AUTHOR QUERIES

AUTHOR PLEASE ANSWER ALL QUERIES

PLEASE NOTE: We cannot accept new source files as corrections for your paper. If possible, please annotate the PDF proof we have sent you with your corrections and upload it via the Author Gateway. Alternatively, you may send us your corrections in list format. You may also upload revised graphics via the Author Gateway.

Carefully check the page proofs (and coordinate with all authors); additional changes or updates WILL NOT be accepted after the article is published online/print in its final form. Please check author names and affiliations, funding, as well as the overall article for any errors prior to sending in your author proof corrections.

AQ1: According to our records, Zehui Xiong is listed as a Senior Member, IEEE. However, the files provided list them as a Member, IEEE. Please verify.

AQ2: Please confirm or add details for any funding or financial support for the research of this article.

AQ3: Please advise where Table I should be mentioned in the text.

AQ4: Please provide the year attained for Ph.D. degree of Z. Xiong.

Thanks for the feedback! For your concerns:

AQ1: Yes, Zehui Xiong is a Senior Member.

AQ2: The following two funding information needs to be added:

1. This research is supported by the National Research Foundation, Singapore, and Infocomm Media Development Authority under its Future Communications Research & Development Programme, DSO National Laboratories under the AI Singapore Programme (AISG Award No: AISG2-RP-2020-019 and FCP-ASTAR-TG-2022-003), and MOE Tier 1 (RG87/22).
2. The research is also supported by the SUTD SRG-ISTD-2021-165, and the Ministry of Education, Singapore, under its SMU-SUTD Joint Grant (22-SIS-SMU-048).

AQ3: Table 1 should be introduced at the beginning of Section III, where you can use the statement: "..toward resource allocation...For convenience, we summarize the notations commonly used in this paper in Table 1."

AQ4: Dr. Zehui Xiong got his PhD in 2020.

PS: Zhilin Wang's affiliation should be adjusted. It should be "Purdue University, Indianapolis, IN, USA". Besides, in his biography, he should not be stated as "Dr. Wang" since he is doing his PhD now.

The above information can also be founded in comments.

Resource Optimization for Blockchain-Based Federated Learning in Mobile Edge Computing

AQ1 Zhilin Wang, *Graduate Student Member, IEEE*, Qin Hu, Zehui Xiong, *Senior Member, IEEE*, Yuan Liu, and Dusit Niyato, *Fellow, IEEE*

***Abstract*—With the booming of mobile edge computing (MEC) and blockchain-based blockchain-based federated learning (BCFL), more studies suggest deploying BCFL on edge servers. In this case, edge servers with restricted resources face the dilemma of serving both mobile devices for their offloading tasks and the BCFL system for model training and blockchain consensus without sacrificing the service quality to any side. To address this challenge, this article proposes a resource allocation scheme for edge servers to provide optimal services at the minimum cost. Specifically, we first analyze the energy consumption of the MEC and BCFL tasks, considering the completion time of each task as the service quality constraint. Then, we model the resource allocation challenge into a multivariate, multiconstraint, and convex optimization problem. While solving the problem in a progressive manner, we design two algorithms based on the alternating direction method of multipliers (ADMMs) in both homogeneous and heterogeneous situations, where equal and on-demand resource distribution strategies are, respectively, adopted. The validity of our proposed algorithms is proved via rigorous theoretical analysis. Moreover, the convergence and efficiency of our proposed resource allocation schemes are evaluated through extensive experiments.**

***Index Terms*—Alternating direction method of multiplier (ADMM), blockchain, federated learning (FL), mobile edge computing (MEC), resource allocation.**

I. Introduction

VARIOUS embedded sensors are widely deployed on mobile devices, enabling them to pervasively perceive the physical world and collect an extensive amount of data. With the advances in hardware technology, it becomes promising for devices to process the collected data locally, such as training machine learning models. However, as the resources of mobile devices are usually inadequate, they may experience difficulty finishing computing-intensive tasks, which drives the emergence of mobile edge computing (MEC). Its basic idea is to facilitate mobile devices offloading computing tasks to their nearby edge servers with sufficient resources and then obtain the calculated results with communication efficiency in close proximity [1], [2]. MEC has been applied to many fields, such as the Internet of Things (IoT) [3], smart healthcare [4], and smart transportation [5].

To address the main challenges of federated learning (FL) [6], such as the single point of failure and the privacy protection of model updates, blockchain has been extensively used to assist in achieving full decentralization with security [7], [8], which is termed as blockchain-based FL (BCFL). This new framework connects participants in FL, i.e., clients, through the blockchain network and requires them to complete both FL and blockchain related operations, such as model training and block generation [9]. As for a client in BCFL, it consumes a large number of resources in completing the BCFL task, making it an impractical job for battery-powered mobile devices with constrained resources. To address this issue, researchers advocate deploying BCFL at the edge as edge servers usually have stronger computing, communication, and storage capabilities for FL model training and blockchain consensus [10], [11].

In this case, the MEC servers are responsible for completing both the BCFL and MEC tasks. For the MEC task, the edge server is required to allocate the communication resource (e.g., bandwidth) for data transferring, the storage resource for data caching, and the computing resources (e.g., CPU cycle frequency) for computation to mobile devices. Similarly, for the BCFL task, the edge server needs to distribute the communication resource for sharing model updates and reaching consensus among blockchain nodes, the storage resource for saving the copy of blockchain data and local training data, and the computing resource for FL model training and updating, as well as the generation of new blocks. Generally, both tasks result in heavy consumption of resources, leading to congestion over resource allocation at edge servers. Since both the MEC and BCFL tasks are usually time-sensitive, the servers have to deal with the limited resource challenges of serving both the lower layer mobile devices and the upper layer BCFL system without significant delay, which makes it necessary to design optimal resource allocation schemes for them.

Manuscript received 2 August 2023; revised 1 October 2023 and 22 November 2023; accepted 13 December 2023. AQ2 This work was supported in part by U.S. NSF under Grant CNS-2105004. *(Corresponding author: Qin Hu.)*

Zhilin Wang is with the Department of Computer Science, Purdue University at Indianapolis, Indianapolis, IN 46202 USA (e-mail: wang5327@purdue.edu).

Qin Hu is with the Department of Computer Science, Indiana University at Indianapolis, Indianapolis, IN 46202 USA (e-mail: qinhu@iu.edu).

Zehui Xiong is with the Pillar of Information Systems Technology and Design, Singapore University of Technology Design, Singapore (e-mail: zehui_xiong@sutd.edu.sg).

Yuan Liu is with the Cyberspace Institute of Advanced Technology, Guangzhou University, Guangzhou 510006, Guangdong, China (e-mail: liuyuan@swc.neu.edu.cn).

Dusit Niyato is with the School of Computer Science and Engineering, Nanyang Technological University, Singapore (e-mail: dniyato@ntu.edu.sg).

Digital Object Identifier 10.1109/JIOT.2023.3347524

Relevant research about resource allocation at edge servers usually focuses on assigning resources to each mobile device for finishing the requested MEC task [12], [13], [14] or distributing resources for model training and block generation processes in the BCFL task [15], [16], [17]. Although state-of-the-art studies can help edge servers allocate resources to well handle either the MEC or BCFL task, these mechanisms have never considered resource conflicts when both tasks are running on servers at the same time.

To fill the gap, we design a resource allocation scheme that allows the edge server to finish both the MEC and BCFL tasks simultaneously and timely. Specifically, we define the cost as the total energy consumed by the edge server in completing both the MEC and BCFL tasks, and then use the corresponding time requirements as the constraints on the quality of services provided by the edge server. We can transform the resource allocation problem into a multivariate, multiconstrained, and convex optimization problem. However, solving this optimization problem faces the following challenges: 1) there are multiple resource-related variables since assigning resources to the MEC task means making decisions on resource allocation to each device, resulting in the number of variables increasing with the device quantity and 2) there are multiple constraints of resource and service quality, making the solution nontrivial.

These two challenges invalidate the application of traditional optimization methods for multiple variable calculations. Therefore, we design a scheme based on a distributed optimization algorithm, i.e., the alternating direction method of multipliers (ADMMs), which determines multiple variables by iterations in a distributed manner [18]. For a better understanding of the solutions, we devise ADMM-based algorithms for the resource allocation problem in two progressive scenarios. Specifically, we first apply *modified general ADMM (G-ADMM)* (MG-ADMM) for the homogeneous scenario with all the MEC tasks having the same data size and time requirement, which distributes resources to each local device equally; then we design the *modified consensus ADMM* (MC-ADMM)-based algorithm to assign resources to devices on demand in the heterogeneous scenario with MEC tasks having different data sizes and time requirements. Finally, we conduct extensive experiments to testify to the convergence and effectiveness of our proposed resource allocation schemes.

To the best of our knowledge, we are the first to tackle the challenge of resource conflict at edge servers in the implementation of edge-based BCFL. Our contributions can be summarized as follows.

1) We formulate the resource allocation problem of BCFL in MEC as a multivariate and multiconstraint optimization problem, with the solution of a resource allocation scheme for edge servers to handle both the MEC and BCFL tasks simultaneously within the time requirements.
2) To solve the optimization problem, we design two algorithms named MG-ADMM and MC-ADMM for homogeneous and heterogeneous scenarios, respectively.
3) To ensure the convergence of algorithms for more than two variables, we add regularization terms in our proposed algorithms based on MG-ADMM and MC-ADMM with theoretical proof.
4) We conduct extensive experimental evaluations to prove that the optimization solutions are valid and our proposed resource allocation schemes are effective.

The rest of this article is organized as follows. We introduce the system model and problem formulation in Section III. The MG-ADMM algorithm to solve the optimization problem in the homogeneous scenario and the MC-ADMM algorithm for the heterogeneous scenario are displayed in Sections IV and V, respectively. Experimental evaluations are presented in Section VI. We discuss the related work in Section II. Finally, we conclude this article in Section VII. The detailed proofs of theorems are presented in the Appendix.

II. Related Work and Background

In this section, we discuss the state-of-the-art research correlated to BCFL in MEC and introduce some preliminaries about ADMM algorithms.

A. Recent Advances of Deploying BCFL in MEC

Recently, there are many studies focusing on deploying BCFL on edge servers. Zhao et al. [10] designed a BCFL system running at the edge with edge servers being responsible for collecting and training the local models, where a device selection mechanism and incentive scheme are proposed to facilitate the performance of the crowdsensing. Rehman et al. [19] devise a blockchain-based reputation-aware fine-gained FL system to enhance the trustworthiness of devices in the MEC system. The work in [20] tries to address the privacy protection issue for BCFL in MEC via resisting a novel property inference attack, which attempts to cause unintended property leakage. Hu et al. [11] deploy a BCFL framework on the MEC edge servers to facilitate finishing mobile crowdsensing tasks, which aims to achieve privacy preservation and incentive rationality at the same time. Qu et al. [21] provide a simulation platform for BCFL in the MEC environment to measure the quality of local updates and configurations of IoT devices. Huang et al. [22] proposed a BCFL framework with the aid of edge servers to address communication delay and security issues. By integrating blockchain and edge computing technologies, BD-FL is proposed for decentralizing FL and solving the incentive issue for participants [23]. From these studies, it can be concluded that the development of BCFL in MEC is promising, even though there are still some challenges that should be tackled.

Specifically, resource allocation is one of the crucial but open challenges. Since the resources of edge servers are usually limited, it is essential to design a resource allocation scheme for edge servers to provide satisfactory services for both the MEC and the BCFL tasks with minimum cost. Wang et al. [24] designed a joint resource allocation mechanism in BCFL, which assists the participants in deciding the proper resources for completing training and mining tasks. Zhang et al. [17] proposed a resource allocation scheme to reduce energy cost and maintain the convergence rate of the FL model by jointly considering the channel allocation, bloc size adjustment, and block generator selection. In [25], a hybrid blockchain-assisted resource trading system is designed to achieve decentralization and efficiency for FL in MEC. Li et al. [16] proposed a BCFL framework to tackle the security and privacy challenges of FL, where a computing resource allocation mechanism for training and mining is also designed by optimizing the upper bound of the global loss function. One main vulnerability of this scheme is that all participants are assumed to be homogeneous, which is clearly impractical in the mobile scenario.

In summary, none of the existing studies related to implementing BCFL in MEC has ever addressed the resource allocation challenge between the MEC tasks and the BCFL task. Because of the dual roles of edge servers in BCFL and MEC, they have to simultaneously finish the upper layer BCFL task and provide MEC services for the lower layer mobile devices. To fill this gap, we devise resource allocation schemes for edge servers in the deployment of BCFL at the edge to guarantee service quality to both sides at the minimum cost.

B. Introduction to ADMM

According to Boyd et al. [18], the ADMMs, combining dual ascend and dual decomposition, is designed to solve problems that are multivariate, separable, and convex.

1) MG-ADMM: First, we introduce *G-ADMM* as the basis of MG-ADMM. G-ADMM tries to solve the following problem:

$$\begin{aligned} &\arg\min_{x,z} \; f(x) + g(z) \\ &\text{s.t.:} \; Ax + Bz = c \end{aligned}$$

where $x \in \mathbb{R}^n$, $z \in \mathbb{R}^m$, $A \in \mathbb{R}^{p*n}$, $B \in \mathbb{R}^{p*m}$, and $c \in \mathbb{R}^p$. Functions $f(x)$ and $g(z)$ are convex regarding x and z. The objective of G-ADMM is to find the optimal value $p^* = \inf\{f(x)+g(z)|Ax+Bz=c\}$. Then, we can form the augmented Lagrangian as $\mathcal{L}_\rho(x,z,y) = f(x)+g(z)+y^T(Ax+Bz-c)+(\rho/2)\|Ax+Bz-c\|_2^2$, where y is the Lagrange multiplier, and $\rho > 0$ is the penalty parameter.

We assume that $k \in \{1, 2, \ldots, K\}$ iterations are required to find the optimal value, and the updates of the iterations are

$$\begin{aligned} x^{k+1} &:= \arg\min \mathcal{L}_\rho\left(x, z^k, y^k\right) \\ z^{k+1} &:= \arg\min \mathcal{L}_\rho\left(x^{k+1}, z, y^k\right) \\ y^{k+1} &:= y^k + \rho\left(Ax^{k+1} + Bz^{k+1} - c\right). \end{aligned}$$

It has been proved that when the following two conditions are satisfied, the G-ADMM algorithm can converge: 1) the functions $f : \mathbb{R}^n \to \mathbb{R} \cup (+\infty)$ and $g : \mathbb{R}^m \to \mathbb{R} \cup (+\infty)$ are closed, proper, and convex; and 2) the augmented Lagrangian $\mathcal{L}_\rho(x,z,y)$ has a saddle point.

The basic G-ADMM algorithm is effective in solving 2-block problems (i.e., two separable functions with two independent variables). When we need to solve the problem with more than two separable functions, directly implementing G-ADMM cannot guarantee convergence. Therefore, He et al. [26] proposed a novel operator splitting method, termed MG-ADMM, which can be applied to multiblock problems. Take the 3-block separable minimization problem as an example to describe MG-ADMM. The form of a 3-block separable minimization problem is

$$\min\{f(x) + g(z) + h(y)|Ax + Bz + Ch = b\}.$$

Then the Lagrangian function is

$$\begin{aligned} \mathcal{L}_\rho(x,z,y,\lambda) = f(x) &+ g(z) + h(y) \\ &+ \lambda^T(Ax + Bz + Ch - b) \\ &+ \|Ax + Bz + Cy - b\|_2^2. \end{aligned}$$

The updates of iterations are

$$\begin{aligned} x^{k+1} &:= \arg\min\left\{\mathcal{L}_\rho^\beta\left(x, z^k, y^k, \lambda^k\right)\right\} \\ z^{k+1} &:= \arg\min\left\{\mathcal{L}_\rho^\beta\left(x^{k+1}, z, y^k, \lambda^k\right) + \frac{\rho}{2}\beta\left\|B\left(z - z^k\right)\right\|_2^2\right\} \\ y^{k+1} &:= \arg\min\left\{\mathcal{L}_\rho^\beta\left(x^{k+1}, z^k, y, \lambda^k\right) + \frac{\rho}{2}\beta\left\|C\left(y - y^k\right)\right\|_2^2\right\} \\ \lambda^{k+1} &:= \lambda^k - \beta\left(Ax^{k+1} + By^{k+1} + Cz^{k+1} - b\right) \end{aligned}$$

where $\beta \in (0, 1]$ is the penalty parameter.

2) MC-ADMM: In the beginning, we introduce C-ADMM as one of ADMM forms to solve the following problem $\arg\min_x \sum_{i=1}^N f_i(x)$, where $x \in \mathbb{R}^n$ and $f_i : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$ are assumed convex.

The basic idea of C-ADMM is dividing a large-scale optimization problem into N subproblems which can be solved in a distributed manner. For $\sum_{i=1}^N f_i(x)$, we can rewrite it as

$$\begin{aligned} &\arg\min_x \sum_{i=1}^N f_i(x_i) \\ &\text{s.t.} \;\; x_i - z = 0 \end{aligned}$$

where $z \in \mathbb{R}^n$ is an auxiliary variable or global variable.

The augmented Lagrangian is

$$\begin{aligned} \mathcal{L}(x_1, x_2, \ldots, x_n, z, y) = \sum_{i=1}^N \Bigg(& f_i(x_i) + y_i^T(x_i - z) \\ &+ \frac{\rho}{2}\|x_i - z\|_2^2\Bigg) \end{aligned}$$

where $(x_1, x_2, \ldots, x_n) \in \mathbb{R}^{nN}$.

The updates of parameters are as follows:

$$\begin{aligned} x_i^{k+1} &:= \arg\min\left\{\mathcal{L}(f_i(x_i), z^k y_i^k)\right\} \\ z^{k+1} &:= \frac{1}{N}\sum_{i=1}^N\left(x_i^{k+1} + \frac{1}{\rho}y_i^k\right) \\ y_i^{k+1} &:= y_i^k + \rho\left(x_i^{k+1} - z^{k+1}\right). \end{aligned}$$

Similar to the MG-ADMM built upon G-ADMM, MC-ADMM is based on C-ADMM by adding regularization terms to the Augmented Lagrangian formula and the variable iteration formulas. Therefore, we omit the detailed formulas of MC-ADMM for brevity.

AQ3

TABLE I
KEY NOTATIONS

Notation	Meaning
N	The total number of local devices
S	The edge server
D_i	The data size of the MEC task from local device i
D_{bcfl}	The data size of the BCFL task
T_i	The time limitation of the MEC task from device i
T_{bcfl}	The time requirement of the BCFL task
F	The maximum CPU cycle frequency of the edge server
B	The maximum available bandwidth of the edge server
α_i	The percentage of bandwidth allocated to device i
α_{bcfl}	The percentage of bandwidth allocated to the BCFL task
γ	The parameter correlated to the architecture of CPU
f_i	The CPU cycle frequency allocated to device i
f_{bcfl}	The CPU cycle frequency allocated to the BCFL task
r_i^{comm}	The data transmission rate between device i and edge server
r_{bcfl}^{comm}	The data transmission rate of the BCFL task
T_i^{comp}	The computing time of the MEC task from device i
T_i^{comm}	The transmission time between device i and the edge server
E_i^{comp}	The energy cost of computing the MEC task from device i
E_i^{comm}	The transmission cost between device i and the edge server
E_{total}^{comp}	The total energy cost of computing the MEC tasks
E_{total}^{comm}	The total transmission cost between devices and edge server

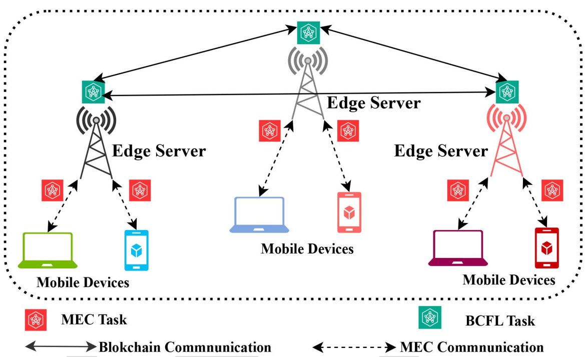


Fig. 1. Topology of BCFL in MEC.

III. SYSTEM MODEL AND PROBLEM FORMULATION

In this section, we discuss the system model from a general perspective and then explore both the communication and computing models of our proposed system. We also analyze the cost model for the formulation of the optimization problem toward resource allocation.

A. System Overview

The structure of our considered system is shown in Fig. 1, where the BCFL system consists of multiple edge servers with each server connecting multiple local mobile devices. In this work, we mainly focus on one server S with N local devices, denoted as $i \in \{1, \ldots, N\}$. Specifically, edge servers as BCFL nodes form the blockchain network to support FL. To protect the security and privacy of the BCFL system, we consider employing the consortium blockchain so that each BCFL node has to be authorized to participate and can thus be trusted. As for the consensus protocol, our proposed system can adopt any existing protocol that is applicable to the consortium blockchain, such as PBFT [27] and Raft [28].

In the MEC system, since mobile devices are usually resource-limited, they can choose to offload their computing tasks to their nearby edge server S. Then server S would prepare the necessary resources to help local devices finish their offloaded tasks. Therefore, edge servers will be responsible for not only providing offloading computing services to local devices but also running the BCFL system simultaneously, and both tasks consume considerable computing and communication resources.

The detailed workflow of our proposed system is below.

1) In the MEC system, local device i first transmits an offloading request $R_i(D_i, T_i)$ to server S, where D_i is the data size of its task and T_i is the corresponding time constraint for this task to be finished. Once server S accepts the tasks, local devices transmit their data to S.
2) As for the BCFL system, edge servers working as the clients of FL train the local models with their local data which may be generated by themselves or collected from other devices, and they also work as blockchain nodes to conduct consensus for generating new blocks that contain the local model updates and the updated global model of FL.

Generally, server S has limited computing capacity and communication bandwidth, denoted as F and B, respectively. Given that both MEC and BCFL tasks are usually time-sensitive, finishing the offloading tasks for lower layer mobile devices and maintaining the upper layer BCFL system without any delay require rigorous design for optimal resource allocation at edge servers.

B. Communication Models

In this section, we model the communication resource consumption for both the MEC and BCFL tasks.

1) MEC Task: The communications between device i and the server S include sending the offloading request, sending original data, and returning computing results. Since the sizes of the offloading request and computing results are much smaller than that of the data, we consider only the transmission of original data from devices to the server.

According to [29], the data transmission rate from local device i to edge server S is defined as

$$r_i^{\text{comm}}(\alpha_i) = \alpha_i B \log_2\left(1 + \frac{P_i G_i}{\delta^2}\right)$$

where $\alpha_i \in (0, 1)$ represents the percentage of bandwidth allocated to local devices i; B is the maximum bandwidth of server S; P_i and G_i are the transmission power and channel gain from i to S, respectively; and δ is the Gaussian noise during the transmission.

Then, we can calculate the time cost of data transmission from device i to server S as

$$T_i^{\text{comm}}(\alpha_i) = \frac{D_i}{r_i^{\text{comm}}(\alpha_i)}$$

which indicates that the transmission time cost is a function of the data size of the MEC task.

Also, the data transmission will cost a certain amount of energy, which can be calculated by

$$E_i^{\text{comm}}(\alpha_i) = P_i T_i^{\text{comm}}(\alpha_i)$$

and the total consumption of transmitting the data from all the local devices to the server is calculated as

$$E_{\text{total}}^{\text{comm}}(\alpha_i) = \sum_{i=1}^{N} E_i^{\text{comm}}(\alpha_i).$$

2) BCFL Task: The communications during the BCFL task are composed of sharing updates in the blockchain network and conducting blockchain consensus. For simplicity, here we treat the communication in BCFL as a combined process. Let α_{bcfl} denote the percentage of total bandwidth distributed to the BCFL task, and let P_{bcfl} and G_{bcfl} represent the transmission power and channel gain of the BCFL task, respectively. Then, we can calculate the data transmission rate in the BCFL task by

$$r_{\text{bcfl}}^{\text{comm}}(\alpha_{\text{bcfl}}) = \alpha_{\text{bcfl}} B \log_2\left(1 + \frac{P_{\text{bcfl}} G_{\text{bcfl}}}{\delta^2}\right).$$

The time cost of transmission in the BCFL task is

$$T_{\text{bcfl}}^{\text{comm}}(\alpha_{\text{bcfl}}) = \frac{\widehat{D}_{\text{bcfl}}}{r_{\text{bcfl}}^{\text{comm}}(\alpha_{\text{bcfl}})}$$

where $\widehat{D}_{\text{bcfl}}$ is the size of required transmission data in the BCFL task, which is smaller than the size of the training and mining data for the BCFL task, denoted as D_{bcfl}, at server S. The energy consumption of the server for conducting the BCFL task can be calculated as

$$E_{\text{bcfl}}^{\text{comm}}(\alpha_{\text{bcfl}}) = P_{\text{bcfl}} T_{\text{bcfl}}^{\text{comm}}(\alpha_{\text{bcfl}}).$$

C. Computing Models

In this part, we describe the time and energy consumed by the MEC server to process the MEC and BCFL tasks.

1) MEC Task: Let $f_i \in (0, F)$ be the CPU cycle frequency allocated to the task of device i. First, we define the total CPU cycles used for the task of device i as μ_i, and it can be calculated as $\mu_i = D_i d_i$ with d_i denoting the unit CPU cycle frequency required to process one data sample of the MEC task from device i. Then, the computing time can be calculated by

$$T_i^{\text{comp}}(f_i) = \frac{\mu_i}{f_i}.$$

According to [30], the energy cost of computing one single task of device i is

$$E_i^{\text{comp}}(f_i) = \gamma \mu_i f_i^2$$

where γ is the parameter correlated to the architecture of the CPU. Thus, the total energy consumption of computing the MEC tasks for all devices is calculated by

$$E_{\text{total}}^{\text{comp}}(f_i) = \sum_{i=1}^{N} E_i^{\text{comp}}(f_i).$$

2) BCFL Task: Similarly, we define $f_{\text{bcfl}} \in (0, F)$ as the CPU cycle frequency allocated to the BCFL task. Let $\mu_{\text{bcfl}} = D_{\text{bcfl}} d_{\text{bcfl}}$ denote the total CPU cycles for processing the BCFL task, where d_{bcfl} means the unit CPU cycle used to process one BCFL data sample. Then, we can calculate the time cost of computing the BCFL task $T_{\text{bcfl}}^{\text{comp}}(f_{\text{bcfl}}) = (\mu_{\text{bcfl}}/f_{\text{bcfl}})$. In this way, the energy cost of computing the BCFL task is calculated as

$$E_{\text{bcfl}}^{\text{comp}}(f_{\text{bcfl}}) = \gamma \mu_{\text{bcfl}} f_{\text{bcfl}}^2.$$

D. Cost Model

We have discussed the energy consumed by the communication and computation of the MEC and the BCFL tasks. Now we can define the cost model of our proposed resource allocation scheme. Denoting the total energy cost as U, based on the above models, we know that U is composed of the transmission cost and the computing cost. Then, we have

$$\begin{aligned} U(\alpha_i, \alpha_{\text{bcfl}}, f_i, f_{\text{bcfl}}) = {} & E_{\text{total}}^{\text{comm}}(\alpha_i) + E_{\text{bcfl}}^{\text{comm}}(\alpha_{\text{bcfl}}) \\ & + E_{\text{total}}^{\text{comp}}(f_i) + E_{\text{bcfl}}^{\text{comp}}(f_{\text{bcfl}}). \end{aligned} \tag{1}$$

E. Problem Formulation

The purpose of our resource allocation mechanism is to allow the edge server to handle both the MEC and BCFL tasks by satisfying resource and time constraints with the minimum cost. The edge server should make the decisions about how many CPU cycles and how much bandwidth should be allocated to each task. Technically, the optimal resource allocation decisions need to consider minimizing the total energy consumption of the edge server. Thus, we can formulate the decision-making challenge of resource allocation into an optimization problem as follows:

$$\begin{aligned} \mathbf{P1}: \ & \underset{\alpha_i, \alpha_{\text{bcfl}}, f_i, f_{\text{bcfl}}}{\arg\min} \ : U(\alpha_i, \alpha_{\text{bcfl}}, f_i, f_{\text{bcfl}}) \\ \text{s.t.:} & \mathbf{C1} \ : T_{\text{bcfl}}^{\text{comm}} + T_{\text{bcfl}}^{\text{comp}} \le T_{\text{bcfl}} \\ & \mathbf{C2} \ : T_i^{\text{comm}} + T_i^{\text{comp}} \le T_i \\ & \mathbf{C3} \ : \alpha_{\text{bcfl}} + \sum_{i=1}^{N} \alpha_i \le 1 \\ & \mathbf{C4} \ : f_{\text{bcfl}} + \sum_{i=1}^{N} f_i \le F \\ & \mathbf{C5} \ : D_{\text{bcfl}} + \widehat{D}_{\text{bcfl}} + \sum_{i=1}^{N} D_i \le D \\ & \mathbf{C6} \ : f_i, f_{\text{bcfl}} \in (0, F), \alpha_i, \alpha_{\text{bcfl}} \in (0, 1) \\ & \qquad i \in \{1, 2, \ldots, N\} \end{aligned}$$

where *C1* and *C2* guarantee that the server can finish the BCFL task and MEC task on time; *C3* and *C4* ensure that the communication and computing resources allocated to each task do not exceed the maximum capacities of the server; *C5* means that the total data size of all the tasks running on the server cannot exceed its maximum storage capacity, denoted as D; *C6* clarifies the ranges of all variables. By analyzing the above optimization problem, we have the following theorem.

Theorem 1: Given that the variables $\alpha_i, \alpha_{\text{bcfl}}, f_i, f_{\text{bcfl}}$ are positive, the optimization objective function $U(\alpha_i, \alpha_{\text{bcfl}}, f_i, f_{\text{bcfl}})$ is convex.

The detailed proof of Theorem 1 is in Appendix A. However, it is still hard to solve *P1* even though the objective function is convex due to the following reasons: 1) there are multiple variables required to be optimized, and they are not fully correlated since they can be separated and 2) there are multiple constraints, making it harder to find the optimal solutions. In addition, since the MEC tasks have different data sizes and time requirements in the homogeneous and heterogeneous cases, we need to adapt *P1* to different cases to solve them separately. We will present our problem reformulations and solutions in the next two sections.

IV. MG-ADMM Solution in the Homogeneous Situation

In this section, we design the resource allocation mechanism in the homogeneous case, where all MEC tasks have the same data size and time requirements. We thus form a simple version of *P1*, where an equal distribution strategy is considered to allocate resources to all local devices, including both the bandwidth and CPU frequencies. The equal distribution strategy means that the edge server distributes the same communication and computing resources to each local device, that is, α_i and f_i are the same for any arbitrary device i. We will solve *P1* with the equal distribution strategy based on the modified G-ADMM method, which is derived from the basic form of ADMM.

A. Problem Reformulation Based on MG-ADMM

In the homogeneous scenario, the edge server distributes the same amount of resources, denoted as α^* and f^*, to each local device. The energy cost of computing is the sum of all devices' costs, and thus can be expressed as $E_{\text{total}}^{\text{comp}}(f^*) = \sum_{i=1}^{N} \gamma \mu_i f_i^2 = N\gamma\mu_i f^{*2}$. Moreover, the communication cost of the MEC tasks can be calculated as $E_{\text{total}}^{\text{comm}}(\alpha^*) = \sum_{i=1}^{N} P_i T_i^{\text{comm}} = NP_i(D_i/[\alpha^* B \log_2(1 + (P_i G_i/\delta^2))])$. Thus, we can rewrite U as

$$\begin{aligned} U'\left(\alpha^*, \alpha_{\text{bcfl}}, f^*, f_{\text{bcfl}}\right) = &\, E_{\text{total}}^{\text{comm}}\left(\alpha^*\right) + E_{\text{bcfl}}^{\text{comm}}(\alpha_{\text{bcfl}}) \\ &+ E_{\text{total}}^{\text{comp}}\left(f^*\right) + E_{\text{bcfl}}^{\text{comp}}(f_{\text{bcfl}}). \end{aligned} \tag{2}$$

Besides, the offloading time costs of communication and computing are $\widehat{T}_i^{\text{comp}}(f^*) = (\mu_i/f^*)$ and $\widehat{T}_i^{\text{comm}}(\alpha^*) = (D_i/[\alpha^* B \log_2(1 + (P_i G_i/\delta^2))])$, respectively. Based on the above analysis, in the case of a homogeneous situation, we need to determine four variables, i.e., $\alpha^*, \alpha_{\text{bcfl}}, f^*$, and f_{bcfl}. We can easily prove that U' is convex based on Theorem 1. Therefore, we apply MG-ADMM to optimize U' and derive the optimal variables. In this way, we can reformulate *P1* as follows:

$$\begin{aligned} \textbf{P2:} \quad \underset{\alpha^*, \alpha_{\text{bcfl}}, f^*, f_{\text{bcfl}}}{\arg\min} \; &: U'\left(\alpha^*, \alpha_{\text{bcfl}}, f^*, f_{\text{bcfl}}\right) \\ \text{s.t.}: \; & \textbf{C1}, \textbf{C5} \text{ in } \textbf{P1} \\ & \textbf{C2}: \widehat{T}_i^{\text{comm}} + \widehat{T}_i^{\text{comp}} \le T_i \\ & \textbf{C3}: \alpha_{\text{bcfl}} + N\alpha^* \le 1 \\ & \textbf{C4}: f_{\text{bcfl}} + Nf^* \le F \\ & \textbf{C6}: f^*, f_{\text{bcfl}} \in (0, F), \alpha^*, \alpha_{\text{bcfl}} \in (0, 1) \\ & \quad i \in \{1, 2, \ldots, N\}. \end{aligned}$$

B. Solution Based on MG-ADMM

First, we form the augmented Lagrangian of *P2* as follows:

$$\begin{aligned} \mathcal{L}_1 &= \mathcal{L}\left(\alpha^*, \alpha_{\text{bcfl}}, f^*, f_{\text{bcfl}}, \lambda_1, \lambda_2, \lambda_3, \lambda_4, \lambda_5\right) \\ &= U' + \lambda_1\left(T_{\text{bcfl}}^{\text{comm}} + T_{\text{bcfl}}^{\text{comp}} - T_{\text{bcfl}}\right) \\ &\quad + \lambda_2\left(\widehat{T}_i^{\text{comm}} + \widehat{T}_i^{\text{comp}} - T_i\right) \\ &\quad + \lambda_3\left(\alpha_{\text{bcfl}} + N\alpha^* - 1\right) \\ &\quad + \lambda_4\left(f_{\text{bcfl}} + Nf^* - F\right) \\ &\quad + \lambda_5\left(D_{\text{bcfl}} + \widehat{D}_{\text{bcfl}} + \sum_{i}^{N} D_i - D\right) \\ &\quad + \frac{\rho}{2}\left\|T_{\text{bcfl}}^{\text{comm}} + T_{\text{bcfl}}^{\text{comp}} - T_{\text{bcfl}}\right\|_2^2 \\ &\quad + \frac{\rho}{2}\left\|\widehat{T}_i^{\text{comm}} + \widehat{T}_i^{\text{comp}} - T_i\right\|_2^2 \\ &\quad + \frac{\rho}{2}\left\|\alpha_{\text{bcfl}} + N\alpha^* - 1\right\|_2^2 \\ &\quad + \frac{\rho}{2}\left\|f_{\text{bcfl}} + Nf^* - 1\right\|_2^2 \\ &\quad + \frac{\rho}{2}\left\|D_{\text{bcfl}} + \widehat{D}_{\text{bcfl}} + \sum_{i}^{N} D_i - D\right\|_2^2 \end{aligned}$$

where $\lambda_m > 0$ with $m \in \{1, 2, 3, 4, 5\}$ is the augmented Lagrange multiplier, and $\rho > 0$ is the penalty parameter.

Theorem 2: Given that the variables $\alpha^*, \alpha_{\text{bcfl}}, f^*, f_{\text{bcfl}}, \lambda_1, \lambda_2, \lambda_3, \lambda_4, \lambda_5$ are positive, the augmented Lagrangian of *P2*, i.e., $\mathcal{L}_1$, has a saddle point.

Please see Appendix B for proofs. Let $k \in \{1, 2, \ldots, K\}$ be the iteration index, and the updates of variables can be expressed as

$$\alpha^{*k+1} := \arg\min \mathcal{L}\left(\alpha^*, \alpha_{\text{bcfl}}^k, f^{*k}, f_{\text{bcfl}}^k, \lambda_1^k, \lambda_2^k, \lambda_3^k, \lambda_4^k, \lambda_5^k\right) \tag{3}$$

$$\begin{aligned} \alpha_{\text{bcfl}}^{k+1} := \arg\min \Big(&\mathcal{L}(\alpha^{*k+1}, \alpha_{\text{bcfl}}, f^{*k}, f_{\text{bcfl}}^k, \\ &\lambda_1^k, \lambda_2^k, \lambda_3^k, \lambda_4^k, \lambda_5^k) + \frac{\rho}{2}\beta\|\alpha_{\text{bcfl}} - \alpha_{\text{bcfl}}^k\|_2^2\Big) \end{aligned} \tag{4}$$

$$\begin{aligned} f^{*k+1} := \arg\min \Big(&\mathcal{L}\Big(\alpha^{*k+1}, \alpha_{\text{bcfl}}^k, f^*, f_{\text{bcfl}}^k, \\ &\lambda_1^k, \lambda_2^k, \lambda_3^k, \lambda_4^k, \lambda_5^k\Big) + \frac{\rho}{2}\beta\|N(f^* - f^{*k})\|_2^2\Big) \end{aligned} \tag{5}$$

$$\begin{aligned} f_{\text{bcfl}}^{k+1} := \arg\min \Big(&\mathcal{L}\Big(\alpha^{*k+1}, \alpha_{\text{bcfl}}^k, f^{*k}, f_{\text{bcfl}} \\ &\lambda_1^k, \lambda_2^k, \lambda_3^k, \lambda_4^k, \lambda_5^k\Big) + \frac{\rho}{2}\beta\|f_{\text{bcfl}} - f_{\text{bcfl}}^k\|_2^2\Big) \end{aligned} \tag{6}$$

where $\beta \in (0, 1]$ is the penalty parameter.

The updates of augmented Lagrange multipliers are

$$\lambda_1^{k+1} := \lambda_1^k - \beta(T_{\text{bcfl}}^{\text{comm}}(\alpha_{\text{bcfl}}^{k+1}) + T_{\text{bcfl}}^{\text{comp}}(f_{\text{bcfl}}^{k+1}) - T_{\text{bcfl}}) \tag{7}$$

$$\lambda_2^{k+1} := \lambda_2^k - \beta(\widehat{T}_i^{\text{comm}}(\alpha^{*k+1}) + \widehat{T}_i^{\text{comp}}(f^{*k+1}) - T_i) \tag{8}$$

$$\lambda_3^{k+1} := \lambda_3^k - \beta(\alpha_{\text{bcfl}} + N\alpha^* - 1) \tag{9}$$

Algorithm 1 Solution of *P2* Based on MG-ADMM Algorithm

Require: P_i, D_i, N, G_i, G_{bcfl}, δ, ψ, G_{bcfl}, F, γ, d_{bcfl}, ρ, P_{bcfl}, D_{bcfl}, k, T_i, T_{bcfl}, β, λ_1, λ_2, λ_3, λ_4, λ_5

Ensure: α^*, α_{bcfl}, f^*, f_{bcfl}, U'

1: Initialize α^*, α_{bcfl}, f^*, f_{bcfl}, λ_1, λ_2, λ_3, λ_4, λ_5
2: **while** Convergence $\neq$ True **do**
3: $\quad\alpha^{*k+1}, \alpha_{\text{bcfl}}^{k+1}, f^{*k+1}, f_{\text{bcfl}}^{k+1} \leftarrow$ optimal values of (3)-(6)
4: $\quad\lambda_1^{k+1}, \lambda_2^{k+1}, \lambda_3^{k+1}, \lambda_4^{k+1}, \lambda_5^{k+1} \leftarrow$ update (7)-(11)
5: $\quad$Calculate (12) and (13)
6: $\quad$**if** (12) and (13) are satisfied **then**
7: $\quad\quad$Convergence $=$ True
8: $\quad$**end if**
9: $\quad k \leftarrow k+1$
10: **end while**
11: Calculate U' via (2)
12: **return** α^*, α_{bcfl}, f^*, f_{bcfl}, U'

$$\lambda_4^{k+1} := \lambda_4^k - \beta(f_{\text{bcfl}} + Nf^* - F) \tag{10}$$

$$\lambda_5^{k+1} := \lambda_5^k - \beta(D_{\text{bcfl}} + \widehat{D}_{\text{bcfl}} + \sum_{i=1}^{N} D_i - D). \tag{11}$$

Then, we can set the stopping criteria for the above iterations

$$\left\|\alpha^{*k+1} - \alpha^{*k}\right\|_2^2 \leq \psi, \left\|f^{*k+1} - f^{*k}\right\|_2^2 \leq \psi \tag{12}$$

$$\left\|\alpha_{\text{bcfl}}^{k+1} - \alpha_{\text{bcfl}}^{k}\right\|_2^2 \leq \psi, \left\|\alpha_{\text{bcfl}}^{k+1} - \alpha_{\text{bcfl}}^{k}\right\|_2^2 \leq \psi \tag{13}$$

where ψ is the predefined threshold [31].

Note that (3) to (6) are quadratic optimization problems and can be solved efficiently. Due to the space limit, we omit the detailed calculations.

It has been proved that when the following two conditions are satisfied, the MG-ADMM algorithm can converge: 1) the objective function is closed, proper, and convex and 2) the augmented Lagrangian has a saddle point. We have proved that the objective function is convex, and it is also closed and proper. Besides, we have proved that $\mathcal{L}_1$ has a saddle point in Theorem 2. Thus, the convergence of *P2* is guaranteed.

We summarize our proposed solution based on MG-ADMM in Algorithm 1. First, we initialize four variables and five augmented Lagrangian multipliers (line 1), and then we update the variables and Lagrange multipliers in an iterative process (lines 2–10). Specifically, we update variables and Lagrange multipliers (lines 3–4) and calculate the stopping criteria (line 5). If the termination condition is satisfied, then the objective function is converged (lines 6–8). In the end, we calculate the optimal value of the objective function, and then all the optimal decisions and the optimal total energy cost are returned (lines 11–12). The time complexity of this algorithm is $\mathcal{O}(K)$, which indicates that we can solve *P2* with a linear time complexity.

V. MC-ADMM Solution in the Heterogeneous Scenario

In this section, we consider the heterogeneous scenario with diverse MEC requests from local devices. To this end, we need to apply an on-demand resource allocation strategy. That is, to say, we have to determine the resource allocation decisions for each MEC task, which is more realistic compared to the equal distribution strategy in the homogeneous scenario. Specifically, we calculate α_i and f_i for $i \in \{1, 2, \ldots, N\}$, as well as α_{bcfl} and f_{bcfl}. Thus, the optimization problem in this scenario is more practical and complicated.

A. Problem Reformulation Based on MC-ADMM

In the heterogeneous scenario, we have to distribute resources to each MEC task and the BCFL task, so there are $2N+2$ variables in total. Directly applying the previous MG-ADMM algorithm, in this case, is not practical since the resource distribution in the heterogeneous situation is much more complicated than the optimization in the homogeneous scenario. Besides, the convergence for $2N+2$ variables in the MG-ADMM algorithm is not guaranteed. Therefore, we resort to the MC-ADMM algorithm, which can solve the large-scale optimization problem in a distributed way.

Intuitively, allocating the resources to each device is to divide the bandwidth and CPU cycle frequency into $N+1$ parts to find the best decision separately. To calculate α_i and f_i for each $i \in \{1, \ldots, N\}$, we first define $\hat{\alpha}$ and $\hat{f}$ as global variables, also called auxiliary variables, to assist the distributed optimization. Besides, we have to consider the constraints of *P1*. For simplicity, we denote the space formed by the constraints related to α_i and f_i (i.e., *C2-C4* of *P1*) as Ω, which is the feasible set of local variables α_i and f_i. While the other constraints not related to α_i and f_i in *P1* need to be kept unchanged because they will influence the rest two variables, i.e., α_{bcfl} and f_{bcfl}. Then we can have the reformulated problem as

$$\begin{aligned}
\textbf{P3:}\ \underset{\alpha_i, \alpha_{\text{bcfl}}, f_i, f_{\text{bcfl}}}{\arg\min}\ & :U(\alpha_i, \alpha_{\text{bcfl}}, f_i, f_{\text{bcfl}}) \\
\text{s.t.:}\mathbf{C1}\ & :\alpha_i = \hat{\alpha}, \\
\mathbf{C2}\ & :f_i = \hat{f}, \\
\mathbf{C3}\ & :T_{\text{bcfl}}^{\text{comm}} + T_{\text{bcfl}}^{\text{comp}} \leq T_{\text{bcfl}}, \\
\mathbf{C4}\ & :D_{\text{bcfl}} + \widehat{D}_{\text{bcfl}} + \sum_{i=1}^{N} D_i \leq D, \\
\mathbf{C5}\ & :(\alpha_i, f_i) \in \Omega, \alpha_{\text{bcfl}}, \hat{\alpha} \in (0, 1), \\
& \quad f_{\text{bcfl}}, \hat{f} \in (0, F), i \in \{1, 2, \ldots, N\}.
\end{aligned}$$

B. Solution Based on MC-ADMM

Here, we detail the solution based on MC-ADMM. First, the augmented Lagrangian form of *P3* is

$$\begin{aligned}
\mathcal{L}_2 &= \mathcal{L}\Big(\alpha_i, \alpha_{\text{bcfl}}, f_i, f_{\text{bcfl}}, \theta_i, \epsilon_i, \eta_1, \eta_2, \hat{\alpha}, \hat{f}\Big) \\
&= U + \sum_{i=1}^{N} \theta_i(\alpha_i - \hat{\alpha}) + \sum_{i=1}^{N} \epsilon_i\Big(f_i - \hat{f}\Big) \\
&\quad + \eta_1\big(T_{\text{bcfl}}^{\text{comm}} + T_{\text{bcfl}}^{\text{comp}} - T_{\text{bcfl}}\big) \\
&\quad + \eta_2\left(D_{\text{bcfl}} + \widehat{D}_{\text{bcfl}} + \sum_{i=1}^{N} D_i - D\right) + \frac{\rho}{2}\left\|\alpha_i - \hat{\alpha}\right\|_2^2
\end{aligned}$$

$$+\frac{\rho}{2}\left\|f_i-\hat{f}\right\|_2^2+\frac{\rho}{2}\left\|T_{\mathrm{bcfl}}^{\mathrm{comm}}+T_{\mathrm{bcfl}}^{\mathrm{comp}}-T_{\mathrm{bcfl}}\right\|_2^2$$
$$+\left\|D_{\mathrm{bcfl}}+\widehat{D}_{\mathrm{bcfl}}+\sum_{i=1}^{N}D_i-D\right\|_2^2$$

where $\theta_i, \epsilon_i, \eta_1, \eta_2 > 0$ are augmented Lagrange multipliers.

Theorem 3: Given that the variables $\alpha_i, \alpha_{\mathrm{bcfl}}, f_i, f_{\mathrm{bcfl}}, \theta_i, \epsilon_i, \eta_1, \eta_2, \hat{\alpha}, \hat{f}$ are positive, the augmented Lagrangian of *P3*, i.e., $\mathcal{L}_2$, has a saddle point.

The proofs are in Appendix C. By applying the method proposed in [26], the updates of local variables (i.e., α_i and f_i) are

$$\left\{\alpha_i^{k+1}, f_i^{k+1}\right\} := \arg\min \mathcal{L}\Big(\alpha_i, \alpha_{\mathrm{bcfl}}^k, f_i, f_{\mathrm{bcfl}}^k, \theta_i^k, \epsilon_i^k, \eta_1^k, \eta_2^k, \widehat{\alpha}^k, \widehat{f}^k\Big). \tag{14}$$

The updates of α_{bcfl} and f_{bcfl} are

$$\alpha_{\mathrm{bcfl}}^{k+1} := \arg\min\bigg(\mathcal{L}\Big(\alpha_i^{k+1}, \alpha_{\mathrm{bcfl}}, f_i^{k+1}, f_{\mathrm{bcfl}}^{k+1}, \theta_i^k, \epsilon_i^k, \eta_1^k, \eta_2^k, \widehat{\alpha}^k, \widehat{f}^k\Big) + \frac{\rho}{2}\beta\left\|\alpha_{\mathrm{bcfl}}-\alpha_{\mathrm{bcfl}}^k\right\|_2^2\bigg) \tag{15}$$

$$f_{\mathrm{bcfl}}^{k+1} := \arg\min\bigg(\mathcal{L}\Big(\alpha_i^{k+1}, \alpha_{\mathrm{bcfl}}^{k+1}, f_i^{k+1}, f_{\mathrm{bcfl}}, \theta_i^k, \epsilon_i^k, \eta_1^k, \eta_2^k, \widehat{\alpha}^k, \widehat{f}^k\Big) + \frac{\rho}{2}\beta\left\|f_{\mathrm{bcfl}}-f_{\mathrm{bcfl}}^k\right\|_2^2\bigg) \tag{16}$$

where $\rho, \beta \in (0, 1]$ are penalty parameters.

The updates of global variables are

$$\widehat{\alpha}^{k+1} := \frac{1}{N}\sum_{i=1}^{N}\Big(\alpha_i^{k+1}+\frac{\rho}{2}\theta_i^k\Big) \tag{17}$$

$$\widehat{f}^{k+1} := \frac{1}{N}\sum_{i=1}^{N}\Big(f_i^{k+1}+\frac{\rho}{2}\epsilon_i^k\Big). \tag{18}$$

Besides, the updates of augmented Lagrange multipliers are

$$\theta_i^{k+1} := \theta_i^k + \rho\Big(\alpha_i^{k+1}-\widehat{\alpha}^{k+1}\Big) \tag{19}$$

$$\epsilon_i^{k+1} := \epsilon_i^k + \rho\Big(f_i^{k+1}-\widehat{f}^{k+1}\Big) \tag{20}$$

$$\eta_1^{k+1} := \eta_1^k - \beta\Big(T_{\mathrm{bcfl}}^{\mathrm{comm}}(\alpha_{\mathrm{bcfl}}^{k+1}) + T_{\mathrm{bcfl}}^{\mathrm{comp}}(f_{\mathrm{bcfl}}^{k+1}) - T_{\mathrm{bcfl}}\Big) \tag{21}$$

$$\eta_2^{k+1} := \eta_2^k - \beta\left(D_{\mathrm{bcfl}}+\widehat{D}_{\mathrm{bcfl}}+\sum_{i=1}^{N}D_i-D\right). \tag{22}$$

Lastly, the stopping criteria can be set as

$$\left\|\alpha_i^{k+1}-\widehat{\alpha}^{k+1}\right\|_2^2 \le \psi_{\mathrm{prim}}, \left\|f_i^{k+1}-\widehat{f}^{k+1}\right\|_2^2 \le \psi_{\mathrm{prim}} \tag{23}$$

$$\left\|\widehat{\alpha}^{k+1}-\widehat{\alpha}^{k}\right\|_2^2 \le \psi_{\mathrm{dual}}, \left\|\widehat{f}^{k+1}-\widehat{f}^{k}\right\|_2^2 \le \psi_{\mathrm{dual}} \tag{24}$$

where ψ_{prim} and ψ_{dual} are the predefined thresholds [31]. Besides, (13) should also be included as a stopping criteria.

Algorithm 2 Solution of *P3* Based on MC-ADMM Algorithm

Require: P_i, D_i, N, G_i, G_{bcfl}, δ, ψ, G_{bcfl}, F, γ, d_{bcfl}, ρ, P_{bcfl}, D_{bcfl}, k, T_i, T_{bcfl}, β, θ_i, ϵ_i, η_1, η_2

Ensure: α_i, α_{bcfl}, f_i, f_{bcfl}, U

1: Initialize α_i, α_{bcfl}, f_i, f_{bcfl}, θ_i, ϵ_i, η_1, η_2, $\hat{\alpha}$, $\hat{f}$
2: **for** $i \in \{1, 2, \ldots, N\}$ **do**
3:   **while** Convergence $\neq$ True **do**
4:     $\alpha_i^{k+1}, f_i^{k+1} \leftarrow$ optimal values of (14)
5:     $\alpha_{\mathrm{bcfl}}^{k+1}, f_{\mathrm{bcfl}}^{k+1}, \widehat{\alpha}^{k+1}, \widehat{f}^{k+1} \leftarrow$ optimal values of (15)-(18)
6:     $\theta_i^{k+1}, \epsilon_i^{k+1}, \eta_1^{k+1}, \eta_2^{k+1} \leftarrow$ update (19)-(22)
7:     Calculate (13), (23) and (24)
8:     **if** (13), (23) and (24) are satisfied **then**
9:       Convergence = True
10:     **end if**
11:     $k \leftarrow k+1$
12:   **end while**
13: **end for**
14: Calculate U via (1)
15: **return** α_i, α_{bcfl}, f_i, f_{bcfl}, U

Even though the forms of *P2* and *P3* are different, the proof of the convergence is similar. According to Theorem 1, we know that U is convex, and it is clear that U is closed and proper. In addition, the augmented Lagrangian $\mathcal{L}_2$ has a saddle point. So the convergence of *P3* is guaranteed with the MC-ADMM algorithm.

For reference, we generalize the solution based on MC-ADMM in Algorithm 2. We first initialize local variables, global variables, and augmented Lagrangian multipliers (Line 1), and then we calculate the optimal decisions for each MEC task (Lines 2-13). In detail, we keep updating parameters until the objective function is converged (Lines 3-12). Then, we can calculate the optimal total energy cost and return the optimal decisions (Lines 14-15). The time complexity is determined by the operations of $2N+2$ tasks and K rounds of iteration for each task, which is $\mathcal{O}(KN)$.

VI. Experimental Evaluation

In this section, we conduct experiments to test the validity and efficiency of our proposed algorithms. We first provide the parameter setting for experiments, then we present and analyze the experimental results. We conduct the experiments using Python 3.8.5 in macOS 11.6 running on an Intel i7 processor with 32 GB RAM and 1 TB SSD.

A. Basic Experimental Setting

We consider a MEC scenario with one edge server and 10 local devices. For brevity, we provide Table II to detail the basic parameter settings in our experiments. As for the settings of certain experiments, we will clarify them later. For the augmented Lagrange multipliers, we set them as 1.0 at the beginning.

TABLE II
BASIC EXPERIMENTAL SETTING

$N = 10$	$\beta = 0.5$	$G_i = 10$	$d_i = 2$	$D_i = 10$
$k = 100$	$P_i = 2$	$G_{bcfl} = 10$	$d_{bcfl} = 2$	$D_{bcfl} = 10$
$\rho = 0.5$	$P_{bcfl} = 2$	$F = 1000$	$\delta = 0.1$	$\gamma = 0.001$
$T_i = 10$	$T_{bcfl} = 50$	$\psi = 10^{-3}$	$\psi_{prim} = 10^{-3}$	$\psi_{dual} = 10^{-3}$

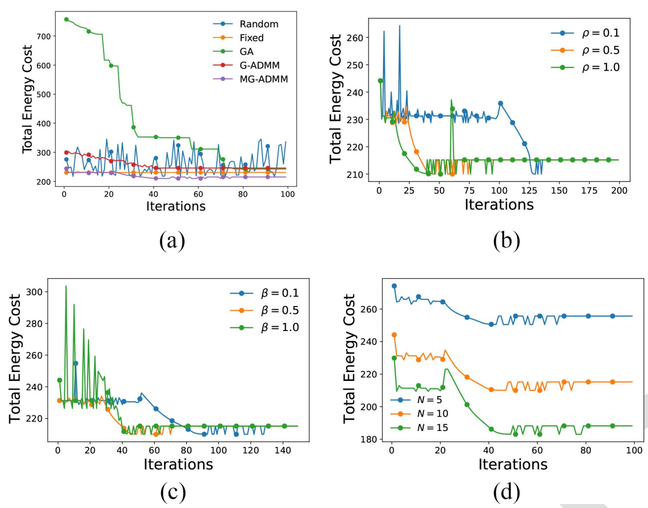


Fig. 2. Convergence of the MG-ADMM algorithm. (a) Scheme comparison. (b) Penalty parameter ρ. (c) Penalty parameter β. (d) Number of devices N.

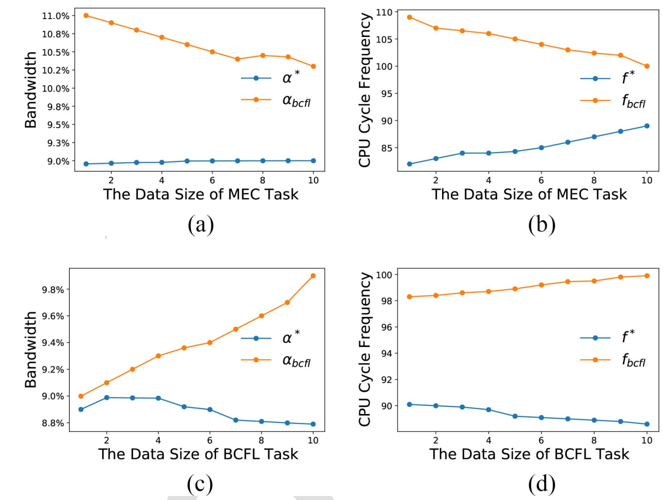


Fig. 3. Optimal resource allocation decisions based on the MG-ADMM algorithm. (a) Bandwidth. (b) CPU cycle frequencies. (c) Bandwidth. (d) CPU cycle frequencies.

B. Experimental Results

We design two parts of the experiments: 1) the evaluation of the MG-ADMM algorithm and 2) the evaluation of the MC-ADMM algorithm. These two algorithms are designed for different scenarios, i.e., homogeneous and heterogeneous. In the homogeneous scenario, we assume that all the parameters of each MEC task are the same, while in the heterogeneous scenario, we treat each MEC task individually. Due to the limitation of space, we only present partial experimental results with importance in this section.

1) Evaluation of the MG-ADMM Algorithm: We first evaluate the MG-ADMM algorithm solving *P2* in the homogeneous scenario, and then we analyze the impacts of the data sizes of both the MEC and the BCFL tasks on the optimal decisions in our resource allocation scheme.

For comparison, we design a random allocation strategy, which assigns the bandwidth and CPU cycle frequencies to the MEC and the BCFL tasks in a random way. We also consider a fixed allocation strategy, which determines the resource allocation with fixed values at the beginning. Besides, we use the G-ADMM algorithm by setting α_{bcfl} and f_{bcfl} to fixed values as another benchmark solution since setting other variables as constants cannot return converged results. Furthermore, we provide a generic algorithm-based method (GA) [32] as the baseline. In our experiment, the genetic algorithm was configured with key parameters: a maximum of 100 iterations, a population size of 100, a mutation probability of 0.1, and an elitism ratio of 0.01. Via comparing the proposed MG-ADMM algorithm with these four solutions, we plot the experimental results in Fig. 2(a). We can see that the MG-ADMM algorithm can converge after about 80 rounds of iteration, while the random strategy cannot converge. In addition, the random and fixed strategies, as well as G-ADMM, inevitably incur a total energy cost larger than that of the MG-ADMM algorithm. As for the GA method, it can be iterated to decrease the energy cost, but the converged result is still poor compared to that of the MG-ADMM method. The results show that the MG-ADMM algorithm outperforms the other strategies.

As the penalty parameters ρ and β will influence the convergence speed of the MG-ADMM algorithm, we set the values of ρ as $\{0.1, 0.5, 1.0\}$, and maintain other parameters unchanged. The results in Fig. 2(b) show that the faster convergence speed as a result of a larger ρ. Similarly, we can see from Fig. 2(c) that the convergence speed will be faster when β is larger. The reason is that the penalty parameters control the length of the step in each iteration and larger penalty parameters will lead to a greater length of each step, so the convergence speed will be faster.

To testify the impact of the number of local devices on the convergence of MG-ADMM, we plot experimental results in Fig. 2(d). We can see that the convergence speed will be slower and the optimal value will be larger when the number of local devices increases, which indicates that it will influence not only the convergence speed but also the optimal value of the total energy cost. This is because with more devices involved in the MEC tasks, the edge server will cost more energy to work for the tasks, and the optimization problem will be more difficult, so more time will be cost to converge.

In the homogeneous scenario, both the bandwidth and CPU cycle frequencies assigned to each local device are the same, so we only need to calculate four variables, i.e., $\alpha^*, \alpha_{\text{bcfl}}, f^*, f_{\text{bcfl}}$ for the optimal allocation decisions. In Fig. 3, for different data sizes of the MEC tasks (D_i) and the BCFL task (D_{bcfl}), the results show that the data sizes of tasks significantly influence the resource allocation decisions. In Fig. 3(a) and (b), it can be seen that the larger the data size of each MEC task, the more communication and computing resources allocated to devices and the fewer resources allocated to the

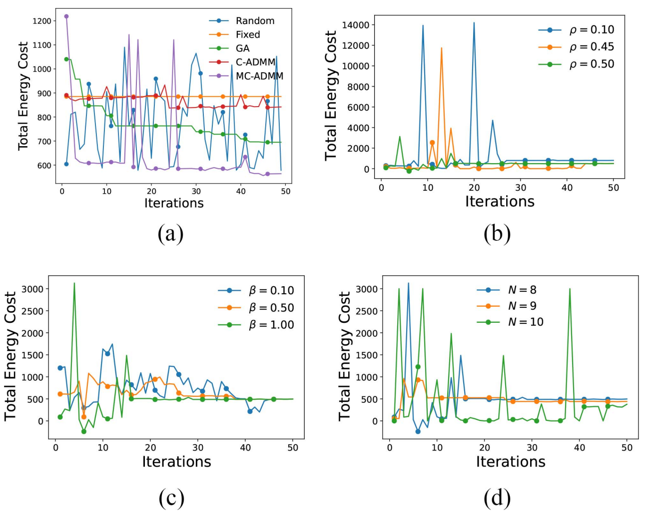


Fig. 4. Convergence of the MC-ADMM algorithm. (a) Strategies comparison. (b) Penalty parameter ρ. (c) Penalty parameter β. (d) Number of devices N.

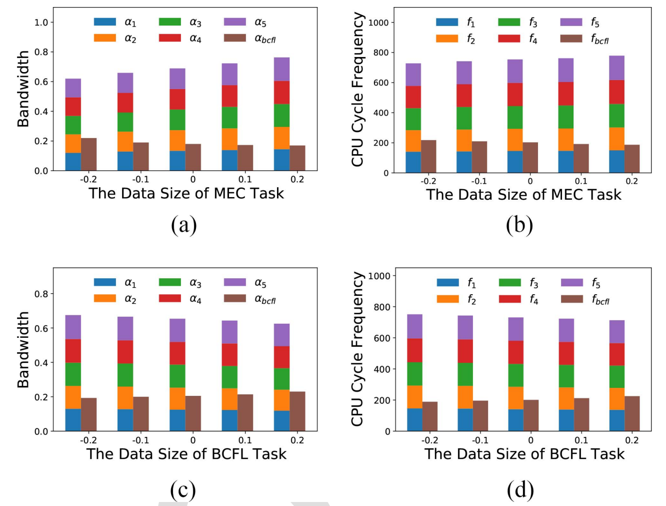


Fig. 5. Optimal resource allocation decisions based on the MC-ADMM algorithm. (a) Bandwidth. (b) CPU cycle frequencies. (c) Bandwidth. (d) CPU cycle frequencies.

BCFL task. Similarly, we can conclude from Fig. 3(c) and (d) that more resources will be distributed to the BCFL task and fewer resources will be assigned to the MEC tasks if the data size of the BCFL task is larger. The results match the intuition that the larger data size of a task requires more resources in communication and computing.

2) Evaluation of MC-ADMM Algorithm: In this part, the experiments are designed to evaluate the optimization objective of *P3* from the perspective of convergence and reveal the relationship between the data sizes of tasks and the optimal resource allocation decisions, i.e., the optimization variables in *P3*. The parameter setting is $D_i \in \{1, 2, 3, \ldots, 10\}$ and $T_i \in \{1, 2, 3, \ldots, 10\}$ with $N = 10$, $\rho = 0.5$ and $\beta = 1.0$, while others are the same with the above experiments.

First, we compare our proposed MC-ADMM algorithm with the above-mentioned benchmark methods. Similar to the setting of evaluating G-ADMM, C-ADMM is implemented by setting α_{bcfl} and f_{bcfl} as the constants. The results are reported in Fig. 4(a), which shows that our proposed algorithm performs well in solving *P3* since it can converge faster and achieve a lower stable value of the total energy cost than the other strategies.

The results are reported in Fig. 4(a), which shows that our proposed algorithm performs well in solving *P3* since it can converge and achieve a lower stable value of the total energy cost than the other three strategies.

Then, we test how penalty parameters $\rho \in \{0.10, 0.45, 0.50\}$ and $\beta \in \{0.10, 0.50, 1.00\}$ influence the convergence speed. From Fig. 4(b) and (c), we can observe that the larger penalties will cause faster convergence speed. What's more, we find that the value of ρ cannot be too large, or the algorithm would not converge. We also test the influence of the number of local devices ($N \in \{8, 9, 10\}$) with the results in Fig. 4(d) showing that more local devices will lead to more cost and slower convergence speed.

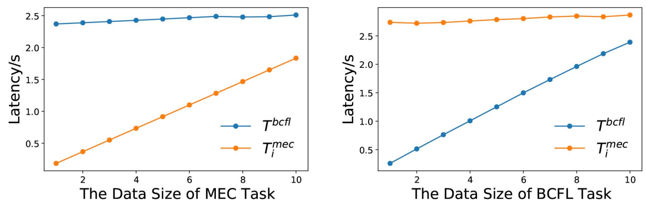
By comparing Figs. 2 and 4, it can be seen that MG-ADMM requires about 80 rounds to converge, while MC-ADMM only needs less than 50 rounds to reach the stable value, which indicates that the distributed algorithm is more effective.

In *P3*, we have to determine α_i and f_i for each $i \in \{1, 2, 3, \ldots, N\}$, as well as α_{bcfl} and f_{bcfl}. Thus, we need to calculate $2N+2$ variables. Here, we set $N = 5$, and we want to investigate how the increase and decrease in the sizes of data for the MEC and BCFL tasks affect the optimal decisions. We first let D_i decrease by 10% and 20%, and then increase it by 10% and 20%. The changes of the percentage are expressed as $\{-0.2, -0.1, 0, 0.1, 0.2\}$ in Fig. 5, where 0 relatively refers to the original data size. From the results in Fig. 5(a) and (b), we can see that more resources are allocated to the MEC tasks and fewer resources are distributed to the BCFL task when D_i increases. Conversely, the results in Fig. 5(c) and (d) show that more resources are assigned to the BCFL task when D_{bcfl} is larger. This is consistent with the changing trends in the homogeneous scenario and can be explained by the same reason that more resources are needed to finish tasks with larger data sizes.

3) Evaluation of Latency: In an ideal scenario, the MEC server can devote the appropriate resources to task processing based on the decisions obtained by the algorithms we designed. In this part, experiments are conducted to evaluate the latency of processing the MEC and BCFL tasks according to the decisions obtained from our algorithms.

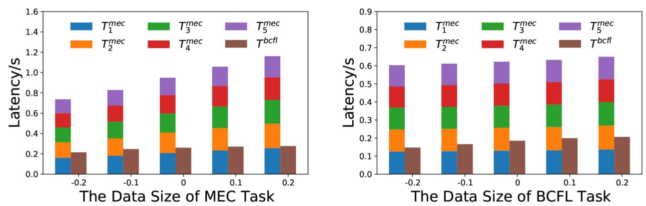
First, we let $T_i^{\mathrm{mec}} = T_i^{\mathrm{comm}} + T_i^{\mathrm{comp}}$ be the total time consumed by the MEC server in processing the MEC task submitted by user i according to the optimal decisions. Similarly, we can define $T^{\mathrm{bcfl}} = T_{\mathrm{bcfl}}^{\mathrm{comm}} + T_{\mathrm{bcfl}}^{\mathrm{comp}}$ as the time consumption for processing the BCFL task.

Based on the same experimental settings as in Fig. 3, we calculate the latency of completing both MEC and BCFL tasks. The results based on MG-ADMM are shown in Fig. 6. In Fig. 6(a), we can see that T^{bcfl} increases slightly and T_i^{mec} increases significantly when D_i increases. This is because when the data size of MEC task is larger, more time will be required to complete this task. While less resources will be allocated to process the BCFL task, T^{bcfl} will be also larger. Similarly, we can see the results with the change of D_{bcfl} in Fig. 6(b).



(a) (b)

Fig. 6. Latency based on the MG-ADMM algorithm. (a) Latency changes with the data size of MEC task. (b) Latency changes with the data size of BCFL task.



(a) (b)

Fig. 7. Latency based on the MC-ADMM algorithm. (a) Latency changes with the data size of MEC task. (b) Latency changes with the data size of BCFL task.

Then, we analyze the latency of the MC-ADMM algorithm with the same settings as in Fig. 5. The results are shown in Fig. 7. It is clear that when the data sizes of the MEC and BCFL tasks required to be processed increase, the time spent by the server to complete the tasks also increases.

VII. Conclusion

In this article, we are the first to address the resource allocation challenge for edge servers when they are required to handle both the BCFL and MEC tasks. We formulate the design of the resource allocation scheme into a convex, multivariate optimization problem with multiple inequity constraints, and then we design two algorithms based on ADMM to solve it in both homogeneous and heterogeneous scenarios. A solid theoretical analysis is conducted to prove the validity of our proposed solutions, and numerous experiments are carried out to evaluate the correctness and effectiveness of the algorithms.

We will enhance this article in the future. Specifically, first, we will study the optimization of energy consumption during blockchain consensus. Then, to fully utilize the resources of the entire blockchain network, we will design a joint optimization mechanism to enhance the cooperation among MEC servers. Lastly, we will design an incentive mechanism to motivate MEC servers to participate in processing both MEC and BCFL tasks.

Appendix A
Proof of Theorem 1

Proof: The Hessian Matrix of U respect to $\alpha_i, \alpha_{\text{bcfl}}, f_i, f_{\text{bcfl}}$ is given by

$$H_1 = \begin{pmatrix} \frac{2D_i p_i}{B\alpha_i^3 \ln\left(\frac{G_i P_i}{\delta^2}+1\right)} & 0 & 0 & 0 \\ 0 & \frac{2D_{\text{bcfl}} p_{\text{bcfl}}}{B\alpha_{\text{bcfl}}^3 \ln\left(\frac{G_{\text{bcfl}} P_{\text{bcfl}}}{\delta^2}+1\right)} & 0 & 0 \\ 0 & 0 & 2N\gamma\mu_i & 0 \\ 0 & 0 & 0 & 2\gamma\mu_{\text{bcfl}} \end{pmatrix}.$$

The eigenvalues of matrix H_1 are

$$V_1 = \begin{bmatrix} 2\gamma\mu_{\text{bcfl}} \\ 2N\gamma\mu_i \\ \frac{2D_{\text{bcfl}} p_{\text{bcfl}}}{B\alpha_{\text{bcfl}}^3 \ln\left(\frac{G_{\text{bcfl}} P_{\text{bcfl}}}{\delta^2}+1\right)} \\ \frac{2D_i p_i}{B\alpha_i^3 \ln\left(\frac{G_i P_i}{\delta^2}+1\right)} \end{bmatrix}.$$

It can be seen that all elements in vector V_1 are positive. Hence, matrix H_1 is a positive definite matrix, and we can prove that the optimization objective function U is convex. ■

Appendix B
Proof of Theorem 2

Proof: The Hessian matrix of $\mathcal{L}_1$ is shown in (25), shown at the top of the page.

Then we calculate the eigenvalues of matrix H_2 as

$$V_2 = \begin{bmatrix} \frac{D_i \log_2(2\lambda_2+2P_i+\rho)}{\alpha_i^3 B \ln\left(1+\frac{P_i G_i}{\delta^2}\right)} - \frac{N^2\rho}{1-N\alpha_i-\alpha_{\text{bcfl}}} \\ \frac{D_{\text{bcfl}} \log_2(2\lambda_1+2P_{\text{bcfl}}+\rho)}{\alpha_{\text{bcfl}}^3 B \ln\left(1+\frac{P_{\text{bcfl}} G_{\text{bcfl}}}{\delta^2}\right)} - \frac{3\rho}{8(1-\alpha_{\text{bcfl}}-N\alpha_i)} \\ 2\gamma\mu_{\text{bcfl}} \\ 2N\gamma\mu_i \end{bmatrix}.$$

In vector V_2, it is clear that $2\gamma\mu_{\text{bcfl}}$ and $2N\gamma\mu_i$ are positive. As for $([D_i \log_2(2\lambda_2 + 2P_i + \rho)]/[\alpha_i^3 B \ln(1+[(P_i G_i)/\delta^2])]) - ([N^2\rho]/[1 - N\alpha_i - \alpha_{\text{bcfl}}])$ and $([D_{\text{bcfl}} \log_2(2\lambda_1 + 2P_{\text{bcfl}} + \rho)]/ [\alpha_{\text{bcfl}}^3 B \ln(1 + [(P_{\text{bcfl}} G_{\text{bcfl}})/\delta^2])]) - ([3\rho]/[8(1 - \alpha_{\text{bcfl}} - N\alpha_i)])$, we cannot know whether they are nonnegative. If we let $([D_i \log_2(2\lambda_2 + 2P_i + \rho)]/[\alpha_i^3 B \ln(1 + [P_i G_i/\delta^2])]) - ([N^2\rho]/[1-N\alpha_i-\alpha_{\text{bcfl}}]) < 0$, then we have $([N^2\rho]/[1-N\alpha_i-\alpha_{\text{bcfl}}]) > ([D_i \log_2(2\lambda_2 + 2P_i + \rho)]/ [\alpha_i^3 B \ln(1 + [P_i G_i/\delta^2])])$. In other words, if the above condition is satisfied, then we can say that at least one of the elements in vector V_2 is negative. In this way, matrix H_2 is a positive semi-definite matrix. Thus, $\mathcal{L}_1$ has a saddle point. ■

Proof: The Hessian matrix of $\mathcal{L}_2$ is shown in (26), shown at the top of the page.

Appendix C
Proof of Theorem 3

Then we calculate the eigenvalues of matrix H_3 as

$$V_3 = \begin{bmatrix} f_{\text{bcfl}}^2 \gamma\mu_{\text{bcfl}} + 2N\gamma\mu_i + \frac{\mu_i \rho}{f_i^3} \\ \frac{4\gamma\mu_{\text{bcfl}}}{f_{\text{bcfl}}^3} \\ -\frac{D_i \log_2(\rho+2NP_i)}{B\alpha_i^3 \ln\left(\frac{G_i P_i}{\delta^2}+1\right)} \\ -\frac{2D_{\text{bcfl}} \log_2(P_{\text{bcfl}}+\eta_1)}{B\alpha_{\text{bcfl}}^3 \ln\left(\frac{G_{\text{bcfl}} P_{\text{bcfl}}}{\delta^2}+1\right)} \end{bmatrix}.$$

Clearly, $f_{\text{bcfl}}^2 \gamma\mu_{\text{bcfl}} + 2N\gamma\mu_i + ([\mu_i\rho]/[f_i^3]) > 0$ and $([4\gamma\mu_{\text{bcfl}}]/[f_{\text{bcfl}}^3]) > 0$, while $-([D_i \log_2(\rho + 2NP_i)]/ [B\alpha_i^3 \ln([G_i P_i/\delta^2] + 1)]) < 0$ and $-([2D_{\text{bcfl}} \log_2(P_{\text{bcfl}} + \eta_1)]/ [B\alpha_{\text{bcfl}}^3 \ln([G_{\text{bcfl}} P_{\text{bcfl}}/\delta^2] + 1)]) < 0$. So H_3 is a semi-definite matrix, and $\mathcal{L}_2$ has a saddle point. ■

$$H_2 = \begin{pmatrix} \frac{N^2\rho}{1-N\alpha_i-\alpha_{\mathrm{bcfl}}} + \frac{D_i \log_2(2\lambda_2+2P_i+\rho)}{\alpha_i^3 B \ln\left(1+\frac{P_i G_i}{\delta^2}\right)} & 0 & 0 & 0 \\ 0 & \frac{3\rho}{8\,(1-\alpha_{\mathrm{bcfl}}-N\,\alpha_i)} + \frac{D_{\mathrm{bcfl}} \log_2(2\lambda_1+2P_{\mathrm{bcfl}}+\rho)}{\alpha_{\mathrm{bcfl}}^3 B \ln\left(1+\frac{P_{\mathrm{bcfl}} G_{\mathrm{bcfl}}}{\delta^2}\right)} & 0 & 0 \\ 0 & 0 & 2N\gamma\,\mu_i & 0 \\ 0 & 0 & 0 & 2\gamma\,\mu_{\mathrm{bcfl}} \end{pmatrix} \tag{25}$$

$$H_3 = \begin{pmatrix} \frac{D_i \log_2(\rho+2NP_i)}{B\alpha_i^3 \log_2\left(\frac{G_i P_i}{\delta^2}+1\right)} & 0 & 0 & 0 \\ 0 & \frac{2D_{\mathrm{bcfl}} \log_2(P_{\mathrm{bcfl}}+\eta_1)}{B\alpha_{\mathrm{bcfl}}^3 \log_2\left(\frac{G_{\mathrm{bcfl}} P_{\mathrm{bcfl}}}{\delta^2}+1\right)} & 0 & 0 \\ 0 & 0 & f_{\mathrm{bcfl}}^2 \gamma\,\mu_{\mathrm{bcfl}} + 2N\gamma\,\mu_i + \frac{\mu_i \rho}{f_i^3} & 0 \\ 0 & 0 & 0 & \frac{4\gamma\,\mu_{\mathrm{bcfl}}}{f_{\mathrm{bcfl}}^3} \end{pmatrix} \tag{26}$$

REFERENCES

[1] N. Abbas, Y. Zhang, A. Taherkordi, and T. Skeie, "Mobile edge computing: A survey," *IEEE Internet Things J.*, vol. 5, no. 1, pp. 450–465, Feb. 2017.

[2] X. Wang et al., "Wireless powered mobile edge computing networks: A survey," *ACM Comput. Surveys*, vol. 55, no. 13, pp. 1–37, 2023.

[3] X. Sun and N. Ansari, "EdgeIoT: Mobile edge computing for the Internet of Things," *IEEE Commun. Mag.*, vol. 54, no. 12, pp. 22–29, Dec. 2016.

[4] A. H. Sodhro, Z. Luo, A. K. Sangaiah, and S. W. Baik, "Mobile edge computing based QoS optimization in medical healthcare applications," *Int. J. Inf. Manage.*, vol. 45, pp. 308–318, Apr. 2019.

[5] Y. Chen, Y. Zhang, S. Maharjan, M. Alam, and T. Wu, "Deep learning for secure mobile edge computing in cyber-physical transportation systems," *IEEE Netw.*, vol. 33, no. 4, pp. 36–41, Aug. 2019.

[6] K. Bonawitz et al., "Towards federated learning at scale: System design," in *Proc. Mach. Learn. Syst.*, 2019, pp. 374–388.

[7] Y. Zhao et al., "Privacy-preserving blockchain-based federated learning for IoT devices," *IEEE Internet Things J.*, vol. 8, no. 3, pp. 1817–1829, Feb. 2021.

[8] S. R. Pokhrel and J. Choi, "Federated learning with blockchain for autonomous vehicles: Analysis and design challenges," *IEEE Trans. Commun.*, vol. 68, no. 8, pp. 4734–4746, Aug. 2020.

[9] Z. Wang and Q. Hu, "Blockchain-based federated learning: A comprehensive survey," 2021, *arXiv:2110.02182*.

[10] Y. Zhao, J. Zhao, L. Jiang, R. Tan, and D. Niyato, "Mobile edge computing, blockchain and reputation-based crowdsourcing IoT federated learning: A secure, decentralized and privacy-preserving system," 2020, *arXiv:1906.10893*.

[11] Q. Hu, Z. Wang, M. Xu, and X. Cheng, "Blockchain and federated edge learning for privacy-preserving mobile crowdsensing," *IEEE Internet Things J.*, vol. 10, no. 14, pp. 12000–12011, Jul. 2023.

[12] S. Sardellitti, G. Scutari, and S. Barbarossa, "Joint optimization of radio and computational resources for multicell mobile-edge computing," *IEEE Trans. Signal Inf. Process. Netw.*, vol. 1, no. 2, pp. 89–103, Jun. 2015.

[13] 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, Sep. 2021.

[14] L. Wan, L. Sun, X. Kong, Y. Yuan, K. Sun, and F. Xia, "Task-driven resource assignment in mobile edge computing exploiting evolutionary computation," *IEEE Wireless Commun.*, vol. 26, no. 6, pp. 94–101, Dec. 2019.

[15] N. Q. Hieu, T. T. Anh, N. C. Luong, D. Niyato, D. I. Kim, and E. Elmroth, "Resource management for blockchain-enabled federated learning: A deep reinforcement learning approach," 2020, *arXiv:2004.04104*.

[16] J. Li et al., "Blockchain assisted decentralized federated learning (BLADE-FL): Performance analysis and resource allocation," 2021, *arXiv:2101.06905*.

[17] J. Zhang, Y. Liu, X. Qin, X. Xu, and P. Zhang, "Adaptive resource allocation for blockchain-based federated learning in Internet of Things," *IEEE Internet Things J.*, vol. 10, no. 12, pp. 10621–10635, Jun. 2023.

[18] S. Boyd, N. Parikh, and E. Chu, *Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers*. Hanover, MA, USA: Now Pub. Inc., 2011.

[19] M. H. Ur Rehman, K. Salah, E. Damiani, and D. Svetinovic, "Towards blockchain-based reputation-aware federated learning," in *Proc. IEEE Conf. Comput. Commun. Workshops*, 2020, pp. 183–188.

[20] M. Shen et al., "Exploiting unintended property leakage in blockchain-assisted federated learning for intelligent edge computing," *IEEE Internet Things J.*, vol. 8, no. 4, pp. 2265–2275, Feb. 2021.

[21] G. Qu, N. Cui, H. Wu, R. Li, and Y. M. Ding, "ChainFL: A simulation platform for joint federated learning and blockchain in edge/cloud computing environments," *IEEE Trans. Ind. Informat.*, vol. 18, no. 5, pp. 3572–3581, May 2022.

[22] X. Huang, Y. Wu, C. Liang, Q. Chen, and J. Zhang, "Distance-aware hierarchical federated learning in blockchain-enabled edge computing network," *IEEE Internet Things J.*, vol. 10, no. 21, pp. 19163–19176, Nov. 2023.

[23] S. Liu, X. Wang, L. Hui, and W. Wu, "Blockchain-based Decentralized federated learning method in edge computing environment," *Appl. Sci.*, vol. 13, no. 3, p. 1677, 2023.

[24] Z. Wang, Q. Hu, R. Li, M. Xu, and Z. Xiong, "Incentive mechanism design for joint resource allocation in blockchain-based federated learning," *IEEE Trans. Parallel Distrib. Syst.*, vol. 34, no. 5, pp. 1536–1547, May 2023.

[25] S. Fan, H. Zhang, Y. Zeng, and W. Cai, "Hybrid blockchain-based resource trading system for federated learning in edge computing," *IEEE Internet Things J.*, vol. 8, no. 4, pp. 2252–2264, Feb. 2021.

[26] B. He, M. Tao, and X. Yuan, "A splitting method for separable convex programming," *IMA J. Numer. Anal.*, vol. 35, no. 1, pp. 394–426, Jan. 2015.

[27] M. Castro and B. Liskov, "Practical byzantine fault tolerance," in *Proc. 3rd Symp. Oper. Syst. Design Implement.*, 1999, pp. 173–186.

[28] D. Ongaro and J. Ousterhout, "In search of an understandable consensus algorithm," in *Proc. USENIX Annu. Tech. Conf. (ATC)*, 2014, pp. 305–319.

[29] C. E. Shannon, "A mathematical theory of communication," *Bell Syst. Tech. J.*, vol. 27, no. 3, pp. 379–423, Jul. 1948.

[30] T. D. Burd and R. W. Brodersen, "Processor design for portable systems," *J. VLSI Signal Process. Syst. Signal, Image Video Technol.*, vol. 13, no. 2, pp. 203–221, 1996.

[31] Z. Xiong, J. Kang, D. Niyato, P. Wang, and H. V. Poor, "Cloud/edge computing service management in blockchain networks: Multi-leader multi-follower game-based ADMM for pricing," *IEEE Trans. Services Comput.*, vol. 13, no. 2, pp. 356–367, Apr. 2019.

[32] A. A. Al-Habob, O. A. Dobre, A. G. Armada, and S. Muhaidat, "Task scheduling for mobile edge computing using genetic algorithm and conflict graphs," *IEEE Trans. Veh. Technol.*, vol. 69, no. 8, pp. 8805–8819, Aug. 2020.

Zhilin Wang (Graduate Student Member, IEEE) is currently pursuing the Ph.D. degree in computer science with Purdue University at Indianapolis, Indianapolis, IN, USA.

His research interests include federated learning, security and privacy, distributed optimization, and blockchain.

Dr. Wang is actively contributed as a reviewer for prestigious academic journals and conferences, such as IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, IEEE INTERNET OF THINGS JOURNAL, *Journal of Network and Computer Applications* (Elsevier), IEEE TCCN, and IEEE ICC.

Qin Hu received the Ph.D. degree in computer science from The George Washington University, Washington, DC, USA, in 2019.

She is currently an Assistant Professor with the Department of Computer Science, Indiana University at Indianapolis, Indianapolis, IN, USA. Her research interests include wireless and mobile security, edge computing, and blockchain.

Dr. Hu serves on the editorial board of two journals, the guest editor for five journals, the TPC/publicity chair for several workshops, and the TPC member for several conferences.

Zehui Xiong (Senior Member, IEEE) received the Ph.D. degree from Nanyang Technological University, Singapore.

He is currently an Assistant Professor with the Pillar of Information Systems Technology and Design, Singapore University of Technology and Design, Singapore. He has published more than 140 research papers in leading journals and flagship conferences and many of them are ESI highly cited papers. His research interests include wireless communications, network games and economics, blockchain, and edge intelligence.

Dr. Xiong has won over ten Best Paper Awards in International Conferences and is listed in the World's Top 2% Scientists identified by Stanford University. He is currently serving as the Editor or a Guest Editor for many leading journals, including IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS, IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY, IEEE INTERNET OF THINGS JOURNAL, TCCN, IEEE TRANSACTIONS ON NETWORK SCIENCE AND ENGINEERING, *Information Systems Journal*, and IEEE/CAA JOURNAL OF AUTOMATICA SINICA.

Yuan Liu received the B.Sc. degree from Honor School, Harbin Institute of Technology, Harbin, China, in 2010, and the Ph.D. degree from the School of Computer Engineering, Nanyang Technological University (NTU), Singapore, in 2014.

She is currently a Professor with the Cyberspace Institute of Advanced Technology, Guangzhou University, Guangzhou, China. From 2015 to 2021, she was an Associate Professor with Northeastern University, Shenyang, China. From 2014 to 2015, she was the Research Fellow with the Joint NTU-UBC Research Centre of Excellence in Active Living for the Elderly, LILY, NTU, Singapore. Her research interests include trust-based incentive mechanism design, federated learning, trust management, blockchain consensus protocols, and blockchain powered artificial intelligence.

Dusit Niyato (Fellow, IEEE) received the B.Eng. degree from King Mongkuts Institute of Technology Ladkrabang, Bangkok, Thailand, in 1999, and the Ph.D. degree in electrical and computer engineering from the University of Manitoba, Winnipeg, MB, Canada, in 2008.

He is a Professor with the School of Computer Science and Engineering, Nanyang Technological University, Singapore. His research interests are in the areas of sustainability, edge intelligence, decentralized machine learning, and incentive mechanism design.