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

Zhilin Wang, Qin Hu, Zehui Xiong

Abstract—With the development of mobile edge computing (MEC) and blockchain-based federated learning (BCFL), a number of studies suggest deploying BCFL on edge servers. In this case, resource-limited edge servers need to serve both mobile devices for their offloading tasks and the BCFL system for model training and blockchain consensus in a cost-efficient manner without sacrificing the service quality to any side. To address this challenge, this paper proposes a resource allocation scheme for edge servers, aiming to provide the optimal services with the minimum cost. Specifically, we first analyze the energy consumed by the MEC and BCFL tasks, and then use the completion time of each task as the service quality constraint. Then, we model the resource allocation challenge into a multivariate, multi-constraint, and convex optimization problem. To solve the problem in a progressive manner, we design two algorithms based on the alternating direction method of multipliers (ADMM) in both the homogeneous and heterogeneous situations with equal and on-demand resource distribution strategies, respectively. The validity of our proposed algorithms is proved via rigorous theoretical analysis. Through extensive experiments, the convergence and efficiency of our proposed resource allocation schemes are evaluated. To the best of our knowledge, this is the first work to investigate the resource allocation dilemma of edge servers for BCFL in MEC.

Index Terms—Blockchain, federated learning, resource allocation, mobile edge computing, ADMM

arXiv:2206.02243v1 [cs.DC] 5 Jun 2022

1 Introduction

As embedded sensors are widely deployed on mobile devices, such as smartphones and smart vehicles, they can pervasively perceive the physical world and collect an extensive amount of data. With the advances of 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], [4], smart healthcare [5], [6], and smart transportation [7], [8].

To address the main challenges of federated learning (FL) [9], [10], 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 [11], [12], which is termed as Blockchain-based FL (BCFL). This new framework connects participants in FL, i.e., clients, through blockchain network and requires them to complete both FL and blockchain related operations, such as data collection, model training, and block generation [13]. As for a client in BCFL, a large amount of resources will be consumed in completing the BCFL task, however, making it an impractical job for mobile devices with constrained resources. To address this issue, researchers advocate deploying BCFL at the edge given that edge servers usually have strong computing, communication, and storage capabilities for FL model training and blockchain consensus [14], [15].

In this case, the MEC servers are responsible for completing both the BCFL and MEC tasks. For the MEC task, the edge server is usually required to allocate the communication resource (e.g., bandwidth) for mobile devices to transfer data, the storage resource for data caching, and the computing resources (e.g., CPU cycle frequency) for computing based on the received data. Similarly, for the BCFL task, the edge server needs to distribute the bandwidth 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 CPU resource for FL model training and updating, as well as the generation of new blocks. Since both the MEC and BCFL tasks are time-sensitive and could be performed simultaneously at the MEC servers, the servers have to deal with the challenge of serving both the lower-layer mobile edge devices and the upper-layer BCFL system without significant delay, which makes it a necessity to design reasonable resource allocation schemes for them.

The existing resource allocation mechanisms for BCFL and MEC can be classified into two main categories, where the first type of studies focus on allocating resources to each device for the requested MEC task [16]–[18], and the other is to allocate resources for model training and block generation in the BCFL task [19], [20]. Although the state-of-the-art studies can help edge servers allocate resources to

This work is partly supported by the US NSF under grant CNS-2105004.

- *Zhilin Wang and Qin Hu are with the Department of Computer & Information Science, Indiana University-Purdue University Indianapolis, USA. The corresponding author is Qin Hu. Email: {wangzhil, qinhu}@iu.edu*
- *Zehui Xiong is with the Pillar of Information Systems Technology & Design, Singapore University of Technology Design, Singapore. Email: zehui_xiong@sutd.edu.sg*

This is the author's manuscript of the article published in final edited form as:
Wang, Z., Hu, Q., Xiong, Z., Liu, Y., & Niyato, D. (2024). Resource Optimization for Blockchain-Based Federated Learning in Mobile Edge Computing. IEEE Internet of Things Journal, 11(9), 15166–15178. IEEE Internet of Things Journal. https://doi.org/10.1109/JIOT.2023.3347524

handling the MEC or BCFL tasks, these mechanisms have never considered the 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, multi-constraint, and convex optimization problem. However, solving this optimization problem faces the following challenges: 1) there are multiple variables since assigning resources to the MEC task means making decisions on resource allocation to each device, where the number of variables increases with the device quantity; and 2) there are too many constraints, making the solution non-trivial.

These two challenges result in the invalidity of traditional optimization methods for multiple variable calculation. Therefore, we design a scheme based on a distributed optimization algorithm, named the alternating direction method of multipliers (ADMM), which is the combination of *dual ascent* and *dual decomposition*, and can determine multiple variables by iterations in a distributed manner [21]. We adopt the ADMM-based algorithm to solve our proposed resource allocation problem in two scenarios progressively for easy understanding. Specifically, we first apply the *modified general ADMM* (MG-ADMM) algorithm for the homogeneous scenario, which distributes resources to each local device equally; then we propose the *modified consensus ADMM* (MC-ADMM) based algorithm to assign resources to devices on demand in the heterogeneous scenario. Besides, by adding additional regularization terms during the variable iteration process, our proposed algorithms can converge in the case of more than two variables, which is confirmed by theoretical analysis. Finally, we conduct extensive experiments to testify the convergence and the effectiveness of our proposed resource allocation schemes.

To the best of our knowledge, we are the first to tackle the challenge of resource allocation for edge servers in the deployment of BCFL at edge. And our contributions can be summarized as follows:

- We formulate the resource allocation problem of BCFL in MEC as a multivariate and multi-constraint optimization problem, where the solution is the resource allocation scheme for edge servers to simultaneously handle both the MEC and BCFL tasks without delay.
- To solve the above optimization problem progressively, we design two algorithms based on MG-ADMM and MC-ADMM for both the homogeneous and heterogeneous scenarios.
- 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, and prove the validity with solid theoretical analysis.
- We conduct numerous experimental evaluations to prove that the optimization solutions are valid and our proposed resource allocation schemes are effective.

The rest of this paper is organized as follows. We introduce the system model and problem formulation in Section 2. The MG-ADMM based algorithm to solve the optimization problem in the homogeneous scenario and the MC-ADMM based algorithm for the heterogeneous scenario are displayed in Sections 3 and 4, respectively. Experimental evaluations are presented in Section 5. We discuss the related work in Section 6. Finally, we conclude this paper in Section 7.

2 System Model and Problem Formulation

In this section, we will discuss the system model from a general perspective, and then we explore both communication models and computing models of our proposed system respectively. We also analyze the cost model of our proposed resource allocation scheme.

2.1 System Model Overview

We consider an edge server S connected with N local mobile devices, denoted as $i \in \{1, 2, 3, \cdots, N\}$. Local devices are usually lacking of resources, so they may choose to offload their computing tasks to its nearby edge server S. In this way, S can provide necessary resources to help local devices finish their offloading tasks in the MEC scenario. At the same time, there are multiple edge servers connecting via blockchain network, where they conduct federated learning (FL) to form a blockchain-based FL (BCFL) service provider. In other words, edge servers will be responsible for not only providing computing services to local devices, but also maintaining the BCFL system at the same time. The topology of our considered system is shown in Fig. 1.

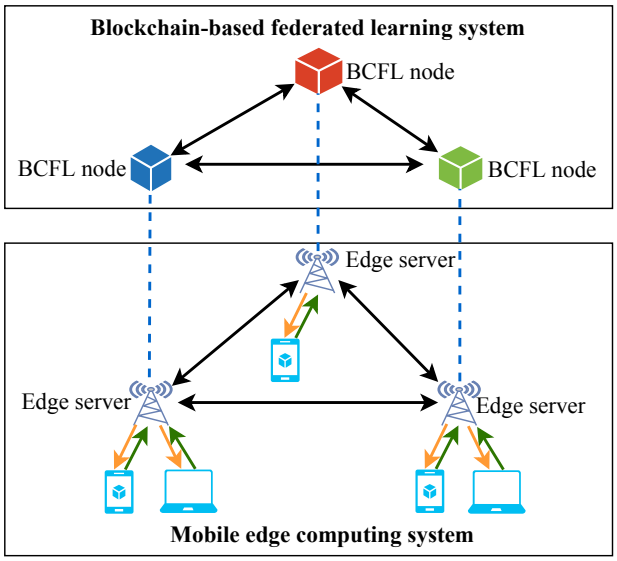


Fig. 1: The topology of BCFL in MEC.

In the MEC, 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 time constraint of this task to

be finished. Once S accepts tasks, local devices transmit their data to S. As for the BCFL system, according to [13], we consider a fully coupled BCFL which runs FL on a consortium blockchain. First, the clients of FL, i.e., edge servers, train the local models using local data which may be collected from other devices or by themselves, and then they also work as blockchain nodes to generate new blocks which contain the local model updates and the newly updated global model of FL. For simplicity, we treat the FL and the blockchain jobs together as the BCFL task, consuming computing, communication, and storage resources.

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

2.2 Communication Models

In this subsection, we would like to model the communication resource consumption for finishing the MEC and BCFL tasks, respectively.

2.2.1 MEC Task

Communications between any device i and the server S include sending offloading request, sending original data and returning computing results. Since the sizes of the offloading request and computing results are smaller than that of the data, we only consider the transmission of original data from devices to the server.

According to Shannon Bound, the data transmission rate from local device i to edge server S is defined as

$$r_i^{comm}(\alpha_i) = \alpha_i B \log 2(1 + \frac{P_i G_i}{\delta^2}),$$

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 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

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

which indicates that the transmission time cost is correlated to the data size of MEC task.

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

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

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

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

2.2.2 BCFL task

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

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

And the time cost of transmission in the BCFL task is

$$T_{bcfl}^{comm}(\alpha_{bcfl}) = \frac{\widehat{D_{bcfl}}}{r_{bcfl}^{comm}(\alpha_{bcfl})},$$

where $\widehat{D_{bcfl}}$ 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

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

2.3 Computing Models

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

2.3.1 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

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

According to [22], the energy cost of computing one single task of device i is

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

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

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

2.3.2 BCFL Task

Similarly, we define $f_{bcfl} \in (0, F)$ as the CPU cycle frequency allocated to the BCFL task. Let $\mu_{bcfl} = D_{bcfl} d_{bcfl}$ 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 have the time cost of computing the BCFL task:

$$T_{bcfl}^{comp}(f_{bcfl}) = \frac{\mu_{bcfl}}{f_{bcfl}}.$$

In this way, the energy cost of computing the BCFL task is calculated as:

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

2.4 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 analysis, we know that U is composed of the transmission cost and the computing cost. Then, we have

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

2.5 Problem Formulation

The purpose of our resource allocation mechanism is to allow the edge server to handle both the MEC and BCFL tasks satisfying resource and time constraints with the minimum cost. The edge server should make the decisions about how many CPU cycle frequencies 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 below:

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

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 are not out of 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.

Theorem 2.1. *The optimization objective function $U(\alpha_i, \alpha_{bcfl}, f_i, f_{bcfl})$ is convex.*

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

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

The eigenvalues of matrix H_1 are:

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

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

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; 2) there are multiple constraints, making it harder to find the optimal solutions. Hence, we need to design solutions for **P1**, which will be introduced in the following sections.

3 MG-ADMM based Solution in the Homogeneous Situation

To present our resource allocation scheme in a progressive way, we will give a benchmark solution of **P1** in the homogeneous situation where all MEC tasks have the same data size and time requirement. In this case, we start from a simple case of **P1** in this section, where an equal distribution strategy is considered to allocate resources to all local devices, including both the bandwidth and CPU cycle frequencies. The equal distribution strategy means that the edge server distributes the communication and computing resources to each local device for MEC tasks in an equal way, that is, α_i and f_i are the same for any arbitrary device i.

According to Boyd *et al.* [21], the alternating direction method of multipliers (ADMM), combining *dual ascend* and *dual decomposition*, is designed to solve problems which are multivariate, separable and convex. We will solve **P1** with the equal distribution strategy based on the modified general ADMM (MG-ADMM) method, which is derived from the basic form of ADMM. In the following, we first introduce MG-ADMM and reformulate the problem based on the MG-ADMM algorithm, and then we explain how we solve **P1**.

3.1 Brief Introduction to MG-ADMM

First, we introduce G-ADMM as the basis of MG-ADMM. According to Boyd *et al.* [21], G-ADMM tries to solve the following problem:

$$\begin{aligned} & \underset{x,y}{\arg\min} \quad f(x) + g(z) \\ & s.t: Ax + Bz = c, \end{aligned}$$

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, and x and z are two

parameters. The objective of G-ADMM is to find the optimal value p^*:

$$p^* = \inf \{f(x) + g(z) | Ax + Bz = c\}.$$

Then, we can form the augmented Lagrangian as below:

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

where y is the Lagrange multiplier, and $\rho > 0$ is the penalty parameter.

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

$$x^{k+1} := \arg\min \mathcal{L}_\rho(x, z^k, y^k),$$

$$z^{k+1} := \arg\min \mathcal{L}_\rho(x^{k+1}, z, y^k),$$

$$y^{k+1} := y^k + \rho(Ax^{k+1} + Bz^{k+1} - c).$$

It has been proved that when the following two conditions are satisfied, the 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; 2) the augmented Lagrangian $\mathcal{L}_\rho(x, z, y)$ has a saddle point.

The above G-ADMM algorithm is the basic form, which is effective to solve the problem which has 2-block (i.e., two separable functions). However, when we need to solve the problem with more than two separable functions, implementing G-ADMM directly can't guarantee convergence.

To handle this issue, He *et al.* [23] propose a novel operator splitting method. In this paper, we term it as MG-ADMM. Let's take 3-block separable minimization problem as the example to describe MG-ADMM when the number of blocks is more then 2.

The form of 3-block separable minimization problem is:

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

Then the Lagrangian function is:

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

And the updates of iterations are:

$$x^{k+1} := \arg\min \left\{\mathcal{L}_\rho^\beta(x, z^k, y^k, \lambda^k)\right\},$$

$$z^{k+1} := \arg\min \left\{\mathcal{L}_\rho^\beta(x^{k+1}, z, y^k, \lambda^k) + \frac{\rho}{2}\beta \left\|B(z - z^k)\right\|_2^2\right\},$$

$$y^{k+1} := \arg\min \left\{\mathcal{L}_\rho^\beta(x^{k+1}, z^k, y, \lambda^k) + \frac{\rho}{2}\beta \left\|C(y - y^k)\right\|_2^2\right\},$$

$$\lambda^{k+1} := \lambda^k - \beta(Ax^{k+1} + By^{k+1} + Cz^{k+1} - b),$$

where $\beta > 1$ is the penalty parameter.

3.2 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. As for the energy cost of computing, it is the sum of all devices' costs:

$$E_{total}^{comp}(f^*) = \sum_{i=1}^{N} \gamma \mu_i f_i^2 = N\gamma\mu_i f^{*2}.$$

And the communication cost of the MEC tasks is

$$E_{total}^{comm}(\alpha^*) = \sum_{i=1}^{N} P_i T_i^{comm} = N P_i \frac{D_i}{\alpha^* B \log 2(1 + \frac{P_i G_i}{\delta^2})}.$$

Thus, we can rewrite U as:

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

Besides, the offloading time costs of communication and computing are:

$$\widehat{T_i^{comp}}(f^*) = \frac{\mu_i}{f^*},$$

$$\widehat{T_i^{comm}}(\alpha^*) = \frac{D_i}{\alpha^* B \log 2(1 + \frac{P_i G_i}{\delta^2})}.$$

Based on the above analysis, in the case of homogeneous situation, we need to determine four variables, i.e., $\alpha^*, \alpha_{bcfl}, f^*$ and f_{bcfl}. We can easily prove that U' is convex based on Theorem 2.1. So we apply MG-ADMM to optimize U' and derive the optimal variables. In this way, we can reformulate **P1** as below:

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

3.3 Solution based on MG-ADMM

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

$$
\begin{aligned}
\mathcal{L}_1 &= \mathcal{L}(\alpha^*, \alpha_{bcfl}, f^*, f_{bcfl}, \lambda_1, \lambda_2, \lambda_3, \lambda_4, \lambda_5) \\
&= U' + \lambda_1(T_{bcfl}^{comm} + T_{bcfl}^{comp} - T_{bcfl}) \\
&+ \lambda_2(\widehat{T_i^{comm}} + \widehat{T_i^{comp}} - T_i) \\
&+ \lambda_3(\alpha_{bcfl} + N\alpha^* - 1) \\
&+ \lambda_4(f_{bcfl} + Nf^* - F) \\
&+ \lambda_5(D_{bcfl} + \widehat{D_{bcfl}} + \sum_i^N D_i - D) \\
&+ \frac{\rho}{2}\left\|T_{bcfl}^{comm} + T_{bcfl}^{comp} - T_{bcfl}\right\|_2^2 \\
&+ \frac{\rho}{2}\left\|\widehat{T_i^{comm}} + \widehat{T_i^{comp}} - T_i\right\|_2^2 \\
&+ \frac{\rho}{2}\left\|\alpha_{bcfl} + N\alpha^* - 1\right\|_2^2 \\
&+ \frac{\rho}{2}\left\|f_{bcfl} + Nf^* - 1\right\|_2^2 \\
&+ \frac{\rho}{2}\left\|D_{bcfl} + \widehat{D_{bcfl}} + \sum_i^N D_i - D\right\|_2^2,
\end{aligned}
$$

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 3.1. *The augmented Lagrangian of **P2**, i.e., $\mathcal{L}_1$, has a saddle point.*

Proof. The Hessian matrix of $\mathcal{L}_1$ is shown in (3).

Then we calculate the eigenvalues of matrix H_2 as

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

In vector V_2, it is clear that $2\gamma\mu_{bcfl}$ and $2N\gamma\mu_i$ are positive. As for $\frac{D_i \log(2)(2\lambda_2+2P_i+\rho)}{\alpha_i^3 B \ln(1+\frac{P_i G_i}{\delta^2})} - \frac{N^2\rho}{1-N\alpha_i-\alpha_{bcfl}}$ and $\frac{D_{bcfl} \log(2)(2\lambda_1+2P_{bcfl}+\rho)}{\alpha_{bcfl}^3 B \ln(1+\frac{P_{bcfl} G_{bcfl}}{\delta^2})} - \frac{3\rho}{8(1-\alpha_{bcfl}-N\alpha_i)}$, we cannot know whether they are non-negative. If we let $\frac{D_i \log(2)(2\lambda_2+2P_i+\rho)}{\alpha_i^3 B \ln(1+\frac{P_i G_i}{\delta^2})} - \frac{N^2\rho}{1-N\alpha_i-\alpha_{bcfl}} < 0$, then we have $\frac{N^2\rho}{1-N\alpha_i-\alpha_{bcfl}} > \frac{D_i \log(2)(2\lambda_2+2P_i+\rho)}{\alpha_i^3 B \ln(1+\frac{P_i G_i}{\delta^2})}$. 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. □

Let $k \in \{1, 2, \cdots, K\}$ be the iteration round, and the updates of variables can be expressed as:

$$
\alpha^{*k+1} := \arg\min \mathcal{L}(\alpha^*, \alpha_{bcfl}^k, f^{*k}, f_{bcfl}^k, \lambda_1^k, \lambda_2^k, \lambda_3^k, \lambda_4^k, \lambda_5^k), \tag{4}
$$

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

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

$$
\begin{aligned}
f_{bcfl}^{k+1} := \arg\min(&\mathcal{L}(\alpha^{*k+1}, \alpha_{bcfl}^k, f^{*k}, f_{bcfl}, \\
&\lambda_1^k, \lambda_2^k, \lambda_3^k, \lambda_4^k, \lambda_5^k) + \frac{\rho}{2}\beta\left\|f_{bcfl} - f_{bcfl}^k\right\|_2^2),
\end{aligned} \tag{7}
$$

where $\beta > 0$ is the penalty parameter.

The updates of augmented Lagrange multipliers are:

$$
\lambda_1^{k+1} := \lambda_1^k - \beta(T_{bcfl}^{comm}(\alpha_{bcfl}^{k+1}) + T_{bcfl}^{comp}(f_{bcfl}^{k+1}) - T_{bcfl}), \tag{8}
$$

$$
\lambda_2^{k+1} := \lambda_2^k - \beta(\widehat{T_i^{comm}}(\alpha^{*k+1}) + \widehat{T_i^{comp}}(f^{*k+1}) - T_i), \tag{9}
$$

$$
\lambda_3^{k+1} := \lambda_3^k - \beta(\alpha_{bcfl} + N\alpha^* - 1), \tag{10}
$$

$$
\lambda_4^{k+1} := \lambda_4^k - \beta(f_{bcfl} + Nf^* - F), \tag{11}
$$

$$
\lambda_5^{k+1} := \lambda_5^k - \beta(D_{bcfl} + \widehat{D_{bcfl}} + \sum_{i=1}^N D_i - D). \tag{12}
$$

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

$$
\left\|\alpha^{*k+1} - \alpha^{*k}\right\|_2^2 \le \psi, \left\|f^{*k+1} - f^{*k}\right\|_2^2 \le \psi, \tag{13}
$$

$$
\left\|\alpha_{bcfl}^{k+1} - \alpha_{bcfl}^k\right\|_2^2 \le \psi, \left\|\alpha_{bcfl}^{k+1} - \alpha_{bcfl}^k\right\|_2^2 \le \psi, \tag{14}
$$

where ψ is the predefined threshold [24].

Note that (4) to (7) are quadratic optimization problems and can be solved easily. Due to the space limitation, 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** 3.1. 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 2-17). Specifically, we update variables and Lagrange multipliers (Lines 3-11), and calculate the stopping criteria (Line 12). If the termination condition is satisfied, then the objective function is converged (Lines 13-15). 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 18-19).

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

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'
Initialize α^*, α_{bcfl}, f^*, f_{bcfl}, λ_1, λ_2, λ_3, λ_4, λ_5
while Convergence $\neq$ True **do**
  $\alpha^{*k+1} \leftarrow$ find the optimal value of (4)
  $\alpha_{bcfl}^{k+1} \leftarrow$ find the optimal value of (5)
  $f^{*k+1} \leftarrow$ find the optimal value of (6)
  $f_{bcfl}^{k+1} \leftarrow$ find the optimal value of (7)
  $\lambda_1^{k+1} \leftarrow$ update (8)
  $\lambda_2^{k+1} \leftarrow$ update (9)
  $\lambda_3^{k+1} \leftarrow$ update (10)
  $\lambda_4^{k+1} \leftarrow$ update (11)
  $\lambda_5^{k+1} \leftarrow$ update (12)
  Calculate (13) and (14)
  if (13) and (14) are satisfied **then**
    Convergence = True
  end if
  $k \leftarrow k+1$
end while
Calculate U' via (2)
return α^*, α_{bcfl}, f^*, f_{bcfl}, U'

4 MC-ADMM based 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 $i \in \{1, 2, \cdots, N\}$, as well as α_{bcfl} and f_{bcfl}. Thus, the optimization problem in this scenario is more practical and complicated. In the following, we first introduce modified consensus ADMM (MC-ADMM), which is another form of ADMM. Then we formulate **P1** based on the MC-ADMM algorithm, and the basic idea is to separate the whole optimization task into multiple subtasks which can be resolved in a distributed manner.

4.1 Brief Introduction to MC-ADMM

At the beginning, we introduce the C-ADMM, which is one of the ADMM forms. It is designed to solve the following problem:

$$\arg\min_x \quad \sum_{i=1}^{N} f_i(x),$$

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. So, for $\sum_{i=1}^{N} f_i(x)$, we can rewrite it as:

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

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

The augmented Lagrangian is:

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

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

The updates of parameters are as:

$$x_i^{k+1} := \arg\min \left\{ \mathcal{L}(f_i(x_i), z^k, y_i^k) \right\},$$

$$z^{k+1} := \frac{1}{N}\sum_{i=1}^{N}(x_i^{k+1} + \frac{1}{\rho}y_i^k),$$

$$y_i^{k+1} := y_i^k + \rho(x_i^{k+1} - z^{k+1}).$$

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.

4.2 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 $i \in \{1, \cdots, N\}$, 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 constrains 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:

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

4.3 Solution based on MC-ADMM

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

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

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

Theorem 4.1. *The augmented Lagrangian of **P3**, i.e., $\mathcal{L}_2$, has a saddle point.*

Proof. The Hessian matrix of $\mathcal{L}_2$ is shown in (15).

Then we calculate the eigenvalues of matrix H_3 as

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

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

By applying the method proposed in [23], the updates of local variables (i.e., α_i and f_i) are:

$$\begin{aligned}
\left\{\alpha_i^{k+1}, f_i^{k+1}\right\} := \arg\min \mathcal{L}(&\alpha_i, \alpha_{bcfl}^k, f_i, f_{bcfl}^k, \\
&\theta_i^k, \epsilon_i^k, \eta_1^k, \eta_2^k, \widehat{\alpha^k}, \widehat{f^k}),
\end{aligned} \tag{16}$$

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

$$\begin{aligned}
\alpha_{bcfl}^{k+1} := \arg\min(&\mathcal{L}(\alpha_i^{k+1}, \alpha_{bcfl}, f_i^{k+1}, f_{bcfl}^{k+1}, \\
&\theta_i^k, \epsilon_i^k, \eta_1^k, \eta_2^k, \widehat{\alpha^k}, \widehat{f^k}) + \\
&\frac{\rho}{2}\beta \left\| \alpha_{bcfl} - \alpha_{bcfl}^k \right\|_2^2),
\end{aligned} \tag{17}$$

$$\begin{aligned}
f_{bcfl}^{k+1} := \arg\min(&\mathcal{L}(\alpha_i^{k+1}, \alpha_{bcfl}^{k+1}, f_i^{k+1}, f_{bcfl}, \\
&\theta_i^k, \epsilon_i^k, \eta_1^k, \eta_2^k, \widehat{\alpha^k}, \widehat{f^k}) \\
&+ \frac{\rho}{2}\beta \left\| f_{bcfl} - f_{bcfl}^k \right\|_2^2),
\end{aligned} \tag{18}$$

where $\rho, \beta > 0$ are penalty parameters.

And the updates of global variables are:

$$\widehat{\alpha^{k+1}} := \frac{1}{N} \sum_{i=1}^{N} (\alpha_i^{k+1} + \frac{\rho}{2}\theta_i^k), \tag{19}$$

$$\widehat{f^{k+1}} := \frac{1}{N} \sum_{i=1}^{N} (f_i^{k+1} + \frac{\rho}{2}\epsilon_i^k), \tag{20}$$

Besides, the updates of augmented Lagrange multipliers are:

$$\theta_i^{k+1} := \theta_i^k + \rho(\alpha_i^{k+1} - \widehat{\alpha^{k+1}}), \tag{21}$$

$$\epsilon_i^{k+1} := \epsilon_i^k + \rho(f_i^{k+1} - \widehat{f^{k+1}}), \tag{22}$$

$$\eta_1^{k+1} := \eta_1^k - \beta(T_{bcfl}^{comm}(\alpha_{bcfl}^{k+1}) + T_{bcfl}^{comp}(f_{bcfl}^{k+1}) - T_{bcfl}), \tag{23}$$

$$\eta_2^{k+1} := \eta_2^k - \beta(D_{bcfl} + \widehat{D_{bcfl}} + \sum_{i=1}^{N} D_i - D). \tag{24}$$

Lastly, the stopping criteria can be set as:

$$\left\| \alpha_i^{k+1} - \widehat{\alpha^{k+1}} \right\|_2^2 \le \psi_{prim}, \left\| f_i^{k+1} - \widehat{f^{k+1}} \right\|_2^2 \le \psi_{prim}, \tag{25}$$

$$\left\| \widehat{\alpha^{k+1}} - \widehat{\alpha^k} \right\|_2^2 \le \psi_{dual}, \left\| \widehat{f^{k+1}} - \widehat{f^k} \right\|_2^2 \le \psi_{dual}, \tag{26}$$

where ψ_{prim} and ψ_{dual} are the predefined thresholds [24]. Besides, (14) should also be included as a stopping criteria.

Even though the forms of **P2** and **P3** are different, the proof of the convergence is similar. According to Theorem 2.1, we know that U is convex, and it's 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,

$$H_3 = \begin{pmatrix} \frac{D_i log(2)(\rho+2NP_i)}{B\alpha_i^3 \log(\frac{G_i P_i}{\delta^2}+1)} & 0 & 0 & 0 \\ 0 & \frac{2D_{bcfl} log(2)(P_{bcfl}+\eta_1)}{B\alpha_{bcfl}^3 \log(\frac{G_{bcfl} P_{bcfl}}{\delta^2}+1)} & 0 & 0 \\ 0 & 0 & f_{bcfl}^2 \gamma\mu_{bcfl} + 2N\gamma\mu_i + \frac{\mu_i\rho}{f_i^3} & 0 \\ 0 & 0 & 0 & \frac{4\gamma\mu_{bcfl}}{f_{bcfl}^3} \end{pmatrix} \tag{15}$$

global variables and augmented Lagrangian multipliers (Line 1), and then we calculate the optimal decisions for each MEC task (Lines 2-19). In detail, we keep updating parameters until the objective function is converged (Lines 3-18). Then, we can calculate the optimal total energy cost and return the optimal decisions (Lines 20-21).

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, $\theta_i, \epsilon_i, \eta_1, \eta_2$
Ensure: α_i, α_{bcfl}, f_i, f_{bcfl}, U
1: Initialize α_i, α_{bcfl}, f_i, f_{bcfl}, $\theta_i, \epsilon_i, \eta_1, \eta_2, \hat{\alpha}, \hat{f}$
2: **for** $i \in \{1, 2, \cdots, N\}$ **do**
3: **while** Convergence $\neq$ True **do**
4: $\widehat{\alpha_i^{k+1}}, f_i^{k+1} \leftarrow$ find the optimal values of (16)
5: $\widehat{\alpha^{k+1}} \leftarrow$ find the optimal value of (19)
6: $\widehat{f^{k+1}} \leftarrow$ find the optimal value of (20)
7: $\alpha_{bcfl}^{k+1} \leftarrow$ find the optimal value of (17)
8: $f_{bcfl}^{k+1} \leftarrow$ find the optimal value of (18)
9: $\theta_i^{k+1} \leftarrow$ update (21)
10: $\epsilon_i^{k+1} \leftarrow$ update (22)
11: $\eta_1^{k+1} \leftarrow$ update (23)
12: $\eta_2^{k+1} \leftarrow$ update (24)
13: Calculate (14), (25) and (26)
14: **if** (14), (25) and (26) are satisfied **then**
15: Convergence = True
16: **end if**
17: $k \leftarrow k + 1$
18: **end while**
19: **end for**
20: Calculate U via (1)
21: **return** α_i, α_{bcfl}, f_i, f_{bcfl}, U

5 Experimental Evaluation

In this section, we design 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 Intel i7 processor with 32 GB RAM and 1 TB SSD.

5.1 Basic Experimental Setting

We consider a mobile edge computing scenario with one edge server and 10 local devices. For brevity, we provide Table 1 to detail the basic parameter settings in our experiments. As for the settings of some certain experiments, we will clarify them later. For the augmented Lagrange multipliers, we set them as 1.0 at the beginning.

TABLE 1: Basic Experimental Setting

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

Note that we have also tested our proposed algorithms with other parameter settings, while it can be found that the values of data and time related parameters would not affect the changing trends of the experimental results. So, we only report the results for the above parameter settings. Besides, to avoid the statistical bias, we report the averaged results for ten rounds of repeated experiments.

5.2 Experimental Results

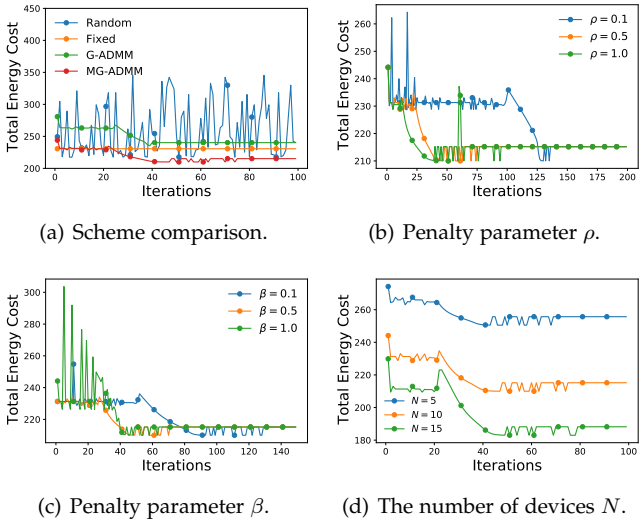
We design two parts of the experiments: the evaluation of the MG-ADMM based algorithm and the evaluation of the MC-ADMM based algorithm. These two algorithms are designed for different scenarios, i.e., homogeneous and heterogeneous. In the homogeneous scenario, we assume that all the parameters of each 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.

5.2.1 The Evaluation of the MG-ADMM based Algorithm

We first evaluate the MG-ADMM based 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. And 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. Via comparing the proposed MG-ADMM based algorithm with these three solutions, we plot the experimental results in Fig. 2(a). We can see that the MG-ADMM based algorithm can converge after about 80 rounds of iteration, while random strategy cannot converge. In addition, the random and fixed strategies and G-ADMM can only get the total energy cost larger than that of the MG-ADMM algorithm. The results show that the MG-ADMM based algorithm outperforms the other three strategies.

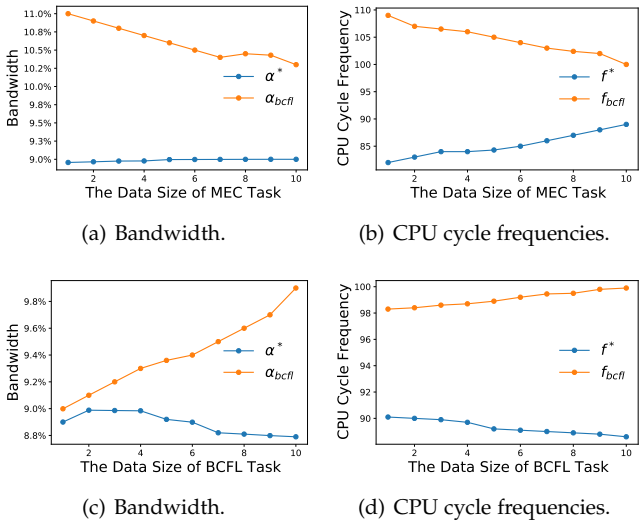
As the penalty parameters ρ and β will influence the convergence speed of the MG-ADMM based algorithm, we set the values of ρ as $\{0.1, 0.5, 1.0\}$, and maintain other



(a) Scheme comparison. (b) Penalty parameter ρ.

(c) Penalty parameter β. (d) The number of devices N.

Fig. 2: The convergence of the MMG-ADMM based algorithm.



(a) Bandwidth. (b) CPU cycle frequencies.

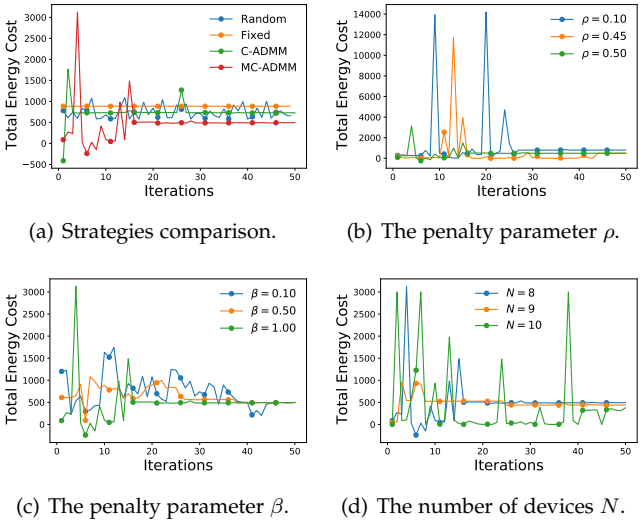
(c) Bandwidth. (d) CPU cycle frequencies.

Fig. 3: The optimal resource allocation decisions based on the MG-ADMM algorithm.

parameters unchanged. The results in Fig. 2(b) show that the faster convergence speed comes for 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 the 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., $\alpha^*, \alpha_{bcfl}, f^*, f_{bcfl}$ 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 Figs. 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 BCFL task. Similarly, we can conclude from Figs. 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.



(a) Strategies comparison. (b) The penalty parameter ρ.

(c) The penalty parameter β. (d) The number of devices N.

Fig. 4: The convergence of the MC-ADMM based algorithm.

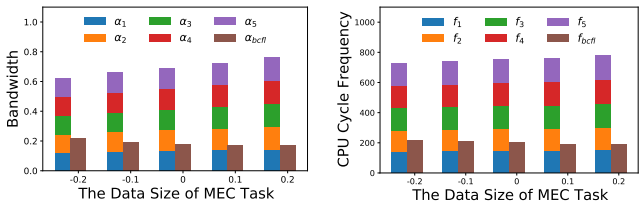
5.2.2 *The Evaluation of MC-ADMM based Algorithm*

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 $D_i \in \{1, 2, 3, \cdots, 10\}$ and $T_i \in \{1, 2, 3, \cdots, 10\}$ with $N = 10$, $\rho = 0.5$ and $\beta = 1.0$, while others are the same with the above experiments.

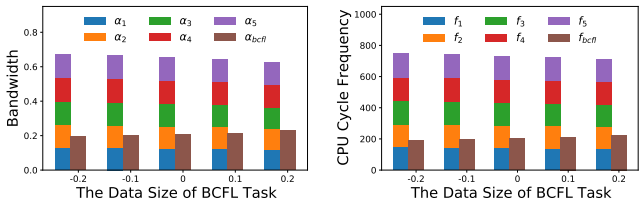
First, we compare our proposed MC-ADMM based algorithm with the above-mentioned random allocation strategy and fixed allocation strategy and C-ADMM in terms of checking the convergence speed of each strategy. Similar to the setting mentioned above regarding G-ADMM, C-ADMM is implemented by setting α_{bcfl} and f_{bcfl} as constants. 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 Figs. 4(b) and (c), we can know 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.



(a) Bandwidth. (b) CPU cycle frequencies.



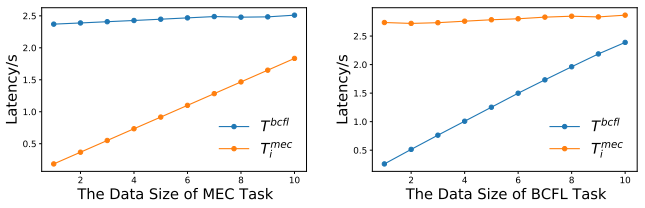
(c) Bandwidth. (d) CPU cycle frequencies.

Fig. 5: The optimal resource allocation decisions based on the MC-ADMM algorithm.

In **P3**, we have to determine α_i and f_i for each $i \in \{1, 2, 3, \cdots, N\}$, 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 know 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 refers to the original data size. From the results in Figs. 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 Figs. 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.

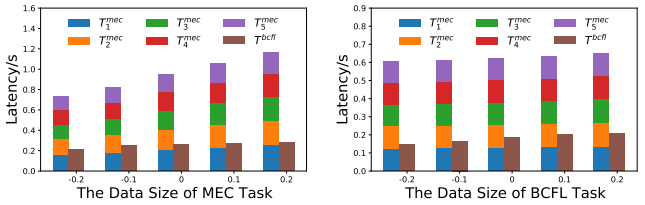
5.2.3 The Evaluation of Latency

In an ideal scenario, the MEC server would 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.



(a) Latency changes with the data size of MEC task. (b) Latency changes with the data size of BCFL task.

Fig. 6: The 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.

Fig. 7: The Latency based on the MC-ADMM algorithm.

First, we let $T_i^{mec} = T_i^{comm} + T_i^{comp}$ 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 $T^{bcfl} = T_{bcfl}^{comm} + T_{bcfl}^{comp}$ 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).

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.

6 Related Work

Recently, there are many studies focusing on deploying BCFL on the edge servers. In [14], the authors design a BCFL system running on 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.* [25] devise a blockchain-based reputation-aware fine-gained FL system to enhance the trustworthiness of devices in the MEC system. The work in [26] tries to address the privacy protection issue for BCFL in MEC via resisting a novel property inference attack, which attempts to cause the unintended property leakage. Hu *et al.* [15] 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.* [27] provide a simulation platform for BCFL in the MEC environment to measure the quality of local updates and configurations of IoT devices. 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 the minimum cost. Wang *et al.* [28] design a joint resource allocation mechanism in BCFL, which assists the participants in deciding the proper resources for completing training and mining tasks. In [29], a hybrid blockchain-assisted resource trading system is designed to achieve the decentralization and efficiency for FL in MEC. Li *et al.* [20] propose 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 the implementation of 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 edge to guarantee the service quality to both sides with the minimum cost.

7 Conclusion

In this paper, 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 the homogeneous and heterogeneous scenarios. Solid theoretic 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.

References

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

[2] B. Liang, V. Wong, R. Schober, D. Ng, and L. Wang, "Mobile edge computing," *Key technologies for 5G wireless systems*, vol. 16, no. 3, pp. 1397–1411, 2017.

[3] X. Sun and N. Ansari, "Edgeiot: Mobile edge computing for the internet of things," *IEEE Communications Magazine*, vol. 54, no. 12, pp. 22–29, 2016.

[4] D. Sabella, A. Vaillant, P. Kuure, U. Rauschenbach, and F. Giust, "Mobile-edge computing architecture: The role of mec in the internet of things," *IEEE Consumer Electronics Magazine*, vol. 5, no. 4, pp. 84–91, 2016.

[5] A. H. Sodhro, Z. Luo, A. K. Sangaiah, and S. W. Baik, "Mobile edge computing based qos optimization in medical healthcare applications," *International Journal of Information Management*, vol. 45, pp. 308–318, 2019.

[6] X. Li, X. Huang, C. Li, R. Yu, and L. Shu, "Edgecare: leveraging edge computing for collaborative data management in mobile healthcare systems," *IEEE Access*, vol. 7, pp. 22 011–22 025, 2019.

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

[8] J. Zhou, H.-N. Dai, and H. Wang, "Lightweight convolution neural networks for mobile edge computing in transportation cyber physical systems," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 10, no. 6, pp. 1–20, 2019.

[9] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial intelligence and statistics*. PMLR, 2017, pp. 1273–1282.

[10] K. Bonawitz, H. Eichner, W. Grieskamp, D. Huba, A. Ingerman, V. Ivanov, C. Kiddon, J. Konečnỳ, S. Mazzocchi, B. McMahan *et al.*, "Towards federated learning at scale: System design," *Proceedings of Machine Learning and Systems*, vol. 1, pp. 374–388, 2019.

[11] Y. Zhao, J. Zhao, L. Jiang, R. Tan, D. Niyato, Z. Li, L. Lyu, and Y. Liu, "Privacy-preserving blockchain-based federated learning for iot devices," *IEEE Internet of Things Journal*, vol. 8, no. 3, pp. 1817–1829, 2020.

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

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

[14] 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.

[15] Q. Hu, Z. Wang, M. Xu, and X. Cheng, "Blockchain and federated edge learning for privacy-preserving mobile crowdsensing," *IEEE Internet of Things Journal*, 2021.

[16] S. Sardellitti, G. Scutari, and S. Barbarossa, "Joint optimization of radio and computational resources for multicell mobile-edge computing," *IEEE Transactions on Signal and Information Processing over Networks*, vol. 1, no. 2, pp. 89–103, 2015.

[17] J. Wang, L. Zhao, J. Liu, and N. Kato, "Smart resource allocation for mobile edge computing: A deep reinforcement learning approach," *IEEE Transactions on emerging topics in computing*, 2019.

[18] 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 Communications*, vol. 26, no. 6, pp. 94–101, 2019.

[19] 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," *arXiv preprint arXiv:2004.04104*, 2020.

[20] J. Li, Y. Shao, K. Wei, M. Ding, C. Ma, L. Shi, Z. Han, and H. V. Poor, "Blockchain assisted decentralized federated learning (blade-fl): Performance analysis and resource allocation," *arXiv preprint arXiv:2101.06905*, 2021.

[21] S. Boyd, N. Parikh, and E. Chu, *Distributed optimization and statistical learning via the alternating direction method of multipliers*. Now Publishers Inc, 2011.

[22] T. D. Burd and R. W. Brodersen, "Processor design for portable systems," *Journal of VLSI signal processing systems for signal, image and video technology*, vol. 13, no. 2, pp. 203–221, 1996.

[23] B. He, M. Tao, and X. Yuan, "A splitting method for separable convex programming," *IMA Journal of Numerical Analysis*, vol. 35, no. 1, pp. 394–426, 2015.

[24] 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 Transactions on Services computing*, vol. 13, no. 2, pp. 356–367, 2019.

[25] M. H. ur Rehman, K. Salah, E. Damiani, and D. Svetinovic, "Towards blockchain-based reputation-aware federated learning," in *IEEE INFOCOM 2020-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*. IEEE, 2020, pp. 183–188.

[26] M. Shen, H. Wang, B. Zhang, L. Zhu, K. Xu, Q. Li, and X. Du, "Exploiting unintended property leakage in blockchain-assisted federated learning for intelligent edge computing," *IEEE Internet of Things Journal*, vol. 8, no. 4, pp. 2265–2275, 2020.
[27] 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 Transactions on Industrial Informatics*, 2021.
[28] Z. Wang, Q. Hu, R. Li, M. Xu, and Z. Xiong, "Incentive mechanism design for joint resource allocation in blockchain-based federated learning," *arXiv preprint arXiv:2202.10938*, 2022.
[29] S. Fan, H. Zhang, Y. Zeng, and W. Cai, "Hybrid blockchain-based resource trading system for federated learning in edge computing," *IEEE Internet of Things Journal*, vol. 8, no. 4, pp. 2252–2264, 2020.



Zhilin Wang received his B.S. from Nanchang University in 2020. He is currently pursuing his Ph.D. degree of Computer and Information Science In Indiana University-Purdue University Indianapolis (IUPUI). He is a Research Assistant with IUPUI, and he is also a reviewer of 2022 IEEE International Conference on Communications (ICC) and IEEE Access. His research interests include blockchain, federated learning, edge computing, and Internet of Things (IoT).



Qin Hu received her Ph.D. degree in Computer Science from the George Washington University in 2019. She is currently an Assistant Professor with the Department of Computer and Information Science, Indiana University-Purdue University Indianapolis (IUPUI). She has served on the Editorial Board of two journals, the Guest Editor for two journals, the TPC/Publicity Co-chair for several workshops, and the TPC Member for several international conferences. Her research interests include wireless and mobile security, edge computing, blockchain, and crowdsensing.



Zehui Xiong is currently an Assistant Professor in the Pillar of Information Systems Technology and Design, Singapore University of Technology and Design. He received the PhD degree in Nanyang Technological University, Singapore. His research interests include wireless communications, network games and economics, blockchain, and edge intelligence. He has published more than 140 research papers in leading journals and flagship conferences and many of them are ESI Highly Cited Papers. He has won over 10 Best Paper Awards in international conferences and is listed in the World's Top 2% Scientists identified by Stanford University. He is now serving as the editor or guest editor for many leading journals including IEEE JSAC, TVT, IoTJ, TCCN, TNSE, ISJ, JAS.