Offloading And Pricing of Server Fairness Maximization

In the previous section, we design a pricing and offloading scheme to optimize the utilitarian utility, which however will lead to unbalanced servers' utilities. To harmonize the interests of all edge servers, we continue to explore the proportional fairness to ensure a fair utility distribution among them [30], [36]. Accordingly, the objective of DSP-VM in Eq. (8) is specified as $\sum_{j=1}^M g([u_j[n]) = \lim_{T\to\infty} \sum_{j=1}^M \log\big(1 + \beta \frac{1}{T} \sum_{n=0}^{T-1} \mathbb{E}[u_j[n]]\big)$.

### A. Fair Offloading And Pricing

*1) Fair Task Offloading:* We first illustrate the offloading decision of mobile users, given that the provider has announced prices $\boldsymbol{p}[n]$. It can be verified that the offloading probability $x[n]$ is also determined following the threshold based rule described in Eq. (24), where $x_u, x_l$ still denote the boundary points of the feasible region derived from delay and fairness constraints in Eqs. (7a)-(7b).

*2) Lyapunov Based Fairness Optimization:* The decision of the provider is to ascertain the optimal prices $\boldsymbol{p}[n]$ for fairness maximization. To achieve this goal, we also rely on Lyapunov optimization to design the fair pricing scheme.

*a) Auxiliary variable:* Note that the objective in Eq. (8) is a logarithm function of time average utility, which is hard to be optimized directly. To bypass this problem, one feasible solution is to transform the objective into a time average logarithm function, and this can be accomplished by introducing auxiliary variables $\zeta_j[n], j \in \mathcal{M}$. Because logarithm function $g(\cdot)$ is concave, then Jensen's inequality

yields that:

$$\frac{1}{T}\sum_{n=0}^{T-1} g(\zeta_j[n]) \le g\Big(\frac{1}{T}\sum_{n=0}^{T-1}\zeta_j[n]\Big),$$
$$\frac{1}{T}\sum_{n=0}^{T-1} \mathbb{E}[g(\zeta_j[n])] \le g\Big(\frac{1}{T}\sum_{n=0}^{T-1}\mathbb{E}[\zeta_j[n]]\Big). \quad (32)$$

Considering that $\zeta_j[n]$ is used to replace the utility $u_j[n]$, it should fulfill $\zeta_j[n] \le u_j^{\max} = x_u\lambda f_j p_u - x_u\lambda f_j\frac{\rho_j}{\gamma_j}$ from Eq. (6). According to [38], instead of solving the original DSP-VM of fairness maximization, we deal with its transformed problem as follows:

$$\max_{\boldsymbol{p}[n],\boldsymbol{\zeta}[n]} \quad \lim_{T\to\infty}\frac{1}{T}\sum_{n=0}^{T-1}\sum_{j=1}^{M}\mathbb{E}[\log(1+\beta\zeta_j[n])] \quad (33)$$

$$\text{s.t.} \lim_{T\to\infty}\frac{1}{T}\sum_{n=0}^{T-1}\mathbb{E}[x[n]\lambda f_j] \le \overline{x}_j, \quad \forall j\in\mathcal{M} \quad (33a)$$
$$p_j[n]\in(0,p_u], \quad \forall j\in\mathcal{M} \quad (33b)$$
$$\overline{\zeta}_j \le \overline{u}_j, \quad \forall j\in\mathcal{M} \quad (33c)$$
$$0\le\zeta_j[n]\le x_u\lambda f_j p_u - x_u\lambda f_j\frac{\rho_j}{\gamma_j}, \quad \forall j\in\mathcal{M}. \quad (33d)$$

Aside from the original constraints, two more constraints Eqs. (33c)-(33d) are introduced for the auxiliary variables.

**Drift-minus-fairness**. Regarding the transformed problem, there are two types of time average constraints, the overloaded constraint Eq. (33a) and the auxiliary constraint Eq. (33c). Consistently, we need to maintain two different virtual queues $\boldsymbol{\Theta}[n] = \{\boldsymbol{X}[n], \boldsymbol{G}[n]\}$, in which $\boldsymbol{X}[n]$ corresponds to the overloaded constraint and $\boldsymbol{G}[n]$ is for the auxiliary constraint. For each $X_j[n], \forall j\in\mathcal{M}$, the queue dynamics still follows Eq. (25). As for the auxiliary queue $G_j[n], \forall j\in\mathcal{M}$, its updating rule obeys:

$$G_j[n+1] = \max(G_j[n]+\zeta_j[n]-u_j[n], 0), \quad (34)$$

where $u_j[n]$ is the utility of server $j$ in Eq. (6).

With $\boldsymbol{\Theta}[n]$, we construct Lyapunov function $L(\boldsymbol{\Theta}[n]) = \sum_{j=1}^{M}\frac{1}{2}X_j^2[n] + \sum_{j=1}^{M}\frac{1}{2}G_j^2[n]$. Analogously, the Lyapunov drift is the change of Lyaunov function:

$$\begin{aligned}\Delta(\boldsymbol{\Theta}[n]) &= \mathbb{E}[L(\boldsymbol{\Theta}[n+1]) - L(\boldsymbol{\Theta}[n])|\boldsymbol{\Theta}[n]]\\ &= \mathbb{E}\Big[\sum_{j=1}^{M}\frac{1}{2}(X_j^2[n+1]-X_j^2[n]\\ &\quad + G_j^2[n+1]-G_j^2[n])\big|\boldsymbol{\Theta}[n]\Big]. \end{aligned} \quad (35)$$

To entail the fairness objective in Eq. (33), we define the drift-minus-fairness $\Delta(\boldsymbol{\Theta}[n]) - V\sum_{j=1}^{M}\mathbb{E}[\log(1+\beta\zeta_j[n])|\boldsymbol{\Theta}[n]]$, which is similar to the drift-minus-utility with $V$ being the importance weight. Optimizing the drift-minus-fairness can *balance the queue minimization and fairness maximization*, so that both overloaded/auxiliary constraint and favorable utility can be guaranteed.

Since directly handling the complex drift-minus-fairness is difficult, a more viable way is to harness a concise bound and minimizing the bound instead. For consistency, we will express the offloading decision $x[n]$ as $x(\boldsymbol{p}[n])$ as well.

*Lemma 5: Drift-minus-fairness* $\Delta(\boldsymbol{\Theta}[n]) - V\sum_{j=1}^{M}\mathbb{E}[\log(1+\beta\zeta_j[n])|\boldsymbol{\Theta}[n]]$ *satisfies the following condition*:

$$\begin{aligned}&\Delta(\boldsymbol{\Theta}[n]) - V\sum_{j=1}^{M}\mathbb{E}[\log(1+\beta\zeta_j[n])|\boldsymbol{\Theta}[n]]\\ &\le \mathbb{E}\Big[\sum_{j=1}^{M}[G_j[n]\zeta_j[n] - V\log(1+\beta\zeta_j[n])]|\boldsymbol{\Theta}[n]\Big]\\ &\quad - \mathbb{E}\Big[\sum_{j=1}^{M}G_j[n]\big(x(\boldsymbol{p}[n])\lambda f_j p_j[n] - x(\boldsymbol{p}[n])\lambda f_j\frac{\rho_j}{\gamma_j}\big)\big|\boldsymbol{\Theta}[n]\Big]\\ &\quad + \mathbb{E}\Big[\sum_{j=1}^{M}X_j[n](x(\boldsymbol{p}[n])\lambda f_j - \overline{x}_j)|\boldsymbol{\Theta}[n]\Big] + D', \end{aligned} \quad (36)$$

*where* $D' = \sum_{j=1}^{M}\max\left(\frac{1}{2}(x_l\lambda f_j - \overline{x}_j)^2, \frac{1}{2}(x_u\lambda f_j - \overline{x}_j)^2\right) + \sum_{j=1}^{M}\frac{1}{2}\left(x_u\lambda f_j p_u - x_u\lambda f_j\frac{\rho_j}{\gamma_j}\right)^2$.

Please refer to Appendix G-C in the supplementary material for the detailed proof. The first term in Eq. (36) is altered by the auxiliary variable $\zeta_j[n]$, while the second and third terms are the utility $u_j[n]$ and overloaded constraint, respectively, which are controlled by prices $\boldsymbol{p}[n]$, and the last term is a constant.

*3) Fair Service Pricing:* Now we illustrate the fair service pricing to minimize the bound of the drift-minus-fairness.

**Optimal auxiliary variable**. First of all, we need to determine each auxiliary variable $\zeta_j[n], \forall j\in\mathcal{M}$ which is achieved by solving the following problem.

$$\min_{\zeta_j[n]} \ G_j[n]\zeta_j[n] - V\log(1+\beta\zeta_j[n]) \quad (37)$$
$$\text{s.t.} \ \ 0\le\zeta_j[n]\le x_u\lambda f_j p_u - x_u\lambda f_j\frac{\rho_j}{\gamma_j}. \quad (37a)$$

Denote $h(\zeta_j[n]) = G_j[n]\zeta_j[n] - V\log(1+\beta\zeta_j[n])$, and derive $h(\zeta_j[n])$ over $\zeta_j[n])$ to obtain:

$$h'(\zeta_j[n]) = G_j[n] - \frac{V\beta}{1+\beta\zeta_j[n]}. \quad (38)$$

Let $h'(\zeta_j[n]) = 0$, and then $\zeta_j[n] = \frac{V}{G_j[n]} - \frac{1}{\beta}$. Still use $u_j^{\max} = x_u\lambda f_j p_u - x_u\lambda f_j\frac{\rho_j}{\gamma_j}$. Jointly considering the constraint Eq. (37a), we have the following result:

$$\zeta_j[n] = \begin{cases} 0 \text{ if } G_j[n] > V\beta,\\ \dfrac{V}{G_j[n]} - \dfrac{1}{\beta} \text{ if } G_j[n]\in\left[\dfrac{V\beta}{1+\beta u_j^{\max}}, V\beta\right),\\ u_j^{\max} \text{ if } G_j[n] < \dfrac{V\beta}{1+\beta u_j^{\max}}. \end{cases} \quad (39)$$

**Fair service price**. After computing the auxiliary variable, we then derive the optimal prices $\boldsymbol{p}[n]$ to minimize the rest terms in Eq. (36). Given $\boldsymbol{p}[n]$, the reacted offloading decision $x[n]$ is made based on Eq. (24), which is the same as that in utility maximization. Therefore, the provider can in turn deduce the optimal prices in each time slot $n$. Denote $\Upsilon(\boldsymbol{p}[n]) = \sum_{j=1}^{M}\big[X_j[n](x(\boldsymbol{p}[n])\lambda f_j - \overline{x}_j) - G_j[n]x(\boldsymbol{p}[n])\lambda f_j p_j[n] + G_j[n]x(\boldsymbol{p}[n])\lambda f_j\frac{\rho_j}{\gamma_j}\big]$, so that we can solve the problem below to attain $\boldsymbol{p}[n]$.

$$\min_{\boldsymbol{p}[n]} \ \Upsilon(\boldsymbol{p}[n]) \quad (40)$$

$$\text{s.t.}\quad p_j[n] \in (0, p_u], \forall j \in \mathcal{M}. \tag{40a}$$

From Eq. (24), if $\sum_{j=1}^{M} f_j p_j[n] \leq \frac{\rho_c^m}{\mu^2} - \rho_t^m \sum_{j=1}^{M} f_j \frac{B}{r_j}$, then the offloading decision is $x[n] = x_u$. As a result, we acquire $\Upsilon_1(\boldsymbol{p}[n]) = \sum_{j=1}^{M} \left[X_j[n](x_u\lambda f_j - \overline{x}_j) - G_j[n]x_u\lambda f_j p_j[n] + G_j[n]x_u\lambda f_j \frac{\rho_j}{\gamma_j}\right]$. Let $\hat{p}_j = \min\left(\frac{\rho_c^m}{\mu^2 f_j} - \frac{\rho_t^m}{f_j}\sum_{m=1}^{M} f_m \frac{B}{r_m}, p_u\right)$, and initialize a set $\mathcal{M}_1$ to $\emptyset$. In order to minimize $\Upsilon_1(\boldsymbol{p}[n])$, we iteratively pick the edge server without replacement, which has the highest queue value $G_j[n]$, and add it into $\mathcal{M}_1$. The iteration terminates when picking a server, say server $j^*$, the value $\sum_{j\in\mathcal{M}_1} f_j\hat{p}_j + f_{j^*}\hat{p}_{j^*}$ first exceeds $\frac{\rho_c^m}{\mu^2} - \rho_t^m \sum_{m=1}^{M} f_m \frac{B}{r_m}$. The prices are set in this way: 1) $p_j[n] = \hat{p}_j, \forall j \in \mathcal{M}_1$; 2) $p_{j^*}[n] = \frac{\rho_c^m}{\mu^2 f_{j^*}} - \frac{\rho_t^m}{f_{j^*}}\sum_{m=1}^{M} f_m \frac{B}{r_m} - \frac{1}{f_{j^*}}\sum_{j\in\mathcal{M}_1} f_j\hat{p}_j$; 3) $p_j[n] = 0, \forall j \in \mathcal{M}\backslash(\mathcal{M}_1 \cup \{j^*\})$.

As for the case $\sum_{j=1}^{M} f_j p_j[n] > \frac{\rho_c^m}{\mu^2} - \rho_t^m \sum_{j=1}^{M} f_j \frac{B}{r_j}$, the offloading decision changes to $x[n] = x_l$, and hence $\Upsilon_2(\boldsymbol{p}[n]) = \sum_{j=1}^{M} \left[X_j[n](x_l\lambda f_j - \overline{x}_j) - G_j[n]x_l\lambda f_j p_j[n] + G_j[n]x_l\lambda f_j \frac{\rho_j}{\gamma_j}\right]$. Intuitively, we should set the prices $p_j[n] = p_u, \forall j \in \mathcal{M}$ to minimize $\Upsilon_2(\boldsymbol{p}[n])$. Finally, we compare values $\Upsilon_1(\boldsymbol{p}[n])$ and $\Upsilon_2(\boldsymbol{p}[n])$ to see which one is smaller, and then decide the prices $\boldsymbol{p}[n]$ accordingly.

After determining the auxiliary variables and service prices, the offloading decision is made and queues $\boldsymbol{X}[n], \boldsymbol{G}[n]$ are updated. The walk-through of online offloading and dynamic pricing considering fairness issue is similar to Algorithm 1, and hence we omit it for brevity.

### B. Performance of Fair Offloading and Pricing Scheme

We analyze the performance of fair offloading and pricing scheme in the sequel.

*Theorem 7: Suppose the initial queue backlog $X_j[0] = 0, G_j[0] = 0, \forall j \in \mathcal{M}$. For any importance weight $V > 0$, the proposed fair task offloading and service pricing satisfy the following properties.*

*a) The backlog of auxiliary queue $G_j[n]$ in any time slot $n$ satisfies $G_j[n] \leq G_j^{\max}$, where $G_j^{\max}$ is*:

$$G_j^{\max} = \max\left(V\beta, \frac{V\beta}{1+\beta u_j^{\max}} + u_j^{\max}\right). \tag{41}$$

*b) The queue backlog $X_j[n]$ in any time slot $n$ is bounded*:

$$X_j[n] \leq \frac{G^{\max}x_u\left(\frac{\rho_c^m}{\mu^2} - \rho_t^m\sum_{j=1}^{M} f_j\frac{B}{r_j}\right) - G^{\max}x_l p_u}{f_j(x_u - x_l)} - G^{\max}\sum_{m=1}^{M}\frac{f_m}{f_j}\frac{\rho_m}{\gamma_m} + x_u\lambda f_j - \overline{x}_j, \tag{42}$$

*where $G^{\max} = \max_j(G_j^{\max})$.*

*c) Denote $U'$ as the optimal time average fairness for DSP-VM, then the achieved fairness satisfies*:

$$\sum_{j=0}^{M} g(\overline{u}_j) \geq U' - \frac{D'}{V}. \tag{43}$$

See Appendix G-D in the supplementary material for the proof. Theorem 7 reveals that the overloaded constraint holds as $X_j[n]$ is capped, and also an asymptotically optimal proportional fairness is obtained. In performance evaluations, we will explicitly show that the gap between maximum and minimum utilities among all edge servers is reduced due to the fairness consideration.

## VIII. Performance Evaluation

In this section, we perform evaluations to illustrate D2D collaboration among users and task offloading to edge servers.

### A. Parameter Setting

Suppose that the provider manages $M = 4$ edge servers. In line with the real measurements [27], we set the average required computation of a task to 1000 Megacycles which is normalized to 1 as stated in the system model, and the average data size $B$ to 2000 KB. Service rates of a mobile device and edge servers are 1 GHz and $\{4, 5, 5, 6\}$ GHz, respectively, thus $\mu = \frac{10^9}{1000*10^6} = 1$ and $\gamma_1 = \frac{4*10^9}{1000*10^6} = 4$, $\gamma_2 = 5$, $\gamma_3 = 5$, $\gamma_4 = 6$ accordingly. Moreover, the typical real-world data rate of 4G cellular uplink is around 10 Mbps [41], so we let $r_j$ randomly sampled in $[9, 11]$ Mbps.

For D2D collaboration, task generation rate $\lambda$ is set to 0.9 for modeling a heavy workload. User's (expected) degree in (dynamic) static graph is uniformly distributed in $\mathcal{K} = \{6, 7, 8, 9\}$, that is $\Pr(k) = \frac{1}{4}, \forall k \in \mathcal{K}$ and $k_{\min} = 6$, $k_{\max} = 9$. Besides, user's realized degree given its expected degree obeys a Poisson distribution for dynamic graph [35], and the graph structure varies over time with rate 1.

Regarding task offloading, let $\overline{d} = 1.6$ for the delay constraint, and $\overline{s} = 0.06$ for the collaboration fairness constraint. The per energy costs for mobile devices are $(\rho_c^m, \rho_t^m) = (0.9, 0.3)$, which correspond to 900 mW, 300 mW for processing and transmitting tasks. The per energy cost for edge server $j$ is in $\rho_j \in [1, 1.2], \forall j \in \mathcal{M}$, namely 1000 mW to 1200 mW for processing tasks in different servers. The probability $f_j$, tasks are offloaded to server $j$, is sampled in $[0.15, 0.4]$ with $\sum_{j\in\mathcal{M}} f_j = 1$. The overloaded threshold $\overline{x}_j, j \in \mathcal{M}$ is set to $[0.1, 0.24]$ where $\overline{x}_j$ satisfies $\overline{x}_j \in [x_l\lambda f_j, x_u\lambda f_j]$, and the highest acceptable price $p_u$ is assigned to 0.5.

### B. Mean Field D2D Collaboration

**Stationary point**. We first demonstrate that the mean field model on graph is effective to characterize D2D collaboration by comparing the theoretical stationary point $\boldsymbol{s}^*$ obtained from the ODE system of Eq. (10) and that from simulating the Po2 choices. In particular, we compute the theoretical stationary point $\boldsymbol{s}^*$ using *scipy.integrate.odeint* in Python to solve the ODE system, since $\boldsymbol{s}(t)$ will converge to $\boldsymbol{s}^*$ when time $t$ is large enough. On the other hand, the simulated MEC system consists of 800 users for static graph and 1000 users for dynamic graph. The static graph is generated by the configuration model [22] with both self-loops and multiple edges between two users being cut off to obtain an uncorrelated D2D network. Similarly, the configuration model is leveraged again when the graph structure changes to produce dynamic graph. Varying the value $x_c\lambda$, namely the proportion of tasks processed via D2D collaboration, from 0.1 to 0.9 with an

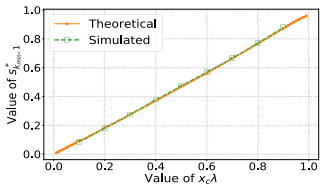


Fig. 5. Stationary point on static graph.

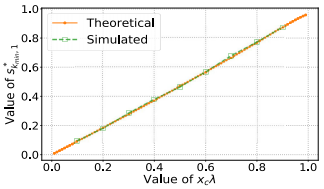


Fig. 6. Stationary point on dynamic graph.

TABLE II
THEORETICAL AND SIMULATED STATIONARY POINTS

| $s^*_{k,i}$ | Theoretical | Static graph | Dynamic graph | Max error |
|---|---|---|---|---|
| $s^*_{6,1}$ | 0.66504 | 0.66706 | 0.67742 | 0.01238 |
| $s^*_{7,1}$ | 0.68972 | 0.68924 | 0.696 | 0.00628 |
| $s^*_{8,1}$ | 0.7123 | 0.71732 | 0.71786 | 0.00556 |
| $s^*_{9,1}$ | 0.73295 | 0.73329 | 0.7325 | 0.00039 |
| $s^*_{6,2}$ | 0.30585 | 0.31042 | 0.32056 | 0.01471 |
| $s^*_{7,2}$ | 0.33239 | 0.34009 | 0.3492 | 0.01681 |
| $s^*_{8,2}$ | 0.35814 | 0.35022 | 0.36905 | 0.00792 |
| $s^*_{9,2}$ | 0.38302 | 0.36972 | 0.3852 | 0.0133 |

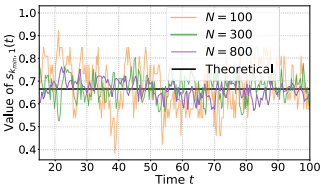


Fig. 7. Convergence to mean field on static graph.

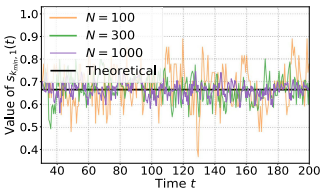


Fig. 8. Convergence to mean field on dynamic graph.

increment of 0.1 each time, we run the simulated Po2 for eight times under each $x_c\lambda$. Figs. 5-6 exhibit the values of theoretical $s^*_{k_{\min},1}$ and averaged simulated $s^*_{k_{\min},1}$ on static and dynamic graphs, respectively, which tell that theoretical results perfectly match with simulated results. Therefore, the mean field model is effective in analyzing D2D collaboration.

Furthermore, we show each averaged simulated $s^*_{k,i}$ and theoretical $s^*_{k,i}$ when $x_c\lambda = 0.7$ in Table II. Results validate the accuracy of mean field model as the error between theoretical and simulated $s^*_{k,i}$ on static/dynamic graph is negligible.[1]

[1]There are evidences showing that the mean field approximation error for $N$-user system is $\mathcal{O}(\frac{1}{N})$ [39], [40]. However, whether this result holds on graph structure will be left as our future work.

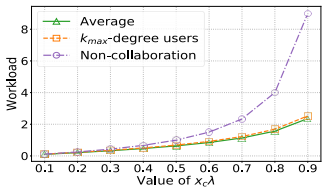


Fig. 9. Workload comparison.

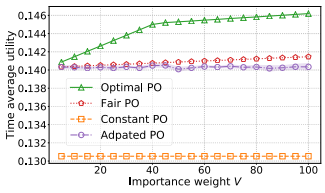


Fig. 10. Utility vs. $V$.

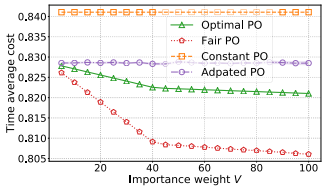


Fig. 11. Cost vs. $V$.

**User number influence**. We then discuss how the simulated Po2 behaves over time for different number of users $N$. In particular, we display the evolution of $s_{k_{\min},1}(t)$ when $N = 100, 300, 800$ for static graph and $N = 100, 300, 1000$ for dynamic graph in Figs. 7 and 8, respectively. We can see that as $N$ increases, simulated results tend to approach the theoretical stationary point with impaired variances.

**Power of collaboration**. Theorem 4 reveals that largest-degree users have heaviest workloads. We now show that, even for those users, their workloads are still effectively mitigated in D2D collaboration, compared to the non-collaborative case which is a M/M/1 queue. Fig. 9 depicts the average workload and workloads for the largest-degree users as well as the non-collaborative case when value of $x_c\lambda$ varies. We can observe that the average and heaviest workloads are much smaller than the non-collaborative scenario, with the workload being mitigated by 73.8% when $x_c\lambda = 0.9$. Hence, the task delay is also significantly reduced as a result of the collaboration.

### C. Lyapunov Based Offloading and Pricing

**Critical points**. We first derive the critical points for task offloading. As for the delay constraint, we implement golden section search to numerically compute critical points $x^*_l, x^*_u$ by using Eq. (23), and hence we obtain $x^*_l = 0.2573$, $x^*_u = 0.7773$. Similarly, golden section search is leveraged to calculate critical points $x'_l, x'_u$ for the fairness constraint, and their values are 0.08505 and 0.49953, respectively. Overall, the feasible region for the offloading decision is $[0.49953, 0.7773]$, namely $x_l = 0.49953, x_u = 0.7773$.

**Utility and cost**. Now we compare the performance of Lyapunov based utility/fairness optimization (Optimal PO/Fair PO) with two baseline methods.

- Constant PO: provider chooses Prices $p_j[n] = p_u, \forall j \in \mathcal{M}$ and users react with the Offloading probability $x_l$.

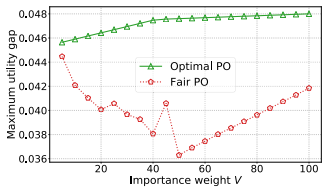


Fig. 12. Utility gap.

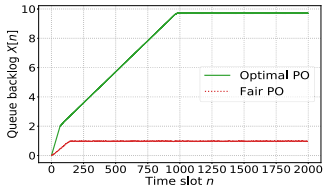


Fig. 13. Queue backlog $X[n]$.

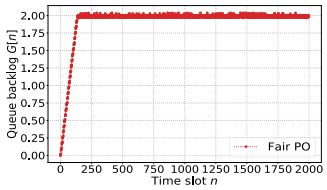


Fig. 14. Queue backlog $G[n]$.

- Adapted PO: based on Eq. (29), provider chooses Prices $p_j[n] = \frac{\rho_c^m}{\mu^2} - \rho_t^m \sum_{j=1}^{M} f_j \frac{B}{r_j}, \forall j \in \mathcal{M}$ or $p_j[n] = p_u, \forall j \in \mathcal{M}$ with probability $\min_j \frac{\overline{x}_j - x_l \lambda f_j}{x_u \lambda f_j - x_l \lambda f_j}$ or $1 - \min_j \frac{\overline{x}_j - x_l \lambda f_j}{x_u \lambda f_j - x_l \lambda f_j}$, and users react with the Offloading probability $x_u$ or $x_l$.

It can be verified that the overloaded constraint also holds for these baseline methods. Let the total time slot $T = 2000$, and we vary the importance weight $V$ from 5 to 100 to obtain the corresponding service utility and system cost, respectively. Fig. 10 displays the time average utility, which shows that the utility will increase over the importance weight $V$ for Optimal PO and Fair PO as more emphasis is on the utility term. Besides, Optimal PO and Fair PO can achieve higher utilities compared to Constant PO and Adapted PO. We then exhibit the time average system cost in Fig. 11, and we can see that Optimal PO and Fair PO lead to lower costs than the baseline methods. With the increase of $V$, the provider is more likely to set lower prices, that is why the system cost will decrease for Optimal PO and Fair PO. Therefore, Optimal PO and Fair PO can achieve high energy efficiency as the system cost mainly counts the energy consumption.

**Utility fairness**. Only maximizing servers' cumulative utility will cause an unfair utility distribution. To verify this point, we display the maximum gap between servers' utilities for Optimal PO and Fair PO in Fig. 12. We can observe that Fair PO has smaller gaps compared to Optimal PO raised from the fairness consideration, i.e., Fair PO can achieve a more balanced service utility. Combining with Fig. 10, Fair PO improves fairness (gap reduction) by $15.87\%$ with only $2.55\%$ utility decrease on average.

**Queue backlog**. Lastly, we depict the queue backlog $X_1[n]$ for Optimal PO and Fair PO in Fig. 13 when the importance weight is $V = 20$. Results demonstrate that the queue backlog is finite, that is the overloaded constraint of DSP-VM is satisfied. The backlog of auxiliary queue $G_1[n]$ is shown in Fig. 14, and it will not go to infinity either. The asymptotic stability of both queues is mainly because the arrival rates will approximately equal to the departure rate after a certain time slots, further implying the queue backlogs are bounded.

## IX. Conclusion

In this paper, we develop a joint D2D collaboration and offloading pricing framework for MEC systems with a large population of mobile users and geo-distributed edge servers. Specifically, to characterize the state evolution of D2D collaboration, we propose a mean field model to analyze the stochastic MEC system by a deterministic ODE system. On this basis, we derive the existence and uniqueness of the stationary point, and further show the global convergence of state evolution to this unique stationary point. By incorporating D2D collaboration, we then design a pricing scheme for task offloading to multiple servers following a Lyapunov optimization framework. In particular, the offloading interaction between users and servers is modeled as a Stackelberg game, where the provider is the leader to determine reasonable prices for service utility and fairness maximization, and users are followers to make the offloading decision for system cost minimization. Extensive evaluations validate the effectiveness of our mean field model and the superiority of Lyapunov optimization.

## References

[1] M. Satyanarayanan, "The emergence of edge computing," *Computer*, vol. 50, no. 1, pp. 30–39, Jan. 2017.

[2] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge computing: Vision and challenges," *IEEE Internet Things J.*, vol. 3, no. 5, pp. 637–646, Oct. 2016.

[3] R. Kelly. (2020). *Internet of Things Data to Top 1.6 Zettabytes by 2020*. [Online]. Available: https://campustechnology.com/articles/2015/04/15/internet-of-things-data-to-top-1-6-zettabytes-by-2020.aspx

[4] X. Chen, L. Pu, L. Gao, W. Wu, and D. Wu, "Exploiting massive D2D collaboration for energy-efficient mobile edge computing," *IEEE Wireless Commun.*, vol. 24, no. 4, pp. 64–71, Aug. 2017.

[5] A. Asadi, Q. Wang, and V. Mancuso, "A survey on device-to-device communication in cellular networks," *IEEE Commun. Surveys Tuts.*, vol. 16, no. 4, pp. 1801–1819, 4th Quart., 2014.

[6] L. Pu, X. Chen, J. Xu, and X. Fu, "D2D fogging: An energy-efficient and incentive-aware task offloading framework via network-assisted D2D collaboration," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 12, pp. 3887–3901, Dec. 2016.

[7] Z. Zhao, W. Zhou, D. Deng, J. Xia, and L. Fan, "Intelligent mobile edge computing with pricing in Internet of Things," *IEEE Access*, vol. 8, pp. 37727–37735, 2020.

[8] Y. Li, H. C. Ng, L. Zhang, and B. Li, "Online cooperative resource allocation at the edge: A privacy-preserving approach," in *Proc. IEEE 28th Int. Conf. Netw. Protocols (ICNP)*, Oct. 2020, pp. 1–11.

[9] H. Xing, L. Liu, J. Xu, and A. Nallanathan, "Joint task assignment and resource allocation for D2D-enabled mobile-edge computing," *IEEE Trans. Commun.*, vol. 67, no. 6, pp. 4193–4207, Jun. 2019.

[10] X. Chen, Z. Zhou, W. Wu, D. Wu, and J. Zhang, "Socially-motivated cooperative mobile edge computing," *IEEE Netw.*, vol. 32, no. 6, pp. 177–183, Nov. 2018.

[11] Y. He, J. Ren, G. Yu, and Y. Cai, "D2D communications meet mobile edge computing for enhanced computation capacity in cellular networks," *IEEE Trans. Wireless Commun.*, vol. 18, no. 3, pp. 1750–1763, Mar. 2019.

[12] Z. Ning, P. Dong, X. Kong, and F. Xia, "A cooperative partial computation offloading scheme for mobile edge computing enabled Internet of Things," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 4804–4814, Jun. 2019.

[13] Y. Li, X. Wang, X. Gan, H. Jin, L. Fu, and X. Wang, "Learning-aided computation offloading for trusted collaborative mobile edge computing," *IEEE Trans. Mobile Comput.*, vol. 19, no. 12, pp. 2833–2849, Dec. 2020.
[14] J.-M. Lasry and P.-L. Lions, "Mean field games," *Jpn. J. Math.*, vol. 2, no. 1, pp. 229–260, 2007.
[15] D. Narasimha, S. Shakkottai, and L. Ying, "A mean field game analysis of distributed MAC in ultra-dense multichannel wireless networks," in *Proc. 20th ACM Int. Symp. Mobile Ad Hoc Netw. Comput.*, Jul. 2019, pp. 1–10.
[16] J. Li, R. Bhattacharyya, S. Paul, S. Shakkottai, and V. Subramanian, "Incentivizing sharing in realtime D2D streaming networks: A mean field game perspective," *IEEE/ACM Trans. Netw.*, vol. 25, no. 1, pp. 3–17, Feb. 2017.
[17] R. Zheng, H. Wang, M. De Mari, M. Cui, X. Chu, and T. Q. S. Quek, "Dynamic computation offloading in ultra-dense networks based on mean field games," *IEEE Trans. Wireless Commun.*, vol. 20, no. 10, pp. 6551–6565, Oct. 2021.
[18] M. Mitzenmacher, "The power of two choices in randomized load balancing," *IEEE Trans. Parallel Distrib. Syst.*, vol. 12, no. 10, pp. 1094–1104, Oct. 2001.
[19] N. Gast, "The power of two choices on graphs: The pair-approximation is accurate?" *ACM SIGMETRICS Perform. Eval. Rev.*, vol. 43, no. 2, pp. 69–71, Sep. 2015.
[20] A. Budhiraja, D. Mukherjee, and R. Wu, "Supermarket model on graphs," 2017, *arXiv:1712.07607*.
[21] P. E. Caines and M. Huang, "Graphon mean field games and the GMFG equations: $\epsilon$-Nash equilibria," in *Proc. IEEE 58th Conf. Decis. Control (CDC)*, Dec. 2019, pp. 286–292.
[22] R. Pastor-Satorras, C. Castellano, P. Van Mieghem, and A. Vespignani, "Epidemic processes in complex networks," *Rev. Mod. Phys.*, vol. 87, no. 3, pp. 925–979, Aug. 2015.
[23] M. J. Farooq and Q. Zhu, "Modeling, analysis, and mitigation of dynamic botnet formation in wireless IoT networks," *IEEE Trans. Inf. Forensics Security*, vol. 14, no. 9, pp. 2412–2426, Sep. 2019.
[24] L. Zhang, W. Wu, and D. Wang, "Time dependent pricing in wireless data networks: Flat-rate vs. usage-based schemes," in *Proc. IEEE Conf. Comput. Commun.*, Apr. 2014, pp. 700–708.
[25] Y. Chen, Z. Li, B. Yang, K. Nai, and K. Li, "A Stackelberg game approach to multiple resources allocation and pricing in mobile edge computing," *Future Gener. Comput. Syst.*, vol. 108, pp. 273–287, Jul. 2020.
[26] A. Asheralieva and D. Niyato, "Combining contract theory and Lyapunov optimization for content sharing with edge caching and device-to-device communications," *IEEE/ACM Trans. Netw.*, vol. 28, no. 3, pp. 1213–1226, Jun. 2020.
[27] J. Kwak, Y. Kim, J. Lee, and S. Chong, "DREAM: Dynamic resource and task allocation for energy minimization in mobile cloud systems," *IEEE J. Sel. Areas Commun.*, vol. 33, no. 12, pp. 2510–2523, Dec. 2015.
[28] L. Chen, S. Zhou, and J. Xu, "Computation peer offloading for energy-constrained mobile edge computing in small-cell networks," *IEEE/ACM Trans. Netw.*, vol. 26, no. 4, pp. 1619–1632, Aug. 2018.
[29] V. Cardellini et al., "A game-theoretic approach to computation offloading in mobile cloud computing," *Math. Program.*, vol. 157, no. 2, pp. 421–449, Jun. 2016.
[30] X. Wang, R. Jia, X. Tian, X. Gan, L. Fu, and X. Wang, "Location-aware crowdsensing: Dynamic task assignment and truth inference," *IEEE Trans. Mobile Comput.*, vol. 19, no. 2, pp. 362–375, Feb. 2020.
[31] M. Benaïm and J.-Y. Le Boudec, "A class of mean field interaction models for computer and communication systems," *Perform. Eval.*, vol. 65, nos. 11–12, pp. 823–838, Nov. 2008.
[32] M. Bramson, Y. Lu, and B. Prabhakar, "Asymptotic independence of queues under randomized load balancing," *Queueing Syst.*, vol. 71, no. 3, pp. 247–292, Jul. 2012.
[33] A. S. Sznitman, "Topics in propagation of chaos," *Ecole D'etéde Probabilités Saint-Flour*, vol. 1464, pp. 165–251, Jan. 1991.
[34] A. Casteigts, P. Flocchini, W. Quattrociocchi, and N. Santoro, "Time-varying graphs and dynamic networks," *Int. J. Parallel, Emergent Distrib. Syst.*, vol. 27, no. 5, p. 3878, 2012.
[35] J. C. Lang, H. De Sterck, J. L. Kaiser, J. C. Miller, and J. Gleeson, "Analytic models for SIR disease spread on random spatial networks," *J. Complex Netw.*, vol. 6, no. 6, pp. 948–970, Dec. 2018.
[36] F. Kelly, "Charging and rate control for elastic traffic," *Eur. Trans. Telecommun.*, vol. 8, no. 1, pp. 33–37, Jan. 1997.
[37] T. G. Kurtz, "Strong approximation theorems for density dependent Markov chains," *Stochastic Processes Appl.*, vol. 6, no. 3, pp. 223–240, Feb. 1978.
[38] M. J. Neely, *Stochastic Network Optimization with Application to Communication and Queueing Systems*. San Rafael, CA, USA: Morgan & Claypool, 2010.
[39] L. Ying, "On the approximation error of mean-field models," in *Proc. ACM SIGMETRICS Int. Conf. Meas. Modeling Comput. Sci.*, Jun. 2016, pp. 285–297.
[40] N. Gast and B. Van Houdt, "A refined mean field approximation," in *Proc. Abstr. ACM Int. Conf. Meas. Model. Comput. Syst.*, Jun. 2018, pp. 1–28.
[41] 4G4U. *4G Speed Tests*. Accessed: May 19, 2023. [Online]. Available: https://www.4g4u.org/4g-speed-tests
[42] X. Wang, J. Ye, and J. C. S. Lui, "Joint D2D collaboration and task offloading for edge computing: A mean field graph approach," in *Proc. IEEE/ACM 29th Int. Symp. Quality Service (IWQOS)*, Jun. 2021, pp. 1–10.

**Xiong Wang** (Member, IEEE) received the B.E. degree in electronic information engineering from the Huazhong University of Science and Technology, Wuhan, China, in 2014, and the Ph.D. degree in electronic engineering from Shanghai Jiao Tong University, Shanghai, China, in 2019. He was a Post-Doctoral Fellow with the Department of Computer Science and Engineering, The Chinese University of Hong Kong, Hong Kong, China, from 2019 to 2021. He is currently an Associate Professor with the School of Computer Science and Technology, Huazhong University of Science and Technology. His research interests include distributed machine learning systems, federated learning, network flow control, mean field analysis, and cloud/edge computing. He is a member of ACM.

**Jiancheng Ye** (Senior Member, IEEE) received the B.E. degree in network engineering from Sun Yat-sen University, Guangzhou, China, in 2008, the M.Phil. degree in computer science and engineering from The Hong Kong University of Science and Technology, Hong Kong, in 2011, and the Ph.D. degree in computer networking from The University of Hong Kong, Hong Kong, in 2018. From 2011 to 2014, he was a Software Engineer with Harmonic Inc. He was a Post-Doctoral Fellow with The University of Hong Kong from 2018 to 2019. He is currently a Researcher with the Network Technology Laboratory, Huawei, Hong Kong. His research interests include congestion control, queue management, optimization of computer networks, edge computing, and online learning. He is a member of ACM.

**John C. S. Lui** (Fellow, IEEE) received the Ph.D. degree in computer science from the University of California at Los Angeles. He is currently the Choh-Ming Li Chair Professor with the Department of Computer Science and Engineering, The Chinese University of Hong Kong. His current research interests include machine learning, online learning (e.g., multi-armed bandit, reinforcement learning), network science, future internet architectures and protocols, network economics, network/system security, and large scale storage systems. He is an elected member of the IFIP WG 7.3, a Fellow of ACM, and a Senior Research Fellow of the Croucher Foundation. He was the Past Chair of the ACM SIGMETRICS (2011–2015). He received various departmental teaching awards and the CUHK Vice-Chancellor Exemplary Teaching Award. He was a co-recipient of the Best Paper Award from the IFIP WG 7.3 Performance 2005, IEEE/IFIP NOMS 2006, SIMPLEX 2013, and ACM RecSys 2017.