An Improved BFT Algorithm in Traceability Data for Supply Chain

Zhiyong Liang[1], Rongwang Jiang[2], Ming Yang[3], Boxiong Yang[4]
School of Information & Intelligence Engineering, University of Sanya, Sanya, China[1, 2, 3]
Supercomputing Center, University of Sanya, Sanya, China[1]
Academician Workstation of Chunming Rong, University of Sanya, Sanya, China[2, 3]
Office of Scientific Research, University of Sanya, Sanya, China[4]

***Abstract*—BFT (Byzantine Fault Tolerance) is a set of fault-tolerance techniques used in distributed computing. In traditional supply chain traceability processes, centralized databases are vulnerable to errors and data tampering. By integrating the BFT consensus algorithm with an alliance blockchain, it is possible to address security challenges such as data deletion, misuse, application attacks, and reduced efficiency in storing supply chain traceability data. This approach represents a future trend for the secure and efficient storage and management of supply chain traceability information.**

***Keywords*—BFT; Consortium Blockchain; consensus; Tendermint; traceability data; supply chain**

I. Introduction

Tendermint began in early 2014 as an improved BFT. It was created from the open-source realization that although PoW was secure in BTC networks, it had serious shortcomings in speed and scalability. BFT is designed to handle abnormal behaviors and meet the requirements of complex problems. Tendermint is a secure state machine replication algorithm in the blockchain paradigm. It takes the form of BFT-ABC with additional accountability, allowing verification of Byzantine node dishonesty. The core of Tendermint runs on all nodes, handling consensus, peer-to-peer (P2P) networking, blockchain storage, and external interactions via RPC functions. Each Tendermint core can communicate with a custom blockchain application, a separate process, through a network socket protocol called TMSP.

A. Consensus

Consensus addresses consistency in distributed systems, ensuring operations are consistent, recognized, and tamper-proof across a distributed network for a defined period, as enforced by a protocol. In blockchain, consensus mechanisms aim to establish decentralized multi-party trust, requiring each node's ledger to match those of others. In centralized systems, this is nearly impossible due to the presence of a central server. Common blockchain consensus mechanisms include PoW, PoS, DPoS, Raft, Ripple, and PBFT.

B. PoW (Proof of Work)

Most public chains or digital currencies, such as BTC, ETH, and ETC, employ PoW for consensus, which allocates currency based on the amount of mining work. In BTC, consistency is achieved by competing for bookkeeping rights via hashrate. ETH now utilizes GPUs, FPGAs, and ASICs, resulting in the centralization of hashrate. In the long term, PoW trends away from decentralization and poses a threat to blockchain network security.

C. PoS (Proof of Stake)

PoS is like storing property in a bank, providing holders with interest based on the amount and duration of held currency. Peercoin was the first to use this, introducing coin-age to adjust the SHA256 difficulty inversely to the transaction input coin-age. While PoS addresses PoW issues, it still allows large stakeholders to gain dominance. Potential issues include Nothing at Stake, Initial Distribution, Long-Range Attack, and Coin Age Accumulation Attack.

D. DPoS (Delegated Proof of Stake)

DPoS, used by Bitshare, aims to mitigate the shortcomings of PoW and PoS by introducing a technical democratic layer to limit centralization. Risks include difficulty dealing with faulty nodes and insufficiently equipped super nodes being vulnerable to DDoS attacks, which can impact network stability.

E. Raft

Raft and similar protocols are more efficient than BFT algorithms when Byzantine faults aren't considered. Raft matches Paxos in efficiency but is simpler to implement. However, Raft only supports crash fault tolerance (CFT), not malicious node resilience, so it is suitable for private blockchains.

F. Ripple

Ripple is an open-source Internet payment protocol enabling decentralized exchanges and payments. Transactions are broadcast via tracking or validating nodes. Ripple is more efficient than PoW, confirming transactions in seconds, but it is best for Permissioned Chains. Ripple's BFT allows tolerance for 20% Byzantine nodes, a lower rate than classic BFT.

G. PBFT (Practical Byzantine Fault Tolerance)

Consortium Blockchains often use PBFT, Raft, or Paxos for strong consistency and consensus speed. PBFT is a mature and widely used protocol that improves upon BFT for real-world applications. Unlike PoW/PoS, PBFT avoids unnecessary computation for consensus, thereby reducing monetary security issues, latency, and low throughput, as well as wasted hashrate.

PBFT addresses BFT's increasing complexity, optimizing from an exponential to a polynomial scale, making Byzantine protocols feasible. It addresses original inefficiencies for trusted transactions, tolerating up to a third of Byzantine nodes without affecting consensus. PBFT uses one or more voting rounds to achieve consensus; its innovation is in a three-stage voting process (Pre-prepare, Prepare, Commit). Nodes 0–3 represent participants, with node 0 as the primary participant and node 3 as the Byzantine (non-responsive) node.

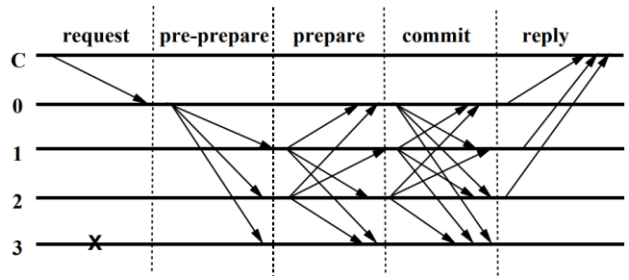

Fig. 1. Flowchart of PBFT.

At the same time, we can also see from Fig. 1 that PBFT requires three rounds of voting to achieve the consensus of the whole distributed system, but in the actual use of Consortium Blockchain, when the number of participating nodes is too many, the existence of the three-round voting mechanism makes the communication complexity still too high, which leads to a very rapid decline in the performance of the whole network, and the network latency also becomes high or even timeout.

II. TENDERMINT

A. Overview

Tendermint belongs to the BFT mechanism, an easy-to-understand and easy-to-implement State Machine Replication, which is based on the assumption of a semi-synchronous network that can tolerate 1/3 of Byzantine nodes with guaranteed Liveness and Safety. Tendermint is a Round-Based protocol that has been further optimized for PBFT. Compared to traditional PBFT, Tendermint ideally requires only two rounds of voting to reach consensus, ensuring that the entire network can achieve consensus securely and reliably.

Tendermint is characterized by the following six points:

- Byzantine Fault-Tolerant

Tendermint tolerates up to 1/3 of your machines failing arbitrarily. This includes explicitly malicious behavior.

- State Machine Replication

Tendermint can replicate deterministic state machines written in any programming language.

- Ultimate Certainty

Once all nodes reach consensus, Tendermint will not fail at some point in the future, like the consensus of BTC or ETH, so there will be no blockchain fork in Tendermint.

- Secure P2P

Gossip protocols and peer discovery are secured via Tendermint's authenticated encryption system.

- Lightning Fast

By avoiding POW consensus, Tendermint can achieve very high transaction throughput. With ideal application data structure support, 42,000 transactions per second can be achieved. Tendermint supports thousands of transactions per second.

- Easy to Deploy

Use Mintnet to deploy your app to any cloud provider supported by Docker Machine.

- 100% Open Source

All Tendermint's source code is licensed under Apache 2.0.

B. Workflow

In the Consortium Blockchain network based on BFP, nodes can reach consensus when fewer than 1/3 of the total number of Byzantine nodes are present. Tendermint is an easy-to-understand BFT consensus protocol that follows a simple state machine with the workflow diagram shown in Fig. 2.

In Fig. 2, there are two roles in Tendermint: Validator and Proposer. A validator is a role or node in the protocol, and different validators have varying vote powers in the voting process. The Proposer is elected by the Validator on a rotating basis, who takes turns proposing blocks for transactions and voting on proposed blocks. Blocks are committed to the chain, and each block has a block height. However, a block can fail to be submitted, in which case the protocol will select the next Proposer to propose a new block at the same height and restart the poll.

If a block needs to be successfully submitted, it must go through two stages of voting, called Pre-vote and Pre-commit. A block is committed only when more than 2/3 of the Validators have Pre-committed votes on the same block in the same round of proposals. Proposers may fail to propose blocks due to reasons such as offline or network latency. This situation is also allowed in Tendermint, as Proposers will wait for a certain amount of time before moving to the next round of offers for receiving blocks proposed by Proposers.

If fewer than one-third of Validators are Byzantine nodes, Tendermint can guarantee that Validators will never repeatedly commit blocks at the same height, thereby causing conflicts. To achieve this, Tendermint introduces a locking mechanism where, once Validators have pre-verified a block, that Validator will be locked to that block. If the block is not successfully submitted in the current round of pre-proposals and pre-votes, the Validator is unlocked, and the next round of pre-submissions for the new block is performed.

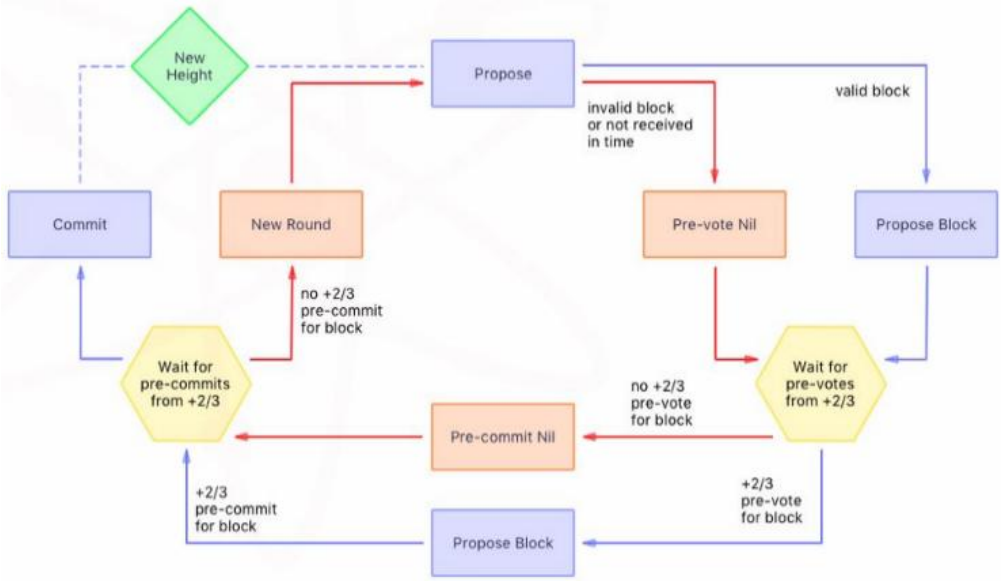

Fig. 2. Flowchart of Tendermint.

C. Comparison between Tendermint and PBFT

A detailed review of the workflows shows that Tendermint may be viewed as an optimized version of PBFT, but several technical features distinguish the two. The following clarifies the similarities and differences between them.

- Similarities
  - Both belong to the BFT system.
  - All are resistant to 1/3 Byzantine node attacks.
  - Three-stage submission, with the first stage broadcasting transactions (blocks) and the last two stages broadcasting signatures (confirmations).
  - Both require a quorum to submit a block.
- Differences
  - The primary difference between Tendermint and PBFT lies in their respective judgments when more than one-third of the Byzantine nodes are network-wide nodes. PBFT can't provide guarantees when the number of Byzantine nodes is between 1/3 and 2/3 of the number of Validators, allowing the attacker to return arbitrary results to the client. Tendermint considers that more than two-thirds of pre-commit acknowledgments are required to commit a block. For example, if 1/2 of the Validators are Byzantine nodes, in Tendermint, these Byzantine nodes can block the submission of blocks, but they themselves cannot submit malicious blocks. In PBFT, Byzantine nodes can submit malicious blocks to the client.
  - In the Byzantine node concept, PBFT refers to the number of nodes, while Tendermint represents the number of interests, or voting power, of the nodes.
  - While PBFT requires a pre-defined set of Validators for the validation process, Tendermint supports dynamic changes in Validators by requiring more than 2/3 of the quorum of validators to approve membership changes.

III. Consensus Performance in Traceability Data for Supply Chain Based on Tendermint and PBFT

A. Problems of Current Supply Chain Traceability Data

In response to problems such as serious centralization, easily tampered information, and opaque data in the process of supply chain data traceability, Consortium Blockchain can be used to realize the safe and effective storage of traceability data. Although PBFT is currently more widely used in federated chains, the theoretical comparison in Section II-C shows that using Tendermint is superior to PBFT in terms of performance and speed.

B. Experimental Environment Setup and Preparation

The experimental part of this paper was conducted on node 1 in the Supercomputing Center at the University of Sanya, which features 20 Intel(R) Xeon(R) Silver 4114 CPUs at 2.20 GHz, 128 GB of RAM, and 436 TB of hard disk space. The operating system is CentOS Linux release 7.6.2009, and the Fabric-Consortium Blockchain is built by deploying a Kubernetes (k8s) Master node and multiple Kubernetes (k8s) nodes with Docker, where each k8s node represents a Consortium Blockchain deployment node. Since Fabric adopts a loosely coupled design, components such as consensus and identity verification are modularized, making it easy to select the corresponding modules according to the application scenario during the application process. Therefore, Tendermint and PBFT will be used to test the consensus performance of supply chain data for comparison throughout the system, respectively.

During the test, the following five performance indicators were examined:

- Fail Rate: Percentage of operation failures (including timeouts)
- Avg Latency: Average time taken to complete a transaction
- Query Throughput: Query requests per second
- Consensus Throughput: Consensus requests per second
- Consistency Throughput: Synchronous operations per second

C. Results Without Byzantine Failures

PBFT was used to test the traceability data for the supply chain, and the indicators are shown in Table I.

TABLE I. TABLE INDICATORS USING PBFT

Serial Number	1	2	3
Test Contents	Supply Chain	Supply Chain	Supply Chain
Data Quantity	1000	2000	5000
Fail Rate	0	0	0
Avg Latency	5.9s	15.24s	42.09s
Query Throughput	948.3tps	1874.2tps	4879.32tps
Consensus Throughput	577tps	570ps	576tps
Consistency Throughput	525tps	530tps	504tps

Tendermint was used to test the traceability data for the supply chain, and the indicators are shown in Table II.

TABLE II. TABLE INDICATORS USING TENDERMINT

Serial Number	1	2	3
Test Contents	Supply Chain	Supply Chain	Supply Chain
Data Quantity	1000	2000	5000
Fail Rate	0	0	0
Avg Latency	1.32s	2.24s	4.19s
Query Throughput	986.7tps	1987.2tps	4983.32tps
Consensus Throughput	993tps	1991tps	4985tps
Consistency Throughput	984tps	1797tps	1877tps

As shown in Table II, after switching to Tendermint, the consensus throughput can reach up to 5,000 TPS, and the Consistency Throughput is close to 1,900 TPS. At the same time, the average latency of Tendermint is also significantly reduced compared to PBFT. Such write performance is comparable to that of a traditional single-node relational database, and can satisfy most commercial scenarios.

From the data in Tables I and II, we can see that the average latency of the algorithm is significantly lower than that of the PBFT consensus algorithm due to the further optimization of the algorithm and the reduction of the number of voting rounds of Tendermint, which reduces the complexity of the whole algorithm, and the comparison of the two latencies is shown in Fig. 3.

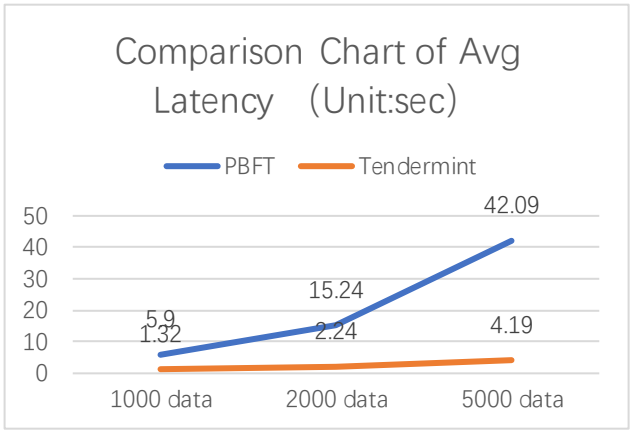

Fig. 3. Comparison chart of average latency.

For the efficiency of the two consensus, the most critical metrics to examine are: Query Throughput, Consensus Throughput, and Consistency Throughput. The performance metrics comparison graph is shown in Fig. 4. From the figure, we can see that the adoption of the optimization algorithm has improved the data throughput of the entire model to some extent, particularly in Consensus Throughput and Consistency Throughput, which have seen a more significant improvement.

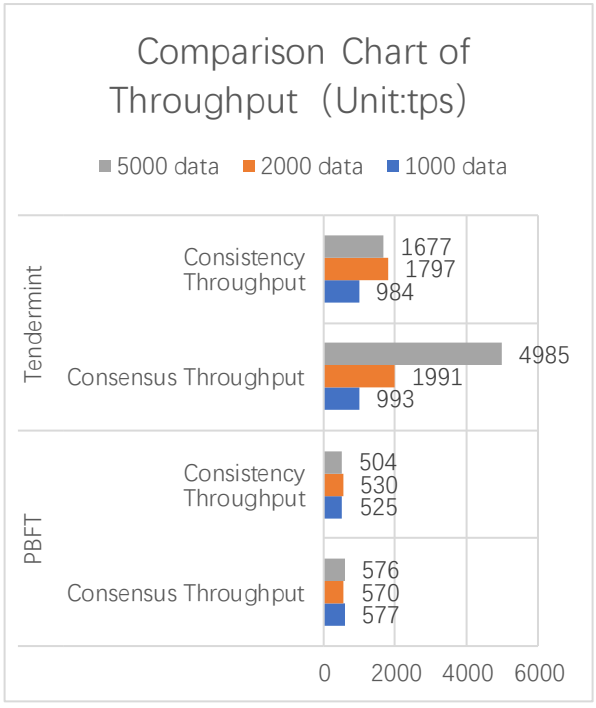

Fig. 4. Comparison chart of throughput.

D. Results with Byzantine Failures

Despite the injected Byzantine faults, which would cause many systems to fail completely and immediately, Tendermint

maintains respectable latencies. When processing more data and encountering more Byzantine failures, Tendermint achieves lower latency and higher throughput than PBFT due to partial optimization in the voting node. The consensus performance of Tendermint and PBFT when processing 5,000 produce traceability data and encountering 30% Byzantine failures is shown in Table III.

TABLE III. CONSENSUS PERFORMANCE PARAMETERS WITH 30% BYZANTINE FAILURES

	Tendermint	PBFT
Fail Rate	0%	0%
Avg Latency	5.38s	50.56s
Query Throughput	4791.9tps	4364.63tps
Consensus Throughput	4801tps	505tps
Consistency Throughput	1658tps	445tps

As shown in Table III, with more traceability data and a higher percentage of Byzantine failures, Tendermint exhibits a low performance loss of approximately 2%-4%, while maintaining a low average latency and a high throughput. However, PBFT suffers from high performance loss and excessive latency, as shown in Fig. 5 and Fig. 6.

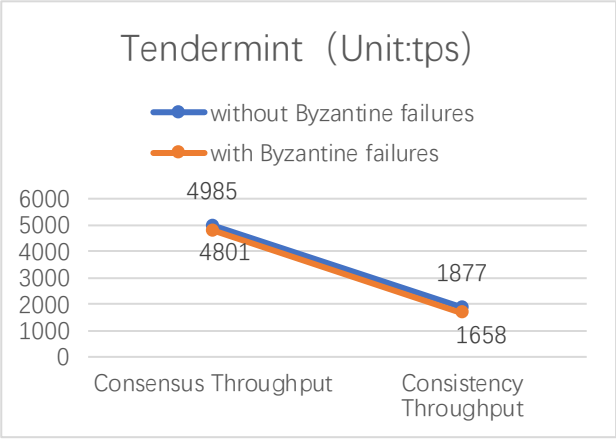

Fig. 5. Core performance metrics comparison chart of Tendermint.

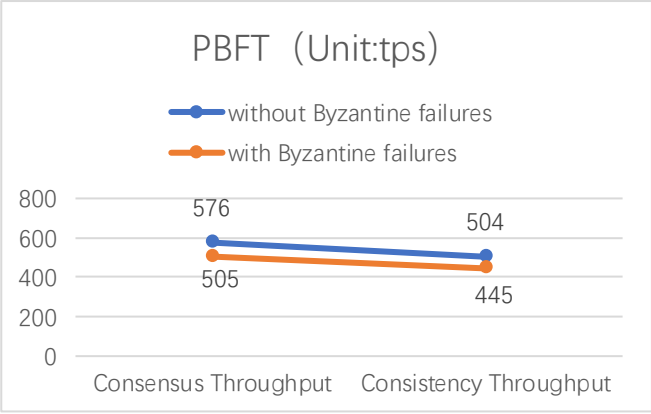

Fig. 6. Core performance metrics comparison chart of PBFT.

IV. CONCLUSION

We introduced Tendermint, an improved blockchain consensus protocol based on BFT. Tendermint optimizes the election and voting sessions based on PBFT, reducing complexity to achieve lower latency and improved data throughput in the consensus process, while also enhancing resistance to attacks with fewer than 1/3 Byzantine failures. In the experimental session, Tendermint outperforms PBFT.

V. ACKNOWLEDGMENT

This work is supported by Hainan Provincial Higher Education Scientific Research Project (Grant Hnky2025-33), Hainan Province Key R&D Program Project (Grant ZDYF2023GXJS007), and University of Sanya Major Special Projects (USY22XK-04). Meanwhile, the experimental part of this paper was conducted on the supercomputing system at the Supercomputing Center, University of Sanya, China.

REFERENCES

[1] Ge Z, Loghin D, Ooi B C, et al. Hybrid Blockchain Database Systems: Design and Performance[J]. VLDB Endowment, 2022, 15(5): 1092-1104.

[2] Alqahtani S M. Analyzing and Improving Performance In BFT Consensus Protocols [J]. 2021.

[3] Alqahtani S, Demirbas M. BigBFT: A Multileader Byzantine Fault Tolerance Protocol for High Throughput[C]//2021 IEEE International Performance, Computing, and Communications Conference (IPCCC). IEEE, 2021: 1-10.

[4] Cason D, Fynn E, Milosevic N, et al. The design, architecture and performance of the Tendermint Blockchain Network[C]//2021 40th International Symposium on Reliable Distributed Systems (SRDS). IEEE, 2021: 23-33.

[5] Karamachoski J, Gavrilovska L. Tendermint Performance with Large Transactions: The Healthcare System Scenario[C]//2021 International Balkan Conference on Communications and Networking (BalkanCom). IEEE, 2021: 85-89.

[6] Arora S K, Kumar G, Kim T. Blockchain based trust model using tendermint in vehicular adhoc networks[J]. Applied Sciences, 2021, 11(5): 1998.

[7] Buchman E. Tendermint: Byzantine fault tolerance in the age of blockchains[D]. University of Guelph, 2016.

[8] Lagaillardie N, Djari M A, Gürcan Ö. A computational study on fairness of the tendermint blockchain protocol[J]. Information, 2019, 10(12): 378.

[9] Braithwaite S, Buchman E, Khoffi I, et al. A Tendermint light client[J]. arXiv preprint arXiv:2010.07031, 2020.

[10] Amoussou-Guenou Y, Del Pozzo A, Potop-Butucaru M, et al. Dissecting tendermint[J]. arXiv preprint arXiv:1809.09858, 2018.

[11] Amoussou-Guenou Y, Biais B, Potop-Butucaru M, et al. Rational vs Byzantine Players in Consensus-based Blockchains [C]//AAMAS. 2020: 43-51.

[12] Lei L, Lan C, Lin L. Chained Tendermint: A Parallel BFT Consensus Mechanism[C]//2020 3rd International Conference on Hot Information-Centric Networking (HotICN). IEEE, 2020: 25-33.

[13] Gerasimov I, Chizhov I. Consensus algorithm of Tendermint platform and Proof Of Lock Change mechanism [J]. International Journal of Open Information Technologies, 2019, 7(6): 24-29.

[14] Buchman E, Kwon J, Milosevic Z. The latest gossip on BFT consensus[J]. arXiv preprint arXiv:1807.04938, 2018.

[15] Gao S, Yu T, Zhu J, et al. T-PBFT: An EigenTrust-based practical Byzantine fault tolerance consensus algorithm[J]. China Communications, 2019, 16(12): 111-123.

[16] Choi B, Sohn J, Han D J, et al. Scalable network-coded PBFT consensus algorithm[C]//2019 IEEE International Symposium on Information Theory (ISIT). IEEE, 2019: 857-861.

[17] Lao L, Dai X, Xiao B, et al. G-PBFT: a location-based and scalable consensus protocol for IOT-Blockchain applications[C]//2020 IEEE International Parallel and Distributed Processing Symposium (IPDPS). IEEE, 2020: 664-673.

[18] Wu Y, Song P, Wang F. Hybrid consensus algorithm optimization: A mathematical method based on POS and PBFT and its application in blockchain[J]. Mathematical Problems in Engineering, 2020, 2020.

[19] Coelho I M, Coelho V N, Araujo R P, et al. Challenges of pbft-inspired consensus for blockchain and enhancements over neo dbft[J]. Future Internet, 2020, 12(8): 129.

[20] Meshcheryakov Y, Melman A, Evsutin O, et al. On performance of PBFT blockchain consensus algorithm for IoT-applications with constrained devices[J]. IEEE Access, 2021, 9: 80559-80570.

[21] Zheng X, Feng W, Huang M, et al. Optimization of PBFT algorithm based on improved C4. 5[J]. Mathematical Problems in Engineering, 2021, 2021.

[22] Kim Y, Park J. Hybrid decentralized PBFT Blockchain Framework for OpenStack message queue[J]. Human-centric Computing and Information Sciences, 2020, 10(1): 1-12.

[23] Xu G, Bai H, Xing J, et al. SG-PBFT: A secure and highly efficient distributed blockchain PBFT consensus algorithm for intelligent Internet of vehicles[J]. Journal of Parallel and Distributed Computing, 2022, 164: 1-11.

[24] Chen J, Zhang X, Shangguan P. Improved PBFT Algorithm Based on Reputation and Voting Mechanism[C]//Journal of Physics: Conference Series. IOP Publishing, 2020, 1486(3): 032023.

[25] Feng X, Ma J, Miao Y, et al. Social Characteristic-based Propagation-efficient PBFT Protocol to Broadcast in Unstructured Overlay Networks[J]. IEEE Transactions on Dependable and Secure Computing, 2021.

[26] Zhu S, Zhang Z, Chen L, et al. A PBFT consensus scheme with reputation value voting based on dynamic clustering[C]//International Conference on Security and Privacy in Digital Economy. Springer, Singapore, 2020: 336-354.