Sharded Blockchain for Collaborative Computing in the Internet of Things: Combined of Dynamic Clustering and Deep Reinforcement Learning Approach

Zhaoxin Yang, Ruizhe Yang, F. Richard Yu, *Fellow, IEEE*, Meng Li, *Member, IEEE*, Yanhua Zhang, and Yinglei Teng, *Senior Member, IEEE*

***Abstract*—Immutability, decentralization, and linear promoted scalability make the sharded blockchain a promising solution, which can effectively address the trust issue in the large-scale Internet of Things (IoT). However, currently, the throughput of sharded blockchains is still limited when it comes to high proportion of cross-shard transactions (CSTs). On the other hand, the assemblage characteristic of the collaborative computing in IoT has not been received attention. Therefore, in this article, we present a clustering-based sharded blockchain strategy for collaborative computing in the IoT, where the sharding of the blockchain system is implemented in two steps: *K-means*-clustering-based user grouping and the assignment of consensus nodes. In this framework, how to reasonably group the IoT users while simultaneously guaranteeing the system performance is the key point. Specifically, we describe the data transactions among IoT devices by data transaction flow graph (DTFG) based on a dynamic stochastic block model. Then, formed as a Markov decision process (MDP), the optimization of the cluster number (shard number) and the adjustment of consensus parameters are jointly trained by deep reinforcement learning (DRL). Simulation results show that the proposed scheme improves the scalability of the sharded blockchain in the IoT application.**

***Index Terms*—Collaborative computing, deep reinforcement learning (DRL), dynamic graph analysis, Internet of Things (IoT), *K-means* clustering, sharded blockchain.**

I. Introduction

RECENTLY, the Internet of Things (IoT) has been regarded as the latest information technology, which is expected to support the ubiquitous connectivity of anything at anytime and anywhere [1]. It is predicted that the connected IoT devices will exceed to 25 billion globally by 2025 with the widespread deployment of applications (e.g., smart city, smart wearable devices, autonomous cars, industrial networks, smart homes, etc.) [2]–[6]. Since numerous IoT devices are continually accessed to wireless networks, the constant growing of IoT data puts an extraordinary stress on all types of data management and networking services, raising the concerns about network security, data privacy, and performance degradation [7]. On the other side, there are various computing tasks for different applications delivered to the IoT network and data sharing generated among the IoT devices. In order to support these computation-intensive and latency-sensitive applications, collaborative computing is widely adopted in IoT networks, enabling geographically distributed collaborators cooperatively to finish the off-loaded tasks [8]. Therefore, the setup of trustworthiness among IoT users is extremely crucial in the collaborative environment.

However, the current centralized architecture in the IoT networks is challenged by single point of failure (SPOF), data privacy, reliability, and robustness [9]. Fortunately, as a decentralized technology, blockchain has been proposed to address the security issue in a distributed way [7]–[10]. The original blockchain works as the distributed ledger system for cryptography, like Bitcoin [11], [12]. Along with the smart contract of distributed applications developed in Ethereum [12], blockchain has been regarded as one of the most reasonable candidates to address the trust issue and management in generalized multipeer systems (e.g., IoT systems). Using blockchain, the data sharing among IoT users are recorded in the IoT transactions and packed as the block through the consensus of all the distributed validators so that the data and its logical relations are stored immutably.

Although blockchain has been applied in several distributed system scenarios, e.g., connected vehicles, content delivery networks and smart grids [13]–[16], the drawback of poor

Manuscript received 21 October 2021; revised 14 January 2022; accepted 7 February 2022. Date of publication 17 February 2022; date of current version 24 August 2022. This work was supported in part by the National Natural Science Foundation of China under Grant 62171062 and Grant 61901011; in part by the Foundation of Beijing Municipal Commission of Education under Grant KM202010005017 and Grant KM202110005021; and in part by the Beijing Natural Science Foundation under Grant L211002. *(Corresponding author: Meng Li.)*

Zhaoxin Yang, Ruizhe Yang, Meng Li, and Yanhua Zhang are with the Faculty of Information Technology, Beijing University of Technology, Beijing 100124, China (e-mail: yangzhaoxin@emails.bjut.edu.cn; yangruizhe@bjut.edu.cn; limeng720@bjut.edu.cn; zhangyh@bjut.edu.cn).

F. Richard Yu is with the Department of Systems and Computer Engineering, Carleton University, Ottawa, ON K1S 5B6, Canada (e-mail: richard.yu@carleton.ca).

Yinglei Teng is with the Beijing Key Laboratory of Space-Ground Interconnection and Convergence, School of Electronic Engineering, Beijing University of Posts and Telecommunications, Beijing 100876, China (e-mail: lilytengtt@gmail.com).

Digital Object Identifier 10.1109/JIOT.2022.3152188

scalability for the current blockchain systems prevents it from being implemented in the large-scale IoT. The main reason is that the traditional single-chain system requires full replication of the computation and storage. In other words, each appended block must be verified and stored by all validators in the system [3]. Under this circumstance, a parallelizing paradigm named sharding is proposed, which splits the entire system into multiple shards to realize higher throughput. A representative sharding platform is *Zilliqa* [17] which splits the validator network into a number of local shards and a directory committee, maximizing the blockchain throughput proportionally with the shard number. Similarly, *Ethereum* 2.0 [18] proposed a version of a sharded blockchain that splits the entire network into multiple subnetworks.

With the rise of blockchain sharding, a large number of sharded-blockchain-based IoT systems and applications have emerged. However, some problems are failed to be considered when sharded blockchain applying in the collaborative computing IoT. First, the dynamic and collaborative computing tasks in the IoT networks potentially result in dynamic assemblage characteristics due to the self-organized nature. Then, a dynamic-clustering-based sharding protocol is required instead of the widely used network sharding that randomly divides the IoT devices into different shards. Second, the current sharded blockchains suffer from low throughput and high latency when it comes to high proportion of cross-shard transaction (CST). Although increasing the shard number can achieve higher throughput, it also increases the percentage of CST [19]. Thus, the selection of shard number is the key point of blockchain sharding. Finally, the validator number within a shard linear decreases with the shard number, which increases the odds of malicious nodes' collusion and might result in a spurious block [20]. Thus, the tradeoff of shard number between higher throughput and security is extremely crucial, and alternative sharding model is required to support the IoT by increasing the throughput and ensuring shards' security.

On the other side, developing such a clustering-based sharded blockchain system entails substantial technical challenges. First, the dynamic assemblage characteristic is considered as the rule for the sharding protocol. However, there is a lack of modeling approaches for analyzing the dynamic clustering characteristics driven by the collaborative computing tasks in the IoT. Therefore, the proper modeling method is essential. Moreover, machine learning (ML) is required for the grouping of devices in a massive-IoT environment instead of the existing statistical approach. However, the sensitivity to initialization parameters (e.g., the number of clusters) and the unmeasurable quality of ML results bring new challenges. In combination with the tradeoff between security and throughput, an optimization scheme is required to guide the ML to a reasonable initial group number. Additionally, the existing consensus mechanisms of CSTs suffer from large communication overhead and high latency due to the asynchronous manner. Thus, to improve consensus efficiency, the workflow design of a synchronous-based mechanism should be considered.

In order to address the above issues and challenges, in this article, we propose a clustering-based sharded blockchain for collaborative computing in IoT, where the system throughput, the grouping of IoT users and the security are all taken into consideration. For the collaborative computing IoT, we describe the cooperative relationship of the users based on a dynamic graph. Then, we adopt a parallel instance of consensus systems, which run independently within each shard [20] and the sharding strategy is implemented based on the grouping of users. To guarantee the security of the sharded blockchain while efficiently handling the CST validation, practical Byzantine fault tolerance (PBFT) consensus is adopted in a synchronous manner and the shard number is bounded by preventing the worst case that the all malicious nodes within one shard exceed the maximum malicious nodes numbers.

The main contributions of this article are described as follows.

1) We develop a sharded blockchain-enabled IoT network in the collaborative computing environment, where the system operates in a parallel way and the consensus of CST is processed in an asynchronous manner based on the PBFT.
2) The theoretical analysis of the IoT data flow based on the dynamic graph is presented, where the dynamic community structure of IoT with the collaborative computation task is described as the dynamic stochastic block model. The adoption of *K-means* in graph partition is introduced.
3) By jointly considering the IoT device clustering and sharded blockchain performance, we formulate the dynamic sharding strategy and the adjustment of blockchain parameters as an optimization problem, which is described as a Markov decision process (MDP) by defining state space, action space, and reward function. The optimization goal is to improve the throughput of the entire system while ensuring security.
4) To handle the dynamic selection of cluster number and adjustment of blockchain parameters, we propose an approach combined of dynamic clustering and deep reinforcement learning (DRL). In particular, the initial training parameter of cluster number is trained by the strategy of DRL with double deep Q network (DQN). PyTorch is used to implement the double DQN.

The remainder of this article is organized as follows. In Section II, we first review the existing research on blockchain-enabled IoT systems and sharding systems, then the graph analysis and DRL technology are applied to the blockchains. Section III presents the system architecture, asynchronous consensus process and the IoT transaction graph model. The proposed sharding strategy is described in Section IV. The joint optimization problem is formulated and solved by the double DQN approach in Section V. Then, the simulation setting, the experiment results, and discussions are presented in Section VI. Finally, in Section VII, the conclusion and future works are given.

II. Related Works

In this section, we first review the related works in the blockchain-enabled IoT and sharded blockchain, and then

the current research on blockchain graph analysis and DRL technology used in blockchains are presented.

A. Blockchain-Enabled IoT System

The IoT is entering a new phase of cross-industry integration, which merges data generated from various systems or domains to construct a more powerful industrial sector. Therefore, it requires a dedicated and large-scale cross-industry platform [5]. Due to the advantages of blockchain in decentralized management, transparency and immutable storage, a variety of new methods for applying blockchain to the IoT systems have been developed in both industry and academia [4]–[7], [21]. These attempts to integrate blockchain into various IoT fields will promote the implementation and deployment of trusted, secure, and reliable network environments for the IoT.

The existing blockchain-enabled IoT systems are mainly focused on the design of the framework and the suitable consensus protocols. The objective of the designed framework is to ensure the security and reliability of the data transactions by using the blockchain structure. For example, Li *et al.* [22] proposed an energy blockchain system based on the consortium blockchain, enabling safe energy trading in the Industrial IoT (IIoT) environments. In the medical scenario, a distributed record management system using blockchain technology is proposed [23], the designed consensus protocols usually aim to reduce the memory and energy costs considering the limited computing capacity and storage space for IoT validators [4], [14], [24]. For instance, Guo *et al.* [25] proposed an adaptive resource allocation and block generation scheme, which reaches the consensus between the nodes while simultaneously guaranteeing the system performance. In [26], a novel lightweight Proof of Block and Trade (PoBT) consensus algorithm is proposed, where the memory requirements of the IoT nodes and computation time are decreased by the solution.

It can be seen that the system performance (scalability, latency and security) is the key concern for a blockchain-enabled IoT system. In different scenarios, there are different Quality-of-Service (QoS) requirements, which necessitate the blockchain adjustable to meet different needs during the deployment process.

B. Sharded Blockchain

The sharding technology has been developed rapidly in recent years. Initially, it is a concept in the traditional large database, which divides the data in the database into multiple data slices, aiming to improve the overall performance. Inspired by this dividing of databases, developers have applied sharding technology to blockchain. With the continuous expansion of blockchain application scenarios, its transaction throughput, as an important indicator of system performance, has gradually become the focus of improvement. Therefore, sharding is considered the effective technique to improve the scalability of blockchain [3].

Network sharding is the most basic way of sharding, which divides the entire blockchain validator network into multiple shards [17]. In order to prevent malicious nodes from overpopulating a single shard, the randomness is established among the nodes by operating the verifiable random functions (VRFs) [18]. Moreover, transaction sharding is usually operating along with the network sharding by allocating the transactions to its corresponding shard with the index. A representative sharding protocol is *Zilliqa* [17] that splits the network into a number of shards and a directory committee. It maximizes the throughput in proportion to the number of shards. Similarly, a scheme called *OmniLedger* [27] is devised based on the bias-resistant distributed randomness generation model which samples and updates subsets of peers in shards to improve security. In addition, a scale-out sharded blockchain with asynchronous consensus zones (CZs) in [19] is proposed which reduces the overhead of cross-shard consensus by eventual atomicity. Furthermore, *Ethereum* 2.0 [18] proposed a novel version of local block named collation which can be stored and generated locally by lightweight PoW within the shard. However, there have been comparatively few studies focusing on reducing the CST. Considering the assemblage characteristic of the IoT devices, the main goal of the sharding strategy in this article is to increase the throughput by reducing the CST proportion.

C. Graph Analysis Applied to Blockchains

Graph analysis is a widely used approach for social network analysis, which abstracts the users and their interaction into a network consisting of nodes and connected edges. Much of the previous literature studies the blockchain user networks by graph analysis. For example, Fleder *et al.* [28] presented a graph-analysis framework capable of tracing and clustering user activity in Bitcoin. Chen *et al.* [29] conducted the first systematic study on Ethereum by leveraging graph analysis to characterize the money transfer, smart contract creation, and smart contract invocation on Ethereum. Similarly to [29], Lin *et al.* [30] modeled the transaction records of Ethereum as a static simple graph and predict the links with the records.

For the IoT devices of collaborative computing environment, the dynamic tasks potentially result in dynamic clustering. Hence, the graph analysis for the IoT devices should be taken into account as the theoretical basis for clustering strategy.

D. DRL Technology Applied to Blockchains

Reinforcement learning (RL) is a branch of ML to solve dynamic control problems by interacting with an unknown environment to maximize the expected long-term reward for the learning agent [31]. With the combination of traditional RL and deep neural networks (DNNs), DRL becomes a well-known approach for solving dynamic optimization problems in complex dynamic programming and diverse space states.

In the existing studies of the performance optimization for blockchain systems [9], [13], [32], in order to fulfill the different QoS requirements in various application scenarios, virtualization distributed ledger technology (vDLT) proposed in [32] is adopted, based on which a service-oriented blockchain system is developed by deploying a DRL training agent on the management/control node (MCN) for adaptive resource allocation and node management in a global view

TABLE I
NOTATIONS

Symbol	Definition	Symbol	Definition
N	The total number of validators	M	The total number of IoT users
N_k	The number of validators within the k-th shard	K	The number of shards/UGs/CZs
ξ	Sending operation of a CST	a	The origin user of a CST
φ	Receiving operation of a CST	b	The destination user of a CST
α	Computing cycles for generating/verifying a signature	$\boldsymbol{Y}^{(t)}$	The transaction edge set of a DTFG snapshot
β	Computing cycles for generating/verifying a MAC	$S(i,j)$	The similarity between a pair of IoT users i and j
$F_{k,p}$	The computing capacity of a primary node,	μ_k	The center vertice of the k-th UG
$F_{k,r}$,	The computing capacity of a replica node	G_k	The vertices group of the k-th UG
$R_{n,n'}$	The transmission rates between a pair of nodes n and n'	Θ	The modularity of the IoT user grouping
p	The proportion of the malicious nodes	g_k	The proportion of CST in the k-th shard
T_I	The block interval of sharded blockchain	S_B	The block size of sharded blockchain
λ	The average transaction size	Ξ	The throughput of sharded blockchain

of the system. For instance, Liu *et al.* [33] first presented a performance optimization framework for blockchain-enabled IIoT systems to improve the performance of data security and efficiency with the selection of the block producer and the adjustment of block size under the tradeoff of scalability, decentralization, security, and latency. Similarly, in [9], a consortium blockchain for the IoT is proposed for improving the QoSs by training the selection of the block producer and consensus with dueling DQN. In addition, Yun *et al.* [34] first applied the DRL to the sharded blockchain by optimizing the throughput and meanwhile considering the dynamic malicious attack scenarios.

Different from these studies, we pay more attention to the dynamic and assemblage of collaborative computing in the IoT and consider the sharding protocol. However, although increasing shard number can achieve higher throughput, the percentage of CST grows and malicious attack risks increase. Therefore, the tradeoff of shard number, throughput and security is a key concern, and an alternative optimization scheme based on the DRL is required.

III. System Model

In this section, the network architecture of the proposed scheme is presented, followed by the consensus model and IoT transaction graph model. For the clarity of the following discussion, the key notations are summarized in Table I.

A. Network Architecture

The network architecture of the sharded blockchain-enabled IoT network is depicted in Fig. 1, which consisted of two layers, i.e., the collaborative computing IoT and the sharded blockchain. For better understanding of the framework, we illustrate some concepts for the corresponding architecture.

1) *IoT Users:* IoT device owners in the IoT network, which share the collected data with other IoT users for the different collaborative computing tasks.

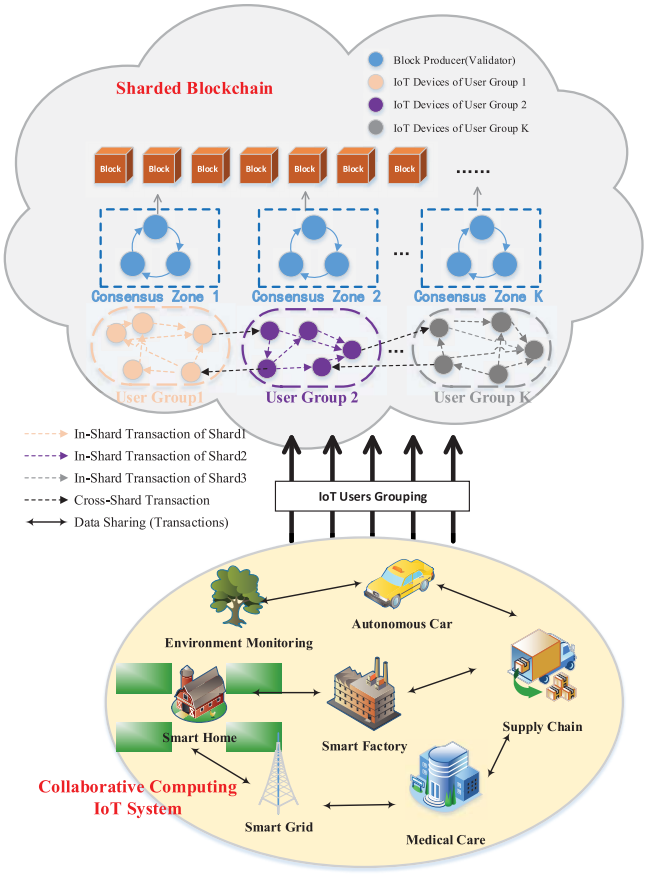

Fig. 1. Sharded-blockchain-based IoT system.

2) *Execution Nodes (Validators):* The nodes that execute consensus to validate the data transactions. Usually, a primary node is selected from the validators to produce transaction blocks and other nodes are the replica nodes to verify the transaction blocks during the consensus process.

3) *Management/Control Node:* A blockchain node is virtualized as the MCN by using vDLT [32], which is responsible for allocating the blockchain resources (e.g., adjusting blockchain parameters) and grouping the IoT users through an optimization policy.

In the IoT network, users are executing different collaborative computing tasks, which requires the users to collect various types of data for different applications with their sensors (e.g., smart city, supply chains, autonomous vehicle, environment monitoring, etc.) [14]–[16]. Meanwhile, data sharing is generated between the IoT users for the same collaborative computing task or the different tasks among various application domains. For instance, the images captured by the embedded cameras of smart machines on the product line might be required by the hand-held terminal in logistics for the supply chain traceability.

The blockchain system serves as a data storage and management platform. Using blockchain, the data sharing among IoT users are recorded as the transactions and packed as the transaction block through the consensus of all the distributed validators so that the shared data and its logical relations are stored immutably. In order to solve the performance constraints on scalability and latency caused by the propagation of a large number of transactions and messages between massive IoT devices and consensus nodes, we adopt a sharded-blockchain architecture.

Here, we assume that there are M IoT devices and N validators divided into K user groups (UGs) and CZs. Each shard is constituted with a UG and a CZ, and the set of shards is denoted as $\mathcal{K} = \{1, \ldots, k, \ldots, K\}$ with the kth UG and CZ. Most intra-shard transactions generated in a UG quickly reach consensus by its CZ, while a fragment of CST is validated to the blockchain through cross-shard consensus. Note that the workload of the entire system is partitioned by sharding, and this parallel processing linearly increases the throughput of blockchain and is more satisfied with the large-scale IoT systems in the smart city.

B. PBFT-Based Asynchronous Consensus Model

For intra-shard transactions, we employ the traditional PBFT consensus, while for inter-shard transactions, we define an asynchronous PBFT consensus based on the sharding protocol in *Monoxide* [19] and *Zilliqa* [17].

In the proposed asynchronous PBFT, to minimize the overhead of cross-shard validation, we use Eventual Atomicity to splits one transaction into multiple atomic operations, each of which relates to different shards so that they can be processed in parallel and overlapping [19]. Moreover, the mechanism for unconfirmed transactions in each shard is able to handle the asynchronous messaging between CZs by the relaying execution. Specifically, for a transaction having a sending operation ξ from device a and a receiving operation φ to device b, it can be immediately handled within the shard if a and b belong to the same UG. When a and b are in different UGs, we allow the sending operation to be executed first, submitted to the corresponding CZ with other intra-shard transactions. Once the sending operation is confirmed, the receiving operation will be started.

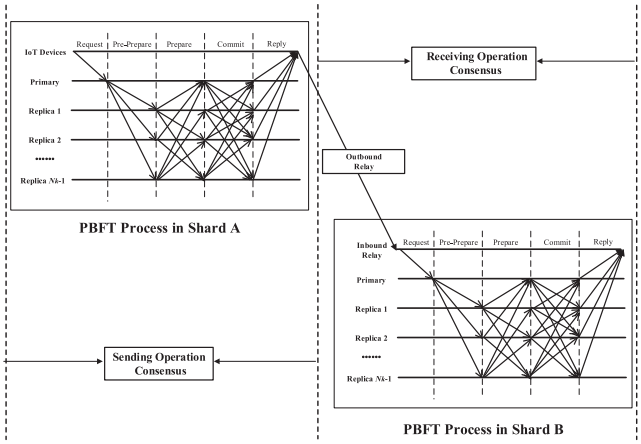

Fig. 2. Eventual atomicity implemented with PBFT-based asynchronous consensus.

1) Sending Operation Consensus in Source Shard: As shown in Fig. 2, for a CST requested from UG A to UG B, the consensus for sending operation in CZ A is described as follows.

1) An IoT device in UG A requests CST, an unconfirmed $\gamma = <\xi, a, \varphi, b>$ is added to the transaction pool of CZ A.
2) The transaction is picked up by the primary node of CZ A to run a PBFT, which is consists of four steps: a) preprepare; b) prepare; c) commit; and d) rely.
3) The sending operations ξ are executed after reaching the PBFT consensus.
4) An outbound relay transaction $\gamma' = <\varphi, b>$ is derived, which will be transmitted to the destination shard, i.e., the device b's shard B.

2) Relay Transaction Consensus in Destination Shard:

1) The inbound relay transaction $\gamma' = <\varphi, b>$ is picked up by the primary node of CZ B.
2) Validators in CZ B run a PBFT to validate the receiving operation, including preprepare, prepare, commit, and rely.
3) The receiving operations φ is executed after reaching the PBFT consensus, and the full transaction $\gamma = <\xi, a, \varphi, b>$ is complete.

Here, for the resource consumption, we only consider the computational cost for the procedures of encryption and decryption. Specifically, there are signature verifying, message authentication (MAC) generating and validating, which require α, β and β CPU cycles, respectively.

The consensus protocol is separated into message transfer and MAC. For the intra-shard message transfer between validators, the data transmission rate μ is obtained based on a finite-state Markov process, whose value is divided into L discrete levels donated as $\boldsymbol{\mathcal{R}} = \{R_0, R_1, \ldots, R_{L-1}\}$. Thus, the $L \times L$ transition probability matrix is defined as $[p_\mu]_{L\times L}$, with $[p_\mu]_{l,l'} = \Pr[\mu(t+1) = R_l | \mu(t) = R_{l'}]$ and $R_l, R_{l'} \in \boldsymbol{\mathcal{R}}$.

Similarly, for the MAC, the computational capacity of validators σ is also modeled as a finite-state Markov process, whose value is divided into J discrete levels donated as $\boldsymbol{\mathcal{F}} = \{F_0, F, \ldots, F_{J-1}\}$. The $J \times J$ transition probability

matrix is defined as $[p_\sigma]_{J\times J}$, where $[p_\sigma]_{j,j'} = \Pr[\sigma(t+1) = F_j | \sigma(t) = F_{j'}]$ and $F_l, F_{l'} \in \mathcal{F}$.

C. IoT Data Transaction Graph Model

Here, the transaction behaviors between devices are represented by monetary flow graph (MFG) [29]. For data sharing, we assume that each IoT device sends only one transaction to another IoT device at a time slot. Then, the transactions between devices in time slot t can be expressed as a snapshot of the dynamic social graph, with the vertices representing IoT devices and the edges representing the flow of transactions between the devices. This dynamic social graph is named the data transaction flow graph (DTFG).

In order to describe the interaction relationship between devices, we encode a snapshot graph of a dynamic DTFG among M IoT devices in time slot t as an $M \times M$ binary sociomatrix $\boldsymbol{Y^{(t)}} = [{y_{i,j}}^{(t)}]$, where we have ${y_{i,j}}^{(t)} = 1$ if the device i send at least one data sharing transaction to device j, and ${y_{i,j}}^{(t)} = 0$ otherwise. In addition, ${y_{i,j}}^{(t)} = 0$ is required, as devices do not send transaction to themselves [35]. Subsequently, we use $\boldsymbol{Y_T} = \{\boldsymbol{Y}^{(1)}, \boldsymbol{Y}^{(2)}, \ldots, \boldsymbol{Y}^{(T)}\}$ to denote the time-series changes of snapshot graphs for a given social network over the time duration T.

Note that the collaborative computing tasks in IoT networks potentially result in clustering, and there will be community structures in the DTFG accordingly. Aimed at this problem, we use a stochastic block model for this community-structure-based snapshot graph [35]. However, the traditional stochastic block model can only handle static graphs [36]. Therefore, we extend it to dynamic model by considering the transition of collaborative computing tasks as a Markov chain. In a snapshot graph $\boldsymbol{Y^{(t)}}$, the edges among M devices are generated with $C(t)$, collaborative computing tasks in IoT network, which means $C(t)$ is the potential community numbers in the graph. We consider that the value of collaborative computing task is partitioned into H discrete intervals, denoted as $\mathcal{C} = \{C_0, C_1, \ldots, C_{H-1}\}$. The transition probability matrix is defined as $[p_C]_{H\times H}$ of size $H \times H$, where we have $[p_C]_{h,h'} = \Pr[C(t+1) = C_h | C(t) = C_{h'}]$ and $C_h, C_{h'} \in \mathcal{C}$. Subsequently, related to [37], the proposed dynamic stochastic block model $\{\boldsymbol{\zeta}(t),\ \boldsymbol{\theta}(t)\}$ for $\boldsymbol{Y^{(t)}}$ are given as: a vector of discrete indicators $\boldsymbol{\zeta}(t) = (\zeta_1(t), \ldots, \zeta_M(t))$, where $\zeta_i(t) = c(t), c(t) = 1, \ldots, C(t)$ if and only if device i is working on task $c(t)$, and a matrix $\boldsymbol{\theta}(t) = [\theta_{c(t),c'(t)}]$ of size $C(t) \times C(t)$, where $\theta_{c(t),c'(t)}$ represents the probability that a device of task c sends transaction to a device of task c'.

IV. Blockchain Sharding Strategy

In this section, the blockchain sharding strategy is presented, consisting of the grouping of UGs based on *K-means* and the assignment of consensus nodes to the IoT group. Then, the system performance of the sharded-blockchain is analyzed in the aspect of scalability, latency and security.

A. Grouping of IoT Devices Based on K-Means

In terms of the atomic operation of sending and receiving that we described previously, the validation of a CST will be doubled. That is, the throughput will be affected if the proportion of CST is high. Therefore, in order to decrease the proportion of CST and maintain the underlying community structure, we propose a K-means-based grouping scheme for IoT devices.

Given a DTFG snapshot $\boldsymbol{\mathcal{G}} = (\boldsymbol{\mathcal{V}}, \boldsymbol{\mathcal{E}})$ at time slot t, $\boldsymbol{\mathcal{V}}$ represents the vertices set that consists of M IoT devices, and $\boldsymbol{\mathcal{E}}$ represents the edge set described by the DTFG binary sociomatrix $\boldsymbol{Y^{(t)}} = [{y_{i,j}}^{(t)}]$. Assume that K is the number of target groups. In the classical K-means algorithm, with randomly selected initial K central vertices, the vertices are clustered according to their minimum distance to the central vertices. Then, the center vertices will be updated when the groupings are obtained. These groupings and center updates iterate until converge. However, the traditional K-means algorithm is widely applied in numerical-based clustering scenarios, which is not compatible with our graph application. Thus, we redefine a sample vertices distance and center vertices updates in the DTFG scenario.

Instead of using the *Euclidean distances*, in order to estimate the similarity between different samples for a reasonable classification, we use the *Jaccard correlation coefficient* [38] to measure the relation degree of each pair of vertices

$$S(i,j) = \frac{N(i) \cap N(j)}{N(i) \cup N(j)} \tag{1}$$

where $N(i)$ and $N(j)$ are the neighbor vertices of vertices i and j, $N(i) \cap N(j)$ is the intersection of neighbor vertices between vertices i and j, and $N(i) \cup N(j)$ is the union of neighbor vertices between vertices i and j. The ratio of intersection to union gives the similarity between the two vertices. If the intersection is 0, the similarity between the two vertices is 0 [38].

For the center vertex, the classic K-means algorithm updates it by averaging the coordinates of all the sample points within the cluster, however, the coordinates do not exist in a graph. Alternatively, here the vertices with the largest sum of similarity to the other vertices within the group are selected as the new center vertices, which can be described as follows:

$$\mu_k^{\text{new}} = \arg\max_p \left(\sum_{p,g \in \boldsymbol{\mathcal{V}}_k} S(p,g) \right) \tag{2}$$

where μ_k^{new} represents the new center vertex of the kth UGs, and $\boldsymbol{\mathcal{V}}_k$ is the vertices set of the kth UGs.

Using the redefined similarity and center vertices updates, the detail of our K-means-based IoT device grouping is shown in Algorithm 1.

In each time slot t, the snapshots of a DTFG is formulated as a binary sociomatrix $[{y_{i,j}}^{(t)}]$ according to the transaction flows among the IoT users. Based on the formulated edge set and the input group number K, the MCN executes the grouping process.

First, K vertices are randomly selected as the initial center vertices, donated by $\{\mu_1^{\text{Init}}, \ldots, \mu_K^{\text{Init}}\}$. For each iteration, each IoT user should calculate the similarity between itself and the K central vertices $S(v, \mu_k^{\text{Init}})$. Then, M users are clustered into the specific K UGs with the maximum similarity to the corresponding center vertices.

Algorithm 1 K-Means-Based Grouping Scheme for UG

Initialization:
Input the DTFG snapshot $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with the binary sociomatrix $\boldsymbol{Y}^{(t)} = [y_{i,j}{}^{(t)}]$;
Input the grouping number of UGs K;
Random select K vertices as center vertices: $\{\mu_1^{Init}, \mu_2^{Init}, \ldots, \mu_K^{Init}\}, \forall \mu_k^{Init} \in \mathcal{V}$ for K UGs $\{G_1, G_2, \ldots, G_K\}$;
Initializing the center vertices by, $\{\mu_1^{train}, \mu_2^{train}, \ldots, \mu_K^{train}\} = \{\mu_1^{Init}, \mu_2^{Init}, \ldots, \mu_K^{Init}\}$;
***K-means*-Based Grouping Process**:
for each vertice in $\mathcal{V}$ **do**
  Calculating the similarity between each vertice and K center vertices $S(v, \mu_k^{train})$;
  Maping the vertice to the UG of the center vertice with maximum similarity;
  Assigning the vertice to the corresponding UG: $G_1 = G_1 \cup \{v\}$;
  for each UG G_k **do**
    Calculating the new center vertice in (2);
    if $\{\mu_1^{new}, \ldots, \mu_K^{new}\} = \{\mu_1^{train}, \ldots, \mu_K^{train}\}$ **then**
      Output the UG Grouping Result: $\{G_1{}^{new}, G_2{}^{new}, \ldots, G_K{}^{new}\}$
    else
      Update the center vertices by, $\{\mu_1^{train}, \ldots, \mu_K^{train}\} = \{\mu_1^{new}, \ldots, \mu_K^{new}\}$
    end if
  end for
end for

Afterward, K center vertices should be updated when all the groupings are obtained. Specifically, each UG should update the central vertex by selecting the vertex with the largest sum of similarity to the all the other vertices within the group, that is $\mu_k^{\text{new}} = \arg\max_p(\sum_{p,g \in \boldsymbol{G}_k} S(p, g))$.

The above grouping and updating process will iterate until convergence that the central vertices no longer change. Then, the grouping process is completed and K UGs are created according to the clustering result.

However, as an unsupervised learning algorithm, the quality of the grouping results is difficult to evaluate. Besides, it is also strongly sensitive to the cluster number K. Here, we introduce the community modularity Θ, a parameter for evaluating the strength of the grouping characteristics [39]. The definition is expressed as follows:

$$\Theta = \frac{1}{2e} \sum_{i,j} \left[\left(y_{i,j}^{(t)} - \frac{h_i h_j}{2e} \right) \delta\left(G_i, G_j\right) \right] \tag{3}$$

where h_i and h_j are the degrees (all the edges connect to a vertex) of vertices i and j, respectively, G_i and G_j are the affiliated group of vertices i and j, and e refers to the total number of edges within the entire graph. Here, when vertices i and j belong to the same group, we have $\delta(G_i, G_j) = 1$; otherwise, equals 0. The calculative value of Θ is between 0 and 1, while in real networks, its actual value is usually between 0.3 and 0.7. Here, the greater value of Θ is, the more evident assemble characteristics there are. In order to extend the grouping method into dynamic DTFG and optimize the grouping result, we use modularity as the reward to optimize the selection of group number K.

B. Sharding of Validators

Corresponding to the IoT device groups, N blockchain validator nodes should be partitioned into K CZs, and then the CZs should be attributed to the UGs in one-to-one correspondence. For this issue, first, a shard number (IDs) is assigned to each node in a decentralized method [34]. Practically, the IDs are set by VRF and the distribution of the validators for K shards is based on the result of VRF. Similar to the sharded *Zilliqa* [17], all the nodes in the blockchain system acquire their own IDs by operating a simple PoW puzzle in the follows:

$$\text{ID} = \mathcal{H}(\text{EpochRand}||\text{IP}||\text{PK}||\text{Nonce}) < D \tag{4}$$

where EpochRand is a random seed for the PoW, and it is reset whenever a shard is reconstructed. Public key (PK), IP, and Nonce are additionally used to compute the node ID. $\mathcal{H}$ is the hash operator and D is the difficulty level of the PoW algorithm [29]. Each node knows its own shard number after the ID computation and should report its ID to the MCN. The validators of the same CZ should set up a point-to-point connection, then the sharding procedure is completed [34].

C. System Performance of Sharded-Blockchain

1) Scalability: The scalability of a sharded blockchain can be evaluated by throughput, defined as the maximum number of validated transactions blocked onto the chain in unit time [34]. It usually has two performance parameters: 1) block size and 2) block interval. Since the sharded blockchain produces blocks in a parallel manner, the throughput is K times increased with sharding

$$\Xi\left(S_B, K, T^I\right) = \frac{K(S_B/\lambda)}{T^I} \tag{5}$$

where S_B is the maximum size (bytes) for a block produced during interval period T^I, and K and λ are the shard number and the average transaction size, respectively. It is obvious that increasing the block size and shard number or decreasing the block interval can improve the throughput in (5).

When the proportion of CST in the kth shard (including the inbound transactions and outbound transactions) is g_k, the maximum throughput of the entire system can be given by

$$\Xi\left(S_B, T^I, K, g_k\right) = \frac{K(S_B/\lambda)}{T^I} \left[1 - \frac{1}{2K} \sum_{k=1}^{K} g_k \right]. \tag{6}$$

Note that the CST take up the double space as that atomicity operation is split into two parts. The first part of (6) is the total throughput ignoring CST, and the second part considers the effect of the excess portion occupied by CST. It can be seen that the reduced g_k brings the increased system throughput; thus, it is important to select a suitable shard number.

2) Latency: Latency is the time of a transaction from being sent by the IoT device to be irreversible on chain. In terms of our asynchronous consensus model, each CZ not only validates the intra-shard transactions but also the sending and receiving operation of CST, which we renamed going-out transaction and coming-in transaction. Note that the consensus latency of the coming-in transaction is equivalent to the intra-shards', thus the total latency T_{Latency} can be considered as the sum of intra-shard consensus delay T_k^{con} and transmission delay of the going-out transaction to other shards T_k^{out}

$$T_{\text{Latency}} = T_k{}^{\text{out}} + T_k^{\text{con}}. \tag{7}$$

Considering the intra-consensus delay of the kth shard T_k^{con} includes message propagation time and message verification delay. In the kth CZ having $N_k (N_k = N/K)$ validators, the primary node and replica nodes based on PBFT perform the following protocols on the transactions.

a) Request: The IoT devices submit the transactions to its CZ for block validation and the primary node collects a batch of transactions for validation. For each transaction request, the primary node should verify one MAC and one signature. Then, the verified transactions, including the index of the shared data and other collaborative tasks information, will be packaged into a new block. This phase proceeded within the block interval period T^I.

Here, the maximum number of transactions in a block is given by S_B/λ, and the computational cost for a primary node can be expressed by $\Delta_{\text{req},n_{k,p}} = (S_B/\lambda)(\alpha + \beta)$.

For this phase, the transmission latency of the transactions request T_k^{req} can be calculated by

$$T_k^{\text{req\_tr}} = \max_{m_{k,i} \in M_k, n_{k,p} \in N_k} \left\{ \frac{\lambda}{R_{m_{k,i},n_{k,p}}} \right\} \tag{8}$$

where λ is the average transaction size, and $R_{m_{k,i},n_{k,p}}$ is the transmission rate between the IoT device i of the kth UG and the primary node of the kth CZ.

b) Preprepare: The primary node broadcasts the packaged block and the preprepare message into its CZ. The replica nodes should verify whether the preprepare message is valid. In this phase, the primary node generates one signature for the newly signed block and $N_k - 1$ MAC in the preprepared message, and each replica node should verify one signature and one MAC for the new block and each transaction within that block.

The computational cost for the primary node is $\Delta_{\text{prep},n_{k,p}} = \alpha + (N_k - 1)\beta$, and the computational cost for the replica node is $\Delta_{\text{prep},n_{k,r}} = ([S_B/\lambda] + 1)(\alpha + \beta)$. In addition, the transmission latency $T_k^{\text{prep\_tr}}$ can be expressed by

$$T_k^{\text{prep\_tr}} = \max_{n_{k,r}, n_{k,p} \in N_k} \left\{ \frac{S_B}{R_{n_{k,p},n_{k,r}}} \right\} \tag{9}$$

where S_B is the maximum size of a newly signed block and $R_{n_{k,p},n_{k,r}}$ is the transmission rate between the primary node and the replica node of the kth CZ.

c) Prepare: All the replica nodes will send a prepare message to its CZ network, and all the validators need to verify whether the preprepare message is contained in the prepare message. After receiving $2f$ of the verified prepare messages, the consensus would enter the next phase.

Here, all the replica nodes should generate a signature for the messages and $N_k - 1$ MACs to all the other validators, and both the primary node and each replica node need to verify $2f$ signatures and MACs for validation received from all the other replica nodes. Therefore, the computational cost for the primary node is $\Delta_{\text{pre},n_{k,p}} = 2f(\alpha + \beta)$, and the computational cost for the replica node is $\Delta_{\text{pre},n_{k,r}} = \alpha + (N_k - 1)\beta + 2f(\beta + \alpha)$.

In addition, the transmission latency $T_k^{\text{pre\_tr}}$ can be expressed by

$$T_k^{\text{pre\_tr}} = \max_{n_{k,r}, n_{k,r'} \in N_k, r \neq r'} \left\{ \frac{S_B}{R_{n_{k,p},n_{k,r'}}} \right\} \tag{10}$$

where $R_{n_{k,p},n_{k,r'}}$ is the transmission rate between the replica nodes of the kth CZ.

d) Commit: All the validators within the CZ (including the primary node and all the replica nodes) will exchange the messages from each other. For the primary node and all the replica nodes, they all need to generate one signature along with $N_k - 1$ MACs, and verify $2f$ signatures and MACs, respectively.

For the commit phase, the computational cost for the primary node can be calculated by $\Delta_{\text{com},n_{k,p}} = \alpha + (N_k - 1)\beta + 2f(\beta + \alpha)$, and the computational cost for the replica node is $\Delta_{\text{com},n_{k,r}} = \alpha + (N_k - 1)\beta + 2f(\beta + \alpha)$. In addition, the transmission latency $T_k^{\text{com\_tr}}$ can be expressed by

$$T_k^{\text{com\_tr}} = \max_{n_{k,r'}, n_{k,r''} \in N_k, r' \neq r''} \left\{ \frac{S_B}{R_{n_{k,r'},n_{k,r''}}} \right\} \tag{11}$$

where $R_{n_{k,r'},n_{k,r''}}$ is the transmission rate between the validators of the kth CZ.

e) Reply: Once upon collecting $2f$ of the verified commit messages, the new package block will be added to local chain by the local shard as a valid block. Here, considering the limited memory size of IoT devices, the reply messages are requested to be submitted to the primary node rather than the IoT devices. Therefore, all the replica nodes generate a final signature and a MAC for each transaction and the primary node verifies $2f$ signatures and MACs. The computational cost for the primary node is $\Delta_{\text{re},n_{k,p}} = 2f(\alpha + \beta)$, and the computational cost for the replica node is $\Delta_{\text{re},n_{k,r}} = 2f(\alpha + \beta)$. In addition, the propagation delay for the reply phase $T_k^{\text{com\_tr}}$ is given as

$$T_k^{\text{re\_tr}} = \max_{n_{k,r''}, n_{k,p} \in N_k, r'' \neq p} \left\{ \frac{S_B}{R_{n_{k,r''},n_{k,p}}} \right\} \tag{12}$$

where $R_{n_{k,r''},n_{k,p}}$ is the transmission rate between the replica node and the primary node of the kth CZ.

By now, the intra-shard consensus is complete. Note that the computational cost for the primary node is $\Delta_{n_{k,p}} = \Delta_{\text{req},n_{k,p}} + \Delta_{\text{prep},n_{k,p}} + \Delta_{\text{pre},n_{k,p}} + \Delta_{\text{com},n_{k,p}} + \Delta_{\text{re},n_{k,p}}$. Then,

the validation time for the primary node of the kth CZ can be expressed by

$$T_{k,p}^{\text{val}} = \frac{\Delta_{n_{k,p}}}{F_{k,p}}. \tag{13}$$

Similarly, the computational cost for the replica node can be given by $\Delta_{n_{k,r}} = \Delta_{\text{req},n_{k,r}} + \Delta_{\text{prep},n_{k,r}} + \Delta_{\text{pre},n_{k,r}} + \Delta_{\text{com},n_{k,r}} + \Delta_{\text{re},n_{k,r}}$. Thus, the validation time for the replica node in the kth CZ can be calculated by

$$T_{k,r}^{\text{val}} = \frac{\Delta_{n_{k,r}}}{F_{k,r}} \tag{14}$$

where $F_{k,p}$ and $F_{k,r}$ are the computing capacity of the primary node of the kth CZ and the replica node in it, respectively. Subsequently, we have the validation time of intra-shard consensus as

$$T_k^{\text{val}} = \max\left\{T_{k,r}^{\text{val}}, T_{k,p}^{\text{val}}\right\}. \tag{15}$$

The propagation time for each request of intra-shard consensus can be written as

$$\begin{aligned} T_k^{\text{tr}} = &\min\left\{T_k^{\text{req\_tr}}, T_{\text{lim}}\right\} + \min\left\{T_k^{\text{prep\_tr}}, T_{\text{lim}}\right\} \\ &+ \min\left\{T_k^{\text{pre\_tr}}, T_{\text{lim}}\right\} + \min\left\{T_k^{\text{com\_tr}}, T_{\text{lim}}\right\} \\ &+ \min\left\{T_k^{\text{re\_tr}}, T_{\text{lim}}\right\} \end{aligned} \tag{16}$$

where T_{lim} is set to be the longest waiting time for the response during the message propagation.

Finally, the intra-shard consensus latency can be derived as

$$T_k^{\text{con}} = T_k^{\text{tr}} + T_k^{\text{val}}. \tag{17}$$

f) Going-out transaction: After the intra-shard consensus, the going-out transaction will be generated by the primary node and delivered to the primary node of the destination CZ. Considering the relay transaction occupies a normal transaction size. Here, its transmission delay T_k^{out} can be expressed by

$$T_k^{\text{out}} = \max_{n_{k,p}\in N_k, n_{k',p}\in N_{k'}} \left\{\frac{\lambda}{R_{n_{k,p},n_{k',p}}}\right\} \tag{18}$$

where $R_{n_{k,p},n_{k',p}}$ represents the transmission rate between the primary node of the kth UG and the primary node of the k'th CZ.

The latency should be less than the generation period of the blocks to satisfy the finality property of the blockchain. That is

$$T_{\text{latency}} \le T^I. \tag{19}$$

3) Security: In the blockchain system, the data transactions are packed as the data block through the consensus of all the distributed validators. Thus, the shared data and its logical relations are jointly approved by the validators and immutably stored in all nodes' local block copies. In other words, all the nodes within the blockchain system store the full backup of the data transactions set. Therefore, the integrity of the shared data is not affected when destroying a certain number of nodes, which could guarantee the robustness of the system.

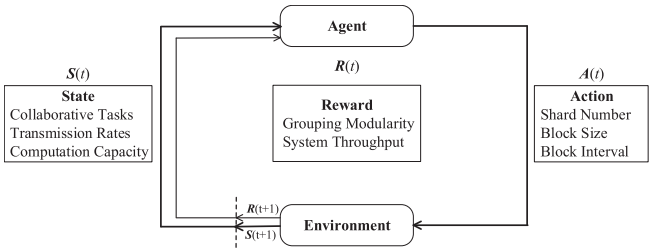

Fig. 3. MDP formulation of joint optimization problem.

However, it is worth noting that the security mostly relies on the specific blockchain consensus. For instance, in Bitcoin, PoW can prevent the blockchain from being controlled by a group of malicious nodes who have no more than 51% of the hash power. In the PBFT protocol, absolute security of the consensus could be guaranteed when at least two-thirds of validators are trustworthy. In other words, for the PBFT of a single-chain system, in order to ensure security, the number of the total validators N_S and the number of malicious nodes f must satisfy

$$3f + 1 \le N_S. \tag{20}$$

In the proposed sharded-blockchain system, each shard runs a PBFT protocol with N_k validators. If the probability of a validator node to be dishonest is p, the total number of malicious nodes within the entire blockchain system is Np. Considering the worst circumstance of the validator allocation that all the malicious nodes of the entire blockchain system gather into one CZ [34], then the following constraint should be met:

$$3Np + 1 \le N_k\left(N_k = \frac{N}{K}\right). \tag{21}$$

Here, we can easily find that the number of shards (the number of UGs and CZs) K is restricted by

$$K \le \frac{N}{3Np + 1}. \tag{22}$$

V. Problem Formulation

In order to achieve a rational grouping of UGs and optimize the performance of the sharded blockchain, it is essential to make the joint decision regarding the shard number selection, and the adjustment of other blockchain parameters. We solve this joint optimization problem by DRL. To implement it, the joint optimization problem is formulated as an MDP composed of system state, system action, and reward function, the detail of the formulation is shown in Fig. 3.

A. State Space

Here, we use $S(t)$ to describe the system state at a discrete time epoch $t(t = 1, 2, 3, \ldots, T)$

$$S(t) = \left\{\mu(t), \sigma(t), C(t), \boldsymbol{Y}^{(t)}\right\} \tag{23}$$

where the transmission rate between validators $\mu(t)$, the computational capacity of a validator $\sigma(t)$, the number of collaborate computing tasks $C(t)$, and the snapshot graphs $\boldsymbol{Y}^{(t)}$ are considered as the system states, whose transition probability have been given in previous sections.

B. Action Space

In terms of the performance analysis in Section IV-C, we know that shard number K, block size S_B, and block interval T^I are important factors. Thus, the action space at decision epoch t is

$$A(t) = \left\{K(t), S_B(t), T^I(t)\right\} \tag{24}$$

and we give the constraints $K(t) \in \{2, \ldots, \dot{K}\}$, $S_B(t) \in \{2, 4, \ldots, \dot{S}\}$, and $T^I(t) \in \{2, 4, \ldots, \dot{T}_I\}$, with the maximum shard number $\dot{K}$, the maximum block size $\dot{S}$, and the maximum block interval $\dot{T}_I$.

C. Reward Function

The optimization goal is achieved by maximizing the long-term reward. Since the system performance depends on both the grouping of IoT devices and the consensus of the sharding, the reward function can be given as the combination of grouping modularity and the throughput while satisfying the security and the latency constraints

$$\begin{aligned} \max_{A} \quad & Q(S, A) \\ & \text{C1} : T_{\text{Latency}} \leq T^I \\ & \text{C2} : K \leq \frac{N}{3Np + 1} \end{aligned} \tag{25}$$

where $Q(S, A)$ denotes the system long-term reward. In order to satisfy the security and latency constraints, a penalty function is adopted within the immediate reward $R(t)$, which donated as

$$R(t) = \begin{cases} \omega_G \Theta(t) + \omega_S \Xi(t), & \text{when C1, C2 are satisfied} \\ 0, & \text{otherwise.} \end{cases} \tag{26}$$

where $\Theta(t)$ donates the modularity of the IoT grouping and $\Xi(t)$ represents the system throughput, and the two constraints satisfy the security and the latency, respectively, and $\omega_G, \omega_S \in (0, 1]$ are the weights coefficient, and there is $\omega_G + \omega_S = 1$. Specifically, the value of the coefficient weight donates the dynamic preference of the optimization goals in different scenarios, where the weight of the individual reward can be adjusted according to its importance, so as to fit the actual optimization scenario.

Thus, the long-term reward can be given by

$$Q(S, A) = \left[\sum_{t=0}^{\infty} \rho^t R(t)(S(t), A(t))\right] \tag{27}$$

with the discount factor $\rho \in (0, 1]$ that reflects the tradeoff between the immediate and future rewards.

D. Workflow of Double DQN

In order to handle the high-dynamic and the large-dimensional character of the IoT devices and the data transaction flows, as well as the joint optimization goals in terms of the user grouping and system scalability, we resort to double deep Q-learning (double DQN) approach to optimize the long-term reward [41]. The workflow is shown in Algorithm 2.

Algorithm 2 Double DQN-Based Optimization Framework for Sharded Blockchain-Enabled Collaborative Computing IoT

Initialization:
Initialize the system state $S(t)$;
Initialize the DTFG by the task number in system state;
Initialize replay memory D with the capacity Z;
Initialize the main Q network of $Q(S, A; \theta)$ with random weights θ);
Initialize the target Q network of $Q^-(S, A; \theta^-)$ with random weights θ^-);
Input maximum training episode $\dot{E}$, maximum training step $\dot{O}$
Double DQN Learning Process:
for each training episode $= 1, \ldots, \dot{E}$ **do**
  for each training step $= 1, \ldots, \dot{O}$ **do**
    Select a random probability δ;
    if $\delta < \varepsilon$ **then**
      Select a random action
    else
      $A(t) = arg \max_A Q(S, A; \theta)$
    end if
    Decrease exploration probability ε;
    Execute action $A(t)$ to select shard number to implement the blockchain sharding and adjust parameters, and observe reward $R(t)$ and proceed to next state $S(t + 1)$;
    Store the experience $(S(t), A(t), R(t), S(t + 1)$ into the replay memory D;
    Random Sample a mini-batch D^- of state transition $(S(i), A(i), R(i), S(i+1))$ from the replay memory D;
    Set $y_t^{DoubleDQN} = r_t + \rho Q\left(S_{t+1}, arg \max_A Q(S_{t+1}, A; \theta); \theta^-\right)$ to calculate the target Q^- value from the target Q network;
    Update target Q network by performing the gradient descent of loss function by $L(\theta) = \left(y_t^{DoubleDQN} - Q(S_t, A_t; \theta)\right)^2$ for every G step;
  end for
end for

Generally, the DRL framework [33] contains an offline DNN construction phase that can approximate the action–state value function $Q(S, A)$ with the corresponding states and actions, and an online deep Q-network phase for dynamic network updating and policy making. Particularly, regarding the challenging factor of long training time, in the designed framework, through the interaction between the IoT blockchain environment (*IBCenv*) and the double DQN agent, the double DQN model is first trained to learn the optimal policies in an offline way regarding the shard number and other blockchain parameters selection. Then, the intelligent strategy can be performed by the MCN of the blockchain system to reasonably execute the IoT grouping with the optimized group number and adaptively adjust the size of the blocks and block interval in an online way. In that case, the challenging issue of the long execution time can be avoided.

In each decision episode, the agent observes the system states $S(t)$ of the *IBCenv*, including the collaborative computing tasks, as well as the computation resources and transmission rates of the validators. Initially, the DTFG is generated according to the task state based on the stochastic block model. Then, the agent executes an action $A(t)$ by exploration or exploitation with probability ε, which consisted of the shard number decision and other blockchain parameters selections. Meanwhile, the agent gets an immediate reward $R(t)$ and changes the *IBCenv* to the next state $S(t+1)$ based on the formulated state transitions in Section V-A.

In order to handle the large dimension system state, using DNN, the action–state value $Q(S, A)$ can be approximated with an approximate value $Q(S, A, \theta)$ [42]. Specifically, the agent inputs the observed *IBCenv* states into the DNN, and then obtains the approximate Q-values of all possible actions as outputs. In order to optimize the system performance, the agent should learn an optimal policy π to maximize the action–state value function $Q_\pi(S, A)$ by iterating the action–state values function based on the Bellman equation.

Similar to DQN, the double DQN also adopts the fixed target network [40] and experience replay to improve the stability of RL. Specifically, in each time step, the agent stores the transition set $(S(t), A(t), R(t), S(t+1))$ into the replay memory D and randomly extracts a minibatch to train the DNN, so as to alleviate the strong correlation between the sample [33]. Additionally, a main network and a target network are initialized at the beginning and the target networks are cloned by the main network in each G step.

On the other side, unlike the DQN that the Q value of target network is updated based on a greedy policy to maximize the Q value of the next state, which is donated by $y_t = r_t + \rho \max_{A_{t+1}} Q(S_{t+1}, A_{t+1}; \theta^-)$. The double DQN approach decomposes the action selection and evaluation from the estimation. Particularly, the evaluation of the Q value is based on the selection of the actions that maximize the action–state value function of the current main network, which is donated by $y_t^{\text{doubleDQN}} = r_t + \rho Q(S_{t+1}, \arg\max_A Q(S_{t+1}, A; \theta); \theta^-)$. Through this double evaluation manner, the overestimation of the Q value can be avoided, which can help yield more accurate estimations and improve the performance of DRL.

Finally, the main network updates the approximate Q values $Q(S, A, \theta)$ by performing the gradient descent of loss function $L(\theta) = (y_t^{\text{doubleDQN}} - Q(S_t, A_t; \theta))^2$. Here, the weights and biases θ in the main network are constantly adjusted to minimize the loss function $L(\theta)$. Then, the optimization results of the system can be adopted by the MCN to execute the policy in an online way, including creating the IoT group and selecting the blockchain parameters.

VI. Simulation Result

We use computer simulation to illustrate the effectiveness of the proposed scheme. Specifically, in order to demonstrate the effectiveness of the proposed optimization scheme and for the ease of the network modeling, we considered a sharded-blockchain for collaborative computing with 180 IoT devices and 120 validators. For double DQN-based optimization, the scheme was deployed on PyTorch 1.7.1 with Python 3.8 in Windows 10 system. To generate the DTFG based on the dynamic stochastic block model, we utilized NetworkX with Python 3.8. The parameters of the sharded blockchain system and double DQN optimization schemes are presented in Table II. For different scenarios, the double DQN-based schemes are trained with different initial parameters.

TABLE II
Simulation Parameters

Symbol	Parameter	value
N	The total number of validators,	120
M	The total number of IoT devices,	180
$\dot{K}$	Maximum number of shards,	24
$\dot{S}$	Maximum block size,	8 Mb
$\dot{T}_I$	Maximum block interval,	16 s
λ	Average Transaction size,	200 $Bytes$
$F_{k,p}, F_{k,r}$	The computing capacity of validators [35],	10-30 Ghz
$R_{i,j}$	The transmission rates between nodes [35],	10-100 Mhz
C	The number of collaborative computing tasks,	3-11
p	The probability of malicious nodes,	3%
α	Computing cycles for generating/verifying signatures [35],	2 Mhz
β	Computing cycles for generating/verifying MACs [35],	1 Mhz
ε	Exploration probability of Double DQN,	0.9
Z	Replay memory buffer size,	4000
D^-	Mini-batch size of sampling,	128
$\dot{E}$	Max training episodes,	100000
$\dot{O}$	Max training steps,	200

A. Simulation Settings

For the consensus, we modeled the computational capacity of the primary nodes and the replica nodes as the Markov model. Assuming that the state of the computational capability can be low, medium, high, and very high, the computational capacity assigned to verify each transaction are from the set {10, 15, 25, 30} GHz [33], whose transition probability matrix $[p_\mu]_{L\times L}$ is set as

$$p_\mu = \begin{bmatrix} 0.45 & 0.3 & 0.15 & 0.1 \\ 0.3 & 0.45 & 0.15 & 0.1 \\ 0.15 & 0.3 & 0.45 & 0.1 \\ 0.15 & 0.1 & 0.3 & 0.45 \end{bmatrix}. \tag{28}$$

For the message transmission during consensus, we modeled the transmission rate among the nodes as the Markov model. Assuming that the state of the transmission rate can be low, medium, high, and very high, the transmission rates for each message are from the set {10, 30, 60, 90} MHz [33] whose

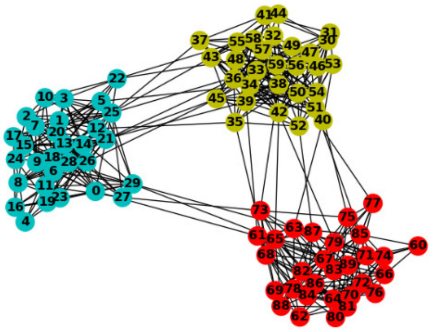

Fig. 4. Example of stochastic-block-model-based collaborative computing IoT.

transition probability matrix $[p_\sigma]_{J\times J}$ is set as

$$p_\sigma = \begin{bmatrix} 0.45 & 0.32 & 0.14 & 0.09 \\ 0.32 & 0.45 & 0.14 & 0.09 \\ 0.14 & 0.32 & 0.45 & 0.09 \\ 0.14 & 0.09 & 0.32 & 0.45 \end{bmatrix}. \tag{29}$$

For collaborative computing IoT, we modeled the collaborative computing tasks within the IoT network as a Markov model. Assuming that the state of the collaborative computing tasks can be very few, less, medium, much, and many, the collaborative computing tasks are from the set $\{3, 5, 7, 9, 11\}$ whose transition probability matrix $[p_C]_{H\times H}$ is set as

$$p_C = \begin{bmatrix} 0.35 & 0.27 & 0.2 & 0.13 & 0.05 \\ 0.27 & 0.35 & 0.2 & 0.13 & 0.05 \\ 0.2 & 0.27 & 0.35 & 0.13 & 0.05 \\ 0.13 & 0.2 & 0.27 & 0.35 & 0.05 \\ 0.05 & 0.13 & 0.2 & 0.27 & 0.35 \end{bmatrix}. \tag{30}$$

In addition, for the generation of DTFG based on the dynamic stochastic block model, we assume the IoT devices are all executing their computing tasks. Furthermore, the probability $\theta_{c(t),c'(t)}$ that a device of task c sends transaction to a device of task c' at time slot t is set to 0.01 and the probability $\theta_{c(t),c(t)}$ of the devices having task c send transactions to each other is set to 0.35 [37].

An example of the DTFG is shown in Fig. 4, where we assume there are three tasks and 90 IoT devices. The vertices represent the IoT devices and the edges represent the transactions generated among the devices. In addition, the devices were numbered by 0–89 and colored by blue, yellow, and red, which represent that the devices were allocated to finish the task 0, task 1, and task 2. Here, considering the evenly allocation of IoT devices and related to [37], the probability matrix $\boldsymbol{\theta}(t) = [\theta_{c(t),c'(t)}]$ is set to $\boldsymbol{\theta}(t) = [0.35, 0.01, 0.01; 0.01, 0.35, 0.01; 0.01, 0.01, 0.35]$, and the vector $\boldsymbol{\zeta}(t)$ is set to $\boldsymbol{\zeta}(t) = \{\underbrace{1, 1, \ldots, 1}_{30}, \underbrace{2, 2, \ldots, 2}_{30}, \underbrace{3, 3, \ldots, 3}_{30}\}$

In order to demonstrate the effectiveness of the proposed approach, the system performance is compared with several existing sharding schemes. For instance, the greedy-modularity-based sharding strategy and random network sharding strategy. Additionally, the impacts of different blockchain parameters are considered to ensure comparison fairness. For instance, different block sizes and different block intervals. Furthermore, considering the security constraint for the intra-shard consensus, the performance under different malicious is presented.

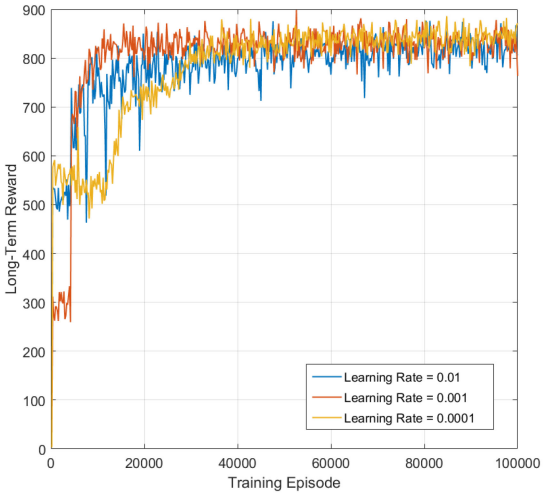

Fig. 5. Long-term reward under different learning rates.

B. Simulation Results and Discussion

Fig. 5 shows the convergence performance of the double DQN-based optimization scheme under the different learning rates, giving the long-term reward along with the training episodes. We can observe that the long-term reward is lower at the beginning of the learning process. With the increase of training episodes, the long-term reward rises and reaches a steady condition after about 20 000 training episodes. The convergence of the long-term reward of Algorithm 2 demonstrates that the double DQN agent has learned the optimal policy to maximize the action–state value function. Therefore, it can prove the effectiveness of the proposed optimization scheme in the considered scenario.

Additionally, as shown in Fig. 5, the long-term rewards keep a more stable performance with a lower learning rate. The reason is that the learning rate in DRL determines the learning effect of the network parameter updated by the gradient of the loss function. In other words, the higher the learning rate leads to faster updates of the parameters but also the greater the impact of anomalous data, which is susceptible to bias. Meanwhile, the lower learning rate is more likely to find the precise position of the global optimum point. Moreover, it can also be seen that performance with the learning rate of 0.001 enables a better convergence than that with 0.0001. Therefore, in the rest simulation results of this article, the learning rate is set to be 0.001.

Figs. 6 and 7 show the system throughput of the proposed scheme under different baselines which, respectively, fix the blockchain shards to 6, block interval to 10 s and block size to 2 MB. Moreover, as can be seen in Fig. 6, the proposed scheme can receive higher throughput than that of the other baselines. The DRL-based scheme without sharding is implemented by adjusting the block size and block interval with a single-chain system. It has the lowest throughput due to the absence of a sharding approach.

The reason is that the throughput is quantified by the largest processed transactions per second (TPS) of the blockchain. Increasing the block size means more transactions could be

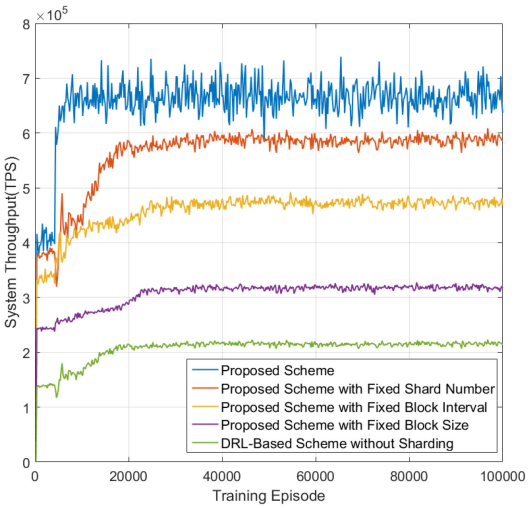

Fig. 6. System throughput under different baselines.

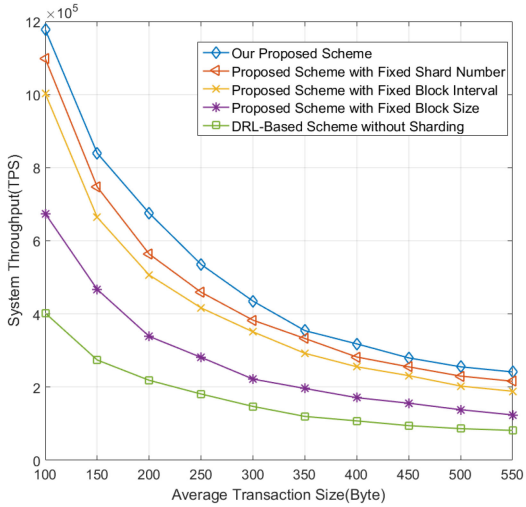

Fig. 7. System throughput with different transaction sizes.

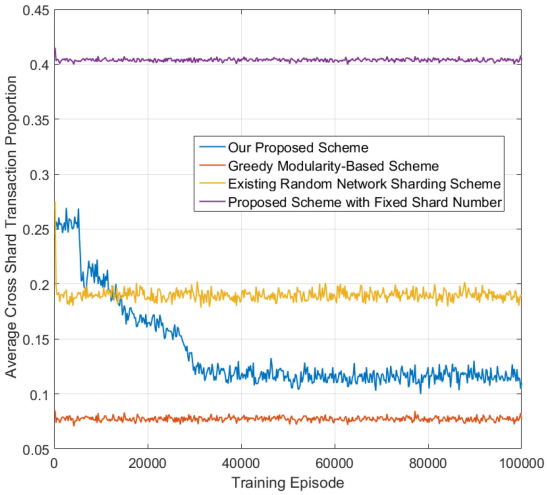

Fig. 8. Average CST proportion under different sharding strategies.

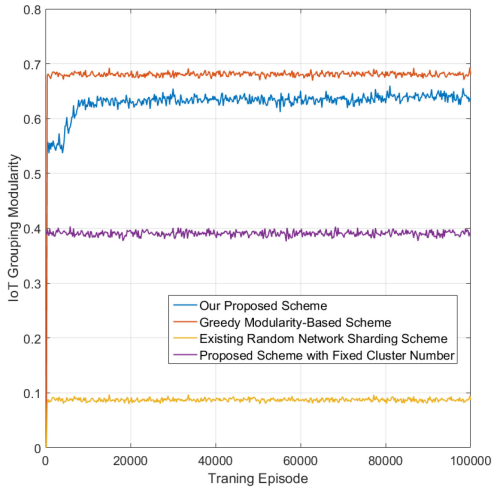

Fig. 9. IoT Grouping modularity under different sharding strategies.

packaged into a block and be processed in a time unit, while decreasing the block interval means in a time unit more blocks could be generated and also increase the TPS. Furthermore, the increasing of shard number means the more linear increased blocks are generated in a time unit. Hence, although one of the shard number, the block size and block interval is limited, the agent would try to increase the throughput by adjusting other parameters.

Additionally, it can be seen in Fig. 7 that with the average transaction size increasing, the throughput of the system decreases. The reason is that when the average transaction size increases, the number of transactions contained in one block decreases. On the other hand, the proposed scheme maintains the highest throughput with the different average transaction size compared with other baselines. The reason is that the proposed scheme is able to adjust the reasonable blockchain parameters to improve the system throughput.

Fig. 8 shows the average CST proportions with different sharding strategies, which are the greedy-modularity-based strategy, our proposed dynamic sharding scheme, the fixed shard number ($k = 6$) *K-means* strategy and the random network sharding strategy. Apparently, the CST proportion of our proposed scheme decreases with the increasing training episode, while the other strategies remain steady. The same situation emerges in Fig. 9, which shows the modularity of the schemes. With the increase of the training episodes, only the agent of our proposed scheme keeps trying to find a more reasonable shard number to group the IoT devices. Thus, according to the modularity contained in the reward function, the average CST proportions reduce with the training episodes and the modularity increases. The convergence of the CST proportion and the modularity demonstrates the effectiveness of the proposed scheme to reasonably group the IoT devices. Moreover, the greedy-modularity-based scheme has the lowest CST proportion, followed by the proposed scheme and the fixed shard number *K-means* strategy. The random network sharding has the highest proportion of CST. For the grouping modularity shown in Fig. 9, the greedy-based has the highest modularity and followed by our proposed scheme and the others.

Fig. 10 compares the throughput performances under different sharding strategies which are the greedy modularity-based sharding scheme, the existing random network sharding scheme, the existing work in [18] without learning method and

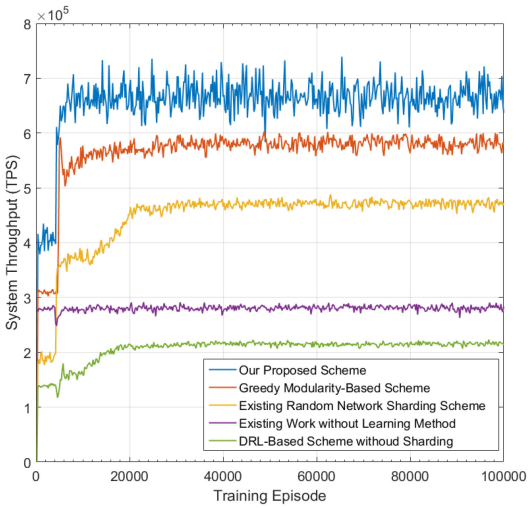

Fig. 10. System throughput under different sharding strategies.

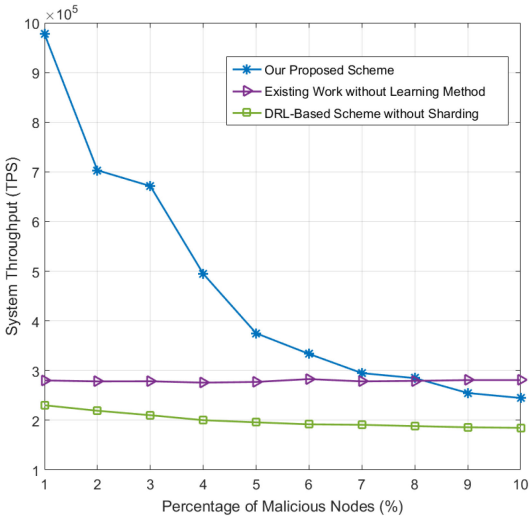

Fig. 11. System throughput with different malicious nodes proportions.

the existing DRL-based framework in [33] without sharding. Here, the greedy-modularity-based sharding scheme and the random network sharding scheme are also performed with the adjustment of block size and block interval. The existing work in [18] introduces parallel CZs, which scale the Ethereum blockchain system linearly with the shard number. For ease of comparison, we established a private blockchain based on [18] with the fixed 16 shards without a learning-based method. Moreover, the framework in [33] adopts a conventional DQN-based optimization framework for blockchain-enabled IIoT. The existing work in [18] remains steady during the training episode as the learning method is not applied. However, the framework in [33] performs the lowest throughput although it adopted the DRL agent for performance optimization. The main reason is that sharding can linearly improve the throughput. Obviously, the proposed scheme has the highest throughput than the other strategies as it jointly optimizes the shard number and other blockchain parameters. Moreover, although the greedy-modularity-based scheme can achieve higher quality IoT grouping, the system performance is bounded as the shard number is not optimized. Therefore, we can find that the proposed scheme can reasonably group IoT users while simultaneously guaranteeing the throughput performance of sharded blockchain systems.

Fig. 11 shows throughput degradation with raising of the proportion of malicious nodes. Here, we compare the proposed scheme with the existing work in [18] and the framework in [33]. Obviously, the throughput of the sharded Ethereum in [18] remains steady as the ratio increases. The reason is that Ethereum adopts the PoW consensus, where all the nodes calculate a specific hash value. The security could be guaranteed when the malicious nodes have no more than 51% computing power of the entire system. Even the entire malicious nodes gather into one shard, the system security will not be affected if they do not have sufficient computing resources. However, the PoW consensus has a large computational cost, which is not applicable in the considered IoT scenario.

On the other side, the proposed method remains the highest throughput compared with the other strategies before the malicious nodes ratio reached to 8%. The reason is that the

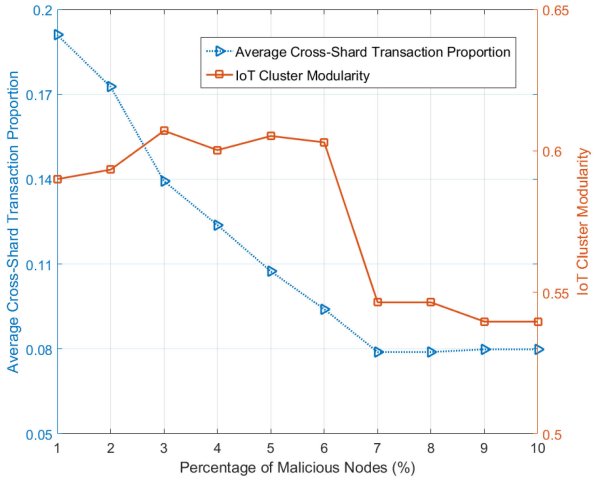

Fig. 12. Average CST proportion versus IoT grouping modularity under different malicious nodes proportions.

maximum shard number for security boundary decreases with the increase of malicious nodes. Specifically, the maximum shard number that satisfies the security constraint in (22) is 24 when the percentage of malicious nodes is 1%. While the percentage increases to 7%, to avoid the redundant malicious nodes gathering into one shard, the secure maximum shard number reduces to 4. In that case, with the increase of the malicious nodes ratio, the maximum system throughput is also affected.

Fig. 12 shows the impact of different percentages of malicious nodes on average CST proportion and IoT grouping modularity. We can observe that both of the two parameters reduce with the increasing of malicious nodes. In fact, the CST proportion and modularity are affected by the reduced shard number. The reason for the drop in CST is that more shards require the vertices to be divided into more parts. In that case, more edges will be recognized as the cross-group edges and vice versa. Additionally, the maximum shard number is bounded by a security constraint. When the percentage of malicious nodes increases from 1% to 3%, the maximum shard number decreased from 24 to 10, the range

for selecting the shard number changes from [2, 24] to [2, 10]. Considering the tasks states transit within {3, 5, 7, 9, 11} and the modularity is higher when the cluster number approaches the task number. In that case, the agent is more likely to select the number with higher modularity. However, when the percentage of malicious nodes increases to 10%, the range changes to [2, 4], accordingly, the agent is more likely to miss the number with higher modularity. That explains the modularity changes with different percentages of malicious nodes.

VII. Conclusion and Future Works

In this article, we proposed a blockchain sharding strategy for collaborative computing in the IoT, where the sharding of IoT was reasonable and the scalability of blockchain was improved. Specifically, the data flow among the IoT devices was described as DTFG based on dynamic stochastic block model, and accordingly, the sharding was implemented by devices grouping and consensus nodes assignment, which is realized by the *K-means* clustering algorithm and VRF, respectively. Furthermore, we formulate the sharding and the consensus protocol as a joint optimization problem, where the shard number, block size, and block interval are jointly trained in the double DQN agent. The simulation results illustrate the efficiency of the proposed scheme.

Future work is in progress to focus on advanced security methods which have less impact on the sharded blockchain performance based on the proposed framework. On the one hand, the future direction can consider the node-reputation-based mechanism which dynamically selects the validators with a higher trust level during the consensus process. On the other hand, the introduction of incentive/punishment mechanisms should be considered which can strict the nodes' behaviors. Additionally, to address the challenge of high computational cost, we consider adopting the DNN compression method [43] into the proposed scheme. Through pruning the redundant weights, the deployment of DNN can save the memory and the calculation time without affecting the accuracy, so as to address the issue of DNN deployment with limited hardware resources.

Acknowledgment

The authors thank the editor and reviewers for their detailed reviews and constructive comments, which have helped to improve the quality of this article.

References

[1] Y. Ai, L. Wang, Z. Han, P. Zhang, and L. Hanzo, "Social networking and caching aided collaborative computing for the Internet of Things," *IEEE Commun. Mag.*, vol. 56, no. 12, pp. 149–155, Dec. 2018.

[2] B. Tang, H. Kang, J. Fan, Q. Li, and R. Sandhu, "IoT passport: A blockchain-based trust framework for collaborative Internet-of-Things," in *Proc. 24th ACM Symp. Access Control Models Technol. (SACMAT)*, New York, NY, USA, 2019, pp. 83–92.

[3] A. Asheralieva and D. Niyato, "Reputation-based coalition formation for secure self-organized and scalable sharding in IoT blockchains with mobile-edge computing," *IEEE Internet Things J.*, vol. 7, no. 12, pp. 11830–11850, Dec. 2020.

[4] K. Yeow, A. Gani, R. W. Ahmad, J. J. P. C. Rodrigues, and K. Ko, "Decentralized consensus for edge-centric Internet of Things: A review, taxonomy, and research issues," *IEEE Access*, vol. 6, pp. 1513–1524, 2018.

[5] J. Kang *et al.*, "Blockchain for secure and efficient data sharing in vehicular edge computing and networks," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 4660–4670, Jun. 2019.

[6] H. Liu, Y. Zhang, and T. Yang, "Blockchain-enabled security in electric vehicles cloud and edge computing," *IEEE Netw.*, vol. 32, no. 3, pp. 78–83, May/Jun. 2018.

[7] O. Novo, "Blockchain meets IoT an architecture for scalable access management in IoT," *IEEE Internet Things J.*, vol. 5, no. 2, pp. 1184–1195, Apr. 2018.

[8] B. Wu, K. Xu, Q. Li, S. Ren, Z. Liu, and Z. Zhang, "Toward blockchain-powered trusted collaborative services for edge-centric networks," *IEEE Netw.*, vol. 34, no. 2, pp. 30–36, Mar./Apr. 2020.

[9] C. Qiu, H. Yao, F. R. Yu, C. Jiang, and S. Guo, "A service-oriented permissioned blockchain for the Internet of Things," *IEEE Trans. Service Comput.*, vol. 13, no. 2, pp. 203–215, Mar./Apr. 2020.

[10] O. Novo, "Scalable access management in IoT using blockchain: A performance evaluation," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 4694–4701, Jun. 2019.

[11] S. Nakamoto. *Bitcoin: A Peer-to-Peer Electronic Cash System*, (2009). [Online]. Available: http://www.bitcoin.org/bitcoin.pdf

[12] G. Wood, "Ethereum: A secure decentralised generalised transaction ledger (EIP-150 revision)," Zug, Switzerland, Ethereum, Yellow Paper, 2017.

[13] C. Qiu, F. R. Yu, H. Yao, C. Jiang, F. Xu, and C. Zhao, "Blockchain-based software-defined Industrial Internet of Things: A dueling deep Q-learning approach," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 4627–4639, Jun. 2019.

[14] J. Kang, Z. Xiong, D. Niyato, D. Ye, D. I. Kim, and J. Zhao, "Toward secure blockchain-enabled Internet of Vehicles: Optimizing consensus management using reputation and contract theory," *IEEE Trans. Veh. Technol.*, vol. 68, no. 3, pp. 2906–2920, Mar. 2019.

[15] W. Wang, D. Niyato, P. Wang, and A. Leshem, "Decentralized caching for content delivery based on blockchain: A game theoretic perspective," in *Proc. IEEE Int. Conf. Commun.*, Kansas City, MO, USA, May 2018, pp. 1–6.

[16] J. Kang, R. Yu, X. Huang, S. Maharjan, Y. Zhang, and E. Hossain, "Enabling localized peer-to-peer electricity trading among plug-in hybrid electric vehicles using consortium blockchains," *IEEE Trans. Ind. Informat.*, vol. 13, no. 6, pp. 3154–3164, Dec. 2017.

[17] Zilliqa. "The Zilliqa technical whitepaper v0.1." Aug. 2017. [Online]. Available: https://docs.zilliqa.com/whitepaper.pdf

[18] V. Buterin. "Ethereum Sharding FAQ." Apr. 2019. [Online]. Available: https://github.com/ethereum/wiki/wiki/ShardingFAQ

[19] J. Wang and H. Wang, "Monoxide: Scale out blockchains with synchronous consensus zones," in *Proc. NSDI*, 2019, pp. 95–112.

[20] T. Duong, L. Fan, T. Veale, and H. S. Zhou. "Securing Bitcoin-Like Backbone Protocols Against a Malicious Majority of Computing Power." 2016. [Online]. Available: https://allquantor.at/blockchainbib/pdf/duong2016securing.pdf

[21] 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 Things J.*, vol. 7, no. 10, pp. 9399–9412, Oct. 2020.

[22] Z. Li, J. Kang, R. Yu, D. Ye, Q. Deng, and Y. Zhang, "Consortium blockchain for secure energy trading in Industrial Internet of Things," *IEEE Trans. Ind. Informat.*, vol. 14, no. 8, pp. 3690–3700, Aug. 2018.

[23] A. Azaria, A. Ekblaw, T. Vieira, and A. Lippman, "MedRec: Using blockchain for medical data access and permission management," in *Proc. Int. Conf. Open Big Data (OBD)*, 2016, pp. 25–30.

[24] W. Wang *et al.*, "A survey on consensus mechanisms and mining strategy management in blockchain networks," *IEEE Access*, vol. 7, pp. 22328–22370, 2019.

[25] F. Guo, F. R. Yu, H. Zhang, H. Ji, M. Liu, and V. C. M. Leung, "Adaptive resource allocation in future wireless networks with blockchain and mobile edge computing," *IEEE Trans. Wireless Commun.*, vol. 19, no. 3, pp. 1689–1703, Mar. 2020.

[26] S. Biswas, K. Sharif, F. Li, S. Maharjan, S. P. Mohanty, and Y. Wang, "PoBT: A lightweight consensus algorithm for scalable IoT business blockchain," *IEEE Internet of Things J.*, vol. 7, no. 3, pp. 2343–2355, Mar. 2020.

[27] E. Kokoris-Kogias, P. Jovanovic, L. Gasser, N. Gailly, E. Syta, and B. Ford, "OmniLedger: A secure, scale-out, decentralized ledger via sharding," in *Proc. IEEE Security Privacy*, 2018, pp. 583–598.

[28] M. Fleder, M. S. Kester, and S. Pillai, "Bitcoin transaction graph analysis," 2015, *arXiv:1502.01657*.
[29] T. Chen *et al.*, "Understanding Ethereum via graph analysis," in *Proc. IEEE INFOCOM.*, Honolulu, HI, USA, Apr. 2018, pp. 1484–1492.
[30] D. Lin, J. Wu, Q. Yuan, and Z. Zheng, "Modeling and understanding Ethereum transaction records via a complex network approach," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 67, no. 11, pp. 2737–2741, Nov. 2020.
[31] V. Mnih *et al.*, "Playing Atari with deep reinforcement learning," in *Proc. NIPS Deep Learn. Workshop*, 2013, pp. 1–9.
[32] F. R. Yu, J. Liu, Y. He, P. Si, and Y. Zhang, "Virtualization for distributed ledger technology (vDLT)," *IEEE Access*, vol. 6, pp. 25019–25028, Apr. 2018.
[33] M. Liu, F. R. Yu, Y. Teng, V. C. M. Leung, and M. Song, "Performance optimization for blockchain-enabled Industrial Internet of Things (IIoT) systems: A deep reinforcement learning approach," *IEEE Trans. Ind. Informat.*, vol. 15, no. 6, pp. 3559–3570, Jun. 2019.
[34] J. Yun, Y. Goh, and J.-M. Chung, "DQN-based optimization framework for secure sharded blockchain systems," *IEEE Internet Things J.*, vol. 8, no. 2, pp. 708–722, Jan. 2021.
[35] B. Brenda, A. Rodriguez, and N. Boyd, "Modelling and prediction of financial trading networks: An application to the NYMEX natural gas futures market," *J. Royal Stat. Soc. C*, vol. 69, no. 1, pp. 195–218, Nov. 2019.
[36] X. Zhang, C. Moore, and M. E. J. Newman, "Random graph models for dynamic networks," *Eur. Phys. J. B*, vol. 90, no. 10, pp. 1111–1136, Dec. 2017.
[37] C. Lee and D. J. Wilkinson, "A review of stochastic block models and extensions for graph clustering," *Appl. Netw. Sci.*, vol. 4, no. 1, p. 122, 2019.
[38] P. Jaccard, "The distribution of the flora in the alpine zone," *New Phytol.*, vol. 11, no. 2, pp. 37–50, 1912.
[39] M. E. J. Newman, "Modularity and community structure in networks," *Proc. Nat. Acad. Sci.*, vol. 103, no. 23, pp. 8577–8582, Jun. 2006.
[40] A. Paszke. *Reinforcement Learning (DQN) Tutorial*, (2018). [Online]. Available: https://pytorch.org/tutorials/intermediate/reinforcement_q_learning.html
[41] H. V. Hasselt, "Double Q-learning," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 23, 2010, pp. 2613–2621. [Online]. Available: http://papers.nips.cc/ paper/3964-double-q-learning.pdf
[42] G. Dulac-Arnold *et al.*, "Deep reinforcement learning in large discrete action spaces," 2016, *arXiv:1512.07679*.
[43] J.-H. Luo, J. Wu, and W. Lin, "ThiNet: A filter level pruning method for deep neural network compression," in *Proc. IEEE Int. Conf. Comput. Vis.*, Venice, Italy, Dec. 2017, pp. 5068–5076.

Zhaoxin Yang received the B.S. and M.S. degrees from Beijing University of Technology, Beijing, China, in 2015 and 2018, respectively, where he is currently pursuing the Ph.D. degree.

From October 2019 to April 2020, he was with Carleton University, Ottawa, ON, Canada, as a visiting Ph.D. student funded by Beijing University of Technology. His current research interests include big data, blockchain, and machine learning.

Ruizhe Yang received the B.S. degree from Beijing University of Technology, Beijing, China, in 2004, and the Ph.D. degree from Beijing University of Posts and Telecommunications, Beijing, in 2009.

She joined Beijing University of Technology in 2009, where she is currently an Associate Professor. From 2017 to 2018, she visited Carleton University, Ottawa, ON, Canada. Her research interests include blockchain, resource management, and channel estimation.

F. Richard Yu (Fellow, IEEE) received the Ph.D. degree in electrical engineering from the University of British Columbia, Vancouver, BC, Canada, in 2003.

From 2002 to 2006, he was with Ericsson, Lund, Sweden, and a start-up in California, USA. He joined Carleton University, Ottawa, ON, Canada, in 2007, where he is currently a Professor. His research interests include connected/autonomous vehicles, security, artificial intelligence, blockchain, and wireless cyber–physical systems.

Dr. Yu received the IEEE TCGCC Best Journal Paper Award in 2019, the Distinguished Service Awards in 2016 and 2019, the Outstanding Leadership Award in 2013, the Carleton Research Achievement Awards in 2012 and 2021, the Ontario Early Researcher Award (formerly Premiers Research Excellence Award) in 2011, the Excellent Contribution Award at IEEE/IFIP TrustCom'10, the Leadership Opportunity Fund Award from Canada Foundation of Innovation in 2009, and the Best Paper Awards at IEEE ICNC'18, VTC'17 Spring, ICC'14, Globecom'12, IEEE/IFIP TrustCom'09, and International Conference on Networking'05. He serves on the editorial boards of several journals, including the Co-Editor-in-Chief for *Ad Hoc and Sensor Wireless Networks*, and a Lead Series Editor for IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY, IEEE COMMUNICATIONS SURVEYS AND TUTORIALS, and IEEE TRANSACTIONS ON GREEN COMMUNICATIONS AND NETWORKING. He has served as the technical program committee co-chair of numerous conferences. He has been named in the Clarivate Analytics list of "Highly Cited Researchers" in 2019 and 2020. He is an IEEE Distinguished Lecturer of both Vehicular Technology Society (VTS) and Communications Society. He is an Elected Member of the Board of Governors of the IEEE VTS and the Editor-in-Chief for IEEE VTS Mobile World Newsletter. He is a registered Professional Engineer in the province of Ontario, Canada, and a Fellow of the Canadian Academy of Engineering, the Engineering Institute of Canada, and the Institution of Engineering and Technology.

Meng Li (Member, IEEE) received the Ph.D. degree in electronic science and technology from Beijing University of Technology, Beijing, China, in 2018.

He is currently an Associate Professor with the Faculty of Information Technology, Beijing University of Technology. From September 2015 to September 2016, he visited Carleton University, Ottawa, ON, Canada, as a visiting Ph.D. student funded by China Scholarship Council. His current research interests include M2M communications, industrial Internet, intelligent edge computing, and blockchain.

Dr. Li has received the Excellent Doctoral Dissertation Award from China Education Society of Electronics in 2019 and the Best Paper Award at IEEE ICCC 2021. He has served as the Technical Program Committee Member of several international conferences, including the IEEE GLOBECOM, IEEE ICC, and IEEE 5GWF.

Yanhua Zhang received the B.E. degree from Xi'an University of Technology, Xi'an, China, in 1982, and the M.S. degree from Lanzhou University, Lanzhou, China, in 1988.

From 1982 to 1990, he was with the Jiuquan Satellite Launch Center, Jiuquan, China. During the 1990s, he is a Visiting Professor with Concordia University, Montreal, QC, Canada. He joined Beijing University of Technology, Beijing, China, in 1997, where he is currently a Professor. His research interests include QoS-aware networking and radio resource management in wireless networks.

Yinglei Teng (Senior Member, IEEE) received the B.S. degree from Shandong University, Jinan, China, in 2005, and the Ph.D. degree in electrical engineering from Beijing University of Posts and Telecommunications (BUPT), Beijing, China, in 2011.

She is currently a Professor with Key Laboratory of Space-Ground Interconnection and Convergence, School of Electronic Engineering, BUPT. Her current research interests include AI-enabled IoTs and massive MIMO.