Joint Parallel Offloading and Load Balancing for Cooperative-MEC Systems with Delay Constraints

Wenqian Zhang, Guanglin Zhang, *Member, IEEE*, and Shiwen Mao, *Fellow, IEEE*

***Abstract*—Mobile-edge computing (MEC) has been recognized as a promising solution to provide efficient communication and computation capabilities for mobile users (MUs). However, the problem relating to parallel offloading and load balancing with multiple cooperative MEC servers and massive delay-sensitive execution workloads remains to be investigated. In this paper, we study a joint parallel offloading and load balancing policy for such an MEC system. We formulate a long-term system average-cost (i.e., weighted sum of energy consumption and execution delays) stochastic programming problem under MU battery level stability and delay constraints. Our aims include optimizing the data sizes for uplink offloading transmissions for optimized communication and computation resource allocation among multi-users, and maximizing the computation capability utilization of cooperative MEC servers by load balancing. To solve this problem, we first design a Lyapunov-based centralized cost management algorithm (LYP-CCMA) to obtain the optimal system average-cost under the battery level stability constraints. Further, we propose two algorithms based on alternating direction method of multipliers (ADMM) to implement distributed resources allocation. Simulation results verify our analysis and demonstrate the superior performance of our proposed schemes over several baseline schemes.**

***Index Terms*—Mobile-edge computing (MEC), online dynamic parallel offloading strategy, alternating direction method of multipliers (ADMM), cooperative multiple servers, delay constraints.**

I. Introduction

As innovative mobile services proliferate, such as online games, augmented reality, and autonomous driving, there is a quickly growing need for delay-sensitive services [1] and more requirements for mobile devices. Mobile-edge computing (MEC) has been proposed as a new computation paradigm [2], which pushes cloud computation functionality from network center to the edge of network and in proximity to end-users, thereby overcoming the basic latency limitations of centralized cloud [3] and the limited computation capability of mobile devices. At the same time, energy harvesting (EH) technology provides an effective guarantee for battery supply and alleviates the capacity limitation of mobile devices' battery, which is to harvest green energy from the environment, such as solar and wind, etc. [4].

MEC servers can be deployed at or near the base station, and share similarities with cellular base stations or wireless access points with cloud-like computation and storage capabilities, providing communication and computation services to MUs [5]. Compared with the central cloud with powerful services capabilities, MEC servers are limited in services resources; thus only a limited number of MUs can be served by an MEC server [6]. In addition, varying human activity and requests of workloads usually cause a sudden increase in the amount of workloads at some MEC servers, resulting in imbalanced workloads among MEC servers [7]. Since the workload arrivals are highly dynamic with different execution requirements, it is difficult for a single MEC server to satisfy the execution requirements of all arrived workloads at all times [8]. In order to overcome these issues, the geographical load balancing (GLB) technique is proposed to enhance the MEC performance. GLB makes full use of the computation resources by re-shaping the workload distribution in the system and achieving load balancing among MEC servers [9]. In addition, in many applications, workloads can usually be partitioned into non-overlapping subsets and executed by the MU, an MEC server, or the centralized remote cloud server. This approach is termed *parallel offloading* in this paper, which takes advantage of the parallelism between MU and MEC server to achieve a better uplink offloading performance [10]. Such an MEC architecture can greatly improve the service performance by serving the massive and delay-sensitive workloads of MUs.

To alleviate the limitations of traditional MEC architecture, it is crucial to jointly optimize the allocation of communication and computation resources in the MEC system with multi-users and multiple MEC servers. Most existing studies focused on the allocation of bandwidth and computation resources among an MEC server and multi-users [11], [12], the uplink offloading optimization, energy consumption savings [13], and execution delay constraints [9]. These prior works investigated MEC system in different aspects, but ignoring the load balancing and computation cooperation among MEC servers. Only few prior works have focused on the load balancing among MEC servers [8], [14]. In these prior works, the busy MEC server with heavy workloads can forward their workloads to neighboring MEC servers with residual computation resources,

This work was supported in part by the National Natural Science Foundation of China 62072096, the Fundamental Research Funds for the Central Universities under Grant 2232020A-12, the International S&T Cooperation Program of Shanghai Science and Technology Commission under Grant 20220713000, the Young Top-notch Talent Program in Shanghai, the "Shuguang Program" of Shanghai Education Development Foundation and Shanghai Municipal Education Commission, the Fundamental Research Funds for the Central Universities and Graduate Student Innovation Fund of Donghua University CUSF-DH-D-2019093. S. Mao's work is supported in part by the NSF under Grant ECCS-1923717.

W. Zhang and G. Zhang are with the College of Information Science and Technology, Donghua University; Engineering Research Center of Digitized Textile and Apparel Technology, Ministry of Education, Shanghai, China. (email: zhangwenqian@mail.dhu.edu.cn, glzhang@dhu.edu.cn). S. Mao is with the Department of Electrical and Computer Engineering, Auburn University, Auburn, AL 36849 USA. (e-mail: smao@ieee.org).

DOI: 10.1109/TVT.2022.3143425

thus load balancing and maximum utilization of computation capability are realized among MEC servers.

Integrating parallel offloading and load balancing into the cooperative MEC system with delay constraints and varying task requirements has become a considerable challenging problem. Because the parallel offloading and the load balancing decisions among multiple cooperative MEC servers are coupled tightly and time varying. Performing optimal parallel offloading and load balancing while achieving optimal communication and computation resource allocation introduces another dimension of complexity. Therefore, how to implement parallel offloading and load balancing jointly for cooperative-MEC systems with delay constraints is still a challenging and open problem.

In this paper, we consider a three-tier cooperative MEC system for communication and computation resources allocation among MUs, MEC servers, and centralized remote cloud. The cooperative MEC servers can exchange workloads to each other to achieve load balancing through a Local Area Network (LAN) [8] and maximize the utilization of computation resources of MEC servers. The communication and parallel offloading between MEC servers and multi-users are through wireless channels. We consider the workloads that allow parallel offloading, and the mobile devices are capable of harvesting renewable energy from the environment. In particular, the following contributions are made in this work:

- We investigate the problem of joint parallel offloading and load balancing optimization in a cooperative-MEC system with multiple different processing tasks, we formulate a long-term average system cost stochastic programming problem under battery level stability and delay constraints of MUs. Particularly, the parallel offloading decisions are related to time-coupled battery dynamics. The parallel offloading decisions and load balancing decisions are related to the system state and coupled over time.
- We first design an LYP-CCMA algorithm for load balancing among MEC servers to achieve the optimal system average-cost under the battery level stability constraints. Moreover, we propose two algorithms based on ADMM for parallel offloading to make the communication and computation resources allocation decisions among the MEC servers and MUs.
- We analyze the feasibility and the performance of our proposed LYP-CCMA and ADMM-DRAR algorithms rigorously. Compared with existing works in four scenarios, our proposed algorithms achieve a smaller time averaged system cost under execution delay constraints of MUs, and maximize the computation capabilities utilization. The effectiveness of the proposed schemes is verified by our simulation study.

The rest of this paper is organized as follows. In Sections II and III, we depict the system model and problem formulation, respectively. In Section IV, we design the online centralized algorithm, LYP-CCMA, for load balancing. In Section V, we propose the ADMM-DRAR algorithm for resources allocation. In Section VI, we verify the feasibility of our proposed MEC system and evaluate the performance of our proposed algorithm by systematic simulations. We discuss related works in Section VII and conclude the paper in Section VIII.

TABLE I: Notation

Notation	*Definition*
s_k	The input data size (in bits/workload) of each workload related to task $I_{u,n}^{(k)}(t)$
h_k	The required service capability (in CPU cycles/bit) to process the task $I_{u,n}^{(k)}(t)$
$\mu_{u,n}^{(k)}$	The service capacity (in bit/second) of local CPU for processing the task $I_{u,n}^{(k)}(t)$
$\mu_n^{(k)}$	The service capacity (in workloads/second) allocated by MEC server n for the task $I_{u,n}^{(k)}(t)$
W_n	The overall uplink bandwidth between MEC server n and MUs in its coverage area
$\mathcal{T}$	Index set of time slots
$\mathcal{N}$	Index set of MEC servers
$\mathcal{M}_n$	Index set of the neighbor MEC servers of MEC server n
$\boldsymbol{\mathcal{U}}$	Index set of MUs
$\mathcal{U}_n$	Index set of MUs served by MEC server n
$\mathcal{K}$	Index set of the types of processing tasks
$\boldsymbol{\alpha}_{u,n}^{(k)}(t)$	Index set of parallel offloading decisions for MU u in coverage area of MEC server n in time slot t
$\boldsymbol{\beta}_n(t)$	Set of load balancing decisions of MEC server n in time t
$A_{u,n}^{(k)}(t)$	Workload arrivals rate (in workload/second) of task $I_{u,n}^{(k)}(t)$ at MU u in time slot t
$D_k^{\max}$	The maximum execution delay deadline (in second) of task $I_{u,n}^{(k)}(t)$
$f_{u,n}^{(k)}(t)$	The service capability (in CPU cycles/second) allocated by MU u to process the task $I_{u,n}^{(k)}(t)$ in time t
$D_{l,u,n}^{(k)}(t)$	The local execution delay of MU u in time slot t
$E_{l,u,n}^{(k)}(t)$	Energy consumption for local execution of MU u in time t
$r_{u,n}(t)$	Achievable transmission rate for uplink offloading in time t
$\omega_{u,n}(t)$	The allocated bandwidth for MU u in time slot t
$D_{tx,u,n}^{(k)}(t)$	The offloading transmission delay from MU u to MEC server n in time slot t
$E_{o,u,n}^{(k)}(t)$	The uplink offloading energy consumption from MU u to MEC server n in time slot t
$\lambda_n(t)$	The overall output workloads of MEC server n in time t
$\nu_n^{(k)}(t)$	Total workloads of the task $I_{u,n}^{(k)}(t)$ to be processed at MEC server n in time t
$B_{u,n}(t)$	Battery energy level of MU u in time slot t
$H_{u,n}(t)$	Renewable energy packets arrive at MU u in time slot t
$G_{u,n}(t)$	Energy harvesting decision of MU u in time slot t

II. System Model

A. Multi-MEC Server System

As depicted in Fig. 1, we consider an MEC system operating in discretized time with a centralized remote cloud, multiple MEC servers, and multiple MUs. The set of time slots is denoted by $\mathcal{T} \triangleq \{0, 1, ..., \}$ and the duration of each time slot is τ. We assume that the coverage areas of the MEC servers consist of fully covered and disjoint regions, and each MEC server serves MUs in its coverage area. The centralized cloud has high service capability can meet the service requirements of MUs. The MEC servers have finite computation capabilities, while an overloaded MEC server can transfer some of its workloads to its neighboring MEC servers with relatively lighter loads. In addition, the workloads generated by MUs in a time slot can be partitioned into non-overlapping subsets to be executed by the MU itself, by an MEC server, or further transmitted to a centralized remote cloud, respectively (which is termed *parallel offloading* in this paper).

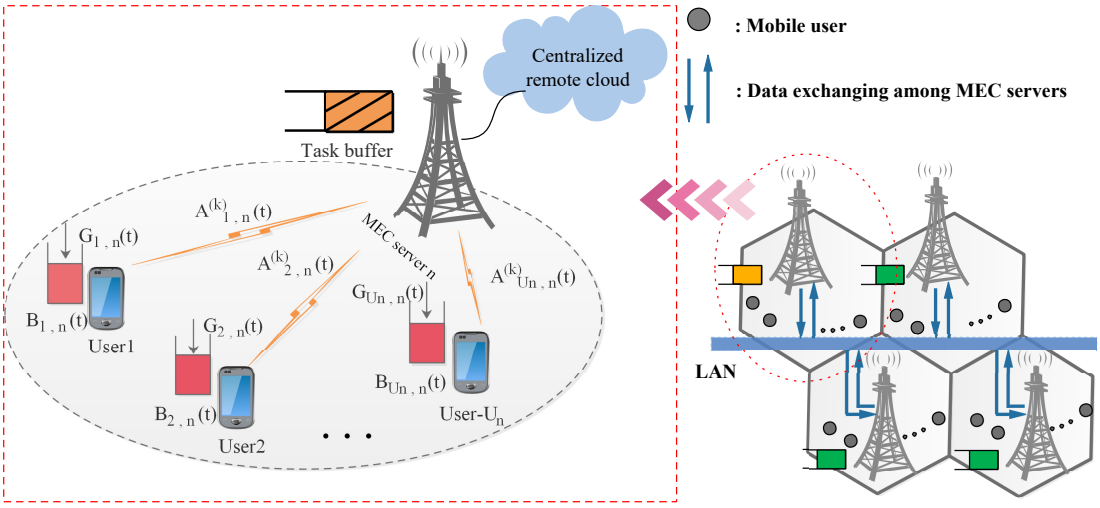

Fig. 1: Architecture of an MEC system with multiple cooperative MEC servers and MUs.

In order to enable communications and achieve load balancing among MEC servers, the MEC system is enabled with the Software Defined Network (SDN) technology with SDN controllers [15]. The SDN controllers can collect a global view of the system state (i.e., the generated workloads, the current computation capabilities of MEC servers and MUs, and the state of wireless channels and LAN links) at the beginning of each time slot. And then make optimal parallel offloading and load balancing decisions at each time slot. In addition, the notation used in this paper is summarized in Table I.

Assume there are N MEC servers. Let $\mathcal{N} \triangleq \{1, 2, ..., N\}$ denote the set of all MEC servers in the system, $\mathcal{M}_n \subseteq \mathcal{N}$ is the set of neighboring MEC servers of MEC server n. Each MEC server serves a set of MUs in its coverage area, denoted by $\mathcal{U}_n = \{1, 2, ..., U_n\}$. Let $\mathcal{U}'_n \subseteq \mathcal{U}_n$ denote the set of MUs that have workloads executed by MEC server n at a time slot. We denote $\boldsymbol{\mathcal{U}} = \{\mathcal{U}_n\}_{n\in\mathcal{N}}$ as the set of all MUs in the network.

B. Workload Arrival and Parallel Offloading Decision

We consider that the cooperative-MEC system has multiple different types of tasks. Similar to [16], we consider that each MU u has K processing tasks, defined as $\mathcal{K} = \{1, 2, \ldots, K\}$. We denote $I^{(k)}_{u,n}(t)$ as the kth processing task of MU u. We assume that each MU u has a computationally intensive task $I^{(k)}_{u,n}(t)$ to process at time slot t, represented by a variable tuple $I^{(k)}_{u,n}(t) \triangleq \{Q^{(k)}_{u,n}(t), h_k, D^{\max}_k\}$. Here $Q^{(k)}_{u,n}(t) = \tau A^{(k)}_{u,n}(t) s_k$ (in bits) denotes the data size of the task $I^{(k)}_{u,n}(t)$, $A^{(k)}_{u,n}(t)$ (in workload/second) is workload arrivals rate of task $I^{(k)}_{u,n}(t)$ in time slot t, s_k (in bits/workload) is the data size of each workload related to task $I^{(k)}_{u,n}(t)$, and τ (in second) is the duration of each time slot. As in [17], we assume workloads arrive at the beginning of the time slot. We follow the general assumption of workload arrival in MEC systems, that the workload arrivals at MU u of task $I^{(k)}_{u,n}(t)$ covered by MEC server n follow a Poisson process with rate $A^{(k)}_{u,n}(t) \in [0, A^{(k),\max}_{u,n}]$ in time slot t [18]. In addition, h_k (in CPU cycles/bit) and $D^{\max}_k$ (in second) are the required service capability and the maximum execution delay deadline of task $I^{(k)}_{u,n}(t)$, respectively.

The workloads $A^{(k)}_{u,n}(t)$ at MU u in time slot t are partitioned into non-overlapping subsets and executed by MU u, MEC server n or a neighboring MEC server in $\mathcal{M}_n$, or further transmitted to a centralized remote cloud, respectively [10]. For simplicity and without loss of generality, similar to [10] and [19], we denote $\boldsymbol{\alpha}^{(k)}_{u,n}(t) = \{\alpha^{(k)}_{l,u,n}(t), \alpha^{(k)}_{o,u,n}(t), \alpha^{(k)}_{r,u,n}(t)\} (\in [0, 1])$ as the parallel offloading decision of MU u in the coverage area of MEC server n in time slot t, indicating the fraction of workloads for local execution ($\alpha^{(k)}_{l,u,n}(t)$), for cooperative execution by the MEC servers ($\alpha^{(k)}_{o,u,n}(t)$), and for centralized remote cloud execution ($\alpha^{(k)}_{r,u,n}(t)$). We have $\alpha^{(k)}_{l,u,n}(t) + \alpha^{(k)}_{o,u,n}(t) + \alpha^{(k)}_{r,u,n}(t) = 1$. We assume that if the communication and computation resources of the MU and MEC servers are insufficient to meet the execution delay deadline of arrived workloads in a time slot, some workloads have to be further transmitted to a centralized remote cloud to be executed there.

C. Local Execution

The data size of task $I^{(k)}_{u,n}(t)$ generated by MU u for local execution in time slot t is $\alpha^{(k)}_{l,u,n}(t)\tau A^{(k)}_{u,n}(t)s_k$. We denote $f^{(k)}_{u,n}(t)$ (in CPU cycles/second) as the service capability allocated by MU u to process the task $I^{(k)}_{u,n}(t)$ in time t, which is bounded by the maximum value $f^{(k),\max}_{u,n}$. The power consumption for MU u's local execution is $P^{(k)}_{l,u,n}(t) = \xi[f^{(k)}_{u,n}(t)]^3$, where ξ is a constant coefficient related to the CPU chip architecture [20]. As in [8], the local execution delay of MU u for task $I^{(k)}_{u,n}(t)$ at time t is given by

$$D^{(k)}_{l,u,n}(t) = \alpha^{(k)}_{l,u,n}(t)\tau A^{(k)}_{u,n}(t)s_k/\mu^{(k)}_{u,n}, \tag{1}$$

where $\mu^{(k)}_{u,n} = f^{(k)}_{u,n}(t)/h_k$ is modeled as the service capacity (in bit/second) of the local CPU for processing the task $I^{(k)}_{u,n}(t)$. The local execution energy consumption of MU u for task $I^{(k)}_{u,n}(t)$ in time slot t is denoted as $E^{(k)}_{l,u,n}(t) = P^{(k)}_{l,u,n}(t) \cdot D^{(k)}_{l,u,n}(t)$ [21], which is given by

$$E^{(k)}_{l,u,n}(t) = \alpha^{(k)}_{l,u,n}(t)\tau A^{(k)}_{u,n}(t)s_k h_k \xi[f^{(k)}_{u,n}(t)]^2. \tag{2}$$

The overall local execution delay in the coverage area of MEC server n is

$$D_{n,l}(t) = \sum_{u\in\mathcal{U}_n}\sum_{k\in\mathcal{K}} D^{(k)}_{l,u,n}(t). \tag{3}$$

D. Cooperative Execution by MEC Servers

The data size of the task $I^{(k)}_{u,n}(t)$ generated by MU u that are offloaded to MEC server n for cooperative execution is $\alpha^{(k)}_{o,u,n}(t)\tau A^{(k)}_{u,n}(t)s_k$. The arrived workloads can be processed by MEC server n, or be further offloaded to neighboring MEC servers of MEC server n to be processed. This happens if MEC server n is heavily loaded, while its neighboring MEC server m ($m \in \mathcal{M}_n$) is lightly loaded. This helps to improve the overall system's execution performance.

The execution delay and energy consumption for the cooperative execution of MEC servers are due to the following three operations: (i) offloading workloads to the MEC server; (ii) load balancing among multiple cooperative MEC servers; and (iii) processing workloads at the MEC server(s). Without loss of generality, we assume that MEC servers have relatively stronger computation capability compared to MUs, the processing results are relatively small, which will be returned to the MU after execution. We ignore the output delay for processing results in this work. In the following, we present the energy consumption and execution delay of MEC servers cooperative execution in detail.

1) Offloading workloads to MEC server: In each time slot t, the MU first offloads workloads to, and then downloads results from the MEC server. We assume that the data transmission operates on orthogonal channels [22]. Since the energy consumption of each MU is mainly due to uplink data transmission to the MEC server, we focus on workload offloading in this paper. According to the bandwidth, wireless channel state, and the transmit power of each MU, the achievable uplink offloading transmission rate $r_{u,n}(t)$ (in bits/second) is given by Shannon capacity as

$$r_{u,n}(t) = \omega_{u,n}(t)\ \log_2\left(1 + C_{u,n}(t)P_{u,n}(t)/\sigma^2\right), \tag{4}$$

where $\omega_{u,n}(t)$ is the allocated bandwidth for MU u in time slot t, $C_{u,n}(t)$ is the wireless channel gain, $P_{u,n}(t)$ is the transmit power of MU u, and σ^2 is the noise power. Furthermore, $\omega_{u,n}(t)$ is constrained by the overall uplink bandwidth W_n of MEC server n, i.e.,

$$\sum_{u=1}^{U_n}\omega_{u,n}(t) \le W_n. \tag{5}$$

The offloading transmission delay for the task $I^{(k)}_{u,n}(t)$ from MU u to MEC server n can be written as

$$D^{(k)}_{tx,u,n}(t) = \alpha^{(k)}_{o,u,n}(t)\tau A^{(k)}_{u,n}(t)s_k/r_{u,n}(t), \tag{6}$$

and the overall offloading transmission delay in the coverage area of MEC server n is expressed by

$$D_{n,tx}(t) = \sum_{u\in\mathcal{U}_n}\sum_{k\in\mathcal{K}} D^{(k)}_{tx,u,n}(t). \tag{7}$$

The uplink offloading energy consumption of MU u for transmitting the task $I^{(k)}_{u,n}(t)$ to MEC server n is given by

$$E^{(k)}_{o,u,n}(t) = P_{u,n}(t)D^{(k)}_{tx,u,n}(t). \tag{8}$$

We obtain the overall offloaded workloads related to $I^{(k)}_{u,n}(t)$ at MEC server n, from MUs in its coverage area, as $A^{(k)}_n(t) \triangleq \sum_{u\in\mathcal{U}_n}\alpha^{(k)}_{o,u,n}(t)A^{(k)}_{u,n}(t)\tau$ in time slot t.

2) Load balancing among MEC servers: To meet the execution deadline $D^{\max}_k$ of each task, it's crucial to balance the workloads among MEC servers to make full advantage of their computation resources. We denote the load balancing decision of MEC server n for the task $I^{(k)}_{u,n}(t)$ in time slot t as $\boldsymbol{\beta}^{(k)}_{n\cdot}(t) = \{\beta^{(k)}_{nm}(t)\}_{m\in\mathcal{N}}$, where $\beta^{(k)}_{nm}(t)$ represents the transmitted workloads of the task $I^{(k)}_{u,n}(t)$ from MEC server n to MEC server m; $\beta^{(k)}_{nn}(t)$ means that the workloads of the task $I^{(k)}_{u,n}(t)$ are processed at the original MEC server n. We denote $\boldsymbol{\beta}_n(t) = \{\boldsymbol{\beta}^{(k)}_{n\cdot}(t)\}_{k\in\mathcal{K}}$ as the overall load balancing decisions of MEC server n for all types of tasks. And the total load balancing decisions in our cooperative-MEC system is $\boldsymbol{\beta}(t) = \{\boldsymbol{\beta}_n(t)\}_{n\in\mathcal{N}}$.

Furthermore, we denote the total workloads of the task $I^{(k)}_{u,n}(t)$ transmitted by MEC server n to its neighboring MEC servers as $\lambda^{(k)}_n(t) = \sum_{m\in\mathcal{M}_n}\beta^{(k)}_{nm}(t)$, and $\boldsymbol{\beta}^{(k)}_{\cdot n}(t) = \{\beta^{(k)}_{mn}(t)\}_{m\in\mathcal{M}_n}$ is the total workloads of the task $I^{(k)}_{u,n}(t)$ MEC server n receives from its neighboring MEC servers. Thus the total workloads of the task $I^{(k)}_{u,n}(t)$ to be processed at MEC server n will be $\nu^{(k)}_n(t) = \sum_{m\in\mathcal{N}}\beta^{(k)}_{mn}(t)$, which consists of two parts: the transmitted workloads $\sum_{m\in\mathcal{M}_n}\beta^{(k)}_{mn}(t)$ from its neighbors and the retained workloads $\beta^{(k)}_{nn}(t)$ of its own. These workloads are buffered in an absolute priority M/M/1 queue for processing by MEC server n, where priority is given to the MEC server n's own workloads. More details are provided in Section II-D3.

Note that the load balancing decisions $\boldsymbol{\beta}(t)$ should satisfy the following three constraints to make it feasible for each MEC server n:

1) $\beta^{(k)}_{nm}(t) \ge 0$: the transmitted workloads must be non-negative;
2) $\sum_{m\in\mathcal{N}}\beta^{(k)}_{nm}(t) = A^{(k)}_n(t)$: in order to guarantee that each workload is transferred only once among the MEC servers to avoid workload transmission loops, the total offloaded workloads of task $I^{(k)}_{u,n}(t)$ at MEC server n, i.e., $A^{(k)}_n(t)$, must be equal to $\sum_{m\in\mathcal{N}}\beta^{(k)}_{nm}(t)$;
3) According to the stable condition of absolute priority M/M/1 model [23] and the delay constraint $D^{\max}_k$ of each task, we obtain the stability constraint of our system as follows:

$$\begin{cases} \nu^{(k)}_n(t) = \mu^{(k)}_n - 1/D^{\max}_k, \\ \sum_{k\in\mathcal{K}}\mu^{(k)}_n \le \mu_n, \forall t, \forall n \in \mathcal{N}, \end{cases} \tag{9}$$

where $\mu^{(k)}_n$ is the service capacity (in workloads/second) allocated by MEC server n for the task $I^{(k)}_{u,n}(t)$, which follows a negative exponential distribution. The overall service capacity of MEC server n is denoted as μ_n.

Load balancing among multiple MEC servers is through a LAN with limited capacity, which incurs additional transmission delays due to congestion. Similar to [18], because of the negative exponentially distributed service capacity, the congestion delay for load balancing among MEC servers can be computed from an M/M/1 queuing model [23], given by

$$D_{n,g}(t) = \lambda_n(t) \cdot \frac{1}{1/\eta - \lambda(t)}, \ \ \lambda(t) < \frac{1}{\eta}, \tag{10}$$

where $1/\eta$ is the service capacity of the LAN, $\lambda_n(t) = \sum_{k\in\mathcal{K}} \lambda_n^{(k)}(t)$ is the total workloads for all types of tasks transmitted by MEC server n to its neighboring MEC servers, and $\lambda(t) = \sum_{n\in\mathcal{N}} \lambda_n(t)$ is the aggregate data traffic rate on the LAN required for the MEC servers to cooperatively process the offloaded workloads.

3) Processing workloads at an MEC server: When MEC server n process the workloads of the task $I_{u,n}^{(k)}(t)$, we assume that MEC server n must process its retained workloads $\beta_{nn}^{(k)}(t)$ first, and then the workloads $\sum_{m\in\mathcal{M}_n} \beta_{mn}^{(k)}(t)$ it receives from neighboring MEC servers. This fits well with the absolute priority M/M/1 queue model [23]. We denote the priority of $\beta_{nn}^{(k)}(t)$ as Level 1 and the priority of $\sum_{m\in\mathcal{M}_n} \beta_{mn}^{(k)}(t)$ as Level 2. Both levels of workloads follow the Poisson process with rates $\beta_{nn}^{(k)}(t)$ and $\sum_{m\in\mathcal{M}_n} \beta_{mn}^{(k)}(t)$, respectively. Note that the MEC server will always serve Level 1 workloads first; it will serve a Level 2 workload when there is no Level 1 workload in the queue. The service of a Level 2 workload will be interrupted by the arrival of a Level 1 workload: the Level 2 workload will be returned to the head-of-line of the queue so the Level 1 workload will be served (i.e., preemptive service). We obtain the average queueing delay by applying Little's law [24] as follows:

1) The average delay for Level 1 workloads of the task $I_{u,n}^{(k)}(t)$:

$$D_{p_1}(t) = \frac{1}{\mu_n^{(k)} - \beta_{nn}^{(k)}(t)}. \tag{11}$$

2) The average delay for Level 2 workloads of the task $I_{u,n}^{(k)}(t)$:

$$D_{p_2}(t) = \frac{1 + \beta_{nn}^{(k)}(t) \cdot D_{p1}(t)}{\mu_n^{(k)} - \nu_n^{(k)}(t)}. \tag{12}$$

The overall workloads related to $I_{u,n}^{(k)}(t)$ processing delay at MEC server n is

$$D_{n,p}^{(k)}(t) = \beta_{nn}^{(k)}(t) D_{p_1}(t) + \sum_{m\in\mathcal{M}_n} \beta_{mn}^{(k)}(t) D_{p_2}(t), \tag{13}$$

and the overall processing delay at MEC server n is $D_{n,p}(t) = \sum_{k\in\mathcal{K}} D_{n,p}^{(k)}(t)$.

With the offloading transmission delay $D_{n,tx}(t)$, the congestion delay $D_{n,g}(t)$ for load balancing among MEC servers, and the MEC server processing delay $D_{n,p}(t)$, we obtain the total execution delay for cooperative execution of MEC server n at time slot t, as

$$D_{n,o}(t) = D_{n,tx}(t) + D_{n,g}(t) + D_{n,p}(t). \tag{14}$$

Since parallel offloading is considered, the total execution delay of workloads in the coverage area of MEC server n at time slot t is denoted as the larger one of the local execution delay and cooperative execution delay [10], which is given by

$$D_n(t) = \max\{D_{n,l}(t), D_{n,o}(t)\}. \tag{15}$$

E. EH Model and Battery Dynamics

We assume that the MUs are powered by renewable resources, such as wind and solar radiation, etc. [25]. Each MU u has its energy harvesting equipment to capture renewable energy from the surrounding environment. To model energy harvesting process, we denote $H_{u,n}(t)$ as the renewable energy packets with quantity $0 \le H_{u,n}(t) \le H_{u,n}^{\max}$ arrive at MU u, where $H_{u,n}(t)$ are independent and identically distributed (i.i.d.) and upper bounded by $H_{u,n}^{\max}$. We denote $G_{u,n}(t)$ as the energy harvesting decision, which is the part of $H_{u,n}(t)$ that will be harvested by MU u in each time slot t, satisfying

$$0 \le G_{u,n}(t) \le H_{u,n}(t). \tag{16}$$

The harvested energy $G_{u,n}(t)$ will be stored in the battery of MU u and used to power its operations in the next time slot. Let $\boldsymbol{G}_n = \{G_{u,n}(t)\}_{u\in\mathcal{U}_n}$ denote the set of energy harvesting decisions for all the MUs in the coverage area of MEC server n. The battery energy level of MU u at the beginning of time slot t, denoted by $B_{u,n}(t)$, satisfies the following constraint.

$$E_{u,n}^{\min} \le B_{u,n}(t) \le E_{u,n}^{\max}, \tag{17}$$

where $E_{u,n}^{\min}$ and $E_{u,n}^{\max}$ are the minimum and maximum energy levels of the battery, respectively, and $0 \le E_{u,n}^{\min} \le E_{u,n}^{\max}$. The overall energy consumption for the task $I_{u,n}^{(k)}(t)$ of MU u, $E_{u,n}^{(k)}(t)$, contains three parts: the local execution energy consumption $E_{l,u,n}^{(k)}(t)$, the uplink offloading energy consumption $E_{o,u,n}^{(k)}(t)$, and the energy consumption of the task $I_{u,n}^{(k)}(t)$ transmitted to centralized remote cloud $E_{r,u,n}^{(k)}(t)$, which is given by

$$E_{u,n}^{(k)}(t) = E_{l,u,n}^{(k)}(t) + E_{o,u,n}^{(k)}(t) + E_{r,u,n}^{(k)}(t), \tag{18}$$

where $E_{r,u,n}^{(k)}(t) = \alpha_{r,u,n}^{(k)}(t)\tau A_{u,n}^{(k)}(t) s_k \cdot e_k$, and e_k is the energy consumption for transmitting each data traffic of the task $I_{u,n}^{(k)}(t)$ to centralized remote cloud. And the overall energy consumption of MU u is $E_{u,n}(t) = \sum_{k\in\mathcal{K}} E_{u,n}^{(k)}(t)$, the following constraint should be satisfied:

$$0 \le E_{u,n}(t) \le B_{u,n}(t), \tag{19}$$

which means that the energy consumption of MU u in time slot t cannot exceed its battery energy level. The overall energy consumption of all MUs in the coverage area of MEC server n is $E_n(t) = \sum_{u\in\mathcal{U}_n} E_{u,n}(t)$. The evolution of the battery of MU u is governed by

$$\begin{aligned} B_{u,n}(t+1) = \max\{&\min\{B_{u,n}(t) - E_{u,n}(t) + G_{u,n}(t), \\ &E_{u,n}^{\max}\}, E_{u,n}^{\min}\}. \end{aligned} \tag{20}$$

III. Problem Formulation

In this section, we formulate a minimization problem with an objective function of the long-term system average-cost (weighted sum of energy consumption and execution delay) under battery level stability and delay constraints. Denote the system control vectors set in the coverage area of MEC server n at time slot t as $\boldsymbol{\Psi}(t) \triangleq [\boldsymbol{\alpha}_n^{(k)}(t), \boldsymbol{\omega}_n(t), \boldsymbol{G}_n(t), \boldsymbol{\beta}_n(t)]$. The goal is to find the optimal parallel offloading decisions $\boldsymbol{\alpha}_n^{(k)}(t) = \{\boldsymbol{\alpha}_{1,n}^{(k)}(t), \boldsymbol{\alpha}_{2,n}^{(k)}(t), ..., \boldsymbol{\alpha}_{U_n,n}^{(k)}(t)\}$, bandwidth allocations $\boldsymbol{\omega}_n(t) = \{\omega_{1,n}(t), \omega_{2,n}(t), ..., \omega_{U_n,n}(t)\}$, harvested renewable energy $\boldsymbol{G}_n(t) = \{G_{1,n}(t), G_{2,n}(t), ..., G_{U_n,n}(t)\}$, and load balancing decisions $\boldsymbol{\beta}_n(t) = \{\boldsymbol{\beta}_{n\cdot}^{(1)}(t), \boldsymbol{\beta}_{n\cdot}^{(2)}(t), ..., \boldsymbol{\beta}_{n\cdot}^{(K)}(t)\}$. Formally, the problem is formulated as

$$\begin{aligned}
\boldsymbol{\mathcal{P}_1}: &\min_{\boldsymbol{\Psi}(t)} \lim_{T\to\infty} \frac{1}{T} \sum_{t=0}^{T-1} \sum_{n=1}^{N} \mathbb{E}\{E_n(t) + \zeta D_n(t)\} \\
&\text{s.t. (5), (9), (10), (16), (17), (19), (20)} \\
&\alpha_{l,u,n}^{(k)}(t), \alpha_{o,u,n}^{(k)}(t), \alpha_{r,u,n}^{(k)}(t) \in [0,1], \\
&\qquad\qquad u \in \mathcal{U}_n, n \in \mathcal{N}, t \in \mathcal{T} \quad (21) \\
&0 \le H_{u,n}(t) \le H_{u,n}^{\max}, u \in \mathcal{U}_n, n \in \mathcal{N}, t \in \mathcal{T}, \quad (22)
\end{aligned}$$

where ζ is a weight to tradeoff energy consumption and the execution delay, constraint (21) is on the parallel offloading decisions, and constraint (22) ensures that the arrived renewable energy at each MU is bounded.

Problem $\mathcal{P}_1$ is a stochastic programming due to the random workload arrivals and energy harvesting, in which the optimization of parallel offloading decisions relates to time-coupled battery dynamics (i.e., the evolution of battery energy level depends on the previous time slot). The parallel offloading decisions and load balancing decisions are related to system information and coupled over the updated time slot. It's difficult to deal with load balancing, resources allocation, and battery level stability in a long-term optimal solution.

IV. Online Centralized Algorithm

In this section, we aim to design an online centralized energy consumption and execution delay management algorithm to achieve load balancing among MEC servers. We first develop a real-time algorithm for load balancing to allow cooperative execution among multiple MEC servers under the delay constraints given by individual MUs. Then, we take the battery stability of MUs into account, and propose the LYP-CCMA algorithm to obtain the optimal solution of the original Problem $\mathcal{P}_1$. The detailed procedure of LYP-CCMA is presented in Algorithm 1.

A. The Real-Time Algorithm

In this subsection, we aim to design a centralized algorithm for load balancing among multiple cooperative MEC servers for given system parameters. The parameters include the input data sizes, the service capacities of local CPUs and MEC servers, and the total uplink bandwidth shared by the MEC servers and their served MUs. With the prior knowledge of system parameters, we use the YALMIP optimization tool in MATLAB to obtain the primary parallel offloading data sizes $\boldsymbol{\alpha}_n^{(k)}(t)' \triangleq [A_{Local,n}^{(k)}(t)^*, A_{MEC,n}^{(k)}(t)^*, A_{Cloud,n}^{(k)}(t)^*]$ and optimal load balancing decisions $\boldsymbol{\beta}_n(t)$. Note that $A_{Local,n}^{(k)}(t)^*$, $A_{MEC,n}^{(k)}(t)^*$, and $A_{Cloud,n}^{(k)}(t)^*$ are the total data sizes for task $I_{u,n}^{(k)}(t)$ executed at local CPU, MEC server, and centralized remote cloud within the coverage area of MEC server n in time slot t, respectively.

However, such an approach can only obtain the primary parallel offloading data sizes and optimal load balancing decisions in *one time slot* t, while the goal is to solve the minimization problem of the the long-term system average-cost under battery stability and delay constraints. In order to solve the original Problem $\mathcal{P}_1$, we first transform the long-term time-averaged constraints into battery queue stability constraints [26]. Therefore, the time-dependent battery-related constraints (20) can be transformed as

$$\lim_{T\to\infty} \frac{1}{T} \sum_{t=0}^{T-1} \mathbb{E}\{B_{u,n}(t)\} = 0, \ u \in \mathcal{U}_n, n \in \mathcal{N}. \quad (23)$$

Then we obtain Problem $\mathcal{P}_2$, the relaxed version of $\mathcal{P}_1$:

$$\begin{aligned}
\boldsymbol{\mathcal{P}_2}: &\min_{\boldsymbol{\Psi}(t)} \lim_{T\to\infty} \frac{1}{T} \sum_{t=0}^{T-1} \sum_{n=1}^{N} \mathbb{E}\{E_n(t) + \zeta D_n(t)\} \\
&\text{s.t. (5), (9), (10), (16), (17), (19), (21)} - \text{(23)}.
\end{aligned}$$

Since Problem $\mathcal{P}_2$ is a relaxed version of Problem $\mathcal{P}_1$, the solution to $\mathcal{P}_2$ maybe not be feasible to $\mathcal{P}_1$. The reason we study $\mathcal{P}_2$ is to accelerate the design of the real-time algorithm for solving $\mathcal{P}_1$. Then we focus on solving $\mathcal{P}_2$ based on Lyapunov optimization instead of solving the original $\mathcal{P}_1$. Moreover, we will show the algorithm design in next section, and will show the performance analysis that the solution of $\mathcal{P}_2$ is the asymptotic optimal solution of $\mathcal{P}_1$ in Section V-C.

B. Lyapunov-based Centralized Energy consumption and Execution Delay Management Algorithm

In this subsection, to solve Problem $\mathcal{P}_2$ with the stability of battery levels constraints while minimizing the long-term system average-cost, we design an online Lyapunov-based centralized energy consumption and execution delay management algorithm. Utilizing the Lyapunov Optimization, the time-coupled battery dynamics can be decoupled and the decisions can be obtained online without the prior system information. In order to utilize Lyapunov Optimization, we first define the two important parameters, virtual energy queue and the perturbation parameter as follows:

Definition 1: Denote $\widetilde{B}_{u,n}(t) = B_{u,n}(t) - \theta_{u,n}$ as a virtual battery queue, where $\theta_{u,n}$ is the perturbation parameter to protect the battery.

Definition 2: The constraints of the perturbation parameter $\theta_{u,n}$ are as follows:

$$\theta_{u,n} > \widetilde{E}_{u,n}^{\max} + V \cdot [\phi + D_k^{\max}](E_{u,n}^{\min})^{-1}, \quad (24)$$

where $V > 0$ is a control parameter to balance the stability of battery and the system cost, and $\widetilde{E}_{u,n}^{\max} = \min\{\max\{E_{l,u,n}^{(k)}(t)^*, E_{o,u,n}^{(k)}(t)^*, E_{r,u,n}^{(k)}(t)^*\}, E_{u,n}^{\max}\}$. Note

Algorithm 1: The Lyapunov optimization-based centralized cost management algorithm (LYP-CCMA)

At the beginning of time slot t, initialize the system's parameters $I_{u,n}^{(k)}(t)$, $\mu_{u,n}^{(k)}$, μ_n and W_n ;
for $n = 1 : N$ **do**
  Calculate the sum of execution delay $D_n(t)$ and energy consumption $E_n(t)$ of Local Execution, MEC servers Execution and centralized remote cloud execution for the workloads in the coverage area of MEC server n, respectively;
end
Determine $A_{Local,n}^{(k)}(t)^*$, $A_{MEC,n}^{(k)}(t)^*$, $A_{Cloud,n}^{(k)}(t)^*$ and $\boldsymbol{\beta}_n(t)$ by solving

$$\min_{\boldsymbol{\alpha}_n^{(k)}(t)', \boldsymbol{\beta}_n(t)} \sum_{n=1}^{N} \{E_n(t) + \zeta D_n(t)\}$$
$$\text{s.t.} \quad \sum_{n\in\mathcal{N}} \sum_{m\in\mathcal{M}_n} \beta_{nm}^{(k)}(t) \leq \frac{1}{\eta};$$

Minimize the long-term system average-cost while stabling the battery levels by solving Problem $\boldsymbol{\mathcal{P}_3}$;

that substituting the optimal values of parallel offloading data sizes into (2) and (8), we obtain the energy consumption for local execution $E_{l,u,n}^{(k)}(t)^*$ and the energy consumption for MEC servers cooperative execution $E_{o,u,n}^{(k)}(t)^*$.

Based on the above two definitions, we define the *Lyapunov function* as follows:

$$L[\widetilde{B}_{u,n}(t)] \triangleq \frac{1}{2} \sum_{n=1}^{N} \sum_{u=1}^{U_n} \widetilde{B}_{u,n}^2(t), \tag{25}$$

and the one-slot conditional *Lyapunov drift* is given by

$$\Delta[\widetilde{B}_{u,n}(t)] \triangleq \mathbb{E}\big\{L[\widetilde{B}_{u,n}(t+1)] - L[\widetilde{B}_{u,n}(t)] \big| \widetilde{B}_{u,n}(t)\big\}. \tag{26}$$

We consider the *drift-plus-penalty function* as

$$\Delta[\widetilde{B}_{u,n}(t)] + V \sum_{n=1}^{N} \mathbb{E}\{E_n(t) + \zeta D_n(t)\}, \tag{27}$$

where the control parameter V (great than zero) is introduced to tradeoff between $\Delta[\widetilde{B}_{u,n}(t)]$ and the system cost $(\sum_{n=1}^{N} \mathbb{E}\{E_n(t) + \zeta D_n(t)\})$.

Lemma 1: Define a constant $\Phi = \frac{1}{2}[(H_{u,n}^{\max})^2 + (\widetilde{E}_{u,n}^{\max})^2]$, we have the following upper bound on $\Delta[\widetilde{B}_{u,n}(t)]$:

$$\Delta[\widetilde{B}_{u,n}(t)] \leq \mathbb{E}\Big\{ \sum_{n=1}^{N} \sum_{u=1}^{U} [\widetilde{B}_{u,n}(t)(G_{u,n}(t) - E_{u,n}(t)) + \Phi] \Big\}. \tag{28}$$

Proof: Please refer to Appendix A. ∎

We next derive an upper bound for the *drift-plus-penalty function*, which is given by

$$\begin{aligned}
&\Delta[\widetilde{B}_{u,n}(t)] + V \sum_{n=1}^{N} \mathbb{E}\{E_n(t) + \zeta D_n(t)\} \\
&\leq \mathbb{E}\Big\{ \sum_{n=1}^{N} \sum_{u=1}^{U_n} [\widetilde{B}_{u,n}(t)(G_{u,n}(t) - E_{u,n}(t)) + \Phi] \Big\} \\
&\quad + V \sum_{n=1}^{N} \mathbb{E}\{E_n(t) + \zeta D_n(t)\}.
\end{aligned} \tag{29}$$

Inferring from the above problem (29), we can minimize the long-term system average-cost by minimizing the upper bound of the drift-plus-penalty function and simultaneously ensure the stability of battery levels constraints. Therefore, we transform $\mathcal{P}_2$ to $\mathcal{P}_3$ as follows:

$$\begin{aligned}
\boldsymbol{\mathcal{P}_3}: &\min_{\boldsymbol{\Psi}(t)} \Big\{ \sum_{n=1}^{N} \sum_{u=1}^{U_n} [\widetilde{B}_{u,n}(t)(G_{u,n}(t) - E_{u,n}(t))] \\
&\quad + V \sum_{n=1}^{N} \{E_n(t) + \zeta D_n(t)\} \Big\} \\
&= \min \Big\{ \sum_{n=1}^{N} \sum_{u=1}^{U_n} \Big\{ [\widetilde{B}_{u,n}(t)(G_{u,n}(t) - E_{u,n}(t)) + V \cdot \phi] \Big\} \\
&\quad + V \sum_{n=1}^{N} \sum_{u=1}^{U_n} [E_{l,u,n}^{(k)}(t) + \zeta D_{l,u,n}^{(k)}(t)] \\
&\quad + V \sum_{n=1}^{N} \left[\left(\sum_{u=1}^{U_n} E_{o,u,n}^{(k)}(t) \right) + \zeta D_{o,u,n}^{(k)}(t) \right] \Big\} \\
&\text{s.t. (5), (9), (10), (16) – (17), (19), (21) – (22).}
\end{aligned} \tag{30}$$

Note that in the first term on the right-hand-side of (30), we include a term $V \cdot \phi$, where $\phi = \sum_{k=1}^{K} [E_{r,u,n}^{(k)}(t) + D_{r,u,n}^{(k)}(t)]$ is the system cost of MU u for transmitting workloads to centralized remote cloud, $D_{r,u,n}^{(k)}(t) = \alpha_{r,u,n}^{(k)}(t) \tau A_{u,n}^{(k)}(t) s_k \cdot d_k$ is the transmission delay of the task $I_{u,n}^{(k)}(t)$ to centralized remote cloud, d_k is the unit transmission delay.

According to our proposed LYP-CCMA algorithm, we have acquired the primary parallel offloading data sizes, the optimal load balancing decisions, and the total system cost within the coverage area of each MEC server in time slot t. However, such a solution does not ensure the long-term system performance. Moreover, the computational complexity of the centralized algorithm will increase with the number of MEC servers and the number of MUs they serve. It would be challenging to obtain the parallel offloading decisions and energy consumptions for each MU at each time slot, which affects the evolution of the MUs' batteries. These challenges call for a distributed algorithm to solve the proposed problem. The proposed distributed algorithm, termed ADMM-DARA, will be introduced in the next section.

V. ADMM-Based Distributed Algorithm for Resources Allocation

In this section, we aim to design a distributed algorithm to solve Problem $\mathcal{P}_3$. In order to obtain the parallel offloading

and resources allocation decisions for each MU, we split $\mathcal{P}_3$ into equivalent three subproblems: $\mathcal{P}_3^{1,2}$ and $\mathcal{P}_3^3$. Based on the alternating direction method of multipliers, we propose two algorithms for the optimization of parallel offloading decisions ($\mathcal{P}_3^1$) and bandwidth allocation ($\mathcal{P}_3^2$), respectively. And then we address the optimization of energy harvesting in $\mathcal{P}_3^3$. The corresponding procedure is summarized in Algorithm 2. In addition, we show the solution process of ADMM-DARA and performance analysis of the proposed algorithms in Section V-B and Section V-C, respectively.

A. Distributed Algorithm Design

To facilitate the development of the algorithm, as well as to obtain the parallel offloading decisions and the allocated bandwidth for every MU, we first denote $\mathbf{\Psi}'(t) \triangleq [\boldsymbol{\alpha}_n^{(k)}(t), \boldsymbol{\omega}_n(t), \boldsymbol{\beta}_n(t)]$ as a control vector set for $\mathcal{P}_3^{1,2}$ in time slot t, and $\mathcal{P}_3^{1,2}$ is written as follows:

$$\begin{aligned}\boldsymbol{\mathcal{P}_3^{1,2}}: \min_{\mathbf{\Psi}'(t)} \Bigg\{ & \sum_{n=1}^{N}\sum_{u=1}^{U_n} \Big\{ -\widetilde{B}_{u,n}(t)E_{u,n}(t) + V\cdot\phi \Big\} \\ & + V\sum_{n=1}^{N}\sum_{u=1}^{U_n} \Big\{ E_{l,u,n}^{(k)}(t) + \zeta D_{l,u,n}^{(k)}(t) \Big\} \\ & + V\sum_{n=1}^{N} \Big\{ [\sum_{u=1}^{U_n} E_{o,u,n}^{(k)}(t)] + \zeta D_{o,u,n}^{(k)}(t) \Big\} \Bigg\} \\ & \text{s.t. (5), (9), (10), (19), (21),} \end{aligned} \tag{31}$$

where $\mathcal{P}_3^{1,2}$ contains two subproblems $\mathcal{P}_3^1$ and $\mathcal{P}_3^2$. And the subproblem $\mathcal{P}_3^3$ will be considered in Section V-A3.

1) Optimization of parallel offloading decisions: From (31), we obtain the optimization problem for local execution, which is written as follows.

$$\begin{aligned}\boldsymbol{\mathcal{P}_3^1}: \min_{\boldsymbol{\alpha}_n^{(k)}(t)} & \sum_{n=1}^{N}\sum_{u=1}^{U_n} \Big\{ V[E_{l,u,n}^{(k)}(t) + \zeta D_{l,u,n}^{(k)}(t)] \\ & - \widetilde{B}_{u,n}(t)E_{l,u,n}^{(k)}(t) \Big\}. \end{aligned} \tag{32}$$

In $\mathcal{P}_3^1$, the new optimization vector is $\mathbf{x} \triangleq [x_{1,1}^{(k)}, ..., x_{U_n,1}^{(k)}, ..., x_{1,N}^{(k)}, ..., x_{U_n,N}^{(k)}]$, which relates to the overall $U_n \cdot N$ variables as $\vec{x}_{u,n}^{(k)} = \boldsymbol{\alpha}_{u,n}^{(k)}(t)$, for $1 \le u \le U_n$ and $1 \le n \le N$. We rewrite $\mathcal{P}_3^1$ as the sum of certain functions of each $\vec{x}_{u,n}^{(k)}$, denoted by $F_n(\vec{x}_{u,n}^{(k)})$, as follows:

$$\begin{aligned}\boldsymbol{\mathcal{P}_3^{1'}}: \min_{\mathbf{x}} & \sum_{n=1}^{N}\sum_{u=1}^{U_n} \Big[F_n(\vec{x}_{u,n}^{(k)}) \Big] \\ & \text{s.t. } \vec{x}_{u,n}^{(k)} \in \mathcal{X},\ \forall u, n, \end{aligned} \tag{33}$$

where

$$\begin{aligned} F_n(\vec{x}_{u,n}^{(k)}) = \Big\{ & [V - \widetilde{B}_{u,n}(t)]\tau s_k h_k \xi [f_{u,n}^{(k)}(t)]^2 \\ & + \frac{V\zeta\tau s_k}{\mu_{u,n}^{(k)}} \Big\} A_{u,n}^{(k)}(t)\vec{x}_{u,n}^{(k)}. \end{aligned}$$

Note that in each time slot t, optimizing the parallel offloading decisions $\boldsymbol{\alpha}_{u,n}^{(k)}(t) \subset \boldsymbol{\alpha}_n^{(k)}(t)$ for MU u is actually to solve $\boldsymbol{A}_{u,n}^{(k)}(t)$. Denote $\boldsymbol{A}_{u,n}^{(k)}(t) = A_{u,n}^{(k)}(t)\boldsymbol{\alpha}_{u,n}^{(k)}(t) = \{A_{l,u,n}^{(k)}(t), A_{o,u,n}^{(k)}(t), A_{r,u,n}^{(k)}(t)\}$, indicating the data sizes for local execution, MEC servers cooperative execution, and centralized remote cloud execution for the task $I_{u,n}^{(k)}(t)$, respectively. Thus, the constraints of $\mathcal{P}_3^{1'}$ are given by $\mathcal{X} \triangleq \{[0, A_{u,n}^{(k),\max}], \sum_{n=1}^{N}\sum_{u=1}^{U_n} A_{l,u,n}^{(k)}(t) = A_{Local,n}^{(k)}(t)^*$, where $A_{l,u,n}^{(k)}(t) = A_{u,n}^{(k)}(t) \cdot \vec{x}_{u,n}^{(k)}(t)$. We then introduce an auxiliary vector $\mathbf{z}$ as a copy of $\mathbf{x}$, and obtain the new optimization Problem $\mathcal{P}_3^{1''}$ which is equivalent to Problem $\mathcal{P}_3^{1'}$.

$$\begin{aligned}\boldsymbol{\mathcal{P}_3^{1''}}: & \min_{\mathbf{x},\mathbf{z}} f(\mathbf{x}) + g(\mathbf{z}) \\ & \text{s.t. } \mathbf{x} - \mathbf{z} = 0, \end{aligned} \tag{34}$$

where

$$f(\mathbf{x}) = \sum_{n=1}^{N}\sum_{u=1}^{U_n} [F_n(\vec{x}_{u,n}^{(k)}) + \mathbf{1}(\vec{x}_{u,n}^{(k)} \in \mathcal{X})]$$

$$g(\mathbf{z}) = \mathbf{1}\Big(\sum_{u=1}^{U_n} \vec{z}_{u,n}^{(k)} = A_{Local,n}^{(k)}(t)^* \Big).$$

In Problem $\mathcal{P}_3^{1''}$, $\mathbf{1}(\cdot)$ is an indicator function: it is 0 if the condition in the parentheses is true; otherwise it is infinite. Following the general ADMM approach [27], we introduce dual variables $\mathbf{y} \triangleq [y_{1,1}^{(k)}, ..., y_{U_n,1}^{(k)}, ..., y_{1,N}^{(k)}, ..., y_{U_n,N}^{(k)}]$, and iteratively update the variables as follows.

$$\begin{aligned} x_{u,n}^{(k),j+1} = \arg\min_{x_{u,n}^{(k)}} \Big\{ & F_n(\vec{x}_{u,n}^{(k)}) + \frac{\rho}{2} \Big\| \vec{x}_{u,n}^{(k)} - z_{u,n}^{(k),j} + \frac{y_{u,n}^{(k),j}}{\rho} \Big\|_2^2 \\ & \Big| \vec{x}_{u,n}^{(k)} \in \mathcal{X} \Big\},\ \forall u, n \end{aligned} \tag{35}$$

$$\begin{aligned} \mathbf{z}_{u,n}^{(k),j+1} = \arg\min_{\mathbf{z}} \Big\{ & g(\mathbf{z}) + \frac{\rho}{2} \Big\| \vec{z}_{u,n}^{(k)} - \frac{y_{u,n}^{(k),j}}{\rho} - x_{u,n}^{(k),j+1} \Big\|_2^2 \\ & \Big| \sum_{u=1}^{U} \vec{z}_{u,n}^{(k)} = A_{Local,n}^{(k)}(t)^* \Big\}, \end{aligned} \tag{36}$$

$$y_{u,n}^{(k),j+1} = y_{u,n}^{(k),j} + \rho(x_{u,n}^{(k),j+1} - z_{u,n}^{(k),j+1}),\ \forall u, n, \tag{37}$$

where $x_{u,n}^{(k),j+1}$, $z_{u,n}^{(k),j+1}$, and $y_{u,n}^{(k),j+1}$ are the respective variable values at the $(j+1)$th iteration, $\rho > 0$ is a penalty parameter to obtain a good convergence performance with an appropriate value.

According to [27], we denote $r^{j+1} = x_{u,n}^{(k),j+1} - z_{u,n}^{(k),j+1}$ and $s^{j+1} = -\rho(z_{u,n}^{(k),j+1} - x_{u,n}^{(k),j})$ as the primal residual and the dual residual at iteration $j+1$, respectively. And we prescribe the stopping criterion of iterations as follows:

$$\big\| r^j \big\|_2 \le \epsilon^{pri} \quad \text{and} \quad \big\| s^j \big\|_2 \le \epsilon^{dual}, \tag{38}$$

where ϵ^{pri} and ϵ^{dual} are the feasibility tolerances for the primal and dual feasibility conditions, respectively. After solving this subproblem, we obtain the optimal parallel offloading decisions of the MEC system at time slot t, which is written as $\boldsymbol{A}_{u,n}^{(k)}(t)^* = \{\boldsymbol{A}_{1,1}^{(k)}(t)^*, ..., \boldsymbol{A}_{U_n,1}^{(k)}(t)^*, ..., \boldsymbol{A}_{1,N}^{(k)}(t)^*, ..., \boldsymbol{A}_{U_n,N}^{(k)}(t)^*\}$.

Remark 1: According to [28], we show that the proposed ADMM-based distributed algorithm achieves a convergence rate of $\mathcal{O}(1/j)$ with j being the number of iterations. In

contrast to the convergence rate $\mathcal{O}(1/\sqrt{j})$ of subgradient-based algorithm, our proposed ADMM-DARA is faster and is suitable for real time implementation.

2) Optimization of bandwidth allocation: According to (31), we obtain the optimization problem for bandwidth allocation, which is written as

$$\boldsymbol{\mathcal{P}_3^2}: \min_{\boldsymbol{\omega}_n(t)} \sum_{n=1}^{N} \Big\{ V\{[\sum_{u=1}^{U_n} E_{o,u,n}^{(k)}(t)] + \zeta[D_{n,tx}(t) + D_{n,g}(t) + D_{n,p}(t)]\} - \sum_{u=1}^{U_n} \widetilde{B}_{u,n}(t) E_{o,u,n}^{(k)}(t) \Big\}. \tag{39}$$

In Problem $\mathcal{P}_3^2$, define the new optimization vector as $\mathbf{x}' \triangleq [x_{1,1}, ..., x_{U_n,1}, ..., x_{1,N}, ..., x_{U_n,N}]$, which relates to $U_n \cdot N$ variables, and $\vec{x}_{u,n} = \omega_{u,n}(t)$, for $u \in [1, U_n]$ and $n \in [1, N]$. Then we derive the following Problem $\mathcal{P}_3^{2'}$.

$$\boldsymbol{\mathcal{P}_3^{2'}}: \min_{\mathbf{x}'} \sum_{n=1}^{N} \sum_{u=1}^{U_n} \Big[F_n(\vec{x}_{u,n}) \Big] \quad \text{s.t. } \vec{x}_{u,n} \in \mathcal{X}', \ \forall k, n. \tag{40}$$

Note that in (39), the optimization of bandwidth $\omega_{u,n}(t)$ is affected by the items that contain $E_{o,u,n}^{(k)}(t)$ and $D_{n,tx}(t)$. Thus, to remove irrelevant items and convert $\omega_{u,n}(t)$ to numerator, we take the negative reciprocal and define $F_n(\vec{x}_{u,n})$ as

$$F_n(\vec{x}_{u,n}) = \frac{\Big[-\log(1 + \frac{C_{u,n}(t) P_{u,n}(t)}{\sigma^2}) \Big] \vec{x}_{u,n}}{\boldsymbol{A}_{u,n}^{(k)}(t)_k^s \Big\{ P_{u,n}(t)[V - \widetilde{B}_{u,n}(t)] + V\zeta \Big\}}, \tag{41}$$

where $\boldsymbol{A}_{u,n}^{(k)}(t)^*$ are the optimal parallel offloading decisions obtained in Section V-A1. The corresponding constraints are $\mathcal{X}' \triangleq \{[0, \omega_{u,n}^{\max}], \sum_{n=1}^{N} \sum_{u=1}^{U_n} \omega_{u,n}(t) = W_n\}$.

Similar to the optimization of parallel offloading decisions, we introduce an auxiliary vector $\mathbf{z}'$ as a copy of $\mathbf{x}'$, and derive a new optimization problem $\mathcal{P}_3^{2''}$ as follows.

$$\boldsymbol{\mathcal{P}_3^{2''}}: \min_{\mathbf{x}', \mathbf{z}'} \ f(\mathbf{x}') + g(\mathbf{z}') \quad \text{s.t. } \mathbf{x}' - \mathbf{z}' = 0, \tag{42}$$

where

$$f(\mathbf{x}') = \sum_{n=1}^{N} \sum_{u=1}^{U_n} \Big[F_n(\vec{x}_{u,n}) + \mathbf{1}(\vec{x}_{u,n} \in \mathcal{X}') \Big]$$

$$g(\mathbf{z}') = \mathbf{1}\Big(\sum_{u=1}^{U_n} \vec{z}_{u,n} = W_n \Big).$$

The variable values are also iteratively updated, while $x_{u,n}^{j+1}$ and $y_{u,n}^{j+1}$ are updated similar to $x_{u,n}^{(k),j+1}$ and $y_{u,n}^{(k),j+1}$ as in (35) and (37), respectively. The update of $z_{u,n}^{j+1}$ is given by

$$\mathbf{z}_{u,n}^{j+1} = \arg\min_{\mathbf{z}'} \Big\{ g(\mathbf{z}') + \frac{\rho'}{2} \Big\| \vec{z}_{u,n} - \frac{y_{u,n}^{j}}{\rho'} - x_{u,n}^{j+1} \Big\|_2^2 \Big| \sum_{u=1}^{U_n} \vec{z}_{u,n} = W_n \Big\}, \tag{43}$$

where $\rho' > 0$ is a penalty parameter. Note that the stopping criterion of the ADMM-based algorithms can be derived similarly, which are omitted for brevity.

Algorithm 2: ADMM-based Distributed Algorithm for Resources Allocation (ADMM-DARA)

for $t = 1 : T$ **do**
- All MUs send their task requests to a dedicated MEC server;
- The MEC system collects all primary offloading decisions $A_{Local,n}^{(k)}(t)^*$, $A_{MEC,n}^{(k)}(t)^*$, and $A_{Cloud,n}^{(k)}(t)^*$ using Algorithm 1 to solve Problem $\mathcal{P}_3^{1,2}$;
- **repeat**
  - **for** $n = 1 : N$ **do**
    - **for** $u = 1 : U_n$ **do**
      - Each MU u individually solves (35) and offloads system states and $x_{u,n}^{(k),j+1}$ to MEC n;
    - **end**
    - MEC server n collects all MU's conditions to obtain $z_{u,n}^{(k),j+1}$ by solving (36);
    - **for** $u = 1 : U_n$ **do**
      - Each MU u reports $y_{u,n}^{(k),j+1}$ to MEC server n;
    - **end**
  - **end**
- **until** (38) *is TRUE*;
- Obtain the solution of subproblem $\mathcal{P}_3^1$ and the optimal offloading decisions $\boldsymbol{A}_{u,n}^{(k)}(t)^*$;
- Solve subproblem $\mathcal{P}_3^2$ in the similar process of the solution of subproblem $\mathcal{P}_3^1$ and obtain the optimal bandwidth allocation $w_{u,n}^*(t)$;
- Obtain the energy consumption $E_{u,n}(t)$ of MU u;
- **for** $u = 1 : U_n$ **do**
  - **if** $\widetilde{B}_{u,n}(t) \leq 0$ **then**
    - $G_{u,n}^*(t) = H_{u,n}^{\max}$;
  - **else**
    - $G_{u,n}^*(t) \to 0$;
  - **end**
- **end**
- Set $t = t + 1$;

end

3) Optimization of EH: According to the above solution procedure in Section V-A, we first obtain the parallel offloading decisions, and then the allocation of bandwidth and the energy consumption for each MU at each time slot t. Finally, we obtain the evolution of each MU's battery. In addition, we'll derive the optimal EH solution depends on the subproblem $\mathcal{P}_3^3$

$$\boldsymbol{\mathcal{P}_3^3}: \min_{0 \leq G_{u,n}(t) \leq H_{u,n}^{\max}} \sum_{n=1}^{N} \sum_{u=1}^{U_n} \widetilde{B}_{u,n}(t) G_{u,n}(t). \tag{44}$$

To acquire the smallest $\mathcal{P}_3^3$, the optimal solution of EH has two forms. When $\widetilde{B}_{u,n}(t) \leq 0$, we have $G_{u,n}^*(t) = H_{u,n}^{\max} \cdot \mathbf{1}\{\widetilde{B}_{u,n}(t) \leq 0\}$. Furthermore, when $\widetilde{B}_{u,n}(t) > 0$, we make $G_{u,n}(t)$ approach to zero to obtain the optimal solution of Problem $\mathcal{P}_3^3$.

TABLE II: Comparison of Existing Work

Scenarios	Scheme	Parallel offloading	Load balancing	Distributed algorithm
ADMM-DRAR	ADMM-DRAR	✓	✓	✓
ADMM-NLB	ADMM-NLB, [29]	✓		✓
LYP-NLB	[30]	✓		
LYP-CCMA	[8], [9], [18]	✓	✓	

The mark ✓ indicates that the scheme considers the corresponding metric.

B. The Solution Process of ADMM-DARA

In this subsection, we describe the solution process of ADMM-based distributed algorithm for resources allocation (ADMM-DARA). In each time slot, the SDN controllers collect all system information and obtain the primary offloading decisions by executing the Algorithm 1. In the initial iteration $j = 0$ of the process of ADMM, each MU u individually solves (35) and sends $z_{u,n}^{(k),j}$ and the updated $x_{u,n}^{(k),j+1}$, to MEC server n. In addition, MEC server n obtains $z_{u,n}^{(k),j+1}$ by executing (36) and broadcasts the information to each MU. Furthermore, each MU obtains $y_{u,n}^{(k),j+1}$ by updating (37), as well as the primal and dual residuals. Until the stopping criterion (38) is satisfied, we obtain the optimal solution to subproblem $\mathcal{P}_3^1$ and the offloading decisions $\boldsymbol{A}_{u,n}^{(k)}(t)^*$. Moreover, similar to the solution process of $\mathcal{P}_3^1$, we obtain the optimal solution to subproblem $\mathcal{P}_3^2$ and the bandwidth allocation $w_{u,n}^*(t)$. Finally, we obtain the energy consumption of each MU and update the state of the batteries.

With the execution of ADMM, r^{j+1} and s^{j+1} will eventually converge to 0. And when j goes to infinity, the dual variable $y_{u,n}^{(k),j}$ converges to the dual optimal point. Meanwhile, under the ADMM optimization method, the subproblem $\mathcal{P}_3^{1''}$ and $\mathcal{P}_3^{2''}$ will converge to the optimum when the stopping criterion (38) is satisfied with $j \to \infty$ [27].

C. Performance Analysis

In this subsection, we analyze the performance and feasibility of our proposed algorithms.

Theorem 1: The upper and lower limits of the battery level $B_{u,n}(t)$ is bounded within $[0, \theta_{u,n} + H_{u,n}^{\max}]$ for all the MU u and time slot $t \in \mathcal{T}$.

Proof: According to the EH optimization Problem $\mathcal{P}_3^3$, we demonstrate the upper bound of $B_{u,n}(t)$ as follows:

1) When $B_{u,n}(t) \leq \theta_{u,n}$, we have $\widetilde{B}_{u,n}(t) \leq 0$. The battery should be charged according to the solution of Problem $\mathcal{P}_3^3$. Then we set $G_{u,n}^*(t) = H_{u,n}^{\max}$, and have $B_{u,n}(t+1) \leq B_{u,n}(t) + G_{u,n}^*(t) \leq \theta_{u,n} + H_{u,n}^{\max}$.
2) When $B_{u,n}(t) \geq \theta_{u,n}$, we have $\widetilde{B}_{u,n}(t) \geq 0$. We set $G_{u,n}^*(t) = 0$ according to the solution of Problem $\mathcal{P}_3^3$. It follows that $B_{u,n}(t) \leq \theta_{u,n} + H_{u,n}^{\max}$. Therefore, we have $B_{u,n}(t+1) \leq B_{u,n}(t) \leq \theta_{u,n} + H_{u,n}^{\max}$.

In conclusion, the battery energy level is always bounded as $B_{u,n}(t) \in [0, \theta_{u,n} + H_{u,n}^{\max}]$ for all MU u and $t \in \mathcal{T}$. ■

Theorem 2: Denote $\bar{E}_n$ as the average energy consumption of all MUs in the coverage area of MEC server n obtained by Algorithm 1, and the optimal of energy consumption solution of the original problem $\mathcal{P}_1$ as E_n^*. The following bound holds:

$$\overline{E}_n \leq E_n^* + \Phi^*/V + \zeta D_n^*, \tag{45}$$

where D_n^* is the optimal solution of execution delay of Problem $\mathcal{P}_1$, and $\Phi^* = \sum_{n=1}^{N} \sum_{u=1}^{U_n} \Phi$ is a constant.

Proof: Please refer to Appendix B. ■

VI. Simulation Results and Analysis

In this section, we evaluate the performance of our proposed algorithms under various system setups. We consider five cooperative MEC servers interconnected to each other operated on discrete time slots which $\tau = 1$ second. Each MEC server covers ten mobile users in its serving area. Each MU has $K = 2$ processing tasks. Based on [18], the generated workloads of each MU follow a Poisson process with rate $A_{u,n}^{(k)}(t) \in [800, 1500]$ (in workload/second), and $s_k = 0.2$Mbit. The maximum execution delay deadline of each task is randomly within $D_k^{\max} \in [2, 4] \times 10^{-3}$s and the required service rate is randomly $h_k \in [4, 8] \times 10^5$ (in CPU cycles) [9]. The harvested energy $H_{u,n}(t)$ is uniformly distributed between 2.5×10^2mJ and 3×10^2mJ, $\xi = 7.8 \times 10^{-21}$, and the allocated local processing capability $f_{u,n}^{(k)}(t)$ (in CPU cycles/second) is uniformly distributed between 40×10^5 and 100×10^5. We set the bandwidth as $W_n = 20$MHz, wireless channel gain is $C_{u,n}(t) \in [-75, -50]$(dB), and noise power $\sigma^2 = -174$dBm/Hz [20]. And the transmit power $P_{u,n}(t)$ is uniformly distributed between 2×10^{-6}W and 5×10^{-6}W. The service capacity of MEC server n is set to $\mu_n \in [10^9, 10 \times 10^9]$ (in workload/second). The mean communication time of the LAN is $\eta = 200$ms [8]. The weight for trading off energy consumption and execution delay is $\zeta = 10^{-6}$.

We compare our proposed LYP-CCMA and ADMM-DRAR schemes with the following four baseline schemes:

1) ADMM-NLB: ADMM-based distributed resource allocation but without load balancing among the MEC servers.

2) LYP-NLB: A centralized management scheme for optimizing system cost but without load balancing among the MEC servers.

3) MEC-only: In this approach, the workloads are executed by the MEC servers only without load balancing.

4) Local-only: The workloads are executed by MUs only.

We first examine load balancing among MEC servers. The relationship between load balancing and service capacity of MEC servers when $V = 1$ is presented in Fig. 2. We show the transmitted data sizes for load balancing among the five MEC servers. It can be seen that in each bar corresponding to an MEC server, most of the workloads are served by the original MEC server, which is consistent with the delay cost of the absolute priority M/M/1 queue in Section II-D3. In addition, the MEC server with a larger service capacity will execute more workloads.

In Fig. 3, we present the bandwidth allocation for 10 MUs in each coverage area of the five MEC servers. In each time

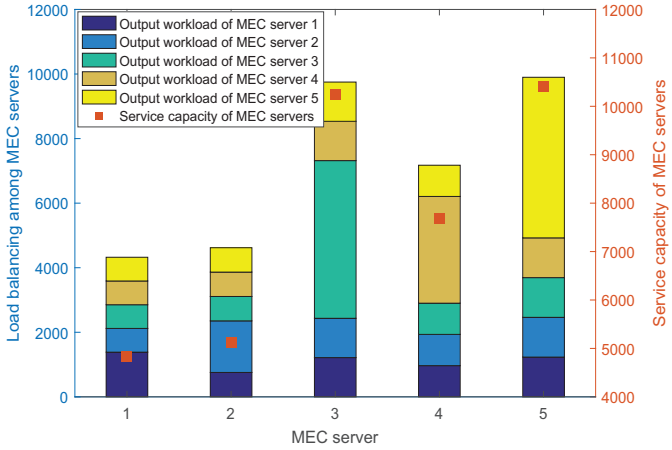

Fig. 2: Load balancing vs. service capacity of MEC servers when V = 1.

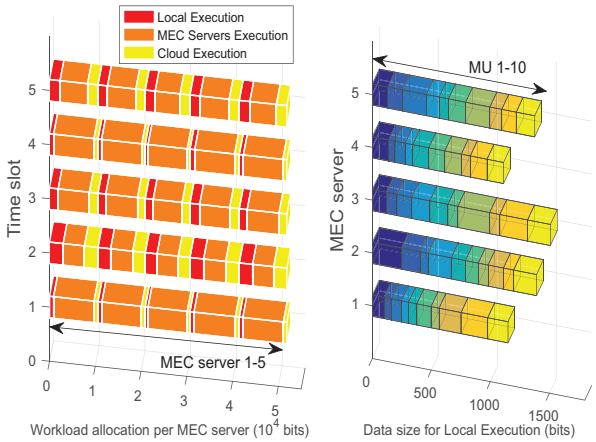

Fig. 4: The optimization of parallel offloading for multiple MEC servers and multi-users.

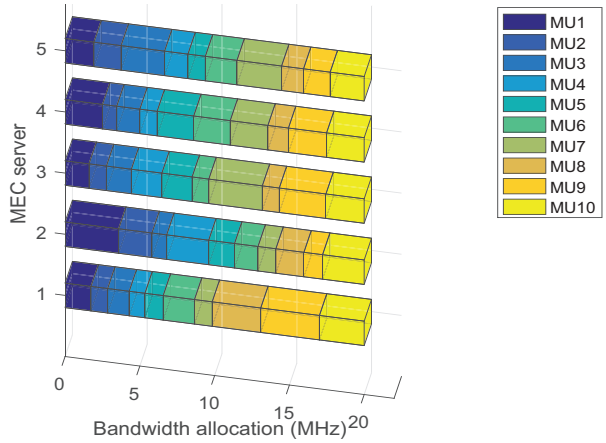

Fig. 3: Bandwidth allocation for 10 MUs in each coverage area of the five MEC servers.

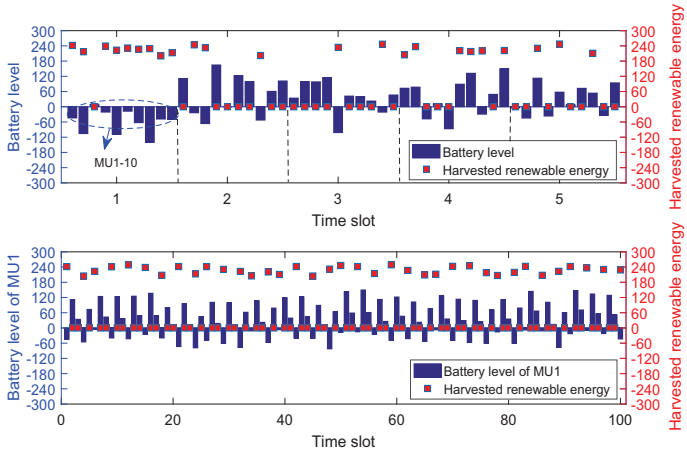

Fig. 5: Electricity charge/discharge over time (V=1).

slot, the ten MUs share the total bandwidth of 20MHz in the coverage area of each MEC server, which is reasonable and verifies the feasibility of our proposed algorithms.

Fig. 4 verifies the optimization of the parallel offloading and the feasibility of our proposed MEC system. The left part of Fig. 4 depicts the workloads allocation in the MEC system from the first to the 5th time slot when $V = 1$. We can see that the overall data sizes for local execution, MEC servers execution, and cloud execution in the five MEC servers' coverage areas. In each time slot, in order to minimize the long-term system average-cost, the SDN controller computes the workload allocation strategy depending on the generated workloads, the current computation capabilities of the MEC servers and MUs, and the state of wireless channels and the LAN link. In the right part of Fig. 4, we show the data sizes of local execution for all the MUs in the coverage areas of the five MEC servers in the same time slot. We can see the data size for locally executed workloads at each MU.

In the upper part of Fig. 5, we present the harvested renewable energy and battery levels for the 10 MUs in one MEC server's coverage area in five time slots; in the lower part of Fig. 5, we plot the battery level at one of the MUs; when $V = 1$. When the battery level is negative, the battery will be charged by harvested renewable energy. When the battery level is positive, there is no renewable energy to be charged at the end of each time slot, as given in the optimization of EH in Section V-A3. Furthermore, the battery level changes along with the MU's energy consumption and its harvested renewable energy.

Fig. 6 specifies the averaged battery level of one MU in each coverage area of the four MEC servers over a long time evolution. It can be seen that the averaged battery level basically converges to the upper limit, which verifies Theorem1 and indicates that the goal of stability of battery level is achieved. We further investigate the relationship of battery level and the control parameter V in Fig. 7. We increase the control parameter V from 1 to 10,000 to show its impact on the battery level with ADMM-DARA (i.e., Algorithm 2). We can see that the battery level increases as V is increased. In other words, a larger V value will lead to a higher battery level. Furthermore, the curve shows that the battery level has a nearly linear relationship with the control parameter V, which is consistent with Theorem 1 and Definition 2.

Finally, we conduct a comparison study of our proposed algorithms ADMM-DARA and LYP-CCMA with these four baseline schemes when $V = 1$ and $\zeta = 10^{-6}$ under the same conditions (i.e., the same workload arrivals, service capabilities, and channel states). We provide a qualitative

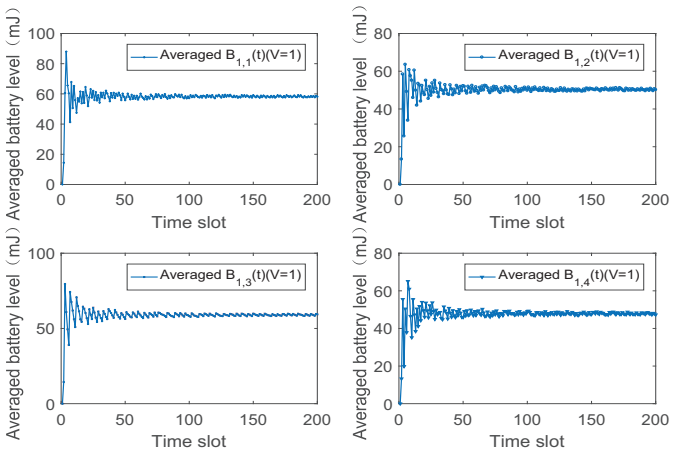

Fig. 6: Averaged battery level $B_{k,n}(t)$ for MUs vs. time slot.

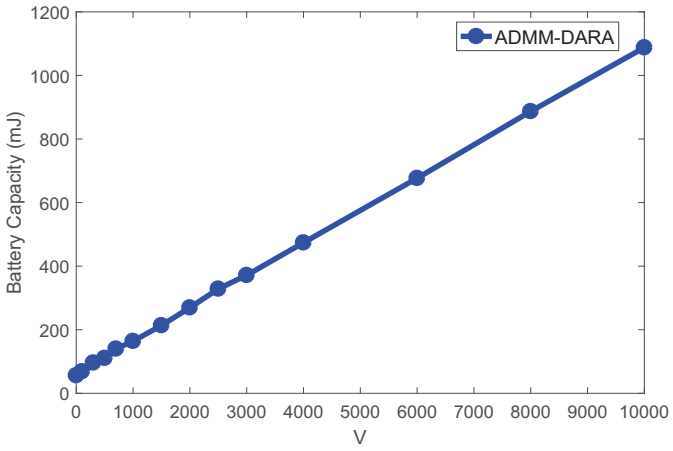

Fig. 7: Battery level vs. the control parameter V.

comparison of the existing work in Table II. The time averaged system cost results of the six schemes are plotted in Fig. 8. The time averaged system cost results of LYP-CCMA and LYP-NLB indicate the optimal system cost under centralized management. The time averaged system cost results of ADMM-DARA and ADMM-NLB are obtained with distributed resource allocation.

In the three columns of Fig. 8, it can be seen that reasonable optimization of offloading and resources allocation can achieve a smaller time averaged system cost and a good system performance. MEC-only and Local-only have relatively larger time averaged system costs due to the insufficient computing capabilities, resulting in lots of workloads transmitted to centralized cloud to execute. In the first two columns of Fig. 8, compared with LYP-CCMA and LYP-NLB, ADMM-DARA and ADMM-NLB achieve a smaller time averaged system cost. This is because the distributed algorithms optimize the uplink offloading decision of each MU, avoid the cost consumption relates to uplink data transmission of redundant workloads, and save the battery energy of MUs. Moreover, compared with ADMM-NLB and LYP-NLB, ADMM-DARA and LYP-CCMA have a smaller time averaged system cost, respectively, which is because the MEC system with load balancing among MEC servers can execute more workloads under

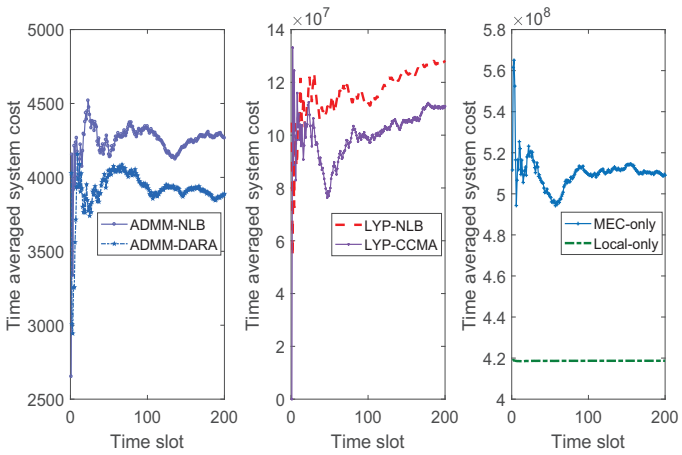

Fig. 8: Comparison of system performance (time averaged system cost) achieved by the six schemes.

the same computation capabilities and the delay constraints of MUs, leading to a lower cost for transmitting workloads to centralized remote cloud. The MEC-only and Local-only results shown in the third column are both worse than the other four schemes.

VII. Related Work

The rapid development of the Internet of Things (IoT) and rich cloud services promote the emergence of MEC, which pushes workloads execution to the edge of the network [31]. Energy harvesting (EH) technology provides a new idea for alleviating the burden on limited battery capacity of mobile devices. Therefore, EH and MEC have been recognized as promising solutions to overcome the tight resource constraints of mobile devices, such as CPU computation capability and battery capacity, which also improves the quality of service (QoS) for time-sensitive and computation-intensive workloads [32], [33]. Moreover, the tremendous growth in workloads and QoS demands of MUs have brought about enormous challenges to MEC systems with limited computing and storage capabilities compared to the cloud [34], [35].

Many existing works focus on improving the MEC system performance with limited communication and computation resources, aiming to optimize the execution delay [36], [29], energy consumption [37], and/or the overall system cost [38]. Most works investigate the computation offloading decisions [39], [30], [40], [41] to implement optimal communication and computation allocation. In [40], [41], these works usually consider that uplink offloading decisions are determined by $\{0, 1\}$ variables. For example, all arrived workloads can only be offloaded or processed by one execution method at the same time. For maximizing the computation capability utilization of cooperative-MEC system, our paper proposes a parallel uplink offloading approach [39], where the arrived workloads can be executed in part simultaneously by MUs and MEC servers, or be transmitted to centralized cloud.

The recent works [8], [9], [18] study load balancing among MEC servers using the geographical load balancing (GLB) technique proposed for data centers [42]. In [8], a novel peer offloading MEC system is proposed to optimize the

computation resources under energy constraint. The works [9], [18], perform joint GLB and admission control to optimize the system performance by leveraged the Lyapunov optimization. However, very few works study an MEC system, which not only considers the uplink offloading between an MEC server and MUs to optimize the radio access performance, but also takes into account the service capabilities of MEC servers to investigate load balancing among MEC servers. These provide a new perspective for the allocation of communication and computation resources in MEC systems with multi-users and multiple MEC servers.

Several works adopt deep reinforcement learning (DRL) to solve the resource allocation problem in MEC systems or virtual eage computing systems [43], [33], [44]. In [44], the authors propose a hierarchical reinforcement learning algorithm to optimize joint pushing and workload caching in MEC networks with multi-users and multi-cast data. The recent work [45] proposes a DRL-based online offloading framework to optimize task offloading decisions and the allocation of wireless resources. In contrast, we develop a Lyapunov-based centralized energy and delay management algorithm and an ADMM-based distributed method because of its low computational complexity and fast convergence characteristics, which also does not need to maintain a large state-action space.

VIII. Conclusions

In this article, the resource optimization problem has been investigated for an MEC system consisting of multiple MEC servers and multiple users, which included the optimization of communication and computation resources among multi-users and the optimization of load balancing among multiple MEC servers. We formulated a stochastic programming problem for resource optimization. To solve this problem, a Lyapunov-based centralized energy and delay management algorithm was first proposed for achieving the optimal system cost under battery level stability constraints. Furthermore, based on ADMM, algorithms were designed for implementing distributed resources allocation relate to communication, computation, and energy resources of each MU. Finally, the efficacy of the proposed algorithms are validated with our simulation study.

References

[1] K. Poularakis, J. Llorca, A. M. Tulino, I. Taylor, and L. Tassiulas, "Joint service placement and request routing in multi-cell mobile edge computing networks," in *Proc. IEEE INFOCOM'19*, Paris, France, Apr./May 2019, pp. 10–18.

[2] X. Chen, Q. Shi, L. Yang, and J. Xu, "ThriftyEdge: Resource-efficient edge computing for intelligent IoT applications," *IEEE Netw.*, vol. 32, no. 1, pp. 61–65, Jan.-Feb. 2018.

[3] T. Ouyang, R. Li, X. Chen, Z. Zhou, and X. Tang, "Adaptive user-managed service placement for mobile edge computing: An online learning approach," in *Proc. IEEE INFOCOM'19*, Paris, France, Apr./May 2019, pp. 1468–1476.

[4] Y. Wu, L. P. Qian, J. Zheng, H. Zhou, and X. S. Shen, "Green-oriented traffic offloading through dual connectivity in future heterogeneous small cell networks," *IEEE Commun.*, vol. 56, no. 5, pp. 140–147, May 2018.

[5] R. Roman, J. Lopez, and M. Mambo, "Mobile edge computing, fog et al.: A survey and analysis of security threats and challenges," *Future Generation Computer Systems*, vol. 78, no. PT.2, pp. 680–698, Jan. 2016.

[6] J. Wang, L. Zhao, J. Liu, and N. Kato, "Smart resource allocation for mobile edge computing: A deep reinforcement learning approach," *IEEE Trans. Emerg. Topics Comput.*, vol. 9, no. 3, pp. 1529–1541, July-Sept. 2021.

[7] F. Tang, Z. M. Fadlullah, B. Mao, and N. Kato, "An intelligent traffic load prediction-based adaptive channel assignment algorithm in SDN-IoT: A deep learning approach," *IEEE Internet Things J.*, vol. 5, no. 6, pp. 5141–5154, Dec. 2018.

[8] 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.

[9] H. Wu, L. Chen, C. Shen, W. Wen, and J. Xu, "Online geographical load balancing for energy-harvesting mobile edge computing," in *Proc. IEEE ICC'18*, Kansas City, MO, May 2018, pp. 1–6.

[10] M. Sheng, Y. Wang, X. Wang, and J. Li, "Energy-efficient multiuser partial computation offloading with collaboration of terminals, radio access network, and edge server," *IEEE Trans. Commun.*, vol. 68, no. 3, pp. 1524–1537, Mar. 2020.

[11] M. Chen, B. Liang, and M. Dong, "Joint offloading and resource allocation for computation and communication in mobile cloud with computing access point," in *Proc. IEEE INFOCOM'17*, Atlanta, GA, May 2017, pp. 1–9.

[12] Y. Lin, L. Feng, W. Li, F. Zhou, and Q. Ou, "Stochastic joint bandwidth and computational allocation for multi-users and multi-edge-servers in 5G D-RANs," in *Proc. IEEE SmartCloud'19*, Tokyo, Japan, Dec. 2019, pp. 65–70.

[13] C. You, K. Huang, H. Chae, and B. Kim, "Energy-efficient resource allocation for mobile-edge computation offloading," *IEEE Trans. Wireless Commun.*, vol. 16, no. 3, pp. 1397–1411, Mar. 2017.

[14] J. Oueis, E. C. Strinati, S. Sardellitti, and S. Barbarossa, "Small cell clustering for efficient distributed fog computing: A multi-user case," in *Proc. IEEE VTC-Fall'15*, Boston, MA, Sept. 2015, pp. 1–5.

[15] H. A. Alameddine, S. Sharafeddine, S. Sebbah, S. Ayoubi, and C. Assi, "Dynamic task offloading and scheduling for low-latency iot services in multi-access edge computing," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 3, pp. 668–682, Mar. 2019.

[16] D. C. Nguyen, P. N. Pathirana, M. Ding, and A. Seneviratne, "Privacy-preserved task offloading in mobile blockchain with deep reinforcement learning," *IEEE Trans. Netw. Service Manag.*, vol. 17, no. 4, pp. 2536–2549, 2020.

[17] M. Chen, M. Dong, and B. Liang, "Resource sharing of a computing access point for multi-user mobile cloud offloading with delay constraints," *IEEE Trans. Mobile Comput.*, vol. 17, no. 12, pp. 2868–2881, Dec. 2018.

[18] L. Chen, J. Xu, and S. Zhou, "Computation peer offloading in mobile edge computing with energy budgets," in *Proc. IEEE GLOBECOM'17*, Singapore, Dec. 2017, pp. 1–6.

[19] Y. Ye, D. Qiu, X. Wu, G. Strbac, and J. Ward, "Model-free real-time autonomous control for a residential multi-energy system using deep reinforcement learning," *IEEE Trans. Smart Grid*, vol. 11, no. 4, pp. 3068–3082, 2020.

[20] 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, Sept. 2017.

[21] J. Du, F. R. Yu, X. Chu, J. Feng, and G. Lu, "Computation offloading and resource allocation in vehicular networks based on dual-side cost minimization," *IEEE Trans. Veh. Technol.*, vol. 68, no. 2, pp. 1079–1092, Feb. 2019.

[22] S. Ko, K. Han, and K. Huang, "Wireless networks for mobile edge computing: Spatial modeling and latency analysis," *IEEE Trans. Wireless Commun.*, vol. 17, no. 8, pp. 5225–5240, Aug. 2018.

[23] R. Cooper, *Introduction to Queueing Theory*. Amsterdam, The Netherlands: Academic Press, 1981.

[24] S. Ross, *Introduction to Probability Models*. Amsterdam, The Netherlands: Academic Press, 2014.

[25] H. Jabbar, Y. S. Song, and T. T. Jeong, "Rf energy harvesting system and circuits for charging of mobile devices," *IEEE Trans. Consum. Electron.*, vol. 56, no. 1, pp. 247–253, Feb. 2010.

[26] S. Sun, M. Dong, and B. Liang, "Distributed real-time power balancing in renewable-integrated power grids with storage and flexible loads," *IEEE Trans. Smart Grid*, vol. 7, no. 5, pp. 2337–2349, Sept. 2016.

[27] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Found. Trends Mach. Learn.*, vol. 3, no. 1, pp. 1–122, Jan. 2010.

[28] M. Akbari, B. Gharesifard, and T. Linder, "Individual regret bounds for the distributed online alternating direction method of multipliers," *IEEE Trans. Automat. Contr.*, vol. 64, no. 4, pp. 1746–1752, Apr. 2019.

[29] Y. Wang, X. Tao, X. Zhang, P. Zhang, and Y. T. Hou, "Cooperative task offloading in three-tier mobile computing networks: An ADMM

framework," *IEEE Trans. Veh. Technol.*, vol. 68, no. 3, pp. 2763–2776, Mar. 2019.

[30] J. Xu, L. Chen, and P. Zhou, "Joint service caching and task offloading for mobile edge computing in dense networks," in *Proc. IEEE INFOCOM'18*, Honolulu, HI, Apr. 2018, pp. 207–215.

[31] 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.

[32] Y. Xu and S. Mao, "A survey of mobile cloud computing for rich media applications," *IEEE Wireless Commun.*, vol. 20, no. 3, pp. 46–53, June 2013.

[33] 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, June 2019.

[34] D. Li, Y. Han, C. Wang, G. Shi, X. Wang, X. Li, and V. C. M. Leung, "Deep reinforcement learning for cooperative edge caching in future mobile networks," in *Proc. IEEE WCNC'19*, Marrakech, Morocco, Mar. 2019, pp. 1–6.

[35] L. Zeng, E. Li, Z. Zhou, and X. Chen, "Boomerang: On-demand cooperative deep neural network inference for edge intelligence on the industrial Internet of Things," *IEEE Netw.*, vol. 33, no. 5, pp. 96–103, Sept.-Oct. 2019.

[36] M. Chen and Y. Hao, "Task offloading for mobile edge computing in software defined ultra-dense network," *IEEE J. Sel. Areas Commun.*, vol. 36, no. 3, pp. 587–597, Mar. 2018.

[37] Q. Zeng, Y. Du, K. Huang, and K. K. Leung, "Energy-efficient resource management for federated edge learning with CPU-GPU heterogeneous computing," *IEEE Trans. Wireless Commun.*, vol. 20, no. 12, pp. 7947–7962, Dec. 2021.

[38] L. Zhang and N. Ansari, "Optimizing the operation cost for UAV-aided mobile edge computing," *IEEE Trans. Veh. Technol.*, vol. 70, no. 6, pp. 6085–6093, June 2021.

[39] A. Ndikumana, N. H. Tran, T. M. Ho, Z. Han, W. Saad, D. Niyato, and C. S. Hong, "Joint communication, computation, caching, and control in big data multi-access edge computing," *IEEE Trans. Mobile Comput.*, vol. 19, no. 6, pp. 1359–1374, June 2020.

[40] Y. Xu, B. Gu, R. Q. Hu, D. Li, and H. Zhang, "Joint computation offloading and radio resource allocation in MEC-based wireless-powered backscatter communication networks," *IEEE Trans. Veh. Technol.*, vol. 70, no. 6, pp. 6200–6205, June 2021.

[41] N. Eshraghi and B. Liang, "Joint offloading decision and resource allocation with uncertain task computing requirement," in *Proc. IEEE INFOCOM'19*, Apr./May, Paris, France 2019, pp. 1414–1422.

[42] J. Luo, L. Rao, and X. Liu, "Spatio-temporal load balancing for energy cost optimization in distributed internet data centers," *IEEE Trans. Cloud Comput.*, vol. 3, no. 3, pp. 387–397, July-Sept. 2015.

[43] J. Feng, F. Richard Yu, Q. Pei, X. Chu, J. Du, and L. Zhu, "Cooperative computation offloading and resource allocation for blockchain-enabled mobile-edge computing: A deep reinforcement learning approach," *IEEE Internet Things J.*, vol. 7, no. 7, pp. 6214–6228, July 2020.

[44] Y. Qian, R. Wang, J. Wu, B. Tan, and H. Ren, "Reinforcement learning-based optimal computing and caching in mobile edge network," *IEEE J. Sel. Areas Commun.*, vol. 38, no. 10, pp. 2343–2355, Oct. 2020.

[45] L. Huang, S. Bi, and Y. J. A. Zhang, "Deep reinforcement learning for online computation offloading in wireless powered mobile-edge computing networks," *IEEE Trans. Mobile Comput.*, vol. 19, no. 11, pp. 2581–2593, Nov. 2020.

APPENDIX A
PROOF OF THE UPPER BOUND $\Delta[\widetilde{B}_{u,n}(t)]$ IN LEMMA 1

Proof: Substituting the Lyapunov function (25) into the one-slot conditional Lyapunov drift (26), and noting that $\widetilde{B}_{u,n}(t) = B_{u,n}(t) - \theta_{u,n}$, we have

$$\begin{aligned}
\Delta[\widetilde{B}_{u,n}(t)] &= \frac{1}{2}\sum_{n=1}^{N}\sum_{u=1}^{U_n}\mathbb{E}\Big\{\widetilde{B}_{u,n}^2(t+1) - \widetilde{B}_{u,n}^2(t)\Big\} \\
&= \frac{1}{2}\sum_{n=1}^{N}\sum_{u=1}^{U_n}\mathbb{E}\Big\{[B_{u,n}(t+1)-\theta_{u,n}]^2 - [B_{u,n}(t)-\theta_{u,n}]^2\Big\} \\
&= \frac{1}{2}\sum_{n=1}^{N}\sum_{u=1}^{U_n}\mathbb{E}\Big\{[B_{u,n}(t+1) - B_{u,n}(t)] \\
&\quad \times [B_{u,n}(t+1) + B_{u,n}(t) - 2\theta_{u,n}]\Big\}. \qquad (46)
\end{aligned}$$

Substitute (20), i.e., the evolution of the battery level of MU u into (46), we have

$$\begin{aligned}
\Delta[\widetilde{B}_{u,n}(t)] &= \frac{1}{2}\sum_{n=1}^{N}\sum_{u=1}^{U_n}\mathbb{E}\Big\{\widetilde{B}_{u,n}^2(t+1) - \widetilde{B}_{u,n}^2(t)\Big\} \\
&= \frac{1}{2}\sum_{n=1}^{N}\sum_{u=1}^{U_n}\mathbb{E}\Big\{2[B_{u,n}(t)-\theta_{u,n}][G_{u,n}(t)-E_{u,n}(t)] \\
&\quad + [G_{u,n}(t)-E_{u,n}(t)]^2\Big\} \\
&\le \mathbb{E}\Big\{\sum_{n=1}^{N}\sum_{u=1}^{U_n}[(B_{u,n}(t)-\theta_{u,n})(G_{u,n}(t)-E_{u,n}(t))] \\
&\quad + \frac{1}{2}[(H_{u,n}^{\max})^2 + ((\widetilde{E}_{u,n}^{\max})^2]\Big\} \\
&= \mathbb{E}\Big\{\sum_{n=1}^{N}\sum_{u=1}^{U_n}[\widetilde{B}_{u,n}(t)(G_{u,n}(t)-E_{u,n}(t)) + \Phi]\Big\}. \qquad (47)
\end{aligned}$$

This concludes the proof. ■

APPENDIX B
PROOF OF THEOREM 2

Proof: We take expectation and sum over the period $[0, T-1]$ on both sides of (20) for all MUs in the coverage area of MEC server n. We have

$$\mathbb{E}\{E_{u,n}(T)\} - \mathbb{E}\{E_{u,n}(0)\} = \sum_{t=0}^{T-1}[\mathbb{E}\{G_{u,n}(t)\} - \mathbb{E}\{E_{u,n}(t)\}]. \qquad (48)$$

Dividing both sides by T and letting T go to infinity. We obtain the relationship about the charge-discharge of battery under the long-term evolution as

$$\lim_{T\to\infty}\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\{G_{u,n}(t)\} = \lim_{T\to\infty}\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\{E_{u,n}(t)\}. \qquad (49)$$

We then have relaxed problem of $\mathcal{P}_1$ as follows.

$$\begin{aligned}
\widehat{\mathcal{P}}: &\min_{\mathbf{\Psi}(t)}\lim_{T\to\infty}\frac{1}{T}\sum_{t=0}^{T-1}\sum_{n=1}^{N}\mathbb{E}\{E_n(t) + \zeta D_n(t)\} \\
&\text{s.t. (5), (9), (10), (21), (23), (49).}
\end{aligned}$$

Since $\widehat{P}$ is obtained with relaxed constraints of $\mathcal{P}_1$, instead to solve the original problem $\mathcal{P}_1$, we can obtain the optimal solution of Problem $\widehat{P}$ by exploiting the drift-plus-penalty

function (29). Denote $\widehat{E}_n$ as the optimal solution of $\widehat{\mathcal{P}}$, satisfying $\widehat{E}_n \leq E_n^*$. We have

$$\begin{aligned}
&\Delta[\widetilde{B}_{u,n}(t)] + V\sum_{n=1}^{N}\mathbb{E}\{E_n(t)+\zeta D_n(t)\} \\
&\leq \mathbb{E}\Big\{\sum_{n=1}^{N}\sum_{u=1}^{U_n}[\widetilde{B}_{u,n}(t)(\widehat{G}_{u,n}(t)-\widehat{E}_{u,n}(t))+\Phi]\Big\} \\
&\quad + V\sum_{n=1}^{N}\mathbb{E}\{\widehat{E}_n(t)+\zeta\widehat{D}_n(t)\}. \qquad (50)
\end{aligned}$$

Note that $\mathbb{E}\{\widehat{G}_{u,n}(t)-\widehat{E}_{u,n}(t)\}=0$. Summing up from 0 to $T-1$, we obtain

$$\begin{aligned}
&\sum_{t=0}^{T-1}\sum_{n=1}^{N}V\mathbb{E}\{E_n(t)+\zeta D_n(t)\} \\
&\leq \mathbb{E}\{L[\widetilde{B}_{u,n}(0)]\} - \mathbb{E}\{L[\widetilde{B}_{u,n}(T)]\} + T\sum_{n=1}^{N}\sum_{u=1}^{U_n}\Phi \\
&\quad + VT\sum_{n=1}^{N}\mathbb{E}\{\widehat{E}_n(t)+\zeta\widehat{D}_n(t)\} \\
&\leq \mathbb{E}\{L[\widetilde{B}_{u,n}(0)]\} + T\sum_{n=1}^{N}\sum_{u=1}^{U_n}\Phi \\
&\quad + VT\sum_{n=1}^{N}\mathbb{E}\{\widehat{E}_n(t)+\zeta\widehat{D}_n(t)\}. \qquad (51)
\end{aligned}$$

We subtract the term $-\mathbb{E}\{L[\widetilde{B}_{u,n}(T)]\}$ in the second inequality of (51) due to the nonnegative property of Lyapunov function. We divide both sides by $V\cdot T$, and let T go to infinity. It follows that

$$\begin{aligned}
&\lim_{T\to\infty}\frac{1}{T}\sum_{t=0}^{T-1}\sum_{n=1}^{N}\mathbb{E}\{E_n(t)+\zeta D_n(t)\} \\
&\leq \lim_{T\to\infty}\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\{L[\widetilde{B}_{u,n}(0)]\} + \frac{1}{V}\sum_{n=1}^{N}\sum_{u=1}^{U_n}\Phi \\
&\quad + \sum_{n=1}^{N}(E_n^*+D_n^*), \qquad (52)
\end{aligned}$$

where $\lim_{T\to\infty}(1/T)\sum_{t=0}^{T-1}\mathbb{E}\{L[\widetilde{B}_{u,n}(0)]\}=0$.

Then we obtain the average energy consumption $\bar{E}_n = \lim_{T\to\infty}(1/T)\sum_{t=0}^{T-1}\mathbb{E}\{E_n(t)\} \leq E_n^* + (\Phi^*/V) + D_n^*$. ■

Wenqian Zhang received the B.S. degree in Electronic Information Science and Technology from Binzhou University in 2016. She is in successive postgraduate and doctoral program since 2016 with Department of Communication Engineering, College of Information Science and Technology, Donghua University, Shanghai, China. From 2019 to 2020, she was a visiting student with the Department of Electrical and Computer Engineering, Auburn University, Auburn, AL. Her research interests are focused on solving the communication and computation resources allocation, task offloading optimization, workload balancing, service caching and energy-delay tradeoff in mobile edge computing system.

Guanglin Zhang received the Ph.D. degree in information and communication engineering from Shanghai Jiao Tong University in 2012. From 2013 to 2014, he was a Postdoctoral Research Associate with the Institute of Network Coding, Chinese University of Hong Kong. He is currently a professor and the department chair with the Department of Communication Engineering, and he is the vice dean with the College of Information Science and Technology, Donghua University. His research interests include capacity scaling of wireless networks, vehicular networks, smart micro-grid, and mobile edge computing. He serves as a Technical Program Committee Member for IEEE Globecom 2016-2017, IEEE ICC 2014, 2015, 2017, IEEE VTC2017-Fall, IEEE/CIC ICCC 2014, and WCSP 2014, APCC 2013, and WASA 2012. He serves as the Local Arrangement Chair of ACM TURC 2017, and Vice TPC Co-Chair of ACM TURC 2018. He serves as Editors on the Editorial Board of China Communications and Journal of Communications and Information Networks.

Shiwen Mao [S'99-M'04-SM'09-F'19] received his Ph.D. in electrical engineering from Polytechnic University, Brooklyn, NY in 2004. After joining Auburn University, Auburn, AL in 2006, he held the McWane Endowed Professorship from 2012 to 2015 and the Samuel Ginn Endowed Professorship from 2015 to 2020 in the Department of Electrical and Computer Engineering. Currently, he is a professor and Earle C. Williams Eminent Scholar, and Director of the Wireless Engineering Research and Education Center at Auburn University. His research interest includes wireless networks, multimedia communications, and smart grid. He is an Associate Editor-in-Chief of IEEE/CIC China Communications, an Area Editor of IEEE Transactions on Wireless Communications, IEEE Internet of Things Journal, IEEE Transactions on Network Science and Engineering, IEEE Open Journal of the Communications Society, and ACM GetMobile, and an Associate Editor of IEEE Transactions on Cognitive Communications and Networking, IEEE Transactions on Mobile Computing, IEEE Multimedia, and IEEE Networking Letters, among others. He is a Distinguished Lecturer of IEEE Communications Society and IEEE RFID Council. He is the General Chair of IEEE INFOCOM 2022, the TPC Co-Chair of IEEE INFOCOM 2018, and the TPC Vice Chair of IEEE GLOBECOM 2022. He received the IEEE ComSoc TC-CSR Distinguished Technical Achievement Award in 2019 and NSF CAREER Award in 2010. He is a co-recipient of the 2021 Best Paper Award of Elsevier/KeAi Digital Communications and Networks Journal, the 2021 IEEE Internet of Things Journal Best Paper Award, the 2021 IEEE Communications Society Outstanding Paper Award, the IEEE Vehicular Technology Society 2020 Jack Neubauer Memorial Award, the IEEE ComSoc MMTC 2018 Best Journal Award and 2017 Best Conference Paper Award, the Best Demo Award of IEEE SECON 2017, the Best Paper Awards from IEEE GLOBECOM 2019, 2016 & 2015, IEEE WCNC 2015, and IEEE ICC 2013, and the 2004 IEEE Communications Society Leonard G. Abraham Prize in the Field of Communications Systems.