Pricing and Budget Allocation for IoT Blockchain with Edge Computing

Xingjian Ding, Jianxiong Guo, Deying Li, and Weili Wu, *Senior Member, IEEE*

Abstract—Attracted by the inherent security and privacy protection of the blockchain, incorporating blockchain into Internet of Things (IoT) has been widely studied in these years. However, the mining process requires high computational power, which prevents IoT devices from directly participating in blockchain construction. For this reason, edge computing service is introduced to help build the IoT blockchain, where IoT devices could purchase computational resources from the edge servers. In this paper, we consider the case that IoT devices also have other tasks that need the help of edge servers, such as data analysis and data storage. The profits they can get from these tasks is closely related to the amounts of resources they purchased from the edge servers. In this scenario, IoT devices will allocate their limited budgets to purchase different resources from different edge servers, such that their profits can be maximized. Moreover, edge servers will set "best" prices such that they can get the biggest benefits. Accordingly, there raise a pricing and budget allocation problem between edge servers and IoT devices. We model the interaction between edge servers and IoT devices as a multi-leader multi-follower Stackelberg game, whose objective is to reach the Stackelberg Equilibrium (SE). We prove the existence and uniqueness of the SE point, and design efficient algorithms to reach the SE point. In the end, we verify our model and algorithms by performing extensive simulations, and the results show the correctness and effectiveness of our designs.

Index Terms—Internet of things, Blockchain, Edge computing, Stackelberg game.

1 INTRODUCTION

In the past few decades, the Internet of Things (IoT) has been greatly developed and attracted more and more attention in academia and industry. The IoT technology helps integrate data by connecting different types of devices and has played an irreplaceable role in many fields, such as smart homes, smart factories, smart grids, and so on. In the traditional centralized IoT system, all IoT devices are connected to a centralized cloud server, which is used to manage devices and coordinate communications among devices. The most serious drawback of this centralized architecture is that it faces many problems, such as single point of failure, poor scalability, and network congestion [1]. Some studies introduce distributed IoT [2] and peer-to-peer (P2P) networks [3] to overcome these problems. However, the above studies didn't solve the inherent threats and vulnerabilities of the IoT, such as security and privacy issues [4].

A very effective way to solve the above issues is to incorporate blockchain into IoT [5]. The blockchain technology has been widely used since it was first implemented for Bitcoin in 2009 [6]. Blockchain records data as a decentralized public ledger, it does not require a third party server to store the data. Instead, data are stored in the form of blocks and maintained by all of the members of the blockchain network. The distributed feature allows blockchain to avoid suffering single point of failure which may happen in centralized systems. The blocks are linked by cryptography, and thus any change in a block will affect the subsequent blocks. The security of a blockchain mainly comes from the way that a new block is generated, which is called *mining*. To generate a new block, the members of the blockchain network need to win the competition of solving a hash puzzle which is very computation-consuming, and the winner will get a reward from the blockchain network platform. In this paper, we consider that the IoT blockchain network adopts the Proof-of-Work (PoW) consensus mechanism, as PoW has been verified on the Bitcoin system [6] for years, and the security of PoW is guaranteed. It is worth mentioning that some researchers have proposed a Directed Acyclic Graph (DAG) based blockchain which is known as *tangle* for lightweight IoT applications, such as IoTA [7]. IOTA has many advantages, such as high throughput, high concurrency, low computing power requirements, which are very suitable for many IoT scenarios. However, IOTA is not fully decentralized as it needs the "Coordinator", which is a client that sends signed messages called milestones that nodes trust and use to confirm messages[1]. Besides, IOTA has not yet been tested on large-scale transactions, so it cannot be determined whether it will come up with scalability issues. Therefore, based on security considerations, we do not adopt IOTA in this paper. Other consensus mechanisms like Proof of Stake (PoS) and Practical Byzantine Fault Tolerance

This work is supported by the National Natural Science Foundation of China (Grant NO.12071478), and partially by NSF 1907472.

X. Ding is with the Faculty of Information Technology, Beijing University of Technology, Beijing, 100124, China (e-mail: dxj@bjut.edu.cn).
J. Guo is with the BNU-UIC Institute of Artificial Intelligence and Future Networks, Beijing Normal University at Zhuhai, Zhuhai, Guangdong 519087, China, and also with the Guangdong Key Lab of AI and Multi-Modal Data Processing, BNU-HKBU United International College, Zhuhai, Guangdong 519087, China. (e-mail: jianxiongguo@bnu.edu.cn).
D. Li is with School of Information, Renmin University of China, Beijing, 100872, China (e-mail: deyingli@ruc.edu.cn).
D. Li is the corresponding author.
W. Wei is with the Department of Computer Science, Erik Jonsson School of Engineering and Computer Science, University of Texas at Dallas, Richardson, TX, 75080, USA (e-mail: weiliwu@utdallas.edu).

1. It can be seen from the documentation on the IOTA's official website: https://wiki.iota.org/learn/about-iota/an-introduction-to-iota.

(PBFT) are also not considered in our model, as PoS is affected by the Matthew effect, where the rich get richer phenomenon will happen [8], and PBFT has poor scalability and high latency [9].

As mentioned before, solving the hash puzzle is computation-consuming, so it's hard for the lightweight IoT devices to participate in the mining process. Fortunately, edge computing service is helpful for establishing an IoT blockchain [10], where IoT devices could purchase computational power from edge servers. Consider such an IoT blockchain network that contains lots of IoT systems such as smart factories or smart homes. Each IoT system can be seen as a group, and all of the groups together maintain the operation of the blockchain. The IoT blockchain network will attract nearby IoT systems to join it due to its security and privacy protection. Motivated by the reward from the blockchain network platform, the IoT devices in an IoT system will purchase the computational resource from the edge server which provides hash computing service (*hash-server*) to participate in the mining process. In addition, these IoT devices may have other tasks that require the help of the edge server which provides task processing service (*task-server*). For example, IoT devices that are used for building smart cities or realizing augmented reality (AR) need to store and process large amounts of data [11], which is very difficult for lightweight IoT devices to accomplish. Thus it's necessary for these devices to purchase resources from the task-server, so that they can perform their tasks with the help of the task-server. Generally, the more resources they purchase, the faster and better they perform their tasks, and the more profits they could get from the tasks. As IoT devices usually have limited budgets, how to allocate the budgets to purchase different resources from the two kinds of servers so as to maximize the profits, therefore, is an important problem for these IoT devices. Besides, in real scenarios, not all IoT devices have well awareness of how to maximize their profits, and they allocate their budgets based on their preferences. These IoT devices are called irrational devices, and should also be considered.

Driven by profit, edge servers will set the unit price of their resources to maximize their utilities, and accordingly, there raise a pricing and budget allocation problem between edge servers and IoT devices. We model the interaction between edge servers and IoT devices as a multi-leader multi-follower Stackelberg game, where edge servers are leaders and IoT devices are followers. The main contributions of this paper are summarized as follows.

- We introduce the IoT blockchain network with edge computing, and describe the operation of the IoT blockchain system.
- We establish a multi-leader multi-follower Stackelberg game to model the interaction between edge servers and IoT devices, in which both rational and irrational IoT devices are considered in our model. We prove that the Stackelberg equilibrium of the game exists and is unique, and then propose algorithms to find the Stackelberg equilibrium in limited interactions.
- We perform extensive simulations to validate the feasibility and effectiveness of our proposed algorithms. Simulation results show that our algorithms can quickly reach the unique Stackelberg equilibrium point compared with the baseline algorithm.

The rest of this paper is structured as follows. Section 2 introduces the related works. Section 3 describes the IoT blockchain with edge computing. Section 4 presents the Stackelberg game. Section 5 designs algorithms to get the Stackelberg equilibrium. Section 6 performs numerical simulations. And finally, Section 7 concludes this paper.

2 RELATED WORKS

Due to the inherent security and privacy protection properties of the blockchain, incorporating blockchain technology into IoT has been widely studied in recent years. Novo [12] design an architecture for scalable access management in IoT based on blockchain technology. To address the privacy and security issues in the smart grid, Gai *et al.* [13] present a permissioned blockchain edge model by combining blockchain and edge computing technologies. Guo *et al.* [14] design a blockchain-enabled energy management system to ensure the security of energy trading between the power grid and energy stations. Li *et al.* [15] propose a resource optimization for delay-tolerant data in blockchain-enabled IoT. They use the blockchain technology to improve the data security and efficiency in the IoT system. Liu *et al.* [16] propose a blockchain-based approach for the data provenance in IoT, which ensures the correctness and integrity of the query results. Qi *et al.* [17] build a compressed and data sharing framework with the help of blockchain technology, which provides efficient and private data management for industrial IoT. Lei *et al.* [18] design the *groupchain* which is a two-chain structured blockchain to ensure the scalability of the IoT services with fog computing.

There are some works that use the Stackelberg game to study the interaction among the participators in the edge computing-based blockchain network, which are closely related to our work. Chang *et al.* [19] study the incentive mechanism for edge computing-based blockchain networks, in which they aim to find the Stackelberg equilibrium between the edge service provider and the miners. Yao *et al.* [20] use a Stackelberg game to model the pricing and resource trading problem between the cloud provider and industrial IoT devices, and they find the near-optimal policy through a multiagent reinforcement learning algorithm. Xiong *et al.* [21], [22] formulate a Stackelberg game to jointly maximize the profit of mobile devices and the edge server in mobile blockchain networks. Ding *et al.* [23] investigate the interaction between the blockchain platform and IoT devices, where their objective is to find the Stackelberg equilibrium such that both the blockchain platform and IoT devices could maximize their utility and profits respectively. Guo *et al.* [24] study a Stackelberg game and double auction based task offloading scheme for mobile blockchain. However, all of these existing works only considered the computational power demand of IoT devices, and didn't consider the different budgets of each IoT devices. Moreover, these works assume that all followers have well awareness of the game, they didn't consider the influence of irrational followers in their game models, which is fundamentally different from our work.

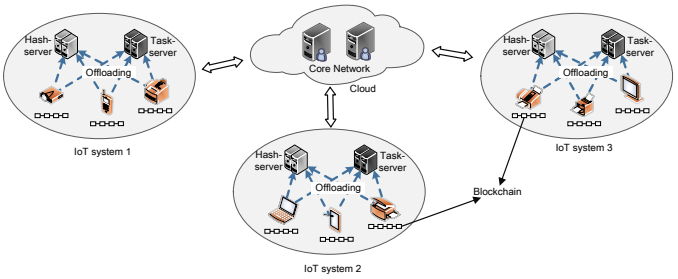


Fig. 1. The architecture of the IoT blockchain with edge computing.

3 IoT Blockchain with Edge Computing

In this section, we introduce the model of the IoT blockchain with edge computing and describe the operation of the blockchain system. Moreover, we analyze the security and reliability of the IoT blockchain network.

3.1 System Model

Fig. 1 depicts the architecture of the system model of this paper. Consider that there is an IoT blockchain network that adopts the proof-of-work consensus mechanism, and it has been running for a period of time. The IoT blockchain network consists of lots of IoT systems such as smart factories or smart homes. Each IoT system includes a set of IoT devices and can be seen as a group. Due to the security and privacy protection brought by the blockchain, the IoT blockchain network will continuously attract other IoT systems to join.

In each IoT system, there are two edge servers that provide hash computing service (*hash-server*) and task processing service (*task-server*), respectively. Motivated by the reward from the blockchain network platform, devices in the IoT system would like to be miners of the blockchain network, that is, they will compete with other miners to scramble the right of generating a new block by solving a hash puzzle. Due to the limited computational power of these devices, they will purchase computational resources from the hash-server and then offload their hash puzzle to the hash-server during the mining process. Moreover, each IoT device has its own tasks, such as data collecting, data analysis, and data processing. IoT devices could benefit from performing these tasks. However, when the amount of data is relatively large, it is difficult for these IoT devices to perform the tasks. Then IoT devices will purchase task processing resources from the task-server to perform their tasks. Generally, the more resource they purchase, the faster and better they perform their tasks, and then the more benefit they get from these tasks.

3.2 Blockchain System

3.2.1 System Initialization

Before an IoT device joins the blockchain network, it needs to register with the Authentication Server (AS) which is a trusted institution authorized by the government or the blockchain platform. For an IoT device s_n, it first selects its own identifier ID_n, and then gets its public/private key pair (PK_n, SK_n) and wallet address WAD_n from AS. The public/private key pair (PK_n, SK_n) are generated with Elliptic Curve Digital Signature Algorithm (ECDSA) asymmetric cryptography [25], and the wallet address WAD_n is generated from the public key PK_n with SHA256, RIPEMD-160 and BASE58 algorithms [26]. The AS will store the information (ID_n, PK_n, WAD_n) about each IoT device s_n.

3.2.2 Create Transactions

In the IoT blockchain network, IoT devices can trade with each other, such as purchasing or exchanging sensing data. For example, if one IoT device wants to purchase sensing data from another device, they will generate a smart contract and signing with their private key. Then the smart contract will be broadcast to the blockchain network and waiting to be packaged into a new block. Once reach a consensus, the new block will be added into the blockchain and the smart contract will be carried out automatically. Besides the trading information between devices, IoT devices also could store important and sensitive sensing data into blockchain. Both the trading records(smart contracts) and sensing data are considered as transactions.

3.2.3 Building Blocks

IoT devices collect a certain number of transactions in a period, and package them into a new block. Each block is composed of two parts: block content and block header. The block content records the detail of transactions in a Merkle tree structure. The block header consists of the Merkle tree root of all the transactions, the previous block hash value which is used as a cryptographic link that creates the chain, a version number that used for tracking for software or protocol updates, a timestamp that records the time at which the block is generated, and a nonce, which is used for solving the PoW puzzle. Denoted h_{data} by the block header excludes the nonce, the PoW puzzle is to find a nonce a such that $Hash(h_{data} + a) < difficulty$ [6], where $difficulty$ is a 256-bits binary number and is controlled by the blockchain platform to adjust the block generation speed. As the hash operation is very costly, each IoT device will offload its PoW puzzle to the hash-server. Once the puzzle is solved, the hash-server will return the result to the IoT device immediately. Note that the hash-server doesn't have any information about the block content (only knows the block header), so the hash-server cannot package a new block by itself, even if it knows the result of the PoW puzzle.

3.2.4 Carrying Out Consensus Process

The device that first solves the PoW puzzle gets the right to generate a new block, and then the new block needs to be verified by other devices to reach the consensus. By adopting the group signature and authentication scheme proposed in [27], each IoT system in the blockchain network can be seen as a group. For a new block which is generated by a device in group i, it needs to pass a two-round validation before being added in the blockchain. In the first round, the block is checked by the devices in group i, and each device will validate the transactions recorded in this block. The block will get a signature if it passes the validation from a device, and it can be broadcast to other groups for the second round validation only if it gets all the signatures

of devices in group i. In the second round, upon receiving the block, devices in other groups only check the signature attached in the block. If more than 51% of the devices agree with the block, the block reaches the consensus and will be added into the blockchain. Due to the constraints in memory, we let each IoT device only stores a certain number of the latest blocks, which is also applied in [28], [29]. The whole blockchain is stored in the monitoring nodes [29] in each group (IoT system), as the monitoring nodes are authoritative and have larger memory capacities.

3.3 Security and Reliability Analysis

Compared with traditional IoT systems, merging the IoT system into a blockchain network has many advantages, especially in terms of security and reliability. Specifically, the IoT blockchain network inherits the security and reliability performance of the blockchain technology, as shown in follows.

Distributed ledger: IoT devices carry out transactions in a P2P manner, and build trust between each other with the help of the smart contract of the blockchain. Besides, as each device has the same rights, there is no single point of failure which may cause extreme damage to the system.

Privacy protection: The communication between IoT devices is protected with the asymmetric encryption technology, so even if malicious devices intercept the message, they cannot know its content.

Integrity: The transactions of IoT devices are recorded in blocks, which are linked together through cryptography. An attacker who attempts to tamper with the transactions of IoT devices needs to dominate the majority of computation power, which is nearly impossible.

Authentication: In this IoT blockchain network, each new block needs to pass a two-round validation before it is added into the blockchain. It's very hard for an attacker to control a whole group (IoT system), so the new block that contains illegal transactions cannot pass the first round validation. Even if some attackers forge the signature of a group, the illegal block cannot pass the second round validation.

4 Multi-leader Multi-follower Stackelberg Game

Considering that a new IoT system now joins the IoT blockchain network, the IoT devices in this system will purchase resources from the edge servers to participate in the mining process and perform their tasks. Driven by profit, the hash-server and task-server will adjust the unit price of their resources to maximize their utilities. After the two servers publish their pricing strategies, each IoT device will determine its strategy for purchasing resources from the two servers according to the resource price and their budgets, such that their profits can be maximized. In this section, we first give the utility functions of the two servers and the profit function of each device, and then describe the problem to be addressed in this paper. Specifically, we model the interaction between the two servers and IoT devices as a multi-leader multi-follower Stackelberg game.

4.1 Utility Function

We assume that each IoT device has a unique budget to purchase resources from edge servers. The amount of resources they purchase from the two edge servers depends on how many profits they can get from the trading and are limited by their budgets. Each IoT device will allocate its budget to purchase different services from the hash-server and the task-server to maximize its profit. For the hash-server and the task-server, they will set the unit price of their resources to maximize their utilities. Moreover, there is competition between the two servers. For example, if the unit price of resources from the hash-server is too high, IoT devices would purchase more resources from the task-server, and vice versa. Naturally, we model the interaction between the two servers and IoT devices as a multi-leader multi-follower Stackelberg game, where the hash-server and the task-server act as leaders who first set the unit price of their resources, and IoT devices act as followers who determine their strategies according to the leaders' bids.

We use $\mathcal{S} = \{s_1, s_2, \ldots, s_n\}$ to denote the set of IoT devices, the budget for each device $s_i \in \mathcal{S}$ to purchase resources is b_i where $b_i > 0$. The unit price of resources from the hash-server and the task-server are denoted by p_h and p_t per day, respectively. Let x_i^h and x_i^t be the amount of resources purchased by s_i from the hash-server and the task-server, respectively. The amount of resources purchased buy each device is limited by its budget, that is, $x_i^h * p_h + x_i^t * p_t \leq b_i, \forall s_i \in \mathcal{S}$.

The profits of each IoT device $s_i \in \mathcal{S}$ includes two parts. The first part comes from mining new blocks for the blockchain network, which is related to the amount of resources x_i^h purchased by s_i from the hash-server. The second part comes from performing tasks, which is related to the amount of resources x_i^t purchased by s_i from the task-server. We use P_i^h and P_i^t to denote the two parts of profits, respectively. In the following, we will describe how to calculate the two parts of profits.

As the blockchain network has been running for a period, in this paper, we assume that the total hash computational power of the blockchain network in a period of time in the future can be estimated and is denoted by H [30]. In the PoW consensus, the first miner who solves the hash puzzle has the right to generate a new block and will get the reward from the blockchain network. The probability of a miner winning the mining competition is directly related to the hash computational power. We use pro_i to denote the probability that device $s_i \in \mathcal{S}$ is the first one to solve the hash puzzle, and pro_i can be estimated by

$$pro_i = \frac{x_i^h}{H + x_i^h}. \tag{1}$$

Generally, the blockchain network will adjust the difficulty of the hash puzzle periodically according to the total hash computation power in the network to stabilize the block generation speed. We assume that an average of N new blocks are generated per day, and the miners will get a reward R for generating a new block. The expected reward obtained by device s_i in per day is $pro_i RN$, and the cost is

$x_i^h p_h$. Then the expected profit that s_i gets from the mining process in a day is calculated as

$$P_i^h = pro_i RN - x_i^h p_h. \tag{2}$$

The profit P_i^t got by s_i comes from performing tasks is related to the amount of resources purchased by s_i from the task server. We use a logarithmic function to estimate the benefits of device s_i for performing tasks, and then P_i^t is calculated as

$$P_i^t = \alpha \log(1 + \beta x_i^t) - x_i^t p_t, \tag{3}$$

where $\alpha > 0$ and $\beta \geq 1$ are two constant parameters.

Therefore, the total profit got by device s_i is calculated as

$$\begin{aligned} P_i &= P_i^h + P_i^t \\ &= RN \frac{x_i^h}{H + x_i^h} - x_i^h p_h + \alpha \log(1 + \beta x_i^t) - x_i^t p_t. \end{aligned} \tag{4}$$

We use U_h and U_t to denote the utility of the hash-server and task-server, respectively. Assume that the unit hash resource cost of the hash-server is c_h, and the unit task resource cost of the task-server is c_t. Then the utilities of the two servers can be calculated as

$$U_h = \sum_{s_i \in \mathcal{S}} (p_h - c_h) x_i^h, \tag{5}$$

$$U_t = \sum_{s_i \in \mathcal{S}} (p_t - c_t) x_i^t. \tag{6}$$

The profit P_i gots by device s_i involves two parts, i.e., P_i^h and P_i^t, which comes from trading with the two servers, respectively. The first-order derivatives of P_i^h and P_i^t are

$$\frac{\partial P_i^h}{\partial x_i^h} = RN \frac{H}{(H + x_i^h)^2} - p_h, \tag{7}$$

$$\frac{\partial P_i^t}{\partial x_i^t} = \frac{\alpha\beta}{1 + \beta x_i^t} - p_t. \tag{8}$$

To make the problem reasonable, the conditions $\frac{\partial P_i^h}{\partial x_i^h}(0) \geq 0$ and $\frac{\partial P_i^t}{\partial x_i^t}(0) \geq 0$ should be hold, otherwise, IoT devices will never purchase resources from the hash-server or the task-server. Hence, we have $p_h \leq \frac{RN}{H}$ and $p_t \leq \alpha\beta$. As servers will never sell their resources at a price below the cost, that is, $p_h \geq c_h$ and $p_t \geq c_t$. Therefore, in this paper, we assume that $c_h \leq p_h \leq \frac{RN}{H}$ and $c_t \leq p_t \leq \alpha\beta$.

4.2 Problem Formulation

The interaction between the two servers and IoT devices has two stages. In the upper stage, the hash-server and the task-server offer a unit price of their resources. In the lower stage, IoT devices determine their strategies to maximize their profits according to the price of different services. In the following, we give a detailed definition of the problem in each stage.

Problem 1. *The problem in the lower stage (followers side).*

$$\max_{x_i^h, x_i^t} \quad P_i \tag{9}$$

$$s.t. \quad x_i^h p_h + x_i^t p_t \leq b_i, \tag{10}$$

$$x_i^h \geq 0, x_i^t \geq 0. \tag{11}$$

Problem 2. *The problem in the upper stage (leaders side).*

$$\max_{p_h} \quad U_h \tag{12}$$

$$s.t. \quad c_h \leq p_h \leq \frac{RN}{H}, \tag{13}$$

and

$$\max_{p_t} \quad U_t \tag{14}$$

$$s.t. \quad c_t \leq p_t \leq \alpha\beta. \tag{15}$$

Note that in the lower stage, each IoT device make their decision independently, and in the upper stage, the two servers are also non-cooperative. Therefore, the problems in the two stages form a non-cooperative multi-leader multi-follower Stackelberg game. Our objective is to find the Stackelberg equilibrium (SE) point of the game, where none of the players of the game wants to change its strategy unilaterally. The SE point in our model is defined as follows.

Definition 1. *Let $\boldsymbol{x_i^*} = \{x_i^{h*}, x_i^{t*}\}$ be a strategy of IoT device s_i, and we use $\boldsymbol{X^*} = \{\boldsymbol{x_1^*}, \boldsymbol{x_2^*}, \ldots, \boldsymbol{x_n^*}\}$ to denote the set of strategies of all of the IoT devices. Let p_h^* and p_t^* be the strategies of the hash-server and the task-server, respectively. The point $(\boldsymbol{X^*}, p_h^*, p_t^*)$ is the Stackelberg equilibrium point if the following conditions are satisfied.*

$$P_i(\boldsymbol{x_i^*}, p_h^*, p_t^*) \geq P_i(\boldsymbol{x_i}, p_h^*, p_t^*), \forall s_i \in \mathcal{S}, \tag{16}$$

$$U_h(p_h^*, p_t^*) \geq U_h(p_h, p_t^*), \tag{17}$$

$$U_t(p_t^*, p_h^*) \geq U_t(p_t, p_h^*), \tag{18}$$

where $\boldsymbol{x_i} = \{x_i^h, x_i^t\}$ is an arbitrary feasible strategy for any device $s_i \in \mathcal{S}$, p_h and p_t are arbitrary feasible strategies for the hash-server and the task-server, respectively.

5 Solution of the Multi-Leader Multi-Follower Stackelberg Game

In this section, we analyze the existence and uniqueness of the Stackelberg equilibrium point of the multi-leader multi-follower Stackelberg game. We first analyze the lower stage of the game, in which we consider both rational and irrational players (followers). The rational players always try to maximize their profits by optimally allocating their budgets, while the irrational players lack awareness of the game and allocate their budgets based on their preferences. Then we analyze the upper stage of the game, where the two servers determine their pricing strategies to maximize their utilities.

5.1 Lower stage (followers side) analysis

5.1.1 Strategies for rational followers

The rational followers always decide their buying strategies based on their profits. In the following, we will describe how to find the best strategy for each rational follower.

The second-order derivatives of the profits function P_i of device s_i are

$$\frac{\partial^2 P_i}{\partial (x_i^h)^2} = \frac{-2RN}{(H + x_i^h)^3} < 0, \tag{19}$$

$$\frac{\partial^2 P_i}{\partial (x_i^t)^2} = \frac{-\alpha\beta^2}{(1+\beta x_i^t)^2} < 0, \tag{20}$$

$$\frac{\partial^2 P_i}{\partial x_i^h \partial x_i^t} = 0. \tag{21}$$

Therefore, the profits function P_i is strictly concave, and the problem for each device s_i in the lower stage is actually a convex optimization problem. Sequentially, we use the Karush–Kuhn–Tucker (KKT) conditions to solve the problem.

Let λ_1, λ_2 and λ_3 be the Lagrange's multipliers that associated with conditions in Eqs. (43) and (44). Then we define the Lagrangian function as follows.

$$L_i = P_i + \lambda_1(b_i - x_i^h p_h - x_i^t p_t) + \lambda_2 x_i^h + \lambda_3 x_i^t. \tag{22}$$

The KKT conditions (including four groups of conditions) of Problem 1 are listed as follows.

Stationarity conditions:

$$\frac{\partial L_i}{\partial x_i^h} = RN\frac{H}{(H+x_i^h)^2} - p_h - \lambda_1 p_h + \lambda_2 = 0, \tag{23}$$

$$\frac{\partial L_i}{\partial x_i^t} = \frac{\alpha\beta}{1+\beta x_i^t} - p_t - \lambda_1 p_t + \lambda_3 = 0. \tag{24}$$

Primal feasibility conditions:

$$b_i - x_i^h p_h - x_i^t p_t \geq 0, \tag{25}$$

$$x_i^h, x_i^t \geq 0. \tag{26}$$

Dual feasibility conditions:

$$\lambda_1, \lambda_2, \lambda_3 \geq 0. \tag{27}$$

Complementary slackness conditions:

$$\lambda_1(b_i - x_i^h p_h - x_i^t p_t) = 0, \tag{28}$$

$$\lambda_2 x_i^h = 0, \tag{29}$$

$$\lambda_3 x_i^t = 0. \tag{30}$$

The optimal solution of the problem is taken in one of the following four cases.

(1) Case 1: $x_i^h = x_i^t = 0$. According to the KKT condition (28), we have $\lambda_1 = 0$ as $b_i > 0$. Substitute it into KKT conditions (23) and (24), we have $\lambda_2 = p_h - \frac{RN}{H}$ and $\lambda_3 = p_t - \alpha\beta$. As $p_h \leq \frac{RN}{H}$ and $p_t \leq \alpha\beta$ hold, we have $\lambda_2 \leq 0$ and $\lambda_3 \leq 0$. The KKT conditions can only be satisfied when $\lambda_2 = 0$ and $\lambda_3 = 0$. Thus we need to check whether $p_h = \frac{RN}{H}$ and $p_t = \alpha\beta$ hold, if yes, the optimal solution is $x_i^h = x_i^t = 0$, otherwise, the optimal solution is not in this case.

(2) Case 2: $x_i^h = 0$ and $x_i^t > 0$. In this case, we have $\lambda_3 = 0$ according to the KKT condition (30).

Case 2-1: Consider $\lambda_1 = 0$, substitute $x_i^h = 0$ and $\lambda_1 = 0$ into (23), we have $\lambda_2 = p_h - \frac{RN}{H} \leq 0$. If $p_h < \frac{RN}{H}$, the KKT condition (27) cannot be satisfied as $\lambda_2 < 0$, which means $\lambda_1 = 0$ is not feasible in this case. Otherwise, if $p_h = \frac{RN}{H}$, we have $\lambda_2 = 0$. Substitute $\lambda_1 = \lambda_3 = 0$ into (24), we have $x_i^t = \frac{\alpha}{p_t} - \frac{1}{\beta}$. Then we need to check whether the primal feasibility condition (25) is satisfied, if yes, the optimal solution is $(0, \frac{\alpha}{p_t} - \frac{1}{\beta})$, if no, $\lambda_1 = 0$ is not feasible in this case.

Case 2-2: Consider $\lambda_1 > 0$, according to (28), we have $b_i - x_i^h p_h - x_i^t p_t = 0$. Combining $x_i^h = 0$, we have

$$x_i^t = \frac{b_i}{p_t}. \tag{31}$$

Substitute (31) and $\lambda_3 = 0$ into KKT condition (24), we have

$$\lambda_1 = \frac{\alpha\beta}{\beta b_i + p_t} - 1. \tag{32}$$

Substitute (32) and $x_i^h = 0$ into (23), we have

$$\lambda_2 = \frac{\alpha\beta p_h}{\beta b_i + p_t} - \frac{RN}{H}. \tag{33}$$

It obvious that $x_i^t = \frac{b_i}{p_t} > 0$ is satisfied. If λ_1 got by (32) satisfies $\lambda_1 > 0$ and λ_2 got by (33) satisfies $\lambda_2 \geq 0$, it means all of the KKT conditions can be satisfied, and thus the optimal solution can be determined as $(0, \frac{b_i}{p_t})$. Otherwise, the optimal solution is not in Case 2.

(3) Case 3: $x_i^h > 0$ and $x_i^t = 0$. In this case, we have $\lambda_2 = 0$ according to the KKT condition (29).

Case 3-1: Consider $\lambda_1 = 0$, substitute $x_i^t = 0$ and $\lambda_1 = 0$ into (24), we have $\lambda_3 = p_t - \alpha\beta \leq 0$. If $p_t < \alpha\beta$, the KKT condition (27) cannot be satisfied as $\lambda_3 < 0$, which implies $\lambda_1 = 0$ is not feasible in this case. Otherwise, if $p_t = \alpha\beta$, we have $\lambda_3 = 0$. Substitute $\lambda_1 = \lambda_2 = 0$ into (23), we have $x_i^h = \sqrt{\frac{RNH}{p_h}} - H$. Then we need to check whether the primal feasibility condition (25) is satisfied, if yes, the optimal solution is $(\sqrt{\frac{RNH}{p_h}} - H, 0)$, if no, $\lambda_1 = 0$ is not feasible in this case.

Case 3-2: Consider $\lambda_1 > 0$, we have $b_i - x_i^h p_h - x_i^t p_t = 0$ according to (28). Combining $x_i^t = 0$, we have

$$x_i^h = \frac{b_i}{p_h}. \tag{34}$$

Substitute (34) and $\lambda_2 = 0$ into KKT condition (23), we have

$$\lambda_1 = \frac{RNHp_h}{(b_i + Hp_h)^2} - 1. \tag{35}$$

Substitute (35) and $x_i^t = 0$ into (24), we have

$$\lambda_3 = \frac{RNHp_h p_t}{(b_i + Hp_h)^2} - \alpha\beta. \tag{36}$$

It's obvious that $x_i^h = \frac{b_i}{p_h} > 0$ is satisfied. If λ_1 got by (35) satisfies $\lambda_1 > 0$ and λ_3 got by (36) satisfies $\lambda_3 \geq 0$, it means that all of the KKT conditions can be satisfied, and thus the optimal solution is $(\frac{b_i}{p_h}, 0)$. Otherwise, the optimal solution is not in Case 3.

(4) Case 4: $x_i^h > 0$ and $x_i^t > 0$. In this case, we have $\lambda_2 = \lambda_3 = 0$ according to the KKT conditions (29) and (30).

Case 4-1: Suppose $\lambda_1 = 0$, substitute it into the stationarity conditions (23) and (24), we have

$$x_i^h = \sqrt{\frac{RNH}{p_h}} - H; x_i^t = \frac{\alpha}{p_t} - \frac{1}{\beta}. \tag{37}$$

As $p_h \leq \frac{RN}{H}$ and $p_t \leq \alpha\beta$, it's easy to know that the condition (26) is satisfied. Then we check whether the condition (25) is satisfied. If yes, then the solution shown in Eq. (37) is the optimal solution. Otherwise, $\lambda_1 = 0$ is not feasible in this case.

Algorithm 1 Find Optimal Strategy for a Device s_i. (FOSD)

Input: H, R, N, α, β, b_i, p_h and p_t;
Output: The optimal strategy (x_i^h, x_i^t) for IoT device s_i;
// **Case 1:**
if $p_h = \frac{RN}{H}$ && $p_t = \alpha\beta$ **then**
  return $(0, 0)$;
end if
// **Case 2-1:**
$x_i^t = \frac{\alpha}{p_t} - \frac{1}{\beta}$;
if $p_h = \frac{RN}{H}$ && $b_i - x_i^t p_t \geq 0$ **then**
  return $(0, x_i^t)$;
end if
// **Case 2-2:**
$x_i^t = \frac{b_i}{p_t}$; $\lambda_1 = \frac{\alpha\beta}{\beta b_i + p_t} - 1$; $\lambda_2 = \frac{\alpha\beta p_h}{\beta b_i + p_t} - \frac{RN}{H}$;
if $\lambda_1 > 0$ && $\lambda_2 \geq 0$ **then**
  return $(0, x_i^t)$;
end if
// **Case 3-1:**
$x_i^h = \sqrt{\frac{RNH}{p_h}} - H$;
if $p_t = \alpha\beta$ && $b_i - x_i^h p_h \geq 0$ **then**
  return $(x_i^h, 0)$;
end if
// **Case 3-2:**
$x_i^h = \frac{b_i}{p_h}$; $\lambda_1 = \frac{RNHp_h}{(b_i + Hp_h)^2} - 1$; $\lambda_3 = \frac{RNHp_h p_t}{(b_i + Hp_h)^2} - \alpha\beta$;
if $\lambda_1 > 0$ && $\lambda_3 \geq 0$ **then**
  return $(x_i^h, 0)$;
end if
// **Case 4-1:**
$x_i^h = \sqrt{\frac{RNH}{p_h}} - H$; $x_i^t = \frac{\alpha}{p_t} - \frac{1}{\beta}$;
if $b_i - x_i^h p_h - x_i^t p_t \geq 0$ **then**
  return (x_i^h, x_i^t);
end if
// **Case 4-2:**
$A = b_i + Hp_h + \frac{1}{\beta}p_t$, $B = \sqrt{RNHp_h}$;
$\lambda_1 = \left(\frac{B + \sqrt{B^2 + 4A\alpha}}{2A}\right)^2 - 1$;
if $\lambda_1 > 0$ **then**
  $x_i^h = \sqrt{\frac{RNH}{p_h + \lambda_1 p_h}} - H$; $x_i^t = \frac{\alpha}{p_t + \lambda_1 p_t} - \frac{1}{\beta}$;
  if $x_i^h > 0$ && $x_i^t > 0$ **then**
    return (x_i^h, x_i^t);
  end if
end if

Case 4-2: Now we consider the case that $\lambda_1 > 0$. According to the condition (28), we have $b_i - x_i^h p_h - x_i^t p_t = 0$. Solving conditions (23) and (24), we have

$$x_i^h = \sqrt{\frac{RNH}{p_h + \lambda_1 p_h}} - H; x_i^t = \frac{\alpha}{p_t + \lambda_1 p_t} - \frac{1}{\beta}. \tag{38}$$

Substitute (38) into $b_i - x_i^h p_h - x_i^t p_t = 0$, we have

$$A - B\sqrt{\frac{1}{1 + \lambda_1}} - \frac{\alpha}{1 + \lambda_1} = 0. \tag{39}$$

where $A = b_i + Hp_h + \frac{1}{\beta}p_t > 0$, and $B = \sqrt{RNHp_h} > 0$. Let $t = \sqrt{1 + \lambda_1} > 0$, substitute t into (39) and solve the function, we have $t = \frac{B \pm \sqrt{B^2 + 4A\alpha}}{2A}$. As $t > 0$, we have

$$t = \sqrt{1 + \lambda_1} = \frac{B + \sqrt{B^2 + 4A\alpha}}{2A}. \tag{40}$$

By solving (40), we have

$$\lambda_1 = \left(\frac{B + \sqrt{B^2 + 4A\alpha}}{2A}\right)^2 - 1. \tag{41}$$

Then we check whether λ_1 got by (41) satisfies $\lambda_1 > 0$. If $\lambda_1 \leq 0$, it means that the solutions found in (38) cannot satisfy all of the KKT conditions, and thus the optimal solution is not in Case 4. Otherwise, we substitute (41) into (38), and we will get a solution (x_i^h, x_i^t). Then we check whether the solution (x_i^h, x_i^t) satisfies $x_i^h > 0$ and $x_i^t > 0$. If yes, the solution (x_i^h, x_i^t) is the optimal solution. If no, the optimal solution is not in Case 4.

The optimal solution will be found in one of the four cases from Case 1 to Case 4. Based on the above analysis, we present the algorithm FOSD to solve Problem 1 in the lower stage. The pseudo-code is shown in Algorithm 1.

5.1.2 Strategies for irrational followers

In practice, some game players may not follow the above method to maximize their profit. We name these players in the lower stage of the game as irrational followers. The irrational followers include the players who lack awareness of the game and thus could not make the best decisions, as well as the players who have individual preferences. For example, some players may be very confident about the blockchain network, and they believe that they could get much more profits from the blockchain network platform in the future. So they are more inclined to allocate their budgets to purchase computational resources from the hash-server to participate in the block mining. On the contrary, some players may not be optimistic about the blockchain network or even not interested at all, and thus they will not participate in the block mining. In this subsection, we consider the above two kinds of irrational followers with individual preferences, and term them as *over-confident followers* and *lack of confidence followers*, respectively.

The over-confident followers are fanatics of the blockchain network. For example, in the Bitcoin network, some people are willing to spend a lot of money to buy mining machines to participate in block mining in order to gain benefits. In our work, we consider that the over-confident followers will allocate all their budgets to purchase computational resources from the hash-server to participate in the block mining. Thus for each s_i of the over-confident followers, we have $x_i^h = \frac{b_i}{p_h}$, and $x_i^t = 0$.

The lack of confidence followers only consider how to allocate their budgets to purchase task processing resources from the task-server to maximize their profits, i.e., $x_i^h = 0$ always holds for each s_i of the lack of confidence followers. Thus the problem in the lower stage (Problem 1) for these followers (devices) is changed to

$$\max_{x_i^t} \quad P_i^t \tag{42}$$

$$s.t. \quad x_i^t p_t \leq b_i, \tag{43}$$

$$x_i^t \geq 0. \tag{44}$$

The second-order derivative of the profit function P_i^t of device s_i is

$$\frac{\partial^2 P_i^t}{\partial (x_i^t)^2} = \frac{-\alpha\beta^2}{(1 + \beta x_i^t)^2} < 0. \tag{45}$$

Therefore, the above problem is to find the maximum value for a univariate concave function. The maximum value point is either where the first derivative of P_i^t is equal to 0, or the end point of the domain. According to Equation (8), when the first derivative of P_i^t is equal to 0, we have $x_i^t = \frac{\alpha}{p_t} - \frac{1}{\beta}$. According to the budget limitation, we have $x_i^t \geq \frac{b_i}{p_t}$. Therefore, for each lack of confidence follower s_i, its strategy will be $x_i^h = 0$, $x_i^t = \min\{\frac{\alpha}{p_t} - \frac{1}{\beta}, \frac{b_i}{p_t}\}$.

5.2 Upper stage (leaders side) analysis

On the leaders' side, hash-server and task-server set their unit prices p_h and p_t of resources to maximize their utilities U_h and U_t which are calculated by Eqs. (5) and (6), respectively. Note that the pricing strategies of leaders will directly affect the strategies of followers, which has been analyzed in Section 5.1. Therefore, the strategies of the hash-server and task-server will affect each other's utility. The game between the two servers is non-cooperative and competitive. To maximize their utilities, each server (leader) should give a suitable unit price of its resource. For example, for the hash-server, if the unit price of resource p_h is too high, the rational followers will prefer to purchase more resources from the task-server, and thus the hash-server will get a very low utility. On the contrary, if p_h is set too low, although the rational followers tend to purchase resources from the hash-server, the total purchased resources are limited due to the budget limitation of these followers, which also results in a low utility. In the following, we will show that the game between the two servers will reach a Nash equilibrium, and we design an algorithm to find the Nash equilibrium point.

The concept of the Nash Equilibrium (NE) of the game between the two servers is defined as follows.

Definition 2. *Let p_h^* and p_t^* be the strategies of the hash-server and the task-server, respectively, (p_h^*, p_t^*) is the Nash equilibrium if the following conditions are satisfied.*

$$U_h(p_h^*, p_t^*) \geq U_h(p_h, p_t^*), \tag{46}$$
$$U_t(p_t^*, p_h^*) \geq U_t(p_t, p_h^*), \tag{47}$$

where p_h and p_t are arbitrary feasible strategies for the hash-server and the task-server, respectively.

According to the definition, at the NE point, none of the servers can improve its utility by unilaterally changing its strategy. Therefore, when the game reaches a NE point, the interaction between the two servers is suspended, and the pricing strategies of the two servers will never change again. Next, we will prove the existence and uniqueness of the NE point of the game between the two servers.

Theorem 1. *The NE point of the game between the two servers exists and is unique.*

Proof. As defined in Problem 2, the strategy space of the two servers is $[c_h, \frac{RN}{H}] \times [c_t, \alpha\beta]$, which is a non-empty, closed and convex subset of the Euclidean space. Next, we calculate the second order derivatives of utility functions $U_h(\cdot)$ and $U_t(\cdot)$.

We first consider the utility of the hash-server. According analysis in subsection 5.1.1, the amount of purchased resources x_i^h of each rational device s_i from the hash-server must be one of the four following cases: (1) $x_i^h = \sqrt{\frac{RNH}{p_h}} - H$; (2) $x_i^h = \sqrt{\frac{RNH}{p_h + \lambda_1 p_h}} - H$, where $\lambda_1 =$ (41); (3) $x_i^h = 0$; (4) $x_i^h = \frac{b_i}{p_h}$. Then the first order derivative of $x_i^h(\cdot)$ with respect to p_h of these four cases is calculated as:

$$\frac{\partial x_i^h}{\partial p_h} = -\frac{1}{2}\sqrt{RNH}\,(p_h)^{-\frac{3}{2}} \tag{48}$$
$$= -\frac{1}{2}\sqrt{\frac{RNH}{1+\lambda_1}}\,(p_h)^{-\frac{3}{2}}, \lambda_1 = (41) \tag{49}$$
$$= 0 \tag{50}$$
$$= -b_i(p_h)^{-2}. \tag{51}$$

The second order derivative of $x_i^h(\cdot)$ with respect to p_h of these four cases is calculated as:

$$\frac{\partial^2 x_i^h}{\partial (p_h)^2} = \frac{3}{4}\sqrt{RNH}\,(p_h)^{-\frac{5}{2}} \tag{52}$$
$$= \frac{3}{4}\sqrt{\frac{RNH}{1+\lambda_1}}\,(p_h)^{-\frac{5}{2}}, \lambda_1 = (41) \tag{53}$$
$$= 0 \tag{54}$$
$$= 2b_i(p_h)^{-3}. \tag{55}$$

As analyzed in subsection 5.1.2, for each over-confident device s_i, we always have $x_i^h = \frac{b_i}{p_h}$, and thus the second order derivative of $x_i^h(\cdot)$ with respect to p_h is equal to Equation (55). For each lack of confidence device s_i, $x_i^h = 0$ and $\frac{\partial^2 x_i^h}{\partial (p_h)^2} = 0$.

According to function (5), the second order derivative of $U_h(\cdot)$ with respect to p_h is

$$\frac{\partial^2 U_h}{\partial (p_h)^2} = \sum_{s_i \in \mathcal{S}} \left(2\frac{\partial x_i^h}{\partial p_h} + (p_h - c_h)\frac{\partial^2 x_i^h}{\partial (p_h)^2} \right). \tag{56}$$

As $p_h - c_h < p_h$, combining the functions of $\frac{\partial x_i^h}{\partial p_h}$ and $\frac{\partial^2 x_i^h}{\partial (p_h)^2}$, we have $\frac{\partial^2 U_h}{\partial (p_h)^2} \leq 0$. Therefore, the utility function $U_h(\cdot)$ of the hash-server is a concave function with respect to p_h.

Similarly, we have $\frac{\partial^2 U_t}{\partial (p_t)^2} \leq 0$, and we know that the utility function $U_t(\cdot)$ of the task-server is a concave function with respect to p_t. Thus the interaction between the two servers forms a concave 2-person game. According to [31], we know that the NE point of the game between the two servers exists and is unique. □

After the leaders (the two servers) issue their strategies, the followers (IoT devices) will determine their strategies for purchasing different resources from the two servers, as is discussed in Section 5.1. The interaction between servers and IoT devices formulates a multi-leader multi-follower Stackelberg game, and the objective is to find the Stackelberg equilibrium (SE) point of the game, which is defined as Definition 1. Next, we will prove that the Stackelberg equilibrium of the game between servers and IoT devices exists and is unique.

Algorithm 2 Find Nash Equilibrium for Servers. (FNES)

Input: The game between the two servers and IoT devices;
Output: The pricing strategy (p_h, p_t) for the two servers;

Initialize: $p_h = \frac{1}{2}(c_h + \frac{RN}{H})$, $p_t = \frac{1}{2}(c_t + \alpha\beta)$;
Set two small steps Δ_h and Δ_t, and two attenuation coefficient δ_{slow} and δ_{fast} of steps;
while true **do**
  $p'_h = p_h$, $p'_t = p_t$;
  // **Adjust strategy for the hash-server**
  Calculate $U_h(p_h, p_t)$, $U_h(p_h + \Delta_h, p_t)$ and $U_h(p_h - \Delta_h, p_t)$ by invoking Algorithm FOSD and analysis in subsection 5.1.2 to get strategies for rational and irrational devices;
  if $U_h(p_h + \Delta_h, p_t) \geq U_h(p_h, p_t)$ && $U_h(p_h + \Delta_h, p_t) \geq U_h(p_h - \Delta_h, p_t)$ **then**
    $p_h = \min\{p_h + \Delta_h, \frac{RN}{H}\}$;
  else if $U_h(p_h - \Delta_h, p_t) \geq U_h(p_h, p_t)$ && $U_h(p_h - \Delta_h, p_t) \geq U_h(p_h + \Delta_h, p_t)$ **then**
    $p_h = \max\{p_h - \Delta_h, c_h\}$;
  end if
  // **Reduce the step for the hash-server**
  if $p'_h == p_h$ **then**
    $\Delta_h = \delta_{fast} \cdot \Delta_h$;
  else
    $\Delta_h = \delta_{slow} \cdot \Delta_h$;
  end if
  // **Adjust strategy for the tash-server**
  Calculate $U_t(p_t, p_h)$, $U_t(p_t + \Delta_t, p_h)$ and $U_t(p_t - \Delta_t, p_h)$ by invoking Algorithm FOSD and analysis in subsection 5.1.2 to get strategies for rational and irrational devices;
  if $U_t(p_t + \Delta_t, p_h) \geq U_t(p_t, p_h)$ && $U_t(p_t + \Delta_t, p_h) \geq U_t(p_t - \Delta_t, p_h)$ **then**
    $p_t = \min\{p_t + \Delta_t, \alpha\beta\}$;
  else if $U_t(p_t - \Delta_t, p_h) \geq U_t(p_t, p_h)$ && $U_t(p_t - \Delta_t, p_h) \geq U_t(p_t + \Delta_t, p_h)$ **then**
    $p_t = \max\{p_t - \Delta_t, c_t\}$;
  end if
  // **Reduce the step for the task-server**
  if $p'_t == p_t$ **then**
    $\Delta_t = \delta_{fast} \cdot \Delta_t$;
  else
    $\Delta_t = \delta_{slow} \cdot \Delta_t$;
  end if
end while
return (p_h, p_t)

Theorem 2. *The SE point of the game between servers and IoT devices exists and is unique.*

Proof. As analyzed in subsection 5.1.1, each rational IoT device will find its optimal strategy in one of the three cases from Case 2 to Case 4, which indicates that the strategy of each rational IoT device is unique after the two servers give their pricing strategies. We consider two kinds of irrational devices as described in subsection 5.1.2, and each irrational device also has a unique strategy after the two servers give their pricing strategies. According to Theorem 1, the game between the two servers has a unique NE point. Thus we can conclude that the SE point of the game between servers and IoT devices exists and is unique. □

To find the NE point of the game between the two servers, we propose an algorithm named FNES to find the final pricing strategies of the two servers. FNES is an improved version based on the sub-gradient technique [32], [33], in which we set two steps for the two servers and set two different attenuation coefficients, so that the algorithm can converge more quickly. The pseudo-code of algorithm FNES is shown in Algorithm 2.

In FNES, we first set a feasible pricing strategy for each server, and two small steps Δ_h and Δ_t are set to update the strategies of two servers, respectively. We iteratively adjust the pricing strategy for each server in turn. For the hash-server, we will calculate its utility U_h with pricing strategies p_h, $p_h + \Delta_h$, and $p_h - \Delta_h$, and the best pricing strategy will be selected as the latest strategy in the next round. Note that the value of U_h is related to the strategy of each IoT device, and thus we need to invoke the FOSD algorithm as well as the analysis for irrational followers to get the strategy of each IoT device. If pricing strategy p_h is changed in a certain iteration, we will update the step Δ_h with the attenuation coefficient δ_{slow}, where δ_{slow} is a real number that is close to (and less than) 1, such as 0.99. Otherwise, we will update the step Δ_h with the attenuation coefficient δ_{fast}, where δ_{fast} is a smaller positive real number, such as 0.5. The reason that we set two different attenuation coefficients is to speed up the convergence rate of our algorithm. The strategy adjustment of the task-server is similar to that of the hash-server, where we use Δ_t to control the step of the pricing strategy. The algorithm terminates when none of the servers will change its pricing strategy, and the two steps Δ_h and Δ_t are sufficiently small.

6 Simulations

In this section, we conduct numerical experiments to validate the feasibility and effectiveness of our algorithms.

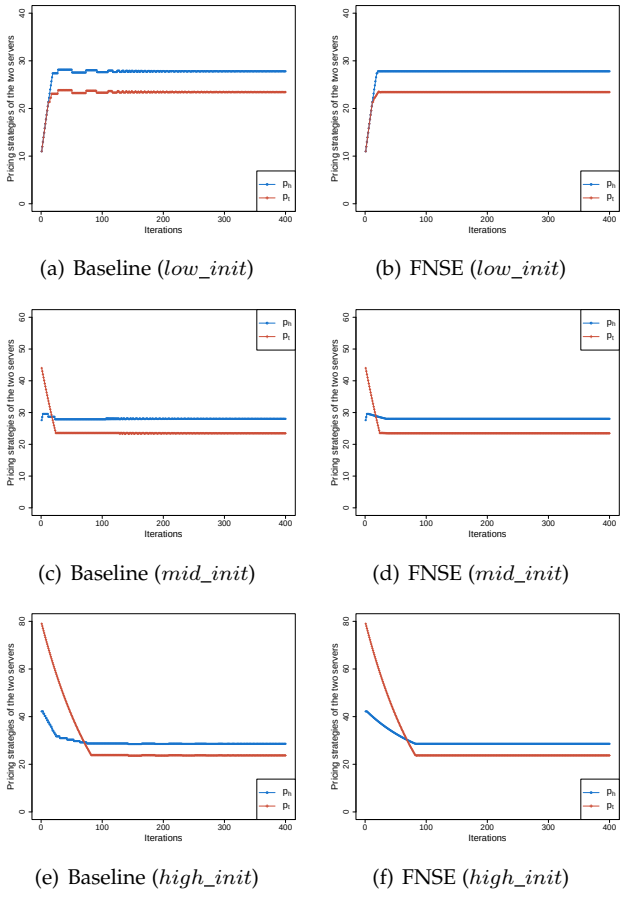
6.1 Experimental settings

In the experiments, we assume the total hash computational power H of the IoT blockchain network in the next period is estimated to be 10^4 GH/s. The mining reward R from the blockchain platform is set to be 300, and the number of generated new blocks per day N is set to be 1440. For the profit function of performing tasks, we set α to be 40 and β to be 2. The unit resource cost of the hash-server and the task-server is set to be 10, that is, $c_h = c_t = 10$. We consider there are 500 IoT devices in the IoT system that would like to purchase resources from the two servers, in which 80% of them are rational followers, 10% of them are over-confident followers, and 10% of them are lack of confidence followers. The budget of each device is randomly chosen in $[20, 90]$. For the algorithm FNES, we set the set $\Delta_h = \Delta_t = 1$, $\delta_{slow} = 0.99$ and $\delta_{fast} = 0.5$. Unless other declared, the above are the default settings of our experiments.

6.2 Results and Analyses

6.2.1 *The convergence of algorithm* FNES

To validate the efficiency of our algorithm, we adopt the sub-gradient method that is used in [33] as the baseline, in

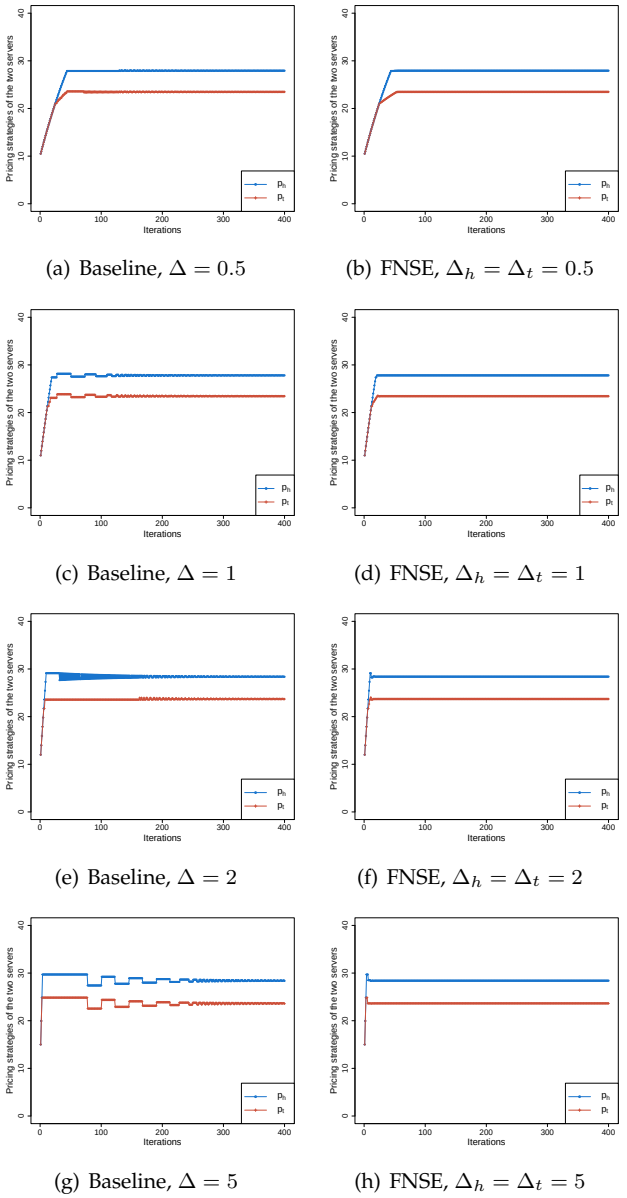


(a) Baseline (low_init) (b) FNSE (low_init)

(c) Baseline (mid_init) (d) FNSE (mid_init)

(e) Baseline ($high\_init$) (f) FNSE ($high\_init$)

Fig. 2. The convergence process of the baseline and FNES with different initializations.



(a) Baseline, $\Delta = 0.5$ (b) FNSE, $\Delta_h = \Delta_t = 0.5$

(c) Baseline, $\Delta = 1$ (d) FNSE, $\Delta_h = \Delta_t = 1$

(e) Baseline, $\Delta = 2$ (f) FNSE, $\Delta_h = \Delta_t = 2$

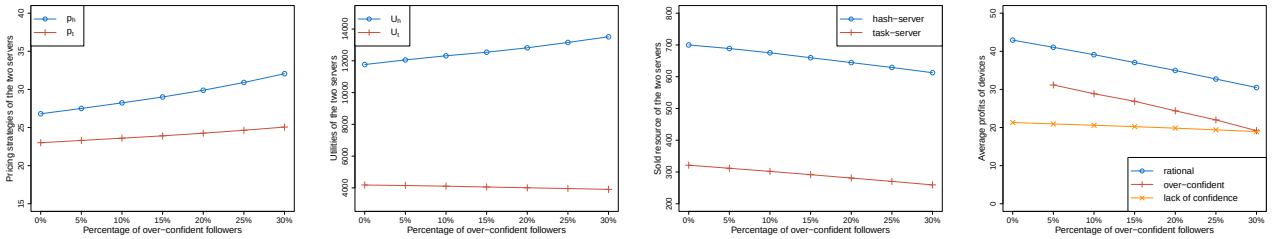
(g) Baseline, $\Delta = 5$ (h) FNSE, $\Delta_h = \Delta_t = 5$

Fig. 3. The convergence process of the baseline and FNES with different steps.

which we set the step Δ to be 1 and the attenuation coefficient is set to be 0.99. Both our algorithm and the baseline need to set initial values of the pricing strategies for the two servers. In our simulation, we set three different initializations named low_init, mid_init and $high\_init$. Specifically, $low\_init = (p_h, p_t) = (c_h, c_t) = (10, 10)$, $mid\_init = (p_h, p_t) = (\frac{1}{2}(c_h + \frac{RN}{H}), \frac{1}{2}(c_t + \alpha\beta)) = (26.6, 45)$, and $high\_init = (p_h, p_t) = (\frac{RN}{H}, \alpha\beta) = (43.2, 80)$.

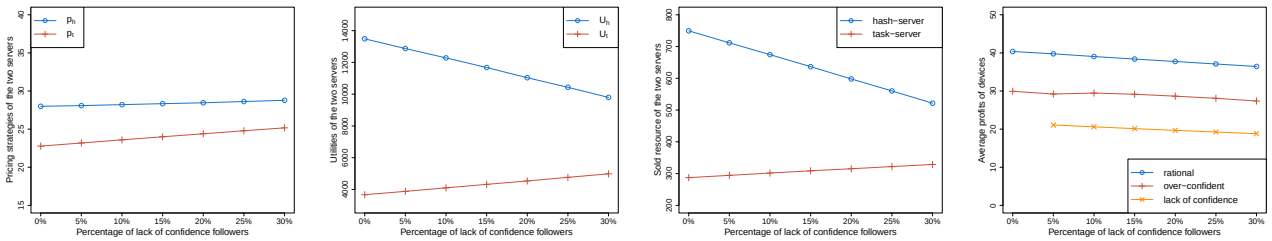
Fig. 2 shows the convergence process of the baseline and FNES with different initializations. We can see that regardless of the initialization settings, our algorithm and the baseline algorithm always reach the same Nash Equilibrium, which implies the correctness of our designs. Besides, we can see that our algorithms always outperform the baseline algorithm. Although the baseline algorithm can quickly close to the Nash Equilibrium, it will oscillate around the equilibrium point, and needs more iterations to stabilize to the equilibrium point. In comparison, our algorithm reaches the equilibrium point very smoothly. The reason is that the baseline algorithm needs more iterations to wait for the step Δ to decay to a sufficiently small level. When we set the initialization as low_init, as shown in Fig. 2(a) and Fig. 2(b), the baseline algorithm almost needs 400 iterations to reach the Nash Equilibrium, while our algorithm only needs about 45 iterations. Similarly, when we use mid_init and $high\_init$ as the initialization, our algorithm needs about 50 and 100 iterations respectively to reach the Nash Equilibrium, while the baseline algorithm always needs nearly (or even more than) 400 iterations to stabilize to the equilibrium point.

We investigate the effect of the step on the convergence of the baseline algorithm and FNES in Fig. 3, in which we use low_init as the initialization. We can see that the baseline algorithm is sensitive to the value of the step Δ. When we increase Δ from 0.5 to 5, the oscillation phenomenon will be more obvious, and it needs more iterations to reach the Nash Equilibrium. On the contrary, our algorithm FNSE can always converge to the Nash Equilibrium quickly no matter how the steps Δ_h and Δ_t are set, and it even performs better when Δ_h and Δ_t are set to relatively large values. As shown in Fig. 3(b), Fig. 3(d), Fig. 3(f), and Fig. 3(h), when the steps Δ_h and Δ_t are set to be 0.5, 1, 2, and 5, FNSE needs about 72, 42, 35, and 34 iterations, respectively, to reach the Nash Equilibrium. By comparison, we can know that the



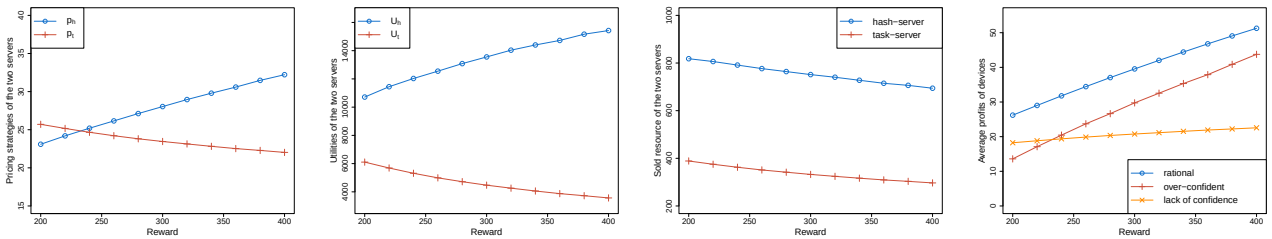
(a) (b) (c) (d)

Fig. 4. The effect of the percentage of the over-confident followers on (a) pricing strategies of servers, (b) utilities of servers, (c) total sold resources of servers and (d) average profits of devices.



(a) (b) (c) (d)

Fig. 5. The effect of the percentage of the lack of confidence followers on (a) pricing strategies of servers, (b) utilities of servers, (c) total sold resources of servers and (d) average profits of devices.



(a) (b) (c) (d)

Fig. 6. The effect of the R on (a) pricing strategies of servers, (b) utilities of servers, (c) total sold resources of servers and (d) average profits of devices.

baseline algorithm requires us to carefully design the step value to get a better convergence effect. In contrast, our algorithm can adapt to different step values and will not oscillate during the convergence process.

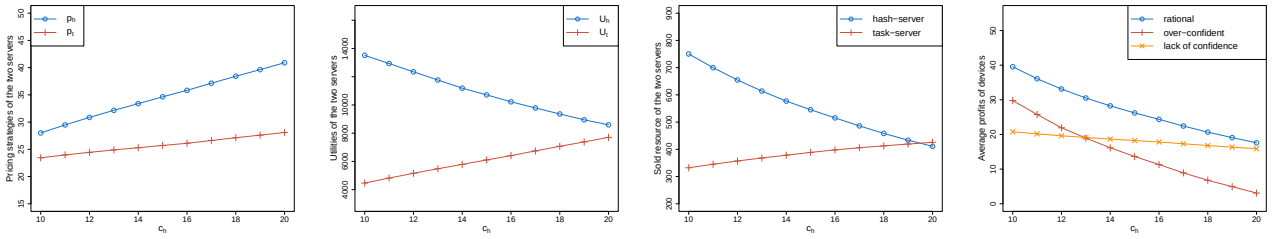
6.2.2 Parameter sensitivity

In this subsection, we investigate how the different values of parameters affect the equilibrium point. Each data point plotted in this subsection is the average 100 runs unless other statements.

1) The effect of the percentage of irrational followers (devices): In our simulations, we consider there are 500 devices in an IoT system, and the percentage of each of the two kinds of irrational followers is 10%; the other devices are rational followers. To investigate the effect of the percentage of irrational followers, we keep the percentage of one kind of irrational followers unchanged, and then increase the number of another kind of irrational followers from 0% to 30% with a step size of 5%. Correspondingly, the number of rational followers is reduced from 90% to 60% with a step size of 5%.

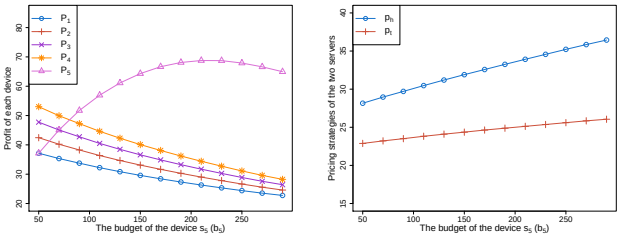
As shown in Fig. 4, when the percentage of the over-confident followers is increased from 0% to 30%, we can see that the hash-server will raise its price to get a larger utility, as the over-confident followers always allocate all of their budgets to purchase resources from the hash-server. The task-server also slightly raises its price as it is more competitive due to the higher price or the hash-server. The results are shown in Fig. 4(a) and Fig. 4(b). As both the hash-server and task-server raise their pricing strategies, the total sold resources of the two servers will decrease when we increase the percentage of the over-confident followers, as shown in Fig. 4(c). In Fig. 4(d), we can see that the average profits of both rational devices and over-confident devices are significantly reduced when there are more over-confident devices in the system, while the average profit of the lack of confidence devices slightly decreases.

When the percentage of the lack of confidence followers is increased from 0% to 30%, both the hash-server and task-server raise their pricing strategies, as shown in Fig. 5(a). As the lack of confidence followers only purchase resources from the task-server, there will be fewer followers purchase resources from the hash-server when the percentage of the



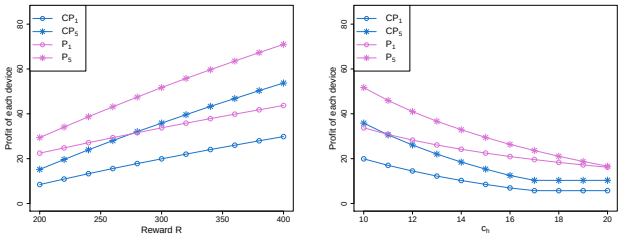
(a) (b) (c) (d)

Fig. 7. The effect of c_h on (a) pricing strategies of servers, (b) utilities of servers, (c) total sold resources of servers and (d) average profits of devices.



(a) (b)

Fig. 8. The effect of the budget b_5 of device s_5.



(a) (b)

Fig. 9. The effect of cooperation between devices s_1 and s_5.

lack of confidence followers is increased. Thus total sold resources and the utilities of the hash-server will decrease, as shown in Fig. 5(b) and Fig. 5(c). In Fig. 5(d), we can see that the average profits of the three kinds of devices will slightly decrease when there are more lack of confidence devices in the system. Combining Fig. 4(d) and Fig. 5(d), we can see that the percentage of over-confident followers has a greater impact on the profits of devices. Moreover, we can also see that rational devices always get more profits than irrational devices.

2) The effect of the mining reward R: We investigate the effect of the mining reward R on the final solution of our problem, as shown in Fig. 6. When the mining reward R increases from 200 to 400, the miners (IoT devices) will get more profit from the mining process, and thus the rational devices prefer to spend their budget on purchasing hash computational power. Therefore, the hash-server will give a higher price to get a larger utility, and the task-server has to lower its price to attract the devices. The results are shown in Fig. 6(a) and Fig. 6(b). From Fig. 6(c) we can see that the total purchased hash resource of devices decreases as the reward R increases. This is because the price of the hash resource has been raised up, and the devices have limited budgets. It indicates that if the blockchain platform wants to attract miners to contribute more computational power by increasing the mining reward, it may have an opposite effect. Devices will get more profit as the mining reward R increases, especially for rational devices and the over-confident devices, as shown in Fig. 6(d).

3) The effect of the unit resource cost of the servers: We keep the unit resource cost c_t of the task-server unchanged, and increase the unit resource cost c_h of the hash-server from 10 to 20. As the unit resource cost c_h raised up, the hash-server will raise its resource price to get a larger utility. Thus rational devices will allocate more budget to purchase resources from the task-server, and then the task-server will raise its resource price as it is more competitive. The results are shown in Fig. 7(a). The total purchased hash resource of devices will decrease due to the above reasons. Meanwhile, devices will purchase more resources from the task-server, as shown in Fig. 7(c). The utility of the hash-server decreases with increasing the unit resource cost c_h. The reason is that the total sold resources of the hash-server decreases as c_h increases. Even though the price p_h has risen, the value $p_h - c_h$ is almost unchanged. The results are shown in Fig. 7(b). As both of the two servers will bid a higher price as c_h increases, devices will get less profit, especially for rational devices and the over- confident devices, as shown in Fig. 7(d). From Fig. 7, we can conclude that the server with a lower unit resource cost will be more competitive and thus obtain more benefits.

4) The effect of the budget of devices: To investigate the effect of the budget of a device on other rational devices, we consider a small blockchain network as well as a small IoT system, in which total hash computational power H is estimated to be 1000 GH/s, and the number of generated new blocks per day N is 144. There are 5 rational devices $(s_1, s_2, s_3, s_4, s_5)$ in the IoT system, and the budget of each device is 50, 60, 70, 80, and 90, respectively. The other parameters are the same as the default settings in subsection 6.1. If we increase the budget b_5 of device s_5 from 50 to 290, while the budgets of other devices keep unchanged, the profit gets by s_5 will first increase and then decrease slowly, while the profits of other devices decrease, as shown in Fig. 8(a). The reason is that the increment of the budget b_5 will cause servers to raise their resource prices, which in turn reduces the amount of resources purchased by other devices, as shown in Fig. 8(b). As the resource prices of the two servers keep rising, device s_5 can not always get

more profits simply by increasing its own budget. The result indicates that the budget of devices will affect each other's profit in an indirect way.

5) The effect of the device collusion: Here we investigate whether the collusion of rational devices could increase their profits. We also consider the above small blockchain network as well as the small IoT system. Assume that device s_1 will collude with device s_5, that is, device s_1 will give its own budget to device s_5, and they will share the profits proportionally. The budgets of devices s_1 and s_5 are 50 and 90, respectively, therefore, s_1 and s_5 get $5/14$ and $9/14$ of the total profits, respectively. We use CP_1 and CP_5 to denote the profits of devices s_1 and s_5 after they collude with each other. As shown in Fig. 9, no matter how the reward R or the resource cost c_h of the hash-server changes, the collusion between devices s_1 and s_5 will not increase their profits, on the contrary, it will reduce their profits. Therefore, it is impossible for devices to collude with each other to obtain more profits, which also contributes to the security of the blockchain network.

7 Conclusion

In this paper, we study the pricing and budget allocation problem between edge servers and IoT devices in an IoT blockchain network. We first introduce the architecture of IoT blockchain with edge computing, and describe the operation of the IoT blockchain system. Then, we model the interaction between edge servers and IoT devices as a multi-leader multi-follower Stackelberg game, in which both rational and irrational devices are considered. We prove the existence and uniqueness of the Stackelberg equilibrium, and design efficient algorithms to get the Stackelberg equilibrium point. Finally, our designs are validated by extensive simulations. In our future work, we will study how to efficiently solve the equilibrium point when there are more leaders, and the influence of the competition and cooperation between leaders on the equilibrium point. Moreover, the influence of the oligopoly market and market power on the equilibrium point is also worthy of further study.

References

[1] M. Rehman, N. Javaid, M. Awais, M. Imran, and N. Naseer, "Cloud based secure service providing for iots using blockchain," in *2019 IEEE Global Communications Conference (GLOBECOM)*. IEEE, 2019, pp. 1–7.

[2] R. Roman, J. Zhou, and J. Lopez, "On the features and challenges of security and privacy in distributed internet of things," *Computer Networks*, vol. 57, no. 10, pp. 2266–2279, 2013.

[3] D.-Y. Kim, A. Lee, and S. Kim, "P2p computing for trusted networking of personalized iot services," *Peer-to-Peer Networking and Applications*, vol. 13, no. 2, pp. 601–609, 2020.

[4] F. A. Alaba, M. Othman, I. A. T. Hashem, and F. Alotaibi, "Internet of things security: A survey," *Journal of Network and Computer Applications*, vol. 88, pp. 10–28, 2017.

[5] N. Kshetri, "Can blockchain strengthen the internet of things?" *IT professional*, vol. 19, no. 4, pp. 68–72, 2017.

[6] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," Manubot, Tech. Rep., 2019.

[7] M. Divya and N. B. Biradar, "Iota-next generation block chain," *International journal of engineering and computer science*, vol. 7, no. 04, pp. 23 823–23 826, 2018.

[8] Z. Zheng, S. Xie, H.-N. Dai, X. Chen, and H. Wang, "Blockchain challenges and opportunities: A survey," *International Journal of Web and Grid Services*, vol. 14, no. 4, pp. 352–375, 2018.

[9] Y. Wu, P. Song, and F. Wang, "Hybrid consensus algorithm optimization: A mathematical method based on pos and pbft and its application in blockchain," *Mathematical Problems in Engineering*, vol. 2020, 2020.

[10] J. Pan, J. Wang, A. Hester, I. Alqerm, Y. Liu, and Y. Zhao, "Edgechain: An edge-iot framework and prototype based on blockchain and smart contracts," *IEEE Internet of Things Journal*, vol. 6, no. 3, pp. 4719–4732, 2018.

[11] R. Morabito, V. Cozzolino, A. Y. Ding, N. Beijar, and J. Ott, "Consolidate iot edge computing with lightweight virtualization," *IEEE Network*, vol. 32, no. 1, pp. 102–111, 2018.

[12] O. Novo, "Blockchain meets iot: An architecture for scalable access management in iot," *IEEE Internet of Things Journal*, vol. 5, no. 2, pp. 1184–1195, 2018.

[13] K. Gai, Y. Wu, L. Zhu, L. Xu, and Y. Zhang, "Permissioned blockchain and edge computing empowered privacy-preserving smart grid networks," *IEEE Internet of Things Journal*, vol. 6, no. 5, pp. 7992–8004, 2019.

[14] J. Guo, X. Ding, and W. Wu, "A blockchain-enabled ecosystem for distributed electricity trading in smart city," *IEEE Internet of Things Journal*, pp. 1–1, 2020.

[15] M. Li, F. R. Yu, P. Si, W. Wu, and Y. Zhang, "Resource optimization for delay-tolerant data in blockchain-enabled iot with edge computing: A deep reinforcement learning approach," *IEEE Internet of Things Journal*, 2020.

[16] D. Liu, J. Ni, C. Huang, X. Lin, and X. Shen, "Secure and efficient distributed network provenance for iot: A blockchain-based approach," *IEEE Internet of Things Journal*, 2020.

[17] S. Qi, Y. Lu, Y. Zheng, Y. Li, and X. Chen, "Cpds: Enabling compressed and private data sharing for industrial iot over blockchain," *IEEE Transactions on Industrial Informatics*, 2020.

[18] K. Lei, M. Du, J. Huang, and T. Jin, "Groupchain: Towards a scalable public blockchain in fog computing of iot services computing," *IEEE Transactions on Services Computing*, vol. 13, no. 2, pp. 252–262, 2020.

[19] Z. Chang, W. Guo, X. Guo, Z. Zhou, and T. Ristaniemi, "Incentive mechanism for edge computing-based blockchain," *IEEE Transactions on Industrial Informatics*, 2020.

[20] H. Yao, T. Mai, J. Wang, Z. Ji, C. Jiang, and Y. Qian, "Resource trading in blockchain-based industrial internet of things," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 6, pp. 3602–3609, 2019.

[21] Z. Xiong, S. Feng, D. Niyato, P. Wang, and Z. Han, "Edge computing resource management and pricing for mobile blockchain," *arXiv preprint arXiv:1710.01567*, 2017.

[22] Z. Xiong, Y. Zhang, D. Niyato, P. Wang, and Z. Han, "When mobile blockchain meets edge computing," *IEEE Communications Magazine*, vol. 56, no. 8, pp. 33–39, 2018.

[23] X. Ding, J. Guo, D. Li, and W. Wu, "An incentive mechanism for building a secure blockchain-based industrial internet of things," *arXiv preprint arXiv:2003.10560*, 2020.

[24] S. Guo, Y. Dai, S. Guo, X. Qiu, and F. Qi, "Blockchain meets edge computing: Stackelberg game and double auction based task offloading for mobile blockchain," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 5, pp. 5549–5561, 2020.

[25] D. Johnson, A. Menezes, and S. Vanstone, "The elliptic curve digital signature algorithm (ecdsa)," *International journal of information security*, vol. 1, no. 1, pp. 36–63, 2001.

[26] N. Z. Aitzhan and D. Svetinovic, "Security and privacy in decentralized energy trading through multi-signatures, blockchain and anonymous messaging streams," *IEEE Transactions on Dependable and Secure Computing*, vol. 15, no. 5, pp. 840–852, 2016.

[27] S. Zhang and J.-H. Lee, "A group signature and authentication scheme for blockchain-based mobile-edge computing," *IEEE Internet of Things Journal*, vol. 7, no. 5, pp. 4557–4565, 2019.

[28] P. Koshy, S. Babu, and B. Manoj, "Sliding window blockchain architecture for internet of things," *IEEE Internet of Things Journal*, vol. 7, no. 4, pp. 3338–3348, 2020.

[29] W. Yang, X. Dai, J. Xiao, and H. Jin, "Ldv: A lightweight dag-based blockchain for vehicular social networks," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 6, pp. 5749–5759, 2020.

[30] G. Li, Q. Zhao, M. Song, D. Du, J. Yuan, X. Chen, and H. Liang, "Predicting global computing power of blockchain using cryptocurrency prices," in *2019 International Conference on Machine Learning and Cybernetics (ICMLC)*. IEEE, 2019, pp. 1–6.

[31] J. B. Rosen, "Existence and uniqueness of equilibrium points for concave n-person games," *Econometrica: Journal of the Econometric Society*, pp. 520–534, 1965.
[32] S. Boyd, S. P. Boyd, and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.
[33] H. Zhang, Y. Xiao, L. X. Cai, D. Niyato, L. Song, and Z. Han, "A multi-leader multi-follower stackelberg game for resource management in lte unlicensed," *IEEE Transactions on Wireless Communications*, vol. 16, no. 1, pp. 348–361, 2016.

Xingjian Ding received his B.E. degree in electronic information engineering from Sichuan University in 2012 and M.S. degree in software engineering from Beijing Forestry University in 2017. He obtained his Ph.D. degree from the School of Information, Renmin University of China in 2021. He is currently an assistant professor at the School of Software Engineering, Beijing University of Technology. His research interests include wireless rechargeable sensor networks, approximation algorithms design and analysis, and blockchain.

Jianxiong Guo received his Ph.D. degree from the Department of Computer Science, University of Texas at Dallas, Richardson, TX, USA, in 2021, and his B.E. degree from the School of Chemistry and Chemical Engineering, South China University of Technology, Guangzhou, Guangdong, China, in 2015. He is currently an Assistant Professor with the BNU-UIC Institute of Artificial Intelligence and Future Networks, Beijing Normal University at Zhuhai, and also with the Guangdong Key Lab of AI and Multi-Modal Data Processing, BNU-HKBU United International College, Zhuhai, Guangdong, China. His research interests include social networks, algorithm design, data mining, IoT application, blockchain, and combinatorial optimization.

Deying Li is a professor of Renmin University of China. She received the B.S. degree and M.S. degree in Mathematics from Huazhong Normal University, China, in 1985 and 1988 respectively. She obtained the PhD degree in Computer Science from City University of Hong Kong in 2004. Her research interests include wireless networks, ad hoc & sensor networks mobile computing, distributed network system, Social Networks, and Algorithm Design etc.

Weili Wu received the Ph.D. and M.S. degrees from the Department of Computer Science, University of Minnesota, Minneapolis, MN, USA, in 2002 and 1998, respectively. She is currently a Full Professor with the Department of Computer Science, The University of Texas at Dallas, Richardson, TX, USA. Her research mainly deals in the general research area of data communication and data management. Her research focuses on the design and analysis of algorithms for optimization problems that occur in wireless networking environments and various database systems.