A Blockchain Dynamic Sharding Scheme Based on Hidden Markov Model in Collaborative IoT

Jinwen Xi, Guosheng Xu, *Member, IEEE*, Shihong Zou, *Member, IEEE*, Yueming Lu, Guoqiang Li, Jiuyun Xu, *Member, IEEE*, and Ruisheng Wang

***Abstract*—Sharded blockchain offers scalability, decentralization, immutability, and linear improvement, making it a promising solution for addressing the trust problem in large-scale collaborative IoT. However, a high proportion of cross-shard transactions (CSTs) can severely limit the performance of decentralized blockchain. Furthermore, the dynamic assemblage characteristic of collaborative sensing in sharded blockchain is often ignored. To overcome these limitations, we propose HMMDShard, a dynamic blockchain sharding scheme based on the hidden Markov model (HMM). HMMDShard leverages fine-grained blockchain sharding and fully embraces the dynamic assemblage characteristic of IoT collaborative sensing. By integrating the HMM, we achieve adaptive dynamic incremental updating of blockchain shards, effectively reducing CSTs across all shards. We conduct a comprehensive analysis of the security issues and properties of HMMDShard, and evaluate its performance through the implementation of a system prototype. The results demonstrate that HMMDShard significantly reduces the proportion of CSTs and outperforms other baselines in terms of system throughput and transaction confirmation latency.**

***Index Terms*—Blockchain sharding, dynamic incremental updating, hidden Markov model (HMM), IoT collaborative sensing.**

I. Introduction

THE PROLIFERATION of IoT devices and their constant connection to wireless networks has led to a diverse range of sensing and computing tasks generated by various applications, which are transmitted to IoT networks, enabling data sharing among IoT devices. This has given rise to the Collaborative Internet of Things (CIoT), which is based on IoT devices, centered around individuals or nodes, and associated with various application scenarios. CIoT leverages personal smart devices, sensors, and agreed-upon protocols to establish connections between people, things, and other information resources. By doing so, CIoT enables personalized, high quality, and convenient experiences for individual users. In essence, CIoT facilitates seamless integration and sharing of information and resources, paving the way for a more interconnected and efficient world.

The decentralization, openness, transparency, and tamper resistance offered by the blockchain technology have positioned it as a popular solution for solving trust problems in multiparty IoT collaborative sensing scenarios. However, due to its inherent scalability limitations, the system throughput and transaction confirm latency struggle to handle massive IoT collaborative sensing tasks. Therefore, this article aims to propose an effective solution for improving scalability and enhancing the efficiency and performance of blockchain in managing large-scale IoT sensing tasks [1].

The sharding technique is widely recognized as the most promising solution for improving blockchain scalability. By adopting the "divide and conquer" approach, sharding partitions the transaction processing workload into node groups, thereby overcoming performance and scalability limitations. In recent years, several notable sharding solutions have been proposed, including Elastico [2], OmniLedger [3], RapidChain [4], TEEShard [5], and Monoxide [6]. Ethereum 2.0 [7] proposed a sharded blockchain combined with the beacon chain, dividing the entire network into multiple subnetworks to achieve a linear improvement in transaction throughput. Furthermore, dynamic sharding schemes, such as DBShard [8] and DRLBShard [9] have been proposed, enabling the size and number of shards to be adjusted without impacting the operation of the blockchain network. The rise of blockchain sharding technology has led to the emergence of numerous systems and applications based on sharded blockchain [10], [11], [12].

Similar to a social network, blockchain-based CIoT typically exhibit a community structure with dynamic assemblage characteristics, where node groups have denser internal connections than the rest of the network [13]. In the context of collaborative sensing networks, task members frequently engage in transactions due to shared attributes or roles, leading to the formation of community structures. To draw a parallel, the nodes in a social network correspond to social

Manuscript received 13 June 2023; revised 29 June 2023; accepted 7 July 2023. Date of publication 11 July 2023; date of current version 8 August 2023. This work was supported by the National Key Research and Development Program of China under Grant 2021YFB3101500. *(Jinwen Xi and Guosheng Xu are co-first authors.) (Corresponding author: Shihong Zou.)*

Jinwen Xi was with the National Engineering Laboratory of Mobile Network Security, the School of Cyberspace Security, Beijing University of Posts and Telecommunications, Beijing 100088, China. He is now with the Third Department of Scientific Research, Zhongguancun Laboratory, Beijing, China (e-mail: xijw@zgclab.edu.cn).

Guosheng Xu and Shihong Zou are with the National Engineering Laboratory of Mobile Network Security, the School of Cyberspace Security, Beijing University of Posts and Telecommunications, Beijing 100088, China (e-mail: guoshengxu@bupt.edu.cn; zoush@bupt.edu.cn).

Yueming Lu is with the School of Cyberspace Security, Beijing University of Posts and Telecommunications, Beijing 100088, China.

Guoqiang Li is with the Institutes of Science and Development, Chinese Academy of Sciences, Beijing 100045, China.

Jiuyun Xu is with the College of Computer Science and Technology, China University of Petroleum (Eastern China), Qingdao 266024, China.

Ruisheng Wang is with the Beijing University of Posts and Telecommunications, Beijing 100088, China, on leave from the Huazhong University of Science and Technology, Wuhan 430074, China.

Digital Object Identifier 10.1109/JIOT.2023.3294234

members, while IoT devices represent nodes in CIoT networks. The edges in the social network correspond to information exchange between members, while in CIoT networks, they correspond to data sharing or transactions. The community structure observed in the social network can be analogous to the shard structure in CIoT networks.

Motivation: Existing sharding solutions typically rely on a top-level trusted leader or reference committee to perform the sharding of bottom node partitions, such as network sharding and random sharding. However, these "top-bottom" sharding solutions ignore the dynamic assemblage characteristics of participating nodes, such as node groups with denser internal connections than the rest of the network [13]. This oversight results in a significant number of cross-shard transactions (CSTs) in blockchain-based systems. As the number of shards increases, nearly all transactions become CSTs, leading to additional overhead and potentially infinite transaction confirmation latency [3]. To address this problem, we propose a novel bottom-top dynamic sharding solution called HMMDShard. This approach aims to mitigate the overhead caused by CSTs by providing an alternative sharding mechanism that does not rely on a trusted third party.

Challenges: To implement a scalable sharded blockchain system that fully considers the dynamic characteristics, several technical challenges must be addressed. First, sharding protocols need to incorporate the dynamic assemblage characteristics as rules, but the absence of modeling methods for analyzing these characteristics of blockchain transactions is a critical issue. Therefore, a suitable modeling approach needs to be developed. Moreover, the current sharded blockchain systems suffer from low throughput and high latency when the proportion of CSTs is high. Reducing the proportion of CSTs presents another significant challenge that hampers the practical adoption of a larger number of shards in real-world applications [9].

To this end, this article proposes HMMDShard, a dynamic sharding scheme for blockchain-based IoT collaborative sensing. The main objective of HMMDShard is to minimize the occurrence of CSTs and enhance the overall system performance.

Contributions: Our study leads to the following contributions.

1) We present a novel dynamic and hierarchical blockchain-based sharding architecture specifically designed for IoT collaborative sensing, which operates in parallel and can effectively handle both intrashard and intershard transactions.
2) We are the first to introduce a novel approach to construct a dynamic transaction graph based on blockchain transactions and utilize the hidden Markov model (HMM) to propose a dynamic fine-grained incremental sharding scheme, which significantly reduces the number of CSTs and improves system performance.
3) We implement the HMMDShard protocol and deploy a fully functional prototype on the AliCloud platform. Through detailed theoretical analysis and experiments, we demonstrate that HMMDShard outperforms state-of-the-art baselines in terms of the proportion of CSTs, modularity, transaction throughput, and confirmation latency.

The remainder of this article is organized as follows: Section II presents a review of state-of-the-art studies and Section III provides an overview of the system model. In Section IV, we introduce the proposed dynamic sharding scheme based on HMM. Section V analyzes the security issues and experimental results. Finally, Section VI concludes this article.

II. Related Work

Blockchain-Based IoT: The increasing ingetration of IoT technology across diverse industries has led to a demand for adaptable and large-scale cross-industry platforms capable of aggregating data from various fields to establish complex industrial networks [14]. Due to the properties of blockchain in terms of decentralization, transparency, and immutability, researchers in both academia and industry have proposed numerous solutions that apply the blockchain technology to IoT systems [15]. These proposed solutions span various application scenarios, such as Smart Home, Smart Transportation, Smart Campus, Smart Office, Smart Factory, and Smart Medical, highlighting the versatility of blockchain in enabling innovative IoT deployments.

Current efforts in blockchain-enabled IoT systems primarily revolve around the design of frameworks and consensus protocols that leverage the blockchain structure to ensure the security and reliability of IoT data. Various application domains have witnessed the development of such systems. For example, in an Industrial IoT environment, a consortium blockchain-based energy system has been proposed to facilitate secure energy transactions [16]. In medical scenarios, Zaabar et al. [17] have proposed a secure blockchain-based healthcare data management system that uses an efficient consensus protocol to reduce memory and communication overhead for blockchain validators. Additionally, a novel lightweight blockchain has been proposed to reduce memory requirements and computation time of IoT [18].

Indeed, on of the key considerations in enabling blockchain-based IoT systems is the system performance, specifically in terms of scalability and latency. As different IoT scenarios may have distinct service quality requirements, it becomes necessary to adjust the blockchain system during the deployment process to meet these diverse requirements effectively.

Blockchain Sharding Protocols: The sharding technology, originally derived from traditional large-scale databases, has experienced significant advancements in recent years. It improves overall system performance by dividing data into multiple data slices. Recognizing the potential benefits of data sharding, researchers have explored its application in the context of blockchain to address the scalability challenge. With the continuous expansion of blockchain application scenarios, transaction throughput has emerged as a crucial performance indicator. Increasing the capacity to handle a higher number of transactions per second has become a focus for improving system performance. In this regard, the sharding technology

has garnered significant attention as a promising approach to enhance the scalability of blockchain systems [19].

Throughput and latency are crucial metrics for evaluating the performance of sharded blockchain. Consequently, the handling of CSTs plays a significant role in ensuring the atomicity and efficiency of transactions, researchers have proposed various solutions to address this issue. Here are some examples, OmniLedger [3] uses a client-driven two-phase commit (2PC) mechanism with lock/unlock operations. RapidChain [4] applies subtransactions to transfer all involved UTXOs to the same shard for transforming a CST to several intrashard transactions. Wang and Wang [6] proposed a scale-out sharded blockchain with asynchronous consensus zones, which reduces the overhead of CST consensus by applying eventual atomicity. While Dang et al. [5] proposed a hardware-assist sharding scheme that tolerates 1/2 of the Byzantine nodes. Pyramid [11] introduces a layered sharding consensus protocol that involves special types of nodes serving two or more shards simultaneously.

Recently, dynamic sharding schemes, such as DBShard [8] and DRLBShard [9] have also been proposed. However, these schemes have certain limitations that can impact their effectiveness and practicality. DBShard focuses primarily on adjusting the size and number of shards to optimize system performance, but it does not specifically address the handling of CSTs. On the other hand, DRLShard optimizes the shard number and adjusts the consensus parameters using deep reinforcement learning (DRL), which introduces additional computational overhead and may not be suitable for resource-constrained blockchain networks. In contrast, the proposed HMMDShard takes a different approach by leveraging the dynamic assemblage characteristics of nodes to handle CSTs. Additionally, the "bottom-top" sharding scheme aims to improve throughput by reducing the proportion of CSTs.

Graph Analysis: Graph analysis is a commonly used approach for analyzing complex networks, as it represents users and their interactions as nodes and edges in a network. Researchers have applied graph analysis techniques to gain insights into the structure, behavior, and dynamics of blockchain transaction networks [20], [21], [22]. For example, a graph analysis framework is proposed to track and cluster user activities in the Bitcoin network [20]. Similarly, Chen et al. [21] conducted a systematic study with graph analysis to describe fund transfers, smart contract creation, and smart contract invocation in the Ethereum network. Another study [22] models the transaction records on Ethereum are modeled as a static simple graph, which is used to predict the linking relationship of nodes.

Therefore, the analysis of IoT device interactions using graph theory can provide a valuable theoretical foundation for the development of dynamic sharding in blockchain systems.

III. Overview of System Model

In this section, we present HMMDShard, a permissioned and sharded consortium blockchain system specifically designed for collaborative sensing. The system is built upon the device–edge–cloud hierarchical architecture, as shown in Fig. 1. It consists of several key components: trusted authority (TA), servers, shards, and IoT devices. The flexibility and versatility of HMMDShard enable its application in various scenarios, such as Smart Home, Smart Transportation, and more. In HMMDShard, all IoT devices are seamlessly integrated into the blockchain network, with different types of devices able to perform distinct roles according to their limited resources (processing, memory, etc.) and enable wireless communication (e.g., Cellular and Wi-Fi). For instance, in static scenarios, such as smart home, devices are less likely to move frequently, but there will be a continuous addition of new devices and replacement of old ones. On the other hand, in dynamic scenarios, such as smart transportation, vehicle nodes may frequently join or leave shards due to the constant movement of vehicles. To ensure clarity and understanding, we provide Table I, which contains a comprehensive list of symbols and notations used throughout our proposed methodology.

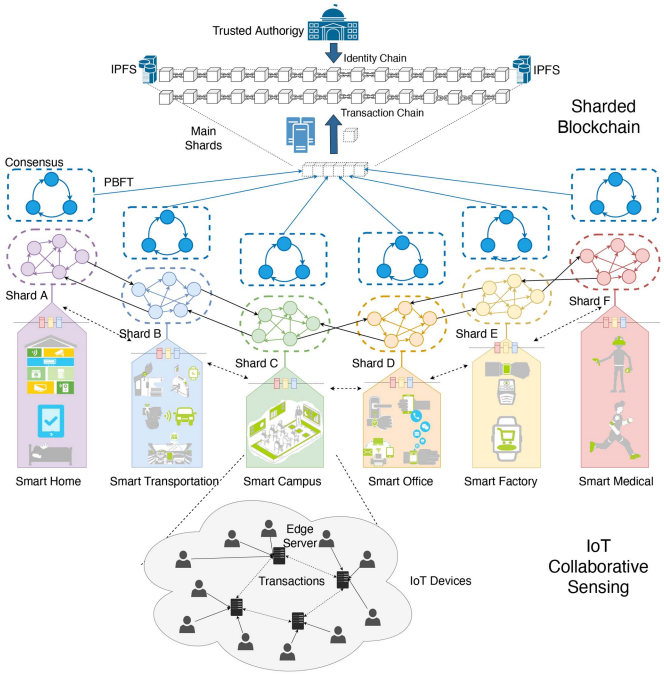


Fig. 1. Blockchain-based device–edge–cloud collaborative sensing architecture.

TABLE I
Symbols and Notations

No	Symbols	Notations
1	(pk_i, sk_i)	public-private key pair of node i
2	n, N	Number of network nodes
3	m, M	Number of communities/shards
4	f	Number of Byzantine nodes
5	v, V	Validators, Set of validators
6	$\delta, S/\mathcal{S}$	Center node, Shards
7	GS	Transaction graph snapshot
8	$\mathbf{A}$	State transition probability matrix
9	$\mathbf{B}$	State observation probability matrix
10	Π	Initial state probability distribution
11	$\mathcal{O}$	Observable sequence
12	$\mathcal{C}/C$	Community structure

A. Trusted Authority

In HMMDShard, TA has two main functions.

1) *User Registration:* Before participating in collaborative sensing activities, users are required to generate a public-private key pair (pk, sk) using the key generator. The generated public key pk is sent to TA and requests a certificate that binds their identity information to the public key. The certificate provides a trusted guarantee between user identity information and pk. TA operates as a reliable and trustworthy entity responsible for managing the registration process. Users are required to provide authentic documents or information to prove their personal identity, such as an ID number, name, gender, date of birth, place of birth, blood type, etc. TA creates a unique identity ID for each user to identify them. Furthermore, different users will also be assigned different identities that represent their social roles in reality. All registration information is stored in the identity chain, which ensures the security and immutability of user information.
2) *IoT Entity Registration:* Furthermore, TA is responsible for creating an account for each IoT entity and linking the account address to the unique identifier of a sensor or a smart device, i.e., MAC address. For this function, TA is treated as an anonymous system, and accounts are generated by assigned key pairs. Likewise, all registration information will be stored in the identity chain.

B. Interplanetary File System Server

Interplanetary file system server (IPFS) is mainly used to store and share specific data in the distributed system, such as sensor readings and specific information, then the data-related indexes stored in the blockchain, thereby significantly reducing storage overhead.

C. Edge Server

There are differences in functions between IoT devices, and those devices with powerful functions, i.e., routers and gateways, can act as edge servers to perform complex computations and storage tasks.

D. Main Shards

The main shard of HMMDShard is composed of high-performance servers chosen from subshards. Each server functions as a full node and is responsible for maintaining the global ledger, as well as managing the final consensus of CSTs and blocks uploaded by subshards, broadcasting consensus, results, or blocks to all other nodes in the entire network, ensuring that all nodes are up-to-date and synchronized.

E. Subshards

Each field (different IoT scenarios) can be divided into multiple subshards, each of which comprises multiple blockchain nodes to jointly maintain a local blockchain ledger. For example, in the Smart Home scenario, the subshards could correspond to specific sections of the home, such as individual rooms or areas, which consists of home appliances with sensors and a large number of wearable smart terminals that communicate through available network links with routers or gateways. The three blue circles above each scenario indicate the consensus process and the five color-filled circles indicate the nodes in the shard. For a CST, for example, the data sharing transaction generated for serving users, such as a smart home scenario where you can access the user's location information in a smart transportation scenario to change the status of smart devices in the home, since both sides of the transaction come from different shards, the subshard cannot process it, so the transaction will be submitted to the consensus committee in the main shard for reaching consensus, and if consensus passed, it will be recorded in the blockchain, otherwise discarded.

F. IoT Entities and Peers/Nodes

The blockchain-based network facilitates collaboration among IoT entities, including servers, gateways, and devices, in a "decentralized" manner. Each IoT entity can host one or more blockchain peers/nodes, which are connected to other peers/nodes through decentralized applications.

In IoT, users can engage in collaborative sensing tasks, utilizing IoT devices or sensors to collect various types of data to fulfill diverse application requirements (e.g., smart city, supply chains, autonomous driving, and environmental monitoring). Meanwhile, data sharing (transactions) occurs among IoT devices for the same collaborative sensing task or across different tasks in different fields. Thus each user is able to register the personal account that containing addresses in each subshard.

We consider a weak synchronous network of N nodes with $f \leq \lfloor (N-1)/3 \rfloor$ as the one in [23]. Therefore, we adopt PBFT protocol for achieving intrashard consensus and preventing blockchain forks in both main shards and subshards.

IV. Blockchain Dynamic Sharding Scheme Based on Hidden Markov Model

To address the challenge of dynamically evolving blockchain transaction networks, this section proposes a novel blockchain dynamic sharding scheme based on the HMM [24]. Specifically, we first employ the blockchain technology to implement transaction data storage and management for data tampering resistance, ensuring the data security during model construction. Then, we utilize the collective knowledge and transaction history of blockchain peers, along with on-chain transaction records to construct a dynamic transaction sensing flow graph (DTSFG), and use the built HMM to detect the dynamic community structures in the graph to form shards. Finally, the IoT devices are allocated to the corresponding shards in a bottom-top manner.

A. Building Dynamic Transaction Sensing Flow Graphs and Hidden Markov Model

First of all, the transactions between nodes can be represented by a transaction flow graph model [money flow graph (MFG)] [21]. For the data-sharing process, we assume that

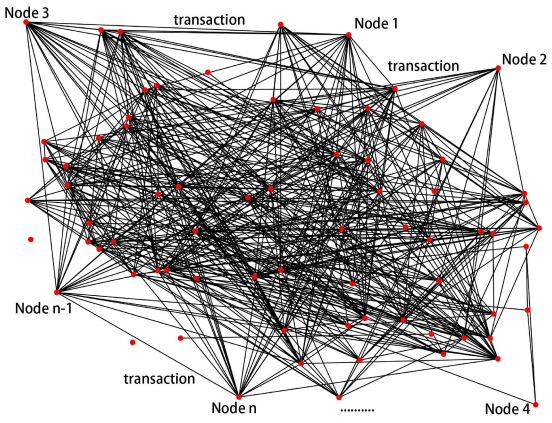


Fig. 2. Snapshot of the DTSFG at time t. (Vertices represent peers and edges represent transactions between peers).

each IoT device sends only one transaction to another IoT device at time t. Then $\mathcal{G} = (V, E, T)$ can be used to represent the blockchain transaction network, where V represents the node set, E represents the transaction set between nodes, and T represents the transaction time to capture the blockchain network changes. Then, the transaction behavior relationship between nodes at time t can be represented as a snapshot of the DTSFG. As shown in Fig. 2, the vertices in the graph represent all participating IoT devices, and the edges represent the transaction flow between devices. The unconnected peers indicate that no transactions have been generated since joining the network.

Definition 1 [Graph Snapshot (GS)*]:* That is, the transaction behavior relationship of network nodes at a certain moment, for the time-varying series within the dynamic network duration time T, can be represented by using a set of graph snapshots, $GS_T = \{GS^{(1)}, GS^{(2)}, \ldots, GS^{(t)}, \ldots, GS^{(T)}\}$. At the same time, to represent the interaction between devices, the DTSFG snapshots at time t can be encoded in the form of a matrix. Assuming that there are N IoT devices, the $N \times N$ binary graph snapshot at time t can be represented as $GS^{(t)} = [e_{i,j}^{(t)}]$, if $e_{i,j}^{(t)} = 1$, it means that there is at least one data-sharing transaction between device i and device j, otherwise $e_{i,j}^{(t)} = 0$. In addition, since there is no transaction between the devices themselves, $e_{i,i}^{(t)} = 0$ is satisfied [25].

Definition 2 [Hidden State Set $(\mathcal{C})$*]:* Based on the blockchain transaction network $\mathcal{G}$, the hidden state set in $GS^{(t)}$ is $\mathcal{C}^{(t)} = \{C_1, C_2, \ldots, C_k, \ldots, C_{M_\mathcal{C}}\}$, where $C_k = (\text{num}_k, n_k, \text{list}_k)$, $1 \le k \le M_C$, which represents the dynamic community structure at time t, num_k is the shard number, n_k is the number of nodes in the shard, list_k is the list of all nodes, and $M_\mathcal{C}$ is the number of communities. As the network changes, the set of $\mathcal{C}$ in the network can be defined as the hidden state sequence $\mathcal{C}_T = \{\mathcal{C}^{(1)}, \mathcal{C}^{(2)}, \ldots, \mathcal{C}^{(t)}, \ldots, \mathcal{C}^{(T)}\}$ of the HMM.

Definition 3 [Observed State Set $(\mathcal{O})$*]:* Based on the blockchain transaction network $\mathcal{G}$, the observed state set is $\mathcal{O}^{(t)} = \{O_1, O_2, \ldots, O_k, \ldots, O_{N_O}\}$, where $O_k = (v_k, e_k, d_k)$, $1 \le k \le N_O$, which represents the node information at time t, v_k is the corresponding node, e_k is the edge set of v_k, d_k is the node degree of v_k, and N_O is the number of observed nodes. As the network changes, $\mathcal{O}$ at each moment in the network can be defined as the observed state sequence $\mathcal{O}_T = \{\mathcal{O}^{(1)}, \mathcal{O}^{(2)}, \ldots, \mathcal{O}^{(T)}\}$ of the HMM.

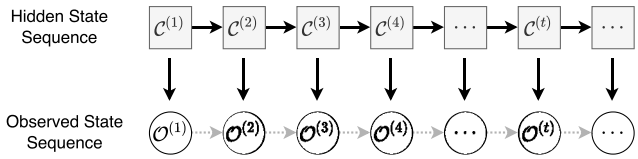


Fig. 3. Blockchain network state transition based on HMM.

As shown in Fig. 3, this article divides the dynamic blockchain transaction network into multiple network snapshots according to the time interval, which is composed of static network representations at multiple times. In order to fully capture the blockchain network changes, the time interval can be dynamically adjustetd with the network changes. Assuming that the transaction network structures at adjacent times are interrelated, the relationship between the community structures in the blockchain transaction network at different times can be described by a first-order Markov chain, since the hidden state set (dynamic community structure) in the blockchain transaction network cannot be directly observed. Therefore, by introducing the HMM, we model the dynamic community structure in the blockchain transaction network at different times as the state chain; and represent the node transaction information (including nodes, edges, and node degrees) as the observation chain. Then, the HMM is used to describe the community detection problem in the dynamic blockchain transaction network, and the optimal hidden state set in the model is decoded to obtain the community structure.

For the HMM $\lambda = (\mathbf{A}, \mathbf{B}, \Pi)$, in a snapshot $GS^{(t)}$, the state sequence can be divided into M discrete intervals, denoted as $\mathcal{C} = \{C_1, \ldots, C_M\}$, then the $M \times M$ state transition probability matrix can be defined as $\mathbf{A}_{M\times M} = [a_{i,j}]_{M\times M}$, where $a_{i,j} = \Pr[q^{(t+1)} = C_j | q^{(t)} = C_i]$, $C_i, C_j \in \mathcal{C}$, indicating the probability that the state of the node at time t is C_i and the state at time $t+1$ is C_j, where $q^{(t)}$ represents the state of the node at time t. $\mathbf{B}$ is the state observation probability matrix, $B = [b_j(k)]_{M\times N}$, which represents the probability that the state j outputs the corresponding observation value, where $b_j(k) = \Pr[O(t) = O_k | q^{(t)} = C_j]$, $C_j \in \mathcal{C}$, $O_k \in \mathcal{O}$. Finally, Π is the initial state probability distribution, $\Pi = [\pi_i]_M$, where $\pi_i = \Pr[q^{(1)} = C_i]$, $C_i \in \mathcal{C}$, and $\sum_{1\le i\le M} \pi_i = 1$. Based on the HMM λ and the observed state sequence $\mathcal{O}$, the Viterbi algorithm [26] can be used to decode the optimal hidden state set of HMM to obtain the community structure, on the basis, and then form shards $\mathcal{S}$.

B. Details of Blockchain Shard Formation Based on Hidden Markov Model

In this section, we present the construction of the DTSFG and the definition of HMM parameters. Then, we describe how to calculate the state transition probability matrix $\mathbf{A}$ and the state observation probability matrix $\mathbf{B}$ of HMM. Our proposed method, blockchain shard formation based on HMM

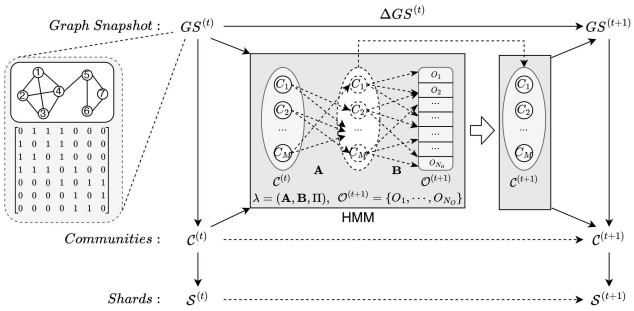


Fig. 4. Basic idea of HMM-BSF.

(HMM-BSF), aims to form shards by detecting the dynamic community structure in the blockchain transaction network using the HMM. HMM-BSF considers the network structure at the previous moment and the characteristics of the network node and community structure to solve the community detection problem. The optimal hidden state set is decoded as the community structure to form shards, as shown in Fig. 4. HMM-BSF can effectively reduce the number of CSTs, decrease the transaction communication overhead, and improve the blockchain scalability.

The steps of HMM-BSF are as follows.

1) State Initialization: Assuming a snapshot $GS^{(1)}$ at $t = 1$, the participating node set is $\{x_1, x_2, \ldots, x_N\}$, the static algorithm [27] can be used for community detection of the snapshot $GS^{(1)}$, and the initial community structure $\mathcal{C}_{\text{init}}$ can be obtained for the nodes in $GS^{(1)}$. It is assumed that the initial state probability distribution is random, that is, the probability of nodes being assigned to different communities is random, and the initial state probability distribution is $\pi = P_{\text{init}}$.

2) Node-Community Relationship and Cluster Center Calculation: At time $t(2 \le t \le T)$, the relationship matrix $NC_{i,j}(t) = [nc_{i,j}]$ between nodes and communities is constructed for the snapshot $GS^{(t)}$ at different times

$$nc_{i,j}(t) = \begin{cases} 1, & \text{if } x_i \text{ in } C_j \\ 0, & \text{otherwise} \end{cases} \tag{1}$$

then the center node $\delta_{C_i}^{\text{cen}}$ (cluster) of each community in the snapshot $GS^{(t)}$ is obtained by calculating the node degree that observed from the blockchain transaction network

$$\delta_{C_i}^{\text{cen}} = \arg\max_{x \in C_i}(\text{Degree}(x)). \tag{2}$$

3) State Transition Probability Matrix Calculation: Next, the cosine similarity between each node x_i and center node $\delta_{C_i}^{\text{cen}}$ in each community under the snapshot $GS^{(t)}$ at time $t(2 \le t \le T)$ can be calculated by using the following formula:

$$\text{Sim}\left(x_i, \delta_{C_i}^{\text{cen}}\right) = \frac{\sum_{i=1}^{N}\left(x_i \times \delta_{C_i}^{\text{cen}}\right)}{\sqrt{\sum_{i=1}^{N} x_i^2 \times \sum_{i=1}^{N}\left(\delta_{C_i}^{\text{cen}}\right)^2}} \tag{3}$$

then the state transition probability of each node $x_i \in V$ in the blockchain transaction network at the current time can be calculated to construct the matrix $\mathbf{A} = [a_{i,j}]$, where

$$a_{i,j} = \frac{\text{Sim}\left(x_i, \delta_{C_j}^{\text{cen}}\right)}{\sum_{C_k \in \mathcal{C}} \text{Sim}\left(x_i, \delta_{C_k}^{\text{cen}}\right)}. \tag{4}$$

4) State Observation Probability Matrix Calculation: To calculate the observation probability, the Membership Function $\text{Com}(x_i, C)$ [28] is introduced to judge the probability that the node x_i in the adjacency matrix R of the blockchain network belongs to a certain community C

$$\text{Com}(x_i, C) = \frac{1}{\text{Sum}(R)} \times \left(r_{i,i} + \sum_{j \in C}\left(r_{i,j} + r_{j,i}\right) - \frac{\sum_k r_{i,k} \times \sum_k r_{k,j}}{\text{Sum}(R)}\right) \tag{5}$$

then the state observation probability of each node $x_i \in V$ in the blockchain network at the current time can be calculated to construct the matrix $\mathbf{B} = [b_i(j)]$, where

$$b_i(j) = \frac{\text{Com}\left(x_j, C_i\right)}{\sum_{C_k \in \mathcal{C}} \text{Com}\left(x_j, C_k\right)}. \tag{6}$$

5) Shard Formation: Combining the parameters $\lambda = (\mathbf{A}, \mathbf{B}, \Pi)$ and observed state sequence $\mathcal{O}$ of the HMM. The Viterbi algorithm [26] is an efficient way to find the most likely sequence of states in a stochastic process, which is used to decode the optimal community structure of the blockchain network at time t for blockchain node allocation, thus forming the shards. The pseudo-code of HHM-BSF is shown in Algorithm 1.

C. Adaptive Dynamic Fine-grained Incremental Updating in Blockchain Sharding

Based on the proposed HMM-BSF algorithm, the shard structure can be incrementally determined from the known blockchain network structure, enabling dynamic updates to the shard structure. This avoids the need for recalculating the entire structure from scratch, thus simplifying the process and saving computational resources.

At the same time, with the operation of the blockchain transaction network, intrashard transactions will strengthen connections among nodes within the same shard, clarifying the shard structure, while intershard transactions can blur the shard structure. To achieve a more fine-grained approach to state changes in the network, we allow the adjustment interval to change dynamically, enabling adaptive incremental updating of sharding. Different from the traditional sharding schemes that use the *epoch* as the sharding interval, we introduce the modularity Q [29], a parameter for evaluating sharding quality, as the benchmark (i.e., $Q \le 0.6$) to dynamically adjust the number and size of shards and optimize shard structure. The definition is expressed as follows:

$$Q = \sum_{C_i \in \mathcal{C}^{(t)}}\left(\frac{e_{C_i}}{E_{\mathcal{C}^{(t)}}} - \left(\frac{\sum_{v_i \in C_i} d_{v_i}}{2 \times E_{\mathcal{C}^{(t)}}}\right)\right) \tag{7}$$

Algorithm 1 HMM-BSF

Input: $\mathcal{C}_{init}$—the initial community structure, $GS^{(t)}$, $GS^{(t+1)}$—the blockchain network snapshot at time t and $t+1$.
Output: $\mathcal{C}^{(t+1)}$—the optimal community/shard structure at time $t+1$.
1: Initialization: $len = Length(GS^{(t)})$, $NC = Zero(len, \mathcal{C})$, $\pi = P_{init} = Rand(1, Num(\mathcal{C}))$;
2: **for** $1 \le i \le len$ **do**
3: $NC(x_i, C(i)) = 1$;
4: **end for**
5: **for** $1 \le j \le Num(\mathcal{C}^{(t)})$ **do**
6: Calculate the center node of each community $\delta_{C_i}^{\text{cen}}$;
7: **end for**
8: **for** $1 \le i \le len$ **do**
9: **for** $1 \le j \le Num(\mathcal{C}^{(t)})$ **do**
10: Calculate the degree of similarity $Sim(x_i, \delta_{C_j}^{cen})$;
11: **end for**
12: Calculate the state transition probability and construct the matrix $\mathbf{A} = [a_{i,j}]$;
13: **for** $1 \le k \le Num(\mathcal{C}^{(t)})$ **do**
14: Calculate the degree of membership $Com(x_i, C_k)$;
15: **end for**
16: Calculate the state observation probability and construct the matrix $\mathbf{B} = [b_i(j)]$;
17: Calculate $Viterbi(x_i, \mathbf{A}, \mathbf{B}, \Pi)$; // the Viterbi algorithm is in [26]
18: **if** $Pr = Max(Viterbi)$ **then**
19: Allocate node x_i to C for maximizing Pr;
20:**end for**
21:**return** $\mathcal{C}^{(t+1)}$

where $E_{\mathcal{C}^{(t)}}$ represents the total edge number of $C^{(t)}$ in snapshot $GS^{(t)}$, e_{C_i} represents the number of edges in different communities, and $\sum_{v_i \in C_i} d_{v_i}$ represents the sum of node degree in the community C_i. The modularity Q is used to measure the tightness of shards and the value is usually between 0 and 1. The greater value of Q, the tighter shard structure there is. We treat all shards as identical and aim to reduce the proportion of CSTs for improving system performance. Given the limited space, we have identified two areas for further research: 1) exploring the security suppression effect between dynamic shards and 2) developing strategies for handling the dynamic process of device interaction data.

This article considers three scenarios that arise due to the characteristics of blockchain: 1) when a new node joins; 2) when a new transaction is generated; and 3) when a node leaves.

1) New Node Joining: When a new node u joins and does not generate a transaction, a new community containing only u is created, while the existing community structures remain unchanged. However, if u generates a transaction, u can be assigned to the appropriate community based on its degree of membership calculated with respect to each community.

2) New Transaction Generation: When a new edge $e = (u, v)$ is created between two existing nodes (i.e., a newly generated transaction), there are two possible scenarios: 1) e is an intrashard transaction, meaning it connects nodes within the same community or 2) e is an intershard transaction, connecting nodes from different communities C_u and C_v. If e is an intrashard transaction, it reinforces the existing shard structure, and therefore the network structure should remain unchanged. On the other hand, if e is an intershard transaction between C_u and C_v, several steps are required to adjust the nodes. First, it is necessary to calculate the similarity degree between the node and the center nodes of both C_u and C_v. Then, the degree of membership between the node to C_u and C_v should be determined, so that the nodes can be properly assigned to the corresponding communities. By doing so, the network can adapt to the new intershard transaction, while preserving the integrity of each community.

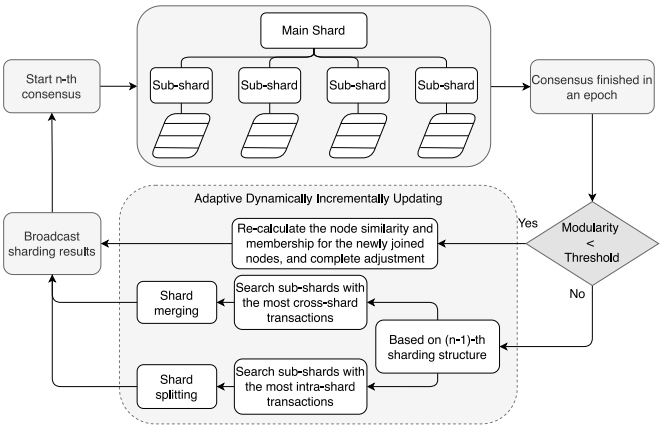


Fig. 5. Adaptive sharding fine-grained incremental updating process.

3) Node Leaving: When the node u leaves a community at time t, the edges associated with it need be removed from the graph. There are two possible scenarios depending on the degree of the departing node: 1) if a single-degree node leaves, the current community structure remains unchanged, and there is no need to modify the shard structure and 2) if a high-degree node leaves, the departure may cause the community to become disconnected or even split and merged. To adapt to these changes, the degrees of similarity and membership need to be calculated for the remaining nodes in the community. By doing so, the community splitting or merging can be completed, and the shard structure can be adjusted accordingly.

The blockchain shards can be dynamically adjusted in a fine-grained manner by calculating the modularity of the t time snapshot and ensuring that the value falls within a preset modularity interval. This involves making incremental updates to the shard structure based on the previous configuration, including shard merging, shard splitting, and node adjustments. This fine-grained approach results in lower compared to a complete sharding reconfiguration. Fig. 5 illustrates the adaptive sharding fine-grained incremental updating process.

BFT Satisfactory: We prove that HMMDShard satisfies the consistency, validity, and liveness conditions of BFT even under compromised termination conditions, for all valid transactions. Specifically, we demonstrate the follow properties.

1) *Consistency:* If an honest node validates a transaction, then another honest node will also validate it.

2) *Validity:* If a valid transaction can be validated by an honest node, then other honest nodes also can validate it.
3) *Liveness:* If an honest node proposes a transaction, which can be validated in a finite time.

Proof:

1) *Proof of Consistency:* If a proposed transaction tx_i is successfully validated by an honest node, then tx_i must be valid. If other nodes are suspicious of the transaction, they can collect the proof and run the validation function themselves. Since the validation function is deterministic, all honest nodes will reach the same result, ensuring consistency.
2) *Proof of Validity:* The proof of a valid transaction can be collected and validated by other honest nodes, ensuring that all honest nodes will agree on the validity of the transaction.
3) *Proof of Liveness:* When a transaction is proposed by an honest node, it implies that the transaction adheres to the criteria of a valid transaction, including valid sources, value equality, and the absence of double spending. To validate such transactions in HMMDShard, we use main shards that employ the PBFT consensus algorithm, which guarantees Byzantine fault tolerance. By using this approach, we can eventually confirm that the transaction meets all the necessary requirements of a valid transaction, thereby validating it.

Algorithm Complexity: The complexity of the HMM-BSF is determined by several factors. Assuming that n is the number of nodes in the blockchain network, m it the number of edges, and M is the initial number of shards, and when calculating the center node in the shard, each node in the set needs to be processed, therefore the complexity is $O(n)$; when calculating the state transition probability matrix and the state observation probability matrix, it is necessary to combine shards, so the calculation complexity is $O(n^2 \times M)$; the Viterbi algorithm applies recursive calling, and its algorithm complexity is $O(n^2 \times \log(n))$; therefore, the total complexity of HMM-BSF is $O(n) + O(n^2 \times M) + O(n^2 \times \log(n))$. ■

V. Performance Evaluation

This section presents the results of our numerical experiments to evaluate the proposed blockchain dynamic sharding scheme HMMDShard in the context of IoT collaborative sensing.

A. Basic Settings

1) Experimental Prototype: To evaluate HMMDShard, we conducted a series of experiments using an experimental prototype implemented in Python 3.8 and Golang 1.17.5 combined with Dedis [30]. We deployed the prototype on several Raspberry Pi 4B, which acted as edge servers, as shown in Fig. 7, as well as three AliCloud servers with 100 virtual machines serving as peers in shards, which supported up to 32 shards. Each virtual machine was equipped with an Intel Xeon i5-10500 CPU and 1-GB RAM, providing a realistic environment for performance evaluation. We imposed limitations on the bandwidth of all network connections between nodes to 5 Mb/s and introduced a 100 ms delay to the communication link to simulate the real IoT collaborative scenario. Then, we ran the prototype system by replaying collected historical transactions from real-world scenarios. Each experiment was repeated 100 times to ensure statistical reliability, and the average value was calculated as the final result for each metric of interest. The used experimental parameters are shown in Table II.

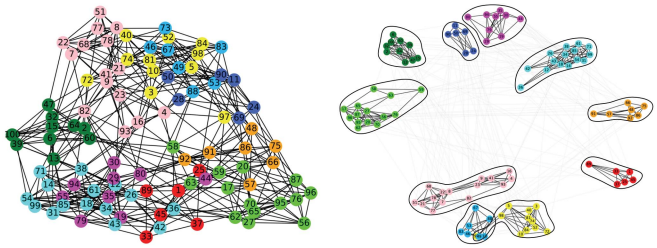


Fig. 6. HMMDShard divides the initial blockchain network into multiple shards in a bottom-top manner.

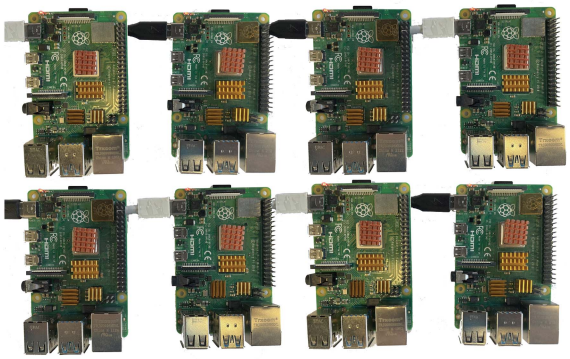


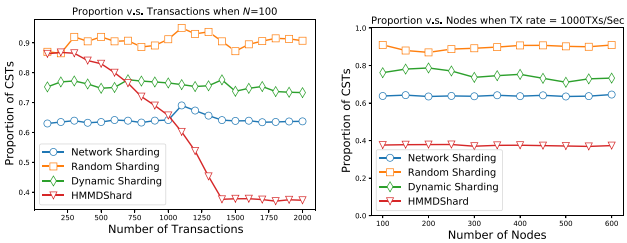
Fig. 7. Deploying the prototype on several Raspberry Pi 4Bs.

TABLE II
Parameter Settings

No	Parameters and Description	Parameter Values
1	All participating nodes (N)	100-600
2	The proportion of Byzantine faults (f/n)	5%, 10%, 20%, 25%
3	Transaction rate (*TXs/Sec*)	1000-3000
4	Maximum number of shards (M)	32

2) Data Set: We used a real Ethereum transaction data set containing 5534 blocks and 1.13 million historical transactions recorded from 7 August 2015 to 7 December 2015. To identify the participating nodes, we selected the transaction addresses of nodes as the registration ID of participating nodes and then combine with NetworkX [31] to generate DTSFGs for constructing the HMM. A specific sharding case is shown in Fig. 6. We divided the initial blockchain network and participating nodes into multiple shards in a bottom-top manner, which minimizes the proportion of CSTs and improves system performance.

3) Benchmark: To evaluate the performance of our proposed HMMDShard, we conducted a comprehensive comparison with several existing sharding strategies, including network sharding, random sharding, and dynamic sharding.



(a) (b)

Fig. 8. Proportion of CSTs with different number of transactions and nodes. (a) $N = 100$. (b) *TX rate* $= 1000$ TXs/s.

For the purpose of comparison, we also evaluated two state-of-the-art static sharding schemes, OmniLedger [3] and TEEShard [5], as well as DBShard [8], which supports dynamic adjustment of shard size and number. Among the existing sharding strategies, Elastico [2], RapidChain [4], and Monoxide [6] exhibit limitations such as the inability to handle CSTs or the lack of available source codes on Github. TEEShard is a well-known network sharding solution that leverages trusted execution environment. Through our evaluation, we aim to demonstrate the effectiveness of HMMDShard in comparison to other existing and state-of-the-art sharding strategies.

4) Metrics: We investigated the effect of several critical system parameters on various system metrics using different state-of-the-art sharding schemes. We specifically focused on evaluating the proportion of CSTs, shard modularity, transaction throughput, and latency as the primary metrics.

B. Experimental Results and Analysis

1) Proportion of CSTs and Modularity: Fig. 8(a) presents the proportion of CSTs for four different sharding strategies. As the number of system transactions increases, the proportion of CSTs remains high under network sharding and random sharding. However, even after modifying the size and number of shards, DBShard still has a high percentage of CSTs due to the random assignment of nodes. In contrast, the proposed HMMDShard considers the dynamic assemblage characteristics, and as the number of CSTs increases, the system incrementally updates the shard structure based on the calculated modularity, thereby reducing the proportion of CSTs to a certain low degree. In addition, Fig. 8(b) shows the different proportion of CSTs in four types of node-sharding networks at a fixed blockchain network transaction rate of 1000 TXs/s. HMMDShard has a lower proportion of CSTs (less than 30%) than the other three schemes as the number of nodes changes. Furthermore, Table IV presents the runtime of sharding and the corresponding optimal number of shards for HMMDShard under different number of nodes.

Furthermore, Fig. 9 illustrates the impact of the number of nodes on the proportion of CSTs and sharding modularity. As the number of nodes increases, the proposed dynamic sharding scheme leverages the HMM-BSF algorithm to determine the optimal shard structure, leading to a low proportion of CSTs

TABLE III
COMPARISON WITH DIFFERENT SHARDING SOLUTIONS

Solutions	Computing Complexity	Time Cost	Number of Shards	Sharding Way	Number of Nodes
OmniLedger	$O(n)$	12.5s	Fixed	Random	600
TEEShard	$O(n)$	10.4s	Fixed	Network	600
DBShard	$O(n)$	9.4s	Dynamic	Random	600
HMMDShard	$O(c)$	1.2s	Dynamic	HMM	600

TABLE IV
RUNTIME AND NUMBER OF SHARDING WITH DIFFERENT NUMBER OF NODES

No	Num of Nodes	Runtime(s) of Sharding	Num of Shards
1	100	1.62s	11
2	150	4.44s	12
3	200	9.03s	13
4	250	15.09s	14
5	300	20.98s	15
6	350	26.47s	16
7	400	34.47s	17
8	450	43.05s	18
9	500	53.18s	17
10	550	65.48s	18
11	600	77.79s	17

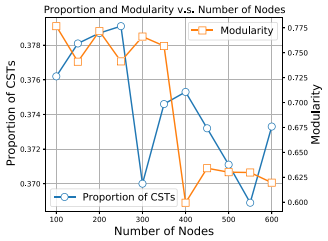


Fig. 9. Proportion of CSTs and sharding modularity.

and improved transaction processing performance. Meanwhile, a higher modularity value (moderating around 0.625) indicates that the sharding quality of HMMDShard is higher. Additionally, Table III presents a comparison of the computing power consumption and time cost of HMMDShard with OmniLedger, TEEShard, and DBShard. While the other solutions reconfigure the sharding structure with all participating nodes after each consensus epoch and have a computing complexity of $O(n)$. HMMDShard fine-tunes the sharding structure based on the adjacent structure at the previous moment, resulting in much less computing complexity compared to the other three solutions.

2) System Throughput and TX Confirmation Latency: We conducted an evaluation of the performance of HMMDShard in different Byzantine environments by analyzing system throughput and TX confirmation latency, and compare it with OmniLedger, TEEShard, and DBShard. Fig. 10 shows that, as we varied the proportion of malicious adversaries (5%, 10%, 20%, and 25%), HMMDShard outperforms OmniLedger, TEEShard, and DBShard in terms of transaction throughput. This can be attributed to the dynamic incremental sharding scheme proposed in this article, which effectively reduces the proportion of CSTs and thereby avoids the extra overhead caused by a large number of CSTs, indirectly improving

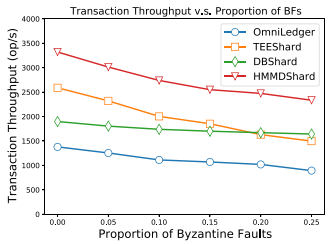


Fig. 10. System throughput of different sharding systems.

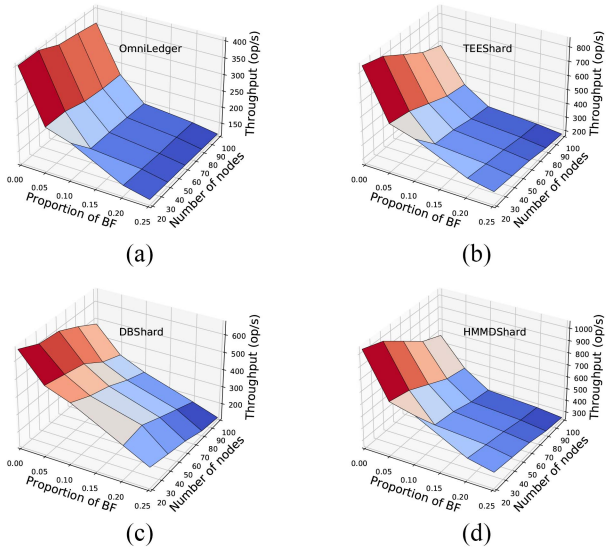


Fig. 11. Throughput of the single shard with different sharding schemes. (a) OmniLedger. (b) TEEShard. (c) DBShard. (d) HMMDShard.

the system throughput. Moreover, we analyzed the impact of Byzantine Fault Tolerance failed nodes on transaction throughput by measuring the throughput of a single shard under different sharding schemes that all use the traditional PBFT consensus mechanism to ensure consistency. Fig. 11 illustrates that HMMDShard achieves the most substantial transaction throughput across different proportion of BFT nodes.

Additional, we conducted experiments to evaluate the average transaction confirmation latency under different sharding schemes by varying the number of nodes and transaction rate. The results are presented in Fig. 12. As shown in Fig. 12(a), when the transaction rate is fixed to 1000, the average transaction confirmation latency increases as the number of nodes N from 20 to 100. This is because a large number of nodes result in more CSTs, which incur additional processing time overhead. However, the proposed HMMDShard achieves a low average transaction confirmation latency of only 2.004 s. In Fig. 12(b), we varied the transaction rate while fixing the number of nodes to $N = 100$. HMMDShard achieves a significantly lower latency compared to the other three baselines, demonstrating its ability to guarantee low latency even under higher transaction rates. Furthermore, Fig. 12(c) and (d) demonstrate the impact of Byzantine nodes on latency with 20% Byzantine nodes. Although the introduction of Byzantine nodes increases the average transaction confirmation latency

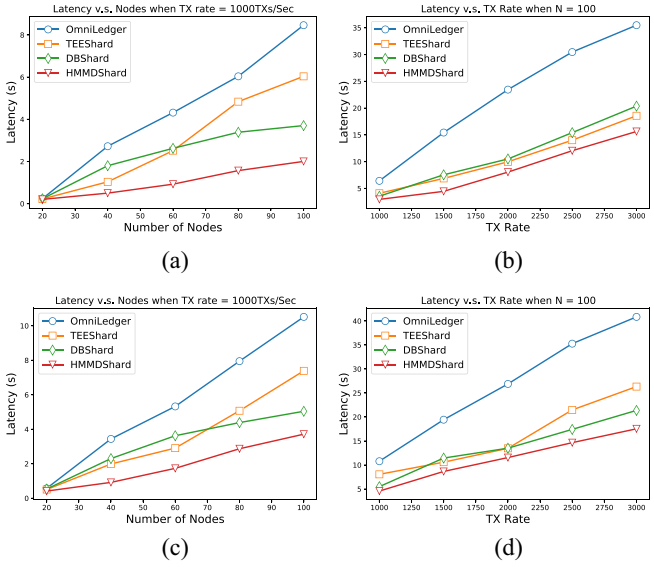


Fig. 12. Proportion of CSTs with different number of transactions and nodes. (a) *TX rate* = 1000 TXs/s without Byzantine nodes. (b) $N = 100$ without Byzantine nodes. (c) *TX rate* = 1000 TXs/s with 20% Byzantine nodes. (d) $N = 100$ with 20% Byzantine nodes.

under all methods, HMMDShard still maintains the lowest latency with PBFT consensus.

VI. Conclusion

This article proposes HMMDShard, a blockchain dynamic fine-grained incremental sharding scheme in CIoT. In HMMDShard, the HMM-BSF is innovatively introduced to achieve adaptive dynamic incremental updating of blockchain shards, effectively reducing the communication and processing overheads associated with CSTs. Our experimental results demonstrate that HMMDShard outperforms other state-of-the-art sharding schemes in terms of system throughput and transaction confirmation latency. In future work, we plan to explore the development of a lightweight sharding consensus mechanism and investigate an IoT device identity management approach to further enhance the security and performance of sharded blockchain in CIoT.

References

[1] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," *Decentralized Bus. Rev.*, Mar. 2009, Art. no. 21260.
[2] L. Luu, V. Narayanan, C. Zheng, K. Baweja, S. Gilbert, and P. Saxena, "A secure sharding protocol for open blockchains," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, 2016, pp. 17–30.
[3] 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 Symp. Secur. Privacy (SP)*, 2018, pp. 583–598.
[4] M. Zamani, M. Movahedi, and M. Raykova, "RapidChain: Scaling blockchain via full sharding," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, 2018, pp. 931–948.
[5] H. Dang, T. T. A. Dinh, D. Loghin, E.-C. Chang, Q. Lin, and B. C. Ooi, "Towards scaling blockchain systems via sharding," in *Proc. Int. Conf. Manage. Data*, 2019, pp. 123–140.
[6] J. Wang and H. Wang, "Monoxide: Scale out blockchain with asynchronous consensus zones," in *Proc. 16th USENIX Symp. Netw. Syst. Des. Implement. (NSDI)*, 2019, pp. 95–112.
[7] V. Buterin, *Ethereum: Platform Review: Opportunities Challenges Private Consortium Blockchains*, Ethereum, Zug, Switzerland, 2016.

[8] D. Tennakoon and V. Gramoli, "Dynamic blockchain sharding," in *Proc. 5th Int. Symp. Found. Appl. Blockchain (FAB)*, 2022, pp. 6:1–6:17.
[9] Z. Yang, R. Yang, F. R. Yu, M. Li, Y. Zhang, and Y. Teng, "Sharded blockchain for collaborative computing in the Internet of Things: Combined of dynamic clustering and deep reinforcement learning approach," *IEEE Internet Things J.*, vol. 9, no. 17, pp. 16494–16509, Sep. 2022.
[10] C. Huang et al., "RepChain: A reputation-based secure, fast, and high incentive blockchain system via sharding," *IEEE Internet Things J.*, vol. 8, no. 6, pp. 4291–4304, Mar. 2021.
[11] Z. Hong, S. Guo, P. Li, and W. Chen, "Pyramid: A layered sharding blockchain system," in *Proc. IEEE INFOCOM Conf. Comput. Commun.*, 2021, pp. 1–10.
[12] Z. Avarikioti and D. Karakostas, "Harmony technical whitepaper," Harmony Team, Mountain View, CA, USA, White Paper, 2022. [Online]. Available: https://harmony.one/whitepaper.pdf
[13] M. Oliveira and J. Gama, "An overview of social network analysis," *Wiley Interdiscipl. Rev. Data Min. Knowl. Discov.*, vol. 2, no. 2, pp. 99–115, 2012.
[14] R. Huo et al., "A comprehensive survey on blockchain in industrial Internet of Things: Motivations, research progresses, and future challenges," *IEEE Commun. Surveys Tuts.*, vol. 24, no. 1, pp. 88–122, 1st Quart., 2022.
[15] U. Majeed, L. U. Khan, I. Yaqoob, S. M. A. Kazmi, K. Salah, and C. S. Hong, "Blockchain for IoT-based smart cities: Recent advances, requirements, and future challenges," *J. Netw. Comput. Appl.*, vol. 181, May 2021, Art. no. 103007.
[16] A. M. Seid, H. N. Abishu, Y. H. Yacob, T. A. Ayall, A. Erbad, and M. Guizan, "Blockchain-based resource trading in multi-UAV-assisted industrial IoT networks: A multi-agent DRL approach," *IEEE Trans. Netw. Service Manag.*, vol. 20, no. 1, pp. 166–181, Mar. 2023.
[17] B. Zaabar, O. Cheikhrouhou, F. Jamil, M. Ammi, and M. Abid, "HealthBlock: A secure blockchain-based healthcare data management system," *Comput. Netw.*, vol. 200, Dec. 2021, Art. no. 108500.
[18] E. Bandara, D. Tosh, P. Foytik, S. Shetty, N. Ranasinghe, and K. De Zoysa, "Tikiri—Towards a lightweight blockchain for IoT," *Future Gener. Comput. Syst.*, vol. 119, pp. 154–165, Jun. 2021.
[19] 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.
[20] M. Caprolu, M. Pontecorvi, M. Signorini, C. Segarra, and R. Di Pietro, "Analysis and patterns of unknown transactions in Bitcoin," in *Proc. IEEE Int. Conf. Blockchain (Blockchain)*, 2021, pp. 170–179.
[21] T. Chen et al., "Understanding Ethereum via graph analysis," *ACM Trans. Internet Technol.*, vol. 20, no. 2, pp. 1–32, 2020.
[22] 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.
[23] M. Castro and B. Liskov, "Practical Byzantine fault tolerance," in *Proc. OsDI*, vol. 99, 1999, pp. 173–186.
[24] C.-C. Lin, W.-Y. Liu, and D.-J. Deng, "A genetic algorithm approach for detecting hierarchical and overlapping community structure in dynamic social networks," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, 2013, pp. 4469–4474.
[25] B. Betancourt, A. Rodríguez, and N. Boyd, "Modelling and prediction of financial trading networks: An application to the New York mercantile exchange natural gas futures market," *J. Roy. Stat. Soc. Ser. C, Appl. Stat.*, vol. 69, no. 1, pp. 195–218, 2020.
[26] J. Cui, Y. Zhou, and L. Chen, "Implementation and optimization of embedded speech recognition system based on DHMM," *J. Univ. Electron. Sci. Technol. China*, vol. 6, pp. 930–934, Nov. 2013.
[27] D. M. Walker and A. Tordesillas, "Examining overlapping community structures within grain property networks," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, 2014, pp. 1275–1278.
[28] T. Y. Berger-Wolf and J. Saia, "A framework for analysis of dynamic social networks," in *Proc. 12th ACM SIGKDD Int. Conf. Knowl. Discov. Data Min.*, 2006, pp. 523–528.
[29] M. E. J. Newman, "Modularity and community structure in networks," *Proc. Nat. Acad. Sci.*, vol. 103, no. 23, pp. 8577–8582, 2006.
[30] M. Raynal, *RandShare: Small-Scale Unbiasable Randomness Protocol*, DEDIS EPFL, Lausanne, Switzerland, 2018.
[31] A. Hagberg and D. Conway. "NetworkX: Network analysis with python." 2020. [Online]. Available: https://networkx.github.io

Jinwen Xi received the M.E. degree from Nanjing University of Information Science and Technology, Nanjing, China, in 2018, and the Ph.D. degree in information security from Beijing University of Posts and Telecommunications, Beijing, China, in 2022.

He is currently working as an Assistant Research Associate with Zhongguancun Laboratory, Beijing. He focuses on the security and privacy issues for Industrial Internet of Things. His current research interests are Internet of Things, blockchain, digital twin, as well as privacy computation.

Guosheng Xu (Member, IEEE) received the Ph.D. degree in information security from Beijing University of Posts and Telecommunications, Beijing, China, in 2008.

He is a Lecturer with the School of Cyberspace Security, Beijing University of Posts and Telecommunications. His research interest is in the areas of deep learning and modern cryptography.

Shihong Zou (Member, IEEE) received the Bachelor of Engineering degree in computer engineering from Nanjing University of Posts and Telecommunications, Nanjing, China, in 1999, and the Ph.D. degree in communication and information systems from Beijng University of Posts and Telecommunications (BUPT), Beijng, China, in 2004.

He is an Associate Professor with the School of CyberSpace Security, BUPT. He has published over 40 papers and applied over 20 patents. His research interests include mobile security, IoT security, blockchain, and wireless networking.

Yueming Lu received the B.S. and M.S. degrees in computer science from Xi'an University of Architecture and Technology, Xi'an, China, in 1994 and 1997, respectively, and the Ph.D. degree in computer architecture from Xi'an Jiaotong University, Xi'an, in 2000.

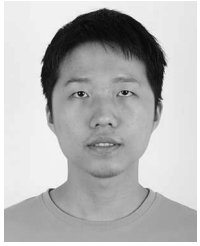
From 2000 to 2003, he was a Researcher with the Optical Network Group Pacific, Lucent, Beijing, China, where he was involved in 10-Gb/s optical transportation networks. He is currently a Professor with Beijing University of Posts and Telecommunications, Beijing, and an Academic Committee Member with the Key Laboratory of Trustworthy Distributed Computing and Service, Ministry of Education.

Guoqiang Li is a Doctor and Assistant Research Associate with the Institutes of Science and Development, Chinese Academy of Sciences, Beijing, China. His research focuses on the impacts and patterns of digital transformation. His current research interests are semantic information processing, digital twin, AI modeling, as well as privacy computation.

Jiuyun Xu (Member, IEEE) received the Ph.D. degree in computer science from Beijing University of Posts and Telecommunications is Beijing, China, in 2004.

He is currently a Professor with China University of Petroleum (Eastern China), Qingdao, China. He has published in excess of 40 international conferences and journal articles. His research interests include service computing and Internet of Things.

Prof. Xu has served on the technical program committees for numerous conferences, including ICWS and SCC. He is a Reviewer for some prestigious journals, including the IEEE TRANSACTIONS ON SERVICES COMPUTING and *International Journal of Interdisciplinary and Multidisciplinary Studies*. He is a member of ACM and a Senior Member of CCF.



Ruisheng Wang received the B.S. degree from the Huazhong University of Science and Technology, Wuhan, China, in 2021. He is currently pursuing the M.S. degree in information security with Beijing University of Posts and Telecommunications, Beijing, China.

He focuses on the security and privacy issues for Industrial Internet of Things. His current research interests are collaborative Internet of Things, blockchain, and consensus protocols.