Journal of Electronic Research and Application, 2025, Volume 9, Issue 6
http://ojs.bbwpublisher.com/index.php/JERA
ISSN Online: 2208-3510
ISSN Print: 2208-3502

An Improved PBFT Algorithm Based on Dual Scoring Mechanism

Peng Zhao, Weixuan Xu

Department of Computer Science and Technology, Taiyuan Normal University, Jinzhong, 030619, Shanxi, China.

**Author to whom correspondence should be addressed.*

Abstract: The traditional Practical Byzantine Fault Tolerance (PBFT) approach suffers from three critical deficiencies: arbitrary primary node election, excessive network transmission overhead, coupled with the absence of node incentive mechanisms. To address these issues, this study proposes a refined PBFT strategy utilizing dual scoring (Double Scoring Practical Byzantine Fault Tolerance, DS-PBFT). The algorithm innovatively combines hardware performance evaluation with a dual-dimensional node scoring system. The algorithm first employs bucket sorting technology to quantitatively evaluate node hardware resources, followed by constructing a comprehensive scoring model through credit values and recommendation values. According to the scoring outcomes, the framework hierarchically divides nodes into primary node, follower node and backup nodes groups in a 1:4:5 ratio, substantially decreasing the quantity of nodes involved in consensus. Additionally, this approach streamlines the Commit-Reply stages within the consistency protocol, substantially reducing communication overhead. Experimental validation demonstrates that DS-PBFT maintains security while achieving notable improvements in consensus efficiency, significant reductions in communication costs, and enhanced defense capabilities against malicious nodes.

Keywords: Blockchain technology; Practical byzantine fault tolerance; Node performance evaluation; Consensus protocol optimization

Online publication: December 16, 2025

1. Introduction

Blockchain technology, representing a decentralized distributed ledger framework, has shown extensive implementation potential across various fields such as finance, supply chain, as well as medical services in recent years [1–3]. However, as blockchain networks continue to grow and the evolving application scenarios improving its security, operational efficiency, and system reliability has become a key challenge in current research. The consensus algorithm serves as one of the fundamental components of blockchain technology, significantly influencing the operational efficiency and scalability of blockchain systems [4,5].

According to different deployment modes, blockchains can be classified into three types: public chains,

consortium networks, and permissioned chains. In consortium and private chains, nodes must be authorized before joining the blockchain network, hence they are also called permissioned chains [6,7]. The consensus algorithms used in permissioned chains are primarily distributed consistency algorithms, including Practical Byzantine Fault Tolerance algorithm and its optimization algorithms, Paxos algorithm, and Raft algorithm [8–10]. The PBFT algorithm can tolerate attacks from a certain number of malicious nodes and improve system reliability and availability through state machine replication mechanism [11]. However, the PBFT algorithm also has critical limitations in real-world deployments. For instance:

(1) The arbitrary selection of primary nodes in PBFT cannot guarantee the reliability of primary nodes, leading to frequent view switching and severely affecting system throughput;
(2) PBFT adopts a multi-stage communication protocol which, despite guaranteeing communication security, greatly increases system overhead and reduces system efficiency;
(3) PBFT lacks incentive and punishment mechanisms, providing insufficient positive reinforcement for honest nodes and inadequate punishment for malicious nodes, making it difficult to maintain node enthusiasm for participating in consensus.

To address these problems, we introduce an enhanced consensus protocol DS-PBFT, which introduces an adaptive dual-dimensional scoring approach that scores nodes based on their historical performance, consensus efficiency, and consensus completion rate, thereby selecting nodes with better performance and more honest behavior as primary nodes. Moreover, addressing the issue that communication overhead increases sharply with the number of nodes within extensive networks, our protocol divides network nodes into primary node, follower node, and backup nodes, and optimizes the PBFT consistency protocol, ensuring continuous node role updates while reducing individual node communication burden and overall communication overhead. Leveraging the comprehensive evaluation model, this paper introduces an incentive-penalty framework that, by rewarding reliable nodes and punishing malicious nodes, reduces malicious node resource occupation while enhancing node participation enthusiasm.

2. PBFT overview

2.1. PBFT theory

The Byzantine Fault Tolerance problem originated from the Byzantine Generals Problem proposed by Leslie Lamport and others in 1982, which is a classic challenge in distributed systems [12]. This problem describes how to ensure all honest nodes reach consensus on a decision in a distributed environment with malicious or faulty nodes. In a Byzantine environment, malicious nodes may adopt arbitrary behaviors, including sending incorrect information, forging messages, selective forwarding, or complete non-response, which pose serious threats to the consistency and security of distributed systems. The Practical Byzantine Fault Tolerance algorithm was proposed by Miguel Castro and Barbara Liskov in 1999, and is the first practical algorithm to achieve efficient Byzantine fault tolerance in asynchronous network environments [8].

2.2. Consistency protocol

This PBFT approach adopts a State Machine Replication (SMR) model, ensuring all honest nodes achieve agreement regarding the processing sequence for requests through multiple rounds of information exchange [13]. Its basic workflow consists of five stages: Request, Pre-preparation, Preparation, Commitment, and Reply.

A client first submits a transaction to the primary node (Request phase). Upon receiving this transaction, the

primary node assigns a sequence number to it and broadcasts the "sequence number and request content" across all participating nodes (Pre-Prepare stage). Then, every node verifies the message's validity upon receiving it from the primary node. If the message is valid, this node broadcasts to all other nodes that it agrees to process the request with the given sequence number (Prepare phase).

When a node collects agreement messages from more than 2/3 of the nodes (including itself), it proceeds to the next phase. This node then broadcasts to all nodes that it confirms everyone is ready (Commit phase). When it collects confirmation messages from more than 2/3 of the nodes, it indicates that a consensus has been reached across the network. Finally, this node actually processes the transaction and returns the processing outcome to the requesting client (Reply phase).

A client can be confident that the result is reliable if it receives at least F + 1 matching responses (where F represents the tolerable quantity of Byzantine nodes). This entire process ensures that even if no more than 1/3 of the nodes fail or act maliciously, the remaining honest nodes can still reach a consensus on the processing sequence and result for the request through two rounds of network-wide voting (Prepare and Commit). The consistency protocol diagram for this approach is illustrated in **Figure 1**.

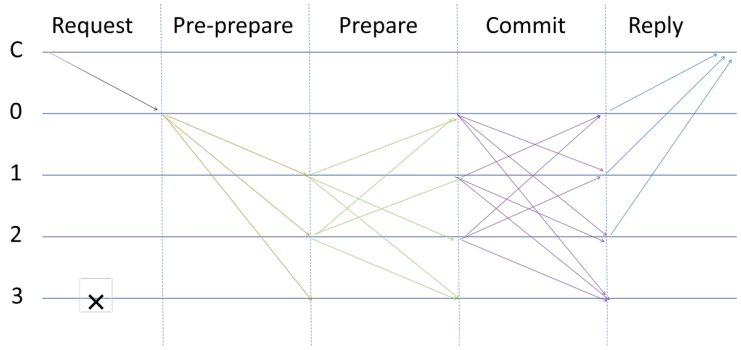

Figure 1. PBFT algorithm consistency protocol.

3. DS-PBFT consensus algorithm model

Our study proposes an enhanced Byzantine Fault-Tolerant protocol utilizing a two-layer evaluation framework, aiming to enhance operational efficiency and throughput of the consensus process. We introduce a Credit values (CV) methodology for assessing node trustworthiness of nodes while classifies them into three categories based on this score: primary nodes, follower node, and backup nodes.

During this consensus procedure, exclusively primary node and follower node, while backup nodes do not, effectively decreasing the quantity of nodes engaged in reaching agreement, consequently improving efficiency. Then, a primary node election mechanism is proposed, that identifies the most reliable nodes as primary nodes based on their credit values. By reducing the frequency of primary node switching, the consensus efficiency is further enhanced. Finally, our approach streamlines the consistency protocol, reducing communication as well as computational overhead and significantly increasing the system's throughput. Overall, this algorithm enhances the system's performance and fault tolerance by optimizing node selection and the consensus protocol.

At the beginning of the consensus phase, the system sets the initial credit value of each node to 50 and introduces hardware indicators to calculate the credit value of the nodes. Based on the calculated credit values, the system sorts all nodes through the bucket sort algorithm and arranges them in descending order. Nodes at the top of the list will become the primary nodes, followed by the follower node, while those with lower credit values

are classified as backup nodes. After each round of the consensus process is completed, the system dynamically updates the credit value of each node to reflect the reliability and efficiency demonstrated by the nodes during the consensus process. This dynamic adjustment mechanism ensures that the credit values of the nodes can be continuously optimized as the system operation conditions change.

Through this approach, the system can achieve self-regulation, ensuring that the selection of nodes in the consensus process is more reasonable and flexible, thereby enhancing the overall consensus efficiency and fault tolerance. The DS-PBFT algorithm model diagram is shown in **Figure 2**.

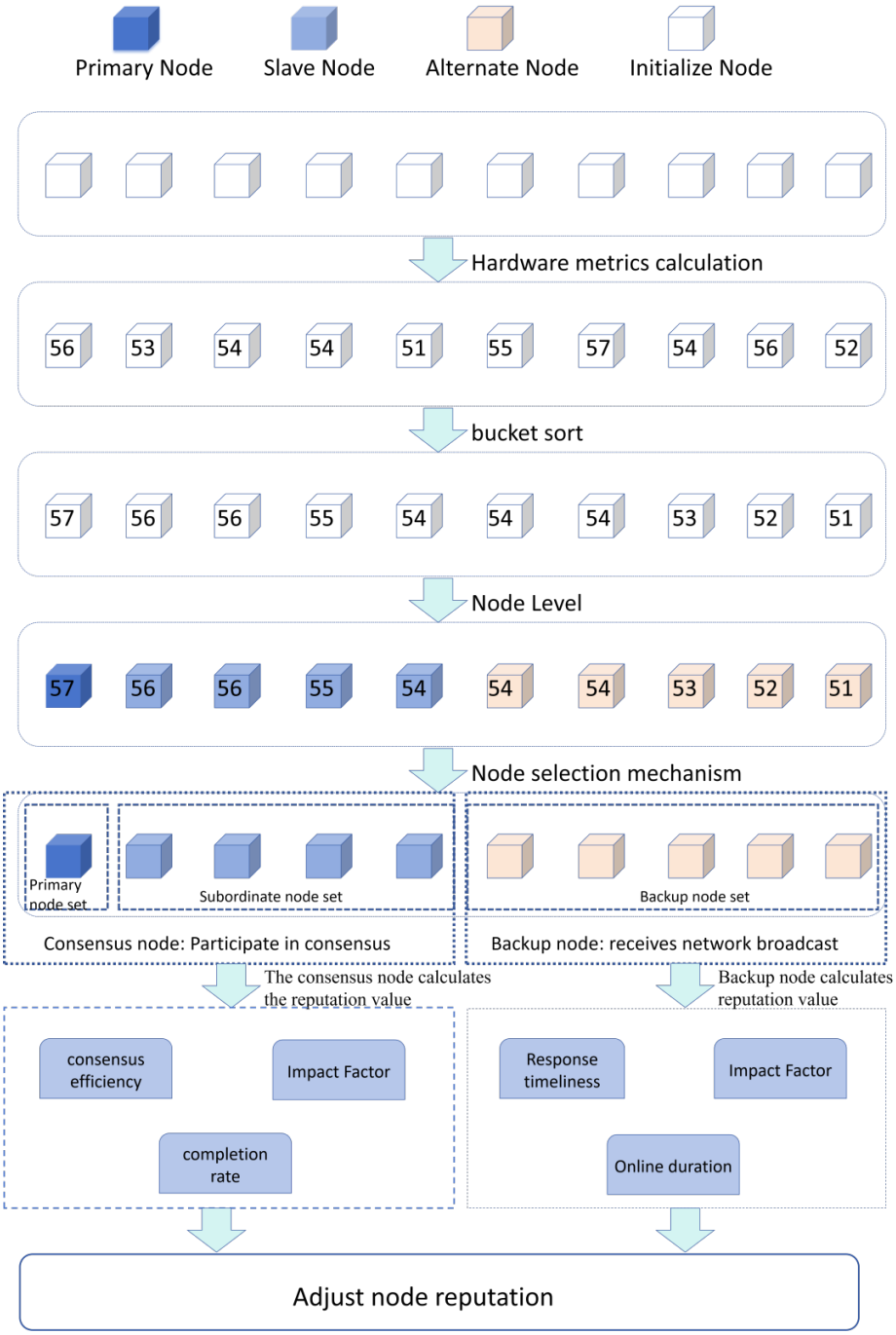

Figure 2. DS-PBFT algorithm model diagram.

3.1. Node grouping

During consensus network initialization, all nodes' credit values are set to 50. To distinguish between malicious nodes and normal nodes, a hardware comprehensive indicator parameter is introduced. This parameter is used to calculate the initial credit value of nodes for differentiation in the early consensus stage.

3.1.1. Hardware comprehensive indicator scoring mechanism

The hardware comprehensive indicator refers to the hardware performance that a node can provide when joining the consensus network. The calculation formula for this indicator is shown below

$$CV_i=50+w_c\cdot C_i+w_m\cdot M_i+w_s\cdot S_i+w_n\cdot N_i \quad (1)$$

Where C_i is the CPU efficiency of node i, M_i is the memory efficiency of node i, S_i is the storage efficiency of node i, and N_i is the network quality of node i.'

3.1.2. Node hierarchy division

Conventional PBFT protocols mandate universal participation from every node during consensus operations, leading to elevated communication costs but also may lead to a decrease in efficiency, especially in scenarios with extensive network scales. Within our DS-PBFT framework, the bucket sort algorithm is adopted. Nodes are divided into three categories: primary node group, follower node group, and backup nodes group according to their credit values in a ratio of 1:4:5. Nodes in the primary node set are given higher weights, with nodes possessing superior credit scores being designated as network's main nodes.

These primary node and follower nodes jointly engage in consensus operations, while backup nodes do not take part in the actual consensus operation. This design significantly decreases node involvement within the consensus process, reducing node quantities for consensus to approximately half of the original, thereby significantly reducing communication costs and computational burden, and improving the efficiency and throughput of the system. Moreover, as primary node and follower node selection relies upon credit values, overall reliability and consensus efficiency of the system have also been optimized, ensuring a more efficient consensus process and stronger fault tolerance.

3.2. Node credit value scoring mechanism

Addressing the problem that traditional PBFT algorithm lacks reward and punishment mechanisms leading to low node participation enthusiasm, this paper proposes a mechanism based on comprehensive node scoring. This mechanism uses credit values to quantitatively evaluate network nodes, comprehensively scoring nodes based on hardware performance, historical performance, behavior records and other factors to generate node credit values. The system prioritizes choosing nodes possessing superior trust scores to assume primary nodes in guiding the current round of consensus process; follower node and backup nodes, ensuring timely replacement when the primary node fails while preserving protocol consistency and operational effectiveness. Our node assessment framework based on comprehensive capability strengthens the system's resistance to attacks while simultaneously decreasing transmission costs and latency through reasonable consensus task allocation, thus enhancing system performance.

3.2.1. Primary node selection

Addressing the problem that random primary node selection in traditional PBFT algorithm leads to unstable node

performance, which in turn causes frequent view switching and affects overall system performance, this paper proposes a primary node selection strategy based on comprehensive node scoring mechanism. This strategy selects the node with the highest comprehensive score from the primary node set to serve as the primary node. When the primary node behaves as a faulty node, the system will deduct 50% of that node's credit value and downgrade it to the backup node set, while simultaneously reselecting the node with the current highest comprehensive score from the primary node set to serve as the new primary node and continue leading the consensus process.

3.2.2. Consensus node selection

Within conventional PBFT protocols, every node in the network serve as validation nodes participating during every validation phase. Nevertheless, in large-scale network environments, this full participation mechanism leads to significantly increased communication overhead. With extensive number of nodes involved in reaching agreement, the communication burden generated by the three-phase consensus protocol is excessive, leading to decreased transaction throughput and inefficient consensus. To solve this problem, DS-PBFT adopts a node grouping mechanism where only nodes from the primary node and follower node category as validation nodes participating in the consensus process, while other nodes serve as backup nodes. DS-PBFT effectively decreases node quantities engaged in agreement operations while ensuring agreement security, significantly lowering transmission overhead while enhancing overall agreement operational effectiveness.

This mechanism combines a dynamic node adjustment strategy as follows:

(1) Dynamic adjustment of primary node set: When a fault occurs within the primary node, our framework selects more trustworthy nodes from follower node nodes according to overall scores to promote into leadership positions, giving these nodes the opportunity to become primary nodes. Meanwhile, faulty nodes within the primary node are deducted 50% credit value and downgraded into the backup nodes set;

(2) Dynamic adjustment of follower node set: When faulty nodes appear in the follower node set, the system selects more trustworthy nodes from the backup node set based on comprehensive scores to supplement the follower node set. Meanwhile, faulty nodes in the follower node set are deducted 50% credit value and downgraded to the backup node set.

3.2.3. Node credit value calculation

Within our DS-PBFT framework, primary node and follower node groups engage in validation operations. That is, trust scores for these active nodes are calculated based on their operational efficiency, task completion rate, and past performance metrics. Although backup nodes do not engage in validation activities, their credit value assessment relies on response timeliness, online availability, and historical impact factor. The specific credit value calculation formula for consensus nodes is as follows:

(1) Node consensus efficiency

$$CE_t=\alpha_1 \cdot log_2(1+\frac{T_{max}-T_i}{T_{max}}) \quad (2)$$

T_i represents the time taken by node i to complete consensus in round t, T_{max} represents the maximum allowed completion time, a_1 is the efficiency weight coefficient, and CE_t is the consensus efficiency score. Where $\frac{T_{max}-T_i}{T_{max}} \in [0,1]$, when the node completes consensus in shorter time, the $\frac{T_{max}-T_i}{T_{max}}$ value is larger, indicating higher consensus efficiency.

(2) Node consensus completion rate

$$CCR_t=\alpha_2 \cdot log_2(1+\frac{M_t}{N_t}) \quad (3)$$

M_t represents the number of successful consensus completions by the node up to round t, N_t represents the total number of consensus sessions the node should have participated in up to round t, a_2 is the completion rate weight coefficient, and CCR_t is the consensus completion rate. Where $\frac{M_t}{N_t} \in [0,1]$, when the node completes more times, the $\frac{M_t}{N_t}$ value is larger, indicating higher consensus efficiency.

(3) Historical impact of consensus node credit value

$$HIF_a = \gamma \cdot RS_{i\text{-}1} \quad (4)$$

RS_{t-1} represents the node's credit value in round t-1, γ is the attenuation coefficient, and HIF_a reflects the influence metric for the present iteration affected by preceding operations. If consensus is completed, this nodes will increase according to prior round performance. Under anomalous consensus conditions, this influence factor is cumulatively applied, as shown in Equation (5).

$$\text{Historical impact factor } \gamma=\begin{cases} p^3 , & x>0, \text{ consensus abnormal} \\ p=2/3 , & x=0, \text{ consensus normal} \end{cases} \quad (5)$$

Where x indicates the count of irregular consensus activities. When a node fails to engage in validation procedures three times, its credit value will decrease to zero and face removal from the validation network. Therefore, the ultimate credit value credit value calculation formula for consensus nodes can be expressed as:

$$CV_i=\begin{cases} 0, \text{node i sends different incorrect information to different nodes} \\ 0.5\ RS_{i\text{-}1}, \text{consensus fault} \\ \alpha_1 \cdot log_2(1+\frac{T_{max}-T_i}{T_{max}}) + \alpha_2 \cdot log_2(1+\frac{M_t}{N_t}) + \gamma RS_{i\text{-}1}, \text{consensus normal} \end{cases} \quad (6)$$

Where: $\alpha_1+\alpha_2+\gamma=1$

Since backup nodes do not participate in consensus, to improve the enthusiasm of backup nodes, we allocate credit values to backup nodes based on node response timeliness, node online duration, and historical impact factors. The credit value calculation formula for backup nodes is as follows:

(1) Backup node response timeliness

$$MRE_t=\beta_1 \cdot log_2(1+\frac{R_{max}-R_i}{R_{max}}) \quad (7)$$

R_i represents the average response time of node i, R_{max} represents the maximum allowed response time, β_1 is the response efficiency weight coefficient, and MRE_t is the message response efficiency score. Where , when the node's average response time is shorter, the $\frac{R_{max}-R_i}{R_{max}}$ value is larger, and MRE_t is larger.

(2) Backup node online duration

$$OD_t=\beta_2 \cdot log_2(1+\frac{T_{online}}{T_{total}}) \quad (8)$$

T_{online} represents the cumulative online duration of node i, T_{total} represents the total duration since the consensus nodes joined the current time, β_2 is the online duration weight coefficient, and OD_t is the online duration score. Where $\frac{T_{online}}{T_{total}} \in [0,1]$, when the node's cumulative online time is larger, the $\frac{T_{online}}{T_{total}}$ value is larger, and is larger.

(3) Historical impact of backup node credit value

$$HIF_b=\mu \cdot RS_{i-1} \tag{9}$$

RS_{t-1} represents the node's credit value in round t-1, μ is the attenuation coefficient, and HIF_b is the impact of the current round affected by the previous round.

Therefore, the final credit value calculation formula for backup nodes can be expressed as:

$$CV_i=\beta_1 \cdot log_2(1+\frac{R_{max}-R_i}{R_{max}})+\beta_2 \cdot log_2(1+\frac{T_{online}}{T_{total}})+\mu RS_{i-1} \tag{10}$$

Where: $\beta_1+\beta_2+\mu=1$

3.3. Consistency protocol

Within conventional PBFT validation mechanisms, preparation phase is a crucial step for achieving consensus. It can effectively identify the Byzantine malicious nodes across the system and continue validation operations even while tolerating up to f compromised nodes. During periods when primary nodes are honest, initial stages (such as preparation stage) guarantee that every honest node achieves uniform consensus provided that Byzantine node count remains within f.

Especially within three-stage confirmation mechanism of the traditional PBFT, the submission stage plays a crucial role, ensuring uniform execution even when primary node switch, thereby preventing main nodes sending distinct transaction data under identical sequence identifiers, and ensuring system stability throughout view transition procedures. Within our DS-PBFT algorithm, this submission stage has been further optimized and adjusted. By introducing an integrated credit value mechanism, DS-PBFT can select the main node more accurately, thereby significantly improving the reliability and security of the main node. Additionally, DS-PBFT introduces a penalty mechanism to reduce the risk of errors by the main node.

This measure effectively enhances the stability of the main node. Based on this, DS-PBFT refines validation procedures of PBFT, especially during submission phase, enabling the system to reduce unnecessary overhead while maintaining high consensus efficiency. Different from the traditional PBFT algorithm, in the DS-PBFT algorithm, only the primary node group with follower node actively engage in validation operations; standby nodes do not join calculation and instead receive and synchronize the execution results broadcast by the main node to their local ledgers.

This design significantly reduces node quantities participating within validation procedures, thus reducing communication burden and improving consensus efficiency. At the same time, DS-PBFT also optimizes the response stage, ensuring the rapid dissemination of consensus results and efficient synchronization of the system, further enhancing the fault tolerance and consistency of the system. Through these innovations, the DS-PBFT algorithm not only maintains the high fault tolerance of PBFT while simultaneously enhancing operational effectiveness and reliability for consensus process. The specific three-phase improvement mechanism is illustrated in **Figure 3**.

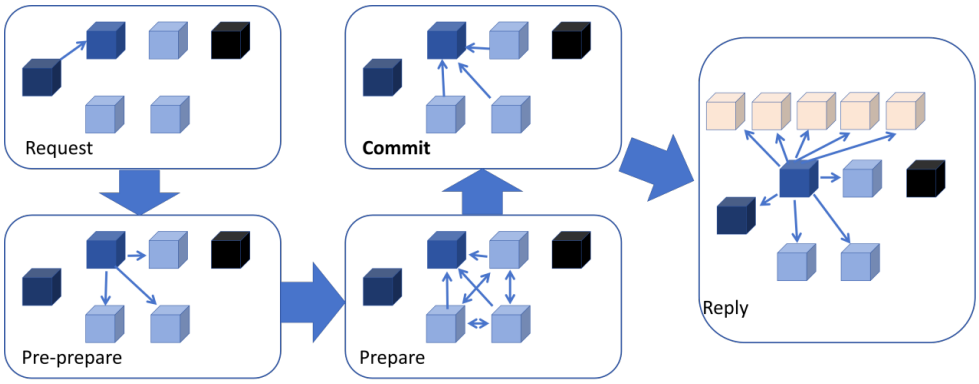

Figure 3. DS-PBFT algorithm three-phase optimization diagram.

4. Experimental analysis

4.1. Theoretical analysis

The DS-PBFT consensus algorithm first introduces a dynamic screening mechanism based on node credit values, selecting high-credibility nodes from the primary node set as primary node candidates, significantly improving the reliability and security of primary node election. The algorithm innovatively adopts a half-node participation consensus mode, which not only ensures consensus efficiency but also allows the consensus node group to tolerate no more than 1/3 Byzantine node abnormal behavior, while the backup node cluster can withstand a higher proportion of malicious node attacks, thereby constructing a dual-layer fault tolerance protection system and effectively enhancing system fault tolerance and security.

4.2. Simulation experiment analysis

This experiment conducts performance simulation experiments on PBFT and DS-PBFT consensus algorithms based on Python programming language. The experiment is set against the background of IoT multi-node application scenarios, with different scales of node numbers set for simulation testing. Performance evaluation is mainly conducted from three dimensions: consensus latency, communication overhead, and consensus energy consumption for comparative analysis. Specific experimental configuration parameters are shown in **Table 1**.

Table 1. Experimental configuration

Component	Configuration
CPU	Intel(R) Core(TM) i7-10875H
Operating system	Windows 11
Software environment	PyCharm
Memory	16GB
Programming language	Python

4.2.1. Consensus latency analysis

In blockchain systems, consensus latency is defined as the time required from client request submission to final confirmation. This experiment sets node quantities to scale from 200 up to 300 in increments of 20 units. To

ensure result reliability across experimental conditions, 10 experiments are repeated under each node scale, and the average latency across 10 iterations is used for the ultimate evaluation metric for that node count. A comparative analysis of response time between DS-PBFT and PBFT algorithms is shown in **Figure 4**.

Testing outcomes demonstrate that response time for our DS-PBFT framework is substantially reduced compared to conventional PBFT protocols. The performance advantage mainly stems from three improvements:

(1) DS-PBFT optimizes the consistency protocol process by removing the Commitment and Response stages;

(2) Node categorization approach decreases node involvement within validation operations;

(3) This streamlined protocol reduces message transmission between nodes.

In contrast, conventional PBFT protocols mandate universal participation from every node during validation process and adopts a three-phase communication mechanism, resulting in larger communication overhead. Therefore, the DS-PBFT algorithm has obvious advantages in consensus latency.

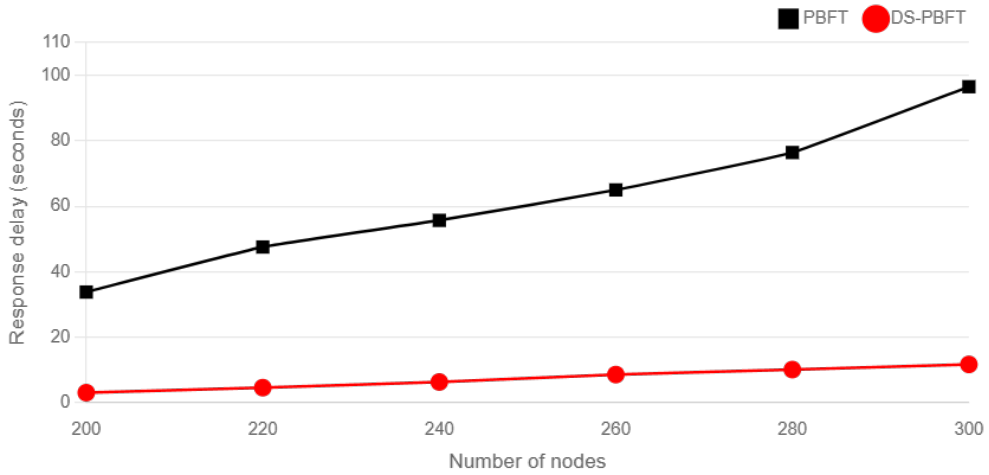

Figure 4. Consensus latency comparison.

4.2.2. Communication overhead analysis

Transmission burden refers to the aggregate volume of message transmission between nodes within validation operations. Our DS-PBFT framework substantially decreases this overhead through two optimizations:

(1) Node categorization approach effectively decreases engagement levels for validation procedures;

(2) It simplifies network communication stages during the consistency protocol.

Specifically, within conventional PBFT protocols, every N node must engage in validation operations, while DS-PBFT adopts a 1:4:5 grouping strategy, with only N/2 nodes participating within validation procedures, and the remaining nodes act as backup nodes only receiving the final result broadcast and not participating in message transmission. To compare the communication overhead of the two algorithms, a comparative experiment was conducted in this section.

Communication overhead is defined here as aggregate message volumes transmission generated by participating nodes throughout validation procedures. Testing outcomes indicate that as node count grows, transmission burden for DS-PBFT maintains a relatively stable growth trajectory, while transmission burden for conventional PBFT protocols rises rapidly. This result fully demonstrates the advantage of DS-PBFT in improving communication efficiency.

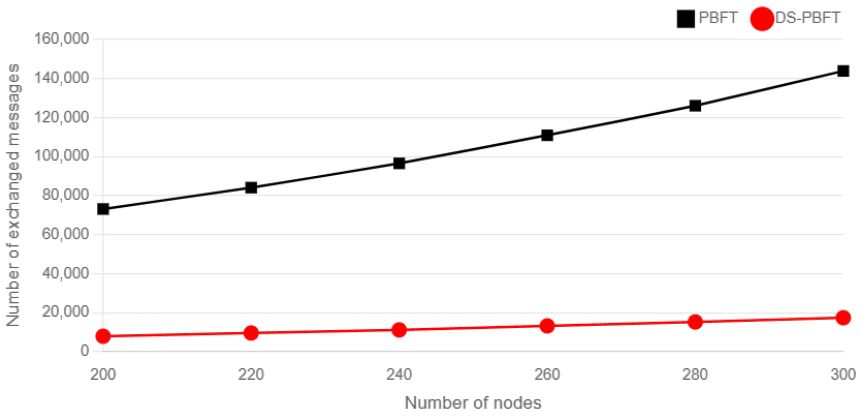

Figure 5. Communication overhead comparison.

4.2.3. Consensus energy consumption analysis

Consensus energy consumption is one of the important indicators for measuring blockchain system performance, mainly reflected in CPU usage and memory occupation. In practical application scenarios, reducing resource consumption during the consensus process is of great significance for improving system scalability and economics. To verify the advantages of the DS-PBFT algorithm in resource consumption, this experiment conducted comparative tests on PBFT and DS-PBFT algorithms from two dimensions: CPU usage and memory occupation, as shown in **Figure 6** and **Figure 7**.

Based on **Figure 6**, it becomes evident that with increasing node quantities, CPU utilization for conventional PBFT algorithm shows a rapid upward trend, mainly because every node is required to engage in multi-stage validation protocol, and substantial volumes of message processing along with verification procedures lead to increased CPU load. In contrast, the CPU usage rate growth of the DS-PBFT algorithm is relatively gentle, which benefits from the following two improvements:

(1) Node categorization approach decreases involvement levels within validation procedures, reducing message volumes that individual nodes must process;

(2) The streamlined consistency protocol communication phases reduce unnecessary computation overhead.

Figure 7 shows memory utilization comparative analysis for the two algorithms. Experimental data shows our DS-PBFT framework demonstrates superior performance compared to conventional PBFT protocols regarding memory occupation. Conventional PBFT protocols mandate universal node involvement to maintain comprehensive message records and state information, and memory occupation increases significantly with growing node counts. Our DS-PBFT framework, through its node grouping strategy, allows backup nodes to not maintain complete consensus message records but only synchronize final block data, thereby effectively reducing overall memory consumption.

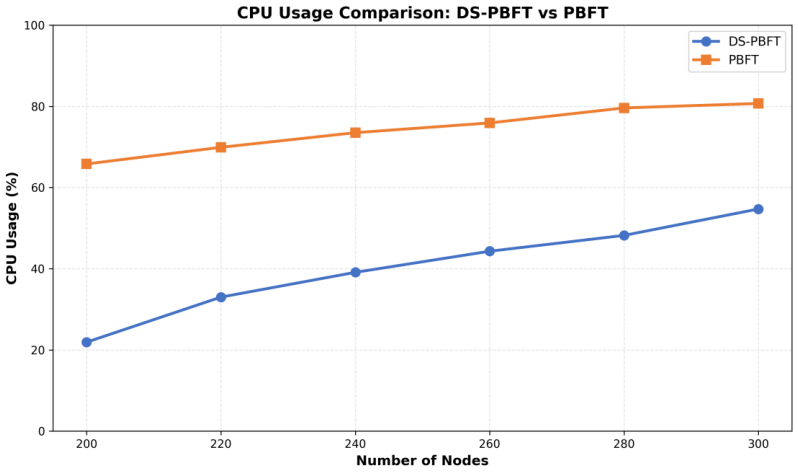

Figure 6. CPU usage comparison.

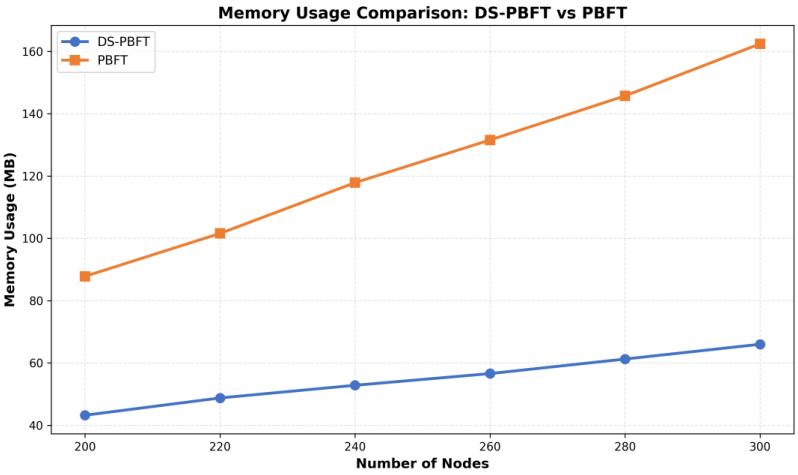

Figure 7. Memory occupation comparison.

4.2.4. Scoring analysis

To verify the effectiveness of the node scoring mechanism in the DS-PBFT algorithm, this experiment simulated the dynamic change trend of node credit values during multiple consensus rounds. The experiment selected 8 nodes as samples to simulate the credit value evolution process within 20 consensus cycles, as shown in **Figure 8**.

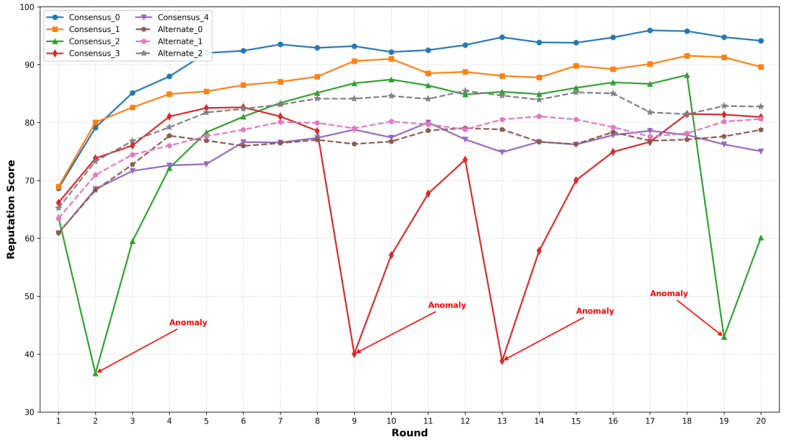

Figure 8. Node reputation trend.

Comprehensively speaking, the experimental results fully verify the effectiveness of the dual-dimensional scoring mechanism in the DS-PBFT algorithm. This mechanism can not only accurately identify and select high-performance nodes to serve as primary nodes and consensus nodes but also effectively incentivize honest nodes and suppress malicious behavior through reward and punishment mechanisms, achieving dynamic optimization adjustment of node roles, thereby improving the security and stability of the overall consensus network.

5. Conclusion

Addressing the challenges of random primary node election, substantial multi-stage communication overhead, and inadequate incentive mechanisms inherent in conventional PBFT protocols, this paper proposes an enhanced PBFT framework, DS-PBFT, based on a dual evaluation methodology. The contributions of the proposed algorithm span three major dimensions. A comprehensive node scoring mechanism is introduced to quantitatively assess node trustworthiness using both credit values and hardware performance indicators, ensuring reliable selection of primary and validation nodes. The node selection strategy is further refined through a 1:4:5 grouping ratio that partitions nodes into primary, follower, and backup categories, with only half of the nodes actively participating in the validation process. This design enhances scalability for large-scale networks while maintaining security. Additionally, the consistency protocol is optimized by improving the Commitment and Response stages, which results in a notable reduction in communication overhead. Experimental results show that DS-PBFT achieves substantially better performance than traditional PBFT protocols in terms of consensus latency, communication cost, and energy consumption. Despite the balanced consideration of efficiency and security, opportunities remain for further refinement, particularly regarding transmission complexity and enhanced Byzantine fault-tolerance capabilities. These aspects represent promising directions for future research on blockchain consensus optimization.

Disclosure statement

The author declares no conflict of interest.

References

[1] Yuan F, Zuo Z, Jiang Y, et al., 2025, AI-Driven Optimization of Blockchain Scalability, Security and Privacy Protection. Algorithms, 18(5): 263.

[2] Wang J, Wang T, Yuan W, et al., 2023, A Review of the Development History of Distributed Ledger Technology. Application Research of Computers, 40(3): 641-648.

[3] Liang B, Yuan F, Deng J, et al., 2025, Cs-pbft: A Comprehensive Scoring-Based Practical Byzantine Fault Tolerance Consensus Algorithm. The Journal of Supercomputing, 81(7): 859.

[4] Deng X, Wang Z, Li J, et al., 2022, Comparative Research on Mainstream Blockchain Consensus Algorithms. Application Research of Computers, 39(1): 1–8.

[5] Wang W, Hoang D, Hu P, et al., 2019, A Survey on Consensus Mechanisms and Mining Management in Blockchain Networks. IEEE Access, 2019(7): 22328–22370.

[6] Veronese G, Correia M, Bessani A, et al., 2013, Efficient Byzantine Fault-Tolerance. IEEE Transactions on Computers, 62(1): 16–30.

[7] Zou J, Zhang H, Tang Y, et al., 2018, Blockchain Technology Guide, China Machine Press, Beijing, 158–161.

[8] Castro M, Liskov B, 1999, Practical Byzantine Fault Tolerance. Proceedings of the 3rd Symposium on Operating Systems Design and Implementation, 173–186.

[9] Wang J, Fan Y, Zhang H, 2020, Topic Discovery and Evolution of Blockchain Literature. Computer Engineering and Applications, 56(20): 1–8.

[10] Ongaro D, Ousterhout J, 2014, In Search of an Understandable Consensus Algorithm. Proceedings of the USENIX Annual Technical Conference, 305–319.

[11] Pease M, Shostak R, Lamport L, 1980, Reaching Agreement in the Presence of Faults. Journal of the ACM, 27(2): 228–234.

[12] Lamport L, Shostak R, Pease M, 1982, The Byzantine Generals Problem. ACM Transactions on Programming Languages and Systems, 4(3): 382–401.

[13] Schneider F, 1990, Implementing Fault-Tolerant Services Using the State Machine Approach: A Tutorial. ACM Computing Surveys, 22(4): 299–319.