International Journal of Digital Crime and Forensics
Volume 14 • Issue 3

Improvement of the PBFT Algorithm Based on Grouping and Reputation Value Voting

Shannan Liu, College of Information Science and Technology, Shihezi University, China
Ronghua Zhang, College of Information Science and Technology, Shihezi University, China
Changzheng Liu, College of Information Science and Technology, Shihezi University, China*
Chenxi Xu, School of Economics and Management, Qilu Normal University, China
Jie Zhou, College of Information Science and Technology, Shihezi University, China
Jiaojiao Wang, College of Information Science and Technology, Shihezi University, China

ABSTRACT

An improved practical Byzantine fault tolerance (Practical Byzantine Fault Tolerant consensus algorithm based on reputation, RPBFT) algorithm based on grouping and reputation value voting is proposed for the problems of high communication complexity, poor scalability, and random selection of master nodes of the practical Byzantine fault tolerance (PBFT) consensus algorithm of the consortium chain. First, the consistency process is optimized to take the response speed of nodes to each group leader as the basis of grouping, and the intragroup consensus is performed. The group leader then takes the result of intragroup consensus and participates in extra-group consensus to reduce the frequency and time of internode communication. Second, the reputation model and voting mechanism are proposed, and the group leader is generated by node reputation value voting, which enhances the initiative and reliability of trusted nodes and reduces the abnormal nodes as group leader. Finally, a simulation and performance testing system based on this improved scheme is built to prove the effectiveness as well as the usability of the scheme through simulation experiments. The experimental results show that when the number of network nodes is 36, the throughput of the RPBFT algorithm is six times that of PBFT. Therefore, the consensus delay is reduced by 91.7%, and the communication overhead is reduced by 37.8%.

KEYWORDS

Blockchain, Consensus Algorithm, Consortium Chain, Node Grouping, PBFT, Reputation Value Model, RPBFT, Voting Mechanism

INTRODUCTION

As a type of distributed database, blockchain has the features of decentralization, non-tampering, traceability, and programmability (Sanka, A. I., & Cheung, R. C. C., 2020; Zheng, X., 2022). To better protect users' privacy and data, blockchain technology has gradually developed from public chains to consortium chains and is widely used in various industries. Practical Byzantine fault tolerance (PBFT)

DOI: 10.4018/IJDCF.315615 *Corresponding Author

is one of the mainstream algorithms of consortium chains at present. However, it has problems such as high communication complexity, poor scalability, and low fault tolerance (Fu, X., et al., 2021; Liu, Y. A., & Stoller, S.D., 2019), which makes it difficult for blockchain-related projects to meet actual business needs, so the optimization of the PBFT algorithm is imperative.

PBFT

Consistency Protocol

The PBFT algorithm was proposed in 1999 as an algorithm specifically designed to solve the Byzantine general problem (Gao, Z., & Yang, L., 2020; Yu, G., et al., 2020). The algorithm aims to solve the problem of ensuring the consistency and correctness of the final decision in the presence of malicious nodes throughout the network.

The PBFT algorithm can tolerate no more than (n-1)/3 amount of malicious or faulty nodes (n is the total number of nodes), as shown in equation (1). If there are f malicious nodes, the number of normal nodes is at least f+1, and the total number of nodes is at least 3f+1 to allow the system to function properly, as shown in equation (2). If there is an extreme case with f malicious nodes as well as f faulty nodes, the total number of nodes is at least 3f+1 to ensure that the system can reach consensus smoothly:

$$n - f > f \Rightarrow n \geq 2f + 1 \tag{1}$$

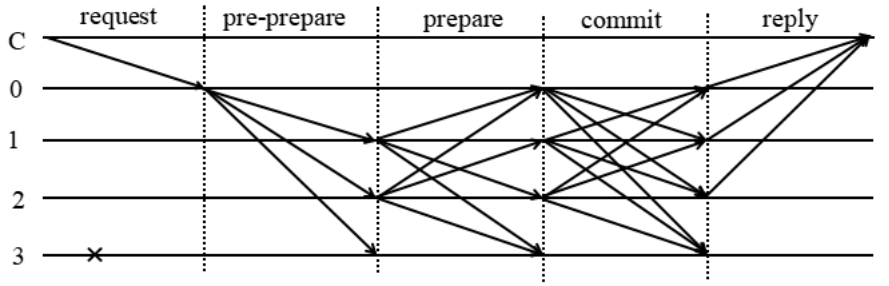
$$n - f - f > f \Rightarrow n \geq 3f + 1 \tag{2}$$

In the PBFT algorithm, all nodes are classified into three types: client, master, and replica nodes. The master node mainly assigns sequence numbers to the transaction requests initiated by the clients, and the replica nodes execute the requests based on the sequence numbers to check whether there is any problem with the master node. When the master node is unable to complete the consensus, it will use the view change mechanism formula to continue the selection of a new master node as shown in equation (3). In this equation P is the master node number, V is the view number, and $|R|$ is the number of nodes:

$$P = V \bmod \left|R\right| \tag{3}$$

The consensus process of PBFT consists of five stages: request, pre-prepare, prepare, commit, and reply. The algorithm flow is shown in Figure 1.

Figure 1. PBFT Algorithm Flow



- **Request:** Client *C* sends a transaction request <REQUEST,o,t,c> to master node *0*, where REQUEST contains the message content *m* and the message digest *d(m)*. The client needs to sign the request; *t* is the timestamp, *o* indicates the operation, and *c* represents this client.
- **Pre-prepare:** Master node *0* receives the request from client *C*, determines whether the number of messages being processed exceeds the limit, and if the limit is exceeded, it is first cached and then packaged and processed together later; if the limit is not exceeded, it generates <<PRE-PREPARE,v,n,d>,m> messages and broadcasts them to other nodes, where *v* is the view number, *n* is the sequence number of the request assigned a sequence number, and *d* is the summary of message *m*.
- **Prepare:** Each replica node receives a "Prepare" message and verifies the current view, sequence number, transaction, and signature. If all pass, it broadcasts a <PREPARE,v,n,d,i> message to the other nodes to indicate that it received and acknowledged the request, where *i* is the node number; if not, it ignores the message.
- **Commit:** After receiving 2*f* valid prepare messages, the node broadcasts a commit message <COMMIT,v,n,d,i> from the node, which receives 2*f* confirmation messages from other nodes, writes the received messages to the log if they are passed, executes the request, and replies with the message <REPLY,v,t,c,i,r> to the client, where *r* is a response of node *i*.
- **Reply:** The result is valid when the client node receives at least *f*+1 identical response results from different nodes.

Defects of the PBFT Algorithm

Although the PBFT algorithm is widely used in the consortium chain, there are still many problems (Feng, L., et al., 2018; Li, Y., et al., 2021). First, there is a security risk in the selection of master nodes. If malicious nodes are selected as master nodes several times, the consensus efficiency will be significantly reduced, thereby wasting system resources and reducing the stability and reliability of the system. Second, clients send requests only to master nodes, and the master nodes become overburdened when there are too many requests. This burden is not suitable for the peer-to-peer (P2P) network environment of blockchain. Finally, the PBFT algorithm has poor scalability and high network communication overhead. It requires three broadcast communications to achieve security in asynchronous mode, which consumes a lot of resources. Moreover, the algorithm is less dynamic and does not have a perfect node joining and exiting mechanism. When nodes join and exit, the whole network needs to be restarted, resulting in a high overhead.

Related Works

In recent years, research on the improvement of the PBFT algorithm has increased significantly. For example, in terms of grouping, Li et al. (2022) combine PBFT with the Raft consensus algorithm, where the improved Raft algorithm is used for consensus within the group and PBFT consensus is used by leaders outside the group, thus ensuring efficient consensus while having Byzantine fault tolerance. Yang et al. (2022) proposed a highly fault-tolerant consistency algorithm, which uses a hashing algorithm to group nodes. The NBFT algorithm can avoid a large number of communications between nodes and reduce the communication complexity of the network. In addition, many other authors have grouped consensus nodes (Feng, L., et al., 2018; Li, Y., et al., 2021; Li, W., et al., 2021; Xu, X., et al., 2021), and although these algorithms improve the performance of PBFT, they tend to ignore the fault tolerance of the consensus network.

Many authors have proposed credibility models (Lei, K., et al., 2018; Zhang, M., et al., 2021); for example, Lao et al. (2021) select fixed, loyal, and capable nodes as master nodes to reduce the overhead of verifying and recording transactions, and the experimental results show that G-PBFT can significantly reduce the consensus time. The T-PBFT algorithm evaluates the trustworthiness of nodes through transactions between them so that highly reliable nodes in the network will together form a

consensus group (Gao, S., et al., 2019). In addition, a master group is used instead of a master node to further enhance the robustness of the blockchain by grouping signatures and mutual supervision. The SG-PBFT(Score Grouping-PBFT) consensus algorithm achieves higher consensus efficiency by optimizing the PBFT consensus process, and using a scoring grouping mechanism, it also effectively prevents single-node attacks (Xu, G., et al., 2022). Wang et al. (2019) replaced the C/S(Client/Server) structure of the original PBFT algorithm with a peer-to-peer (P2P) structure, reducing the negotiation steps, and used voting to elect the master node. The literature (Xu, X., et al. 2021; Liu, W., et al., 2022) introduces a node scoring mechanism to classify nodes according to their reliability marked with different trust states, which solves the problem of a lack of penalty mechanism for malicious nodes in the PBFT algorithm, but also leads to an increase of running time and transmission delay.

The main research of this paper is as follows.

First, the five steps of traditional PBFT consensus are changed into two parts, including intragroup local consensus and global consensus. Using the response speed of nodes to the group leader as the basis of grouping, an intragroup consensus is performed first, and then the group leader takes the result of intragroup consensus and participates in extra-group consensus, which can largely reduce the communication volume of nodes and effectively improve the consensus efficiency.

Second, to improve the reliability of master node election in the original PBFT consensus algorithm, the concept of reputation value is added, and the reputation value of the nodes is calculated in real time with reference to the performance of each node in the previous rounds of consensus. The nodes in different reputation value ranges are also divided into different levels of good and bad, and the nodes in different levels are given different weights in the voting election.

Third, simulation and experimental tests are conducted to verify the effectiveness and reliability of RPBFT in terms of throughput, consensus delay, and communication overhead.

RPBFT Algorithm Design

The RPBFT consensus algorithm is divided into two parts: local consensus within the group and global group leader consensus. Local consensus is first, and then the group leader participates in the global consensus with the local consensus result. If you assume that the number of nodes in the whole network is N, the number of nodes per group $K \geq 3$ and the number of groups $G \geq 4$ need to be satisfied. The main idea of the improved algorithm is shown in Figure 2.

Each group has a group leader, and the grouping is based on the response speed of each node to the group leader. The election of the group leader is decided by a combination of reputation points and a voting mechanism, and the group leader is elected from the well-performing nodes. The intragroup consensus will be conducted first, and then the group leader will take the results of the intragroup partial consensus and participate in the extra-group consensus at the end.

Grouping Strategy

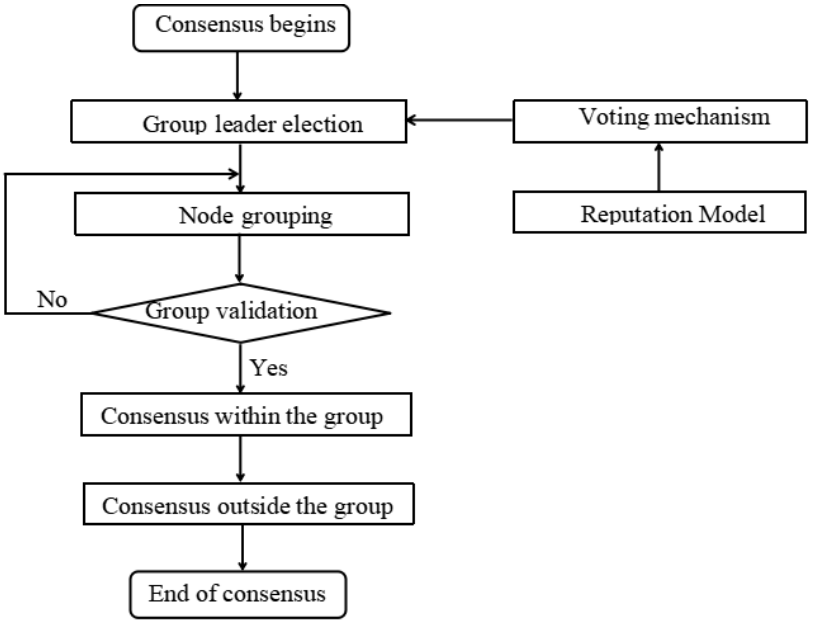
The identity verification mechanism of nodes joining the consortium chain is used to select G nodes as prepared group leaders, and nodes are divided into G groups based on their responses to the prepared group leaders as a grouping basis.

The prepared group leader $c \in (1,2,3,...,N)$ initiates a group probing message <SUBSET,t_1,S_c> to other nodes around it, where $t1$ is the timestamp and S_c is the signature of the prepared group leader c.

Node $x \in (1,2,3,...,N)$ receives a group probe message, verifies that the message signature is correct, and if it is correct, initiates a request to the prepared group leader <SUBSET-REQUEST,x,t_2,S_x>, where t_2 is the timestamp of node x and S_x is the current round signature.

Prepared group leader c checks if the group membership is greater than N_{max}, and if it is less than N_{max}, then verifies the information of node x. If the verification passes, it will receive the application of x to join the group and add it to the group list G_x. If the group membership is greater than N_{max} or the verification of node information does not pass, a rejection message is sent, where N_{max} is the maximum number of nodes in the group.

Figure 2. RPBFT Consensus Process



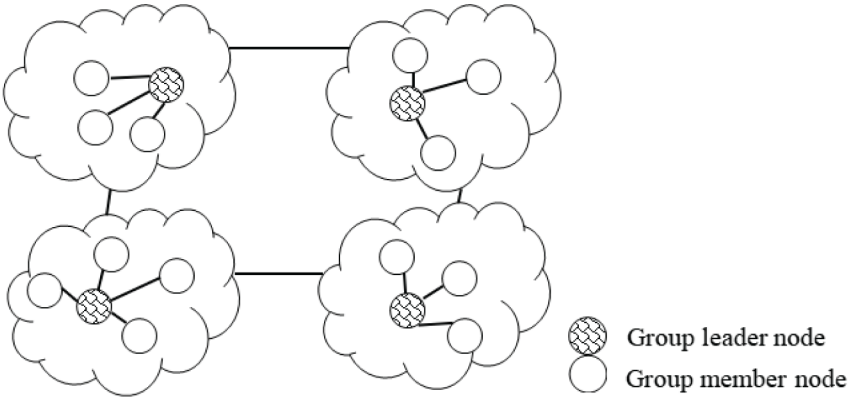
Node *x*, if it receives a reject message, can send a request <SUBSET-REQUEST,x,t_2,S_x> to the next prepared group leader and repeat the above steps.

At the end of grouping, the prepared group leaders broadcast the obtained group list among themselves, verify the node information, and distribute this information to each group member to complete the grouping confirmation, as shown in Figure 3. Note that when the grouping time exceeds the given value or a new node joins, the grouping needs to be restarted.

Reputation Model and Voting Mechanism

In the PBFT algorithm, the master node is elected with the current view number v+1, so all blockchain nodes have the chance to become the master node, and if a malicious node is elected as the master node, the performance of the system will be seriously affected. The reputation model proposed in this

Figure 3. Grouping Formation Process



paper can make the reliability of the elected master node much higher and reduce the influence of Byzantine nodes. As shown in equation (4), The node reputation value is defined as the probability of a node's normal participation in the consensus process in its history; the higher the reputation value, the more reliable the node is, and the opposite is true for lower reputation values. The following is the definition of reputation value:

- In round *t* consensus, when a new block is created, the reputation value of the group leader and the group members who have the same voting result as the group leader will increase.
- When the group leader does not generate a new block, the group members do not participate in the voting or if the voting result is different from the consensus result, then their reputation value decreases.
- When the group leader or group members send different messages to different nodes, the node is considered a node of evil, the node's trustworthiness is invalidated, and the reputation value is cleared.
- The reputation value of node *x*, $C_x(t)$, is set in the range [0,10], and the initial reputation value is 5:

$$C_x(t+1)=\begin{cases}0, \text{Send different messages to different nodes}\\ yC_x(t),(0<y<1)\begin{cases}\text{The group leader did not participate in the out-group consensus}\\ \text{Group members did not participate in the voting}\\ \text{Group voting differs from consensus results}\end{cases}\\ \min\left(C_x(t)\left(1+\frac{1}{t+1}\right),10\right),\begin{cases}\text{New block generation}\\ \text{Group members and group leaders have the same voting results}\end{cases}\end{cases} \tag{4}$$

The nodes participating in consensus calculate the reputation value of nodes based on the reputation model, add a status identifier for each node according to the reputation value, and set three different node states, all of which have different voting weights, as shown in Table 1. Among them, an "excellent" node is a stable, reliable node with good transactions; the "middle" node is a node with interruption or no response; and the "poor" node is a node with malicious behavior.

The election of the group leader is ultimately decided by a combination of reputation points and voting mechanisms, and the formula in equation (5) calculates the final score of the node:

$$S_x = C_x(t) * m + \left(\frac{1}{K}\sum_{i=1}^{K} weight_i * C_i(t)\right) * n \tag{5}$$

In equation (5) *K* is the total number of nodes in the group, $i \in (1,2,3,...,N)$ and $x \neq i$, *m,* and *n* are two parameters, and $m+n=1$. The final score of a node is divided into the basic score and voting score. The basic score is the product of the credibility value of the node and the parameter *m*, and the voting

Table 1. Node States and Weights

$C_x(t)$	Condition	Weight
[0,3]	Poor	0.5
[4,6]	Middle	1
[7,10]	Excellent	1.2

score is the product of the credit value of other nodes in the group and the weight. m=n=0.5 is chosen in this paper. Choosing this voting method can ensure that the node with high credibility becomes the group leader, guarantee the fairness of the voting election, and improve the security of the system.

The four specific steps are as follows:

Step 1: Broadcast the reputation values of the nodes in the group and update the list of reputation values.

Step 2: Vote for the group leader according to equation (5), with more than half of the votes cast being valid.

Step 3: The group leader initiates a data synchronization request; the message format is <<DATA-REQUEST,num,t_3,S_G>,data>, where *num* is the block number, t_3 is the timestamp, S_G is the group leader's signature, and *data* is the data to be synchronized. The group members receive the synchronization message and then check the signature and other information; if there is no objection, they update their backup data.

Step 4: The group member's data is synchronized successfully and replies to the group leader with a synchronization confirmation message <DATA-COMMIT,t_4,S_x>, where t_4 is the time stamp and S_x is the group member's signature. The data synchronization process is shown in Figure 4.

RPBFT Improves the Consensus Process

The RPBFT improves the consensus process via these seven steps:

Step 1: The order node collects the transaction requests sent by several clients <REQUEST,t,c,tx,S_c>, verifies that the transactions and signatures are correct, and then packages them into a block and sends them to the group leader <<OUT-PRE-PREPARE,n,t,m,S_o>D(m)>.

Step 2: The group leader receives the message to initiate a consensus proposal <IN-PREPARE,n,t,D(m),S_G> to all group members.

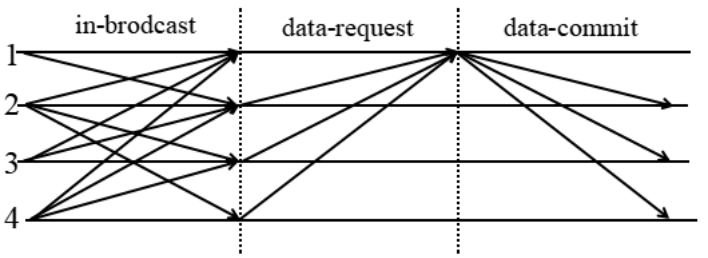
Step 3: The group member verifies the received proposal, and if it is correct, sends the response to the proposal <IN-COMMIT,n,t,D(m),S_x> to the group leader.

Step 4: The group leader confirms the responses from group members, verifies that the group consensus is complete, and all group leaders participate in the global consensus on behalf of each group, broadcasting <OUT-PREPARE,n,t,D(m),S_G> to all group leaders except themselves and entering the prepare phase.

Step 5: Each group leader verifies the received messages, discards them if there is any objection, and broadcasts <OUT-COMMIT,n,t,D(m),S_G> to other group leaders if more than 2f identical messages are received.

Step 6: The group leaders verify the received confirmation messages and reply to each member of the group <IN-REPLY,n,t,D(m),S_G> and to the order node <OUT-REPLY,n,t,D(m),S_G> if they receive more than 2f messages that pass the verification.

Figure 4. Group Leader Election and Data Synchronization



Step 7: The order node finally replies to the client <SUBMIT,n,t,D(m),S_o> and the consensus is reached.

In the RPBFT improved consensus process shown in Figure 5, *t* is the timestamp, *c* is the client number, *tx* is the specific transaction, S_c is the client signature, *n* denotes the block number, *m* denotes the message to be consensus, and *D(m)* is the summary of the message.

EXPERIMENTS AND RESULTS ANALYSIS

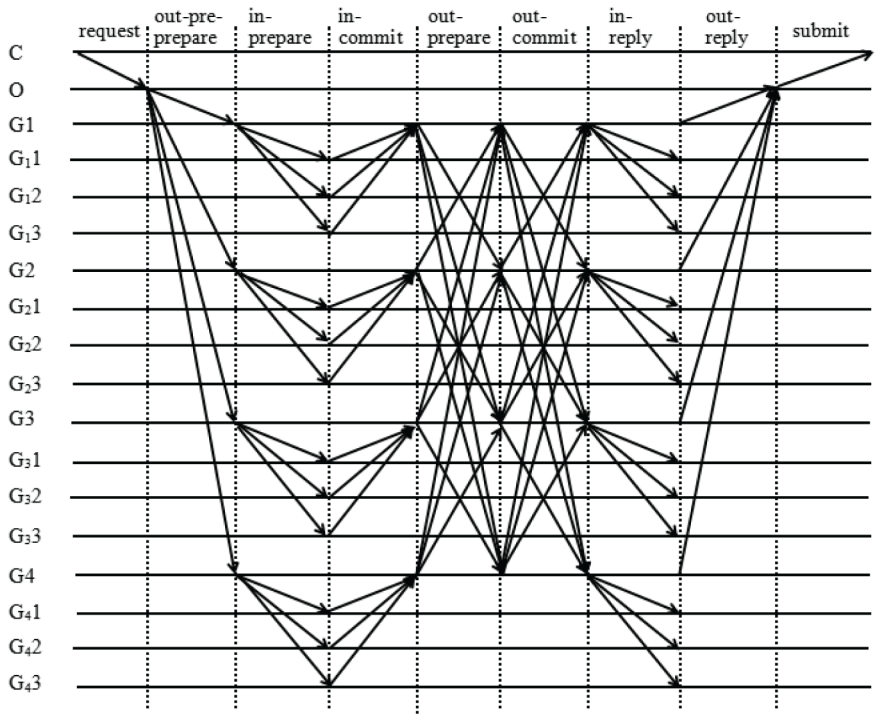
Experimental Environment

A blockchain system was simulated and implemented based on Java programming language. The initial conditions of the experiment are as follows: hardware conditions are an AMD R7 5800H CPU, equipped with 16GB of memory; a Windows 10 64-bit operating system; and an Eclipse 4.21 software environment. The RPBFT algorithm was verified in this system, and the performance of the algorithm was evaluated in terms of throughput, latency, communication overhead for 12, 16, 20, 24, 28, 32, 36, and 40 nodes, respectively. The test was conducted 100 times; each time, the client sent 200 request messages, and the average of 100 times was taken as the test result.

Consensus Time Delay

Consensus latency indicates the time required from the time the client sends a transaction request to the time the client completes a transaction confirmation. Low latency indicates that the consensus algorithm takes less time to execute, the blockchain is less prone to forking, and the system is more secure. The latency formula is shown in equation (6):

Figure 5. RPBFT Improved Consensus Process



$$Delay_{consensus} = T_C - T_R \tag{6}$$

In equation (6) T_C denotes the transaction confirmation time, and T_R denotes the transaction generation time. Under the same conditions, the time delays of the three algorithms of PBFT, DPBFT (Yao, Y., et al., 2022), and RPBFT are shown in Figure 6.

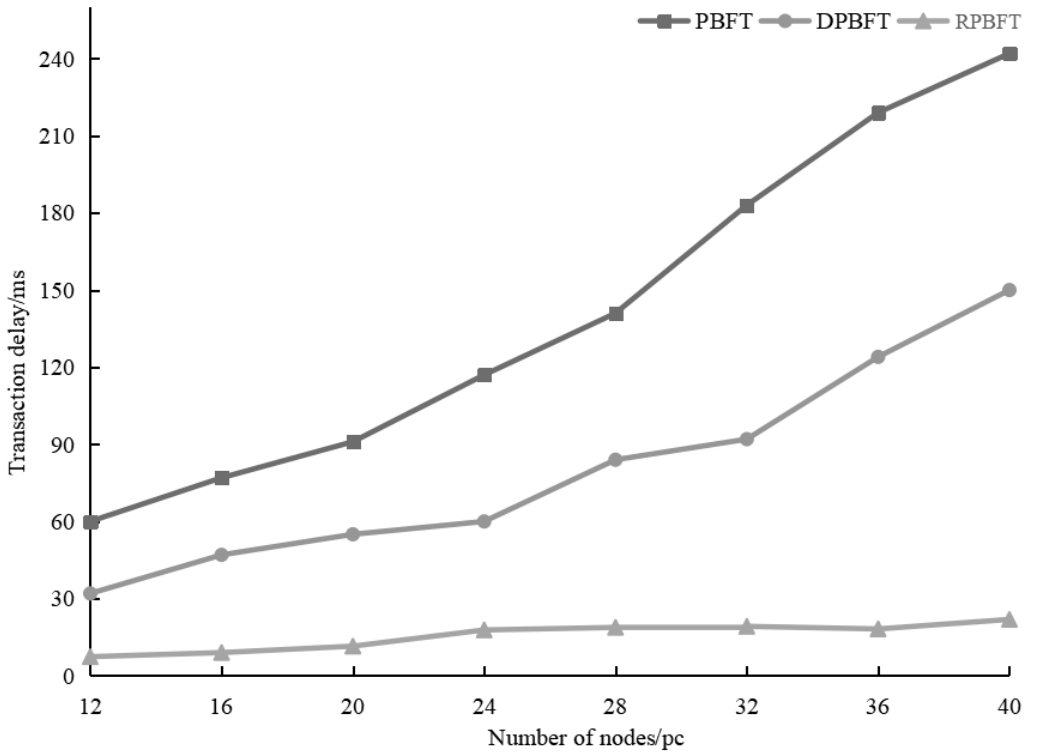
In Figure 6, the RPBFT algorithm latency is significantly better than the PBFT algorithm and the DPBFT algorithm. With the increase of consensus nodes, the increase of consensus latency of the RPBFT algorithm is significantly smaller than that of the PBFT and DPBFT algorithms, especially when the number of nodes exceeds 24 and the latency of PBFT increases sharply. The results indicate that the RPBFT algorithm also has a great improvement in latency.

Communication Overhead

Among the three main phases of the PBFT consensus process, in the pre-prepare phase, the master node broadcasts a message to all replica nodes with a communication count of (N-1) (where N is the number of nodes). In the prepare phase, the nodes send verification messages to all nodes except themselves with a communication count of $(N\text{-}1)^2$. In the commit phase, each node broadcasts the confirmation message to other nodes, and the number of communications is $N*(N\text{-}1)$. Therefore, the total number of communications for the PBFT algorithm is shown in equation (7):

$$T_1 = N - 1 + (N-1)^2 + N*(N-1) = 2N*(N-1) \tag{7}$$

Figure 6. Comparison of PBFT, DPBFT, and RPBFT Transaction Latency



The improved RPBFT algorithm adds a grouping phase to the process of node consensus as well as the calculation of node reputation values. In the RPBFT algorithm supervising nodes initiate proposals to group leaders with a communication count of G (where *G* is the number of groups). In the in-group prepare phase, group leaders broadcast messages to group members with a communication count of [(*N*-1)/*G*-1]**G*, and in the in-group commit phase, each group member sends a message to group leaders to confirm that the number of communications is [(*N*-1)/*G*-1]**G*. In the out-group prepare phase, group leaders broadcast messages to other group leaders except themselves, and the number of communications is (*G*-1)**G*. In the out-group commit phase, each group leader verifies the received messages and broadcasts them to other group leaders, and the number of communications is (*G*-1)**G*. The total number of communications for the RPBFT algorithm is shown in equation (8):

$$\begin{aligned} T_2 &= G + 2[(N-1)/G-1]*G + 2(G-1)*G \\ &= \left(2G-3\right)*G + 2(N-1) \end{aligned} \tag{8}$$

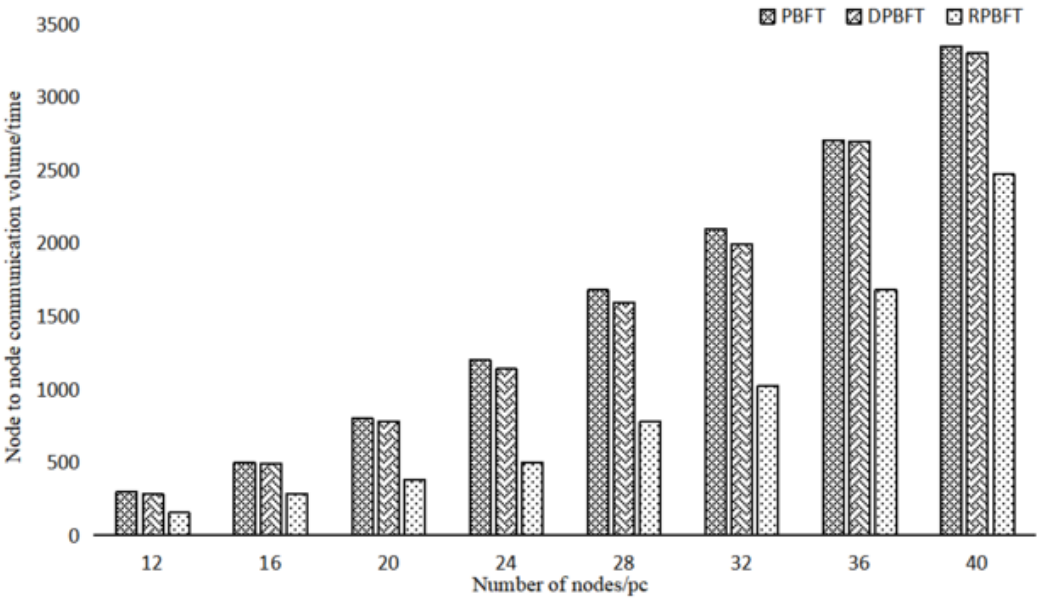
Suppose $G=N/K$, K denotes the number of members in the group and is an arbitrary constant greater than 3. Equation (8) is then reduced to the one shown in equation (9):

$$T_2 = N*\left(2N-3K+2K^2\right)*\left(\frac{1}{K^2}\right)-2 \tag{9}$$

When $N>12$, $T_2<T_1$, the improved communication time is less. Figure 7 shows the comparison of the number of communications among PBFT, DPBFT, and RPBFT with different numbers of nodes.

The figure shows that the communication overhead of RPBFT is much smaller than that of the PBFT and DPBFT algorithms, and the communication overhead increases slowly with the increase

Figure 7. Comparison of PBFT, DPBFT, and RPBFT Traffic



in the number of consensus nodes. When the number of network nodes is 36, the communication overhead of PBFT is 2,703, and RPBFT is 1,680, a 37.8% reduction in communication overhead.

Throughput

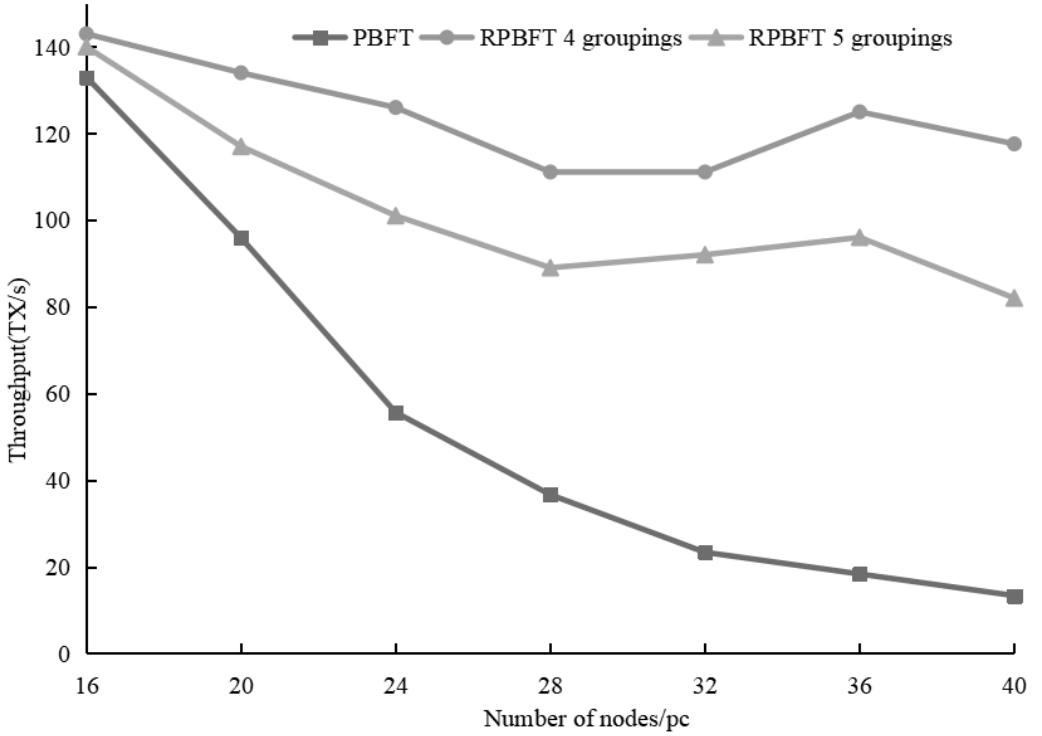
In blockchain systems, throughput is the number of transactions processed per unit of time, and a high-throughput system can process more transactions in a given amount of time. The calculation formula is shown in equation (10):

$$TPS = \frac{Transactions_{\Delta t}}{\Delta t} \tag{10}$$

In this formula $Transactions_{\Delta t}$ denotes the total number of transactions processed in Δt time, and Δt denotes the time interval between the issuance of a transaction and its uploading on the chain. In the experiment, the number of transactions completed in 1 second is calculated by setting different numbers of nodes, and the final results are shown in Figure 8.

Figure 8 shows that the improved RPBFT algorithm has significantly improved its throughput compared with the original PBFT algorithm, whether divided into four or five groups. As the number of nodes increases, the overall throughput of the algorithm shows a decreasing trend, and when the number of nodes is 24, the throughput of the PBFT algorithm decreases to 56, while the throughput of the RPBFT algorithm (four groups) is 128, an improvement of 128.6%.

Figure 8. PBFT, DPBFT, and RPBFT Throughput Comparison



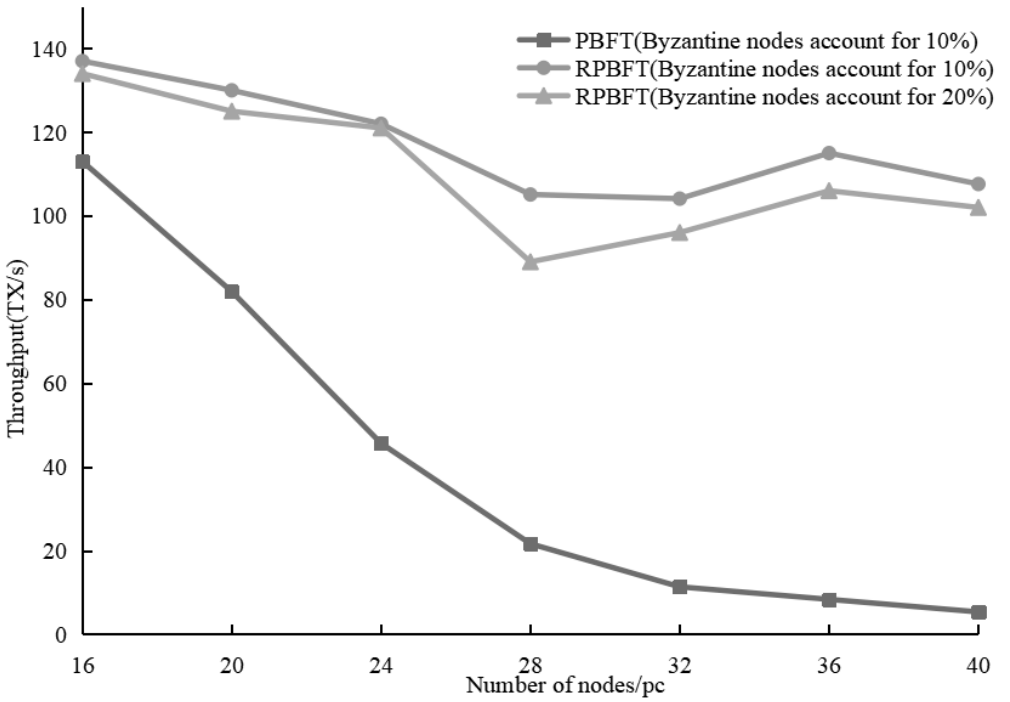
Security Analysis

The RPBFT algorithm is improved based on a PBFT mechanism. The algorithm does not change the core fault tolerance mechanism of the PBFT, so the final consensus result of each group at the bottom, group representative malicious node fault tolerance, is the same as the Byzantine attack tolerance of the PBFT. However, the RPBFT algorithm is based on the reputation model and voting mechanism of the node grouping mechanism so that more Byzantine nodes can be tolerated within the group. Therefore, by using the RPBFT consensus protocol under the same conditions, the blockchain network will improve fault tolerance after a period of consensus, increasing consensus efficiency and overall system trustworthiness. Figure 9 shows the throughput comparison between PBFT and RPBFT (with 10% and 20% of Byzantine nodes in the network, respectively) for different nodes. Note that the reputation model and the voting mechanism help the system to exclude the influence of some malicious nodes, making no significant change in the system's throughput after increasing the number of Byzantine nodes in the RPBFT algorithm.

CONCLUSION

In this paper, the RPBFT algorithm based on reputation value voting is proposed to address the problems of arbitrary selection of master nodes and the high communication complexity of the PBFT consensus algorithm. The message response speed is used as the basis for grouping, and nodes are divided into groups. The group leader first participates in the out-group consensus with the results of the in-group consensus. The election of the group leader adopts the reputation model and voting mechanism and gives different voting weights according to the reputation value, which can effectively reduce the possibility of abnormal nodes as the group leader nodes. Through simulation experiments,

Figure 9. PBFT, RPBFT Throughput Comparison Under Different Byzantine Nodes



the simulation results show that the RPBFT algorithm has significant improvements in throughput, latency, and communication overhead compared with the PBFT and DPBFT algorithm. Further research will be done in the future on the security aspects of the consensus algorithm to continue to optimize the algorithm in this paper and promote the development of the consortium chain.

FUNDING AGENCY

Open Access Funding for this article has been covered by the authors of this manuscript.

ACKNOWLEDGMENT

This research was supported by the National Natural Science Foundation of China (NSFC) under Grant 62141205 and the Fund Project of XJPCC(2022CB002-08, 2022CA007, 2019AB001).

REFERENCES

Alofi, A., Bahsoon, R., & Hendley, R. (2021). MinerRepu: A reputation model for miners in blockchain networks. In 2021 IEEE International Conference on Web Services (ICWS). IEEE. https://doi.org/10.1109/ICWS53863.2021.00100

Feng, L., Zhang, H., Chen, Y., & Lou, L. (2018). Scalable dynamic multi-agent practical Byzantine fault-tolerant consensus in permissioned blockchain. *Applied Sciences-Basel, 8*(10).

Fu, X., Wang, H., & Shi, P. (2021). A survey of blockchain consensus algorithms: Mechanism, design, and applications. *Science China. Information Sciences*, *64*(2).

Gao, S., Yu, T., Zhu, J., & Cai, W. (2019). T-PBFT: An EigenTrust-based practical Byzantine fault tolerance consensus algorithm. *China Communications*, *16*(12), 111–123.

Gao, Z., & Yang, L. (2020). Optimization scheme of consensus mechanism based on practical Byzantine fault tolerance algorithm. *Communications in Computer and Information Science*, *1176*, 187-195.

Gu, R., Chen, B., & Huang, D. (2021). Primary node selection algorithm of PBFT based on anomaly detection and reputation model. *Lecture Notes in Electrical Engineering, 808*, 767-776. 10.1007/978-981-16-6554-7_178

Lao, L., Dai, X., Xiao, B., & Guo, S. (2020). G-PBFT: A location-based and scalable consensus protocol for IoT-blockchain applications. In 2020 IEEE 34th International Parallel and Distributed Processing Symposium (IPDPS). IEEE. https://doi.org/10.1109/IPDPS47924.2020.00074

Lei, K., Zhang, Q., Xu, L., & Qi, Z. (2018). Reputation-based Byzantine fault-tolerance for consortium blockchain. In 2018 IEEE 24th International Conference on Parallel and Distributed Systems (ICPADS). IEEE. https://doi.org/10.1109/PADSW.2018.8644933

Li, C., Zhang, J., & Yang, X. (2022). Scalable blockchain storage mechanism based on two-layer structure and improved distributed consensus. *The Journal of Supercomputing*, *78*, 4850–4881.

Li, W., Feng, C., Zhang, L., Xu, H., Cao, B., & Imran, M. A. (2021). A scalable multi-layer PBFT consensus for blockchain. *IEEE Transactions on Parallel and Distributed Systems*, *32*(5), 1146–1160.

Li, Y., Qiao, L., & Lv, Z. (2021). An optimized byzantine fault tolerance algorithm for consortium blockchain. *Peer-to-Peer Networking and Applications*, *14*, 2826–2839. https://doi.org/10.1007/s12083-021-01103-8

Liu, W., Zhang, X., Feng, W., Huang, M., & Xu, Y. (2022). Optimization of PBFT algorithm based on QoS-aware trust service evaluation. *Sensors (Basel)*, *22*(12).

Liu, Y. A., & Stoller, S. D. (2019). From classical to blockchain consensus: What are the exact algorithms? *Proceedings of the 2019 ACM Symposium on Principles of Distributed Computing*, 544-545.

Sanka, A. I., & Cheung, R. C. C. (2021). A systematic review of blockchain scalability: Issues, solutions, analysis, and future research. *Journal of Network and Computer Applications*, *195*, 103232. Advance online publication. doi:10.1016/j.jnca.2021.103232

Wang, Y., Song, Z., & Cheng, T. (2020). Improvement research of PBFT consensus algorithm based on credit. In Z. Zheng, H. N. Dai, M. Tang, & X. Chen (Eds.), *Blockchain and Trustworthy Systems, Blocksys 2019. Communications in Computer and Information Science* (Vol. 1156, pp. 47–59). Springer.

Xu, G., Bai, H., Xing, J., Luo, T., Xiong, N. N., Cheng, X., Liu, S., & Zheng, X. (2022). SG-PBFT: A secure and highly efficient distributed blockchain PBFT consensus algorithm for intelligent Internet of vehicles. *Journal of Parallel and Distributed Computing*, *164*, 1–11.

Xu, X., Zhu, D., Yang, X., Wang, S., Qi, L., & Dou, W. (2021). Concurrent practical Byzantine fault tolerance for integration of blockchain and supply chain. *ACM Transactions on Internet Technology*, *21*(1).

Yang, J., Jia, Z., Su, R., Wu, X., & Qin, J. (2022). Improved fault-tolerant consensus based on the PBFT algorithm. *IEEE Access: Practical Innovations, Open Solutions*, *10*, 30274–30283.

Yao, Y., Yan, Y., Guo, S. Y., Xiong, A., & Zhang, W. (2022). Distributed and efficient identity authentication based on consortium blockchain. *Journal of Application of Electronic Technique*, *48*(03), 104–108.

Yu, G., Wu, B., & Niu, X. (2020). Improved blockchain consensus mechanism based on PBFT algorithm. In *Proceedings of 2020 2nd International Conference on Advances in Computer Technology, Information Science and Communications (CTISC 2020).* IEEE. 10.1109/CTISC49998.2020.00009

Zhang, M., Cheng, Y., Deng, X., Wang, B., Xie, J., Yang, Y., & Zhang, J. (2021). Accelerating transactions relay in blockchain networks via reputation. In 2021 IEEE/ACM 29th International Symposium on Quality of Service (IWQOS). IEEE. https://doi.org/10.1109/IWQOS52092.2021.9521324

Zheng, X. (2022). Research on news copyright management and protection mechanism based on blockchain technology. *Proceedings of 3rd International Symposium on Frontiers of Economics and Management Science (FEMS 2022),* 543-548. 10.54691/bcpbm.v19i.846

Shaanan Liu, female, master's student, born in March 1998, received her bachelor's degree in Computer Science and Technology in 2016. Her main research interests include machine learning, distributed computing, blockchain, and consensus algorithms.

Ronghua Zhang, female, master, associate professor, born in December 1980, main research interests include big data, blockchain.

Changzheng Liu, Male, PhD, Professor, born in October 1979, received his PhD degree in Computer Application Technology from Huazhong University of Science and Technology in 2020. His main research interests include computer application, signal processing, blockchain, etc.

Chenxi Xu, School of Economics and Management, Qilu Normal University, His areas of interest are signal processing, blockchain.

Jie Zhou, Male, PhD, Associate Professor, received the Doctor of Philosophy degree in Wireless Communication from Macquarie University, Australia in 2016. His main research interests include artificial intelligence, cyberspace security, etc.

Jiaojiao Wang, female, Master's degree student, born in November 1996, received her Bachelor's degree in Computer Science and Technology in 2016. Her area of interest is computer vision perception, machine learning.