Distributed Computing and Networking Coordination for Task Offloading Under Uncertainties

Shichao Xia, *Member, IEEE*, Zhixiu Yao, Yun Li, *Member, IEEE*, Zhitong Xing, and Shiwen Mao, *Fellow, IEEE*

***Abstract*—The multi-access edge computing (MEC) and ultra-dense network (UDN) are regarded as essential and complementary technologies in the age of Internet of Things (IoT). Deploying MEC servers at the macro-cell and small-cell stations can significantly improve user experience as well as increase network capacity. Nevertheless, there still remain many obstacles in practical MEC-enabled UDNs. Among them, a unique challenge is how to coordinate computing and networking to fit the diverse offloading demands of IoT applications in dynamic network environments. To this end, this paper first investigates a distributed delay-constrained computation offloading methodology based on computing and networking coordination in the UDN. An extended game-theoretic approach based on the Lyapunov optimization theory is designed to achieve adaptive task offloading and computing power management in time-varying environments. Furthermore, considering the uncertainty in users' mobility and limited edge resources, distributed two-stage and multi-stage stochastic programming algorithms under various uncertainties are proposed. The proposed algorithms take posterior recourse actions to compensate for inaccurate predicted network information. Extensive simulations validate the effectiveness and rationality of the proposed algorithms and their superior performance over several benchmark schemes.**

***Index Terms*—Multi-access edge computing (MEC), ultra-dense network (UDN), distributed game, stochastic programming, uncertainty.**

I. Introduction

With the rapid development and convergence of the mobile Internet and the Internet of Things (IoT), the number of mobile devices (e.g., smartphones, wearable devices, and intelligent sensors) and computation-intensive applications (e.g., virtual reality (VR), augmented reality (AR), autonomous driving, and online gaming) are increasing dramatically, which keep on driving the computing power and storage resources to the edge of network to reduce transmission latency and avoid network congestion. Driven by this trend, multi-access edge computing (MEC) has become a hot research topic, which can significantly enhance users' experience by offloading some or all computation tasks to edge-cloud servers [1], [2].

In order to meet the requirements (including large system capacity, high peak data rate, and massive connection density) of the fifth generation (5G) mobile communication systems, ultra-dense network (UDN) has been recognized as a highly promising technology, and can effectively improve network throughput and access capacity by deploying some low-power and short-range small base stations (SBSs) over the coverage of the macro base station (MBS) [3], [4]. Existing studies show that integrating MEC into UDN has important and realistic significance [5]. As shown in Fig. 1, a general way is to deploy some lightweight MEC servers at the SBS side, and to offload parts of the computation loads to SBSs to relieve the burden at the MBS. It can further improve mobile services and allow users to enjoy ubiquitous computation services anytime and anywhere. However, such an integration has also brought about many new challenges.

First, compared with traditional central-cloud deployment scenarios (e.g., single MBS with MEC servers), the distributed edge cloud nodes disperse the computing power resources in the MEC-enabled UDN. How to efficiently use the decentralized computing power is an important issue. Meanwhile, dense and heterogeneous network access also brings about new opportunities and challenges to mobile communications and computing [3]. Consequently, to meet the ultra-reliable, low-latency, and

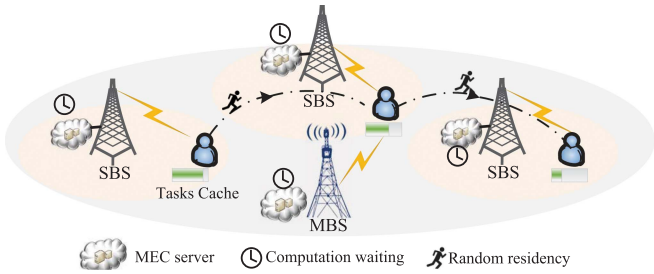


Fig. 1. Typical MEC-enabled UDN scenario.

Manuscript received 9 January 2023; revised 18 June 2023; accepted 8 August 2023. Date of publication 15 August 2023; date of current version 4 April 2024. This work was supported in part by the National Natural Science Foundation of China under Grants 62071077 and 62221005, in part by the Natural Science Foundation of Chongqing under Grant 2022NSCQ-LZX0191, in part by the Scientific and Technological Research Program of Chongqing Municipal Education Commission under Grant KJQN202200606, and in part by China Postdoctoral Science Foundation under Grant 2023MD734137. Recommended for acceptance by J. Ren. *(Corresponding author: Yun Li.)*

Shichao Xia is with the School of Software Engineering, Chongqing University of Posts and Telecommunications, Chongqing 400065, China (e-mail: xiashichao65@163.com).

Zhixiu Yao, Yun Li, and Zhitong Xing are with the School of Communication and Information Engineering, Chongqing University of Posts and Telecommunications, Chongqing 400065, China (e-mail: zhixiuyao@163.com; liyun@cqupt.edu.cn; xingzt@cqupt.edu.cn).

Shiwen Mao is with the Department of Electrical and Computer Engineering, Auburn University, Auburn, AL 36849-5201 USA (e-mail: smao@ieee.org).

This article has supplementary downloadable material available at https://doi.org/10.1109/TMC.2023.3305013, provided by the authors.

Digital Object Identifier 10.1109/TMC.2023.3305013

energy-efficiency computation offloading (ULECO) requirements in UDN, exploring the "computing + networking" based computation offloading methodologies will be essential and instrumental to enhance users' quality of experience, especially for delay-constrained applications.

Second, due to the random movement of mobile devices, frequent service switching among different cloud access points is unavoidable. Once a mobile device moves and is associated with another access point, a general method is to re-transmit the already offloaded tasks to the new MEC servers [6], which will incur considerable re-offloading or/and resources re-allocation costs, and even cause violation of offloading deadline. Proportional computation offloading by predicting the residence time under the coverage of MEC servers is an effective approach [7], [8], but it is usually difficult or even impossible to obtain perfect predictions in time-varying environments. Furthermore, the diverse lightweight edge-cloud servers with different function characteristics may not be able to respond quickly and timely to the bursty computation requirements. The random computation waiting delay (including the task queueing delay, decompression time, safety analysis time, etc.) of offloaded tasks also has a big impact [9], [10]. Careful designs are needed to deal with the stochastic residence time and computation waiting delay.

Finally, traditional centralized optimization frameworks require precise and timely updates of the global information of system states (e.g., traffic characteristics, network loads, channel state information (CSI), user mobility and association, etc.), which are usually highly challenging to obtain (if not impossible) in practical scenarios. The distributed MEC-enabled UDN provides an effective alternative solution. Tasks offloading and computing power allocation decisions should be formulated with a more efficient approach and in a distributed manner to increase the flexibility, adaptibility, and scalability of the network.

This work aims to design a distributed task offloading and computing power allocation methodology based on computing and networking coordination under various uncertainties in the random MEC-enabled UDN environment. The main contributions of this work can be summarized as follows:

- To meet the ultra-reliable, low-latency, and energy-efficiency computation offloading requirements, we first investigate a distributed delay-constrained computation offloading methodology based on "computing + networking" coordination in the MEC-enabled UDN.
- In order to achieve distributed task offloading and adaptive computing power management in time-varying network environments, a multi-round pricing method is designed by invoking an extended game-theoretic approach based on the Lyapunov optimization theory, which determines the computing power price dynamically by balancing the offloading revenue and queue backlog.
- Considering the stochastic residence time and computation waiting delay, a distributed two-stage algorithm and a multi-stage stochastic programming algorithm are proposed. Moreover, the multi-stage stochastic programming problem is transformed into a deterministic equivalent problem (DEP) using a *scenario tree* to derive the optimal computation offloading strategies.
- The performance of the proposed algorithms are evaluated with extensive simulations and compared with existing baseline schemes in terms of revenue, computation offloading success probability, offloading cost, and tasks backlog level, where their superior performance is demonstrated.

The rest of this paper is organized as follows. Section II reviews some related work. In Section III, we describe the system model and define the functions to formulate offloading and computing power management problem. Game theory model with Lyapunov optimization framework is designed in Section IV. Sections V and VI introduce the two-stage and multi-stage stochastic programming approach for the uncertain offloading game, respectively. Optimal game strategies and equilibrium existence are proved in Section VII. Section VIII evaluates the performances and discusses the numerical results. Finally, Section IX concludes this paper.

II. Related Work

The gap between the ever-increasing computing-intensive applications and resources-constrained mobile devices has brought about unprecedented challenges to the development of the Internet of Things (IoT) [11]. Driven by Big Data and artificial intelligence (AI), multi-access edge computing plays an increasingly important role in enabling low-latency and energy-efficient computation services [12].

In such a context, one of the key issues is how to effectively offload tasks to center/edge-cloud servers. To minimize energy consumption under delay constraint, Deng et al. in [13] formulated a task allocation problem based on the interplay and cooperation of fog and cloud, aiming to balance the power consumption and transmission delay to optimize the workload allocation in the fog and cloud. Considering the characteristics of computing power resources in the central cloud and the low transmission delay in MEC, Ning et al. in [14] solved an offloading delay minimization problem based on the cooperation of cloud computing and MEC in the IoT. To reduce the energy consumption while satisfying the task computation delay constraint, Guo et al. in [15] provided a distributed energy-efficient dynamic offloading and resource scheduling algorithm. Without requiring a priori knowledge of network, Chen et al. in [16] proposed a dynamic task offloading algorithm based on a double deep Q-network to maximize the long-term system revenue. Hou et al. in [17] proposed a framework for jointly addressing three QoS criteria: end-to-end delay, delivery ratio, and channel reliability to deal with the traffic with QoS constraints.

As one of the key technologies of the fifth generation (5G), ultra-dense network (UDN) can effectively improve the network throughput and access capacity [4], [18]. To achieve user association in UDN, Liakopoulos et al. in [19] developed a user association mapping method by invoking data-driven robust optimization theory to predict future traffic fluctuations. To jointly solve the problems of time resources allocation, user association, and beamforming design, Kwon et al. in [20] proposed a two-stage design approach to reduce complexity and maximize the weighted sum rate. In order to solve large-scale

load balancing problem of UDN, Xu et al. in [21] proposed a mobile load balancing algorithm using deep reinforcement learning along with a two-layer architecture, which supported the combination of multiple behavior strategies.

The convergence of MEC and UDN has been recognized as an advanced computation offloading architecture to provide efficient, reliable, convenient, and flexible services [22]. Guo et al. in [5] proposed a heuristic greedy computation offloading algorithm, and verified the superiority and necessity of task offloading over multiple MEC servers in the UDN system. Based on this, the authors proposed a two-layer game theory, greedy task offloading algorithm for MEC-enabled UDN in [23]. To jointly optimize task offloading, computing power scheduling, and radio resource allocation in MEC-enabled UDN, Zhang et al. in [24] formulated a random mixed integer nonlinear programming problem based on the Lyapunov optimization theory to minimize the system energy consumption. Considering the time-varying computation offloading environment, Deng et al. in [25] designed a dynamic task offloading and resource allocation method by applying Lyapunov optimization theory in MEC-enabled UDN.

Moreover, few studies combine game theory and Lyapunov optimization theory. Thereinto, Asheralieva et al. in [26] proposed a novel framework based on contract theory and Lyapunov optimization for content sharing in a wireless content delivery network, and achieving optimal performance and stability while considering incomplete information and unknown network state. Similarly, a framework combining cooperative game theory and Lyapunov optimization was proposed in [27] for a cloud-based caching system, and aiming to maximize content providers' expected payoff and stabilize the queuing system. Zhu et al. in [28] proposed a network selection scheme based on game theory and Lyapunov optimization. The problem is formulated as a repeated stochastic game and a Lyapunov optimization algorithm is developed from the game manager's perspective to maximize total utility and obtain optimal actions for each user. To address the access network selection problem in mobile devices equipped with multiple network interface, Li et al. in [29] formulated the problem as a repeated stochastic game, and Lyapunov optimization theory is used to obtain the network access strategy to maximize the total user utility.

In summary, numerous significant enhancements have been proposed and achieved in mobility management, computation offloading, resources allocation, and system stability for MEC and UDN network. However, there remain several challenges that have not been fully addressed in the practical MEC-enabled UDN system, e.g., 1) how to flexibly allocate resource to achieve the diverse user demands and ensure users' quality of experiences; 2) how to achieve the stability of computation performances in the multi-dimensional uncertain network; and 3) how to efficient coordination of computing and networking in a distributed manner. Different from the aforementioned existing studies, this paper emphasizes the diverse user demands, decentralized computation offloading and resource allocation, and aims to develop a distributed resource allocation methodology based on computing and networking coordination under various uncertainties environment for MEC-enabled UDN.

III. System Model and Problem Formulation

We consider an MEC-enabled UDN system operates in discrete time $\mathcal{T} = \{0, 1, \ldots\}$, and let $\tau \in \mathcal{T}$ denote the duration of each time slice. As shown in Fig. 1, the system comprises a macro base station (MBS) and n small base stations (SBSs) serving a region. Let $\mathcal{N} = \{1, 2, \ldots, n\}$ denote the set of SBSs, and M and $S \in \mathcal{N}$ denote the MBS and SBS S, respectively. The m mobile devices (MDs) are indexed by $\mathcal{M} = \{1, 2, \ldots, m\}$ in the region. Each BS is equipped with computing functionalities, as MEC server that provides computing power to the MDs within its radio coverage. The MDs can process tasks locally and/or offload tasks to the MEC servers of BSs, and each MD can simultaneously connect to the MBS and one of the nearby SBS with dual connectivity or coordinated multiple points (CoMP) transmission/reception techniques developed for 5G networks [5].

A. Computation Task and Processing Model

The computation tasks of MDs follow the data-partition model [30], [31], namely, the task-input bits are bit-wise independent and can be arbitrarily divided into different groups to be executed at local or edge-cloud servers. Considering differentiated service quality requirements of applications, the tasks of MD i can be characterized by a tuple with four parameters as $\Lambda_i(t) = \langle Q_i(t), D_i(t), \tau_i^d, \gamma_i \rangle$,[1] in which $Q_i(t)$ and $D_i(t)$ denote the task queue backlog and the size of task that needs to be processed for device i in time slice t, respectively; τ_i^d denotes the maximum computation latency constraint of $D_i(t)$; and γ_i is the computation density (in *cycles/bit*), which can be obtained through off-line measurements [32].

Let $A_i(t)$ and $\mathcal{A}(t) = \{A_1(t), A_2(t), \ldots, A_m(t)\}$ denote the arrived tasks at MD i ($i \in \mathcal{M}$) and the set of all MDs during time slice t, respectively. Since the amount of arrived tasks in a time slice is finite, we have $0 \leq A_i(t) \leq A_i^{\max}$, $t \in \mathcal{T}$, and $A_i^{\max}$ is the maximum arrived task size. Assume that the task arrival rates of all MDs are independent and identically distributed (i.i.d.). The evolution of the queue backlog of MD i is given by

$$Q_i(t+1) = Q_i(t) - D_i(t) + A_i(t),\ i \in \mathcal{M}, t \in \mathcal{T}, \quad (1)$$

where $D_i(t) = D_{i,L}(t) + \sum_{k \in \{M,S\}} D_{i,k}(t)$, in which $D_{i,L}(t)$, $D_{i,M}(t)$, and $D_{i,S}(t)$ denote the amounts of tasks processed locally, at the MBS, and at the SBS, respectively.

At beginning of a time slice, each mobile device shall determine its task processing strategies: the amount of locally executed tasks (i.e., $D_{i,L}(t)$) and/or the amount of offloaded tasks (i.e., $D_{i,k}(t), k \in \{M, S\}$). As shown in Fig. 2, there are three stages in the offloading procedure. First, MDs upload computation tasks ($D_{i,k}(t)$) to BSs through their wireless channels.

[1] To simplify analysis, we assume that the computation latency constraint and computation density of a processed task do not change in each time slice.

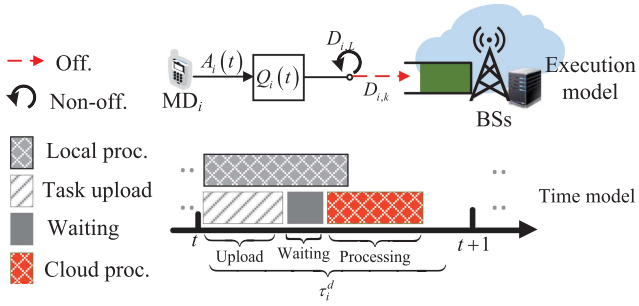


Fig. 2. Tasks execution and time slice model.

Let $T_{i,k}^{up}(t)$ be the task upload latency in this case. Then, there will be a waiting time at the servers, referred to as waiting time $T_{i,k}^{wt}(t)$. The offloaded tasks will be executed by the servers, and the processing time is denoted as $T_{i,k}^{pt}(t)$. Note that no matter where a task is executed, it must be processed within the delay constraint τ_i^d; Otherwise, it will be regarded as an offloading failure. We next analyze the task execution models and communication model in the following.

B. Local Execution Model

Limited by size and cost of the hardware, the battery capacity of mobile devices is finite. To achieve energy savings under the delay constrain. We consider that the MDs can process tasks at an appropriate CPU clock frequency determined by the dynamic voltage frequency scaling (DVFS) technique [33]. The relationship between the processed tasks and locally computing resources is given by

$$D_{i,L}(t) = \int_{T_{i,L}^{pt}(t)} \frac{f_{i,L}(t)}{\gamma_i} dt, \tag{2}$$

where $T_{i,L}^{pt}(t) \leq \tau_i^d$ is the local execution time, $f_{i,L}(t)$ ($f_{i,L}^{\min} \leq f_{i,L}(t) \leq f_{i,L}^{\max}$) is the allocated CPU clock frequency (in *cycles/s*) in time slice t, and $f_{i,L}^{\min}$ and $f_{i,L}^{\max}$ are the minimum and maximum CPU clock frequencies of MD i, respectively.

For each MD, we assume that the energy consumption is mainly due to the CPU options. Typically, the energy consumption $E_{i,L}^{exe}(t)$ of the CPU is proportional to the CPU clock frequency, modeled as follows.

$$E_{i,L}^{exe}(t) = \kappa_i \int_{T_{i,L}^{pt}(t)} (f_{i,L}(t))^2 dt, \tag{3}$$

where κ_i is the effective energy coefficient associated with chip architecture of the device, which can be obtained through off-line measurements [34].

C. MEC Server Execution Model

Without loss of generality, we first make several general assumptions. First, if MDs are within the coverage of MBS and SBSs, they can communicate with the MBS or/and one of the nearby SBS simultaneously in each time slice, which means that each MD can simultaneously offload tasks to the MBS or/and one nearby SBS. Furthermore, due to the random movements and the bursty computation requirements of the MDs, the residence time for an MD to stay in the coverage of an SBS (denoted by $T_{i,S}^{st}(t), i \in \mathcal{M}$) and waiting delay (i.e., $T_{i,k}^{wt}(t), k \in \{M, S\}$) in different BSs are stochastic and i.i.d. In addition, when MDs leave the radio coverage of the SBS while uploading tasks to the SBS (the uploaded tasks are represented by $D_{i,S}^{tr}(t)$), the remaining amount of tasks (i.e., $D_{i,S}(t) - D_{i,S}^{tr}(t)$) will have to be uploaded to the MBS first, and then to be forwarded to the SBS by the MBS through a wired optical fiber network, e.g., via the X2 interface.[2]

1) Communication Model: For delay-sensitive applications, it is necessary that the uploaded latency of tasks is less than the constraint time, e.g., $T_{i,k}^{up}(t) < \tau_i^d$. Otherwise, the tasks is bound to execution failed. Since computation results are usually much smaller than input tasks size of most applications, we neglect time cost for downloading the computation results back from the BSs. Once each MD decides offload tasks to the MBS or/and the nearby SBS, the tasks upload latency, and communication energy consumption can be calculated as follows, respectively.

Offloading Tasks to the MEC Server at the MBS: For each MD, the input bits of the tasks shall be delivered to the MBS via the wireless channel. Note that as the number of users is increased, the interference environment in the UDN become more and more complicated, and the offloading decisions should take the mutual interference among MDs into account. According to the Shannon-Hartley formula, the uploading data rate $r_{i,M}(t)$ is given by

$$r_{i,M}(t) = \omega_M \log_2 \times \left(1 + \frac{P_i(t) g_{i,M}(t)}{\sigma^2(t) + \sum_{x \in \mathcal{M} \setminus x=i} P_x(t) g_{x,M}(t) \cdot \mathbf{1}(D_{x,M}(t))}\right), \tag{4}$$

where ω_M and $g_{i,M}(t)$ are the bandwidth and channel gain of the MBS, respectively; $\sigma^2(t)$ is the average background noise power; and $P_x(t)$ is the transmit power of MD x in time slice t. $\mathbf{1}(\cdot)$ is an indicator function, and when $D_{x,M} > 0$, $\mathbf{1}(D_{x,M}) = 1$, otherwise, $\mathbf{1}(D_{x,M}) = 0$.

Accordingly, we compute the task upload latency $T_{i,M}^{up}(t)$ and the communication energy consumption $E_{i,M}^{up}(t)$ as

$$T_{i,M}^{up}(t) = \frac{D_{i,M}(t)}{r_{i,M}(t)}, \ i \in \mathcal{M} \tag{5}$$

$$E_{i,M}^{up}(t) = P_i(t) \frac{D_{i,M}(t)}{r_{i,M}(t)}, \ i \in \mathcal{M}. \tag{6}$$

Offloading Tasks to the MEC Server at an SBS. Similarly, if MD i decides to offload tasks $D_{i,S}(t)$ to the MEC server at an SBS, the upload date rate $r_{i,S}(t)$ is given by

$$r_{i,S}(t) = \omega_S \log_2 \times \left(1 + \frac{P_i(t) g_{i,S}(t)}{\sigma^2(t) + \sum_{x \in \mathcal{M} \setminus x=i} P_x(t) g_{x,S}(t) \cdot \mathbf{1}(D_{x,S}(t))}\right), \tag{7}$$

[2]It is assumed that the switching speed between different BSs is sufficiently fast and thus the switching time can be ignored.

where ω_S and $g_{i,S}(t)$ are the bandwidth and channel gain of the SBS channel, respectively.

Generally, the coverage of an SBS is smaller than the MBS. According to the aforementioned assumptions, if the MDs leave the radio coverage area of the SBS when uploading tasks, the remaining tasks will be uploaded to the MBS first, and then will be forwarded to the SBS. The uploading latency is given by

$$T_{i,S}^{up}(t)=\begin{cases}T_{i,S}^{st}(t)+\hat{T}_{i,M}^{up}+T_{i,M}^{f}(t), & \text{if } T_{i,S}^{st}<\frac{D_{i,S}(t)}{r_{i,S}(t)}\\ \frac{D_{i,S}(t)}{r_{i,S}(t)}, & \text{if } T_{i,S}^{st}\geq\frac{D_{i,S}(t)}{r_{i,S}(t)},\end{cases} \tag{8}$$

where $T_{i,S}^{st}(t)$ is the residence time of MD i in the coverage of SBS S; $\hat{T}_{i,M}^{up}=\frac{D_{i,S}(t)-T_{i,S}^{st}r_{i,S}(t)}{r_{i,M}(t)}$ and $T_{i,M}^{f}(t)=\frac{D_{i,S}(t)-T_{i,S}^{st}r_{i,S}(t)}{o}$ denote the uploading and forwarding time of the remaining tasks, respectively; and o is the data rate of the optical fiber link.

The communication energy consumption $E_{i,S}^{up}(t)$ for uploading $D_{i,S}(t)$ to the SBS is given by

$$E_{i,S}^{up}(t)=\begin{cases}P_i(t)\left(T_{i,S}^{st}(t)+\hat{T}_{i,M}^{up}\right), & \text{if } T_{i,S}^{st}<\frac{D_{i,S}(t)}{r_{i,S}(t)}\\ P_i(t)\frac{D_{i,S}(t)}{r_{i,S}(t)}, & \text{if } T_{i,S}^{st}\geq\frac{D_{i,S}(t)}{r_{i,S}(t)}.\end{cases} \tag{9}$$

The total energy consumption of MD i in time slice t is computed as follows.

$$E_{i,L}(t)=E_{i,L}^{exe}(t)+\sum_{k\in\{M,S\}}E_{i,k}^{up}(t). \tag{10}$$

To ensure timely processing of the offloaded tasks, it is essential to ensure that the total computation offloading time $T_{i,k}^{co}(t)$ (including the uploading time, waiting delay, and processing time) does not exceed the delay constraint (i.e., $T_{i,k}^{co}(t)\leq\tau_i^d$) in each time slice. According to (5) and (8), the total computation offloading time for the MBS or SBS is given by[3]

$$T_{i,k}^{co}(t)=T_{i,k}^{up}(t)+T_{i,k}^{wt}(t)+T_{i,k}^{pt}(t),\ k\in\{M,S\}. \tag{11}$$

2) MEC Server Execution Model: Assume that the MEC servers are equipped with an L_k-core CPU, and the set of CPU cores is denoted as $\mathcal{L}_k=\{1,\ldots,L_k\}, k\in\{M,S\}$. According to [35], the energy consumption to process tasks $D_{i,k}(t)$ at the MEC server is

$$E_{i,k}(t)=\sum_{l_k\in\mathcal{L}_k}\kappa_{l_k}(\beta_{i,l_k}(t))^2T_{i,k}^{pt}(t),\ k\in\{M,S\}, \tag{12}$$

where κ_{l_k} and $\beta_{i,l_k}(t)$ ($\beta_{l_k}^{\min}\leq\beta_{i,l_k}(t)\leq\beta_{l_k}^{\max}$) denote the effective switched capacitance and CPU clock frequency of the l_kth CPU core at the MEC server, respectively; and $\beta_{l_k}^{\min}$ and $\beta_{l_k}^{\max}$ are the minimum and maximum CPU clock frequency of each CPU core, respectively.

[3]If there is a non-negligible switching time (e.g., a constant d) from the SBS to MBS, then the total computation offloading time for SBS can be rewrote as $T_{i,S}^{co}(t)=T_{i,S}^{up}(t)+T_{i,S}^{wt}(t)+T_{i,S}^{pt}(t)+d$.

D. Utility and Cost Model

The analysis in Section III-C, shows that the computation offloading latency strongly depends on the task communication delay, waiting delay, and processing delay. To ensure the offloaded tasks be processed within the delay constraint τ_i^d, it is necessary to take the computing factors (i.e., the computation waiting time $T_{i,k}^{wt}(t)$, the computing power $f_{i,L}$ and $\beta_{i,l_k}(t)$) and the networking factors (i.e., the residence time $T_{i,S}^{st}(t)$ and the communication time $T_{i,k}^{up}(t)$) into consideration. Consequently, exploring computation offloading based on computing and networking coordination will be essential to further improve users' experiences. It is worth noting that, the edge computing service costs is also one of the important factors that affect users' offloading strategies [31]. As a result, to evaluate the task processing performance, we first define the task processing utility function and the task processing cost functions.

1) Utility Function Model: To evaluate the benefit of processing tasks in each time slice, we adopt the logarithmic utility function for each MD, which has been widely used in the wireless communications and mobile computing domains [36], [37], [38]. The utility of MD i, $u_i[D_i(t)]$, is given by

$$u_i\left[D_i(t)\right]=\sum_{\hat{k}\in\{L,M,S\}}\rho_i\log\left[1+D_{i,\hat{k}}(t)\right],\ t\in\mathcal{T}, \tag{13}$$

where ρ_i is the weight of MD i.

2) Tasks Processing Cost Function Model: We next define the task processing cost functions, including the communication cost, the energy consumption cost, and the edge computing service cost.

Communication Cost Function: We first obtain the transmitted tasks over the MBS ($D_{i,M}^{tr}(t)$) and an SBS ($D_{i,S}^{tr}(t)$) as follows.

$$D_{i,M}^{tr}(t)=\begin{cases}D_{i,M}(t)+D_{i,S}(t)\\ \qquad -T_{i,S}^{st}r_{i,S}(t), & \text{if } T_{i,S}^{st}<\frac{D_{i,S}(t)}{r_{i,S}(t)}\\ D_{i,M}(t), & \text{if } T_{i,S}^{st}\geq\frac{D_{i,S}(t)}{r_{i,S}(t)}\end{cases}$$

$$D_{i,S}^{tr}(t)=\begin{cases}T_{i,S}^{st}r_{i,S}(t), & \text{if } T_{i,S}^{st}<\frac{D_{i,S}(t)}{r_{i,S}(t)}\\ D_{i,S}(t), & \text{if } T_{i,S}^{st}\geq\frac{D_{i,S}(t)}{r_{i,S}(t)}.\end{cases}$$

Following [38], we define the communication cost $c_i[D_i(t)]$ of MD i as follows.

$$c_i\left[D_i(t)\right]=\theta_iD_{i,M}^{tr}(t)+\eta_iD_{i,S}^{tr}(t),\ t\in\mathcal{T}, \tag{14}$$

where θ_i and η_i are the communication cost per bit of MD i at the MBS and SBS, respectively.

Energy Consumption Cost Function: To simplify analysis, we assume that the unit energy consumption cost is denoted by $\lambda_{\hat{k}}\geq0$, $\hat{k}\in\{L,M,S\}$. Accordingly, we have the energy consumption cost $e_{i,\hat{k}}[D_i(t)]$ as follows.

$$e_{i,\hat{k}}\left[D_i(t)\right]=\lambda_{\hat{k}}E_{i,\hat{k}}(t),\ t\in\mathcal{T}. \tag{15}$$

Edge Computing Service Cost Function: Providing computation offloading services incurs costs to the cloud server providers (e.g., energy consumption cost, hardware cost, etc.). It is obvious

that the computation offloading services are not free of charge. The edge-cloud servers should also be compensated for sharing their resources [40], [41]. Let $p_{i,k}(t), k \in \{M, S\}$ (in $\$/bit$) denote the payment of MD i to BS k. We define the computing service cost $s_i[D_i(t)]$ of MD i as follows.

$$s_i\left[D_i(t)\right] = \sum_{k\in\{M,S\}} p_{i,k}(t) D_{i,k}(t),\ i \in \mathcal{M}. \tag{16}$$

Accordingly, the utility of BS k is defined as

$$u_k\left[\boldsymbol{p}_k(t)\right] = \sum_{\forall i\in\mathcal{M}} p_{i,k}(t) D_{i,k}(t),\ k \in \{M, S\}, \tag{17}$$

where $\boldsymbol{p}_k(t) \triangleq [p_{1,k}(t), p_{2,k}(t), \ldots, p_{m,k}(t)]$ denotes the set of pricing strategies of the MBS or SBSs.

IV. Distributed Dynamic Pricing Based Computation Offloading

Generally, a suitable pricing strategy $\boldsymbol{p}_k(t)$ can not only provide an incentive for offloading, but also promote the rationality useful for edge resources. For instance, if $\boldsymbol{p}_k(t)$ is higher, users will be more willing to process tasks locally. On the contrary, if $\boldsymbol{p}_k(t)$ is low, users will seek more computing resources to reduce their tasks backlog level, and even beyond their needs since users are greedy [42], [43]. This can be viewed as a "market," in which the mobile devices can be regarded as *buyers* who buy products (i.e., computing power) from the market, and the BSs can be regarded as *sellers* who provide computing power resources.

In this section, we aim to develop a dynamic pricing method to address the diverse user demands and time-varying environment. We first formulate task offloading and computing power allocation as a deterministic optimization problem, in which the residence time $(T^{st}_{i,S}(t), S \in \mathcal{N})$ and computation waiting delay $(T^{wt}_{i,k}(t), k \in \{M, S\})$ are assumed perfectly and fully known in advance. Then the problem with uncertain factors will be further addressed in Sections V and VI.

A. MDs/Buyers Game With Lyapunov Optimization

We assume that the MDs are always rational and seeking an optimal offloading decision to maximize their long term revenue. It is obvious that the optimal decisions of MDs/buyers should take offloaded task utility, communication cost, energy consumption cost, and payment into account. According to (13), (14), (15), and (16), the object function of MD/buyer i in time slice t is given by

$$U_{b_i}\left[D_i(t)\right] = u_i\left[D_i(t)\right] - c_i\left[D_i(t), T^{st}_{i,S}(t)\right] - e_{i,L}\left[D_i(t), T^{st}_{i,S}(t)\right] - s_i\left[D_i(t)\right]. \tag{18}$$

To guarantee the computation performance in the long-term evolution, we define problem $\mathcal{P}1$-Buyer for MD/buyer i as follows.

$\mathcal{P}1$-Buyer

$$\max_{f_{i,L}(t), D_i(t)} \bar{U}_{b_i} = \lim_{T\to+\infty} \frac{1}{T}\mathbb{E}\left[\sum_{t=0}^{T-1}\left\{U_{b_i}\left[D_i(t)\right]\right\}\right] \tag{19}$$

$$\text{s.t. } T^{pt}_{i,L}(t) \le \tau^d_i \tag{20}$$

$$f^{\min}_{i,L} \le f_{i,L}(t) \le f^{\max}_{i,L},\ t \in \mathcal{T} \tag{21}$$

$$0 \le D_i(t) \le Q_i(t),\ t \in \mathcal{T} \tag{22}$$

$$\bar{Q}_i = \lim_{T\to+\infty} \sup \frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\{Q_i(t)\} < +\infty. \tag{23}$$

To develop the task offloading decision under the time-varying environment and decouple the time-dependent information, we first define the Lyapunov function for the computation task queue of MD i as follows.

$$L_i\left(Q_i(t)\right) = \frac{1}{2}\{Q_i(t)\}^2 \ge 0. \tag{24}$$

According to [45], the conditional Lyapunov Drift is given by

$$\Delta\left(Q_i(t)\right) = \mathbb{E}\left\{L_i\left(Q_{i,t+1}\right) - L_i\left(Q_i(t)\right)|Q_i(t)\right\}. \tag{25}$$

The underlying objective of the online optimal decision is to minimize the upper bound of the *drift-minus-utility* function, which is defined as follows.

$$\Delta\left(Q_i(t)\right) - V_i\mathbb{E}\left\{U_i\left[D_i(t)\right]|Q_i(t)\right\}, \tag{26}$$

where $V_i \ge 0$ is a non-negative controllable parameter. To obtain the upper bound of the drift-minus-utility, we have the following theorem below.

Theorem 1: For any given control parameter $V_i \ge 0$ and $A_i(t) \in [0, A_i^{\max}]$ under any possible decision, we have

$$\Delta\left(Q_i(t)\right) - V_i\mathbb{E}\left\{U_i\left[D_i(t)\right]|Q_i(t)\right\} \le B_i + Q_i(t)A_i(t) - \mathbb{E}\left\{Q_i(t)D_i(t)|Q_i(t)\right\} - V_i\mathbb{E}\left\{U_i\left[D_i(t)\right]|Q_i(t)\right\}, \tag{27}$$

where $B_i = \frac{1}{2}\{(D_i(t)^{\max})^2 + (A_i(t)^{\max})^2\}$.

Proof: Obviously we have $Q_i(t) \ge 0$, $D_{i,k}(t) \ge 0$ and $A_i(t) \ge 0$. It follows that

$$\left\{[Q_i(t) - b_i(t)]^+ + A_i(t)\right\}^2 \le Q_i(t)^2 + A_i(t)^2 + D_{i,k}(t)^2 + 2Q_i(t)\left(A_i(t) - D_{i,k}(t)\right). \tag{28}$$

Following inequality (28), we deduce the follow inequality.

$$\begin{aligned} L\left(Q_{i,t+1}\right) - L\left(Q_i(t)\right) &= \frac{1}{2}\left({Q_{i,t+1}}^2 - Q_i(t)^2\right) \\ &= \frac{1}{2}\left\{[Q_i(t) - D_{i,k}(t)]^+ + A_i(t)\right\}^2 - \frac{1}{2}Q_i(t)^2 \\ &\le \frac{A_i(t)^2 + D_{i,k}(t)^2}{2} + Q_i(t)\left(A_i(t) - D_{i,k}(t)\right). \end{aligned} \tag{29}$$

Solving the conditional expectation and adding $-V_i\mathbb{E}(U_{b_i}(t)|Q_i(t))$ to both sides of (29), we have

$$\begin{aligned} &\Delta\left(Q_i(t)\right) - V_i\mathbb{E}\left(U_i\left[D_i(t)\right]|Q_i(t)\right) \\ &\le \mathbb{E}\left(\frac{A_i(t)^2 + D_{i,k}(t)^2}{2}|Q_i(t)\right) \\ &+ Q_i(t)\lambda_i - \mathbb{E}\left(Q_i(t)D_{i,k}(t)|Q_i(t)\right) - V_i\mathbb{E}\left(U_i\left[D_i(t)\right]|Q_i(t)\right). \end{aligned}$$

Letting $B_i = \frac{(D_{i,k}(t)^{\max})^2 + \lambda_i^2}{2} \ge \mathbb{E}(\frac{A_i(t)^2 + D_{i,k}(t)^2}{2}|Q_i(t))$, we then derive the inequality (27).

It can be seen that minimizing the upper bound of drift-minus-utility is equivalent to minimize the Right-Hand-Side (RHS) of inequality (27). As a result, we can transform problem $\mathcal{P}1$-Buyer to $\mathcal{P}2$-Buyer as follows.

$\mathcal{P}2$-Buyer

$$\max_{f_{i,L}(t),D_{i,k}(t)} Y_{b_i}(t) = V_i \cdot U_{b_i}\left[D_i(t)\right] + \sum_{k\in\{L,M,S\}} Q_i(t)\cdot D_{i,k}(t) - Q_i(t)\cdot A_i(t)$$
$$\text{s.t.} \quad (20)\text{-}(23). \tag{30}$$

B. BSs/Sellers Game With Dynamic Pricing

For BSs/sellers, according to (15), (16), and (17), the revenue $\bar{U}_{s_k}, k \in \{M, S\}$ of the MBS or an SBS can be defined, respectively. The $\mathcal{P}1$-Seller problems are defined in the following.

$\mathcal{P}1$-Seller (MBS)

$$\max_{p_{i,M}(t),\beta_{i,l_M}(t)} \bar{U}_{s_M} = \lim_{T\to+\infty}\frac{1}{T}\mathbb{E}\left[\sum_{t=0}^{T-1}\left\{u_M\left[\boldsymbol{p}_M(t)\right] - \sum_{\forall i} e_{i,M}\left[D_{i,M}(t), T^{wt}_{i,M}(t)\right]\right\}\right] \tag{31}$$
$$\text{s.t.} \quad p_{i,M}(t) \geq 0, t \in \mathcal{T} \tag{32}$$
$$T^{co}_{i,M}(t) \leq \tau^d_i, t \in \mathcal{T} \tag{33}$$
$$\beta^{\min}_{l_M} \leq \beta_{i,l_M}(t) \leq \beta^{\max}_{l_M},\ t \in \mathcal{T}. \tag{34}$$

$\mathcal{P}1$-Seller (SBS)

$$\max_{p_{i,S}(t),\beta_{i,l_S}(t)} \bar{U}_{s_S} = \lim_{T\to+\infty}\frac{1}{T}\mathbb{E}\sum_{t=0}^{T-1}\left\{u_S\left[\boldsymbol{p}_S(t)\right] - \sum_{\forall i} e_{i,S}\left[D_{i,S}(t), T^{st}_{i,S}(t), T^{wt}_{i,S}(t)\right]\right\} \tag{35}$$
$$\text{s.t.} \quad p_{i,S}(t) \geq 0,\ t \in \mathcal{T} \tag{36}$$
$$T^{co}_{i,S}(t) \leq \tau^d_i,\ t \in \mathcal{T}, \tag{37}$$
$$\beta^{\min}_{l_S} \leq \beta_{i,l_S}(t) \leq \beta^{\max}_{l_S},\ t \in \mathcal{T}. \tag{38}$$

Noted that both the revenues of the MBS and SBS are dependent on the pricing set $p_{i,k}(t)$, allocated computing power $\beta_{i,l_k}(t)$, and computation waiting delay $T^{wt}_{i,k}(t)$ in time slice t. To ensure the delay constraints of the offloaded tasks, SBS S shall take the residence time $T^{st}_{i,S}(t)$ of MD i into consideration. Furthermore, we can transform problem $\mathcal{P}1$-Seller to $\mathcal{P}2$-Seller problems based on the maximum value theory as follows.

$\mathcal{P}2$-Seller (MBS)

$$\max_{p_M(t),\beta_{i,l_M}(t)} Y_{s_M}(t) = u_M\left[\boldsymbol{p}_M(t)\right] - \sum_{\forall i} e_{i,M}\left[D_{i,M}(t), T^{wt}_{i,M}(t)\right] \tag{39}$$
$$\text{s.t.} \quad (32), (33), (34). \tag{40}$$

$\mathcal{P}2$-Seller (MBS)

$$\max_{p_S(t),\beta_{i,l_S}(t)} Y_{s_S}(t) = u_S\left[\boldsymbol{p}_S(t)\right] - \sum_{\forall i} e_{i,S}\left[D_{i,S}(t), T^{st}_{i,S}(t), T^{wt}_{i,S}(t)\right] \tag{41}$$
$$\text{s.t.} \quad (36), (37), (38). \tag{42}$$

According to $\mathcal{P}2$-Seller, each seller periodically announces the latest round market price $\boldsymbol{p}_k(t)$ by means of users' requirements $D_{i,k}(t)$ and current network status (e.g., the computation waiting delay $T^{wt}_{i,k}(t)$ and residence time $T^{st}_{i,S}(t)$) in each time slice. As the requirements and network state change over time, the sellers will also dynamically change their pricing strategies until the market equilibrium is reached. Meanwhile, the allocated computing power $\beta_{i,l_k}(t)$ will also be adjusted. We will also analyze the optimal game strategies and prove that the optimal solutions are Stackelberg Equilibrium (SE) solutions in Section VII.

V. Distributed Two-Stage Stochastic Programming Under Uncertainty

In the previous sections, both the residence time ($T^{st}_{i,S}(t)$) and waiting delay ($T^{wt}_{i,k}(t), k \in \{M, S\}$) are assumed to be perfectly and fully known. However, due to the stochastic movements and bursting computation requirements of users, such information is usually uncertain at beginning of each time slice. It is natural that we can make decisions by using the average historical values or predicting values [46], [47], but it is usually challenging to accurately predict such values in practice. The unavoidable prediction errors could affect the performance of the offloading scheme. For example, if the predicted results are larger than the actual residence time and waiting delay, it will increase the computation offloading cost and even lead to failure of offloading. To this end, we adopt a two-stage stochastic programming approach to take posterior recourse actions and compensate for the previously imprecise predictions.

A. Uncertain Residence Time and Computation Waiting Delay

To handle the uncertain residence time ($T^{st}_{i,S}(t)$) and waiting delay ($T^{wt}_{i,k}(t)$), we first take into account a set of scenarios, which corresponds to a set of uncertain parameters [44]. Let $\Omega^{st}_{i,S}(t)$ denote the set of possible residence times of MD i when covered by an SBS, and let $\Omega^{wt}_{i,k}(t)$ denote the set of possible waiting delays of MD i when served by BS $k, k \in \{M, S\}$.[4] With the Cartesian product, we obtain the *composite scenarios* of residence time $\boldsymbol{\Omega}^{st}_S(t)$ and waiting delay $\boldsymbol{\Omega}^{wt}_k(t)$ of all MDs as follows.

$$\boldsymbol{\Omega}^{st}_S(t) = \prod_{i=1}^{m}\Omega^{st}_{i,S}(t) = \Omega^{st}_{1,S}(t) \times \cdots \times \Omega^{st}_{m,S}(t) \tag{43}$$

[4]Assume that both $\Omega^{st}_{i,S}(t)$ and $\Omega^{wt}_{i,k}(t)$ take a finite or countable number of support.

$$\mathbf{\Omega}_k^{wt}(t) = \prod_{i=1}^{m} \Omega_{i,k}^{wt}(t) = \Omega_{1,k}^{wt}(t) \times \cdots \times \Omega_{m,k}^{wt}(t). \quad (44)$$

According to (18), the revenue of MDs is dependent on the residence time scenarios $\Omega_{i,S}^{st}(t)$. Correspondingly, the revenues of the MBS and SBSs are related to the composite scenarios of $\mathbf{\Omega}_M^{wt}(t)$ and $\mathbf{\Omega}_S(t) = \mathbf{\Omega}_S^{st}(t) \times \mathbf{\Omega}_S^{wt}(t)$, respectively, in which $\mathbf{\Omega}_S(t)$ is the composite scenarios of composite time (residence time and waiting delay when served by SBS *S*). Next, we analyze the distributed two-stage stochastic programming model of the game. For simplicity, we focus on the SBS model in the following, while the analysis of the MBS model can be derived similarly.

B. Two-Stage Stochastic Programming Formulation

In the uncertain and time-varying network environment, the exact values of a stochastic variable is only known after realization. That is, the residence time $T_{i,S}^{st}(t)$ and waiting delay $T_{i,S}^{wt}(t)$ are only known after the game. However, we can still take posterior recourse actions to compensate the game strategies after observing the realizations. According to the stochastic programming theory, we divide the decision set into the following two groups:

1) First-stage (game stage) decisions: the strategies of task offloading $D_i(t)$ and announced price $p_{i,S}(t)$ have to be taken before the knowledge of $T_{i,S}^{st}(t)$ and $T_{i,S}^{wt}(t)$ through the game, and the period is called the *first stage* or *game stage*;
2) Second-stage (recourse stage) decisions: the computing power allocation $\beta_{i,l_S}(t)$ can be taken after observing the realization of $T_{i,S}^{st}(t)$ and $T_{i,S}^{wt}(t)$. This is called the second-stage decision, and the corresponding period is called the *second stage* or *recourse stage*.

Let $T_{i,S}^{st}(t) \in \Omega_{i,S}^{st}(t)$ and $\omega_S(t) = (T_{1,S}^{st}, ..., T_{m,S}^{st}, T_{1,S}^{wt}, ..., T_{m,S}^{wt}) \in \mathbf{\Omega}_S(t)$ denote the realizations of the residence time and composite realization in time slice t, respectively. And let $p[T_{i,S}^{st}(t)] \in [0,1]$ and $p[\omega_S(t)] \in [0,1]$ denote the probabilities. According to the stochastic programming theory, problem $\mathcal{P}2$-Buyer can be rewritten as a two-stage stochastic programming problem as follows.

$\mathcal{P}3$-Buyer (two-stage)

$$\begin{aligned}\max_{f_{i,L}(t), D_{i,k}(t)} \; & Y_{b_i}(t) = V_i \Big\{ u_i\left[D_i(t)\right] - s_i\left[D_i(t)\right] \\ & - \mathbb{E}_{\Omega_{i,S}^{st}(t)} \left\{ c_i \left(D_{i,t}(t), T_{i,S}^{st}(t) | T_{i,S}^{st}(t) \in \Omega_{i,S}^{st}(t) \right) \right\} \\ & - \mathbb{E}_{\Omega_{i,S}^{st}(t)} \left\{ e_{i,L} \left(D_{i,t}(t), T_{i,S}^{st}(t) | T_{i,S}^{st}(t) \in \Omega_{i,S}^{st}(t) \right) \right\} \Big\} \\ & + \sum_{k \in \{L,M,S\}} Q_i(t) D_{i,k}(t) - Q_i(t) A_i(t) \\ \text{s.t.} \quad & (20)\text{-}(23). \end{aligned} \quad (45)$$

Similarly, problem $\mathcal{P}2$-Seller (SBS) can be rewritten as follows.

$\mathcal{P}3$-Seller (two-stage)

$$\begin{aligned}\max_{p_S(t), \beta_{i,l_S}(t)} \; & Y_{s_S}(t) = u_S\left[\boldsymbol{p}_S(t)\right] \\ & - \mathbb{E}_{\Omega_S(t)} \left[R\left(\beta_{i,l_S}(t), \Omega_S(t)\right) \right] \\ \text{s.t.} \quad & (36)\text{-}(38). \end{aligned} \quad (46)$$

where

$$\begin{aligned} & R\left(\beta_{i,l_S}(t), \Omega_S(t)\right) \\ & = \sum_{\forall i} e_{i,S} \left[D_{i,S}(t), T_{i,S}^{st}\left(\omega_S(t)\right), T_{i,S}^{wt}\left(\omega_S(t)\right) \right]. \end{aligned} \quad (47)$$

Here $R(\beta_{i,l_S}(t), \Omega_S(t))$ is called the *recourse function*.

VI. Distributed Multi-Stage Stochastic Programming for Uncertainty

As shown in Fig. 1, each MD may move randomly passing two or more SBSs. However, the strategies based on the two-stage stochastic programming are decided once in each time slice, which may lead to sub-optimal strategies. To capture the statistical characteristics of $T_{i,S}^{st}(t)$ and $T_{i,S}^{wt}(t)$ more accurately and make better offloading, pricing, and computing power allocation strategies, we further develop the strategies with multi-stage stochastic programming. With this approach, the computation offloading process is split into H steps (denoted as $\mathcal{H} \triangleq \{\tau_1, \tau_2, \ldots, \tau_H\}$), and the task offloading $D_i(\tau_h)$, $\tau_h \in \mathcal{H}$, pricing $\boldsymbol{p}_{i,S}(\tau_h)$, and computing power allocation $\beta_{i,l_k}(t)$ decisions in a step τ_h are optimized using multi-stage stochastic programming. Let $\xi_{i,S}^{st}(\tau_h)$ and $\xi_S(\tau_h)$ denote the set of the possible residence time and composite time in step τ_h. Then the set of all composite scenarios of residence times $\boldsymbol{\xi}_{i,S}^{st}$ and composite times $\boldsymbol{\xi}_S$ of all the steps are as follows.

$$\boldsymbol{\xi}_{i,S}^{st} = \prod_{h=1}^{H} \xi_{i,S}^{st}(\tau_h) = \xi_{i,S}^{st}(\tau_1) \times \cdots \times \xi_{i,S}^{st}(\tau_H) \quad (48)$$

$$\boldsymbol{\xi}_S = \prod_{h=1}^{H} \xi_S(\tau_h) = \xi_S(\tau_1) \times \cdots \times \xi_S(\tau_H). \quad (49)$$

A. Multi-Stage Stochastic Programming Formulation

With the H-step approach, a distributed stochastic programming model with $2H$ stages is formulated as follows.[5]

1) Multi-Stage Formulation for MDs/Buyers: The task offloading problem of MD/buyer i can be rewritten as the optimization problem of $\mathcal{P}3$-Buyer (multi-stage) as in (50)–(53).

$\mathcal{P}3$-Buyer (multi-stage)

$$\begin{aligned} & \max_{f_{i,L}(\tau_h), D_{i,k}(\tau_h)} Y_{b_i} \left\{ D_i\left(\tau_1\right), .., D_i\left(\tau_H\right), \right. \\ & \quad \left. f_{i,L}\left(\tau_1\right), .., f_{i,L}\left(\tau_H\right) \right\} \\ & = V_i \Big\{ u_i\left[D_i\left(\tau_1\right)\right] - c_i\left[D_i\left(\tau_1\right)\right] - e_{i,L}\left[D_i\left(\tau_1\right)\right] - s_i\left[D_i\left(\tau_1\right)\right] \\ & \quad + \mathbb{E}_{\boldsymbol{\xi}_{i,S}^{st}(\tau_h) | \boldsymbol{\xi}_{i,S}^{st}(\tau_{h-1})} \sum_{h=2}^{H} \left\{ u_i\left[D_i\left(\tau_h\right)\right] - s_i\left[D_i\left(\tau_h\right)\right] \right\} \end{aligned}$$

[5]Following the previous two-stage stochastic programming model, each step τ_h here comprises two stages (i.e., the game stage and recourse stage).

$$- \mathbb{E}_{\boldsymbol{\xi}_{i,S}^{st}(\tau_h)|\boldsymbol{\xi}_{i,S}^{st}(\tau_{h-1})} \sum_{h=2}^{H} c_i \left[D_i(\tau_h), T_{i,S}^{st}(\tau_h)\right]$$

$$- \mathbb{E}_{\boldsymbol{\xi}_{i,S}^{st}(\tau_h)|\boldsymbol{\xi}_{i,S}^{st}(\tau_{h-1})} \sum_{h=2}^{H} e_{i,L} \left[D_i(\tau_h), T_{i,S}^{st}(\tau_h)\right] \Big\}$$

$$+ Q_i(\tau_1) D_i(\tau_1) + \mathbb{E}_{\boldsymbol{\xi}_{i,S}^{st}(\tau_h)|\boldsymbol{\xi}_{i,S}^{st}(\tau_{h-1})} \sum_{h=2}^{H} Q_i(\tau_h) D_i(\tau_h)$$

$$- Q_i(\tau_1) A_i(\tau_1) - \mathbb{E}_{\boldsymbol{\xi}_{i,S}^{st}(\tau_h)|\boldsymbol{\xi}_{i,S}^{st}(\tau_{h-1})} \sum_{h=2}^{H} Q_i(\tau_h) A_i(\tau_h) \tag{50}$$

$$\text{s.t.} \quad f_{i,L}^{\min} \le f_{i,L}(\tau_h) \le f_{i,L}^{\max}, \ \tau_h \in \mathcal{H} \tag{51}$$

$$\sum_{h=1}^{H} D_i(\tau_h) = D_i(t), \ \tau_h \in \mathcal{H} \tag{52}$$

$$0 \le \sum_{h=1}^{H} D_i(\tau_h) \le Q_i(t), \ \tau_h \in \mathcal{H}. \tag{53}$$

Here $D_i(\tau_h)$ and $f_{i,L}(\tau_h)$ are the processed tasks and CPU frequency of MD i in step τ_h, respectively.

2) Multi-Stage Formulation for SBSs/Sellers: The price should be decided according to the market's environment and requirements. Obviously, a single-round pricing method may not be suitable for the stochastic and time-varying market environment, and a multi-round dynamic pricing method is designed to address the diverse user demands and time-varying environment. Similarly, we can rewrite the SBSs/sellers optimization problem as $\mathcal{P}$3-Seller (multi-stage) as in (54)–(56).

$\mathcal{P}$3-Seller (multi-stage)

$$\max_{p_{i,S}(\tau_h), \beta_{i,S}(\tau_h)} Y_{S_s}\{p_{i,S}(\tau_1), .., p_{i,S}(\tau_H), \beta_{i,S}(\tau_1), .., \beta_{i,S}(\tau_H)\}$$

$$= u_S[p_S(\tau_1)] - \sum_{i=1}^{m} \Big\{ e_{i,S}[D_{i,S}(\tau_1)] + p_{i,S}^{f}[D_{i,S}(\tau_1)] \Big\}$$

$$+ \mathbb{E}_{\boldsymbol{\xi}_S(\tau_h)|\boldsymbol{\xi}_S(\tau_{h-1})} \sum_{h=2}^{H} u_S[p_S(\tau_h)]$$

$$- \mathbb{E}_{\boldsymbol{\xi}_S(\tau_h)|\boldsymbol{\xi}_S(\tau_{h-1})} \sum_{i=1}^{m} \sum_{h=2}^{H} e_{i,S}[D_{i,S}(\tau_h), T_{i,S}^{st}(\tau_h), T_{i,S}^{wt}(\tau_h)] \tag{54}$$

$$\text{s.t.} \quad p_{i,S}(\tau_h) \ge 0, \tau_h \in \mathcal{H} \tag{55}$$

$$\mathbb{E}_{\boldsymbol{\xi}_S} \left\{ \sum_{h=1}^{H} \left[T_{i,S}^{up}(\tau_h) + T_{i,S}^{wt}(\tau_h) + \sum_{l_k \in L_k} \frac{D_i(\tau_h)\gamma_i}{\beta_{i,l_k}} \right] \right\} \le \tau_i^d \tag{56}$$

$$\beta_{l_S}^{\min} \le \beta_{i,l_S}(\tau_h) \le \beta_{l_S}^{\max}, \tau_h \in \mathcal{H}, \tag{57}$$

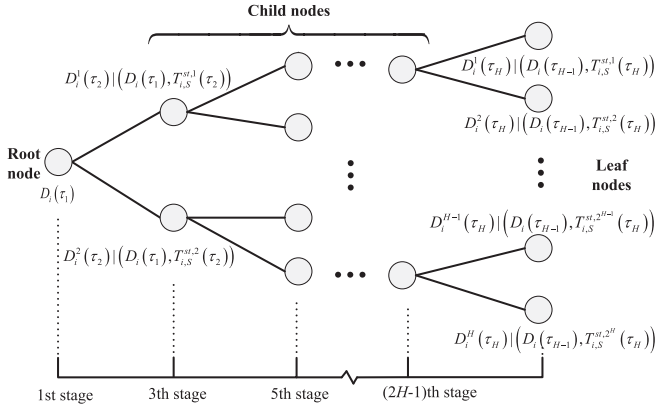


Fig. 3. Residence time scenario tree of MD i.[6]

Inequality (56) is to ensure that the sum of the computation offloading time of all the steps does not exceed the delay constraint τ_i^d.

B. Deterministic Equivalent Problem (DEP)

To solve problems $\mathcal{P}$3-Buyer (multi-stage) and $\mathcal{P}$3-Seller (multi-stage), we transform the stochastic programming formulations into their *deterministic equivalent problem* (*DEP*) by invoking the scenario tree [44], which is introduced below.

Let $\pi_{i,S}^{st} \in \boldsymbol{\xi}_{i,S}^{st}$ and $\pi_S \in \boldsymbol{\xi}_S$ be the realizations of $\boldsymbol{\xi}_{i,S}^{st}$ and $\boldsymbol{\xi}_S$, respectively. The scenario tree divides into branches according to the realizations of $\boldsymbol{\xi}_{i,S}^{st}(\tau_h)$ and $\boldsymbol{\xi}_S(\tau_h)$, $\tau_h \in \mathcal{H}$. An example of scenario tree of the residence time of MD/buyer i is shown in Fig. 3 that has two realizations to describe the evolution of $\xi_{i,S}^{st}(\tau_h), \tau_h \in \mathcal{H}$. In the residence time scenario tree, a root node is associated with the first decision stage where observation of residence time is absent. The root node is connected with child nodes associated with further stages, and each node is connected with its child nodes associated to the next stage, until the leaf nodes are reached.[7] A child node has two realizations, which are related to two random residence times $T_{i,S}^{st} : T_{i,S}^{st,1}(\tau_h), T_{i,S}^{st,2}(\tau_h)$.

After constructing the scenario tree, we next convert the buyers and sellers stochastic programming problems $\mathcal{P}$3-Buyer (multi-stage) and $\mathcal{P}$3-Seller (multi-stage) into *DEP* in the following.

1) DEP Formulation for MDs/Buyers: Let $PT_{\pi_{i,S}^{st}}$ denote the path from the root node to a leaf node in the residence time scenario tree. Once a scenario $\pi_{i,S}^{st} \in \boldsymbol{\xi}_{i,S}^{st}$ is given, $PT_{\pi_{i,S}^{st}}$ will also be determined. Let $D_i(\tau_1)$ and $D_i(\tau_h)$ be the task offloading decisions at the root node and the node of the $2\,h$th stage in path $PT_{\pi_{i,S}^{st}}$, respectively. Then, the multi-stage stochastic programming for MD/buyer i can be transformed to a DEP as follows.

$$\max_{f_{i,L}(\tau_h), D_{i,S}(\tau_h)} Y_{b_i}(t)$$

$$= V_i\{u_i[D_i(\tau_1)] - c_i[D_i(\tau_1)]$$

[7]The child nodes size of each father node is related to the realization space.

$$
\begin{aligned}
&- e_{i,L}\left[D_i\left(\tau_1\right)\right] - s_i\left[D_i\left(\tau_1\right)\right] \\
&+ \sum_{\pi_{i,S}^{st}\in\boldsymbol{\xi}_{i,S}^{st}} p\left(\pi_{i,S}^{st}\right) \sum_{h=2}^{H} \left\{ u_i\left[D_i^{\pi_{i,S}^{st}}\left(\tau_h\right)\right] - s_i\left[D_i^{\pi_{i,S}^{st}}\left(\tau_h\right)\right] \right\} \\
&- \sum_{\pi_{i,S}^{st}\in\boldsymbol{\xi}_{i,S}^{st}} p\left(\pi_{i,S}^{st}\right) \sum_{h=2}^{H} \left\{ c_i\left[D_i^{\pi_{i,S}^{st}}\left(\tau_h\right), T_{i,S}^{st,\pi_{i,S}^{st}}\left(\tau_h\right)\right] \right\} \\
&- \left. \sum_{\pi_{i,S}^{st}\in\boldsymbol{\xi}_{i,S}^{st}} p\left(\pi_{i,S}^{st}\right) \sum_{h=2}^{H} \left\{ e_{i,L}\left[D_i^{\pi_{i,S}^{st}}\left(\tau_h\right), T_{i,S}^{st,\pi_{i,S}^{st}}\left(\tau_h\right)\right] \right\} \right\} \\
&+ Q_i\left(\tau_1\right) D_i^{\pi_{i,S}^{st}}\left(\tau_1\right) - Q_i\left(\tau_1\right) A_i\left(\tau_1\right) \\
&+ \sum_{\pi_{i,S}^{st}\in\boldsymbol{\xi}_{i,S}^{st}} p\left(\pi_{i,S}^{st}\right) \sum_{h=2}^{H} \left\{ Q_i\left(\tau_h\right) D_i^{\pi_{i,S}^{st}}\left(\tau_h\right) - Q_i\left(\tau_h\right) A_i\left(\tau_h\right) \right\}
\end{aligned} \tag{58}
$$

s.t. (51)-(53)

$$
\begin{aligned}
&D_i^{\pi_{i,S}^{st}}(\tau_h) = D_i^{\pi_{i,S'}^{st}}(\tau_h),\ \forall \pi_{i,S}^{st}, \pi_{i,S}^{st} \in \xi_{i,S}^{st}, \\
&\pi_{i,S}^{st} \neq \pi_{i,S'}^{st},\ PT\left(D_i^{\pi_{i,S}^{st}}(\tau_h)\right) = PT\left(D_i^{\pi_{i,S'}^{st}}(\tau_h)\right),
\end{aligned} \tag{59}
$$

where $p(\pi_{i,S}^{st})$ is the probability of scenario $\pi_{i,S}^{st}$. Constraint (59) represents the nonanticipativity constraints, which ensures that the offloading decisions should be equivalent in different paths.

2) DEP Formulation for BSs/Sellers: Similarly, let PT_{π_S} be the path from the root node to leaf node of the composite time scenario tree. For a given scenario $\pi_S \in \boldsymbol{\xi}_S(\tau_h)$, the DEP for SBSs/sellers can be formulated as follows.

$$
\begin{aligned}
&\max_{p_{i,S}(\tau_h),\beta_{i,S}(\tau_h)} Y_{s_S}(t) \\
&= u_S\left[p_S\left(\tau_1\right)\right] - \sum_{i=1}^{m} \left\{ e_{i,S}\left[D_{i,S}\left(\tau_1\right)\right] - p_{i,S}^{f}\left[D_{i,S}\left(\tau_h\right)\right] \right\} \\
&\quad + \sum_{\pi_S\in\boldsymbol{\xi}_S} p\left(\pi_S\right) \sum_{h=2}^{H} u_S\left[p_S^{\pi_S}\left(\tau_h\right)\right] \\
&\quad - \sum_{i=1}^{m} \sum_{\pi_S\in\boldsymbol{\xi}_S} p\left(\pi_S\right) \sum_{h=2}^{H} e_{i,S} \\
&\quad \times \left[D_{i,S}^{\pi_S}\left(\tau_h\right), T_{i,S}^{st,\pi_S}\left(\tau_h\right), T_{i,S}^{qt,\pi_S}\left(\tau_h\right)\right]
\end{aligned} \tag{60}
$$

s.t. (55)- (57)

$$
\begin{aligned}
&D_i^{\pi_S}(\tau_h) = D_i^{\pi_{S'}}(\tau_h), \forall \pi_S, \pi_{S'} \in \boldsymbol{\xi}_S, \\
&\pi_S \neq \pi_{S'}, PT\left(D_i^{\pi_S}(\tau_h)\right) = PT\left(D_i^{\pi_{S'}}(\tau_h)\right),
\end{aligned} \tag{61}
$$

where $p(\pi_S)$ is the probability of scenario π_S, and $D_i^{\pi_S}(\tau_h)$ is the processed tasks in the hth stage in path PT_{π_S}.

Recall that each step τ_h comprises two stages. As shown in Fig. 4, the even numbered stages (e.g., stages $2, 4, \ldots$) will take computing power recourse actions (i.e., $\beta_{i,S}(\tau_h)$) for the odd numbered stages (e.g., stages $1, 3, \ldots$) to compensate the uncertain residence time and computation waiting delay in a step. The latter step will take recourse actions for offloaded tasks (i.e., $D_{i,S}(\tau_h)$) for the previous step to make the compensation more accurate and ensure the offloaded tasks be processed within the delay constrain.

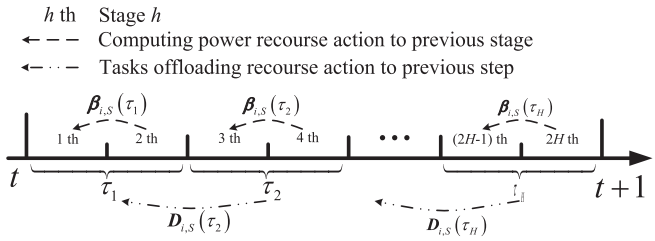


Fig. 4. Recourse and compensation illustrated.

VII. Optimal Game Strategy and Equilibrium Existence Analysis

For simplicity, we analyze the optimal game strategies in one stage, which can be easily extended to other stages in the same manner.

A. Optimal Game Strategy Analysis

1) MDs Optimal Strategy Analysis: Following (30), we derive the first-order derivatives $\frac{\partial Y_{b_i}(t)}{\partial D_{i,L}(t)}$, $\frac{\partial Y_{b_i}(t)}{\partial D_{i,M}(t)}$, and $\frac{\partial Y_{b_i}(t)}{\partial D_{i,S}(t)}$ as as in (62), (63), and (64), respectively.

$$
\frac{\partial Y_{b_i}(t)}{\partial D_{i,L}(t)} = V_i \left\{ \frac{\rho_i}{(1+D_{i,L}(t))\ln 2} \right\} + Q_i(t) \tag{62}
$$

$$
\begin{aligned}
&\frac{\partial Y_{b_i}(t)}{\partial D_{i,M}(t)} = \\
&V_i \left\{ \frac{\rho_i}{(1+D_{i,M}(t))\ln 2} - \theta_i - \frac{\lambda_i P_i(t)}{r_{i,M}(t)} - p_{i,M}(t) \right\} + Q_i(t)
\end{aligned} \tag{63}
$$

$$
\frac{\partial Y_{b_i}(t)}{\partial D_{i,S}(t)} = \begin{cases} V_i \left\{ \frac{\rho_i}{(1+D_{i,S}(t))\ln 2} - \theta_i - \frac{\lambda_i P_i(t)}{r_{i,M}(t)} - p_{i,S}(t) \right\} \\ \quad + Q_i(t), \text{ if } T_{i,S}^{st} < \frac{D_{i,S}(t)}{r_{i,S}(t)} \\ V_i \left\{ \frac{\rho_i}{(1+D_{i,S}(t))\ln 2} - \eta_i - \frac{\lambda_i P_i(t)}{r_{i,S}(t)} - p_{i,S}(t) \right\} \\ \quad + Q_i(t), \text{ if } T_{i,S}^{st} \geq \frac{D_{i,S}(t)}{r_{i,S}(t)}. \end{cases} \tag{64}
$$

Furthermore, we can easily obtain the second-order derivatives as $\frac{\partial^2 Y_{b_i}(t)}{\partial (D_{i,L}(t))^2} \leq 0$, $\frac{\partial^2 Y_{b_i}(t)}{\partial (D_{i,M}(t))^2} < 0$, and $\frac{\partial^2 Y_{b_i}(t)}{\partial (D_{i,S}(t))^2} < 0$, respectively. Since inequalities (21), (22), and (23) are affine functions, $Y_{b_i}(t)$ is convex in $D_i(t)$. Then, the optimization problem of buyers/MDs can be solved by the Lagrangian Multiplier method and the Karush-Kuhn-Tucker (KKT) conditions [48],

and the optimal strategies are given below.

$$f_{i,L}^{*}(t)=\left(\frac{\rho_i}{A_L\cdot\ln 2}-1\right)\frac{\gamma_i}{\tau_i^d},\ t\in\mathcal{T} \tag{65}$$

$$D_{i,k}^{*}(t)=\frac{\rho_i}{A_k\cdot\ln 2}-1, k\in\{M,S\}, t\in\mathcal{T}, \tag{66}$$

where $A_L=-\frac{Q_i(t)}{V_i}$, $A_M=p_{i,M}(t)+\theta_i+\frac{\lambda_i P_i(t)}{r_{i,M}(t)}-\frac{Q_i(t)}{V_i}$, and $A_S=\begin{cases}\theta_i+\frac{\lambda_i P_i(t)}{r_{i,M}(t)}+p_{i,S}(t)-\frac{Q_i(t)}{V_i}, \text{ if } T_{i,S}^{st}<\frac{D_{i,S}(t)}{r_{i,S}(t)}\\ \eta_i+\frac{\lambda_i P_i(t)}{r_{i,S}(t)}+p_{i,S}(t)-\frac{Q_i(t)}{V_i}, \text{ if } T_{i,S}^{st}\geq\frac{D_{i,S}(t)}{r_{i,S}(t)}.\end{cases}$

2) BSs Optimal Strategy Analysis: According to (41), we obtain the first-order derivative of $Y_{s_k}(t)$ with respect to $p_{i,k}(t)$ as follows.

$$\begin{aligned}\frac{\partial Y_{s_k}(t)}{\partial p_{i,k}(t)}=&D_{i,k}(t)+p_{i,k}(t)\frac{\partial D_{i,k}(t)}{\partial p_{i,k}(t)}\\&-\frac{2\lambda_k\kappa_k D_{i,k}(t)\gamma_i^2}{T_{i,k}^{pt}}\frac{\partial D_{i,k}(t)}{\partial p_{i,k}(t)}\\&-\delta_{i,k}^{f}\frac{\partial D_{i,k}(t)}{\partial p_{i,k}(t)}\cdot\mathbf{1}\left\{T_{i,k}^{co}(t)\geq\tau_i^d\right\}.\end{aligned}$$

Theorem 2: If the transaction price in the market satisfies $p_{i,k}(t)\geq 0$, we have $\frac{\partial^2 Y_{s_k}(t)}{\partial(p_{i,k}(t))^2}<0, k\in\{M,S\}$.

Proof: According to (41), we obtain the second order derivative of $Y_{s_k}(t)$ with respect to $p_{i,k}(t)$ as

$$\begin{aligned}\frac{\partial^2 Y_{s_k}(t)}{\partial(p_{i,k}(t))^2}=&-\frac{2\rho_i}{A_k^2\ln 2}+p_{i,k}(t)\frac{2\rho_i}{A_k^3\ln 2}-\frac{2\lambda_k\kappa_k\gamma_i^2}{T_{i,k}^{pt}}\frac{{\rho_i}^2}{A_k^4(\ln 2)^2}\\&-\frac{2\lambda_k\kappa_{l_k}D_{i,k}(t)\gamma_i^2}{T_{i,k}^{pt}}\frac{2\rho_i}{A_k^3\ln 2}-\delta_{i,k}^{f}\frac{2\rho_i}{A_k^3\ln 2}\cdot\mathbf{1}\left\{T_{i,k}^{co}(t)\geq\tau_i^d\right\}\\=&-\frac{2\rho_i}{A_k^3\ln 2}\left(A_k-p_{i,k}(t)\right)-\frac{2\lambda_k\kappa_{l_k}\gamma_i^2}{T_{i,k}^{pt}}\frac{{\rho_i}^2}{A_k^4(\ln 2)^2}\\&-\frac{2\lambda_k\kappa_{l_k}D_{i,k}(t)\gamma_i^2}{T_{i,k}^{pt}}\frac{2\rho_i}{A_k^3\ln 2}-\delta_{i,k}^{f}\frac{2\rho_i}{A_k^3\ln 2}\cdot\mathbf{1}\left\{T_{i,k}^{co}(t)\geq\tau_i^d\right\}.\end{aligned}$$

From the above equation, if and only if $A_k\geq p_{i,k}(t)$ is satisfied, we can obtain $\frac{\partial^2 Y_{s_k}(t)}{\partial(p_{i,k}(t))^2}<0$.

According to (65), the optimal amount of offloaded tasks $D_{i,k}(t)$ decreases as the seller's price $p_{i,k}(t)$ is increased. If MD i offloads tasks to MEC server k, it must satisfy $D_{i,k}^{*}\geq 0$, i.e., $0\leq A_k\leq\frac{\rho_i}{\ln 2}$, for all $p_{i,k}(t)\geq 0$. We have

- For the MBS, while $p_{i,M}(t)=0$, we have $0\leq\theta_i+\frac{\lambda_i P_i(t)}{r_{i,M}(t)}-\frac{Q_i(t)}{V_i}\leq\frac{\rho_i}{\ln 2}$, namely, $A_M\geq p_{i,M}(t)$.
- For the SBS, while $p_{i,S}(t)=0$, we have $0\leq\begin{cases}\theta_i+\frac{\lambda_i P_i(t)}{r_{i,M}(t)}-\frac{Q_i(t)}{V_i}, T_{i,S}^{st}<\frac{D_{i,S}(t)}{r_{i,S}(t)}\\ \eta_i+\frac{\lambda_i P_i(t)}{r_{i,S}(t)}-\frac{Q_i(t)}{V_i}, T_{i,S}^{st}\geq\frac{D_{i,S}(t)}{r_{i,S}(t)}\end{cases}\leq\frac{\rho_i}{\ln 2}$, namely, $A_S\geq p_{i,S}(t)$.

To sum up, the price must satisfy $p_{i,k}(t)\leq A_k$ and $\frac{\partial^2 Y_{s_k}(t)}{\partial(p_{i,k}(t))^2}<0$.

Since inequalities (40) and (42) are affine functions, $Y_{s_k}(t)$ is convex in $p_{i,k}(t)$. Then, the optimization problem of BSs/sellers can be solved by the Lagrangian Multiplier method and KKT conditions, and the optimal strategies are derived as follows.

$$p_{i,k}^{*}(t)=2\lambda_k\kappa_{l_k}\frac{D_{i,k}^{*}(t)\gamma_i^2}{T_{i,k}^{pt}}-\frac{D_{i,k}^{*}(t)}{\Theta_k}, \tag{67}$$

where $\Theta_k=\frac{\partial D_{i,k}^{*}(t)}{\partial p_{i,k}(t)}$ and $k\in\{M,S\}$.

B. Existence of the Stackelberg Equilibrium

We next prove that the optimal solutions $(D_{i,k}^{*}(t),p_{i,k}^{*}(t))$, $\forall k\in\{M,S\},\forall t\in\mathcal{T}$ are Stackelberg Equilibrium (SE) solutions. The SE of the proposed game is defined in Definition 1 below.

Definition 1: If the price $p_{i,k}(t)$ of seller k is determined, $D_{i,k}^{\mathrm{SE}}(t)$ satisfies

$$Y_{b_i}\left(D_{i,k}^{\mathrm{SE}}(t)\right)=\sup_{D_{i,k}^{\min}\leq D_{i,k}(t)\leq D_{i,k}^{\max}}\left\{Y_{b_i}\left(D_{i,k}(t)\right)\right\},\ \forall t\in\mathcal{T}.$$

If the offloaded task $D_{i,k}(t)$ of buyer i is determined, $p_{i,k}^{\mathrm{SE}}(t)$ satisfies

$$Y_{s_k}\left(p_{i,k}^{\mathrm{SE}}(t)\right)=\sup_{p_{i,k}(t)\geq 0}\left\{Y_{s_k}\left(p_{i,k}(t)\right)\right\},\ \forall t\in\mathcal{T}.$$

Then $D_{i,k}^{\mathrm{SE}}(t)$ and $p_{i,k}^{\mathrm{SE}}(t)$ are the SE solutions.

Now we prove that the optimal solution $(D_{i,k}^{*}(t),p_{i,k}^{*}(t))$ is $(D_{i,k}^{\mathrm{SE}}(t),p_{i,k}^{\mathrm{SE}}(t))$ by the following three lemmas.

Lemma 1: If the price $p_{i,k}(t)$ of BS/seller k is determined, the revenue function $Y_{b_i}(D_{i,k}(t))$ of the MD/buyer will take the maximum value at $D_{i,k}^{*}(t)$.

Proof: We have proved that the revenue function Y_{b_i} is convex with respect to $D_{i,k}(t)$ in Section VII-A. Consequently, $Y_{b_i}(D_{i,k}(t))$ takes the maximum value at $D_{i,k}^{*}(t)$. According to *Definition 1*, $D_{i,k}^{*}(t)$ is the SE solution $D_{i,k}^{\mathrm{SE}}(t)$.

Lemma 2: For buyers, the optimal amount of offloaded tasks $D_{i,k}^{*}(t)$ decreases as the seller's price $p_{i,k}(t)$ is increased.

Proof: According to (65), we have

$$\frac{\partial D_{i,k}^{*}(t)}{\partial p_{i,k}(t)}=-\frac{\rho_i}{{A_k}^2}\ln 2<0. \tag{68}$$

Therefore, we can shown that $D_{i,k}^{*}(t)$ is a monotonous decreasing function of $p_{i,k}(t)$. This means that if the transaction price is increased, the amount of offloaded tasks of buyers will decrease, which leads to little or even no revenue for the sellers. The sellers should adopt a suitable price to maximum its revenue. The seller's optimal price can be derived by solving $\frac{\partial Y_{s_k}(p_{i,k}(t))}{\partial p_{i,k}(t)}=0$.

Lemma 3: If the optimal amount of offloaded tasks $D_{i,k}^{*}(t)$ of MD/buyer i is determined, $Y_{s_k}(p_{i,k}(t))$ will take the maximum value at $p_{i,k}^{*}(t)$.

Proof: We have shown that the revenue of sellers Y_{s_k} is convex with respect to $p_{i,k}(t)$ in Section VII-A. Then $Y_{s_k}(p_{i,k}(t))$ takes the maximum value at $p_{i,k}^{*}(t)$. According to *Definition 1*, $p_{i,k}^{*}(t)$ is the SE solution $p_{i,k}^{\mathrm{SE}}(t)$.

In summary, $(D_{i,k}^{*}(t),p_{i,k}^{*}(t)),\forall t\in\mathcal{T}$ is the optimal offloaded tasks and price strategies, and it is also the SE solutions $(D_{i,k}^{\mathrm{SE}}(t),p_{i,k}^{\mathrm{SE}}(t)),\forall t\in\mathcal{T}$.

C. Computational Complexity Discussion

In this part, we analyze the computational complexity of the proposed algorithm both in the BS side and user side. For the MBS/SBS, the computational complexity is related to the number of split stages H, composite scenarios $\boldsymbol{\xi}_k$, and users $\mathcal{M}$. According to the *scenario tree*, the computational complexity of the MBS/SBS is $O(2^H \tilde{m}_k \cdot |\boldsymbol{\xi}_k(t)|)$, where $\tilde{m}_k$ $(0 \leq \tilde{m}_k \leq m)$ and $|\boldsymbol{\xi}_k(t)|$ are the number of users and the size of composite scenarios space in the BS k, respectively. Similarly, the computational complexity of the user i is $O(2^H \cdot |\boldsymbol{\xi}^{st}_{i,k}(t)|)$[8].

VIII. Performance Evaluation

To evaluate the efficacy and rationality of the proposed algorithm, we design two sets of simulations. First, we evaluate the performances of the proposed games. Second, we conduct a comparison study with several benchmark schemes.

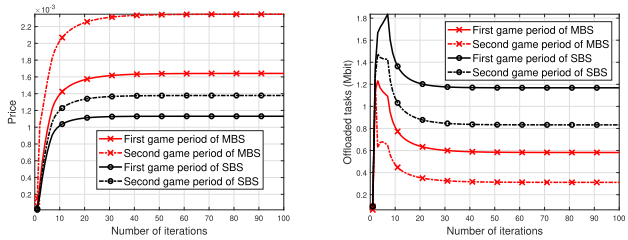
A. Simulation Setting

In the following simulations, we consider a stochastic time-varying MEC-enabled UDN system. As shown in Fig. 1, the tasks arrival rate at each MD is uniformly distributed in $[0, 20]$ Mbit/s, and computation density with parameters γ_i in $[500, 1500]$ cycles/bit. As suggested in [6] and [37], the maximum CPU clock frequency of each MD is set to 1 GHz. The residence time of each MD under SBSs follows an exponential distribution with parameter μ from 1 s to 10 s. In addition, we set $\rho_i = 2, \kappa_i = 1 \times 10^{-6} \sim 2 \times 10^{-7}, P_i(t) = 23$ dBm, $\sigma^2 = 1 \times 10^{-8}$, $\theta_i = 0.15$, $\eta_i = 0.05$, $\tau = 10$ s, $\lambda_i = 1$, and $o = 1$ Gbps. For the BSs, the bandwidth of MBS and SBSs are set to 40 MHz and 20 MHz, respectively. The maximum CPU clock frequency of MBS and SBSs are set to quad-core 2.5 GHz and quad-core 2 GHz, respectively. According to [9], let the computation waiting delay of MBS and SBSs follow an exponential distribution with parameters $\zeta_{\{M,S\}}$ in $[0, 6]$ s, and the possible waiting delay of each BSs be 10 ($|\Omega^{wt}_k| = 10$). Moreover, we set $\kappa_M = 1 \times 10^{-7}$, $\kappa_S = 1 \times 10^{-7}$, and $\lambda_M = \lambda_S = 0.5$. The price update step is set to 5×10^{-5}.

B. Performance of the Proposed Game

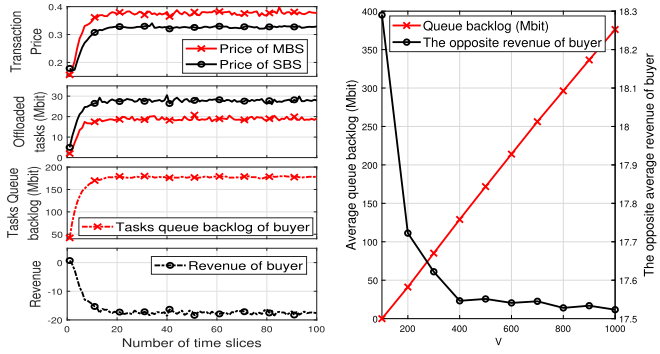
In this section, we examine the convergence speed and computation offloading performance in long-term evolution of the proposed game. For easy observation, we consider one mobile device (i.e., $m = 1$) in the region and disable its local execution option. Moreover, we set $\tau^d = 10$ s, $\mu = 5$ s, $\zeta_M = 3$ s, $\zeta_S = 1$ s, and $\kappa_i = 1 \times 10^{-7}$.

1) Convergence Speed of the Proposed Game: In this simulation, we illustrate the pricing and task offloading processes in the proposed game solved by the multi-stage stochastic programming (e.g., four stages ($H = 2$), including two game periods).



(a) Price (b) Offloaded tasks

Fig. 5. Price versus offloaded tasks ($V_i = 500, \gamma_i = 800$ cycles/bit, and $A_i(t) = 5$ Mbit/s).



(a) Computation offloading performance versus time ($V = 500$) (b) Computation offloading performance versus V

Fig. 6. Computation offloading performance ($\gamma_i = 800$ cycles/bit and $A_i(t) = 5$ Mbit/s).

Fig. 5 shows the offloaded task and price iterations of the four-stage stochastic programming, which includes two game periods. We can see from Fig. 5(a) that the iterations of price updates are non-decreasing. It takes about 40 iterations to converge to the optimal strategies (i.e., $\boldsymbol{p}^*_{i,k}(t)$). Correspondingly, the offloading strategies also reach to a steady state when the price no longer increases, which means that the buyer and sellers have reached equilibrium, as can be seen from Fig. 5(b), and the amount of offloaded tasks decreases gradually when the price is increased. Moreover, the equilibrium prices of the MBS are higher than that of the SBS in the two periods. This is because the computation waiting delay of the MBS is larger than that of the SBS, and processing the same amount of offloaded tasks requires more computing power and higher computation cost at the MBS. We can also obtain the sum of offloaded tasks of the two periods, which is less than the total amount of tasks, meaning that some tasks are backlogged in the tasks queue and are waiting to be processed.

2) Offloading Performance of the Proposed Game: Fig. 6 presents the experimental results of the computation offloading performance over time and varying V. Fig. 6(a) shows the transaction price between buyers and sellers, offloaded tasks, tasks backlog levels, and revenue of buyers. The transaction

[8]For the larger H, we can use the methods provided by the references [49] and [50] to work out the solution of the proposed multi-stage stochastic programming problem. For example, Stochastic Dual Dynamic Programming (SDDP) in [50] can cut selection procedure, and use a lower bound improvement scheme to reduce the computational complexity.

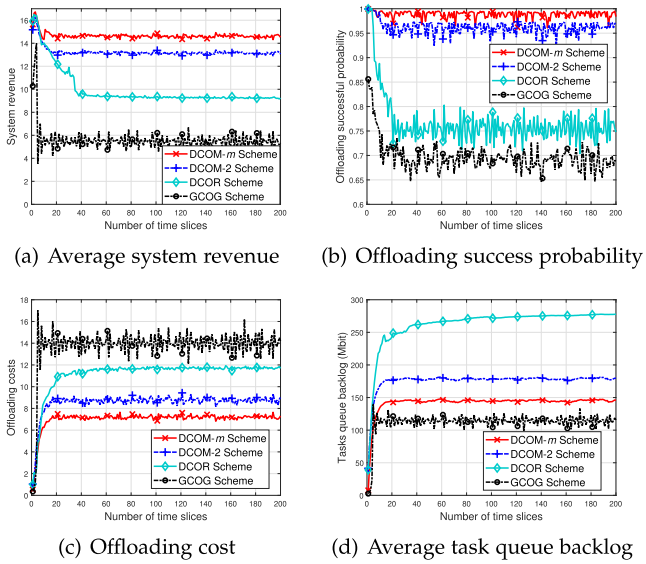
price $p_k(t)$ in the market gradually increases along with the task queue backlog of buyers, as indicated in (66) and (67). The revenue of buyers gradually decreases with increased offloading cost. Furthermore, we can see from Fig. 6(b) that there is an $[O(1/V), O(V)]$ tradeoff between the average queue backlog and revenue, as proven in [39].

C. Comparison With Benchmark Schemes

In this simulation, we conduct experiments to evaluate the proposed scheme, which is termed DCOM for **d**istributed computing and networking **c**oordination for tak **o**ffloading under **m**ultiple uncertainties, For DCOM, we evaluate the two-stage (DCOM-2) and multi-stage (DCOM-m) versions (e.g., four-stage, $m = 4$) with four MDs. We also compare the proposed schemes with the following two benchmark schemes in terms of average system revenue, offloading successful probability, average offloading cost, and average tasks queue backlog. The average system revenue is the sum of the revenue of buyers and sellers in the market, and corresponding to the weighted sum of (56) and (58). The offloading successful probability is defined as the processed completed tasks size in MEC divided by MDs' total offloaded tasks. Average offloading cost include the communication cost and computation cost of MDs. Average tasks queue backlog is the unprocessed tasks size of MDs

- The *greedy computation offloading based on game-theory* (GCOG) scheme proposed in [23]. With the GCOG scheme, a single-round pricing game is adopted, and task offloading only depends on the available computing power in the network; it does not take into account the random mobility of users and the variability of computational waiting delay. Compared with this scheme, we can show the performances of multi-round dynamic pricing method and the impact of considering both "computing + networking" in stochastic time-varying environments.
- The *dynamic computation offloading and resources allocation* (DCOR) scheme, which is a dynamic task offloading and resource allocation method. DCOR calculates the communication cost and energy cost based on the average observed residence time and waiting delay in history [25]. Compared with this scheme, we can demonstrate the benefit of the multi-stage stochastic programming with recourse and compensation.

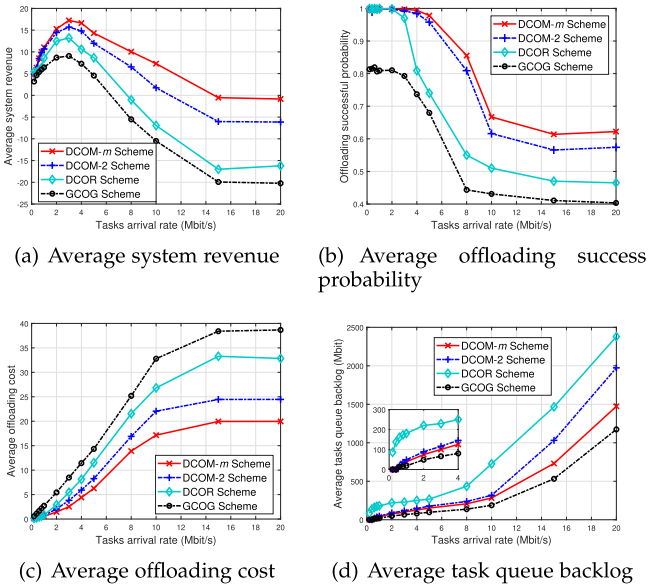
1) Performance Versus Time: Fig. 7 compares the performance of system revenue, offloading success probability, offloading cost, and task backlog level over long-term evolution. As we can see from Fig. 7(a) and (b), the system revenue and offloading success probability gradually decrease initially. This is because that more tasks are backlogged into the tasks cache over time, as can be seen from Fig. 7(d), and there are more tasks to be processed at the expense of revenue to ensure the stability of the task queue. Similarly, the offloading success probability gradually decreases until convergence. Because the DCOM schemes take both computing and networking factors into consideration, they can better optimize the task offloading decisions (such as the processed task strategies for local, MBS, or SBSs) and achieve a lower tasks backlog level. Furthermore,



(a) Average system revenue (b) Offloading success probability

(c) Offloading cost (d) Average task queue backlog

Fig. 7. Performance comparison over time ($A_i(t) = 5$ Mbit/s, $\mu_i = 5$ s, $\zeta_M = 3$ s, and $\zeta_S = 1$ s).



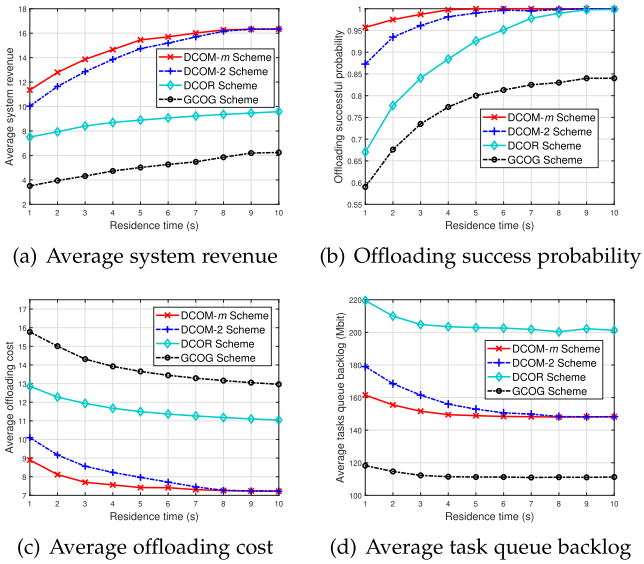
(a) Average system revenue (b) Average offloading success probability

(c) Average offloading cost (d) Average task queue backlog

Fig. 8. Performance comparison under increased tasks arrival rate ($\mu_i = 5$ s, $\zeta_M = 3$ s, and $\zeta_S = 1$ s).

the performances of the DCOM-m scheme is better than that of DCOM-2 since the former can capture the network statistical characteristics more accurately and find better computation offloading strategies.

2) Varying Task Arrival Rate: Fig. 8 presents a performance comparison of the four schemes under different task arrival rates, by considering the average system revenue, offloading success probability, average offloading cost, and average tasks backlog level. As we can see from Fig. 8(a), the average system revenue increases first, gradually decreases, and finally stabilizes when the task arrival rate is further increased. The maximum revenue is achieved when the task arrival rate is about 3 Mbit/s.



(a) Average system revenue (b) Offloading success probability

(c) Average offloading cost (d) Average task queue backlog

Fig. 9. Performance comparison under increased average residence time ($A_i(t) = 5$ Mbit/s and $\zeta_k = 0$ s).

Moreover, a smaller task arrival rate (e.g., $A_i(t) \leq 2$ Mbit/s) has almost no effect on the offloading success probability, as can be seen from Fig. 8(b). The offloading success probability then gradually decreases until it stabilizes. This is because that the offloaded tasks gradually increase as the arrival rate is increased; it approaches the maximum value when the tasks arrival task rate is greater than 15 Mbit/s. Correspondingly, as we can see from Fig. 8(c) and (d), the average offloading cost gradually increases until it stabilizes, and the tasks queue backlog also gradually increases when the task arrival rate is increased.

3) Varying Residence Time: Fig. 9 compares the performance of average system revenue, offloading success probability, offloading cost, and tasks queue backlog of the four schemes. We can see from Fig. 9(a) and (b) that the system revenue and offloading success probability gradually increase as the average residence time is increased. Fig. 9(c) shows that the offloading cost gradually decreases. This is because that communication cost gradually decreases when the average residence time is increased. Furthermore, we can see from Fig. 9(d) that the task queue backlog also gradually decreases. It should be noted that although the task backlog level of the GCOG scheme is the lowest, its offloading failure probability is the highest, which is unsatisfactory.

IX. Conclusion

In this paper, we developed a distributed joint tasks offloading and computing resource allocation methodology for MEC-enabled UDNs. The goal was to improve the computation revenue and offloading success probability based on computing and networking coordination under uncertainties in the wireless network environment. In order to achieve distributed task offloading and adaptive computing power management in the time-varying environment, a multi-round pricing method was designed by applying an extended game-theoretic approach based on Lyapunov optimization, which determines the computing power price dynamically by balancing the offloading revenue and queue backlog. Moreover, considering the stochastic residence time and computation waiting delay, a distributed two-stage algorithm and a multi-stage stochastic programming algorithm were proposed, and the multi-stage stochastic programming problem was transformed into a deterministic equivalent problem using a scenario tree to decide the optimal computation offloading strategies. The superior performance of the proposed algorithms were validated with simulations and comparison with two benchmark schemes.

References

[1] W. Shu and Y. Li, "Joint offloading strategy based on quantum particle swarm optimization for MEC-enabled vehicular networks," *Digit. Commun. Netw.*, vol. 9, no. 1, pp. 56–66, 2023.

[2] G. Perin, F. Meneghello, R. Carli, L. Schenato, and M. Rossi, "EASE: Energy-aware job scheduling for vehicular edge networks with renewable energy resources," *IEEE Trans. Green Commun. Netw.*, vol. 7, no. 1, pp. 339–353, Mar. 2023.

[3] Y. Zhang, L. Jia, N. Qi, Y. Xu, and M. Wang, "Anti-jamming channel access in 5G ultra-dense networks: A game-theoretic learning approach," *Digit. Commun. Netw.*, vol. 9, no. 2, pp. 523–533, 2023.

[4] M. Elbayoumi, M. Kamel, W. Hamouda, and A. Youssef, "NOMA-assisted machine-type communications in UDN: State-of-the-art and challenges," *IEEE Commun. Surveys Tut.*, vol. 22, no. 2, pp. 1276–1304, Second Quarter 2020.

[5] H. Guo, J. Liu, and J. Zhang, "Computation offloading for multi-access mobile edge computing in ultra-dense networks," *IEEE Commun. Mag.*, vol. 56, no. 8, pp. 14–19, Aug. 2018.

[6] Y. Sun, S. Zhou, and J. Xu, "EMM: Energy-aware mobility management for mobile edge computing in ultra dense networks," *IEEE J. Sel. Areas Commun.*, vol. 35, no. 11, pp. 2637–2646, Nov. 2017.

[7] I. Labriji et al., "Mobility aware and dynamic migration of MEC services for the Internet of Vehicles," *IEEE Trans. Netw. Service Manag.*, vol. 18, no. 1, pp. 570–584, Mar. 2021.

[8] S. Xia, Z. Yao, Y. Li, and S. Mao, "Online distributed offloading and computing resource management with energy harvesting for heterogeneous MEC-enabled IoT," *IEEE Trans. Wireless Commun.*, vol. 20, no. 10, pp. 6743–6757, Oct. 2021.

[9] J. Cao, W. Feng, N. Ge, and J. Lu, "Delay characterization of mobile-edge computing for 6G time-sensitive services," *IEEE Internet Things J.*, vol. 8, no. 5, pp. 3758–3773, Mar. 2021.

[10] Y. Deng, Z. Chen, and X. Chen, "Resource allocation for multi-user mobile-edge computing systems with delay constraints," in *Proc. IEEE Glob. Commun. Conf.*, Taipei, Taiwan, China, 2020, pp. 1–6.

[11] P. Porambage, J. Okwuibe, M. Liyanage, M. Ylianttila, and T. Taleb, "Survey on multi-access edge computing for Internet of Things realization," *IEEE Commun. Surveys Tut.*, vol. 20, no. 4, pp. 2961–2991, Fourth Quarter 2018.

[12] P. Ranaweera, A. D. Jurcut, and M. Liyanage, "Survey on multi-access edge computing security and privacy," *IEEE Commun. Surveys Tut.*, vol. 23, no. 2, pp. 1078–1124, Second Quarter 2021.

[13] R. Deng, R. Lu, C. Lai, T. H. Luan, and H. Liang, "Optimal workload allocation in fog-cloud computing toward balanced delay and power consumption," *IEEE Internet Things J.*, vol. 3, no. 6, pp. 1171–1181, Dec. 2016.

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

[15] S. Guo, J. Liu, Y. Yang, B. Xiao, and Z. Li, "Energy-efficient dynamic computation offloading and cooperative task scheduling in mobile cloud computing," *IEEE Trans. Mobile Comput.*, vol. 18, no. 2, pp. 319–333, Feb. 2019.

[16] X. Chen, H. Zhang, C. Wu, S. Mao, Y. Ji, and M. Bennis, "Optimized computation offloading performance in virtual edge computing systems via deep reinforcement learning," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 4005–4018, Jun. 2019.

[17] I. H. Hou, V. Borkar, and P. R. Kumar, "A theory of QoS for wireless," in *Proc. IEEE INFOCOM*, Rio de Janeiro, Brazil, 2009, pp. 486–494.
[18] Y. Teng, M. Liu, F. R. Yu, V. C. M. Leung, M. Song, and Y. Zhang, "Resource allocation for ultra-dense networks: A survey, some research Issues and challenges," *IEEE Commun. Surveys Tut.*, vol. 21, no. 3, pp. 2134–2168, Third Quarter 2019.
[19] N. Liakopoulos, G. S. Paschos, and T. Spyropoulos, "Robust optimization framework for proactive user association in UDNs: A data-driven approach," *IEEE/ACM Trans. Netw.*, vol. 27, no. 4, pp. 1683–1695, Aug. 2019.
[20] G. Kwon and H. Park, "Joint user association and beamforming design for millimeter wave UDN with wireless Backhaul," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 12, pp. 2653–2668, Dec. 2019.
[21] Y. Xu, W. Xu, Z. Wang, J. Lin, and S. Cui, "Load balancing for ultradense networks: A deep reinforcement learning-based approach," *IEEE Internet Things J.*, vol. 6, no. 6, pp. 9399–9412, Dec. 2019.
[22] S. Yu, X. Chen, Z. Zhou, X. Gong, and D. Wu, "When deep reinforcement learning meets federated learning: Intelligent multitimescale resource management for multiaccess edge computing in 5G ultradense network," *IEEE Internet Things J.*, vol. 8, no. 4, pp. 2238–2251, Feb. 2021.
[23] H. Guo, J. Liu, J. Zhang, W. Sun, and N. Kato, "Mobile-edge computation offloading for ultradense IoT networks," *IEEE Internet Things J.*, vol. 5, no. 6, pp. 4977–4988, Dec. 2018.
[24] Q. Zhang, L. Gui, F. Hou, J. Chen, S. Zhu, and F. Tian, "Dynamic task offloading and resource allocation for mobile-edge computing in dense cloud RAN," *IEEE Internet Things J.*, vol. 7, no. 4, pp. 3282–3299, Apr. 2020.
[25] X. Deng, J. Li, L. Shi, Z. Wei, X. Zhou, and J. Yuan, "Wireless powered mobile edge computing: Dynamic resource allocation and throughput maximization," *IEEE Trans. Mobile Comput.*, vol. 21, no. 6, pp. 2271–2288, Jun. 2022.
[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] A. Asheralieva and D. Niyato, "Game theory and Lyapunov optimization for cloud-based content delivery networks with device-to-device and UAV-enabled caching," *IEEE Trans. Veh. Technol*, vol. 68, no. 10, pp. 10094–10110, Oct. 2019.
[28] Y. Zhu, J. Li, Q. Huang, and D. Wu, "Game theoretic approach for network access control in heterogeneous networks," *IEEE Trans. Veh. Technol*, vol. 67, no. 10, pp. 9856–9866, Oct. 2018.
[29] X. Li, Q. Huang, and D. Wu, "A repeated stochastic game approach for strategic network selection in heterogeneous networks," in *Proc. IEEE Conf. Comput. Commun. Workshops*, Honolulu, HI, USA, 2018, pp. 88–93.
[30] H. Dinh, C. Lee, D. Niyato, and P. Wang, "A survey of mobile cloud computing: Architecture, applications, and approaches," *Wirel. Commun. Mobile Comput.*, vol. 13, no. 18, pp. 1587–1611, 2013.
[31] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief, "A survey on mobile edge computing: The communication perspective," *IEEE Commun. Surveys Tut.*, vol. 19, no. 4, pp. 2322–2358, Fourth Quarter 2017.
[32] A. P. Miettinen and J. K. Nurminen, "Energy efficiency of mobile clients in cloud computing," in *Proc. USENIX Conf. Hot Top. Cloud Comput.*, Boston, MA, USA, 2010, pp. 1–7.
[33] J. M. Rabaey, A. Chandrakasan, and B. Nikoli, *Digital Integrated Circuits: A Design Perspective*, 2nd ed. Upper Saddle River, NJ, USA: Prentice-Hall, 2003.
[34] K. Son and B. Krishnamachari, "SpeedBalance: Speed-scaling-aware optimal load balancing for green cellular networks," in *Proc. IEEE INFOCOM*, Orlando, FL, USA, 2012, pp. 2816–2820.
[35] Y. Mao, J. Zhang, S. H. Song, and K. B. Letaief, "Stochastic joint radio and computational resource management for multi-user mobile-edge computing systems," *IEEE Trans. Wireless Commun.*, vol. 16, no. 9, pp. 5994–6009, Sep. 2017.
[36] Q. D. La, M. V. Ngo, T. Q. Dinh, T. Q. Quek, and H. Shin, "Enabling intelligence in fog computing to achieve energy and latency reduction," *Digit. Commun. Netw.*, vol. 5, no. 1, pp. 3–9, 2019.
[37] D. Niyato, E. Hossain, and P. Wang, "Optimal channel access management with QoS support for cognitive vehicular networks," *IEEE Trans. Mobile Comput.*, vol. 10, no. 4, pp. 573–591, Apr. 2011.
[38] T. Truong-Huu, C. K. Tham, and D. Niyato, "A stochastic workload distribution approach for an ad-hoc mobile cloud," in *Proc. IEEE 6th Int. Conf. Cloud Comput. Technol. Sci.*, Singapore, 2014, pp. 174–181.
[39] Y. Li, S. Xia, M. Zheng, B. Cao, and Q. Liu, "Lyapunov optimization based trade-off policy for mobile cloud offloading in heterogeneous wireless networks," *IEEE Trans. Cloud Comput.*, vol. 10, no. 1, pp. 491–505, First Quarter 2022.
[40] L. Chen, C. Shen, P. Zhou, and J. Xu, "Collaborative service placement for edge computing in dense small cell networks," *IEEE Trans. Mobile Comput.*, vol. 20, no. 2, pp. 377–390, Feb. 2021.
[41] Y. Ma, W. Liang, M. Huang, W. Xu, and S. Guo, "Virtual network function service provisioning in MEC via trading off the usages between computing and communication resources," *IEEE Trans. Cloud Comput.*, vol. 10, no. 4, pp. 2949–2963, Fourth Quarter 2022.
[42] C.-K. Tham and B. Cao, "Stochastic programming methods for workload assignment in an ad hoc mobile cloud," *IEEE Trans. Mobile Comput.*, vol. 17, no. 7, pp. 1709–1722, Jul. 2018.
[43] B. Cao, S. Xia, J. Han, and Y. Li, "A distributed game methodology for crowdsensing in uncertain wireless scenario," *IEEE Trans. Mobile Comput.*, vol. 19, no. 1, pp. 15–28, Jan. 2020.
[44] J. R. Birge and F. Louveaux, *Introduction to Stochastic Programming*, New York, NY, USA: Springer Press, 2011, pp. 128–136.
[45] Neely and J. Michael, "Stochastic network optimization with application to communication and queueing systems," *Synth. Lectures Commun. Netw.*, vol. 3, no. 1, pp. 1–211, 2010.
[46] L. Sun, L. Wan, and X. Wang, "Learning-based resource allocation strategy for industrial IoT in UAV-enabled MEC systems," *IEEE Trans. Ind. Informat.*, vol. 17, no. 7, pp. 5031–5040, Jul. 2021.
[47] Y. Cao, X. Zhang, B. Zhou, X. Duan, D. Tian, and X. Dai, "MEC intelligence driven electro-mobility management for battery switch service," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 7, pp. 4016–4029, Jul. 2021.
[48] M. S. Barzaraa, *Nonlinear Programming: Theory and Algorithms*, 2nd ed. Hoboken, NJ, USA: Wiley, 1993.
[49] J. R. Birge and F. Louveaux, *Introduction to Stochastic Programming*, New York, NY, USA: Springer Press, 2011, pp. 128–136.
[50] A. Bhattacharya, J. P. Kharoufeh, and B. Zeng, "Managing energy storage in microgrids: A multistage stochastic programming approach," *IEEE Trans. Smart Grid*, vol. 9, no. 1, pp. 483–496, Jan. 2018.