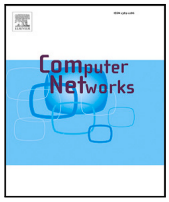
Computer Networks 244 (2024) 110343

Contents lists available at ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet



TBDD: A new trust-based, DRL-driven framework for blockchain sharding in IoT

Zixu Zhang [a], Guangsheng Yu [b], Caijun Sun [c,*], Xu Wang [a], Ying Wang [d], Ming Zhang [e], Wei Ni [b], Ren Ping Liu [a], Andrew Reeves [f], Nektarios Georgalas [f]

[a] Global Big Data Technologies Centre, University of Technology, Sydney, 2007, Australia
[b] Data61, CSIRO, Sydney, 2015, Australia
[c] Zhejiang Lab, Hangzhou, China
[d] School of Electronics and Information, Hangzhou Dianzi University, Hangzhou 310018, China
[e] School of Cyber Engineering, Xidian University, Xi'an, 710071, China
[f] Applied Research, British Telecom, Martlesham, UK

ARTICLE INFO

Keywords:
Blockchain
Sharding
Collusion attacks
Trustworthiness
Deep reinforcement learning
Internet-of-Things

ABSTRACT

Integrating sharded blockchain with IoT presents a solution for trust issues and optimized data flow. Sharding boosts blockchain scalability by dividing its nodes into parallel shards, yet it is vulnerable to the 1% attacks where dishonest nodes target a shard to corrupt the entire blockchain. Balancing security with scalability is pivotal for such systems. Deep Reinforcement Learning (DRL) adeptly handles dynamic, complex systems and multi-dimensional optimization. This paper introduces a Trust-based and DRL-driven (TBDD) framework, crafted to counter collusion attack risks and dynamically adjust node allocation, enhancing throughput while maintaining network security. With a comprehensive trust evaluation mechanism, TBDD discerns node types and performs targeted resharding against potential threats. The TBDD framework maximizes the tolerance for dishonest nodes, optimizes node movement frequency, ensures even node distribution in shards, and balances sharding risks. Extensive evaluations validate TBDD's superiority over conventional random-, community-, and trust-based sharding methods in shard risk equilibrium and reducing cross-shard transactions.

1. Introduction

By interconnecting devices, vehicles, and appliances, the Internet of Things (IoT) has transformed sectors ranging from smart cities [1], which optimize traffic and energy use, to autonomous vehicles [2] that aim for safer roads, industrial networks [3] that redefine production processes, e-health solutions [4] that prioritize patient care, and smart homes [5] that enrich daily living. The massive and sensitive nature of the data generated demands strong security and integrity, leading to the integration of IoT and blockchain [6–8]. The convergence of decentralized, immutable, and traceable characteristics provides a fundamental structure for secure data exchanges, as outlined in Mathur's 2023 survey [9]. Sharding, a key strategy in blockchain, addresses vast data needs and ensures scalability [10]. By dividing the network into smaller segments, sharding enables simultaneous transaction processing, lightening the load on nodes and increasing transaction throughput.

Sharded blockchains are susceptible to the 1% attack, a type of collusion attack where even a mere 1% of dishonest nodes can launch a successful attack on a shard if they coincidentally share the same shard and surpass the security threshold of the shard. To address the 1% attack, random-based sharding schemes have been proposed [11–14] by regularly assigning nodes to shards in a random way. However, the schemes require frequent node reassignments across shards, increasing sharding costs and subsequently decreasing throughput. Conversely, community-based sharding technologies [15,16] prioritize throughput by clustering frequently communicating blockchain nodes into the same shard, but they are vulnerable to adaptive collusion attacks, wherein dishonest nodes mimic community-like communication patterns. To address the adaptive collusion attack, trust mechanisms [17] have been designed to identify dishonest nodes and re-shard the blockchain. Nevertheless, existing trust-based sharding schemes [18] often struggle to find the optimal trade-off between security and performance. Deep

* Corresponding author.
E-mail addresses: zixu.zhang@student.uts.edu.au (Z. Zhang), saber.yu@data61.csiro.au (G. Yu), sun.cj@zhejianglab.com (C. Sun), xu.wang-1@uts.edu.au (X. Wang), 90023@hdu.edu.cn (Y. Wang), m1ngzhang@stu.xidian.edu.cn (M. Zhang), wei.ni@data61.csiro.au (W. Ni), renping.liu@uts.edu.au (R.P. Liu), andrew.reeves@bt.com (A. Reeves), nektarios.georgalas@bt.com (N. Georgalas).

https://doi.org/10.1016/j.comnet.2024.110343
Received 8 November 2023; Received in revised form 8 February 2024; Accepted 15 March 2024
Available online 18 March 2024

Reinforcement Learning (DRL), as an advanced machine learning technique that enables an agent to learn optimal decision-making strategies by interacting with its environment and receiving feedback in the form of rewards, has become instrumental in the confluence of IoT and blockchain sharding due to its proficiency in handling multiple sharding variables [19–25]. DRL technologies, such as Deep Q-Network (DQN) [26] and Proximal Policy Optimization (PPO) [27], can provide real-time adaptability and support dynamic shard adjustments that enhance resource use and overall system efficiency. However, many DRL-focused studies underestimate the challenges of adaptive collusion attacks, where dishonest nodes might disguise their identity or increase cross-shard transactions to hide their intent.

This paper proposes a new blockchain sharding framework, entitled TBDD, that incorporates a new trust table and DRL technology. The designed trust table advances the sharding process by incorporating a multi-dimensional approach to gather feedback within the voting mechanism. This multi-faceted feedback includes direct feedback, indirect feedback, and historical behaviors during the voting process, all contributing to a robust defense mechanism against the adaptive collusion attack. Concurrently, the DRL-driven sharding framework is designed to dynamically allocate shards in real-time, which streamlines node synchronization and minimizes cross-shard transactions (CSTs). TBDD also ensures a balanced distribution of dishonest nodes, aiming to decrease the necessity for frequent resharding.

Compared with existing sharding techniques, the proposed TBDD framework offers a new way to shard blockchain that achieves an optimal balance between security and throughput. In contrast to random-based sharding [11–14], the TBDD methods exhibit a lower node movement ratio and less CST, reducing overhead and enhancing overall system stability. Compared with community-based sharding [15,16], the TBDD framework is specifically designed to tackle collusion attacks by ensuring a more evenly distributed node allocation across shards, thereby mitigating associated risks. Moreover, unlike trust-based sharding methods [17,18], the proposed TBDD sharding approach learns the optimal trade-off between security and throughput through a novel trust table design and a DRL-driven sharding framework.

The main contributions of this paper can be summarized as follows,

- A new blockchain node trust scheme is designed, which translates the blockchain voting process into multidimensional feedback to evaluate the risk of blockchain nodes and detect possible collusion.
- A novel DRL-driven blockchain sharding framework, entitled TBDD, is proposed to enhance the security and efficiency of blockchain sharding systems. A comprehensive reward function is formulated, capturing the risk of shards, sharding balance, blockchain efficiency, and sharding cost via the trust table.
- The TBDD framework is implemented with two popular DRL algorithms, i.e., DQN and PPO. Extensive experiments are conducted to validate the security and efficiency of the new sharding framework.

Comprehensive experiment results reveal that the TBDD system is particularly effective in minimizing CST and ensuring a balanced distribution of trust across shards. When the number of dishonest nodes remains within the security threshold of non-sharded blockchains, the TBDD framework offers about a 10% throughput advantage over random-based sharding and a 13% improvement compared to the trust-based approach. Additionally, the TBDD framework maintains the minimum corrupted shards across the sharding schemes. Evaluation of the time overhead underscores the adaptability and feasible latency of TBDD.

The organization of the remaining paper is as follows: Section 2 reviews relevant literature. Section 3 introduces the proposed system model and problem definition. Section 4 comprehensively presents the proposed TBDD system. Experiments and assessments are conducted in Section 5. The paper is concluded in Section 6.

2. Related work

In this section, the existing related work is presented in the field of blockchain-enabled IoT, followed by a review of blockchain sharding techniques such as random-based, community-based, and trust-based sharding. Additionally, the application and impact of DRL technology in sharded blockchains are compared and analyzed.

2.1. Blockchain and IoT

With the rapid development of blockchain technology, blockchain-enabled IoT has become more secure and reliable. Existing blockchain-enabled IoT systems primarily focus on framework design and consensus protocol design. For instance, Kang et al. [2] proposed a solution to address security and privacy challenges in Vehicular Edge Computing and Networks (VECONs) based on consortium blockchain and smart contracts. In the medical scenario, Gadekallu et al. [4] presented a blockchain-based solution to enhance the security of datasets generated from IoT devices in e-health applications. The proposed solution utilizes a blockchain platform and a private cloud to secure and verify the datasets, reducing tampering risks and ensuring reliable results. Xu et al. proposed a blockchain-enabled system for data provenance with 5G and LoRa network [28,29].

2.2. Blockchain sharding

2.2.1. Random-based sharding

Within the domain of blockchain sharding research, various techniques centered on random sharding have been developed to improve system scalability and fault tolerance. Elastico [11] stands out as an early protocol introducing sharding in a permissionless environment, scaling the network dynamically as the node count increases. Building on these foundations, OmniLedger [12] advances these concepts, guaranteeing linear scalability through an efficient sharding mechanism adept at handling cross-shard transactions seamlessly. RapidChain [13] built upon these concepts with a full-sharding solution that streamlines the partitioning of blockchain states and the processing of transactions. Monoxide [14] introduced asynchronous consensus areas to boost transaction efficiency, though its static sharding strategy may fall short under dynamically changing network conditions. These innovations have significantly contributed to system robustness by improving randomness and fault tolerance, yet challenges such as frequent node reassignments remain, leading to non-trivial system overheads.

2.2.2. Community-based sharding

Community-based sharding is a partitioning method in blockchain networks that divides the system into smaller subsets or shards based on the interactions between nodes. Nodes and transactions are depicted as a graph and are partitioned into smaller subgraphs or shards, each comprising a subset of nodes and transactions. Fynn et al. [30] investigate the Ethereum blockchain's scalability through sharding implementation. They model the Ethereum blockchain as a graph and analyze different algorithms for graph partitioning, such as the Kernighan–Lin algorithm [31] and METIS [32]. Zhang et al. [15,33] proposed community detection-based sharding to enhance scalability, grouping nodes with frequent transactions into the same shard to reduce cross-shard transactions (CSTs). The TxAllo algorithm [16] further optimizes this concept by dynamically allocating transactions across shards, aiming to improve system throughput and minimize CSTs. However, both these solutions demand significant computational resources and incur high communication costs.

2.2.3. Trust-based sharding

Trust-based blockchain sharding divides the network into smaller shards based on node trust. Nodes are evaluated by reputation and behavior, grouping trusted nodes to enhance security and performance and evenly distributing dishonest nodes. Yun et al. [17] presented a Trust Based Shard Distribution (TBSD) scheme to enhance system security and prevent collusion, specifically aiming to address the 1% attack. However, their approach does not consider shard load balance and trust difference, potentially leading to system delays. Huang et al. [34] proposed RepChain, a reputation-based blockchain system with sharding for high throughput, security, and node cooperation. It utilizes a double-chain architecture: transaction chain and reputation chain. However, the double-chain architecture chain increases system complexity and resource needs, resulting in additional overhead. Zhang et al. [18] proposed a new blockchain sharding model to enhance security and efficiency. Their approach considers shard trust, latency, and node count differences, reducing the risk of blockchain failure. However, the effectiveness of their proposed model heavily depends on obtaining accurate information, which can be challenging to obtain in dynamic blockchain sharding scenarios.

Existing works in IoT and blockchain have primarily focused on enhancing scalability and security through various sharding techniques. These include random-based [11–14], community-based [15,16], and trust-based methods [17,18]. Despite their advancements, these methods still face significant challenges, particularly in handling advanced threats such as adaptive collusion attacks and their limitations in dynamically adjusting to the evolving demands of IoT networks. The proposed framework, TBDD, fills this research gap by leveraging DRL to optimize the sharding process in real-time, offering a robust solution against strategic collusion and ensuring a balanced node distribution. TBDD not only enhances the security and scalability of blockchain systems within IoT contexts but also improves throughput and efficiency, making it highly adaptable to the changing conditions of real-world IoT environments.

2.3. Deep reinforcement learning-based sharding

DRL-based sharding solutions have been making strides in the realm of IoT. Liu et al. [22] were pioneers, leveraging DRL in a blockchain framework tailored for Industrial IoT (IIoT). While they dynamically adjusted critical parameters, they missed addressing dishonest attacks. Similarly, another work by Liu et al. [23] harnessed Ethereum and DRL for IIoT data security but sidestepped throughput scalability concerns. On the other hand, Qiu et al. presented service-oriented blockchain solutions, using DRL for refined block production and bandwidth allocation [24,25]. However, these lacked centralized security measures, making them susceptible to dishonest threats.

Further innovations came from Yun et al. [20], who introduced a DQN-optimized framework for sharded blockchains. Despite its advancements, the approach overlooked intricate attack strategies in DRL-based sharding and was confined by its central Q-learning reliance. Meanwhile, Yang et al. [21] incorporated K-means clustering in a sharded blockchain, utilizing DRL for optimization. Yet, their methodology was marred by the computational intensity and intricacy of DRL. Most existing articles [22–25] lack effective analysis of dishonest attacks in IoT using DRL-based blockchain, neglecting the complexities and depth of practical security threats. In contrast, the proposed model analyzes the 1% attack problem in the blockchain sharding model and considers strategic collusion attacks involved in existing related studies. Proactive defense mechanisms and risk mitigation strategies are developed by understanding attackers' motivation. A trustworthy DRL-based blockchain sharding system is designed in this paper under IoT scenarios.

3. System model: Mining in permissioned sharded blockchain networks

The architecture of the proposed sharded blockchain designed to support an IoT network is presented in this section, followed by roles involved in the system, providing a workflow overview and elucidating the system assumptions.

3.1. System overview

System Model. Fig. 1 illustrates the proposed sharding framework used in IoT deployments. This framework aims to enhance the scalability and efficiency of blockchain applications within the vast and interconnected realm of IoT devices. Central to the architecture is the sharding mechanism that partitions the broader network into smaller, more manageable shards. Each shard handles a subset of the overall transactions, allowing for simultaneous processing and increasing throughput. Additionally, the design ensures a balanced distribution of nodes across shards to mitigate adaptive collusion attacks.

Architecture. Fig. 2 illustrates TBDD's architecture. Within each shard managed by an edge server, any node can propose a block as a leader. Each node maintains a local trust table with trust scores assigned to other network nodes. The introduced TBDD COMMITTEE (TC) is composed of members democratically selected by the network's users. Drawing inspiration from Elastico [11], the TC ensures decentralized and reliable oversight. This design prevents single points of failure and minimizes risks from a centralized authority. The TC functions as a decentralized, trusted overseer, managing the node list for the entire network and assigning nodes to different shards. By aggregating nodes' local trust tables, it derives a global trust metric per node, thus ensuring the integrity of trust and node distribution. A key component of TBDD is its resharding mechanism, which routinely reassigns nodes among shards. Such adaptability is crucial, enabling the system to remain scalable and secure, and to handle increasing transaction volumes without sacrificing security.

Roles. In the TBDD framework, participants in the network, referred to as nodes or servers, are categorized into two distinct roles: validators and leaders. Each node in the network has the potential to serve as a validator, participating in the validation of block proposals within a network that is segmented into D shards. The network consists of N nodes, denoted as $\mathbb{N} = \{1, \ldots, N\}$. An episode, represented by ε, is defined as a blockchain period during which every node has successfully served as a leader at least once. In this role, a node is responsible for proposing blocks to its shard peers for validation. Notably, the leadership role is considered trivial, as the system's focus is not on the consensus algorithm but on the distributed verification process.

3.2. Workflow overview

The proposed TBDD framework follows the workflow in Fig. 3.

Step-1. *Trust table updated.* The first step in the workflow is updating the trust table. The trust table shows the global trust scores for individual users. Then, the system checks if conditions for triggering Algo. 1 are met.

Step-2. *Train the DRL algorithm.* This step trains the DRL algorithm using the collected network data, which includes running simulations to optimize shard allocation policies based on real-time network conditions. After initiating the DRL training process, the system enters a waiting state and dedicates its computational resources to the training procedure. Note that the TC conducts "virtual resharding" trials to evaluate outcomes and rewards for different actions; however, these

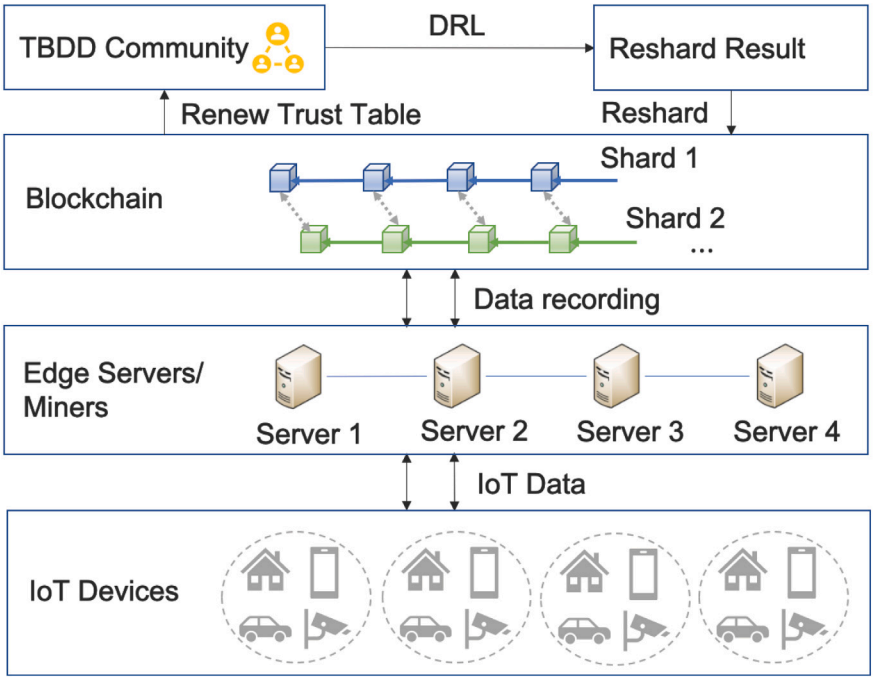


Fig. 1. System model.

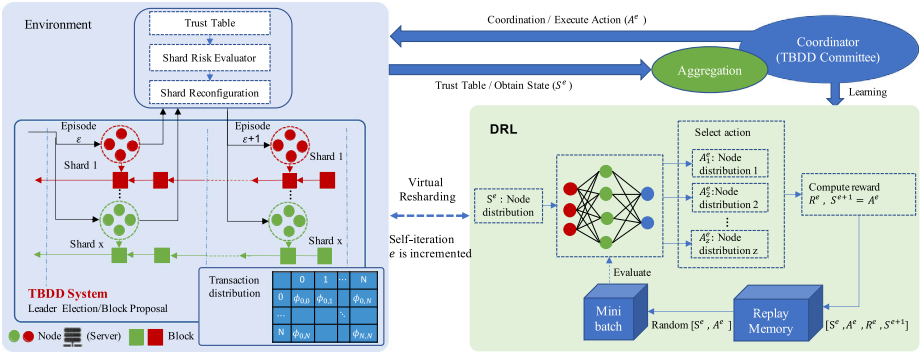


Fig. 2. This figure illustrates the proposed blockchain sharding system architecture - TBDD. The left section shows the TBDD system with nodes in red and green indicating different shards, where blocks are created and linked red and green arrows within each shard across episodes ε and $\varepsilon+1$. Black arrows signify the TBDD monitoring and resharding processes. The transaction distribution table represents node transaction volumes, highlighting potential discrepancies between honest and dishonest nodes. On the right, the DRL mechanism evaluates node distributions and selects actions to optimize shard configuration. The coordinator, symbolized by the blue oval, aggregates information and guides the learning process. This cohesive depiction aims to clarify the intricacies of our sharding approach and its operational flow.

trials occur solely within the TC, and the final sharding action is not implemented until it has fully converged.

Step-3. *Update shard allocation.* TC updates shard allocation policies in real time once the DRL model completes training and allocates nodes to shards based on the result.

Step-4. *Monitor network performance.* As shard allocation policies are updated, monitoring network performance is important, which includes tracking network metrics such as transaction distribution and volume, node location, and colluding risk.

After ***Step-4***, the TC moves on to the retraining phase of the DRL model, incorporating the latest information from the trust table obtained in ***Step-1***. Subsequently, the TC updates the shard allocation strategy in ***Step-3*** based on the new insights gained from the retrained DRL algorithm. This cyclical process follows a similar set of steps, beginning from ***Step-1*** and progressing through ***Step-4***.

3.3. System assumptions

There are several assumptions in this paper. These assumptions are considered from multiple perspectives, such as attack and system environment.

3.3.1. Attack assumption

The collusion behavior of nodes refers to the situation where multiple nodes work together to manipulate the network and gain some unfair advantage. This dishonest behavior can overload the network, leading to delays and service disruptions. An attack model focusing on collusion attacks in blockchain sharding is designed in this paper, which considers two types of participants: **dishonest nodes** and **honest nodes**.

Dishonest nodes. Dishonest nodes aim to be allocated in the same shard and then intentionally propose invalid results to honest nodes' blocks. This consequence can be achieved by sending many invalid transactions across dishonest nodes, regardless of whether they are cross-shard or intra-shard transactions. Suppose a dishonest node finds

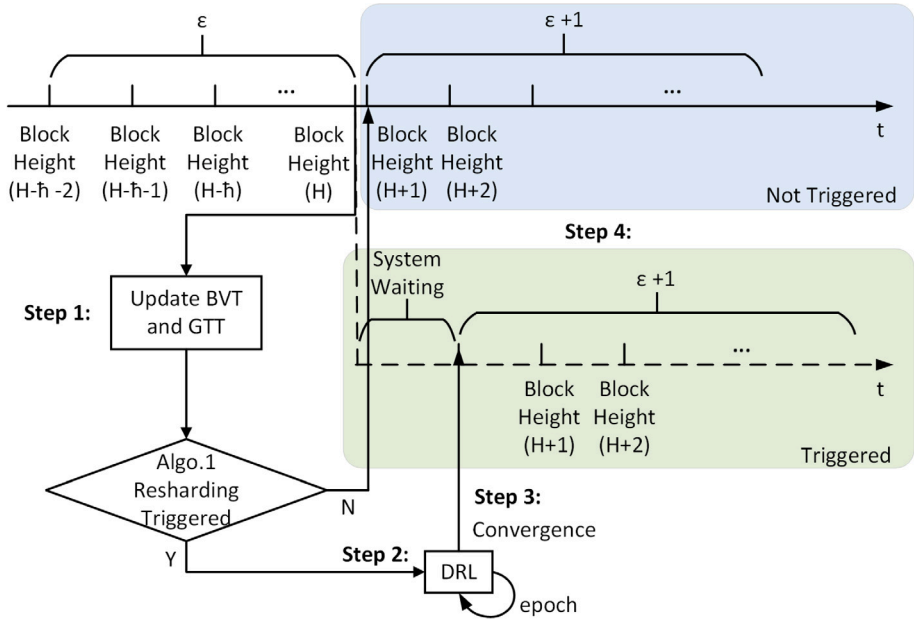


Fig. 3. The proposed blockchain sharding system flowchart - TBDD, including four steps. ① Trust table updated: update the Block Verification Table (BVT) and Global Trust Table (GTT), checking whether triggering Algo. 1. ② Train the DRL algorithm: use the DRL-based model to virtually resharding through several epochs and output a new node allocation result. ③ Update shard allocation: allocate nodes to the shards according to the DRL-based training result. ④ Monitor network performance: monitor and step into retraining.

no other teammates in the same shard. In that case, it may hide by pretending to be honest and following honest nodes' behavior, avoiding suspicion. Dishonest nodes can also utilize the global trust table's information to enhance their trust by imitating the conduct of honest nodes, selectively endorsing or opposing blocks according to proposers' trust scores.

In (1), the trust score $\mathcal{G}_i^{\varepsilon}$ is normalized of the ith node to a range of $[0, 1]$. Let $\mathcal{G}_{min}^{\varepsilon}$ and $\mathcal{G}_{max}^{\varepsilon}$ be the minimum and maximum possible raw trust scores. The current episode's normalized value $\mathbf{G}_i$ is calculated by the previous episode's global trust score.

$$\mathbf{G}_i = \frac{\mathcal{G}_i^{\varepsilon-1} - \mathcal{G}_{min}^{\varepsilon-1}}{\mathcal{G}_{max}^{\varepsilon-1} - \mathcal{G}_{min}^{\varepsilon-1}}. \tag{1}$$

The probabilities related to voting behavior are utilized, in which the probability of a node voting for a block is denoted as P_{vote}, while the probability of voting against a block is represented by $P_{\text{not\_vote}}$. Furthermore, the probabilities of voting behavior between dishonest and honest nodes are distinguished. The probability of dishonest nodes engaging in honest voting and dishonest voting is respectively denoted as $P_{\text{vote}}^{\text{dishonest}}$ and $P_{\text{not\_vote}}^{\text{dishonest}}$. Let F be the probability of a node's vote failing to reach others due to network issues, with $F \in [0, 1]$. Assume A is the proportion of dishonest nodes in the shard, with $A \in [0, 1]$. A strategy threshold τ can be defined, with $\tau \in [0, 1]$, determining when a dishonest node would try to hide by pretending to be honest. If $A < \tau$, the dishonest nodes pretend to be honest and follow honest nodes' behavior; otherwise, they follow the collusion strategy and favor their conspirators. A block verification result, denoted by U, takes a $U = 1$ if it matches the local version and $U = 0$ otherwise. A weighting factor w_G based on the normalized trust score $\mathbf{G}$ and a weighting factor w_U can be defined based on the block verification result in U. The probability distribution for dishonest nodes (2) and (3) can be defined as follows:

$$P_{\text{vote}}^{\text{dishonest}} = \begin{cases} (1-F) \cdot w_G \cdot \mathbf{G}_i \cdot w_U \cdot U, & \text{if } A < \tau \\ (1-F) \cdot \textbf{CollusionStrategy}(\kappa), & \text{otherwise} \end{cases} \tag{2}$$

$$P_{\text{not\_vote}}^{\text{dishonest}} = 1 - P_{\text{vote}}^{\text{dishonest}}, \tag{3}$$

where $\textbf{CollusionStrategy}(\kappa)$ signifies a strategy in which attackers consistently vote in favor of their teammates while voting against honest leaders with a probability denoted by $\kappa \in [0, 1]$. The collusion strategy (4) is to give valid results to all dishonest partners, attack honest nodes according to a particular proportion, and give them non-valid.

$$\textbf{CollusionStrategy}(\kappa) = \begin{cases} 1, & \text{dishonest nodes} \\ 1-\kappa, & \text{honest nodes} \end{cases} \tag{4}$$

Honest nodes. Honest nodes rely on the global trust table, providing an overview of high-risk or low-risk users without explicitly identifying dishonest nodes. During the intra-consensus phase, honest nodes rely on block verification, voting for a block only if it matches their local version. A composite probability distribution considering the trust-based voting distribution and the block verification distribution is proposed in this paper by weighting the probability of voting for a block based on the proposer's trust score and adjusting the weight according to the block verification result. This composite distribution allows honest nodes to make more informed decisions when voting for or against proposed blocks, considering both the proposer's trust score and the consistency of the proposed block with their local version. The combined probability distribution for honest nodes (5) and (6) can be calculated as follows:

$$P_{\text{vote}}^{\text{honest}} = (1-F) \cdot w_G \cdot \mathbf{G}_i \cdot w_U \cdot U, \tag{5}$$

$$P_{\text{not\_vote}}^{\text{honest}} = 1 - P_{\text{vote}}^{\text{honest}}. \tag{6}$$

The block verification table can be obtained by simulating honest and dishonest nodes' behavior using these attack models. The effects of collusion attacks on the overall system's security and performance in blockchain sharding systems can be analyzed.

Along with honest and dishonest nodes, the concept of high-risk and low-risk nodes is introduced in the proposed system. It is important to clarify that high-risk and low-risk nodes differ from honest and dishonest nodes. The node evaluation principle classifies nodes as high-risk when their trust scores fall below a specific threshold, while nodes with trust scores above this threshold are categorized as low-risk nodes. These classifications do not necessarily mean high-risk or low-risk nodes are inherently honest or dishonest. Instead, they indicate the level of trustworthiness based on the evaluation criteria. By

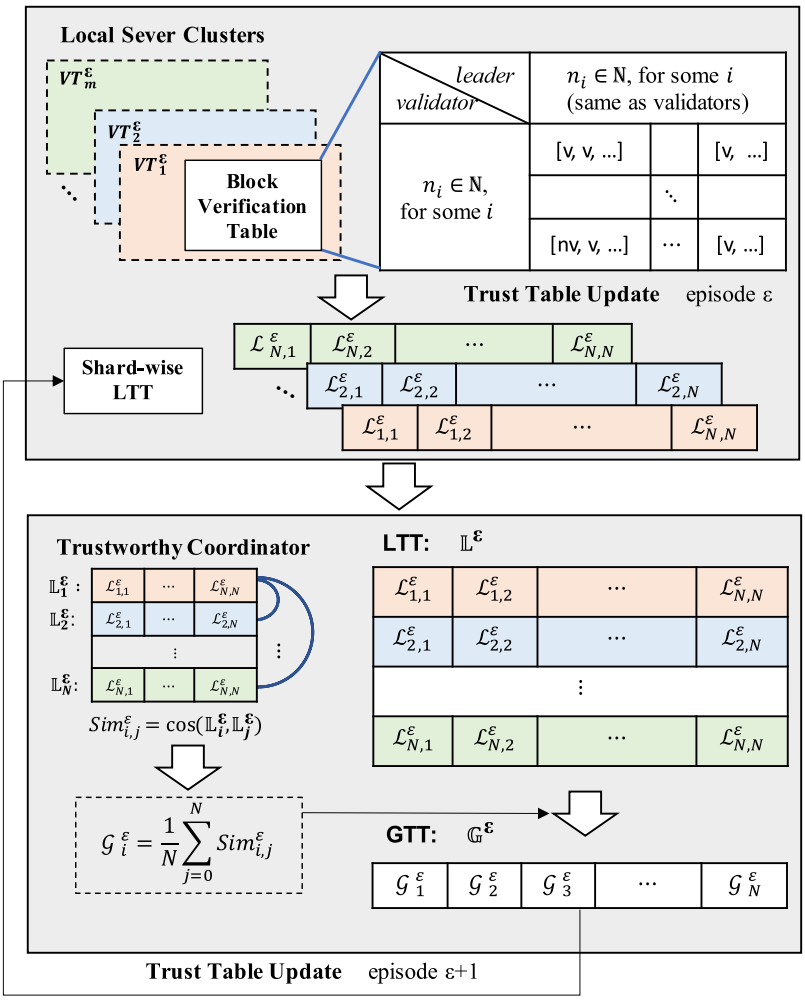


Fig. 4. The flow diagram of the proposed computing trust score system comprises BVT, LTT, and GTT.

considering the presence of high-risk and low-risk nodes in the network, The system can better identify and respond to potential collusion attacks, improving the security and integrity of the blockchain sharding framework.

3.3.2. Environment assumption

The system operates within permissioned blockchain systems, restricting network access to specific nodes. Despite this limitation, the system remains decentralized as it is maintained by the TC, a committee fairly elected by all peers within the network. For the training process to be considered trustworthy, it is important that the TC is reliable and has a transparent record when training the DRL model. Furthermore, Assumptions involve at least a certain number of members per shard, and the claim that the count of dishonest nodes in each shard does not surpass a certain number of the overall node count is based on the Byzantine Fault Tolerance (BFT) principle [35]. In case of node failure or a dishonest node attack, the system continues functioning with the remaining nodes.

4. TBDD: A trust-driven and DRL-based approach to optimize throughput and security

TBDD is proposed based on the results and analysis in Section 3, which is composed of the Block Verification Table (BVT), Local Trust Table (LTT), and Global Trust Table (GTT), Shard Risk evaluator and Shard reconfiguration (see Fig. 4 and Table 1).

4.1. Trust scheme

4.1.1. Block verification table (BVT)

The BVT aims to record the validation results for each shard. The verification table for the xth shard in the εth episode is denoted as VT^{ε}_{x}. The size of VT^{ε}_{x} is determined by the number of nodes in the shard, with dimensions of $N_x \times N_x$, where N_x represents the number of nodes in the xth shard. The verification result table can be visualized as a two-dimensional matrix where each cell corresponds to the set of verification results generated by the node that proposed a block. The size of each cell in the matrix corresponds to the number of times the leader produced blocks. During the trust table update episode, assume that there are sufficient votes to guarantee that each node within the shard will be elected as a leader at least once, resulting in block production. In this assumption, the leader can expect to obtain a minimum of one ballot from himself. Furthermore, the consensus process complies with the weak synchrony assumption [36], which indicates a finite upper limit on message delays.

4.1.2. Trust table

In TBDD, two distinct trust tables are utilized: LTT and GTT. These tables are dynamic and regularly refreshed to capture the latest data on each node's performance and behaviors within the network. By constantly monitoring and evaluating the LTT and GTT, the TBDD system can make strategic and informed shard allocation decisions, ensuring a reliable and secure network operation.

Table 1
Notation and definition used in the trust table.

Notation	Description
N	The number of nodes in the network
N_x^ε	The number of nodes in the xth shard in episode ε
$\mathbb{N}$	The set of all nodes in the network
$n_i \in \mathbb{N}$	The ith node in the network
ι_j^ε	The number of times when the jth node is elected as the leader in episode ε
D	The number of shards in the network
ε	The trust table update in episode ε
e	The eth epoch during the DRL iteration
$\mathcal{F}_{i,j}^\varepsilon$	The indirected feedback of jth leader from the ith node in episode ε
$\hat{\mathcal{F}}_{i,j}^\varepsilon$	The directed feedback from the jth node to the ith leader in episode ε
$\mathcal{L}_{i,j}^\varepsilon$	The local trust from the ith node to the jth node in episode ε
$\mathbb{L}_i^\varepsilon$	The local trust table for the ith node in episode ε
$\mathbb{L}^\varepsilon$	The concatenated local trust tables across all nodes in episode ε
$\mathcal{G}_i^\varepsilon$	The global trust for the ith node in episode ε
$\mathbb{G}^\varepsilon$	The global trust table in episode ε
$\mathbf{G}_i$	The normalized trust score of the ith node
θ_x^ε	The average global trust of xth shard in episode ε
$\bar{\theta}^\varepsilon$	The average value of all θ_x^ε
h_x	The list of high-risk nodes in the xth shard
f_{total}	The entire network's fault tolerance threshold for dishonest nodes
f_{intra}	The shard's fault tolerance threshold for dishonest nodes
ϕ_{in}	The number of intra-shard transactions (ISTs)
ϕ_{cr}	The number of cross-shard transactions (CSTs)

The local trust table is denoted as $\mathbb{L}^\varepsilon$. Each row of this table is specifically assigned to a node within the shard, and these nodes are represented as $1 \times N$ entries within the table, which is term as the local trust table of the ith node at the ε epoch, denoted as $\mathbb{L}_i^\varepsilon$. When these individual local trust tables are concatenated, it is represented as $\mathbb{L}^\varepsilon$, forming a comprehensive $N \times N$ table where $i, j \leq N$. Within this table, each element represents the trust score sent from the ith node to the jth node, denoted as $\mathcal{L}_{i,j}^\varepsilon$. The computation of each element in the LTT of ith node is shown in (7):

$$\mathcal{L}_{i,j}^\varepsilon = \begin{cases} \alpha \mathcal{F}_{i,j}^\varepsilon + \beta \hat{\mathcal{F}}_{i,j}^\varepsilon + \mu \mathcal{G}_i^{\varepsilon-1}, \\ \qquad \text{if } n_i, n_j \text{ in the same shard} \\ \mathcal{G}_i^{\varepsilon-1}, \\ \qquad \text{if } n_i, n_j \text{ in different shards} \end{cases} \tag{7}$$

when calculating the trust score in the same shard, three feedbacks are included: Indirected feedback $\mathcal{F}_{i,j}^\varepsilon$, Directed feedback $\hat{\mathcal{F}}_{i,j}^\varepsilon$ and Global trust score from the last episode $\mathcal{G}_i^{\varepsilon-1}$. Each component has a distinct proportion represented by α, β, μ. These proportions sum up to 1 (i.e., $\alpha+\beta+\mu = 1$). Only the global trust score from the previous episode is considered for calculating the trust scores of nodes in different shards.

Indirected feedback of each leader. The proportion of verification that the jth node passes when he is the leader at ε episode is shown in (8):

$$V_j^\varepsilon = \frac{\sum_{i=1}^{N_x^\varepsilon} v_{i,j}^\varepsilon}{\iota_j^\varepsilon N_x^\varepsilon}, \tag{8}$$

where ι_j^ε represents the times of the jth node being elected as the leader in the εth episode, and $v_{i,j}^\varepsilon$ signifies the total number of valid votes v cast by the ith node for the jth leader. Then, the indirected feedback $\mathcal{F}_{i,j}^\varepsilon$ is calculated as follows (9):

$$\mathcal{F}_{i,j}^\varepsilon = \gamma V_j^\varepsilon + \frac{\gamma^2}{N_x^\varepsilon - 2} \sum_{p \in \{i,j\}^C} \delta_{p,j}^\varepsilon \left(V_p^\varepsilon\right)^{\iota_j^\varepsilon - \delta_{p,j}^\varepsilon + 1}, \tag{9}$$

where the node, denoted as n_p, provides indirect feedback and participates in the voting process for the current leader l_j. The n_p node does not include block proposer l_j and the voter n_i itself. $\delta_{p,j}^\varepsilon$ refers to the ratio of non-empty votes (both valid and non-valid votes) cast from the pth node for the jth leader. γ is a discount rate similar to that used in reinforcement learning.

Directed feedback of each leader. The direct feedback of trust score is calculated as shown in (10):

$$\hat{\mathcal{F}}_{i,j}^\varepsilon = \frac{v_{j,i}^\varepsilon}{\iota_i^\varepsilon}, \tag{10}$$

where $v_{j,i}^\varepsilon$ denotes the count of valid votes v cast by the jth node for the ith node when n_i is leader. ι_i^ε indicates that the number of times of the ith node being elected as the leader during the εth episode.

Global trust of each leader from the history. The historical trust of each leader $\mathcal{G}_i^{\varepsilon-1}$ is inherited from the last episode.

The global trust table is denoted as $\mathbb{G}^\varepsilon$. Inspired by federated learning principles, the coordinator enhances credibility by updating the LTT according to the GTT's outcome after every iteration.

Cosine similarity calculation. Comparisons of cosine similarity among LTT rows reflect deviations in node-scoring behavior (11):

$$Sim_{i,j}^\varepsilon = cos(\mathbb{L}_i^\varepsilon, \mathbb{L}_j^\varepsilon), \quad i, j < N. \tag{11}$$

The global trust score for each node is determined by computing the mean of the cosine similarity between the node's scoring behavior in the LTT and the entire LTT (12):

$$\mathcal{G}_i^\varepsilon = \frac{1}{N} \sum_{j=1}^{N} Sim_{i,j}^\varepsilon, \tag{12}$$

$\mathcal{G}^\varepsilon$ is a $1 \times N$ vector capturing the global trust, where element $\mathcal{G}_i^\varepsilon$ is the global trust of the ith node in the current shard.

4.2. Resharding trigger: The shard risk evaluator

The resharding process is triggered when certain conditions are met, leading to the trigger phase (the green block in Fig. 3), as shown in Algo. 1. A global trust threshold ρ_t is defined to differentiate between low-risk and high-risk nodes. Nodes are high-risk and possibly dishonest if their $\mathcal{G}^\varepsilon$ are lower than ρ_t. Nodes are *more likely* to be honest if their global trust exceeds ρ_t. The fault tolerance threshold f_{intra} is defined within each shard. If the number of dishonest nodes exceeds the threshold f_{intra} in the shard, the shard is labeled as corrupted and triggers resharding. Furthermore, resharding is also triggered when the

Algorithm 1: Shard risk evaluator

Input:
x: The x-th shard;
N_x^ϵ: Number of nodes in the x-th shard at ϵ episode;
$\mathcal{G}_i^\epsilon$: Global trust of the i-th node;
f_{intra}: The faulty tolerance of dishonest nodes in any shard;
φ_{cr}: The CST ratio since last episode;
ρ_t: The threshold of differentiating a low-risk or high-risk node in terms of trust score;
ρ_{cr} : The threshold in terms of CST;
Output:
{*Trigger*, *Not trigger*}

$\rho_{cr} := 0$
$h_x := []$ **for each shard** x
for $x \in [1, D)$ **do**
  for $n_i \in x$ **and** $i \in [1, N)$ **do**
    if $\mathcal{G}_i^\epsilon < \rho_t$ **then**
      $h_x \leftarrow n_i$
  if $|h_x| / N_x > f_{intra}$ **then**
    return *Trigger*
if $\varphi_{cr} > \rho_{cr}$ **then**
  return *Trigger*
return *Not Trigger*

ratio of CST surpasses the setting threshold ρ_{cr}. The ratio of the CST is calculated as follows:

$$\varphi_{cr} = \frac{\phi_{cr}}{\phi_{cr} + \phi_{in}}, \tag{13}$$

where ϕ_{cr} represents the CST count, as given by:

$$\phi_{cr} = \frac{1}{2}\left(\sum_{i=1}^{N}\sum_{j=1}^{N}\phi_{i,j} - \sum_{x=1}^{D}\sum_{i=1}^{N_x^\epsilon}\sum_{k=1}^{N_x^\epsilon}\phi_{i,k}\right), \tag{14}$$

where $\phi_{i,j}$ represents the transaction count between the ith and jth nodes in the network. $\phi_{i,k}$ represents the transaction count between the ith node and the kth node in the shard, which equals the sum of the Intra-Shard Transaction (IST). The ϕ_{cr} is obtained by subtracting the IST count from the total transactions count among network nodes.

4.3. DRL framework

4.3.1. Optimizations, rewards, and DRL

The DRL-based model enhances blockchain sharding systems by dynamically allocating nodes depending on network conditions and automating complex decision-making procedures. The reward function optimizes node allocation while maintaining security constraints through DRL adaptation. In TBDD, the agent aims to maximize earnings by calculating the objective function that is composed of 6 reward components, where E_a, E_b, λ_a, λ_a, and λ_c are all constant.

Shard load balance (ξ). Aiming to distribute the number of nodes across each shard evenly, (15) is defined. If there is a significant disparity in the node count per shard, resharding is augmented to counteract potential 1% attacks. [37]

$$\xi = \begin{cases} E_a, & \text{shard load balance} \\ -E_a, & \text{shard load unbalance} \end{cases} \tag{15}$$

Corrupted shards portion (ϱ). As shown in (16), the agent receives a reward if no shard is occupied. Otherwise, the agent receives a penalty.

$$\varrho = \begin{cases} E_b, & \text{no shard is corrupted} \\ -E_b, & \text{at least one shard is corrupted} \end{cases} \tag{16}$$

CST ratio (η). As shown in (17), if the CST ratio is less than the specified threshold ρ_{cr}, the agent gets rewarded; otherwise, it receives a penalty.

$$\eta = \lambda_a\left(\rho_{cr} - \varphi_{cr}\right)\lambda_b^{\lambda_c|\varphi_{cr}-\rho_{cr}|}, \tag{17}$$

Nodes shifting ratio (ψ). As shown in (18), if a node switches the shard, its action is noted as 1; otherwise, it is 0. The overall count of shifted nodes corresponds with the penalty score. The more nodes relocate, the more computational resources are consumed by shard synchronization, resulting in a higher level of punishment.

$$\psi = \frac{1}{N}\sum_{j=1}^{N}\upsilon_j, \quad \upsilon_j = \begin{cases} 1, & \text{nodes moving} \\ 0, & \text{nodes staying} \end{cases} \tag{18}$$

Intra-shard's trust variance (Ω_{in}). As shown in (19), trust variance represents the trust distribution within each shard. A larger trust variance of a shard indicates a better distinction between trustworthy and untrustworthy participants. Thus, a larger intra-shard trust variance collected by the agent serves as a reward to indicate a clearer differentiation between honest and dishonest nodes.

$$\Omega_{in} = \frac{1}{D}\sum_{x=1}^{D}\frac{1}{N_x^\epsilon}\sum_{k=1}^{N_x^\epsilon}\left|\mathcal{G}_k^\epsilon - \theta_x^\epsilon\right|^2, \tag{19}$$

where $\mathcal{G}_k^\epsilon$ is the global trust value of nodes in the xth shard and θ_x^ϵ is the average of global trust in the xth shard.

Cross-shard's trust variance (Ω_{cr}). As shown in (20), it represents the deviation of trust value among different shards. A minor cross-shard trust variance indicates a more uniform distribution of dishonest nodes.

$$\Omega_{cr} = \frac{1}{D}\sum_{x=1}^{D}\left|\theta_x^\epsilon - \bar{\theta}^\epsilon\right|^2, \tag{20}$$

where $\bar{\theta}^\epsilon$ is the average value of all $\theta_x^\epsilon, x \in D$. Thus, the objective function can be defined as:

$$R = \xi + \varrho + \eta - \psi + \Omega_{in} - \Omega_{cr}, \tag{21}$$

$$\begin{aligned} &\text{Let } \mathbf{R} = [\xi, \varrho, \eta, \psi, \Omega_{in}, \Omega_{cr}], \\ &\text{objective: } \max_{\mathbf{R}} \sum^{\epsilon_{max}} R(\mathbf{R}), \\ &\text{s.t.} \quad |h_x| < N_x f_{intra}, \\ &\qquad N_x \geq N_{min}, \end{aligned} \tag{22}$$

where ξ is the balance reward for shard load; ϱ is the reward for the number of corrupted shards; η signifies the reward for low-level CST; ψ indicates the reward for the number of shifted nodes; Ω_{in} stands for the reward of intra-shard trust variance; and Ω_{cr} represents the reward of cross-shard trust variance. $\mathbf{R}$ is defined as the set of all rewards. Restrictions of the objective function R ensure the dishonest node count stays below the shard's fault tolerance. N_{min} denotes the minimum node requirement for each shard. Each shard mandates a minimum of four nodes in the setting, i.e., $N_{min} = 4$.

4.3.2. DRL-based sharding optimization model

The DRL model is used to assist in the blockchain reconfiguration process. The agent of the DRL model is acted by the TC, a committee elected by all peers in the network. The agent obtains the state from the node allocation. Then, the agent gains the reward by virtually resharding, deciding the optimal allocation strategy and executing the action for the new node allocation.

Agent: The agent is perceived as the TC, which consists of several nodes. These nodes implement the PBFT protocol to achieve consensus, representing the collective action of the TC. The agent executes a learning process and decides shard allocation based on real-time network conditions.

Environment: As illustrated in Fig. 2, the environment is viewed as a black box that executes the **Action** of the agents and obtains the **State**.

Table 2
Hyperparameters.

Notation	Description	Value
e_{max}	The number of epochs in DRL	[30, 100]
F	The probability of node failing to vote	20%
A	The fraction of dishonest nodes within one shard	[0, 30]
τ	The threshold for triggering colluding strategy	10%
γ	The discount factor for calculating OT of each leader	0.9
ρ_t	The threshold of differentiating low-risk nodes and high-risk nodes in terms of trust	0.67
f_{intra}	The threshold of faulty tolerance within one shard	$\lfloor (N_x - 1)/3 \rfloor$
ρ_{cr}	The threshold of triggering resharding in terms of CST ratio	0.4

In this paper, the sharding reconfiguration process in the blockchain sharding network is the environment.

The agent is considered to obtain a state $s \in \mathcal{S}$ based on the node allocation at the current episode ϵ in this paper. s denotes the distribution of all nodes across various shards in the current episode, while m precisely identifies the specific node present within a particular shard.

$$s = \left[m_{x,1}, \ldots, m_{x,i}, \ldots, m_{x,N}\right], x \in D, i \in N, \tag{23}$$

where $m_{x,i}$ is the x-shard to which the ith node belongs.

Episode (ϵ). An episode in the blockchain context is characterized as a period during which each node has successfully assumed the role of a leader at least once. The initiation of a new episode is signaled by the updating of the trust table, indicating that the duration of an episode can be flexibly modified to align with diverse requirements.

Actions ($\mathcal{A}$). The agent executes an action $a \in \mathcal{A}$ based on its decision produced by its local DRL-based learning during the current episode ϵ. The valid action space for the agent is formatted identically to $\mathcal{S}$, as given by $\mathcal{A} = \mathcal{S}$ with $s^{\epsilon+1} = a^{\epsilon}$.

Policy (π). A policy determines what action the agent would take next based on a set of rules. In this case, the process of nodes assigned to different shards is based on the policy that is continually updated and trained. i.e., $\pi(s, a) : \mathcal{S} \rightarrow \mathcal{A}$.

Reward function (r). The reward function is inherited from the objective function, $r : R(\xi, \varrho, \phi_{cr}, \psi, \Omega_{in}, \Omega_{cr})$.

DRL is a versatile method that allows agents to make decisions in dynamic environments effectively. Its ability to handle complex data and its proactive defense against malicious attacks make it an ideal solution for managing node and transaction interactions in sharding systems. The sharding optimization problem, which involves assigning each node to a specific shard, is a known non-convex, NP-hard problem due to the binary nature of decision variables [38]. Such complexity highlights the appropriateness of DRL as an approach to address this challenge compared to traditional methods such as convex optimization, underlining its significance in managing the complexities of sharded blockchain systems.

5. Experiment and evaluation

The experiment assesses the DRL-based sharding approach regarding convergence performance and stability under various environment settings, including the number of shards, the number of nodes, the resource distribution, and other practical settings. Note that the parameters in the following experiments align with real-world IoT scenarios, such as Mobile Edge Computing (MEC), where edge servers on vehicles or drones collect data from end devices such as sensors. These servers initiate resharding to optimize data processing by redistributing workload when moving across regions.

5.1. Experiment framework

An experimental framework is implemented in an environment with 2.5 GHz, 20 cores, 2 × Intel(R) Xeon(R) Gold 6248 CPU, NVIDIA Quadro P4000, 768 GB memory to evaluate a proposed blockchain sharding scheme. A virtual machine is created using Python 3.8.10 and Pytorch 1.13.1. In this setting, the discrete DRL algorithms, DQN and PPO, are used and run over 30−100 epochs. The range of total nodes from 0−16 is set in the environment. The transaction distribution model between nodes follows the normal distribution. The trustworthy coordinator is implemented by deploying the smart contract.

In the experimental setup, the blockchain sharding system TBDD is assessed against other sharding techniques such as random-based sharding [11], community-based sharding [15], and trust-based sharding [17]. The dishonest nodes in the range of 0−5 are accounted for, which employ collusion strategies during block verification. These nodes may produce deceptive block verifications and send CSTs across different shards. Initially, these dishonest nodes are scattered randomly across shards. Positive noises are introduced to their transaction counts to model interactions between dishonest nodes in distinct shards. The resulting transaction distribution table is influenced by normally distributed transactions. The experiments' hyperparameters are detailed in Table 2.

As the intra-shard fault tolerance threshold f_{intra} is set to $\lfloor (N_x - 1)/3 \rfloor$, the relationship between the total number of nodes N in the network, the number of shards D, and the total fault tolerance f_{total}, is evaluated as (24)

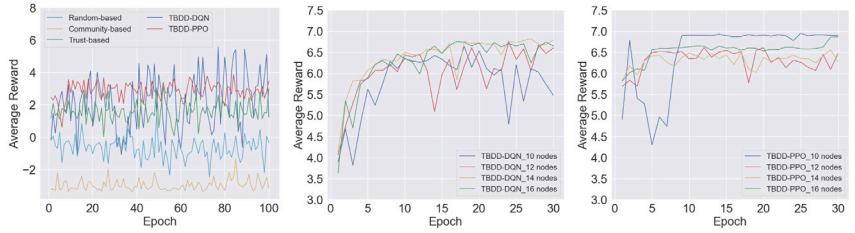
$$f_{total} = \left\lfloor \left(\left\lfloor \frac{N}{D} \right\rfloor - 1 \right) / 3 \right\rfloor \times D, \tag{24}$$

by which one can realize whether the entire network has failed.

5.2. Experiment results

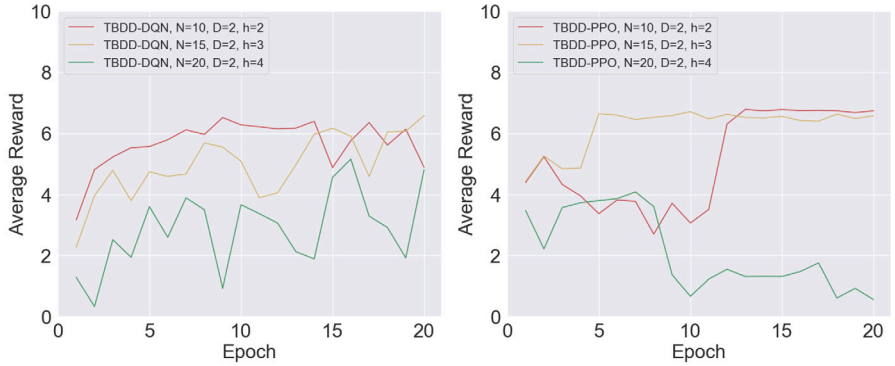
In the experimental evaluations, the performance of the TBDD system is compared with random-based, community-based, and trust-based sharding approaches using metrics such as CST ratio, shard risk variance, corrupted shard number, and convergence speed. Through these evaluations, the effectiveness and robustness are validated by the proposed approach. In the following figures, each figure is labeled in terms of the number of nodes N, shards D, and dishonest nodes h, respectively.

Fig. 5(a) illustrates the average rewards achieved in a single epoch with different sharding schemes. The community-based sharding technique recorded the lowest reward. While the scheme effectively reduces cross-shard transactions, it remains susceptible to adaptive collusion attacks due to its tendency to group a large number of dishonest nodes within the same shard, thereby compromising the shard's security. The random-based scheme fares slightly better, with its rewards hovering around zero. The trust-based sharding technique is more proficient at evenly distributing dishonest nodes across different shards, mitigating the risk of a single shard being dominated. However, it leads to a higher number of cross-shard transactions. Consequently, its rewards are marginally less than those of DQN and PPO. Upon reaching the 10th epoch, both TBDD-PPO and TBDD-DQN consistently outperform



(a) $N = 16, D = 2, h = 4$. (b) $N = \{10, 12, 14, 16\}$, $D = 2$, $h = 2$. (c) $N = \{10, 12, 14, 16\}$, $D = 2$, $h = 2$.

Fig. 5. Fig. 5(a) represents the comparison among Random-based, Community-based, Trust-based, TbDd-DQN and TbDd-PPO across 100 epochs. Figs. 5(b) and 5(c) represent the reward performance between TbDd-DQN and TbDd-PPO across varying node numbers in the environment settings across 30 epochs, respectively.



(a) $N = \{10, 15, 20\}$, $D = 2$, $h = \{2, 3, 4\}$, TbDd-DQN (b) $N = \{10, 15, 20\}$, $D = 2$, $h = \{2, 3, 4\}$, TbDd-PPO

Fig. 6. Tracing of the impact of the number of nodes under the same dishonest node ratio with respect to the reward performance, $N = \{10, 15, 20\}$, $D = 2$, $h = \{2, 3, 4\}$.

the other sharding schemes, exhibiting consistently high rewards. This observation indicates that the agent received the maximum reward associated with the action.

Figs. 5(b)–5(c) depict the convergence of rewards under various node numbers for TbDd-DQN and TbDd-PPO, respectively. The reward becomes more stable as the number of nodes increases. Across Figs. 5(a)–5(c), a consistent trend emerges caused by the constrained approach in the policy update process. PPO achieves more stable rewards. The instability of DQN is because it combines the direct value function estimation method and ϵ greedy exploration strategy.

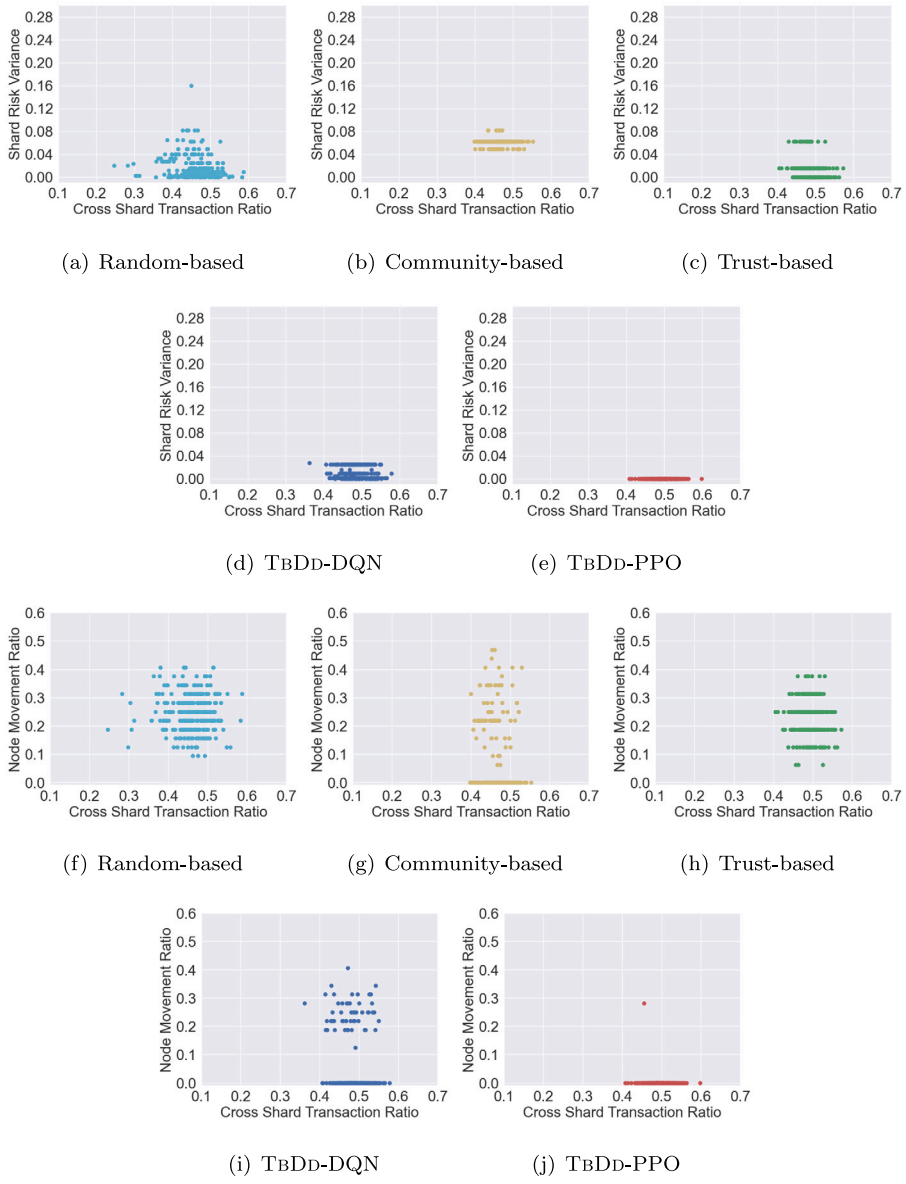
Fig. 6 shows the impact of the varying network scale with 10, 15, and 20 blockchain nodes and a constant ratio of dishonest to honest nodes (i.e., 0.2). It can be seen from the figure that fewer nodes lead to higher average rewards, consequently enhancing security and efficiency, for both the TbDd-DQN and TbDd-PPO results. This trend arises because as the number of dishonest nodes grows, the likelihood of collusion attacks increases. Additionally, the results indicate that the PPO method exhibits a greater stability compared to the DQN scheme.

Figs. 7(a)–7(e) (Shard Risk Variance vs. CST Ratio) illustrate the trade-offs between the risk distribution among shards and the system's ability to handle cross-shard transactions. The ideal positioning in these scatter plots is towards the bottom-left corner, indicating low shard risk variance (a secure and balanced distribution of nodes) and low CST ratio (high scalability with minimal cross-shard transactions). The random-based approach shows a scattered distribution, signifying a lack of predictability in achieving security and scalability. Community-based and trust-based strategies display a compromise between scalability and security, with community-based leaning towards scalability, sacrificing some security, and trust-based focusing heavily on security, which may limit scalability. The proposed TbDd-DQN and TbDd-PPO methods demonstrate a more balanced approach, achieving lower risk variance without significantly compromising on the CST ratio, thereby endorsing the efficacy of the TbDd framework.

Figs. 7(f)–7(j) (Node Movement Ratio vs. CST Ratio) present the scalability and efficiency aspect, where a lower Node Movement Ratio implies reduced overhead and increased stability due to less frequent reassignments of nodes to different shards. The Random-based sharding approach results in an unpredictable pattern, leading to potentially frequent node reassignments that can increase system complexity. The adaptive collusion behavior of dishonest nodes significantly impacts both community-based and trust-based sharding strategies. This malicious behavior intentionally misleads the sharding mechanisms, resulting in increased node movements for both strategies. In contrast, the TbDd-DQN and TbDd-PPO strategies demonstrate a reduction in node movements, with TbDd-PPO showing the greatest stability and efficiency in system operation among the evaluated strategies. Fig. 7 highlights the TbDd framework's ability to achieve a balanced and secure blockchain system, capable of addressing the complex trade-offs involved in blockchain system design.

In Fig. 8, it is shown that the average trust of dishonest nodes surpasses that of honest nodes as the number of dishonest nodes increases. The uppermost horizontal line in each column represents the maximum trust value among all nodes, while the bottom horizontal line represents the minimum trust value. The horizontal line at the midpoint signifies the median trust value of all nodes. When the total number of nodes is 16, and the total shard number is 2, the overall fault tolerance threshold f_{total} is 4 based on (13). Consequently, a shard becomes vulnerable and is corrupted when the number of dishonest nodes $h \geq 5$.

As shown in Figs. 9(a)–9(l), the blue curves represent the average trust of honest nodes, and the red curves are dishonest nodes. Consistent results from Fig. 8 reveal that the system can effectively perform sharding if the proportion of dishonest nodes remains below



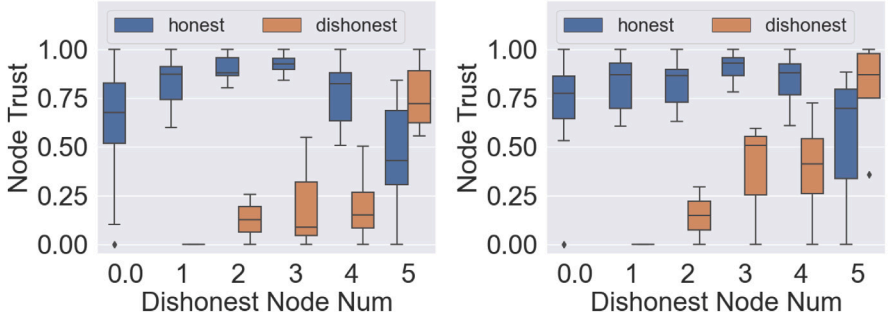
(a) Random-based (b) Community-based (c) Trust-based

(d) TBDD-DQN (e) TBDD-PPO

(f) Random-based (g) Community-based (h) Trust-based

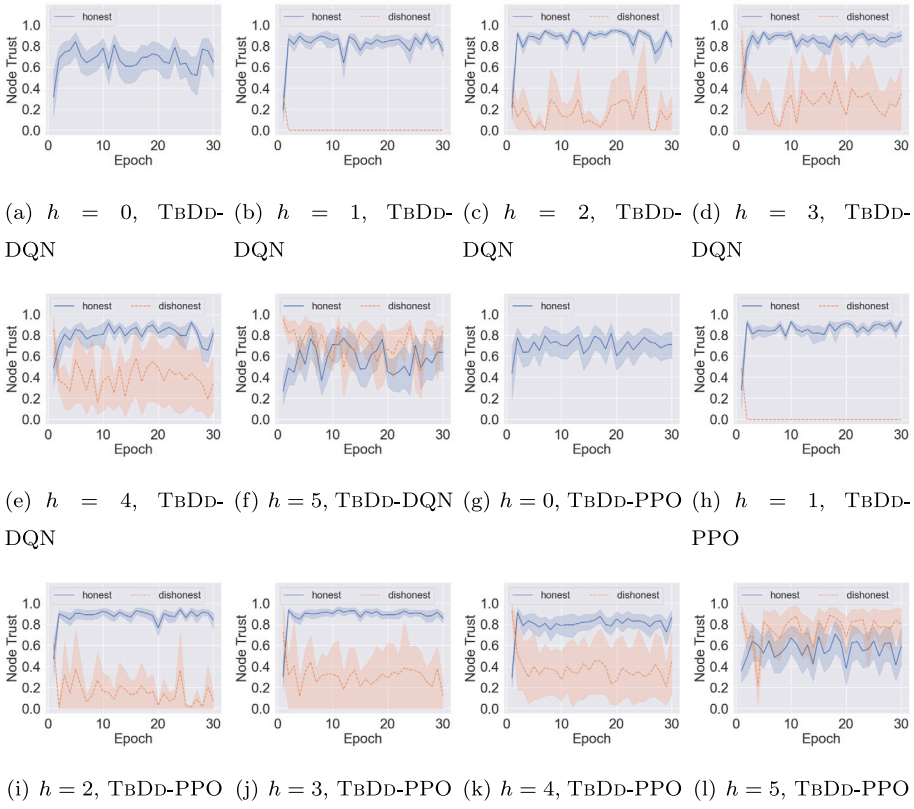
(i) TBDD-DQN (j) TBDD-PPO

Fig. 7. Comparison of different sharding schemes between random-based, community-based, trust-based, TBDD-DQN, and TBDD-PPO with $N = 16, D = 2, h = 4$ over 100 epochs. The x-axis represents the CST ratio, and the y-axis represents the cross-shard's trust variance through Figs. (a)–(e), while through (f)–(j), the x-axis represents the CST ratio and the y-axis represents the Node Movement Ratio. Lower values in the Shard Risk Variance and Node Movement Ratio indicate enhanced security and system stability, respectively. Lower CST Ratios indicate higher scalability of the sharding scheme.



(a) $N = 16, D = 2, h = \{0, 1, 2, 3, 4, 5\}$, TBDD-DQN (b) $N = 16, D = 2, h = \{0, 1, 2, 3, 4, 5\}$, TBDD-PPO

Fig. 8. Tracing of the impact of the number of dishonest nodes for node trust with DRL approach, $N = 16, D = 2, h = \{0, 1, 2, 3, 4, 5\}$. The left subfigure uses the DQN algorithm. The right subfigure uses the PPO algorithm.

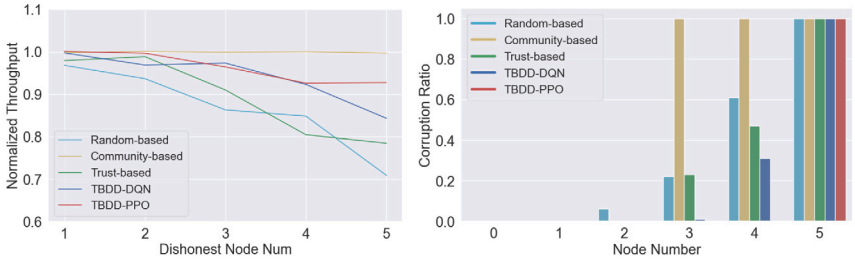


(a) $h = 0$, TBDD-DQN (b) $h = 1$, TBDD-DQN (c) $h = 2$, TBDD-DQN (d) $h = 3$, TBDD-DQN

(e) $h = 4$, TBDD-DQN (f) $h = 5$, TBDD-DQN (g) $h = 0$, TBDD-PPO (h) $h = 1$, TBDD-PPO

(i) $h = 2$, TBDD-PPO (j) $h = 3$, TBDD-PPO (k) $h = 4$, TBDD-PPO (l) $h = 5$, TBDD-PPO

Fig. 9. Tracking global trusts $\mathcal{G}$ between honest and dishonest nodes as the number of dishonest nodes escalates within the TBDD-DQN and TBDD-PPO schemes with $N = 16$, $D = 2$, $h = \{0, 1, 2, 3, 4, 5\}$.



(a) $N = 16, D = 2, h = \{0, 1, 2, 3, 4, 5\}$ (b) $N = 16, D = 2, h = \{0, 1, 2, 3, 4, 5\}$

Fig. 10. Fig. 10(a) shows the relationship between the number of dishonest nodes, system throughput, and corrupted shards is interconnected. Fig. 10(b) compares corrupted shard ratio with different sharding techniques across 100 epochs.

f_{total} of the total node count. However, if the number of dishonest nodes exceeds the f_{total} threshold, any sharding approaches, including the proposed schemes TBDD-DQN and TBDD-PPO, cannot safely execute sharding and are vulnerable to the 1% attack, which is beyond the scope of the investigation. On the other hand, having a lower count of dishonest nodes still enables the implementation of more shards within the system, leading to improved overall performance and scalability.

The normalized throughput metric evaluates throughput across different sharding methods influenced by dishonest nodes. This metric compares the throughput with and without dishonest nodes. A higher ratio suggests dishonest nodes have minimal impact on transaction throughput, while a lower one indicates a significant negative effect. Additionally, the benefits of the proposed system are highlighted by comparing the shard corruption ratio over the last 100 rounds among various sharding approaches. As depicted in Figs. 10(a)–10(b), random-based sharding exhibits the lowest throughput and compromised security. As dishonest node counts rise, there is a pronounced decline in scalability coupled with an increased number of corrupted shards. The Community-based sharding method has the best scalability and maintains a high throughput. Yet, its oversight on the security front results in a higher rate of shard corruption. In comparison, the trust-based sharding method reduces the chances of shard corruption but lags in throughput, signaling its scalability constraints. Without collusion attacks and within the tolerable limit of dishonest nodes, the proposed TBDD-DQN and TBDD-PPO in this paper achieves approximately a 10% throughput improvement over random-based sharding method and a 13% increase compared to the trust-based method.

5.3. Discussion

Latency. The evaluation of network latency and overall system latency is presented in Table 3. Distributed blockchain nodes employ the Practical Byzantine Fault Tolerance (PBFT) consensus [36] algorithm for block generation. The latency across distributed nodes follows a unified normal distribution with mean latencies of 0.1, 0.5, and 1 seconds and a

Table 3
Latency (in seconds) with different numbers of nodes, dishonest nodes, and network latency.

Nodes	Network latency (s)	Block (s)	DQN (s)	DQN per block (s)	PPO (s)	PPO per block (s)
$N = 10, h = 2$	0.1	0.458	2092	0.011	1369	0.007
	0.5	1.727		0.043		0.028
	1	3.223		0.080		0.052
$N = 12, h = 2$	0.1	0.334	2320	0.009	1787	0.007
	0.5	1.602		0.044		0.034
	1	3.094		0.085		0.065
$N = 14, h = 2$	0.1	0.374	2412	0.011	2194	0.010
	0.5	1.633		0.047		0.043
	1	3.139		0.090		0.082
$N = 15, h = 3$	0.1	0.375	2478	0.011	2646	0.012
	0.5	1.649		0.049		0.052
	1	3.150		0.093		0.100
$N = 16, h = 3$	0.1	0.446	4021	0.022	4388	0.024
	0.5	1.730		0.084		0.093
	1	3.227		0.157		0.173
$N = 18, h = 3$	0.1	0.361	4475	0.020	10031	0.047
	0.5	1.636		0.089		0.215
	1	3.140		0.172		0.412
$N = 20, h = 4$	0.1	0.418	11592	0.065	49466	0.560
	0.5	1.703		0.264		2.281
	1	3.203		0.496		4.289

standard deviation of 0.1 s. The DRL algorithms in this paper, i.e., DQN and PPO, are executed on a high-performance computer featuring a 2.5 GHz CPU and 768 GB memory. Throughout the experiment, a sharding cycle of one day is considered [12,13], during which the DRL algorithms are executed daily, and block consensus proceeds following the results of DRL-based sharding to calculate the per-block latency resulting from the DRL algorithms.

Regarding the impact of network latency, Table 3 illustrates that the block time is around three times the network latency under the normal distribution network latency model and PBFT consensus protocol. A slight decrease in consensus latency is observed when the number of dishonest nodes remains constant while the total number of nodes increases. Conversely, an increase in the number of dishonest nodes is linked to a notable rise in consensus latency, indicating that a greater presence of dishonest nodes negatively impacts the time required to achieve consensus.

In terms of the impact of varying numbers of nodes on system latency, the experimental results in Table 3 indicate a small variation in block time with changes in the total number of nodes. Despite the exponential growth in DRL latency as the total number of nodes increases, the latency per block remains manageable. Table 3 further demonstrates that the PPO latency is lower than the DQN latency for smaller node quantities. However, the PPO latency increases at a faster rate compared to the latency of DQN. This observation highlights a trade-off between reward performance and system latency, as the PPO method typically yields superior rewards compared to the DQN method.

Other types of latency include block verification, sharding strategy optimization, and resharding phases. Tolerating delay during block verification is a necessary trade-off to prevent double-spending issues [39]. However, an extended offline DRL learning phase for the sharding strategy is not favorable. Mitigation can be achieved by aligning online sharding strategy learning with the execution of resharding. Latency concerns during resharding, due to extensive node synchronization, can be alleviated through state channels [40] that expedite off-chain transactions, thus reducing blockchain load and hastening resharding. Integrating state channels into TBDD enables certain IoT transactions to be executed off-chain during proposed block verification, which lessens the node verification load and enhances overall system responsiveness.

Edge-driven Protocol. The architecture of TBDD uniquely operates on edge servers, not directly on IoT sensor end nodes. This strategic placement ensures that the system can manage extensive computations associated with blockchain operations without overwhelming individual IoT devices. As a result, the scaling exhibited in the experiments is consistent and realistic, representing a practical implementation in real-world IoT networks. This edge-driven approach aligns with the broader move towards edge computing in IoT, capitalizing on its benefits to improve scalability and responsiveness.

Decentralized Coordination. Our design leverages a decentralized TC, acting as a trustworthy third-party coordinator, eliminating the vulnerabilities associated with a single centralized coordinator. This approach significantly mitigates the risks of a single point of failure in the system. The intra-consensus security within the decentralized TC ensures robust and reliable decision-making processes. Such a structure has been mirrored in existing studies [11,12], affirming its practicality and security. By embedding this design, TBDD further ensures system robustness, sustaining its promises of trustworthiness and integrity in diverse IoT settings.

6. Conclusion

In this paper, we introduce TBDD, a novel trust and DRL-based sharding framework, which represents a significant advancement in blockchain technology for IoT environments. TBDD surpasses existing random, community, and trust-based methods, demonstrating a 10% throughput advantage over random-based sharding and a 13% improvement compared to trust-based methods, along with achieving the lowest rate of corrupted shards. This enhancement in security and scalability makes it well-suited for real-world IoT applications such as smart cities and industrial IoT. By integrating trust mechanisms with DRL, TBDD effectively addresses major IoT issues such as scalability and security, elevating sharding technology and offering a robust, efficient solution for diverse IoT contexts. Its adaptability across various node sizes and minimal system latency showcase its potential to set new standards in the field.

CRediT authorship contribution statement

Zixu Zhang: Writing – original draft, Conceptualization, Data curation, Formal analysis, Investigation, Methodology, Resources, Software, Validation, Visualization, Writing – review & editing. **Guangsheng Yu:** Conceptualization, Formal analysis, Methodology, Supervision, Writing – review & editing. **Caijun Sun:** Data curation, Software, Visualization,

Writing – review & editing. **Xu Wang:** Conceptualization, Formal analysis, Project administration, Supervision, Validation, Writing – review & editing, Funding acquisition. **Ying Wang:** Writing – review & editing, Formal analysis, Supervision. **Ming Zhang:** Writing – review & editing. **Wei Ni:** Writing – review & editing. **Ren Ping Liu:** Writing – review & editing. **Andrew Reeves:** Writing – review & editing, Funding acquisition. **Nektarios Georgalas:** Writing – review & editing, Funding acquisition.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgments

This work was supported by BT Australasia Pty Ltd, through "*Open DLT-based Network Inventory and Service Management*".

References

[1] Y. Qian, D. Wu, W. Bao, P. Lorenz, The internet of things for smart cities: Technologies and applications, IEEE Netw. 33 (2) (2019) 4–5.

[2] J. Kang, R. Yu, X. Huang, M. Wu, S. Maharjan, S. Xie, Y. Zhang, Blockchain for secure and efficient data sharing in vehicular edge computing and networks, IEEE Internet Things J. 6 (3) (2018) 4660–4670.

[3] M. Singh, G.S. Aujla, A. Singh, N. Kumar, S. Garg, Deep-learning-based blockchain framework for secure software-defined industrial networks, IEEE Trans. Ind. Inform. 17 (1) (2020) 606–616.

[4] T.R. Gadekallu, M. Manoj, N. Kumar, S. Hakak, S. Bhattacharya, et al., Blockchain-based attack detection on machine learning algorithms for IoT-based e-health applications, IEEE Internet Things Mag. 4 (3) (2021) 30–33.

[5] E. Park, Y. Cho, J. Han, S.J. Kwon, Comprehensive approaches to user acceptance of internet of things in a smart home environment, IEEE Internet Things J. 4 (6) (2017) 2342–2350.

[6] X. Wang, X. Zha, W. Ni, R.P. Liu, Y.J. Guo, X. Niu, K. Zheng, Survey on blockchain for internet of things, Comput. Commun. 136 (2019) 10–29.

[7] H. Wang, T. Wang, L. Shi, N. Liu, S. Zhang, A blockchain-empowered framework for decentralized trust management in internet of battlefield things, Comput. Netw. (2023) 110048.

[8] X. Wang, P. Yu, G. Yu, X. Zha, W. Ni, R.P. Liu, Y.J. Guo, A high-performance hybrid blockchain system for traceable IoT applications, in: J.K. Liu, X. Huang (Eds.), Network and System Security, Springer International Publishing, Cham, 2019, pp. 721–728.

[9] S. Mathur, A. Kalla, G. Gür, M.K. Bohra, M. Liyanage, A survey on role of blockchain for IoT: Applications and technical aspects, Comput. Netw. 227 (2023) 109726.

[10] G. Yu, X. Wang, K. Yu, W. Ni, J.A. Zhang, R.P. Liu, Survey: Sharding in blockchains, IEEE Access 8 (2020) 14155–14181.

[11] L. Luu, V. Narayanan, C. Zheng, K. Baweja, S. Gilbert, P. Saxena, A secure sharding protocol for open blockchains, in: Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, 2016, pp. 17–30.

[12] E. Kokoris-Kogias, P. Jovanovic, L. Gasser, N. Gailly, E. Syta, B. Ford, Omniledger: A secure, scale-out, decentralized ledger via sharding, in: 2018 IEEE Symposium on Security and Privacy, SP, IEEE, 2018, pp. 583–598.

[13] M. Zamani, M. Movahedi, M. Raykova, Rapidchain: Scaling blockchain via full sharding, in: Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, 2018, pp. 931–948.

[14] J. Wang, H. Wang, Monoxide: Scale out blockchains with asynchronous consensus zones., in: NSDI, vol. 2019, 2019, pp. 95–112.

[15] Z. Zhang, X. Wang, G. Yu, W. Ni, R.P. Liu, N. Georgalas, A. Reeves, A community detection-based blockchain sharding scheme, in: Blockchain–ICBC 2022: 5th International Conference, Held As Part of the Services Conference Federation, SCF 2022, Honolulu, HI, USA, December 10–14, 2022, Proceedings, Springer, 2022, pp. 78–91.

[16] Y. Zhang, S. Pan, J. Yu, Txallo: Dynamic transaction allocation in sharded blockchain systems, in: 2023 IEEE 39th International Conference on Data Engineering, ICDE, IEEE, 2023, pp. 721–733.

[17] J. Yun, Y. Goh, J.-M. Chung, Trust-based shard distribution scheme for fault-tolerant shard blockchain networks, IEEE Access 7 (2019) 135164–135175.

[18] P. Zhang, W. Guo, Z. Liu, M. Zhou, B. Huang, K. Sedraoui, Optimized blockchain sharding model based on node trust and allocation, IEEE Trans. Netw. Serv. Manag. (2023).

[19] G. Yu, X. Wang, W. Ni, Q. Lu, X. Xu, R.P. Liu, L. Zhu, Adaptive resource scheduling in permissionless sharded-blockchains: A decentralized multiagent deep reinforcement learning approach, IEEE Trans. Syst., Man, Cybern.: Syst. 53 (11) (2023) 7256–7268, http://dx.doi.org/10.1109/TSMC.2023.3296614.

[20] J. Yun, Y. Goh, J.-M. Chung, DQN-based optimization framework for secure sharded blockchain systems, IEEE Internet Things J. 8 (2) (2020) 708–722.

[21] Z. Yang, R. Yang, F.R. Yu, M. Li, Y. Zhang, 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. 9 (17) (2022) 16494–16509.

[22] M. Liu, F.R. Yu, Y. Teng, V.C. Leung, M. Song, Performance optimization for blockchain-enabled industrial internet of things (IIoT) systems: A deep reinforcement learning approach, IEEE Trans. Ind. Inform. 15 (6) (2019) 3559–3570.

[23] C.H. Liu, Q. Lin, S. Wen, Blockchain-enabled data collection and sharing for industrial IoT with deep reinforcement learning, IEEE Trans. Ind. Inform. 15 (6) (2018) 3516–3526.

[24] C. Qiu, H. Yao, F.R. Yu, C. Jiang, S. Guo, A service-oriented permissioned blockchain for the internet of things, IEEE Trans. Serv. Comput. 13 (2) (2019) 203–215.

[25] C. Qiu, F.R. Yu, H. Yao, C. Jiang, F. Xu, C. Zhao, Blockchain-based software-defined industrial internet of things: A dueling deep Q-learning approach, IEEE Internet Things J. 6 (3) (2018) 4627–4639.

[26] V. Mnih, K. Kavukcuoglu, D. Silver, A.A. Rusu, J. Veness, M.G. Bellemare, A. Graves, M. Riedmiller, A.K. Fidjeland, G. Ostrovski, et al., Human-level control through deep reinforcement learning, Nature 518 (7540) (2015) 529–533.

[27] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, O. Klimov, Proximal policy optimization algorithms, 2017, arXiv preprint arXiv:1707.06347.

[28] X. Wang, G. Yu, R.P. Liu, J. Zhang, Q. Wu, S.W. Su, Y. He, Z. Zhang, L. Yu, T. Liu, W. Zhang, P. Loneragan, E. Dutkiewicz, E. Poole, N. Paton, Blockchain-enabled fish provenance and quality tracking system, IEEE Internet Things J. 9 (11) (2022) 8130–8142, http://dx.doi.org/10.1109/JIOT.2021.3109313.

[29] G. Yu, L. Zhang, X. Wang, K. Yu, W. Ni, J.A. Zhang, R.P. Liu, A novel dual-blockchained structure for contract-theoretic lora-based information systems, Inf. Process. Manage. 58 (3) (2021) 102492, http://dx.doi.org/10.1016/j.ipm.2021.102492, URL https://www.sciencedirect.com/science/article/pii/S0306457321000030.

[30] E. Fynn, F. Pedone, Challenges and pitfalls of partitioning blockchains, in: 2018 48th Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshops, (DSN-W), IEEE, 2018, pp. 128–133.

[31] B.W. Kernighan, S. Lin, An efficient heuristic procedure for partitioning graphs, Bell Syst. Tech. J. 49 (2) (1970) 291–307.

[32] G. Karypis, V. Kumar, A fast and high quality multilevel scheme for partitioning irregular graphs, SIAM J. Sci. Comput. 20 (1) (1998) 359–392.

[33] Z. Zhang, Y. Wang, G. Yu, X. Wang, M. Zhang, W. Ni, R.P. Liu, A community-based strategy for blockchain sharding: Enabling more budget-friendly transactions, in: 2023 IEEE International Conference on Blockchain (Blockchain), IEEE, 2023, pp. 370–376.

[34] C. Huang, Z. Wang, H. Chen, Q. Hu, Q. Zhang, W. Wang, X. Guan, Repchain: A reputation-based secure, fast, and high incentive blockchain system via sharding, IEEE Internet Things J. 8 (6) (2020) 4291–4304.

[35] A. Clement, E. Wong, L. Alvisi, M. Dahlin, M. Marchetti, et al., Making Byzantine fault tolerant systems tolerate Byzantine faults, in: Proceedings of the 6th USENIX Symposium on Networked Systems Design and Implementation, The USENIX Association, 2009.

[36] M. Castro, B. Liskov, et al., Practical byzantine fault tolerance, in: OsDI, vol. 99, (1999) 1999, pp. 173–186.

[37] R. Han, J. Yu, R. Zhang, Analysing and improving shard allocation protocols for sharded blockchains, in: Proceedings of the 4th ACM Conference on Advances in Financial Technologies, 2022, pp. 198–216.

[38] M. Li, Y. Qin, Scaling the blockchain-based access control framework for IoT via sharding, in: ICC 2021 - IEEE International Conference on Communications, 2021, pp. 1–6, http://dx.doi.org/10.1109/ICC42927.2021.9500403.

[39] S. Nakamoto, Bitcoin: A peer-to-peer electronic cash system, Decentralized Bus. Rev. (2008) 21260.

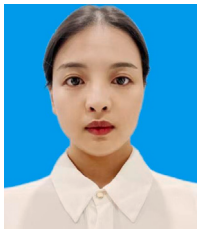
[40] A. Miller, I. Bentov, S. Bakshi, R. Kumaresan, P. McCorry, Sprites and state channels: Payment networks that go faster than lightning, in: International Conference on Financial Cryptography and Data Security, Springer, 2019, pp. 508–526.

Zixu Zhang received the A.A. degree from Santa Monica College, USA, in 2018. He received the M.Sc. degree from the University of Technology Sydney, Australia, in 2020. He is currently pursuing the Ph.D. degree with the Global Big Data Technologies Centre, Faculty of Engineering and Information Technology, University of Technology, Sydney. His main research interests include sharding blockchain, applying blockchain to the IoT, and deep reinforcement learning.

Guangsheng Yu is currently a Post-Doctoral Research Fellow with CSIRO Data61. He received the Ph.D. degree in 2021 with the Faculty of Engineering and Information Technology, University of Technology, Sydney, Australia. He received the B.Sc. degree and M.Sc degree from the University of New South Wales, Sydney, Australia, from 2011 to 2015. His main research interests lie in security and privacy of distributed and intelligent systems including blockchain and federated learning.

Caijun Sun received the Ph.D. degree from Beijing University of Posts and Telecommunications, Beijing, China in 2020. He received his B.E degree from Hangzhou Normal University, Hangzhou, China in 2013. He is currently a senior security engineer with Zhejiang Lab, China. His research interests include malware analysis and data security.

Xu Wang (Member IEEE) received his B.E. from Beijing Information Science and Technology University, China, in 2010, and dual Ph.D. degrees from Beijing University of Posts and Telecommunications, China, in 2019 and University of Technology Sydney, Australia, in 2020. He is currently a Senior Lecturer with the School of Electrical and Data Engineering, University of Technology Sydney, Australia. His research interests include cybersecurity, blockchain, privacy, and network dynamics.



Ying Wang received the PhD. degree in control science and engineering from Northeastern University in 2023, and now is a postdoc at Hangzhou Dianzi University, Hangzhou, China. Her current research interests focus on terahertz communication, data processing and information security.

Ming Zhang received the BTech degree in information security from the School of Cyber Engineering, Xidian University in 2019, and now is a Ph.D. student at the School of Cyber Engineering, Xidian University, Xi'an, China. His current research interests focus on blockchain security, privacy-preserving, and network security. He has a publication in conferences such as INFOCOM workshop MobiSec.

Wei Ni received the B.E. and Ph.D. degrees in Communication Science and Engineering from Fudan University, Shanghai, China, in 2000 and 2005, respectively. Currently, he is a Principal Research Scientist at CSIRO, Sydney, Australia, a Conjoint Professor at the University of New South Wales, an Adjunct Professor at the University of Technology Sydney, and an Honorary Professor at Macquarie University. He was a Postdoctoral Research Fellow at Shanghai Jiaotong University from 2005 to 2008; Deputy Project Manager at the Bell Labs, Alcatel/Alcatel-Lucent from 2005 to 2008; and Senior Researcher at Devices R&D, Nokia from 2008 to 2009. He has authored seven book chapters, more than 300 journal papers, 120 conference papers, 26 patents, and 13 standard proposals accepted by IEEE and ISO. His research interests include machine learning, online learning, stochastic optimization, and their applications to system efficiency and integrity. He is a Fellow of IEEE.

Dr Ni has served as an Editor of IEEE Transactions on Wireless Communications since 2018, an Editor of IEEE Transactions on Vehicular Technology since 2022, and an Editor of IEEE Communication Surveys and Tutorials and IEEE Transactions on Information Forensics and Security since 2024. He served first as the Secretary, Vice-Chair and then Chair of IEEE Vehicular Technology Society New South Wales Chapter from 2015 to 2022, Track Chair for VTC-Spring 2017, Track Co-chair for IEEE VTC-Spring 2016, Publication Chair fo

Ren Ping Liu (M'09-SM'14) received his B.E. degree from Beijing University of Posts and Telecommunications, China, and the Ph.D. degree from the University of Newcastle, Australia, in 1985 and 1996, respectively.

Ren Ping Liu is a Professor and Head of Discipline of Network & Cybersecurity at University of Technology Sydney (UTS). As a research leader, a certified network professional, and a full stack web developer, he has delivered networking and cybersecurity solutions to government agencies and industry customers. His research interests include wireless networking, 5G, IoT, Vehicular Networks, 6G, Cybersecurity, and Blockchain. He has supervised over 30 PhD students, and has over 200 research publications.

Professor Liu was the winner of NSW iAwards 2020 for leading the BeFAQT (Blockchain enabled Fish provenance And Quality Tracking) project. He was awarded the Australian Engineering Innovation Award 2012 and CSIRO Chairman's medal for his contribution in the Wireless Backhaul project. Professor Liu was the founding chair of IEEE NSW VTS Chapter and a Senior Member of IEEE.

Andrew Reeves leads BT's IoT Research team within its Research and Network Strategy organization. His interests include applying emerging Internet of Things (IoT) technologies for the benefit of individuals and organizations. The work is conducted in collaboration with industry and academic partners and covers topics such as edge compute, analytics, blockchain, orchestration and service management.

Andrew has worked various aspects of IoT technologies at BT for over 20 years, holding a number of patents and publications in this area. Prior to joining BT, Andrew completed a Ph.D. in Physics with the Condensed Matter Theory Group at Durham University.

Nektarios Georgalas is a seasoned technology leader at British Telecom, currently serving as Manager for Innovation and Solutions Architecture, driving the implementation of Autonomous IoT ecosystems through AI, machine learning, and advanced analytics. With 26 years at BT, Nektarios has served as Senior and Principal Researcher in the BT Research department and currently leads the IoT program at BT Ireland Innovation Centre (BTIIC). He pioneers AI-driven Autonomous IoT ecosystems through collaboration with universities, BT teams, partners, and customers. He is been pivotal in our co-innovation programs, focusing on cloud services, security, smart cities, and IoT. He is recognized with 20 awards, including TeleManagement Forum's Excellence Award for Innovation, Global Telecoms Business's Business Service Innovation Award, and several IEEE Outstanding Leadership and Service Awards. Nektarios is the inventor and co-inventor of 16 patents, has published over 90 papers, chaired 7 IEEE conferences, and co-edited 7 conference proceedings.