Shard-based and Reputation-enhanced Byzantine Fault-tolerant Scheme for Secure Data Sharing in Latency-sensitive Computing Services

Samuel D. Okegbile, *Member, IEEE,* Jun Cai, *Senior Member, IEEE,* Jiayuan Chen, and Changyan Yi, *Member, IEEE*

***Abstract*—This paper presents a blockchain-enabled data sharing framework for latency-sensitive computing services. To ensure an improved throughput while reducing the overall latency during data sharing processes, we propose a parallel validation-based reputation-enhanced practical Byzantine fault tolerance consensus framework with a priority-based block appending process to avoid forking attacks. The proposed framework allows multiple simultaneous validation processes thereby improving the overall performance of the data sharing system while ensuring that important requirements of the blockchain-enabled framework such as security and decentralization are not compromised. Furthermore, we formulate the communication process among validators and their computation resource allocation as a Markov decision process to optimize the transaction throughput while reducing the overall latency. We then adopt the branching dueling Q-network approach to address the large dimensions action space issue in our formulated problem and obtain simulation results to evaluate the performance of the proposed framework. The results show that the proposed framework can significantly improve the performance of blockchain-enabled data sharing in mobile and computing services by ensuring higher throughput and lower latency while maintaining an acceptable level of security among the validating nodes.**

***Index Terms*—Blockchain, branching dueling Q-network, data sharing, parallel validation, reliability.**

I. Introduction

BLOCKCHAIN continues to receive a lot of attention in various data sharing applications and systems owing to its ability to ensure security, transparency as well as data integrity. The technology has earlier been proposed as a distributed ledger of the Bitcoin system to prevent the problem of double-spending in cryptocurrency [1]. Since its success in cryptocurrency applications, many researchers have focused on the adoption of blockchain technology in various large-scale computing systems and services including internet-of-things (IoT), edge and cloud computing. When adopted, blockchain can allow trusts to be established among untrusted parties in a decentralized manner. This decentralized architecture means each node in the blockchain system contains a replica of the cryptographically and tamper-proof chained blocks, containing various transaction records, as agreed during the consensus process [2]. As a result, blockchain can guarantee secure and privacy-preserving data sharing among untrusted nodes.

Despite its ability to improve security in data sharing systems, blockchain suffers from many limitations including high latency, low transactions per second (TPS) rates, and scalability issues. Latency in blockchain-enabled systems can increase significantly with an increase in data size and the number of users/consensus nodes since such will increase the overall processing time due to the complicated validation process. Similarly, scalability issues can arise as the ledger size increases [3]. These have led to various studies on the suitability of blockchain, especially in latency-sensitive services [4]. To facilitate the adoption of blockchain in latency-sensitive systems, there is a need to re-define the blockchain-enabled data sharing framework to ensure that the existing latency, scalability, and throughput issues are well resolved.

Furthermore, the recent advances in 5G mobile communications and current research towards the development of 6G wireless systems reveal that secure, ultra-reliable, low latency communications and processing are essential requirements of future systems ranging from autonomous systems to extended reality applications. Some of these applications, for instance, autonomous vehicles and enhanced health applications, rely on efficient and effective data sharing schemes. Thus, a reliable low latency and high throughput performance are required for future blockchain applications and systems to ensure that the huge volumes of transactions from massive IoT devices are properly handled. To achieve this, an effective blockchain-enabled data sharing framework must be developed to meet the requirement of these latency-sensitive applications.

Existing efforts have adopted consensus algorithms such as proof-of-stake (PoS) [5] and practical Byzantine fault tolerance (PBFT) scheme [6], as opposed to proof-of-work (PoW) consensus algorithm, to reduce the latency and improve system performance. Similarly, the TPS scaling method [3] is often adopted by adjusting various blockchain parameters such as block size, block interval, and block producer. A delegated PoS [7] is another consensus protocol that has been proposed to reduce the consensus latency by reducing the number of consensus nodes, although such methods suffer from security and reliability threats. When compared with the proof-based consensus protocol, the BFT-based algorithms provide deterministic execution results, while achieving relatively high performance. This makes such protocols suitable in

S. D. Okegbile and J. Cai are with the Network Intelligence and Innovation Lab (NI^2L), Department of Electrical and Computer Engineering, Concordia University, Montreal QC H3G 1M8, Canada (Emails: samuel.okegbile@concordia.ca; jun.cai@concordia.ca).

J. Chen and C. Yi are with the College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing, Jiangsu, 211106, China (Emails: jiayuan.chen@nuaa.edu.cn; changyan.yi@nuaa.edu.cn).

permissioned blockchains. The BFT-based consensus protocol eliminates the limitations of the proof-based consensus protocol by exchanging data among a group of validators called replicas and can achieve a lower transaction latency than the PoW scheme [8]. The frequent message transmissions among different validators during the consensus process, however, mean the communication cost can make such a protocol difficult to be applied in large-scale networks [8]. Thus, a careful performance evaluation is necessary. To date, only a limited number of works have focused on the analysis and management of validation latency in BFT-based blockchain systems owing to its difficulty [9].

Sharding technique can eliminate the scalability issue in PBFT-based blockchain systems by parallelizing transaction processing thereby maximizing the overall throughput in proportion to the number of shards [10]. While such a technique can suffer from single shard takeover as the number of shards increases [3], its capability to improve the system performance when carefully adopted means the sharding technique is a very useful method. Indeed, the well-discussed trilemma of blockchain systems generally believes that any blockchain framework can only satisfy at most two of the three features: decentralization, security and scalability. Thus, a trade-off relationship often exists among these features since maximizing one feature can degrade the others. Finding an optimal scalability point without compromising security and decentralization, therefore, remains an open issue.

This paper thus presents a blockchain-enabled data sharing (sBeDS) framework for latency-sensitive computing services. To address the issues of scalability, latency and throughput, while ensuring that the decentralization and security features are not compromised, we propose a shard-based reputation-enhanced PBFT consensus framework with a priority-based block appending process to avoid forking attacks [11], [12]. To the best of our knowledge, such an approach has not been considered in any existing study. The contributions of this paper are summarized as follows:

- We present a sBeDS framework for large-scale multi-user networks with parallel offloading and validation processes to improve the system throughput while reducing latency. To ensure security, we integrate trust-based proof of reputation and PBFT consensus techniques.
- We then propose a priority-based block appending process using the prioritized queuing theory to prevent potential forking attacks in the proposed sBeDS framework. We obtain analysis for relevant metrics of interest.
- Next, we formulate the transaction offloading and computation resource allocation problem as a Markov decision process (MDP) to allow optimization of transaction throughput while reducing both communication and computation latency.
- By adopting the branching dueling Q-network (BDQ) approach, we provide solutions that are capable of compensating for the large dimensions of action space to the optimization problem.
- Finally, we carry out numerical evaluations and simulations to investigate the performance of the proposed sBeDS framework.

TABLE I
COMMON NOTATIONS USED

Notation	Definition
B	Maximum block capacity limit
t_{int}	Time interval
a_{tran}	Number of arrived transactions during any time interval
$P_{a,b}$	Offloading power of node a when offloading to node b
$h_{a,b}$	Channel gain between any nodes a and b
W; σ^2	Bandwidth; Noise signal power
χ; S^B	Average transaction size; Block size
N; $N_{s,k}$	Total number of validators; Number of validators in shard k
K	Number of shards
f	Total number of possible faulty or malicious validators
f_k	Number of possible faulty or malicious validators in shard k
$con_{v_i^k,v_j^k}$	Number of consistent responses of validator v_j^k at validator v_i^k
$incon_{v_i^k,v_j^k}$	Number of inconsistent responses of v_j^k at v_i^k
d_{th}; $R_{d_{th}}$	Pre-defined reputation threshold; Data rate threshold

The remainder of this paper is organized as follows. In Section II, we discuss various related studies. The details of the proposed sBeDS framework is presented in Section III. Section IV introduces the details of the reputation-enabled PBFT consensus protocol, while Section V conducts the analysis of the relevant metrics of interest. In Section VI, the corresponding resource optimization problem is formulated and Section VII shows the simulation results. Finally, Section VIII concludes the paper. Common notations used in this paper are presented in Table I.

II. RELATED WORK

In this section, we present some relevant literature by reviewing some related work. We categorized these works into four aspects: performance issues in data sharing, reputations-enhanced consensus schemes, performance optimization techniques and parallel validation methods in the blockchain.

A. Performance issues in data sharing

Improving system performances in large-scale blockchain-enabled data sharing systems has recently been the subject of much research with a focus on reducing the overall latency while maximizing the throughput. In [13], [14], the authors focus on maximizing the throughput by studying the relationship between communication and blockchain in a spatiotemporal domain. The maximization of transaction throughput was investigated in [15], [16] to guarantee the decentralization, latency, and security of the blockchain system. Since consensus latency is a significant component of any blockchain-enabled systems performance, the authors in [17] carried out joint modeling of transmission latency and consensus latency. Generally, there are two types of consensus protocols in the blockchain: proof-based and BFT-based consensus, with BFT-based consensus protocol sometimes preferred in large-scale systems owing to its better performance [9].

The PBFT consensus protocol was adopted to demonstrate its suitability in the internet of vehicle networks [15], industrial IoT systems [16] and IoT [18]. Similarly a BFT decentralized federated learning method with privacy-preservation for

autonomous vehicles was also presented in [19]. The fault-tolerance of the BFT was evaluated in [20] under different network settings in terms of throughput and latency, while the performance of the PBFT was improved in [21] through an Eigen trust-based PBFT to ensure only high quality of nodes in the network are selected to construct a consensus group. The authors in [22], [23], carried out performance modeling of BFT schemes to minimize the consensus latency through a multi-block approach [22] and multi-core processors [23]. It is worth noting that none of these works [13]–[23] considered parallel validation, while only [15], [16], [18] considered both the latency experienced during the validation and message exchange periods in their analyses.

B. Reputations-enhanced consensus schemes

A consensus mechanism is a core component of any blockchain framework. Any set of nodes selected to participate in the validation process can influence the performance of such systems since this set of distrusting nodes must reach a consensus to append any block to the blockchain. As demonstrated in [21], trust-based analysis can improve the performance of any blockchain-enabled data sharing scheme. Such solutions can ensure that only trusted nodes or nodes with high historical reputations participate in the consensus process. It is thus unsurprising that many recent works continue to integrate trust-based reputation schemes into the blockchain framework. A trust evaluation mechanism (ATEM) was proposed in [12] to evaluate the trustworthiness of nodes in the blockchain, while an optimized PBFT (T-PBFT) algorithm was proposed in [21] to improve consensus efficiency.

Furthermore, a trust-enabled blockchain (TeB) framework was proposed in [24], where only nodes with high trust values participate in the relaying and consensus tasks. A reputation-based voting scheme (BIoV) was also presented in [25], where nodes with high reputation were selected as miners. Similarly, an attack-resistant trust model based on multidimensional trust metrics (ARTMM) was proposed in [26] to reduce unreliable underwater communications. One key missing component of most existing trust and reputation-based schemes is the need to capture the link quality among nodes when estimating the reputation or trust value of each node since inconsistent contributions are not always a result of malicious intentions but may be a result of poor link quality. This is essential to the proposed sBeDS framework.

C. Performance optimization techniques

To further improve the system performance, optimization techniques, such as deep reinforcement learning (DRL), have lately been exploited in blockchain-enabled data sharing systems due to its ability to optimize the resources with low complexity in complicated networks. The work in [15] considered a DRL-based performance optimization framework for blockchain-enabled internet of vehicles, where the transaction throughput was maximized. Because of the unstable network connection, the reliability of the data is not always guaranteed since unstable connections can increase the tendency of data loss, eavesdropping and data manipulation. Thus, the data sharing process can lack security and scalability. As a result, the work in [27] integrated the multi-access edge computing and blockchain technologies to the internet of autonomous vehicles to optimize the blockchain system transaction throughput, while reducing the latency of the mobile edge computing (MEC) system. The joint optimization problem was modeled as a MDP using DRL.

In similar works, an advantage actor-critic algorithm was used to solve the DRL-enabled optimization problem in [28], where a delegated Byzantine fault tolerance consensus mechanism was adopted with blockchain nodes deployed on the edge servers. A new BDQ approach was proposed in [29] to enable the use of discrete-action algorithms in DRL for high-dimensional discrete or continuous action space domains and was adopted in [30] owing to its suitability. Furthermore, the performance of user sharing-based caching was improved in [31] through a blockchain-incentivized device-to-device and MEC caching system. The weighted sum of the computation rate and the transaction throughput was maximized in [24] by jointly optimizing the cooperative offloading decision and resource allocation. Parameters such as offloading decision, power allocation, block size, and block interval were jointly considered in the problem formulation. Generally, DRL-based optimization techniques can improve the performance of blockchain-enabled data sharing systems although the adoption of a single validation process means such a performance continues to be limited.

D. Parallel validation methods

Parallel validation methods can significantly improve the validation process thereby improving the overall system performance [32]–[34]. A parallel blockchain validation was first considered using the sharding technique in [3], where a deep Q network sharding-based blockchain scheme was proposed. In [33], shards were created by considering shard trust difference, communication delay difference and node count difference among shards with a focus on reducing the failure probability of blockchain. A clustering-based sharded blockchain strategy for collaborative computing in IoT networks was similarly presented in [35], while the work in [34] analyzed the security issues in sharding blockchain-based fog computing networks. Sharding technique was also adopted in [36], [37]. Under the sharding method, the blockchain validators were clustered into a different group of shards such that each shard independently creates and validates blocks through intra-shard consensus processes. In [3], [34]–[36], each validated block from each shard was merged and validated again by a final consensus process (following a double-layer consensus mechanism) before the new block was appended to the chain.

Although increasing the number of shards to increase the TPS can decrease the security level due to the possibility of a single shard takeover event, blockchain sharding techniques can improve blockchain scalability and throughput [3]. To ensure security, the existing shard-based technique often followed a double-layer consensus mechanism, which can increase the overall consensus latency since the inclusion of

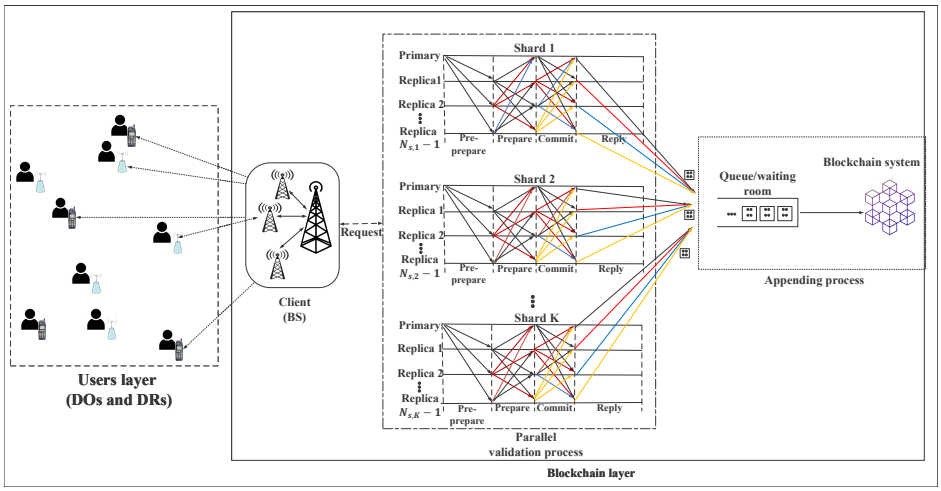

Fig. 1. Shard-based blockchain-enabled data sharing framework.

the final shard means the process can be reduced to a single shard problem. Instead, in this paper, we integrate PBFT and trust-based proof-of-reputation consensus mechanisms where a shard is only created if the security constraint is satisfied. With this, the security level is maintained as in the conventional PBFT scheme while the scalability, decentralization and throughput are improved.

III. System model

This section presents the network model, the general framework of the adopted PBFT consensus framework and the details of the reputation method. We define a shard as each partition in the blockchain system containing a group of validators with the ability to make distinctive and independent validation decisions when compared to other groups.

A. Network model

We consider a blockchain-enabled data sharing system, where users can be data owners (DOs) or data requesters (DRs) as shown in Fig. 1. The validation process is parallelized such that $K \geq 1$ number of validation processes take place simultaneously, where only one block is validated at any time in each shard. We consider the time to be discrete, where time is segmented into equal time slots $t = 1, ..., \infty$. Thus, arrivals and departures (i.e., completion of validation processes) of blocks occur within the time slot boundaries, i.e., the departure from the system can only occur in the interval (t^-, t), while the arrival can only occur in the interval (t, t^+).

Similar to [27], [28], we consider orthogonal spectrum for the transmission between users $U = \{u_1, u_2, ..., u_N\}$ and the group of base stations (BS), called clients, when users offload transactions to the blockchain layer for validation. Each block is generated immediately after the maximum block capacity limit B is reached. When the block maximum capacity is not reached within any time slot interval t_{int}, all arrived transactions at the client during the interval t_{int} are packaged into a single block. Thus, each block is made up of at least one transaction.

At the end of every interval t_{int}, where the number of arrived transactions $a_{tran} \geq 1$, the block is generated and forwarded to any randomly selected available shard. We considered the entire system to be stable, hence the block generation rate is less than the average joint validation rate. With this, there is always at least a shard available for each generated block. We adopted the PBFT consensus protocol. Thus, each block goes through each stage of the PBFT protocol. It becomes immediately clear that such a system may suffer from a forking attack – an attack that occurs when more than one shard attempt to append their validated blocks to the chain at the same time. To address this issue, we propose a priority-based block appending process, where shards are classified into different classes of priority based on the average reputation scores of validators within each shard, such that the block from the shard with the higher priority is appended first to the chain. More details about the shard-based validation process will be provided in the next sections.

Consider a tagged user u_0 and its associated client, located at the origin 0. The achievable data rate between the user u_0 and the associated client can be captured as

$$R_{u_0,0} = W \log_2 \left(1 + \frac{P_{u_0,0} h_{0,0}}{\sigma^2}\right), \tag{1}$$

where W is the bandwidth. Given that χ is the average size of each transaction, the average offloading time of any transaction is given as

$$T^{\text{off}}_{u_i,0} = \frac{\chi}{R_{u_i,0}}. \tag{2}$$

Considering any block generation phase in a given slot, let the first transaction arrive at the client at $t = t_0$ and let the Bth transaction arrive at $t = t_B$. The time to generate a block (TGB) is obtained as

$$TGB = \min\{t_B - t_0, t_{int}\}. \tag{3}$$

B. Shard formation

The shard formation is an essential part of the validation process and can be very complicated due to the need to maintain a high level of security in each shard. Since a high data rate is central to reduced validation latency and improved transactions throughput in any PBFT-based system, we considered a clustering-based shard formation process, where validators within the same coverage area [38] in the blockchain layer are fairly selected by the blockchain system to form a shard subject to PBFT consensus protocol constraints. This is necessary to prevent large communication overhead and high latency that often characterize the validation process in the BFT-based consensus process. A similar shard formation approach was adopted in [37], where geographically closer nodes form a shard and in [35], where the clustering process follows the K-means algorithm.

To ensure the reliability of each validator, we integrated trust-based proof of reputation and PBFT consensus protocols, where the reputation of each validator is taken into consideration within a shard. Validators with reputation values below a pre-defined acceptable threshold a_{th} are removed from the blockchain system and are not allowed to participate in the shard formation process, while the validator with the highest reputation value within a shard is selected as the primary such that within any shard k, a validator $v_i^k \in V_k$ is selected as the primary with the probability $p_{v_i^k} = \frac{D^{\text{trust}}_{v_i^k}}{\sum_{j=1}^{V_k} D^{\text{trust}}_{v_j^k}}$, where $D^{\text{trust}}_{v_*^k}$ is the reputation value of the validator v_*^k. In addition, any newly joined validator is assigned the base reputation value of a_{th}. The proposed shard formation process, therefore, relies on the geographic location, reputation and achievable data rate of each validator.

Let $N \geq 3f + 1$ represent the total number of validators available during any shard formation process at the blockchain layer, where f denotes the total number of possible faulty or malicious validators, while $N_{s,k} \leq N, (\forall k \in \{1, 2, ..., K\}$ and $N \geq 3f + 1)$, captures the number of validators in shard k. It is worth noting that, an increase in $N_{s,k}$ increases the security level (but reducing the number of shards, K), though reduces the decentralization and scalability degrees. It becomes immediately clear that if we reduce the number of possible faulty or malicious validators f_k within each shard k, we can minimize $N_{s,k}$, such that the security level remains acceptable (as in the single shard PBFT-based consensus scheme), while the decentralization, throughput and scalability levels increase. Thus, given N, it is desirable to

$$\begin{aligned} &\min_{v_i \in V} f \leq \frac{N-1}{3}, \\ \text{s.t.} \quad &f_k \leq \frac{N_{s,k}-1}{3}, \forall k \in \{1, 2, ..., K\}, \\ &\sum_k N_{s,k} \leq N, \\ &\sum_k f_k \leq f, \end{aligned} \tag{4}$$

where $v_i \in V$ represents each validator in the blockchain layer. The adopted reputation-based PBFT protocol ensures a reputation-centric validator selection process, where the blockchain system randomly allocates replicas to different shards based on their reputation values and locations. With that, the constraints in (4) can be achieved. To prevent adversary corruption attacks, the shard reconfiguration process [39] can be carried out after every PBFT view to ensure that the proportion of honest validators in each shard always satisfies the first constraint of (4). To further prevent whitewashing attacks, each validator $v_i \in V$ is required to solve a cryptographic puzzle during its registration on the system to obtain its unique identity $Id_{v_i^k}$ such that the cost of whitewashing outweighs its benefit. Thus, any validator v_i is generally recognized through its tuple $< Id_{v_i}, D^{\text{trust}}_{v_i} >$.

C. PBFT consensus protocol

The PBFT consensus process generally has five phases as shown in Fig. 1: REQUEST, PRE-PREPARE, PREPARE, COMMIT, and REPLY. During the REQUEST phase, the client forwards any newly generated block to a randomly selected available primary for validation. The selected primary then verifies the message authentication code (MAC) of each transaction in the received data block during the PRE-PREPARE phase. After the initial verification, the primary broadcasts the block to $N_{s,k} - 1$ replicas for validation.

In the PREPARE phase, each replica authenticates the received pre-prepare decision message and exchanges MACs with all other replicas within the corresponding shard to ensure that the same block was received from the primary. The validators then enter the COMMIT phase, where the block is validated. After validation, each validator sends its validation outcome to the client during the REPLY phase, where the block from each shard is appended to the chain if the consensus is reached among validators in such a shard and subject to the inter-shard priority class. Generally, a block moves from one phase to the next phase of the PBFT scheme if two-thirds of the responses from the participating nodes consent [3].

D. Reputation model

We adopt a trust-based reputation model where each validator $v_i^k \in V_k$ generates a reputation opinion or value about each validating pair $v_i^k, v_j^k \in V_k, \forall j \neq i$ within the same shard k after every validation process. If the received validation decision from v_j^k is consistent with the majority of the received decisions, the validator v_i^k updates its direct consistent value $con_{v_i^k, v_j^k}$ of validator v_j^k, otherwise, it updates its direct inconsistent reputation value, $incon_{v_i^k, v_j^k}$. These values are continually aggregated and securely stored in the blockchain system and are used during the replicas selection process. The reputation value of each validator is calculated using both direct and indirect trust values as will be discussed in Section IV. Generally, the direct trust value of any validator v_i^k for another validator v_j^k is defined as trust values obtained through previous direct transactions between the pair v_i^k and v_j^k, while indirect trust values of validator v_i^k for any validator v_l^k is based on the recommendation of another validator (for instance v_j^k) based on the previous transactions between v_j^k and v_l^k.

TABLE II
COMPARISON WITH EXISTING REPUTATION-BASED SCHEMES

Reputation-based Schemes	T-PBFT	ATEM	TeB	BIoV	ARTMM	sBeDS
Historical Reputation	✓	✓	✓	✓	✓	✓
Direct Trust	✓	X	✓	✓	✓	✓
Indirect Trust	✓	X	✓	✓	✓	✓
Recommendation Reliability	X	✓	✓	X	✓	✓
Consensus Protocol	BFT-based	Proof-based	BFT-based	Proof-based	–	BFT-based
Considered Link Quality	X	X	✓	X	✓	✓
Considered relationship between Reputation and f	X	X	X	X	X	✓

Note that the reputation score of each validator is obtained through the accumulation of its previous transactions. Hence, the validator with a high reputation score has a high behaviour consistency and thus high trustworthiness as in [12], [21], [24]–[26]. With this, we know that f_k depends on the reputations of selected validators within any shard k. It is worth mentioning that, these reputation opinions are not only based on the actual intentions of the participating validators but are also influenced by link quality [26]. As a result, a clustering-based shard formation technique is adopted, such that the distance between nodes within the same shard is limited, thus reducing the effect of bad link quality. To prevent misrepresentation of trust values (e.g., v_i^k generating a wrong recommendation of v_j^k to mislead other validators), we integrate recommendation reliability into the sBeDS framework to ensure validators with inconsistent recommendations are always detected and penalized. The proposed trust-based proof of reputation scheme eliminates the limitations of the existing approaches as presented in Table II.

E. Association rule model

In this subsection, we model the transaction-block, block-shard and validator-shard association rules for more clarification and details. We adopted a tuple $G(S, B_l, V, K)$ to describe the presented shard-based blockchain-enabled data sharing framework, where S is the set of transactions and B_l is the set of blocks. We can use the weight matrix $SB = [sb_{ij}]$ to represent the association relations between transactions and blocks, where $sb_{ij} = 1$ indicates that a transaction s_i is packaged into a block b_j and $sb_{ij} = 0$ if otherwise. Thus, a transaction can only be packaged into only one block, such that $\sum_j sb_{ij} = 1$. The transaction-block association matrix is generally of the form

$$\begin{bmatrix} sb_{11} & sb_{12} & \dots & sb_{1B} \\ sb_{21} & sb_{22} & \dots & sb_{2B} \\ \vdots & \vdots & \dots & \vdots \\ sb_{S1} & sb_{S2} & \dots & sb_{SB} \end{bmatrix}. \tag{5}$$

Similarly, a block b_i can only be validated in a single shard k_j, such that the weight matrix $BK = [bk_{ij}]$, with $\sum_j bk_{ij} = 1$, while a validator v_i can only belong to one shard k_j at any observation time, with the weight matrix $VK = [vk_{ij}]$ and $\sum_j vk_{ij} = 1$. Thus, the block-shard and validator-shard association matrices follow the same form as in (5).

One crucial consideration in the shard-based blockchain is cross-shard transaction processing. This includes transactions that require more than one shard for processing, making the need to minimize the cross-shard validation overhead essential. By adopting eventual atomicity [40], each transaction that requires cross-shard processing can be split into multiple atomic transactions by the client. Each of these atomic transactions relates to different shards and is processed in parallel at different shards. The proposed sBeDS scheme guarantees the ACID property – atomicity, consistency, isolation, and durability – of transactions by ensuring (i) atomicity: a transaction is only sent for block appending process if it has been successfully validated through the five stages of PBFT; (ii) consistency: each block of transactions remains unchanged throughout the validation process; (iii) isolation: the conditions $\sum_j sb_{ij} = 1$, $\sum_j bk_{ij}$ and $\sum_j vk_{ij}$ are always true; and (iv) durability: blockchain is immutable, thus each validated block is irreversible.

IV. REPUTATION-ENABLED PBFT CONSENSUS FRAMEWORK

In this section, we present the details of the adopted reputation-enabled PBFT consensus algorithm focusing on the analysis of direct and indirect trust values. This will be integrated into the analyses of some key performance metrics presented in Section V.

A. Integrated trust-based reputation and PBFT scheme

To characterize the proposed reputation-enabled PBFT consensus algorithm-based data sharing framework, let $N \in \{1, 2, ..., \mathcal{N}\}$ represents the number of nodes available as validators at any time slot t such that the set of available validators is given as $\{v_1, v_2, ..., v_N\} \in V$. From (4), the number of validators in each shard $N_{s,k}$ satisfies the PBFT constraints

$$N_{s,k} \geq 3f_k + 1, \forall \sum_k N_{s,k} \leq N, k \in K. \tag{6}$$

To minimize the number of malicious (or failed) validators, we adopted a trust-based reputation scheme for the PBFT-based validation process. Given that $d_{th} > a_{th}$ is the predefined reputation threshold, each validator v_i^k with reputation values $D^{\text{trust}}_{B,v_i^k} < d_{th}$, as evaluated by the blockchain system, is tagged as malicious (or node with a high probability of failure), while the validator with the reputation value, $\max_{v_i^k \in V_k}\{D^{\text{trust}}_{B,v_i^k}\}$, within any shard k at any observation slot is selected as the primary $v_p^k \in V_k$, as presented in Algorithm 1. The blockchain system continuously compares

the reporting trust values of each node and removes nodes with low reputation values. The updated aggregated trust values of validators $D^{\text{trust}}_{B,v_i^k}$ are stored in the blockchain to facilitate the removal of validators with low reputation values. With this, the blockchain system can estimate the reliability of received indirect trust values (i.e., recommendations) from each node and penalize nodes with malicious recommendations.

During each stage of the PBFT, each tagged validator $v_j^k \in V_k$ develops a direct trust value for every other validator $v_i^k \in V_k, (\forall j \neq i, V_k \in V)$ within the same shard k. These trust values are forwarded to the trust aggregation server located in the blockchain system at the end of each observation phase via dedicated error-free communication links. Thus, the blockchain system maintains a continuously updated and aggregated indirect trust value for each validator $v_i \in V$ based on the direct trust values received from other validating nodes. To consider the possible influence of bad link quality, the replica selection process should consider both the reputation value and the achievable data rate of each validator within the coverage region of the selected primary. Hence, we can allow the transmission data rate R_{0,v_i^k} between the center of any shard k and each validator $v_i^k \in V_k, \forall p \neq i$ to depend on the overall reputation value $D^{\text{trust}}_{v_p,v_i}$ of each validator $v_i^k \in V_k$. With this, the primary can select any validator v_i^k at time t, through the help of blockchain system, if the $R_{0,v_i^k}(t) \geq R_{d_{th}}$, as presented in Algorithm 2, where $R_{d_{th}}$ is the data rate threshold obtained from d_{th}.

Algorithm 1: Primary election process

```
function ShardFormation
Require: D^trust_{B,v_i^k}, ∀v_i^k ∈ V_k
  PrimaryNode ← empty
  HighestReputationNode ← 0
  for each validator v_i^k ∈ V_k: do
    if D^trust_{B,v_i^k} ≥ HighestReputationNode: then
      HighestReputationNode ← v_i^k
    end if
    PrimaryNode ← HighestReputationNode
    primary ← PrimaryNode
  end for
  Return (primary)
```

Between any two nodes v_i^k and v_j^k, let $D^{\text{trust}}_{v_i^k,v_j^k} \in [0,1]$ denote the trust value of the node v_j^k from node v_i^k. The data transmission rate of node v_j^k received at the node v_i^k can be obtained from (1) as

$$R_{v_i^k,v_j^k}(t) = WD^{\text{trust}}_{v_i^k,v_j^k}(t) \log_2\Big(1 + \frac{P_{v_i^k,v_j^k} h_{v_i^k,v_j^k}}{\sigma^2 + \sum_{v_l^k \in V_k \setminus v_j^k} P_{v_l^k,v_l^k} h_{v_i^k,v_l^k}}\Big). \tag{7}$$

From (7), the transmission time of any block of size S^B between any two validators $v_i^k, v_j^k \in V_k$ is thus given as

$$\varphi_{v_i^k,v_j^k} = \frac{S^B}{R_{v_i^k,v_j^k}}, \forall R_{v_i^k,v_j^k} \neq R_{v_j^k,v_i^k}. \tag{8}$$

Algorithm 2: Replicas selection method

```
Require: N_{s,k}, V_k
  ListofReplica_k ← empty
  for n < N_{s,k}: do
    TrustedRate ← R_{d_th}
    for each validator v_i^k ∈ V_k: do
      Compute R_{0,v_i^k} from D^trust_{v_p,v_i^k}
      if R_{0,v_i^k} >TrustedRate: then
        TrustedRate ← R_{0,v_i^k}
      end if
    end for
    Remove v_i^k from V_k
    Add v_i^k to ListofReplica_k
  end for
  SelectedReplicas ← ListofReplica_k
  Empty V_k
  V_k ← SelectedReplicas
  Return (SelectedReplicas; V_k)
```

Next, we present the analysis for direct and indirect trusts between any two nodes, which helps to obtain the analysis of $D^{\text{trust}}_{v_i^k,v_j^k}$. Generally, direct trust depicts an aggregated trust value of any validator v_j, obtained by another validator v_i through their previous direct transactions, with $i \neq j$. On the other hand, indirect trust is an aggregated trust value of any validator v_j that is obtained by validator v_i through opinions and recommendations of other validators v_k, $\forall k \neq i, j$.

B. Direct trust

The direct trust between any two validators v_i^k and v_j^k is obtained following the subjective logic framework, described as a tuple $\omega_{v_i^k,v_j^k} = \{b_{v_i^k,v_j^k}, d_{v_i^k,v_j^k}, v_{v_i^k,v_j^k}\}$, where $b_{v_i^k,v_j^k}$ and $d_{v_i^k,v_j^k}$ are the belief and disbelief respectively of node v_i^k for node v_j^k, while $v_{v_i^k,v_j^k}$ is the degree of uncertainty in the belief system. These parameters satisfy the constraints

$$\begin{aligned} b_{v_i^k,v_j^k}, d_{v_i^k,v_j^k}, v_{v_i^k,v_j^k} \in [0,1], \\ b_{v_i^k,v_j^k} + d_{v_i^k,v_j^k} + v_{v_i^k,v_j^k} = 1. \end{aligned} \tag{9}$$

Hence, the reliability level of a validator v_j^k from any validator v_i^k can be obtained as

$$RD_{v_i^k,v_j^k} = b_{v_i^k,v_j^k} + \varepsilon v_{v_i^k,v_j^k}, \tag{10}$$

where $0 \leq \varepsilon \leq 1$ captures the influence of the trust uncertainty. From Section III-D, we can define

$$\begin{aligned} b_{v_i^k,v_j^k} &= \frac{con_{v_i^k,v_j^k}(1 - v_{v_i^k,v_j^k})}{con_{v_i^k,v_j^k} + incon_{v_i^k,v_j^k}}, \\ d_{v_i^k,v_j^k} &= \frac{incon_{v_i^k,v_j^k}(1 - v_{v_i^k,v_j^k})}{con_{v_i^k,v_j^k} + incon_{v_i^k,v_j^k}}, \\ v_{v_i^k,v_j^k} &= 1 - Q_{v_i^k,v_j^k}. \end{aligned} \tag{11}$$

Note that $con_{v_i^k,v_j^k}$ and $incon_{v_i^k,v_j^k}$ represent the overall historical number of consistent and inconsistent responses received

from any v_j^k by any v_i^k, while $Q_{v_i^k,v_j^k}$ captures the quality of the validation method. Since the parameters $con_{v_i^k,v_j^k}$ and $incon_{v_i^k,v_j^k}$ are not only a result of malicious intentions or faulty nodes but are also influenced by the unreliable communication links, the transmission error can contribute to the trust values of each node. With this,

$$\begin{aligned} con_{v_i^k,v_j^k} &= con_{v_j^k} + p_{tras}(con_{v_j^k} + incon_{v_j^k}), \\ incon_{v_i^k,v_j^k} &= incon_{v_j^k} - p_{tras}(con_{v_j^k} + incon_{v_j^k}), \end{aligned} \tag{12}$$

where $con_{v_j^k}$ and $incon_{v_j^k}$ receptively represent the number of consistent and inconsistent validation decisions made by the validator v_j^k. The transmission error rate [26] is given as

$$p_{tras} = 1 - \frac{\sum_i^n \omega(i) \times \omega(i)}{\sum_i^n \omega(i)}, \tag{13}$$

where $\omega(i)$ is the weight of the historical link-state, with $\aleph = (\omega(1), \omega(2), ..., \omega(n))$ representing the historical link-state record and $\omega(i) = \frac{2i}{n(n+1)}$. The average aggregated direct trust $D^{dir}_{v_i^k,v_j^k}$ is thus obtained from $RD_{v_i^k,v_j^k}$ as

$$D^{dir}_{v_i^k,v_j^k} = \frac{\sum_n RD_{v_i^k,v_j^k}(n)}{n}. \tag{14}$$

C. Indirect trust

For the indirect trust values, any node (for instance, the primary) can obtain indirect trust values of other validators from the blockchain system or other neighbouring nodes. Similarly, the blockchain system can evaluate the reputations of some nodes through indirect trust. Since every validator forwards the trust values of their communicating pairs to the blockchain system after every observation period, the aggregated indirect trust value of each validator is always available to improve the validation decision at every observation time. To estimate the indirect trust value, suppose $D^{dir}_{v_i^k,v_l^k}$ and $D^{dir}_{v_j^k,v_l^k}$ are respectively the average aggregated direct trust values of nodes v_i^k and v_j^k about node v_l^k. Then, the collective trust values of nodes v_i^k and v_j^k about node v_l^k is given as

$$D^{v_i^k,v_j^k}_{v_l^k} = D^{dir}_{v_i^k,v_l^k} \bigoplus D^{dir}_{v_j^k,v_l^k} = (b^{v_i^k,v_j^k}_{v_l^k}, d^{v_i^k,v_j^k}_{v_l^k}, v^{v_i^k,v_j^k}_{v_l^k}), \tag{15}$$

where

$$\begin{cases} b^{v_i^k,v_j^k}_{v_l^k} = (b_{v_i^k,v_l^k} v_{v_j^k,v_l^k} + b_{v_j^k,v_l^k} v_{v_i^k,v_l^k})/q \\ d^{v_i^k,v_j^k}_{v_l^k} = (d_{v_i^k,v_l^k} v_{v_j^k,v_l^k} + d_{v_j^k,v_l^k} v_{v_i^k,v_l^k})/q \\ v^{v_i^k,v_j^k}_{v_l^k} = (v_{v_j^k,v_l^k} v_{v_i^k,v_l^k})/q \end{cases} \tag{16}$$

and

$$q = v_{v_j^k,v_l^k} + v_{v_i^k,v_l^k} - v_{v_j^k,v_l^k} v_{v_i^k,v_l^k}, \forall q \neq 0. \tag{17}$$

Similarly, given that $D^{dir}_{v_i^k,v_j^k}$ and $D^{dir}_{v_j^k,v_l^k}$ represent the direct trust values of validator v_i^k for validator v_j^k and validator v_j^k for validator v_l^k, respectively. We can obtain the indirect trust value of validator v_i^k for validator v_l^k as

$$D^{v_i^k,v_j^k}_{v_l^k} = D^{dir}_{v_i^k,v_j^k} \bigotimes D^{dir}_{v_j^k,v_l^k} = (b^{v_i^k,v_j^k}_{v_l^k}, d^{v_i^k,v_j^k}_{v_l^k}, v^{v_i^k,v_j^k}_{v_l^k}), \tag{18}$$

where

$$\begin{cases} b^{v_i^k,v_j^k}_{v_l^k} = b_{v_i^k,v_j^k} b_{v_j^k,v_l^k} \\ d^{v_i^k,v_j^k}_{v_l^k} = b_{v_i^k,v_j^k} d_{v_j^k,v_l^k} \\ v^{v_i^k,v_j^k}_{v_l^k} = d_{v_i^k,v_j^k} + v_{v_i^k,v_j^k} + b_{v_i^k,v_j^k} v_{v_j^k,v_l^k}. \end{cases} \tag{19}$$

The indirect trust value $D^{ind}_{v_i^k,v_l^k}$ is obtained as presented in (14). To investigate the reliability of recommendation in (18), let $D^{ave}_{v_l^k}$ represent the average value of all received recommendations for v_l^k. Then we can obtain the difference between $D^{v_i^k,v_j^k}_{v_l^k}$ and $D^{ave}_{v_l^k}$. The greater the difference, the lower the reliability of the recommendation received from any validator. Therefore, the recommendation reliability is given as

$$RelD^{v_i^k,v_j^k}_{v_l^k} = 1 - |D^{v_i^k,v_j^k}_{v_l^k} - D^{ave}_{v_l^k}|. \tag{20}$$

Blockchain continuously penalizes validators with inconsistent recommendations and may lead to removal from the system. Finally, the reputation is obtained as a function of both direct and indirect trusts, such that

$$D^{\text{trust}}_{v_i^k,v_j^k} = \omega_{dir} D^{dir}_{v_i^k,v_j^k} + \omega_{ind} D^{ind}_{v_i^k,v_j^k}, \tag{21}$$

where ω_{dir} and ω_{ind} are the weights of direct and indirect trust values respectively, given that $\omega_{dir} + \omega_{ind} = 1$. As an abuse of notation, let $D^{\text{trust}}_{v_i}$ represent the average aggregate reputation value of each validator $v_i \in V$, to ensure security, each validator with reputation value below $D^{\text{trust}}_{v_i} < a_{th}$ are removed from the system, such that f is defined as

$$f = \sum_{v_i=1}^{N} 1_{[a_{th} \leq D^{\text{trust}}_{v_i} < d_{th}]}, \tag{22}$$

where $1_{[.]}$ is an indicator function that is equal to 1 if [.] is true and 0 if otherwise. From this, the number of shards (or clusters), K, can be bounded as

$$K \leq \frac{N}{3(N - \sum_{v_i=1}^{N} 1_{[D^{\text{trust}}_{v_i} \geq d_{th}]}) + 1}. \tag{23}$$

From (23), it is clear that the possible number of shards is limited by the reputations of all available validators. Hence, a shard is only created if the security constraint is met.

V. Performance Analysis

Next, we present the analyses of some performance metrics of interest for the proposed sBeDS framework. We first obtain analysis for scalability before considering the block generation and consensus processes time.

A. Scalability

Scalability is an important metric when characterizing blockchain-enabled data sharing systems. It measures the number of transactions that can be processed per second. This transaction throughput can be improved by either increasing the block size S^B or reducing the block interval T^I, although an increase in the S^B or a decrease in T^I can impose stricter

constraints on consensus latency. Hence, the choice of appropriate method, as well as the adopted consensus algorithm, should be properly considered to obtain the trade-off between scalability and latency. The number of transactions that can be processed per second in the proposed sBeDS framework is given as

$$T_{thru}(S^B, T^I) = \frac{K\lfloor S^B/\chi \rfloor}{T^I}, \tag{24}$$

where the block interval T^I follows from (3) and represents the average time required to generate a new block. From (24), it can be observed that an increase in K can further increase the number of transactions per second, $T_{thru}(S^B, T^I)$. In a case where there is a c^k_{sh} proportion of cross-shard transactions in the kth shard, (24) is upper-bounded at

$$T_{thru}(S^B, T^I, c^k_{sh}) = \frac{K\lfloor S^B/\chi \rfloor}{T^I} - \frac{\lfloor S^B/\chi \rfloor}{2T^I} \sum_{k=1}^{K} c^k_{sh}. \tag{25}$$

In such a case, reducing c^k_{sh} increases the scalability.

B. Latency

Latency is an important metric in any blockchain-related analysis [41], [42] and can be measured as the time to finality, which is the time required to successfully append a block to the chain through any shard k. This is the same as the time until a transaction written in the blockchain is irreversible. This latency in the presented framework includes three main components: block generation time T^I, consensus time and block appending time. The time to finality can be obtained as

$$T_{TTF} = T^I + T^k_{con} + T^k_{app}, \tag{26}$$

where T^k_{con} and T^k_{app} are the consensus time and block appending time respectively for any shard k. The consensus time depends on the PBFT scheme and is obtained as

$$T^k_{con} = T^k_{deliv} + T^k_{val}, \tag{27}$$

where T^k_{deliv} and T^k_{val} are the message delivery and block validation time within any shard k respectively. To avoid unnecessary complication, we evaluated the validation time as a function of cryptographic operations computing cost [43] similar to [15], [16], where a block validation process includes signatures validation, generation and validation of MACs using ζ, η and η CPUs cycles respectively.

In the REQUEST stage, the client v_c sends a block validation request to any available primary $v_p, \forall p \neq c$, where only one MAC verification is performed. Each validation request contains one signature, which requires verification of each validator during any consensus process. During the PRE-PREPARE stage, the primary in any shard k processes a batch of M validation requests and forwards a single pre-prepare message to all replicas within shard k. In this case, the primary generates $N_{s,k} - 1$ MACs and each replica process one MAC for verifications. Each replica then authenticates the received pre-prepare message during the PREPARE stage by generating $N_{s,k} - 1$ MACs to every other replica within shard k including the primary, while verifying $N_{s,k} - 2$ MACs. Next, each validator carries out block validation in the COMMIT stage, where all validators within shard k including the primary sends and receives $N_{s,k} - 1$ commit messages, thereby generating and validating $N_{s,k} - 1$ MACs. Finally, each validator generates one MAC for each validation request to reply to the client during the REPLY stage. It becomes immediately clear that for each M validation request in any shard k, the primary processes $2M + 4(N_{s,k} - 1)$ MAC operations, while each replica processes $M + 4(N_{s,k} - 1)$ MAC operations. The validation time of a primary in any shard k is given as

$$O^k_{v_p} = \frac{M\zeta + [2M + 4(N_{s,k} + f_k - 1)]\eta}{c_{v_p}}, \tag{28}$$

where c_{v_p} is the computation capacity of the primary v_p. Similarly, the validation time of any replica is given as

$$O^k_{v_i} = \frac{M\zeta + [M + 4(N_{s,k} + f_k - 1)]\eta}{c_{v_i}}, \tag{29}$$

where c_{v_i} is the computation capacity of any validator v_i. The total validation time T^k_{val} can thus be obtained as

$$T^k_{val} = \frac{1}{M} \max_{v^k_i \in V_k} \{O^k_{v_p}, O^k_{v_i}\}. \tag{30}$$

Similarly, the message delivery time T^k_{deliv} depends on the total time to transmit a block from the client to the primary and the total message exchanging time during validation. From (8), the time to transmit a block from the client to the primary is given as

$$\varphi_{v_c, v^k_p} = \frac{MS^B}{R_{v_c, v^k_p}}. \tag{31}$$

From (31), we can obtain the message delivery time T^k_{deliv}, given that τ is the timeout, as

$$\begin{aligned} T^k_{deliv} &= \frac{1}{M}(T^k_{request} + T^k_{pre-prepare} + T^k_{prepare} + \\ &\qquad T^k_{commit} + T^k_{reply}) \\ &= \frac{1}{M}\Big(\min\{\varphi_{v_c, v^k_p}, \tau\} + \min\Big\{ \max_{v^k_i \neq v^k_p, v_c} \varphi_{v^k_p, v^k_i}, \tau \Big\} + \\ &\qquad \min\Big\{ \max_{v^k_i \neq v^k_j; v^k_i, v^k_j \neq v_c} \varphi_{v^k_i, v^k_j}, \tau \Big\} + \\ &\min\Big\{ \max_{v^k_i \neq v^k_j; v^k_i, v^k_j \neq v_c} \varphi_{v^k_i, v^k_j}, \tau \Big\} + \min\Big\{ \max_{v^k_i \neq v_c} \varphi_{v^k_i, v_c}, \tau \Big\} \Big). \end{aligned} \tag{32}$$

From (32), the consensus time T^k_{cons} is obtained. Note that only one block can be validated at the same time in a single shard. Hence, $M = 1$.

C. Block appending time

Since the validation process in each shard is independent of the validation process in other shards, two or more shards may complete the validation of a block at the same time slot. To ensure an efficient appending process, each shard is assigned a different priority such that the block appending process is based on the non-preemptive priority of each shard. Thus, the block appending process can avoid forking attacks. We modeled the block appending process as a Geo/G/1 queuing system with non-preemptive priority, where any shard k has non-preemptive priority over shard $k+m, 0 < m \leq K-1$, such

that the priority of blocks from shard $1 > 2 > K-1 > K$. The arrival of validated blocks from each shard, therefore, follows an independent Bernoulli process with a probability λ_k, while each validated block requires a general appending service with service probability μ_k. Under the considered stable condition and at any time slot,

$$\rho = \sum_{k=1}^{K} \frac{\lambda_k}{\mu_k} < 1. \tag{33}$$

Following the proposed priority-based block appending technique, the block from shard 1 is appended to the chain before the block from shard 2, while the block from shard 2 is appended to the chain before the block from shard 3, etc. As in real-life systems, the appending time of any block from shard 1 is not affected by the appending time of blocks from lower priority shards $k > 1$, while the appending time of any block from shard 2 is only affected by the appending time of blocks from higher priority shard 1. It follows that the appending time of blocks from any shard k is only affected by the appending time of blocks from higher priority shards $m > k$. Thus, the proposed block appending process can be captured for two special classes: higher priority class and lower priority class. The block appending time of a block from the highest priority shard $k = 1$ at any time slot can be calculated as

$$T^1_{app} = \frac{1}{2\mu_1} + \frac{\varpi_{\lambda_1}\frac{1}{\mu_1} + \lambda_1^2 \varpi_{b_1}}{2\lambda_1(1-\rho_1)} + \frac{\lambda_2\left(\varpi_{b_1} + \frac{1}{\mu_2}\left[\frac{1}{\mu_2} - 1\right]\right)}{2(1-\rho_1)}, \tag{34}$$

where ϖ_* is the variance of $*$, while $b_* = \frac{1}{\mu_*}$. Similarly, the block appending time of any block from any lower priority shard (say $k = 2$) can be obtained following

$$\begin{aligned} T^2_{app} = \frac{1}{2\mu_2} + \frac{\varpi_{\lambda_2}\frac{1}{\mu_2}}{2\lambda_2(1-\rho)} + \frac{\lambda_2 \varpi_{b_2}}{2(1-\rho)(1-\rho_h)} \\ + \frac{\varpi_{\lambda_h}\left(\frac{1}{\mu_h}\right)^2 + \lambda_1 \varpi_{b_h}}{2(1-\rho)(1-\rho_h)}. \end{aligned} \tag{35}$$

The parameters λ_h and μ_h in (35) captures the joint arrival and service probabilities of blocks from higher priority shards respectively. The proof of (34) and (35) follows from the analysis of the discrete-time single server queueing system provided in [44]. At any slot, the average appending time can thus be approximated as

$$T^{ave}_{app} = \frac{1}{K}\sum_{k=1}^{K} T^k_{app}. \tag{36}$$

To ensure that the latency requirements of the blockchain-enabled data sharing system are satisfied, the total latency should be within some consecutive block intervals ξ ($\xi > 1$), such that

$$T_{TTF} \leq \xi T^I, k = 1, ..., K. \tag{37}$$

VI. Performance optimization using DRL

In this section, we first formulate the proposed shard-based blockchain-enabled data sharing system problem as a MDP to allow optimization of transaction throughput and minimization of overall latency, while ensuring that the security constraints are satisfied as in conventional PBFT consensus protocol-based systems. By formulating the shard-based blockchain-enabled data sharing system as a discrete MDP, we can maximize the system reward such that the MDP is defined using the tuple $(\mathcal{S}^{(t)}, \mathcal{A}^{(t)}, \mathcal{P}^{(t)}, \mathcal{R}^{(t)})$, where $\mathcal{S}^{(t)}$ is the state space, $\mathcal{A}^{(t)}$ is the action space, $\mathcal{P}^{(t)}$ is the state transition probabilities and $\mathcal{R}^{(t)}$ is the reward function. Next, we introduce DRL to capture and address the dynamic nature of the proposed system. Note that the DRL framework can have two phases: (i) an offline deep neural network construction phase, where the action-value function can be approximated with corresponding states and actions, and (ii) an online dynamic deep Q learning phase, which is used for action selection, system control, and dynamic network updating. Later, we present the details of the adopted BDQ algorithm.

A. State space and transition probability

At any decision epoch t ($t \geq 1$), the state space $\mathcal{S}^{(t)}$ is defined as the union of data achievable rate $R = \{R_{i,j}\}$, average transaction size χ, computation capacity of validators c_v, and the reputation value $D^{\text{trust}}_{i,j}$ of validators. This can be represented as

$$\mathcal{S}^{(t)} = [R, \chi, c_v, D^{\text{trust}}_{i,j}]^{(t)}. \tag{38}$$

Note that state space in (38) is continuous, thus the probability of being in a certain state can be assumed to be zero. With this, the process transition from state $s^{(t)}$ to the next state $s^{(t+1)}$ through the action $a^{(t)} \in \mathcal{A}^{(t)}$ is given as

$$Pr(s^{(t+1)}|s^{(t)}, a^{(t)}) = \int_{\mathcal{S}^{(t+1)}} \mathbb{F}(s^{(t)}, a^{(t)}, s')ds', \tag{39}$$

where $\mathbb{F}$ is the state transition probability density function.

B. Action space

The action space $\mathcal{A}^{(t)}$ at any decision epoch t includes the offloading decision $a = \{a_n\}$, $a_n \in \{0, 1\}$, block size S^B, block interval T^I, and the number of shards K^*. This is given as

$$\mathcal{A}^{(t)} = [a, S^B, T^I, K^*]^{(t)}, \tag{40}$$

where $a_n = 1$ when a validating node participates in the validation process and $a_n = 0$ otherwise. Similarly, $S^B \in \{0.2, 0.4, ..., \dot{S^B}\}$, $T^I \in \{0.5, 1, ..., \dot{T^I}\}$, and $K^* \in \{1, 2, ..., \dot{K^*}\}$, where $\dot{S^B}$, $\dot{T^I}$ and $\dot{K^*}$ are the block size limit, maximum block interval and largest shard number satisfying the security constraint respectively.

C. Reward Function

We aim to simultaneously optimize the transaction throughput and minimize the overall latency in the shard-based blockchain-enabled data sharing system. The objective of the system is given as

$$O = \Theta_1 T_{TTF} - (1 - \Theta_1)\Theta_2 T_{thru}, \tag{41}$$

where $\Theta_1 (0 < \Theta_1 < 1)$ is a weight factor, which is useful in combining two objective functions into a single one and Θ_2 is

a mapping factor that ensures two objective functions are at the same scale. From (41), the optimization problem can be obtained as

$$\begin{aligned} \min_{\mathcal{A}^{(t)}} \mathbb{E}\left[\sum_{t=0}^{+\infty}\left(\Theta_1 T_{TTF} - (1-\Theta_1)\Theta_2 T_{thru}\right)\right] \\ \text{s.t. (C1):} \quad T_{TTF} \leq \xi T^{I} \\ \text{(C2):} \quad f_k \leq \frac{N_{s,k}-1}{3} \\ \text{(C3):} \quad \sum_k N_{s,k} \leq N \\ \text{(C4):} \quad \sum_k f_k \leq f \\ \text{(C5):} \quad a_n \in \{0,1\} \\ \text{(C6):} \quad \rho < 1. \end{aligned} \tag{42}$$

The reward function is thus obtained as

$$\mathcal{R}^{(t)} = \begin{cases} -O(t), & \text{if } C1-C6 \text{ are satisfied} \\ 0, & \text{otherwise.} \end{cases} \tag{43}$$

D. Branching dueling Q-network

Because of the dynamic and large-dimensional characteristics of the proposed sBeDS problem, it is imperative to adopt the DRL technique. The large action space introduced by the proposed scheme, however, brings great challenges to the discrete-action-based DRL optimization methods since the number of actions that need to be explicitly represented in the conventional deep deterministic policy gradient (DDPG) and deep Q-network (DQN)-based agents grows exponentially with an increasing number of validators, which makes it hard to achieve convergence. To compensate for large dimensions of action space, we apply the BDQ algorithm.

BDQ is a branching variant of the dueling double deep Q-network that incorporates the action branching architecture into the DQN to decrease the number of estimated actions [29]. The advantage of the action branching architecture can be observed when solving problems in multidimensional action spaces since it is possible to optimize each action dimension with a degree of independence. BDQ enhances scalability by ensuring the linear growth of the total number of network outputs with increasing action dimensionality. The agent in BDQ can scale gracefully to environments with increasing action dimensionality and it was shown in [29] to perform competitively when compared with the conventional DDPG and other related algorithms. BDQ allows the adoption of discrete-action algorithms in DRL for domains with high-dimensional continuous or discrete action spaces. For any action dimension $d \in \{1, 2, ..., N_d\}$, each sub-action has $|\mathcal{A}_d| = \vartheta$ discrete sub-actions. The Q-value of any branch at any state $s \in \mathcal{S}$ and sub-action $a_d \in \mathcal{A}_d$ can be expressed as a function of the common state value $V(s)$ and the corresponding sub-action advantage $A_d(s, a_d)$ following

$$Q_d(s, a_d) = V(s) + \left[A_d(s, a_d) - \frac{1}{\vartheta}\sum_{a_d \in \mathcal{A}_d} A_d(s, a'_d)\right], \tag{44}$$

TABLE III
SIMULATION PARAMETERS

Parameter	Value
Average transaction size, χ	200 Bytes
Computation resource of each validator, c_v	[10, 30] MHz
Computing cost for validating signatures and generating/validating MACs, ζ, η	2 MHz/1 MHz
Timeout, τ	10 s
Validation interval, ξ	20
Acceptable threshold, a_{th}	0.5
Pre-defined reputation threshold, d_{th}	0.6
Bandwidth, W	1 MHz
Offloading power of each validator, $P_{v_i^k}$	1 W
Noise signal power, σ^2	10^{-9} W
Block size limit, $\dot{S^B}$	8 MB
Maximum block interval, $\dot{T^I}$	10 s
Maximum allowable shard number, $\dot{K^*}$	8

such that the temporal-difference target

$$y = \mathcal{R} + \gamma \frac{1}{N_d}\sum_d \bar{Q}_d\left(s', \text{argmax}_{a'_d \in \mathcal{A}_d}(s', a'_d)\right), \tag{45}$$

where parameters $\bar{Q}_d$ and γ are the branch d of the target network $\bar{Q}$ and the learning rate respectively. From (45), the loss function can be expressed as the expected value of the mean squared temporal-difference error across the branches, given as

$$L = \mathbb{E}_{(s,a^*,r,s')\sim\mathcal{D}}\left[\frac{1}{N_d}\sum_d (y_d - Q_d(s, a_d))^2\right], \tag{46}$$

where $\mathcal{D}$ is the experience replay buffer and a^* captures the joint action tuple $(a_1^*, a_2^*, ..., a_{N_d}^*)$. To preserve the magnitudes of the errors, the unified prioritization error can be expressed as

$$e_r(s, a^*, r, s') = \sum_d |y_d - Q_d(s, a_d)|. \tag{47}$$

VII. NUMERICAL RESULTS

In this section, we present numerical results to demonstrate the performance of the proposed multiple validation process-based blockchain-enabled data sharing system.

A. Simulation Parameters

To demonstrate the performance of the proposed blockchain-enabled data sharing framework, we carried out numerical simulations, consisting of $N = 100$ validators. To address issues related to the blockchain-enabled systems, the simulation results focus on the performance of the PBFT and reputation-enabled shard-based consensus process. To evaluate the message exchanging process among validators, we adopted the Rayleigh fading assumption such that the channel gain among interfering validators is independent and identically distributed exponential fading coefficient. Except otherwise mentioned, the parameters settings selected similar to [3], [16] are summarized in Table III.

For comparison, we modified the proposed sBeDS scheme to produce three existing schemes: single shard scheme, $K = 1$;

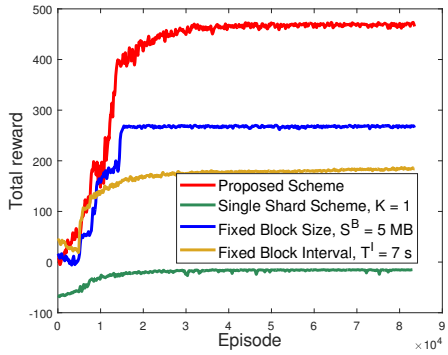

Fig. 2. Convergence performance with rewards.

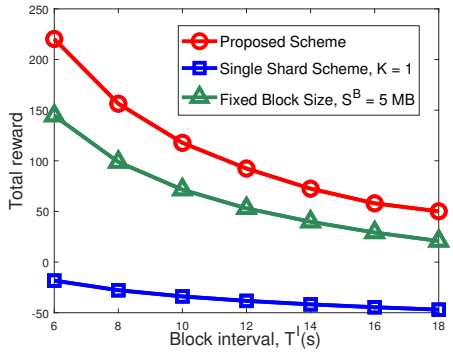

Fig. 3. Effects of block interval on the performance.

fixed block interval scheme, T^I = 7 s; and fixed block size scheme, S^B = 5 MB. For simplicity, we refer to the sBeDS scheme with all parameters optimized as the proposed scheme and set $\Theta_1 = 0.9$ and $\Theta_2 = 0.001$ similar to [24], [27], [28].

B. Simulation Results

In our simulations, we used a computer system with 10 CPU cores. The CPU is Intel(R) Core(TM) i9-10900X with 3.70GHz. We used PyTorch 0.4.1 and Python 3.6.6 as the software environment. The convergence performance of the proposed scheme is presented in Fig. 2. The total reward is shown to increase with the learning process until the optimal blockchain parameters are found. The proposed multi-shard scheme achieves higher throughput and lower latency thus a higher reward is observed compared to the other schemes. Interestingly, the convergence speed is relatively low when compared to the other schemes since the adopted multi-shard approach with variable block size and interval imposes more learning tasks on the agent at the initial learning stage. Notwithstanding that, the proposed multi-shard scheme can still achieve a reasonable convergence speed, while providing a higher total reward. It is worth noting that, even at $\xi = 20$, the single shard scheme continues to provide a total reward closer to zero since such an approach suffers from low throughput and high overall latency.

Next, we investigate the effects of the block interval on the performance of the proposed approach as presented in Fig. 3. The total reward is observed to decrease as the block interval increases since an increase in the block interval increases the overall latency while reducing the throughput. Under the fixed block interval scheme, an increase in the block interval does not affect the performance, thus a multi-shard scheme with a fixed block interval continues to produce a constant total reward. For other schemes with variable block intervals, the proposed scheme achieves a better total reward when compared with the single shard and fixed block size schemes. This further justifies that a multi-shard approach with variable block size and interval can achieve better performance. It is worth mentioning that all the considered schemes have been implemented by taking into consideration the required

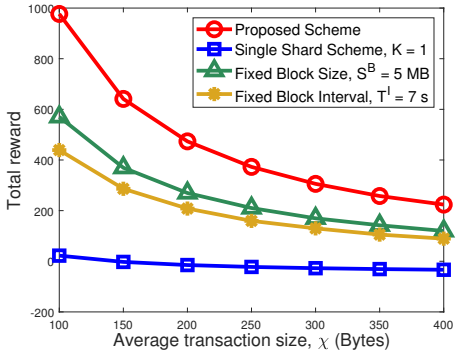

Fig. 4. Effects of transaction size on the performance.

security constraint as in the conventional PBFT consensus protocol. Thus, a multi-shard approach not only outperforms other schemes but also achieves the same level of security.

In Fig. 4, we evaluate the effects of the average transaction size on the overall performance of the proposed scheme. As the transaction size increases, the overall performance of the proposed scheme reduces since an increase in the transaction size increases validation time, which further increases the overall latency while reducing the average throughput. Interestingly, the proposed scheme can provide better performance compared to other schemes because such a scheme can optimally adjust other blockchain parameters to compensate for the increase in the average transaction size.

To investigate the effects of the computation capacity of validators on average throughput, Fig. 5 shows that the average throughput increases as the computation capacity limit of validators increases. This is expected since an increase in c_v will improve the validation process, thereby reducing the overall latency. Similarly, the proposed scheme achieves a better average throughput compared to other schemes. For the scheme with a fixed block generation interval, the average throughput remains constant as c_v increases since an improved validation experience (i.e., reduced validation time) in such a scheme mean validators will remain idle for a longer period owing to the fixed block interval. The fixed block interval

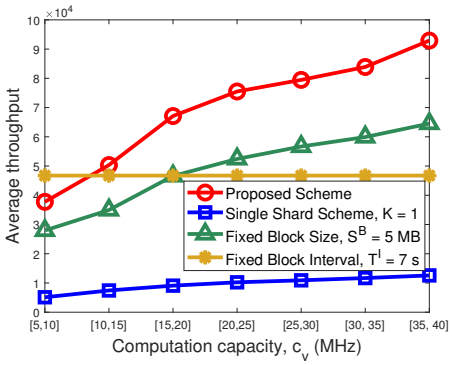

Fig. 5. Average throughput performance.

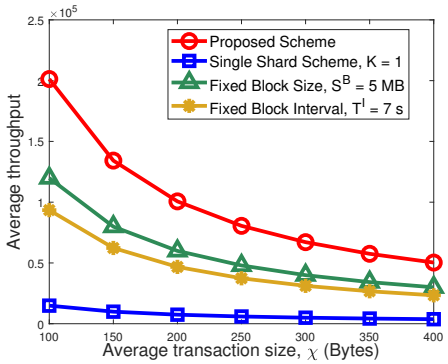

Fig. 7. Effects of transaction size on average throughput.

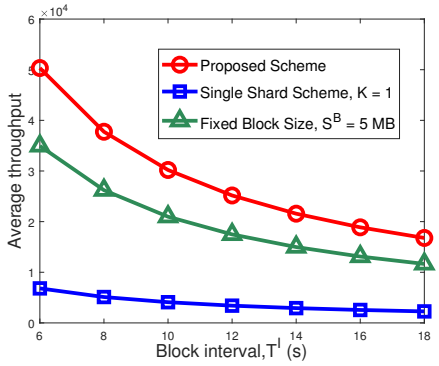

Fig. 6. Average throughput vs block interval.

scheme is expected to produce an improved performance when c_v is low as can be seen in Fig. 5.

Furthermore, Fig. 6 represents the effects of block interval on the average throughput. This is similar to Fig. 3, where the impacts of block interval on the overall performance of the proposed scheme were investigated. As expected, the average throughput reduces as the block interval increases, while the proposed scheme continues to produce better throughput. Similarly, Fig. 7 demonstrates the relationship between the average throughput and the average transaction size. The average throughput is observed to decrease as the transaction size increases since an increased transaction size further increases the validation time as well as message exchanging time among validators, which directly affects the overall latency. When compared with the single shard scheme, the proposed multi-shard schemes achieve better performance justifying the importance of the multiple validation process towards improving the validation process of blockchain-enabled data sharing systems.

Finally, the proposed PBFT and reputation-enabled shard-based scheme is capable of improving the scalability and throughput of blockchain-based data sharing systems, reduce overall latency, while ensuring sufficient decentralization and security features. As can be observed from the proposed framework, the approach is decentralized while the proposed integrated PBFT and trust-based proof of reputation scheme ensures that the security level is not compromised. Estimating the value of f based on the reputation values means only validators with acceptable historical reputations are always selected during the shard formation process thus, eliminating the probability of appending malicious blocks to the chain as in the conventional PBFT scheme, while providing an improved validation experience.

VIII. Conclusion

In this paper, we present a PBFT and reputation-enabled shard-based framework for data sharing process in large-scale blockchain-enabled systems to improve scalability and throughput, while ensuring that decentralization and security features are not compromised. We integrated trust-based proof-of reputation and PBFT consensus techniques and adopted queueing theory with priority technique to capture the block appending process. We then obtained analysis for important performance metrics of interest to demonstrate the importance of parallel validation techniques to blockchain-enabled data sharing systems. The joint transaction offloading among validators and computation resource allocation problem was formulated as an MDP to allow optimization of transaction throughput while reducing both communication and computation latency. We then applied the BDQ approach owing to its suitability when dealing with large dimensions of action space. Simulation results show that the proposed trust-based proof-of reputation and PBFT-enabled parallel validation technique can improve the overall performance of blockchain-enabled data sharing systems which will particularly enhance users' experience in large-scale applications.

In the future, we will consider shard formation through the coalition or cooperative game theory. We believe this may improve the shard formation process.

References

[1] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," Decentralized Business Review, pp. 21260, Oct. 2008.
[2] M. B. Mollah, J. Zhao, D. Niyato, Y. L. Guan, C. Yuen, S. Sun, K. Y. Lam, and L. H. Koh, "Blockchain for the internet of vehicles towards intelligent transportation systems: A survey," IEEE Internet of Things Journal, vol. 8, no. 6, pp. 4157-4185, Oct. 2020.

[3] J. Yun, Y. Goh, and J. M. Chung, "DQN-Based Optimization Framework for Secure Sharded Blockchain Systems," IEEE Internet of Things Journal, vol. 8, no. 2, pp. 708–722, Jul. 2020.

[4] S. Lee, M. Kim, J. Lee, R. H. Hsu, and T. Q. Quek, "Is Blockchain Suitable for Data Freshness? An Age-of-Information Perspective," IEEE Network, vol. 35, no. 2, pp. 96–103, Feb. 2021.

[5] O. Vashchuk and R. Shuwar, "Pros and cons of consensus algorithm proof of stake. Difference in the network safety in proof of work and proof of stake," Electronics and Information Technologies, vol. 9, no. 9, pp. 106–112, Jan. 2018.

[6] M. Castro and B. Liskov, "Practical Byzantine fault tolerance," Operation System Design Implementation, vol. 99, no. 1999, pp. 173–186, Feb. 1999.

[7] Y. Du, Z. Wang, J. Li, L. Shi, D. Jayakody, Chen, Q. Chen, W. Chen, and Z. Han, "Blockchain-Aided Edge Computing Market: Smart Contract and Consensus Mechanisms," IEEE Transactions on Mobile Computing, Jan. 2022, DOI: 10.1109/TMC.2021.3140080.

[8] C. Fan, S. Ghaemi, H. Khazaei, and P. Musilek, "Performance evaluation of blockchain systems: A systematic survey," IEEE Access, 8, pp. 126927–126950, Jun. 2020.

[9] M. Alaslani, F. Nawab, and B. Shihada, "Blockchain in IoT systems: End-to-end delay evaluation," IEEE Internet of Things Journal, vol. 6, no. 5, pp. 8332–8344, Oct. 2019.

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

[11] R. Yang, X. Chang, J. Mišić, V. B. Mišić, and H. Kang, "Quantitative Comparison of Two Chain-Selection Protocols under Selfish Mining Attack," IEEE Transactions on Network and Service Management, vol. 19, no. 2, pp. 1142–1158, Jun. 2022.

[12] J. Ye, X. Kang, Y. C. Liang, and S. Sun, "A Trust-Centric Privacy-Preserving Blockchain for Dynamic Spectrum Management in IoT Networks," IEEE Internet of Things Journal, vol., no. 15, pp. 13263–13278, Aug. 2022.

[13] Y. Sun, L. Zhang, G. Feng, B. Yang, B. Cao, and M. A. Imran, "Blockchain-enabled wireless Internet of Things: Performance analysis and optimal communication node deployment," IEEE Internet of Things Journal, vol. 6, no. 3, pp. 5791–5802, Mar. 2019.

[14] Y. Sun, L. Zhang, G. Feng, B. Yang, B. Cao, and M. Imran, "Performance analysis for blockchain driven wireless IOT systems based on tempo-spatial model," in IEEE International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery, Guilin, Oct. 2019, pp. 348–353.

[15] M. Liu, Y. Teng, F. R. Yu, V. C. Leung, and M. Song, "Deep reinforcement learning based performance optimization in blockchain-enabled internet of vehicle," in IEEE International Conference on Communications, Shanghai, May 2019, pp. 1–6.

[16] M. Liu, F. R. Yu, Y. Teng, V. C. Leung, and M. Song, "Performance optimization for blockchain-enabled industrial Internet of Things (IIoT) systems: A deep reinforcement learning approach," IEEE Transactions on Industrial Informatics, vol. 15, no. 6, pp. 3559–3570, Feb. 2019.

[17] M. Kim, S. Lee, C. Park, J. Lee, and W. Saad, "Ensuring Data Freshness for Blockchain-enabled Monitoring Networks," IEEE Internet of Things Journal, vol. 9, no. 12, pp. 9775–9788, Jun. 2022.

[18] S. D. Okegbile, J. Cai, A. S. Alfa, "Performance analysis of blockchain-enabled data sharing scheme in cloud-edge computing-based IoT networks," IEEE Internet of Things Journal, Jun. 2022, DOI: 10.1109/JIOT.2022.3181556.

[19] J. H. Chen, M. R. Chen, G. Q. Zeng, and J. S. Weng, "BDFL: a byzantine-fault-tolerance decentralized federated learning method for autonomous vehicle," IEEE Transactions on Vehicular Technology, vol. 70, no. 9, pp. 8639–8652, Aug. 2021.

[20] O. Alfandi, S. Otoum, and Y. Jararweh, "Blockchain solution for IOT-based critical infrastructures: Byzantine fault tolerance," in IEEE Network Operations and Management Symposium, Budapest, Jun. 2020, pp. 1–4.

[21] S. Gao, T. Yu, J. Zhu, and W. Cai, "T-PBFT: An EigenTrust-based practical Byzantine fault tolerance consensus algorithm," China Communications, vol. 16, no. 12, pp. 111–123, Dec. 2019.

[22] S. Kim, S. Lee, C. Jeong, and S. Cho, "Byzantine Fault Tolerance Based Multi-Block Consensus Algorithm for Throughput Scalability," in IEEE International Conference on Electronics, Information, and Communication, Barcelona, Jan. 2020, pp. 1–3.

[23] A. Loveless, R. Dreslinski, B. Kasikci, and L. T. Phan, "IGOR: Accelerating byzantine fault tolerance for real-time systems with eager execution," in IEEE Real-Time and Embedded Technology and Applications Symposium, Nashville, Jul. 2021, pp. 360–373.

[24] J. Feng, F. R. Yu, Q. Pei, X. Chu, J. Du, and L. Zhu, "Cooperative computation offloading and resource allocation for blockchain-enabled mobile-edge computing: A deep reinforcement learning approach," IEEE Internet of Things Journal, vol. 7, no. 7, pp. 6214–6228, Dec. 2019.

[25] J. Kang, Z. Xiong, D. Niyato, D. Ye, D. I. Kim, and J. Zhao, "Toward secure blockchain-enabled Internet of vehicles: Optimizing consensus management using reputation and contract theory," IEEE Transactions on Vehicular Technology, vol. 68, no. 3, pp. 2906–2920, Jan. 2019.

[26] G. Han, J. Jiang, L. Shu, and M. Guizani, "An attack-resistant trust model based on multidimensional trust metrics in underwater acoustic sensor network," IEEE Transactions on Mobile Computing, vol. 14, no. 12, pp. 2447–2459, Feb. 2015.

[27] X. Jiang, F. R. Yu, T. Song, and V. C. Leung, "Intelligent resource allocation for video analytics in blockchain-enabled internet of autonomous vehicles with edge computing," IEEE Internet of Things Journal, vol. 9, no. 16, pp. 14260–14272, Aug. 2022.

[28] L. Liu, J. Feng, Q. Pei, C. Chen, Y. Ming, B. Shang, and M. Dong, "Blockchain-enabled secure data sharing scheme in mobile-edge computing: An asynchronous advantage actor-critic learning approach," IEEE Internet of Things Journal, vol. 8, no. 4, pp. 2342–2353, Dec. 2020.

[29] A. Tavakoli, E. Pardo, and P. Kormushev, "Action branching architectures for deep reinforcement learning," Proceedings of the AAAI Conference on Artificial Intelligence, vol. 32, no. 1, April. 2018.

[30] F. Wei, G. Feng, Y. Sun, Y. Wang, S. Qin, and Y. C. Liang, "Network slice reconfiguration by exploiting deep reinforcement learning with large action space," IEEE Transactions on Network and Service Management, vol. 17, no. 4, pp. 2197–2211, Aug. 2020.

[31] R. Zhang, F. R. Yu, J. Liu, T. Huang, and Y. Liu, Y. "Deep reinforcement learning (DRL)-based device-to-device (D2D) caching with blockchain and mobile edge computing," IEEE Transactions on Wireless Communications, vol. 19, no. 10, pp. 6469–6485, Jun. 2020.

[32] A. Mizrahi, O. Rottenstreich, "Blockchain state sharding with space-aware representations," IEEE Transactions on Network and Service Management, vol. 18, no. 2, pp. 1571–1583, Jun. 2021.

[33] P. Zhang, W. Guo, Z. Liu, M. Zhou, B. Huang, and K. Sedraoui, "Optimized Blockchain Sharding Model Based on Node Trust and Allocation," IEEE Transactions on Network and Service Management, Jan. 2023, doi: 10.1109/TNSM.2022.3233570.

[34] X. Huang, Y. Wang, Q. Chen, and J. Zhang, "Security Analyze with Malicious Nodes in Sharding Blockchain based Fog Computing Networks," in IEEE Vehicular Technology Conference, Norman, Sept. 2021, pp. 1–5.

[35] 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 of Things Journal, vol. 9, no. 17, pp. 16494–16509, Sept. 2022.

[36] S. Yuan, J. Li, J. Liang, Y. Zhu, X. Yu, J. Chen, and C. Wu, "Sharding for Blockchain based Mobile Edge Computing System: A Deep Reinforcement Learning Approach," in IEEE Global Communications Conference, Madrid, Dec. 2021, pp. 1–6.

[37] N. Gao, R. Huo, S. Wang, T. Huang, and Y. Liu, "Sharding-Hashgraph: A High Performance Blockchain-Based Framework for Industrial Internet of Things with Hashgraph Mechanism," IEEE Internet of Things Journal, vol. 9, no. 18, pp. 17070–17079, Sept. 2022.

[38] S. D. Okegbile, B. T. Maharaj, and A. S. Alfa, "Interference characterization in underlay cognitive networks with intra-network and inter-network dependence," IEEE Transactions on Mobile Computing, vol. 20, no. 10, pp. 2977–2991, Oct. 2021.

[39] Y. Liu, J. Liu, Y. Hei, W. Tan, and Q. Wu, "A secure shard reconfiguration protocol for sharding blockchains without a randomness," in IEEE International Conference on Trust, Security and Privacy in Computing and Communications, Guangzhou, Dec. 2020, pp. 1012-1019.

[40] J. Wang, and H. Wang, "Monoxide: Scale out blockchains with synchronous consensus zones," in Proc. of Networked Systems Design and Implementation, 2019, pp. 95–112.

[41] S. D. Okegbile, and J. Cai, "Edge-assisted human-to-virtual twin connectivity scheme for human digital twin frameworks," in IEEE VTC Conference, Helsinki, Jun. 2022, pp. 1–6.

[42] S. D. Okegbile, J. Cai, C. Yi, and D. Niyato, "Human Digital Twin for Personalized Healthcare: Vision, Architecture and Future Directions," IEEE Network, Jul. 2022, DOI: 10.1109/MNET.118.2200071.

[43] A. Clement, E. Wong, L. Alvisi, and M. Dahlin, "Making Byzantine fault tolerant systems tolerate Byzantine faults," in Proceeding of Networked Systems Design and Implementation, Boston, Apr. 2009, pp. 153–168.

[44] J. Walraevens, D. Fiems, and H. Bruneel, "Performance analysis of priority queueing systems in discrete time," in Network Performance Engineering. Berlin, Germany: Springer, 2011, pp. 203–232.