Trustworthy Blockchain-Empowered Collaborative Edge Computing-as-a-Service Scheduling and Data Sharing in the IIoE

Fenhua Bai, *Graduate Student Member, IEEE*, Tao Shen, *Member, IEEE*, Zhuo Yu, Kai Zeng, and Bei Gong, *Member, IEEE*

***Abstract*—Owing to the technology of 5G and beyond, collaborative edge computing-as-a-service has enabled trillions of interconnected edge applications. It has also become a prospective paradigm for providing computing services by offloading computationally intensive assignments to mobile-edge servers or fog nodes due to terminals constrained computing and caching resources. Nevertheless, in this process, trust of computing-as-a-service scheduling and edge data sharing in heterogeneous systems is an unavoidable challenge of paramount importance. As a powerful tool that addresses security issues, blockchains can ensure the trustworthiness and irreversibility of computing data by consensus mechanisms. However, in the Industrial Internet of Energy (IIoE), the storage burden of a single blockchain has increased. Therefore, from the perspective of a stable real-time operation, we propose a multiedgechain structure that accommodates thousands of edge data and promotes on-chain data efficiency to achieve cross-chain edge data sharing for heterogeneous blockchain systems. Moreover, aiming at the profits of computing resource scheduling in the IIoE, a two-stage Stackelberg game strategy with an optimal scheduling demand and reward is provided considering the edge user's preferences and risk factors. Finally, the simulation results verify the superiority of the proposed scheme, regarding the game equilibrium, utility optimization, and data sharing efficiency of cloud–edge collaboration.**

***Index Terms*—Blockchain, cross-chain, edge computing, Industrial Internet of Energy (IIoE), noncooperative games.**

I. Introduction

With the continuous implementation of new infrastructure in the Industrial Internet of Energy (IIoE), the number of edge devices and applications has grown exponentially. For various applications, such as electric vehicles, smart energy metering, and unmanned patrol inspection machines, edge/fog computing can meet the requirement of real-time operation and provide edge intelligent services near the edge [1]–[2]. However, a nonnegligible challenge when exploiting these services in the IIoE is that the required computing and storing capacity may not be well provided by the limited resources of edge devices [3]. Mobile-edge servers (MSs) or fog nodes (FNs) with intensive computing resources can undertake computing tasks and improve the efficiency of edge computing services [4], [5].

When utilizing the computation and storage capacity of MSs and FNs, it is worth mentioning that the offloaded task scheduling should be concerned with the participant's cost optimization to determine the optimal strategy. Many researchers [6]–[9] have discussed the issue of edge computing resource scheduling. For example, Li *et al.* [6] optimized computational offloading with minimum energy consumption. In [7], the proposed methods reduced the computing task latency of edge devices and significantly improved the application service quality. Goudarzi *et al.* [8] and Luo *et al.* [9] provided a dynamic calculation for the real-time vehicle operation. From the perspective of participant cost optimization, a Stackelberg game was applied to construct interactions among neighboring devices, edge cloud operators, and collaborative mining networks to ensure the maximum profit of edge–cloud operators [10]. In [11] and [12], noncooperative games were also formulated to determine the optimal price of both sides when tradable activities occur. A risk-aware computation offloading policy was modeled as a Bayesian–Stackelberg game to safely distribute computation tasks in [13]. Although the reward mechanism is crucial for the collaborative edge computing-as-a-service (CaaS) scheduling problem in the IIoE, except for mining networks [14], there has been little work that analyzes the optimal rewards by means of a game approach. Thus, efficient CaaS scheduling and reward schemes are needed to compensate for constrained resources and improve the edge computing resource availability in the IIoE.

Moreover, a trustworthy data consistency mechanism between the edge computing-unloaded devices and providers is important. Fortunately, the data integrity and computation verifiability of the system can be considerably improved by

Manuscript received 15 August 2020; revised 24 December 2020 and 21 January 2021; accepted 30 January 2021. Date of publication 9 February 2021; date of current version 8 August 2022. This work was supported in part by the Major Scientific and Technological Projects in Yunnan Province under Grant 202002AB080001; in part by the National Natural Science Foundation of China under Grant 61702128; and in part by the Yunnan Applied Basic Research Projects under Grant 2018FA034. The work of Tao Shen was supported in part by the Yunnan Young Top Talents of Ten Thousands Plan under Grant 201873. *(Corresponding author: Tao Shen.)*

Fenhua Bai, Tao Shen, and Kai Zeng are with the Faculty of Information Engineering and Automation, Kunming University of Science and Technology of China, Kunming 650500, China (e-mail: bofenhua@stu.kust.edu.cn; shentao@kust.edu.cn; zengkailink@sina.com).

Zhuo Yu is with the Department of Research and Development, State Grid Information and Telecommunication Company, Ltd., Beijing 102209, China (e-mail: yuzhuo@sgitg.sgcc.com.cn).

Bei Gong is with the Department of Computing, Beijing University of Technology, Beijing 100124, China (e-mail: gongbei@bjut.edu.cn).

Digital Object Identifier 10.1109/JIOT.2021.3058125

integrating blockchains into the edge computing network [3]. For the majority of the existing schemes [10], [15], [16], the blockchain technological paradigm can offer reliable access control and storage over a great number of decentralized edge nodes. Essentially, a blockchain can act as a trustworthy solution because of the consensus algorithm that it adopts with strong data consistency and irreversible hash encryption. It can resist information leakage attacks from both sides for interactive computation. What is more, as a distributed database, blockchains can store edge computing data and outcomes reliably in a chronological order so that the results of being on the blockchain cannot be changed [17].

Clearly, as a new infrastructure develops in the IIoE, the burden of blockchain data storage has gradually increased, leading to significant disadvantages of low efficiency. Consequently, how to efficiently and securely share the massive amount of available data to improve the edge computing experience and provide a higher quality of computing is of significance. To alleviate the burden of blockchains and increase their transaction efficiency, the sidechain structure was first proposed by Back *et al.* [18]. Most notably, each sidechain is completely independent in this scheme [19], and there are different consensus algorithms or blockchain types (e.g., private, consortium, or public blockchains) [20]. Certainly, as a technology for accomplishing cross-chain interoperability, sidechains can perform cross-chain data interaction at the edge of the network in the IIoE under the condition of security.

Regarding cross-chain verification, edge data consistency is still an outstanding issue in data sharing among heterogeneous blockchains. Zaghloul *et al.* [21] provided a scheme for multilevel organizational data sharing in cloud computing but does not consider the edge computing. The existing different assets trading the validation of sidechain technology depends on the two-way peg protocol by simplified payment verification (SPV) [18]. This tends be slow and involves more time overhead because the data user has to wait for the confirm and contest periods before having access to sharing data on either the sidechain or the mainchain [19]. Accordingly, in this article, we adopt the cloudlet chain as a cross-chain application programming interface (API) to reduce the complexity of cross-chain data interaction. Both edgechains register a cross-chain service on the cloudlet chain. Then, edge data sharing among edgechains can be quickly accomplished.

The specific contributions of this article are threefold.

1) Aiming at the real-time and stability requirements of the edge devices dealing with computing tasks in the IIoE, we propose a multiedgechain and cloudlet chain structure. Each edge blockchain can run independently, and collaborative edge computing results on chain storage can be performed concurrently. Moreover, when employing a consortium blockchain or private blockchain, the edgechains are responsible for the transaction of CaaS scheduling in different domains and provide an unalterable storage method for a variety of computing data from different domains (such as energy trading, intelligent manufacturing factories and electric vehicles) in the IIoE.
2) We construct a noncooperative game model in which we use a two-stage Stackelberg game to address the problem of optimizing participants' utilities. This game contains offloading calculation demand and reward strategies between the constrained edge devices and MSs or FNs for CaaS scheduling while practically considering the edge user's preferences and risk factors. Additionally, gradient descent is used to solve the game equilibrium rapidly to satisfy the real-time requirements of CaaS scheduling.
3) In particular, in our devised multichain structure, the double-level improved practical Byzantine fault tolerance (DLPBFT) consensus algorithm is proposed to reduce transaction processing time and guarantee the data consistency stored on the blockchains. Then, concerning the interoperability among heterogeneous edgechains and edge-cloud collaboration in the IIoE, the cloudlet chain is used to provide a cross-chain interface, bridge each of the edgechains, and further support cross-chain data sharing for itself or other edgechains with edge terminals and applications in different domains.

The remainder of this article is organized as follows. Section II introduces the game theory for CaaS scheduling and the cross-chain technology that aims at heterogeneous blockchain interoperability, and then gives an overview of related works. Section III presents the system formulation and the proposed multiedgechain architecture of this article. Section IV describes the evaluation and simulation results of this research. Finally, the conclusion is given in Section V.

II. Related Works

Recently, the blockchain concept has emerged as one of the most influential technologies that enables its application with the Internet of Things (IoT), artificial intelligence, edge computing, and big data. Nevertheless, blockchains have run into a wall in terms of challenges regarding scalability, interoperability, security, etc., [3]. Since a single blockchain to rule them all is impracticable [19], rather than having disparate blockchains, it would seem more worthwhile to make separate blockchains be interoperable such that they can communicate and interact with one another.

A. Interoperability of the State-of-the-Art Cross-Chains

1) On-Chain Interoperability: One way to solve the interoperability problem is to use another blockchain as a communication bridge. In short, a third blockchain is built between two blockchains to record the transaction and message data in a securely encrypted manner. In general, the hub-spoke mode, in which the parent blockchain is the hub of other blockchains (also known as sidechains), is the most common mode for on-chain interaction. At present, Polkadot, Cosmos, and Ethereum have adopted this model in several sidechain proposals (plasma, matic, and loom) [19].

Furthermore, the general bridge can be used to verify, record, and store cross-chain interactive data into accounts, and provide blockchain consensus with a timestamp.

2) Off-Chain Interoperability: The function of Oracle is to connect two blockchains as a universal bridge. This function cannot only achieve information exchange between blockchains but also interact with any enterprise system

beyond the blockchains. For example, transaction data on one blockchain are used as the input to trigger a smart contract on another blockchain. Oracles can empower many kinds of cross-chain interaction modes that others cannot.

Chainlink, the first decentralized oracle network, is equivalent to the HTTP protocol in function (or HTTPS in a trusted execution environment); it performs on-chain and off-chain information transmission in the protocol and application layers. Chainlink nodes are able to package information and data from the validated APIs into a format that smart contracts can read. Moreover, a Chainlink node can be connected to any API, whether it is a blockchain, enterprise system, Web API or IoT device. If a task is not supported by the Chainlink core node, it is also easy to establish an external adapter to extend the node function. Although a new era for smart contracts 3.0 is expected to open, Chainlink is relatively immature and still in the exploratory stage.

B. Existing Cross-Chain Technology

Currently, there are three kinds of mainstream cross-chain technology: 1) notary mechanisms; 2) sidechains/relays; and 3) hash locking.

1) Notary Mechanism: A notary mechanism is a centralized cross-chain mode. Both sides of the cross-chain establish trust through a trustworthy third party elected as a notary, which is in charge of validating the legitimacy and consistency of information only and does not participate in business details. The most prominent characteristic of the notary mechanism is its operation irrespective of the structure and consensus mechanism for both parties in the blockchain. Representative cases are Interleger and Corda [22]. However, the notary mechanism has an obvious defect, which is that there need to be sufficient trust in the notary, and there is a great risk of centralization for the single node notary mechanism [19].

2) Sidechains/Relays: Seeing the obstruction in the application and further growth of blockchains for constructing scalable, advanced, and implementable information systems, Back *et al.* [18] first proposed the concept of two-way anchoring sidechains in 2014 for addressing the interoperability of traditional blockchains. Essentially, the structure of a sidechain is a main blockchain with secondary subblockchains, which are connected to other blockchains with the help of a two-way peg based on SPV [23]. Two-way anchoring is a mechanism that allows bidirectional interactions between the mainchain and the sidechain. Sidechains may have their own consensus protocol and implementation, which can be thoroughly distinct from the parent blockchain. The adjustability of this design provides flexibility to users regarding access, especially for heterogeneous functionalities and features offered on a sidechain. Furthermore, sidechains are separated from the parent blockchain so that protecting the mainchain from malicious attacks due to damage is completely limited to the sidechain itself [19].

3) Hash Locking: Hash locking is a mechanism that relies on the unidirectionality and low collision of hash functions. Cross-chain interactions in different blockchains are realized by setting triggers that depend on a smart contract. The first project to use hash locking technology resulted from the Bitcoin's lightning network [24], which establishes trust without a notary. As long as there is a channel connecting the two sides of the transaction in the network, this channel can be used for transactions. In simple terms, the two chains receive unlocking information within the specified time and then send assets. As a result, the operation cost of hash locking technology is high and is accompanied by insufficient security [22].

C. Edge Computing Resource Scheduling

As mentioned above, to decrease the communication overhead of cloud computing, as well as address the limited edge equipment computing ability, edge computing has emerged as a promising solution in which distributed energy terminals can offload computationally intensive tasks [25] to adjacent edge servers or FNs in the IIoE. Computational resource-optimized scheduling approaches are important for providing effective and low-delay computing services during task offloading. Several strategies have been adopted to optimize edge computing resource scheduling, such as deep reinforcement learning (DRL) [3], [26], [27]–[29], auction theory [10], [30], [31], and game theory [32], [33]. Among these approaches, the game theory has been the most extensively applied to optimize the problem of resource scheduling. The contract theory is also a powerful tool to solve the incentive problem with asymmetric information [34].

D. Cross-Chain Edge Data Sharing

Edge data sharing is a crucial segment among heterogeneous edge blockchains in the IIoE. To provide better schedule-based decision making, users in different energy domains need to share their data on the chain with adjacent edge chain data [35], [36]. For instance, smart energy factories can share edge data resources with energy production plants to optimize CaaS scheduling and further facilitate energy management. In the process of cross-chain data sharing, cryptographic mechanisms that consider privacy preservation are also applied to encrypt the data so that only the user authorized by the data owner can decrypt the data and preserve the data confidentiality of edge terminals [37].

Although a large number of studies have analyzed how to share and avoid the excessive disclosure of the data of the owner in this process [36], [38], the existing schemes for data sharing based on blockchain leave out of consideration the problem of the heterogeneous blockchain in IIoE systems, and the related research is insufficient.

E. Blockchain Consensus Mechanism

As a crucial component of blockchain technology, the consensus algorithm is commonly used to reach an agreement among the validated nodes and decide whether to generate a new block into the blockchain. It is the consensus algorithm that ensures the trust and confidentiality of the generated block.

There are some widely used consensus mechanisms for security and data consistency. For public blockchains, Proof

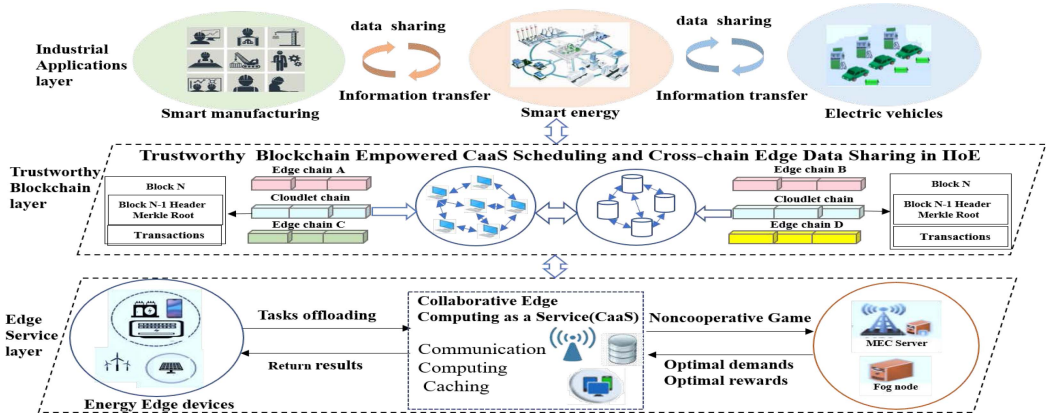

Fig. 1. Proposed system architecture in the IIoE.

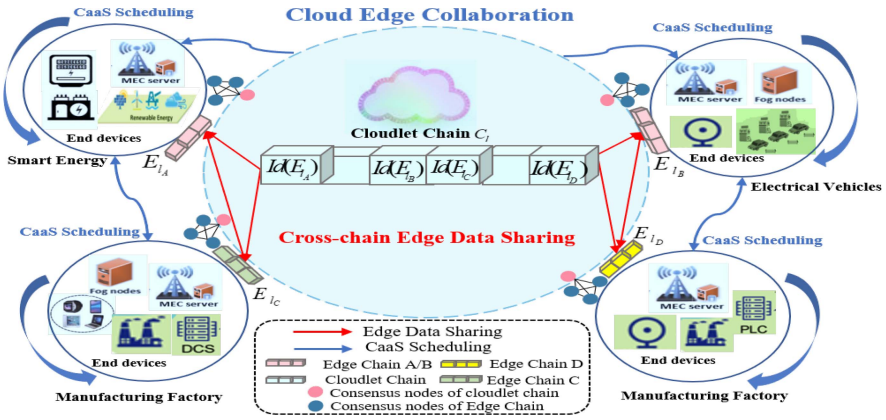

Fig. 2. CaaS scheduling and edge–cloud collaboration.

of Work (PoW) [39] (depending on the node's computational power to win the right of adding a new block by solving a computationally expensive puzzle) is commonly used in Bitcoin, and Proof of Stake (PoS) [40] is applied in Ethereum. To address the problem of expensive investment of energy in PoW, the Proof of Elapsed Time (PoET) was introduced by Intel for permissioned blockchain applications [41] in Hyperledger Sawtooth. The validator with the shortest waiting time for a particular block is selected as the leader in the trusted enclave. Delegated PoS (DPoS) [42] is a variant of PoS; the primary difference between them is that DPoS is faster than PoS in the generated block because the former chooses only the delegated nodes to participate in the consensus. Generally, applying in alliance blockchains, PBFT solves the challenges associated with the Byzantine general problem [43] for asynchronous circumstances. It is necessary that at least two-thirds of the participants are honest. Moreover, other newly consensus algorithms, including Ouroboros [44], Algorand [45], and Snowflake to Avalanche [46], are emerging.

III. System Formulation

In this section, the triple-layer architecture shown in Fig. 1 is proposed to address the challenge of trustworthy CaaS scheduling and cross-chain data sharing in the IIoE. The constructed architecture comprises the edge service layer, intelligent blockchain layer, and industrial application layer.

1) *Edge Service Layer:* A large number of mobile-edge energy terminals are interconnected in the IIoE. Thus, the end devices with constrained computing resources can offload the computational tasks to MSs and FNs by CaaS scheduling to improve the efficiency of the calculation and make full use of the computing resources. To optimize CaaS scheduling, we make a model of a noncooperative game between edge energy devices and MSs or FNs to determine the optimal CaaS dispatching demands and best rewards after finishing the computing tasks.
2) *Trustworthy Blockchain Layer:* In this layer, a trustworthy blockchain secures reliable data coherency and provides tamper-resistant storage for the computing results. Moreover, the multiedgechain structure can decrease the storage burden and efficiency dilemma of the single blockchain. In addition to describing the data structure itself, the proposed structure can achieve cross-chain data sharing among heterogeneous edge blockchains in the IIoE.
3) *Industrial Application Layer:* For the industrial application layer, the success of CaaS scheduling is achieved through the trustworthy blockchain and cross-chain edge data sharing. Consequently, industrial applications, such as energy trading, intelligent manufacturing, and electric vehicles can be further integrated into the best services and bring about efficiency improvements for the IIoE.

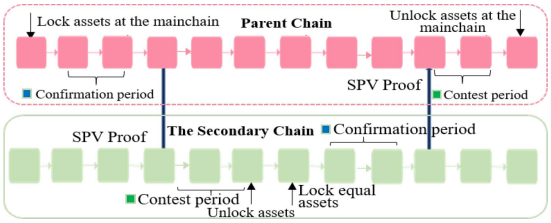

Fig. 3. Two-way peg based on SPV.

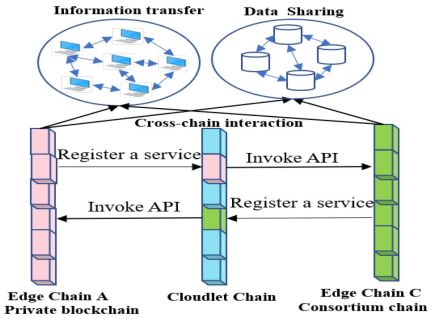

Fig. 4. Cross-chain data interactions.

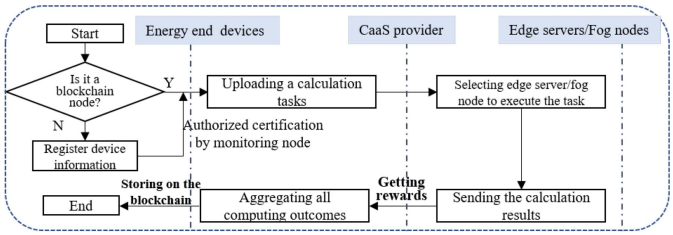

Fig. 5. Trustworthy access control.

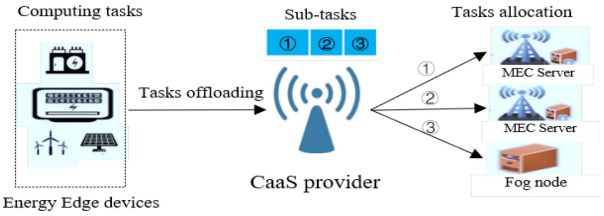

Fig. 6. Task offloading and allocation.

The purpose of the multiedgechain structure (shown in Fig. 2) in this article is to decrease the single blockchain's burden due to the massive CaaS scheduling data and computing result storage, concurrently preventing the blocks from undergoing tampering. Since enabling interactions among different edgechains is a significant problem, sidechain technology has received much attention in recent years. The majority of the existing research on sidechain technology has focused on cross-chain value circulation with two-way pegs [20]; then, SPV is applied to verify that a transaction does exist in a block according to a Merkle path proof. This is obviously time consuming due to the confirmation and contest periods (shown in Fig. 3). The time complexity of SPV is $O(n)$.

Herein, we adopt a single-cloudlet-chain and multi-edgechain structure. As shown in Fig. 4, the function of the cloudlet chain is to act as a bridge that connects two arbitrarily edgechains (for example, edgechains A and C). In the process of cross-chain data interaction, each edgechain should register services on the cloudlet chain. The cloudlet chain is responsible for storing and maintaining the services provided by each edgechain, and the service directory is open to each edgechain. When an edgechain needs cross-chain data interaction, the edgechain service recorded in the cloudlet chain should be used for data crosslink sharing. First, the operator for each of the edgechains registers as a service node on the cloudlet chain. Next, if edgechain A needs to share the data or information of edgechain C, it invokes the API of edgechain C on the cloudlet chain. Then, edgechain C sends back the hashed data to edgechain A. Additionally, the cloudlet chain can share the information of all the edgechains so that cloud edge collaboration is realized.

A. Optimizing CaaS Scheduling

In our proposed system model, CaaS participants exploit blockchain technology to record edge computing service transaction information on the edgechain and ensure the maximum scheduling profit of edge computing services through game theory.

1) Trustworthy Access Control: In our devised architecture, we leverage the blockchain as a trustworthy network access control technique among the participants (such as the energy end users, CaaS vendors, and MSs or FNs in Fig. 6) through the characteristics of consistent consensus. Particularly, in this process, the requesting terminators and MSs or FNs use a hash function designed to anonymously obtain information when participating in the blockchain network as a consensus node in a secure manner. Specifically, the entities in the interaction adopt the DLPBFT consensus algorithm to reach an agreement for CaaS scheduling.

Moreover, to ensure the reliability of the task calculation and manage all the equipment's identification, the CaaS provider is designed as a monitoring node on one edgechain in the form of a consortium blockchain to carry out the management functions and control entering or leaving for edge energy devices and MSs or FNs. Notably, the supervised node makes scheduling decisions and has the right to reward the remaining nodes; it also makes credit evaluations according to the calculation results.

By coordinating the computing resources of MSs with FNs, the resource capacity at the edge of the IIoE can be expanded, and the processing ability can be further enhanced. This ensures that the CaaS provider deploys a better computing offloading service near terminal users. Based on the collaborative computing of the MSs and FNs, the service provider can optimize the distribution of the computing load to maximize the energy terminal user's satisfaction. Specifically, the service provider divides a complete calculation task uploaded into corresponding copies and distributes several copies to different MSs and FNs for independent execution. Once the task is finished, all executors summarize the calculation results for the CaaS provider, which aggregates them into the final calculation results and then returns them to the requesting end equipment.

2) Utility Function Formulation: Provided that the set of edge energy devices is $N = \{1, 2, \ldots, i, \ldots, n\}$, the end terminal i sends the request of task offloading. Some of its computing tasks can be unloaded to MSs, and the rest can be offloaded to FN. The load to be calculated is denoted as $L = \sum_{i=1}^{N} h_i D_i$, where D_i is the size of the data and h_i represents a coefficient related to the type of running application. A calculation task with a computing load L is divided into a group of subtasks and distributed to a group of selected MEC servers κ within the resource scheduling period. Each MEC server chooses to perform subtasks with different computing loads. If the total calculation load is too large during calculation offloading, a part of the task is allocated to FNs.

Therefore, the collaborative edge calculation process that coordinates the MSs with the FN is shown in Fig. 6. The computing services supplier responds to the request of the edge energy device in light of the MEC server's utility function. At the same time, the MEC server determines whether to perform a subtask according to its own utility function.

The utility function of the MEC server concerns mainly the reward and computing load status. When responding to energy end user i, x_k^i represents the calculated load promised by the MEC server k, $k \in \kappa$. In addition, r_k^i ($r_k^i \geq 0$) and c_k are the per-unit calculation load reward given by the requesting device i and the per-unit load cost of the MEC server k, respectively. Thus, the profits of the MEC server are $(r_k^i - c_k)x_k^i$. Obviously, when $r_k^i \geq c_k$, the MS can obtain revenue or reach the breakeven point. Furthermore, we should consider the maximum calculated load $s_{\max}^k$ that the MEC server can support. The risk factors caused by temporary participation in the calculation of task offloading are assumed to be $\xi_k (x_k^i)^2$, where $\xi_k = w_k s_{\max}^k$ and w_k are the weight coefficients of the risk factors. Hence, the kth MEC server's utility can be formulated as

$$U_k^i = \left(r_k^i - c_k\right)x_k^i - \xi_k \left(x_k^i\right)^2. \tag{1}$$

Similarly, the utility of the jth FN ($j \in J$, where J represents the set of FNs) is

$$U_j^i = \left(r_j^i - c_j\right)x_j^i - \xi_j \left(x_j^i\right)^2 \tag{2}$$

where $\xi_j = w_j g_{\max}^j$ and where $g_{\max}^j$ denotes that the jth FN can bear the maximum calculated load.

For the requesting end devices, the cost function is relevant mainly to the service cost to be paid. The service charge of the MS processing unit calculation load is p_m; then, what is the charge received by the MS is $p_m \sum_{k\in\kappa} x_k^i$. What is more, if the calculation load exceeds the maximum bearing capacity $s_{\max}^k$, the MSs can transfer the remaining tasks x_j^i to FNs. Therefore, the service overhead of the requested edge energy devices in the calculation of offloading can be written as

$$\begin{aligned} C_i &= p_m \sum_{k\in\kappa} x_k^i + p_f \sum_{j\in J} x_j^i + \sum_{k\in\kappa} r_k^i x_k^i + \sum_{j\in J} r_j^i x_j^i \\ \text{s.t. } \lambda_i L &\leq \sum_{k\in\kappa} x_k^i + \sum_{j\in J} x_j^i \leq L \end{aligned} \tag{3}$$

where p_m and p_f are the service charges of the MS and FN when undertaking the calculation load, respectively. The goal of the requesting user i is to optimize the reward strategy for all MSs and FNs, and further minimize the service cost under the condition of certain constraints while avoiding an excessive calculated load.

Indeed, the end devices have personal preferences when the CaaS provider allocates the computational load to MSs or FNs. Specifically, we introduce a preference ratio $\lambda_i (0 \leq \lambda_i \leq 1)$ on the basis of the recorded credit evaluation of MSs and FNs on the blockchain that is created by the monitoring nodes. Consequently, the requesting client can propose a proportional constraint λ_i [as shown in (3)] on the calculated load when the CaaS provider is distributed to the MSs and FNs.

3) CaaS Scheduling: From the perspective of the end requestors, as an intelligent agent, the CaaS provider represents the client in negotiating with the MS and FN about the reward strategy of the calculated load bearing. This relationship between requestors and responders exhibits a leader–follower attribute. Thus, we model this optimized problem as a two-stage Stackelberg game as

$$\Pi = \left\{ \begin{array}{l} \left(N \cup \{s_k\}_{k\in\kappa} \cup \left\{g_j\right\}_{j\in J}\right) \\ \left(\left\{x_k^i\right\}_{k\in\kappa}, \left\{x_j^i\right\}_{j\in J}, \left\{r_k^i\right\}_{k\in\kappa}, \left\{r_j^i\right\}_{j\in J}\right) \\ \left(C_i, \left\{U_k^i\right\}_{k\in\kappa}, \left\{U_j^i\right\}_{j\in J}\right) \end{array} \right\}.$$

In the first stage, the computing task scheduling optimal demands that $\{x_k^i\}_{k\in\kappa}, \{x_j^i\}_{j\in J}$ be obtained; in the second stage, the best reward mechanism $\{r_k^i\}_{k\in\kappa}, \{r_j^i\}_{j\in J}$ is determined by the CaaS provider under the maximization of the energy end user's utility function.

4) Game Equilibrium: The optimal solution of the Stackelberg game model is named the Stackelberg equilibrium. At this point, the leader (requesting devices) obtains the minimum edge computing service cost given the best response of followers (MS and FN). For the Stackelberg game model Π, the Stackelberg equilibrium is defined as follows.

Definition: A set of strategies $(\{(r_k^{i*}, x_k^{i*}), (r_j^{i*}, x_j^{i*})\}_{k\in\kappa, j\in J})$ is considered to reach the Stackelberg equilibrium only if it satisfies the following group of inequalities:

$$\left\{ \begin{array}{ll} \forall r_k^i, r_j^i, & C_i\left(\left\{\left(r_k^{i*}, x_k^{i*}\right), \left(r_j^{i*}, x_j^{i*}\right)\right\}_{k\in\kappa, j\in J}\right) \\ & \leq C_i\left(\left\{\left(r_k^i, x_k^{i*}\right), \left(r_j^i, x_j^{i*}\right)\right\}_{k\in\kappa, j\in J}\right) \\ \forall x_k^i, & U_k^i\left(r_k^{i*}, x_k^{i*}\right) \geq U_k^i\left(r_k^{i*}, x_k^i\right) \\ \forall x_j^i, & U_j^i\left(r_j^{i*}, x_j^{i*}\right) \geq U_j^i\left(r_j^{i*}, x_j^i\right). \end{array} \right. \tag{4}$$

The goal of the Stackelberg game model Π is to solve the unique Stackelberg equilibrium so that neither the CaaS provider (the agent of the requesting devices) nor the energy device has a motivation to change their decisions. At this point, no player can benefit from a unilateral change in strategy in terms of overall cost and personal utility. In other words, when all participants are in the Stackelberg equilibrium, the service provider cannot reduce the reward parameters from the Stackelberg equilibrium value (r_k^{i*}, r_j^{i*}) to help the requesting device reduce the service cost. Similarly, except for the

committed Stackelberg equilibrium value (x_k^{i*}, x_j^{i*}), no arbitrary device can further improve personal utility by changing different calculated load bearing capacities. The best response of the followers.

It is clear that the second-order derivatives U_k^i with respect to x_k^i are less than 0; i.e., the utility function is concave. Therefore, when $\partial U_k^i/\partial x_k^i = 0$, we can obtain the maximum value of the function, and the best response is as follows:

$$x_k^{i*} = \frac{r_k^i - c_k}{2\xi_k} = \frac{r_k^i - c_k}{2w_k s_{\max}^k}, r_k^i \geq c_k. \tag{5}$$

When $\partial U_j^i/\partial x_j^i = 0$, we obtain

$$x_j^{i*} = \frac{r_j^i - c_j}{2\xi_j} = \frac{r_j^i - c_j}{2w_j g_{\max}^j}, r_j^i \geq c_j. \tag{6}$$

Optimal cost of the leaders.

By substituting x_k^{i*} and x_j^{i*} into the end user's cost function C_i, the CaaS provider rewrites the cost minimization problem as

$$\begin{aligned}
\min_{r_k^i, r_j^i} \; & p_m \sum_{k\in\kappa} \frac{r_k^i - c_k}{2\xi_k} + p_f \sum_{j\in J} \frac{r_j^i - c_j}{2\xi_j} \\
& + \sum_{k\in\kappa} \frac{\left(r_k^i\right)^2 - r_k^i c_k}{2\xi_k} + \sum_{j\in J} \frac{\left(r_j^i\right)^2 - r_j^i c_j}{2\xi_j} \\
\text{s.t. } & \lambda_i L \leq \sum_{k\in\kappa} \frac{r_k^i - c_k}{2\xi_k} + \sum_{j\in J} \frac{r_j^i - c_j}{2\xi_j} \leq L \\
& r_k^i \geq c_k \\
& r_j^i \geq c_j.
\end{aligned} \tag{7}$$

The Lagrange multipliers α, β, γ, and μ are introduced into the inequality-constrained problem; then, the Lagrange function can be expressed as

$$\begin{aligned}
\mathscr{L} = & \left(p_m \sum_{k\in\kappa} \frac{r_k^i - c_k}{2\xi_k} + p_f \sum_{j\in J} \frac{r_j^i - c_j}{2\xi_j} \right. \\
& \left. + \sum_{k\in\kappa} \frac{\left(r_k^i\right)^2 - r_k^i c_k}{2\xi_k} + \sum_{j\in J} \frac{\left(r_j^i\right)^2 - r_j^i c_j}{2\xi_j} \right) \\
& - \alpha \left(\sum_{k\in\kappa} \frac{r_k^i - c_k}{2\xi_k} + \sum_{j\in J} \frac{r_j^i - c_j}{2\xi_j} - \lambda_i L \right) \\
& + \beta \left(\sum_{k\in\kappa} \frac{r_k^i - c_k}{2\xi_k} + \sum_{j\in J} \frac{r_j^i - c_j}{2\xi_j} - L \right) \\
& - \left(r_k^i - c_k\right) - \mu \left(r_j^i - c_j\right) \\
\text{s.t. } & \lambda_i L \leq \sum_{k\in\kappa} \frac{r_k^i - c_k}{2\xi_k} + \sum_{j\in J} \frac{r_j^i - c_j}{2\xi_j} \leq L \\
& r_k^i \geq c_k \\
& r_j^i \geq c_j.
\end{aligned} \tag{8}$$

Equation (8) shows that $\partial^2 \mathscr{L}/\partial (r_k^i)^2 = 1/\xi_k > 0$, $\partial^2 \mathscr{L}/\partial (r_j^i)^2 = 1/\xi_j > 0$. Equation (8) is a strictly convex optimization problem with linear constraints. In other words, under the condition that x_k^{i*} and x_j^{i*} are known, there are always unique r_k^{i*} and r_j^{i*} to be solved.

5) Game-Solving Gradient-Based Algorithm: In this article, we adopt an algorithm based on a gradient to find the solution of the strictly convex optimization problems mentioned above and to obtain r_k^{i*} and r_j^{i*}. Equation (8) can be rewritten as $\min_{r_k^{i*}, r_j^{i*}} \mathscr{L}(r_k^i, r_j^i, \alpha, \beta, \gamma, \mu)$, s.t. $\alpha, \beta, \gamma, \mu \geq 0$. The Lagrange multipliers above are updated in turn according to the following formula:

$$\begin{aligned}
\alpha^{l+1} &= \left| \alpha^l - \theta \left(\sum_{k\in\kappa} \frac{r_k^i - c_k}{2\xi_k} + \sum_{j\in J} \frac{r_j^i - c_j}{2\xi_j} - \lambda_i L \right) \right|^+ \\
\beta^{l+1} &= \left| \beta^l + \phi \left(\sum_{k\in\kappa} \frac{r_k^i - c_k}{2\xi_k} + \sum_{j\in J} \frac{r_j^i - c_j}{2\xi_j} - L \right) \right|^+ \\
\gamma^{l+1} &= \left| \gamma^l - \vartheta \left(r_k^i - c_k\right) \right|^+ \\
\mu^{l+1} &= \left| \mu^l - \sigma \left(r_j^i - c_j\right) \right|^+
\end{aligned} \tag{9}$$

where l indicates the updated rounds and θ, ϕ, ϑ, and σ represent the updated step size. In addition, $|\,.\,|^+ = \max(., 0)$. Regarding $(r_k^i)^{l+1}$ and $(r_j^i)^{l+1}$, we calculate these terms according to the KKT condition. From $\partial L/\partial r_k^i = 0$ and $\partial L/\partial r_j^i = 0$, we obtain

$$\begin{aligned}
\left(r_k^i\right)^{l+1} &= \frac{1}{2}\left(c_k + \alpha^l - \beta^l - p_m\right) + \xi_k \gamma^l \\
\left(r_j^i\right)^{l+1} &= \frac{1}{2}\left(c_j + \alpha^l - \beta^l - p_f\right) + \xi_j \mu^l.
\end{aligned} \tag{10}$$

According to the aforementioned updating expression, an iterative algorithm based on the gradient method is proposed (Algorithm 1) to determine the Stackelberg equilibrium. In the application, the CaaS provider acts as the agent of the requesting devices to negotiate with the MS or FN in each Stackelberg game. We input the data size of computing tasks and basic parameters, including the MS/FN charge of CaaS, risk factors bearing the computing task, and edge energy devices' preferences for offloading tasks. We then initialize the number of iterations, Lagrange multipliers, updated step size, etc. Then, we iteratively calculate the rewards of the MS and FN until the condition satisfies threshold ε; hence, the current rounds of the MS and FN rewards are the optimal rewards $r_k^{i*} = (r_k^i)^l$ and $r_j^{i*} = (r_j^i)^l$. Next, we can calculate the follower's optimal offloading tasks x_k^{i*} and x_j^{i*}. Finally, we output the computing load distribution and reward strategy of CaaS scheduling $\{(x_k^{i*}, x_j^{i*}), (r_k^{i*}, r_j^{i*})\}_{k\in\kappa, j\in J}$. The utilities of the participants are U_k^{i*}, U_j^{i*}, and C_i^*.

Theorem: There is a unique Stackelberg equilibrium between the CaaS provider and all edge energy devices.

Proof: Given the incentive mechanism, the edge energy devices act as followers. Since the utility function is a concave function, it always has optimal responses x_k^{i*} and x_j^{i*}. As a leader, based on the prediction of x_k^{i*} and x_j^{i*}, the CaaS

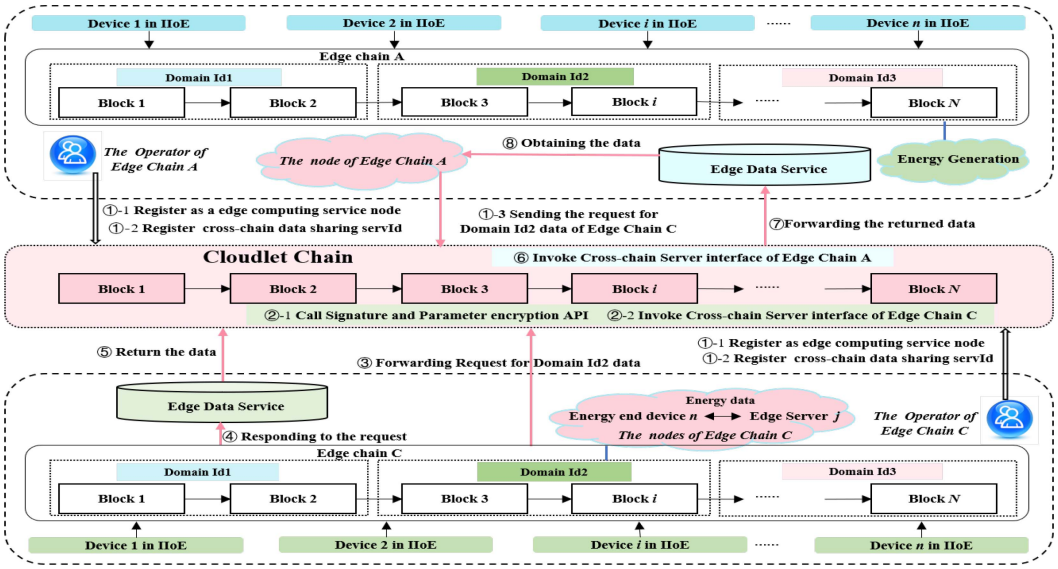

Fig. 7. Flow of cross-chain data sharing.

Algorithm 1 Gradient-Based Iterative Algorithm to Find the Equilibrium of the Stackelberg Game

Input: Edge computing tasks and corresponding parameters $p_m, p_f, \{h_i, D_i, \lambda_i\}, \{c_k, \omega_k, s^k_{\max}\}_{k\in\kappa}$ and $\left\{c_j, \omega_j, g^j_{\max}\right\}_{j\in J}$.

Output: Computing load distribution and reward strategy of CaaS scheduling $\left\{(x^{i}_{k}{}^*, x^{i*}_{j}), (r^{i}_{k}{}^*, r^{i*}_{j})\right\}_{k\in\kappa, j\in J}$; the utility of participants $U^{i}_{k}{}^*, U^{i*}_{j}, C_i{}^*$.

Initialize: $l = 0, \alpha^0, \beta^0, \gamma^0, \mu^0$, $r^{i}_{k}{}^0, r^{i0}_{j}$ and $\theta, \phi, \vartheta, \sigma$.

Calculate $L = \sum_{i=1}^{N} h_i D_i$, $\xi_k = w_k s^k_{\max}$, $\xi_j = w_j g^j_{\max}$.

While $l = l + 1$

The CaaS provider updates $\alpha^{l+1}, \beta^{l+1}, \gamma^{l+1}, \mu^{l+1}$ based on (9);

The CaaS provider updates $(r^i_k)^{l+1}, (r^i_j)^{l+1}$ according to (10);

Until $\left|(r^i_k)^l - (r^i_k)^{l-1}\right| \le \varepsilon$ and $\left|(r^i_j)^l - (r^i_j)^{l-1}\right| \le \varepsilon$

Determine the optimal reward: $r^{i}_{k}{}^* = (r^i_k)^l, r^{i*}_{j} = (r^i_j)^l$.

For all the edge energy devices in the set of N, do

according to Eqs. (5) and (6), calculate $x^{i}_{k}{}^*$ and x^{i*}_{j} based on $r^{i}_{k}{}^* = (r^i_k)^l, r^{i*}_{j} = (r^i_j)^l$.

If $L - \sum_{k\in\kappa} x^{i}_{k}{}^* - \sum_{j\in J} x^{i*}_{j} \neq 0$

Allocate the remaining calculation $L - \sum_{k\in\kappa} x^{i}_{k}{}^* - \sum_{j\in J} x^{i*}_{j}$ to the near edge energy devices.

end

With Eqs. (1)-(3), calculate the utility of the MS, FN and edge energy devices $U^{i}_{k}{}^*, U^{i*}_{j}, C_i{}^*$, respectively.

End

provider can make the optimization function L in (8), which is proven to be a strictly convex optimization problem. Therefore, the CaaS provider can establish the optimal solutions $r^{i}_{k}{}^*$ and r^{i*}_{j} by using Algorithm 1. For the leader, given the best strategies of all edge energy devices, the CaaS provider always has a unique optimal strategy. In the end, both leaders and followers are completely satisfied, and the decisions $\{(r^{i}_{k}{}^*, x^{i}_{k}{}^*), (r^{i*}_{j}, x^{i*}_{j})\}_{k\in\kappa, j\in J}$ maximize their respective utility simultaneously. In addition, all participants have their optimized payments and expenses. Considering the strategies chosen by other members in the game model, the members have no reason to change their decisions and take other actions. Therefore, in this game model, the unique Stackelberg equilibrium state represented by $\{(r^{i}_{k}{}^*, x^{i}_{k}{}^*), (r^{i*}_{j}, x^{i*}_{j})\}_{k\in\kappa, j\in J}$ can be achieved. ■

B. Cross-Chain Data Sharing

Assume that an edge server on edgechain A needs to have access to the data, specifically the data information in domain Id2 on edgechain C to improve the computing efficiency. The specific flow is shown in Fig. 7. When receiving the request of edgechain A, the service API of edgechain C on the cloudlet chain is invoked. Then, the request is forwarded until edgechain C responds to this request and returns the data by the hash encryption from domain Id2 to the cloudlet chain. Next, the service API of edgechain A is invoked for edge data sharing service. Finally, the data arrive at the requested edge server on edgechain A. Similarly, data sharing between any two edgechains can be realized by this approach.

C. Communication Complexity and Overhead

In our proposed system, the communication complexity and overhead are analyzed (Fig. 8). This involves tasks uploading, tasks scheduling, tasks offloading, tasks computing, and computing result storage.

S1 (Task Uploading:) The constrained edge energy devices N upload the tasks to the CaaS provider. During this progress, the communication complexity is $O(N)$, and the time overhead

Fig. 8. Communication flow of our proposed system.

is expressed as

$$T_i^{\text{tran}} = |D_i|/r_i = |D_i|/W\log_2(1+\text{SNR}) \tag{11}$$

where W_e is the bandwidth and the signal-to-noise ratio is SNR $\in$ [70, 110] dB.

S2 (Task Scheduling): The CaaS provider determines the optimal offloading and rewards between the requesters and MS/FN according to the two-stage Stackelberg game. The time overhead and complexity are T^{game} and $O(N)$, respectively.

S3 (Task Offloading): After obtaining the scheduling strategy, the provider conducts tasks assignment, offloading the computing data to the MS/FN near requested devices in the same domain. The communication complexity is $O(N)$. Similarly, the time cost is T_i^{tran}.

S4 (Task Computing): The designated task undertaker accomplishes the assignment. The communication complexity is $O(N)$ (because the number of MSs and FNs is less than the large number of edge energy devices N). The time overhead is

$$T_i^{\text{com}} = I\frac{c_e|D_i|}{f_e}. \tag{12}$$

Here, $I \in (0, 1)$ means the computing accuracy, and the data size to be calculated for the edge energy device is $|D_i|$, c_e indicates the number of CPU cycles for the MSs/FNs to process per data size, and f_e means the CPU frequency of MSs/FNs.

S5 (Computing Result Storage): Once the MSs/FNs finish the calculation task, they receive the previously determined rewards by game. For the computing results, the consensus mechanism is employed to ensure trusted data storage and tamper resistance. For PBFT, the primary node accepts the consensus request from the client, and then it sends the data to the remaining $(\mathbb{N}-1)$ nodes for voting [Fig. 9(a)]; hence, the interaction complexity is $O(\mathbb{N}^2)$.

To decrease the communication interaction to $O(\mathbb{N})$ while considering the multichain consensus in our devised system, the DLPBFT consensus method is applied, as shown in Fig. 9(b).

The time of consensus is $T_i^{\text{cons}} = t_{\text{store}} - t_{\text{vote}} = t_3{}' - t_1$; t_{store} means the block time stored in the chain, and t_{vote} indicates the time to start voting.

The leader of the edgechain is selected according to the edge node's comprehensive ability in terms of computing, communicating, and caching. Then, the second-level consensus is implemented among each edgechain leader.

Based on the above analysis, the total communication complexity is $O(N)$ (since the number of blockchain consensus nodes is less than the large number of edge energy devices N), and the total time spent is

$$T = 2T_i^{\text{tran}} + T^{\text{game}} + T_i^{\text{com}} + T_i^{\text{cons}}. \tag{13}$$

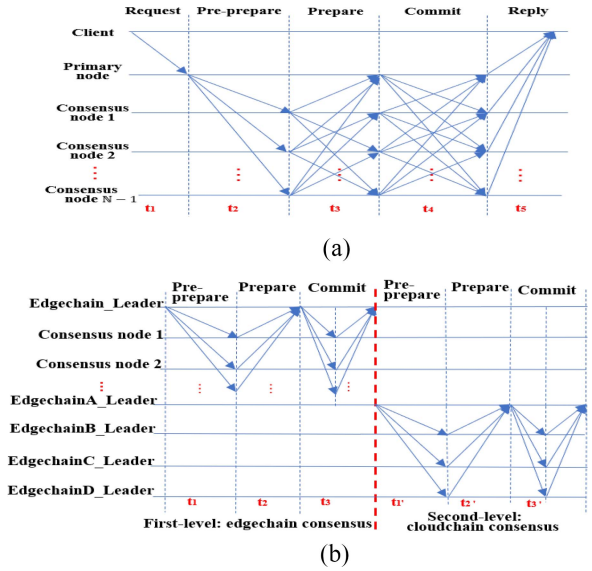

Fig. 9. Consensus process comparison. (a) PBFT. (b) DLPBFT.

D. Tamper Resistance

In our proposed multiedgechain structure, every edgechain $E_{l_A}, E_{l_B}, E_{l_C}, E_{l_D}$ is responsible for the data storage of the corresponding domain, including energy trading, intelligent manufacturing, and electric vehicles in the IIoE. Each edge block format is E_l = <pre_hash, h_{E_l}, t, m_root, Sig, Id_device, Id_MS/FN>. The Ids of four edgechains and the summary info of the cloud computing results are stored on the cloudlet chain C_l = <Id(E_l), h_c, t_c>. The purpose of this design lies in data isolation and classified storage for security, and convenient management as well as promoting the data processing speed in parallel.

When CaaS scheduling is finished, the related data and computing results are broadcasted in the corresponding type of edgechain (or the two edgechains for CaaS scheduling between two edgechains). In the edgechain, consensus nodes of this period collect all the corresponding types of consensus requests within this time and then produce a block. After the edge block is constructed, the builder of the edge block is responsible for raising another consensus request and broadcasting in the cloudlet chain. The voting representative nodes (namely, the builders of the four edgechain blocks) of the cloud chain reach a consensus on information submitted by all creators of edge blocks within one block time and store it in the next cloud block attaching the edgechain Id of the data source. In this way, the construction of the cloud chain block is completed.

The trusted storage of a multichain structure adopts a cost-based approach to ensure that the blockchain model cannot be tampered with, but its security is further improved over that of the single chain structure. If a malicious node seeks to modify any block, it needs to not only modify all subsequent

blocks under the chain but also alter the subsequent blocks of all other chains related to the block. Assuming that the height of block m tampered with by malicious node in C_l is h_{c_m}, the block height in $E_l = E_{l_A}, E_{l_B}, E_{l_C}, E_{l_D}$ storing related and more detailed computing data is $h_{E_m} = h_{E_{mA}}, h_{E_{mB}}, h_{E_{mC}}, h_{E_{mD}}$, the current height of C_l is h_c, and the current height of E_l is $h_{E_l} = \{h_{E_{l_A}}, h_{E_{l_B}}, h_{E_{l_C}}, h_{E_{l_D}}\}$. Suppose that when falsifying block m, the minimum number of blocks to be modified on a single node is as follows:

$$\begin{aligned} B_{\text{num}} = &\left(h_{E_{l_A}} - h_{E_{mA}}\right) \times n_{E_{l_A}} + \left(h_{E_{l_B}} - h_{E_{mB}}\right) \times n_{E_{l_B}} \\ &+ \left(h_{E_{l_C}} - h_{E_{mC}}\right) \times n_{E_{l_C}} + \left(h_{E_{l_D}} - h_{E_{mD}}\right) \times n_{E_{l_D}} \\ &+ \left(h_c - h_{c_m}\right) \times \mathbb{N} \end{aligned} \tag{14}$$

where $n_{E_{l_A}}, n_{E_{l_B}}, n_{E_{l_C}}$, and $n_{E_{l_D}}$ are the blockchain node numbers of $E_{l_A}, E_{l_B}, E_{l_C}$, and E_{l_D}, respectively. Because the verification block is saved in the whole network, $\mathbb{N}$ represents the total node numbers in the edge–cloudlet chain.

To tamper with a certain block, at least B_{num} related blocks need to be tampered with in the whole network. In this case, modifying any data requires modifying almost all network data, incurring a very large cost. In the majority of cases, the price is far greater than the benefits of tampering. Moreover, under the normal operation of the whole network, the tampering cost increases with time, and the growth rate is much higher than that of a single chain structure. Only by controlling the voting rights of all edgechains and cloud chain can the attacker tamper with the block data, which means that the attacker must control almost all nodes in the system. Based on this, the multichain structure cannot be tampered with under the condition that a certain number of trusted nodes can complete the consensus normally.

IV. Simulation Experiments and Results Analysis

First, we analyze the utility of the offloading computation with several sets of experimental parameters in MATLAB for the proposed Stackelberg game-based CaaS scheme. Then, both computational result storage and cross-chain data sharing are implemented on the multichain structure, which is most applicable to the increasing number of edge devices and applications in the IIoE. Four edgechains are deployed on a distributed local area network and require a Core CPU with 32-GB memory. Here, the total number of nodes in each private edgechain is set to 5, and the number of nodes in each consortium edgechain is set to 20. In particular, edgechain A acts as a private blockchain and edgechain C acts as a consortium blockchain. All the nodes on both edgechains are equipment with JDK 1.8, distributed data storage MySQL 5.7. In addition, Spring is adopted as the underlying framework. GRPC serves as the communication protocol among blockchain nodes on the same edgechain.

A. Security Analysis

1) Asymmetric Cryptographic Algorithm Based on ECC: An elliptic curve $Ep(a, b)$ is a set defined as $y^2 = x^3 + ax + b$, $a \geq 0, b \geq 0$, $x, y \in [0, p-1]$ (where p is a prime). A cyclic group G is formed by all points on the elliptic curve and the infinite point O [47]. Let $G1$ and $G2$ be two cyclic groups with the same prime order q. $G1$ and $G2$ are the additive cyclic group and multiplicative cyclic group, respectively. We assume that $e : G1 \times G1 \rightarrow G2$ has the bilinear map's properties of nondegeneracy, bilinearity, and computability [47].

Parameter Establishment: It is assumed that with a secure parameter k, an edge energy node selects two groups G_1 and G_2 with the same prime order q. In addition, e is a bilinear mapping. We randomly select a generator $P \in G_1$. Then, a number $s \in Z_q^*$ is chosen as the node's master private key, and the public key is calculated according to $P_{\text{pub}} = s \cdot P$. The pair of keys is used to encrypt and decrypt computing messages preventing malicious manipulation. To protect the identity anonymity and message security of energy terminals, two secure hash functions are selected: 1) $H_1 : \{0,1\}^* \rightarrow G_1$ and 2) $H_2 : \{0,1\}^* \rightarrow Z_q^*$, respectively. Public parameters $\{G_1, G_2, e, n, q, P, P_{\text{pub}}, h_1, h_2\}$ are broadcast among nodes.

Public–Private Key Pair Generation of Energy Devices: By randomly selecting a number $s_i \in Z_q^*$ as the private key, the energy terminal device calculates the corresponding public key $P_i = s_i \cdot P$ so that signatures can be generated and identity can be verified.

Note that the public parameters must be kept confidential. Then, the energy terminals and MSs/FNs can confirm the reliability of the received messages. Furthermore, they send encrypted messages with their signatures during the calculation results (namely, block) verification. Due to the inability to solve the key mathematical problem, it is nearly impossible for attackers to compute correct digital signatures given ECC results. With asymmetric cryptography, the blockchain guarantees the integrity and security of computing messages in the broadcasting.

2) DLPBFT Consensus Mechanism: In the proposed multichain structure, the DLPBFT consensus algorithm is implemented to vote for the calculation messages, including data and results after game optimization, and then provides traceability and secure data confidentiality. In the first level of DLPBFT, we assume that the number of edgechain nodes is n_{E_l}. For each round of consensus, we select the edgechain primary peer $n_{E_{l_p}} (p = (h \bmod n_{E_l}) + 1)$ based on the node's comprehensive ability ranking. The top 80% of nodes (except for the master node) are ordinary consensus nodes and vote for the received message from master peer broadcasting; the remaining nodes serve as the storage node. In the second level of DLPBFT, the consensus nodes consist of each edgechain's primary peers so that the trust of nodes is guaranteed, and the consensus efficiency can be improved as a result of reducing the number of consensus nodes.

Moreover, during each round of the consensus process, voting is divided into two phases: 1) reserve block voting and 2) formal block voting. First stage: the receiving block message is expressed as $<\text{preblock}, \text{Sig}_p>$ (Sig_p is the signature of the master peer). If the master peer receives votes from over $2f + 1$ ($f = |(n_{E_l} - 1)/3|$ represents the maximum number of malicious peers) different ordinary nodes, then it enters the second stage. It then broadcasts the formal block $<\text{pre\_hash}, h, t, m\_\text{root}, \text{Sig}_{od}>$ messages among the others. (Here, *pre_hash* is the previous block hash, t is

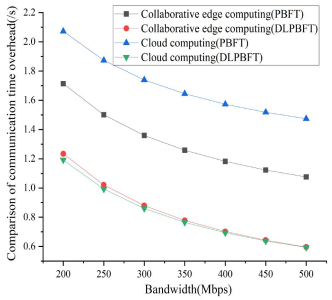

Fig. 10. Comparison of communication time overhead.

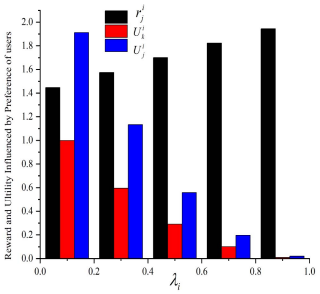

Fig. 11. Impact of preferences.

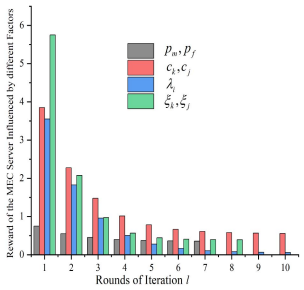

Fig. 12. Iterations of MSs reward.

the timestamp, *m_root* represents the Merkel root, which comprises hashed transaction, and Sig_{od} stands for the signature of ordinary peer.) As long as the primary node receives over $2f + 1$ votes, consensus is reached, generating a new block on the edgechain. Otherwise, the block is abandoned, and a new consensus round starts. Based on this trusted and fault-tolerant consensus mechanism, voting can securely operate even though malicious peers exist.

3) Data Confidentiality and Tamper-Proofness: It is well known that the data confidentiality of blockchain depends upon the consensus mechanism and encrypted storage. The blockchain storage structure comprises chronological blocks in which each block includes the cryptographic hash of the previous block. Therefore, it is difficult to falsify one block without tampering with all the subsequent blocks. Moreover, the computing data contained in a block are encrypted with an asymmetric encryption algorithm. It would take a large amount of resources to decrypt the encrypted data without knowing the private key.

4) Transparency and Traceability: The transparency of blockchain technology is ensured by permitting all users and the edge computing provider to have access to the blockchain

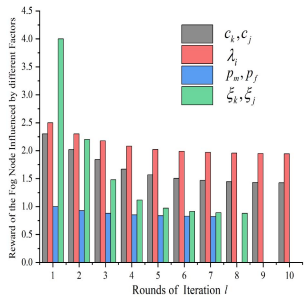

Fig. 13. Iterations of FNs reward.

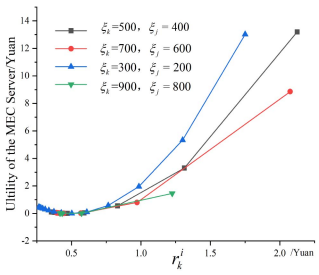

Fig. 14. Utilities of the MS with various risk factors.

and monitor the corresponding data. That is, the calculated data are not stored by one single node but are open to all entities in the consortium. As a result, any malicious data modification is noticeable and traceable.

B. Performance Analysis

1) Communication Overhead: In contrast with the cloud computing mode, which uploads all the computing tasks from the edge energy devices to the remote cloud server, the advantage of collaborative edge computing in the edge range lies in its low time delay and fast response for the calculation requests of edge energy IoT devices. Under the analysis in Section III, $W_e \in [200, 500]$ Mb/s is the 5G bandwidth, the data size to be calculated for the edge energy device is $|D_i| = 200$ MB, and the CPU frequency of each MS /FN c_e, f_e is 3 GHz. The total time consumption of our proposed scheme and cloud computing is shown in Fig. 10. Subjected to the transmission distance as well as signal interference factors, the average communication overhead of collaborative edge computing is 35.5% less than that of cloud computing when adopting the DLPBFT algorithm. Moreover, even in collaborative edge computing, the average time overhead of PBFT is 20% higher than that of DLPBFT. The reduced consensus time complexity indicates that the consensus efficiency has been improved.

2) Utility Analysis: Because of the limited computing capacity of edge devices, it is optimal for the MSs and FNs to undertake the computation-intensive tasks offloaded by the end devices. Combined with the Stackelberg game-theoretical approach, the offloading cost of the edge energy devices can be minimized and the utilities of the MS/FN are maximized. According to Section III, as a leader in the Stackelberg game, optimal offloading tasks of edge energy devices rely on the follower MS and FN reward strategy solved by the gradient-based method. After acquiring the rewards, the utility of the

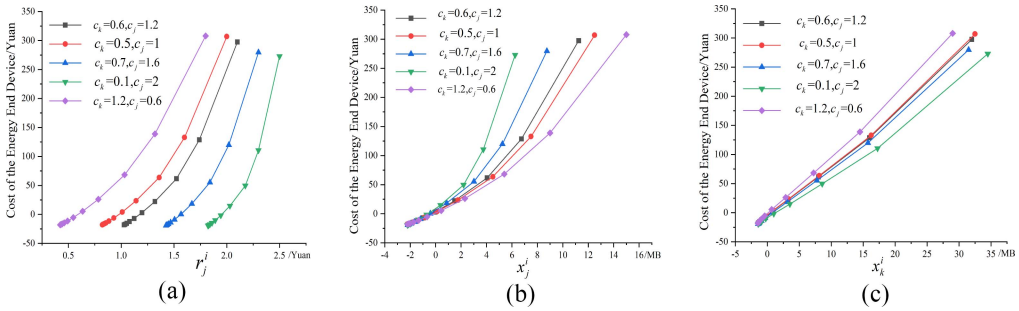

Fig. 15. Relationship between the unit cost of the MS or FN bearing the computation and the corresponding rewards as well as providing demands. (a) Utilities vary with FN rewards. (b) Utilities vary with FN's bearing tasks. (c) Utilities vary with MS's bearing tasks.

MS/FN and the offloading cost of the edge terminals can be obtained. Considering the requester's preferences and the computing provider's risk factors, we set the range of preference, unit cost, risk factors, and charge of offloading computation as $\lambda_i \in [0, 1]$, $c_k, c_j \in [0.1, 1.5]$, $\xi_k, \xi_j \in [100, 1000]$, and $p_m, p_f \in [1, 10]$, respectively.

Fig. 11 illustrates the impact of preferences on the unit reward and the utilities of the MS and FN. With the growth of the preference value, the unit incentive provided by the edge device becomes stronger. Although the unit reward is raised, the utilities of the FN and MS are decreased. This reflects that the computation offloading requester's preference has a great influence on the provider's utilities. Thus, the CaaS provider should consider the profits of participants as well as the user's preference and rationally allocate the computing tasks.

Figs. 12 and 13 show the iterations of rewards r_k^i and r_j^i influenced by different parameters, respectively. To carry out a reasonable analysis, we compare the rounds of iterations of the rewards r_k^i and r_j^i under the different charges of CaaS, unit cost, preferences, and risk factors when bearing the offloading tasks. From Fig. 12, the iteration of the rewards for the MEC server converges in rounds 7, 10, 6, and 8 under the influence of the charge, unit cost, preference, and risk factors, respectively. The performance is faster than that of [48], converging at round 18. The number of iterations of the FN's reward received from the end energy device converges at rounds 7, 10, 10, and 8 under the influence of the charge, unit cost, preference, and risk factors, respectively. This proves that the proposed gradient-based iterative algorithm is effective in seeking game equilibria.

We consider a scenario, where the utilities of the MS and FN have various factors of risk ξ_k and ξ_j and vary with the different rewards r_k^i and r_j^i and offloading tasks x_k^i and x_j^i. As shown in Fig. 14, there is a downward trend for the utility of the MS when it is faced with increased risk factors. Specifically, the minimum risk factors $\xi_k = 300$ and $\xi_j = 200$ lead to the greatest benefits for the MS, and the largest risk factors $\xi_k = 900$ and $\xi_j = 800$ appear to be the lowest profits curve considering the reward change. On the one hand, under the same risk factor, more rewards r_k^i assigned by the CaaS provider contribute to the increased revenue.

In addition, Fig. 15(a)–(c) shows the relationship between the unit cost of the MS/FN and the offloading tasks sent by the energy end device. As the unit costs of the MS and FN decline, both the MS and FN tend to undertake more offloading computation x_k^i and x_j^i; hence, the corresponding rewards gradually increase.

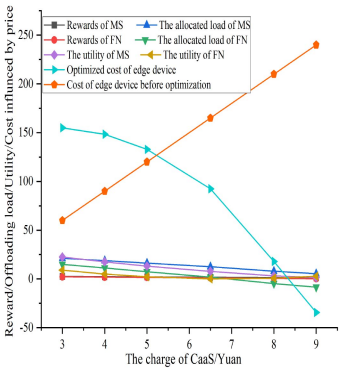

Fig. 16. Different parameters influenced by CaaS charges.

Finally, we discuss the impact of different charges on MS/FN rewards, offloading loads, and profits. As shown in Fig. 16, before game optimization, the cost of the end device linearly increases. With a higher charge for the edge computing services p_m and p_f, the rewards r_k^i and r_j^i, number of calculated tasks x_k^i and x_j^i, and utilities U_k^i and U_j^i decrease. This is because the improved price reduces the willingness of the edge device to offload computing tasks. Furthermore, although the CaaS price increases, it is clear that the cost of the end device C_i is reduced by our proposed optimized approach.

3) Difficulty of Block Tampering: Assuming that there are four edgechains $E_l = E_{l_A}, E_{l_B}, E_{l_C}, E_{l_D}$, the current height of each edgechain and cloudchain is 30, and the height of block m tampered with by malicious node is 10. Based on the analysis in Section III, we can calculate the number of blocks to be tampered with for a single node in the single-edge blockchain and multichain structures. Fig. 17 shows the comparison of the number of blocks to be modified between the single blockchain and multichain structures. The more nodes there are in each edgechain, the more difficulty the malicious node faces in modifying blocks. However, when comparing the number of blocks tampered with, it is not hard to find that our designed multichain structure is more difficult to modify than single chain and double blockchains.

4) TPS of the Edgechain: For one edgechain, we adopt the DLPBFT consensus algorithm to maintain data consistency of

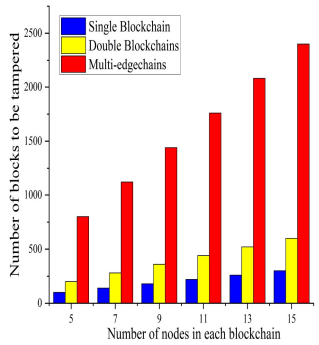

Fig. 17. Comparison of the number of blocks to be tampered.

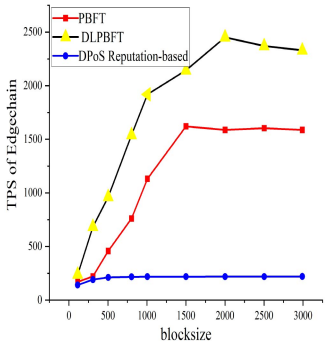

Fig. 18. TPS of the edgechain.

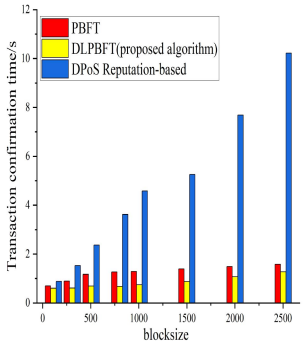

Fig. 19. Comparison of consensus algorithms.

computing. If there is only one blockchain in the IIoE, it is clear that the tolerable number of malicious nodes is nonnegligible and is in fact large for a high rate of energy terminal nodes joining the blockchain system. At this time, the advantages of the multiedgechain structure we designed emerge. Since each edgechain can operate independently and achieve consensus by itself, the TPS of the edgechain is significantly improved over the situation of one blockchain, and the security is ensured because the number of malicious nodes declines to an acceptable level. We now analyze the performance of the edgechain when we utilize the DLPBFT. As shown in Fig. 18, to guarantee that the TPS is not affected, the optimal block size is 2000 transactions, which is larger than the 1500 transactions of the PBFT consensus algorithm. Moreover, the optimal block size of the DPoS reputation-based consensus algorithm in [49] is nearly one-quarter that of the DLPBFT. Fig. 19 shows that the transaction confirmation time of PBFT is greater than that

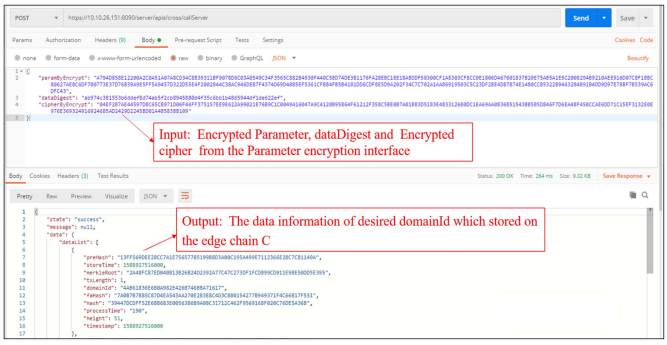

Fig. 20. Realization of cross-chain edge data sharing.

of our proposed algorithm. Additionally, the DPoS reputation-based confirmation time is eight times that of DLPBFT as the blocksize (meaning the number of transactions included in a block) increases.

5) Cross-Chain Edge Data Sharing: Herein, we use Postman to verify cross-chain interface call testing (Fig. 20). Through the proposed methods, the edge data of the desired sharing in one edgechain can be called and read by the requesters from other edgechains.

V. Conclusion

In this article, we proposed a structure of multiedgechain for edge devices dealing with real-time computing tasks in the IIoE. The proposed framework, which reduces the storage burden of a single blockchain and improves the computing efficiency, enables the edge devices and applications to reliably store the computing information. Then, the problem of offloading computation in collaborative edge computing was modeled as a two-stage leader–follower game. The gradient-based iterative algorithm was applied to obtain the optimal reward and a computational offloading strategy under the various impact indices including preferences and risk factors. We also considered the interoperability among heterogeneous edgechains and edge–cloud collaboration in the IIoE. Therefore, a method of cross-chain edge data sharing by cloudlet chain is achieved. The simulation results validated the theoretical analysis and the effectiveness of the proposed algorithms. In future studies, we plan to specifically analyze the energy consumption for the proposed architecture and consider the optimized blockchain node classification methods. There are potential directions worth exploring in cross-chain energy trading and other digital asset trading among different fields.

References

[1] X. Li, J. Wan, H. N. Dai, M. Imran, M. Xia, and A. Celesti, "A hybrid computing solution and resource scheduling strategy for edge computing in smart manufacturing," *IEEE Trans. Ind. Informat.*, vol. 15, no. 7, pp. 4225–4234, Jul. 2019.

[2] C. Yu, B. Lin, P. Guo, W. Zhang, S. Li, and R. He, "Deployment and dimensioning of fog computing-based Internet of Vehicle infrastructure for autonomous driving," *IEEE Internet Things J.*, vol. 6, no. 1, pp. 149–160, Feb. 2019.

[3] R. Yang, F. R. Yu, P. Si, Z. Yang, and Y. Zhang, "Integrated blockchain and edge computing systems: A survey, some research issues and challenges," *IEEE Commun. Surveys Tuts.*, vol. 21, no. 2, pp. 1508–1532, 2nd Quart., 2019.

[4] J. Xu, B. Palanisamy, H. Ludwig, and Q. Wang, "Zenith: Utility-aware resource allocation for edge computing," in *Proc. IEEE Int. Conf. Edge Comput. (EDGE)*, 2017, pp. 47–54.

[5] L. Yin, J. Luo, and H. Luo, "Tasks scheduling and resource allocation in fog computing based on containers for smart manufacturing," *IEEE Trans. Ind. Informat.*, vol. 14, no. 10, pp. 4712–4721, Oct. 2018.

[6] M. Li, N. Cheng, J. Gao, Y. Wang, L. Zhao, and X. Shen, "Energy-efficient UAV-assisted mobile edge computing: Resource allocation and trajectory optimization," *IEEE Trans. Veh. Technol.*, vol. 69, no. 3, pp. 3424–3438, Mar. 2020.

[7] J. Fang and A. Ma, "IoT application modules placement and dynamic task processing in edge-cloud computing," *IEEE Internet Things J.*, early access, Jul. 7, 2020, doi: 10.1109/JIOT.2020.3007751.

[8] S. Goudarzi, M. H. Anisi, H. Ahmadi, and L. Mousavian, "Dynamic resource allocation model for distribution operations using SDN," *IEEE Internet Things J.*, vol. 8, no. 2, pp. 976–988, Jan. 2021.

[9] Q. Luo, C. Li, T. H. Luan, and W. Shi, "Collaborative data scheduling for vehicular edge computing via deep reinforcement learning," *IEEE Internet Things J.*, vol. 7, no. 10, pp. 9637–9650, Oct. 2020.

[10] 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 Trans. Veh. Technol.*, vol. 69, no. 5, pp. 5549–5561, May 2020.

[11] Q. Xu, Z. Su, Q. Zheng, M. Luo, B. Dong, and K. Zhang, "Game theoretical secure caching scheme in multihoming edge computing-enabled heterogeneous networks," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 4536–4546, Jun. 2019.

[12] Y. Chen, Z. Li, B. Yang, K. Nai, and K. Li, "A Stackelberg game approach to multiple resources allocation and pricing in mobile edge computing," *Future Gener. Comput. Syst.*, vol. 108, pp. 273–287, Jul. 2020.

[13] Y. Bai, L. Chen, L. Song, and J. Xu, "Risk-aware edge computation offloading using Bayesian Stackelberg game," *IEEE Trans. Netw. Serv. Manag.*, vol. 17, no. 2, pp. 1000–1012, Jun. 2020.

[14] Z. Chang, W. Guo, X. Guo, Z. Zhou, and T. Ristaniemi, "Incentive mechanism for edge-computing-based blockchain," *IEEE Trans. Ind. Informat.*, vol. 16, no. 11, pp. 7105–7114, Nov. 2020.

[15] B. Huang *et al.*, "BPS: A reliable and efficient pub/sub communication model with blockchain-enhanced paradigm in multi-tenant edge cloud," *J. Parallel Distrib. Comput.*, vol. 143, pp. 167–178, Sep. 2020.

[16] S. Zhang and J.-H. Lee, "A group signature and authentication scheme for blockchain-based mobile-edge computing," *IEEE Internet Things J.*, vol. 7, no. 5, pp. 4557–4565, May 2020.

[17] Q. Kong, L. Su, and M. Ma, "Achieving privacy-preserving and verifiable data sharing in vehicular fog with blockchain," *IEEE Trans. Intell. Transp. Syst.*, early access, Apr. 16, 2020, doi: 10.1109/TITS.2020.2983466.

[18] A. Back *et al.*, "Enabling blockchain innovations with pegged sidechains," 2014.

[19] A. Singh, K. Click, R. M. Parzi, Q. Zhang, A. Dehghantanha, and K. K. R. Choo, "Sidechain technologies in blockchain networks: An examination and state-of-the-art review," *J. Netw. Comput. Appl.*, vol. 149, Jan. 2020, Art. no. 102471.

[20] M. Li, H. Tang, A. R. Hussein, and X. Wang, "A sidechain-based decentralized authentication scheme via optimized two-way peg protocol for smart community," *IEEE Open J. Commun. Soc.*, vol. 1, pp. 282–292, 2020.

[21] E. Zaghloul, K. Zhou, and J. Ren, "P-MOD: Secure privilege-based multilevel organizational data-sharing in cloud computing," *IEEE Trans. Big Data*, vol. 6, no. 4, pp. 804–815, Dec. 2020.

[22] H. He *et al.*, "Joint operation mechanism of distributed photovoltaic power generation market and carbon market based on cross-chain trading technology," *IEEE Access*, vol. 8, pp. 66116–66130, 2020.

[23] N.-Y. Lee, J. Yang, M. M. H. Onik, and C.-S. Kim, "Modifiable public blockchains using truncated hashing and sidechains," *IEEE Access*, vol. 7, pp. 173571–173582, 2019.

[24] I. A. Seres, L. Gulyás, D. A. Nagy, and P. Burcsi, *Topological Analysis of Bitcoin's Lightning Network*, 2019.

[25] X. Xu, X. Zhang, H. Gao, Y. Xue, L. Qi, and W. Dou, "Become: Blockchain-enabled computation offloading for IoT in mobile edge computing," *IEEE Trans. Ind. Informat.*, vol. 16, no. 6, pp. 4187–4195, Jun. 2020.

[26] P. Dong, X. X. Wang, J. J. P. C. Rodrigues, and Z. Ning, "Deep reinforcement learning for vehicular edge computing: An intelligent offloading system," *ACM Trans. Intell. Syst. Technol.*, vol. 10, no. 6, p. 60, 2019.

[27] X. Qiu, L. Liu, W. Chen, Z. Hong, and Z. Zheng, "Online deep reinforcement learning for computation offloading in blockchain-empowered mobile edge computing," *IEEE Trans. Veh. Technol.*, vol. 68, no. 8, pp. 8050–8062, Aug. 2019.

[28] E. Chang, K. Y. Chan, P. Clark, and V. Potdar, "Guest editorial: Blockchain and AI enabled 5G mobile edge computing," *IEEE Trans. Ind. Informat.*, vol. 16, no. 11, pp. 7067–7069, Nov. 2020.

[29] K. Zhang, Y. Zhu, S. Maharjan, and Y. Zhang, "Edge intelligence and blockchain empowered 5G beyond for the industrial Internet of Things," *IEEE Netw.*, vol. 33, no. 5, pp. 12–19, Oct. 2019.

[30] Y. Jiao, P. Wang, D. Niyato, and Z. Xiong, "Social welfare maximization auction in edge computing resource allocation for mobile blockchain," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Kansas City, MO, USA, 2018, pp. 1–6.

[31] N. C. Luong, Z. Xiong, P. Wang, and D. Niyato, "Optimal auction for edge computing resource management in mobile blockchain networks: A deep learning approach," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Kansas City, MO, USA, 2018, pp. 1–6.

[32] Z. Xiong, S. Feng, D. Niyato, P. Wang, and Z. Han, "Optimal pricing-based edge computing resource management in mobile blockchain," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Kansas City, MO, USA, 2018, pp. 1–6.

[33] A. Asheralieva and D. Niyato, "Distributed dynamic resource management and pricing in the IoT systems with blockchain-as-a-service and UAV-enabled mobile edge computing," *IEEE Internet Things J.*, vol. 7, no. 3, pp. 1974–1993, Mar. 2020.

[34] L. Duan, L. Gao, and J. Huang, "Cooperative spectrum sharing: A contract-based approach," *IEEE Trans. Mobile Comput.*, vol. 13, no. 1, pp. 174–187, Jan. 2014.

[35] A. Fu, S. Yu, Y. Zhang, H. Wang, and C. Huang, "NPP: A new privacy-aware public auditing scheme for cloud data sharing with group users," *IEEE Trans. Big Data*, early access, May 5, 2017, doi: 10.1109/TBDATA.2017.2701347.

[36] Y. Pu, C. Hu, S. Deng, and A. Alrawais, "R^2PEDS: A recoverable and revocable privacy-preserving edge data sharing scheme," *IEEE Internet Things J.*, vol. 7, no. 9, pp. 8077–8089, Sep. 2020.

[37] H. Deng, Z. Qin, L. Sha, and H. Yin, "A flexible privacy-preserving data sharing scheme in cloud-assisted IoT," *IEEE Internet Things J.*, vol. 7, no. 12, pp. 11601–11611, Dec. 2020.

[38] S. Qi, Y. Lu, Y. Zheng, Y. Li, and X. Chen, "Cpds: Enabling compressed and private data sharing for industrial Internet of Things over blockchain," *IEEE Trans. Ind. Informat.*, vol. 17, no. 4, pp. 2376–2387, Apr. 2021.

[39] S. Nakamoto. *Bitcoin: A Peer-to-Peer Electronic Cash System*. Accessed: Oct. 31, 2008. [Online]. Available: https://bitcoin.org/bitcoin.pdf

[40] M. B. Mollah *et al.*, "Blockchain for future smart grid: A comprehensive survey," *IEEE Internet Things J.*, vol. 8, no. 1, pp. 18–43, Jan. 2021.

[41] L. Chen, L. Xu, N. Shah, Z. Gao, Y. Lu, and W. Shi, "On security analysis of proof-of-elapsed-time (poET)," in *Proc. Int. Symp. Stabilization Safety Security Distrib. Syst.*, 2017, pp. 282–297.

[42] D. Larimer. *Delegated Proof of Stake Blockchain*. Accessed: Apr. 3, 2014. [Online]. Available: https://en.bitcoinwiki.org/wiki/DPoS

[43] L. Lamport, R. Shostak, and M. Pease, "The Byzantine generals problem," *ACM Trans. Program. Lang. Syst.*, vol. 4, no. 3, pp. 382–401, 1982.

[44] A. Kiayias, A. Russell, B. David, and R. Oliynykov, *Ouroboros: A Provably Secure Proof-of-Stake Blockchain Protocol*, 2017.

[45] Y. Gilad, R. Hemo, S. Micali, G. Vlachos, and N. Zeldovich. (2017). *Algorand: Scaling Byzantine Agreements for Cryptocurrencies*. [Online]. Available: https://people.csail.mit.edu/nickolai/papers/gilad-algorand.pdf

[46] T. Rocketet, "Snowflake to avalanche: A novel metastable consensus protocol family for cryptocurrencie," 2018.

[47] S. Guo, X. Hu, S. Guo, X. Qiu, and F. Qi, "Blockchain meets edge computing: A distributed and trusted authentication system," *IEEE Trans. Ind. Informat.*, vol. 16, no. 3, pp. 1972–1983, Mar. 2020.

[48] X. Huang, "Research on resource and service optimization in edge computing of Internet of Vehicles," Ph.D. dissertation, School Autom., Guangdong Univ. Technol., Guangzhou, China, 2019.

[49] F. Wei, "Research and application of the improved DPoS blockchain consensus mechanism," Ph.D. dissertation, School info., Yunnan Univ. Finance Econ., Kunming, China, 2020.

Fenhua Bai (Graduate Student Member, IEEE) received the B.S. degree from the Faculty of Information Engineering and Automation, Kunming University of Science and Technology, Kunming, China, in 2017, where she is currently pursuing the Ph.D. degree.

Her current research interests include blockchains, edge computing, games application, and IoT.

Tao Shen (Member, IEEE) received the Ph.D. degree from the Illinois Institute of Technology, Chicago, IL, USA, in 2013.

He is a Professor and the Deputy Dean of the College of Information Engineering and Automation, Kunming University of Science and Technology, Kunming, China. His research interests cover blockchain technology, artificial intelligence, and the Internet of Energy.

Zhuo Yu received the Ph.D. degree from Beijing University of Aeronautics and Astronautics, Beijing, China, in 2011.

He is a Senior Engineer with State Grid Information and Telecommunication Company, Ltd., Beijing. His research directions include blockchains, artificial intelligence, VR/AR, GIS, BPM, and information consulting.

Kai Zeng received the Ph.D. degree from the University of Electronic Science and Technology of China, Chengdu, China, in 2015.

He is currently an Associate Professor with Kunming University of Science and Technology, Kunming, China. His research interests include distributed computing, granular computing, and deep learning.

Bei Gong (Member, IEEE) received the Ph.D. degree from Beijing University of Technology, Beijing, China, in 2012.

He has published six national invention patents and one monograph textbook. In the past five years, he has published more than 30 papers in first-class SCI/EI and other internationally famous journals and top international conferences in relevant research fields. He has presided over eight national projects, such as the National Natural Science Foundation, and six provincial and ministerial projects, such as for the general science and technology program of the Beijing Municipal Education Commission. His research interests include trusted computing, Internet-of-Things security, the mobile Internet of Things, and mobile-edge computing.