A PBFT Consensus Scheme with Reputation Value Voting Based on Dynamic Clustering

Shenchen Zhu[1], Ziyan Zhang[1], Liquan Chen[1,2](✉), Hui Chen[1], and Yanbo Wang[1]

[1] School of Cyber Science and Engineering, Southeast University, Nanjing 211100, Jiangsu, China
lqchen@seu.edu.cn

[2] Purple Mountain Laboratories for Network and Communication Security, Nanjing 211111, Jiangsu, China

Abstract. At present, the consensus algorithm based on reputation voting generally has the problem of credit value accumulation caused by Matthew effect, which will lead to the risk of system centralization. Therefore, we propose a new blockchain consensus scheme based on PBFT mechanism, which divides the nodes into three categories: production node, upper node and common node, and the first two types are generated by node selection algorithm and replaced regularly. In the node selection algorithm, random parameters are introduced to make the reputation value no longer the only standard. In addition, in order to solve the problems of high message complexity and poor scalability shortcomings in PBFT, we use ISODATA algorithm to segment the nodes in the system, and simplify the consensus process of these existing PBFT algorithm, which greatly reduces the message complexity of the consensus processing without compromising the fault-tolerant performance of the system.

Keywords: Blockchain · PBFT · Dynamic clustering · Reputation model

1 Introduction

As the underlying core technology of Bitcoin, blockchain is essentially considered as a decentralized distributed ledger. With the core advantages of decentralization, blockchain has huge development potential in the fields of finance, IoT, healthcare, privacy data management and so on [1–3]. Consensus mechanism is an important part of blockchain technology, and its quality directly affect the performance of the system. Although public blockchain algorithms such as Pow [4], PoS [5], and DPoS [6] have good security, they still have some problems including high time-delay and low throughput capacity. Paxos [7] and Raft [8] are traditional representative distributed consensus algorithms, which are mainly oriented to databases, logs and other underlying storage areas however, the Byzantine-fault-tolerance problem was not considered in these algorithms. The PBFT algorithm [9] solves the Byzantine-fault-tolerance problem well, but the nodes cannot dynamically join the system while PBFT is running, and its scalability needs to be improved. Scalability will be one of the biggest challenges in the application

S. Yu et al. (Eds.): SPDE 2020, CCIS 1268, pp. 336–354, 2020.
https://doi.org/10.1007/978-981-15-9129-7_24

of blockchain technology [10]. Although there are some problems with the PBFT algorithm, due to its advantages of low time-delay, low energy consumption, no bifurcation, and resolution of Byzantine-fault-tolerance, PBFT has a better overall performance and is more suitable for most customized application scenarios [11].

Among the related researches on the improvement of the PBFT algorithm, the research based on credit and voting has a relatively good performance in balance of the scalability, security and decentralization. However, the following two problems still exist:

1) *Problem of centralization tendency caused by time accumulation*: When the system continues running for a long period of time, some nodes increase their reputation value by holding positions with functions such as voting and production. As the reputation value is improved, these nodes will have an advantage in the next election. The repetition of this process will continuously speed up the improvement of the reputation value of these nodes, and eventually lead to the centralization of the entire system. The CDBFT [12] algorithm proposed by Y. Wang et al. establishes a privilege classification mechanism for nodes, which effectively prevents "Expected" nodes from being elected as master nodes. "Normal" nodes can be elected as primary nodes only after all "Credible" nodes have been elected or are not eligible for vote [12]. It is difficult for "Normal" nodes to obtain equal voting rights; therefore, the centralization trend of the system is inevitable. The CPBFT [13] algorithm proposed by Y. Wang and Z. Song et al. divides nodes by the credit rating and assigns corresponding credit coefficients to different levels of nodes. The election of the master node will continue to favor the "A" node. Other algorithms [14, 15] based on credit and voting also have this problem.
2) *Problem of overall efficiency*: In the existing related research, the system often defines an overly complicated role system and election mechanism. The VPBFT [16] algorithm defines four roles of nodes: production node, voting node, ordinary node, and candidate node. The number of nodes in all roles changes dynamically in real time. The system will consume a lot of unnecessary resources in role allocation, role switching and role statistics. The vBFT algorithm [17] simplifies the roles: only defines three roles of master, client, and slave. However, the global voting mechanism and the design of the global data pool still restrict the scalability. The CoT algorithm [14] is based on P2P architecture. By generating a credit graph and a credit matrix, the credit value of nodes is calculated to select "delegated" nodes and perform PBFT consensus between "delegated" nodes. The cost of iterative calculation of credit values and consensus stage of "delegated" nodes is relatively large.

In order to solve the above problems, we put random factors in the election process. Voting is randomly conducted to select some candidate nodes, of which the node with the highest reputation value can be selected. Combined with the mechanism of reputation value update, each node has an equal opportunity to be elected, and the malicious nodes are limited. Therefore, a decentralized fair election that is not affected by running time is realized. In addition, in order to reduce the overall complexity of the system, we will use dynamic clustering as the basis to limit the voting and election activities within the group, and conduct production voting activities only in the entire network.

By combining the POV mechanism [18] with the PBFT algorithm, this paper proposes a PBFT consensus mechanism (RC-VPBFT) based on dynamic clustering and reputation value voting. RC-VPBFT will adopt the idea of POV mechanism, simply divide nodes into three roles, establish a reputation value evaluation system, and design a new election algorithm so that the reputation value is no longer the only criterion for node selection, which avoid the centralization tendency of system with time accumulation. In addition, the RC-VPBFT mechanism will use the network delay between nodes as the "node distance", cluster the nodes dynamically, and simplify the traditional PBFT algorithm. Therefore, without reducing the fault-tolerance performance of the system, the message complexity of the achieving consensus is greatly reduced. Finally, we also design a new network extension protocol to support the dynamic joining and exiting of nodes in RC-VPBFT.

2 RC-VPBFT Mechanism Model

In order to improve the operating efficiency of the system, the RC-VPBFT mechanism uses dynamic clustering algorithms to group nodes, and each group is called a "consensus cluster". Production activities are performed on a global scale, and reputation value updates and node elections activities are performed within the consensus cluster.

2.1 Roles for Nodes

The topology of the RC-VPBFT mechanism is shown in Fig. 1. There are three types of nodes in each consensus cluster. Different kinds of nodes supervise and restrict each other, jointly maintain balance of the system:

1) *Ordinary node*: The Ordinary node is responsible for selecting the superior node, accepting and responding to queries from the production node, as well as impeaching the superior node and production node which has malicious behavior. All network nodes joining the system have the identity of ordinary node.
2) *Superior node*: The superior node is responsible for not only selecting production nodes but also accepting and responding to queries from production nodes. The superior nodes are selected by a mechanism that combines the random recommendation of ordinary nodes and the comparison of reputation values.
3) *Production node*: The production node is responsible for confirming transactions, packaging blocks, and extending the blockchain. It is selected by a mechanism that combines the random recommendation of superior nodes and the comparison of reputation values.

2.2 Running Framework

The algorithm flow of the RC-VPBFT mechanism is shown in Fig. 2. RC-VPBFT takes "round" as the unit during the execution process, and the complete process from the preparation stage to the end stage is called a round. Each consensus cluster generates a block after a round. As shown in Fig. 2, each round of execution process is divided into three stages:

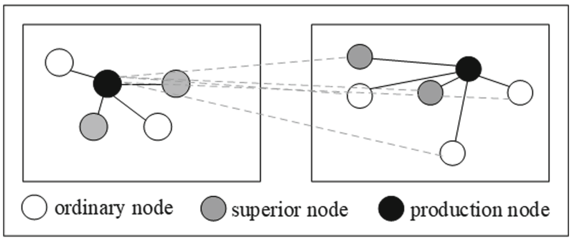


Fig. 1. Topological structure of consensus mechanism

1) *Preparation stage*: Firstly, group the nodes to determine whether the ISODATA algorithm needs to be executed at this time. If necessary, execute the ISODATA algorithm, otherwise proceed to the next step. Then determine whether superior node recommendation is needed at this time. If needed, the superior node recommendation algorithm will be executed, otherwise the next step will be taken. Finally, determine whether production node recommendation is needed at this time. If there is a need, the production node recommendation algorithm should be executed, otherwise it will enter the consensus stage.
2) *Consensus stage*: To start with, according to the improved PBFT algorithm, verify the transactions in the consensus cluster by voting. Then package all the verified transactions and generate a block after the number of transactions verified by voting is greater than the set value, or after the runtime is greater than the set value.
3) *Final stage*: In the first instance, verify the survival of all nodes in the cluster. Then process the join request of the new node in this round. If there is no join request, skip this stage and enter a new round.

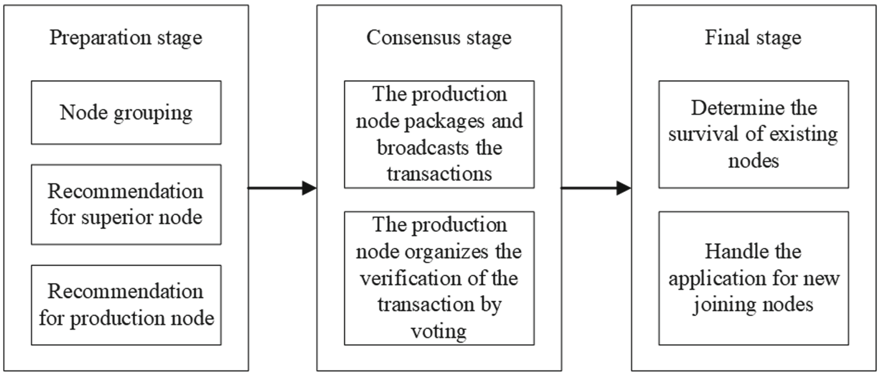

Preparation stage
Node grouping
Recommendation for superior node
Recommendation for production node
Consensus stage
The production node packages and broadcasts the transactions
The production node organizes the verification of the transaction by voting
Final stage
Determine the survival of existing nodes
Handle the application for new joining nodes

Fig. 2. Block diagram of algorithm flow

3 RC-VPBFT Mechanism

The main improvements of the RC-VPBFT mechanism are shown in Fig. 3, including reducing the system centralization trend, improving system operating efficiency, supporting system dynamic expansion, and enhancing system security performance.

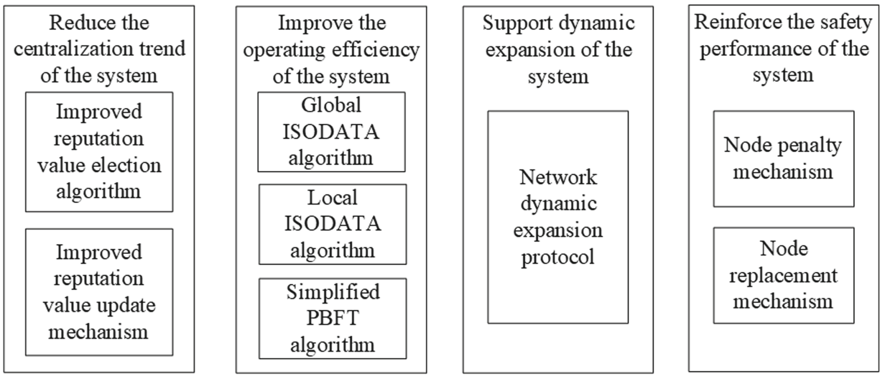

Reduce the centralization trend of the system
Improved reputation value election algorithm
Improved reputation value update mechanism
Improve the operating efficiency of the system
Global ISODATA algorithm
Local ISODATA algorithm
Simplified PBFT algorithm
Support dynamic expansion of the system
Network dynamic expansion protocol
Reinforce the safety performance of the system
Node penalty mechanism
Node replacement mechanism

Fig. 3. Main improvements of RC-VPBFT

3.1 Network Dynamic Expansion Protocol

Dynamic Join of Nodes
When a new node (called A) tries to join a running blockchain network, it needs to know the IP information of at least one running node (called B) in the network, then Node A sends a join request to node B. After receiving A's request, B feeds back the IP information of all nodes in the consensus cluster where B is located to A. A uses the obtained IP information to issue connection requests in sequence.

When the running node in the network receives the connection request of the new node, it puts information of the new node IP into the temporary node list. The nodes in the temporary node list do not participate in the consensus behavior in the consensus cluster.

When nodes in a consensus cluster run to the final stage, all nodes add the IP information in the temporary node list to the official node list, and these newly joined nodes will participate in the consensus from the next round.

3.2 ISODATA Dynamic Clustering Algorithm

Global ISODATA Dynamic Clustering Algorithm
PBFT consensus enables transaction verification. In order to ensure the security of the system, the PBFT consensus must be maintained globally. However, node activities such as reputation value update, node election, and node replacement do not require

the consensus throughout the network. Therefore, the consensus mechanism proposed in this paper uses the ISODATA algorithm [19] to group the nodes, which transfer the reputation value update, node election and node replacement activities from the entire system to each group. That can reduce the complexity of the election activities and improve performance of the system.

Since the number of nodes in the system changes dynamically, the number of groups is not fixed, so ISODATA is more suitable for blockchain environment than the K-means algorithm [20] and K-medoids algorithm [21], which needs to determine the number of groups in advance.

In actual operation, RC-VPBFT uses the number of nodes as the dimension and the round-trip time returned by ping as the distance to perform clustering operations, divides the nodes with lower time delay of mutual communication into a group to improve the communication efficiency as much as possible.

Improved Local ISODATA Dynamic Clustering Algorithm

Although ISODATA dynamic clustering algorithm has good grouping performance, it is too complicated, and the overhead of continuous calculation is too large. The dynamic clustering algorithm is only introduced for node grouping in this paper, the accuracy of grouping does always not need to maintain. In this regard, this paper proposes an improved local ISODATA dynamic clustering algorithm for single group splitting. The specific algorithm is designed as follows:

a) Groups with a long network delay or large number of nodes are likely to cause congestion, which slows down the operation of the entire system. Therefore, these groups need to be split to improve efficiency. Groups with lower network delay or fewer nodes have some waste of resources but does not affect the overall operation of the system. Therefore, there is no need to constantly control the size of these groups through merging. In summary, the main purpose of the local algorithm is to split the single group that have an adverse effect on system efficiency.
b) The distance in the global algorithm is the network delay between each node, and the dimension is represented by the number of nodes. For the reason of the local algorithm needs to be executed in real time, the definition of this distance cannot meet the real-time requirements. In the steps such as standard deviation calculation, only the distance from the given node to the central node is used, so the multidimensional vector can be simplified to one dimension. Therefore, the distance in the local algorithm is defined as the network delay from the node to the central node, and the central node is defined as the node with the smallest total network delay.
c) On account of the basis of clustering is the network delay time, the standard deviation vector of a single group reflects the communication quality within the group to a certain extent and can be used to split a single group with a long network delay and a large number of nodes.
d) When the average time for the nodes in the group to reach consensus over a period exceeds the threshold, the system is decided to be in an inefficient state, and a local ISODATA algorithm will be triggered.

Through the above design, the local ISODATA dynamic clustering algorithm maintains a good bipartite splitting effect, while achieving vector mapping from multi-dimensional to one-dimensional, greatly improving the efficiency of local clustering. The specific algorithm is as follows:

Step 1. Initialization
The initialization parameters is set as Table 1:

Table 1. Initialization parameters of local ISODATA algorithm

Parameter	Description
θ_N	Minimum sample size in each cluster that affect merging
N	Number of samples in current group
α	Weight of network delay that affects splitting
$\mathrm{z_j}$	Center sample of group j
T_l	Seconds of greenwich mean time of the last global algorithm
T	Current seconds of greenwich mean time

Step 2. Global Judgment
If $T - T_1 \geq 604800$, set T_1 to T, jump out, and go to the global ISODATA algorithm;
Otherwise, continue execution.

Step 3. Preliminary Judgment
If $N \leq 2\theta_N$, the group does not have the conditions for splitting, jump to step 8;

Step 4. Standard Deviation Calculate
For group j, calculate the standard deviation of the group: $\sigma_j = \sqrt{\frac{1}{N}\sum_{y_k \in S_j} y_{ki}^2}$,, where S_j is a set of single group, y_k is the i-th component of the k-th sample in the j-th category, and i is the serial number of sample in z_j.

Step 5. Merge/Split Judgment
If $N \geq \frac{\alpha\theta_N}{\sigma_j}$,continue execution;
Otherwise, skip to step 8;

Step 6. Split
Split S_j into two groups, the center of which is z_{j+} and z_{j-}. the calculation methods of z_{j+} and z_{j-} are as follows: given a value of k, 0 < k < 1, let $r_j = k\sigma_j$; then $z_{j+} = z_j + r_j$, $z_{j-} = z_j - r_j$, where the value of k should make the distance different from the samples in Sj to z_{j+} and z_{j-}. Meanwhile, k needs to make the samples in S_j still in the split new sample class.

Step 7. Parameter Update
Update the number of samples N in the group.

Update the single group center z_j through calculation $z_j = \max_{y_k \in S_j}\left\{\sum_{i=1}^{N} y_{\mathrm{ki}}\right\}$.

Step 8. Termination Judgment
Over a period, monitor the average time for nodes in the group to reach consensus. If the trigger condition is still met, return to step 4 and obtain a second splitting opportunity for group with samples smaller than the minimum sample size;

Otherwise, the algorithm ends.

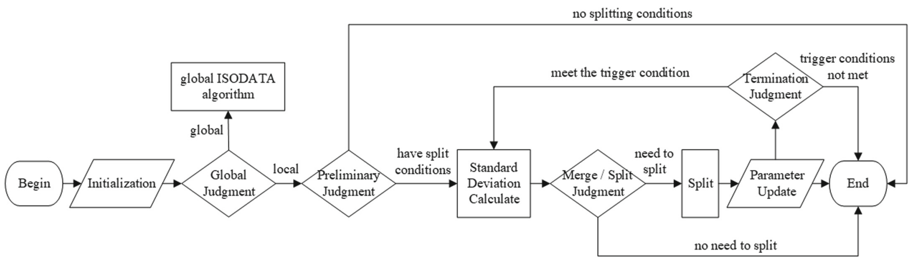


Fig. 4. Local ISODATA algorithm flowchart

The overall structure of the local ISODATA algorithm is shown in Fig. 4. The global judgment avoids the repeated operation of the global algorithm and the local algorithm. The preliminary judgment completes the pre-pruning in most cases, and the termination judgment guarantees the efficiency of the algorithm in extreme cases where the delay is too large.

Through the execution of the above steps, RC-VPBFT uses the local ISODATA algorithm to split the consensus cluster during normal operation; uses the global ISODATA algorithm occasionally to obtain the best division scheme for the consensus cluster. It not only avoids the huge overhead and improves the operating efficiency of the system, but also ensures the rationality of node grouping to a certain extent.

3.3 Mechanism of Reputation Value Update

The mechanism of reputation value update can effectively reduce the adverse effects of malicious nodes on the system and encourage nodes to comply with system regulations [22]. the reputation value is set to update before the superior node election. The nodes in the group send out a list which contains the reputation value of whole group, and each node updates the reputation value of the nodes according to most of the received messages.

Definition 1. The reputation value of newly added node is 1.

Definition 2. The node with a non-positive reputation value is regarded as a malicious node, and it will be actively driven away from the system.

Definition 3. After the generation of a valid block is completed, the reputation value of all nodes increases, and the increasing can be represented by:

$$R_{i+1} = lg(10^{R_i} + p) \tag{1}$$

where p is the growth factor, which can be set to 10. the logarithmic model is chosen to make the growth rate of reputation value faster in the early stage, while gradually decreases in the later stage.

Definition 4. When the node has malicious behavior, the reputation value decreases, and the reduction can be written as:

$$R_{i+1} = \begin{cases} R_i - q_1 & R_i \le \lambda \\ q_2 R_i & R_i > \lambda \end{cases} \tag{2}$$

The definition of malicious behavior is shown in Table 2, where q_1 is the low-speed deceleration factor, q_2 is the high-speed degeneration factor, and λ is the deceleration limit. In this paper, q_1 is set to 0.1, while q_2 is set to 0.5, and λ is set to 2.

Table 2. Definition of malicious behavior

Node type	Malicious behavior
Production node	Data verification and packaging not completed within the specified time Tampering with verification results
Superior node	Publish fake voting results Initiate incorrect impeachment of the production node
Ordinary node	The judgment is inconsistent with the final voting result Initiate incorrect impeachment of the production node or superior node

With a piecewise linear regression model, RPBFT can instantly adjust the reputation value of nodes according to their behavior. We run a network with 200 nodes in which there are 16 evil nodes and we record the fluctuations of reputation as Fig. 5. It can be seen from the experiment that the decline of the reputation value is much faster than the rise of the reputation value, furthermore, the rising trend of reputation value gradually slowed. Under this mechanism, the increase of reputation value is a long-term process, but the reduction of reputation value is very easy, which makes each node cherish its own reputation value and take cautious actions.

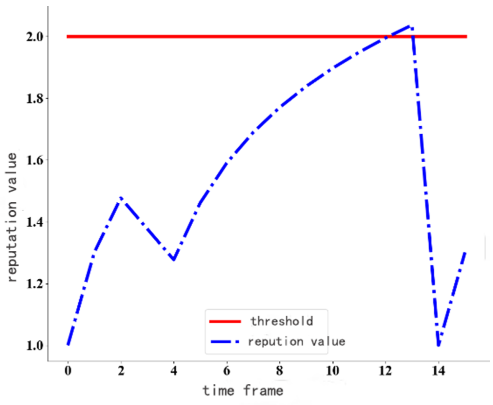


Fig. 5. Change of node reputation value

3.4 Simplified PBFT Algorithm

The traditional PBFT algorithm consists of five steps. When there are n nodes in the system, the number of messages required to reach a consensus is approximately equal to 2n ˆ 2. As the number of nodes in the system increases, the number of messages required to reach a consensus will increase in square order. The explosive increase in the number of messages will greatly extend the time it takes to reach consensus, thereby becoming a bottleneck in system performance and limiting the size of the system.

In order to solve this problem, this paper simplifies the traditional PBFT algorithm as shown in Fig. 6, node 0 is the master node and nodes 1 to 3 are slave nodes.

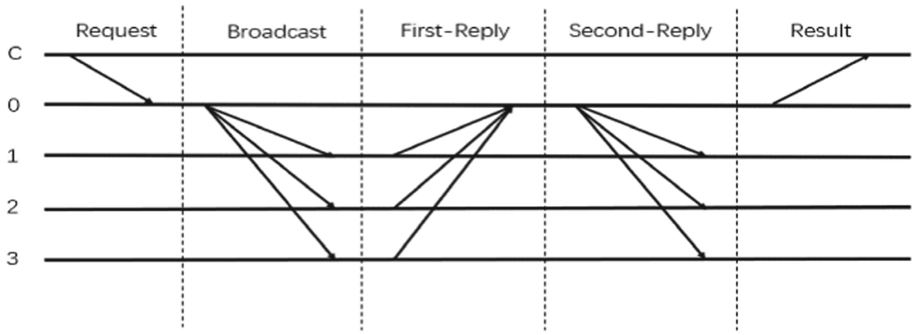


Fig. 6. Simplified PBFT consensus flow chart

Compared with the PBFT algorithm (Fig. 7), we have eliminated the complex message broadcasting in the PERPARE phase and the COMMIT phase. Each node makes its own judgment, and the master node summarizes all the judgments and makes a final decision.

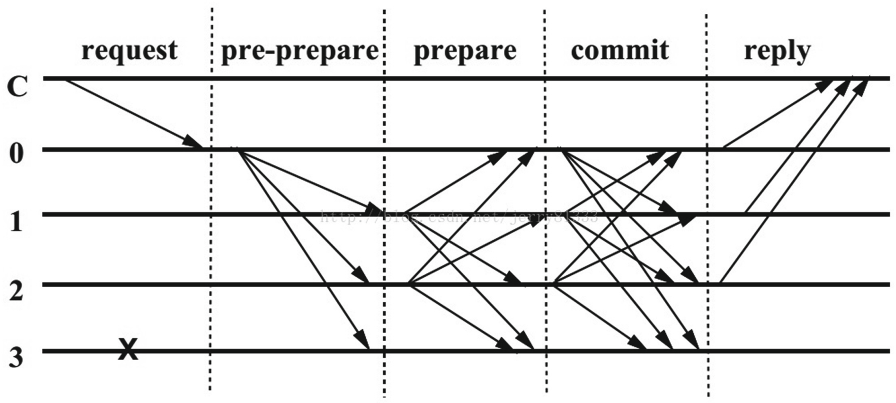


Fig. 7. PBFT consensus flow chart

Overall, the Simplified PBFT algorithm mainly includes the following two steps.

1) Each slave node signs its own judgment and sends it directly to the master node.
2) The master node summarizes all the judgments, makes a final judgment by majority rule, and broadcasts the following content to all slave nodes.

   a) The judgment of each node
   b) Final decision.

Since the judgment of each slave node has been signed by itself, the master node can only count the judgments but cannot forge or tamper the judgments. After simplification, the complexity of the message is reduced from O (n ˆ 2) to O (n) with enough security, thereby greatly improving the efficiency of the algorithm and the scalability of the system.

Combined with the simplified PBFT algorithm, the steps for us to reach consensus are as follows:

1) When a node in a consensus cluster attempts to initiate a transaction, it will sign the transaction and broadcast it in the consensus cluster.
2) After receiving the broadcast transaction, the production nodes with sign the transaction and forward it to the random m nodes in all consensus clusters to request verification. The signed message has the form of $K_{P_i}\langle timestamp, T\rangle$, where K_{P_i} is the private key of the production node. The size of m can be adjusted freely to meet different requirements for system security. The security of the system increases with the increase of m, and accordingly, the time required to reach consensus also increases with the increase of m.
3) After receiving the message from the production node, the m nodes in step 2 make their own judgments on the transactions contained in the message, and sign the judgments then send them back to the production node. The signed message has the form of $K_{N_i}\langle timestamp, T, judgement\rangle$, where K_{N_i} is the private key of the node.

4) When the production node receives more than or equal top judgments that agree with the transaction, the transaction is deemed to have passed verification, and the system reaches consensus on this message. The size of p can be adjusted freely to meet different requirements for system security.

When the number of verified transactions stored in the production node reaches a certain number, the production node must package these transactions and broadcast it to all nodes in the system.

Since the block contains the judgment records of all nodes, the node received the broadcast block can use these records to verify whether each transaction in the block has passed the verification, or the block will be treated as an illegal block. The node that received the illegal block will reject the block, broadcast the illegal block in the consensus cluster, and initiate impeachment on the production node; otherwise, the node will add the block to the local blockchain.

3.5 Node Election Mechanism

Superior Node Election

Assume that k superior nodes need to be elected.

The election process of the superior node is as follows: first, each node randomly generates k node numbers as its own "referral target" and sends the numbers to all other nodes. After receiving the "referral targets" from all other nodes, each node summarizes and arranges them in descending order, then selects the first 2 k nodes as candidate nodes. If multiple nodes have the same number of votes, select the node that joined the network early.

Production Node Election

Each superior node randomly generates a node number as "referral target" and sends the numbers to other superior node. After receiving the "referral targets" from other superior node, each superior node summarizes and selects the node with the highest reputation value as the production node. If multiple nodes have the same reputation value, select the node that joined the network early.

3.6 Node Replacement Mechanism

According to our design, there are three types of nodes in the system, of which production nodes and superior nodes have more permissions than ordinary nodes. In order to avoid the risk of the production node or the upper node being occupied by the same node for a long time, we introduced a node replacement mechanism, which aims to regularly replace the production nodes and the superior node, reducing the degree of centralization and risks of the system

The node replacement mechanism is improved from the PBFT's view switching mechanism, focusing on strengthening the supervision of production nodes and superior nodes. Normally, node replacement occurs in the following three situations:

1) A node ends its term
2) A node was found to be dereliction of duty;
3) A node has malicious behavior.

When the production node or the upper node is replaced due to the end of the term, the ordinary node will reselect the production node or the superior node according to the normal process, and the reputation value of the old production node or the old superior node will not be affected. When the production node or the superior node is replaced due to malicious behavior or malfeasance, the ordinary node will initiate impeachment against the former. If the impeachment is successful, the former will be punished by the reduction of the reputation value, and the authority of the superior node/production node will be cancelled immediately. The ordinary node will then select a new superior node/production node.

In general, in the RC-VPBFT algorithm, ordinary nodes elect and supervise superior nodes, while superior nodes elect and supervise production nodes (as shown in Fig. 8). Through this mechanism, the ordinary node, the superior node, and the production node form a mutual check and balance to ensure the safety of the system.

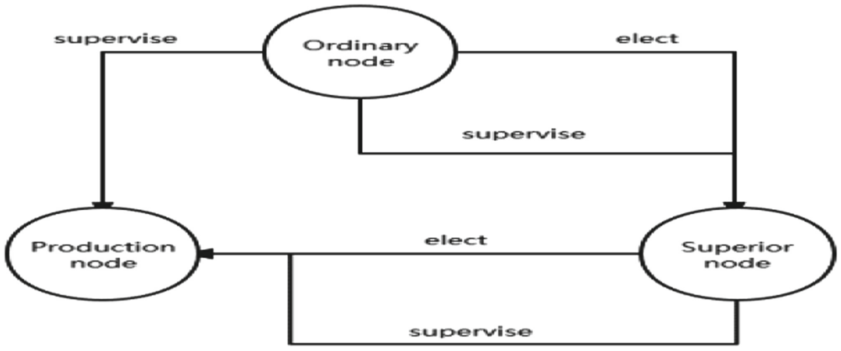


Fig. 8. Mutual supervision of nodes

Production Node Replacement

The production node is replaced when the following situation occurs:

The number of blocks packed by the production node reaches the threshold

The initiator of the replacement is the superior node. The superior node is responsible for recording the number of blocks packed by the current production node. By limiting the workload of production nodes, the system is prevented from tending to be centralized.

The production node did not package the transaction and generate new block within the specified time.

The initiator of the replacement is the sender of the transaction.

Suppose A is the sender of a transaction. If A does not receive a valid block containing this transaction within a specified time, A broadcasts a query in the cluster. If other nodes receive a valid block containing this transaction, A synchronizes this block from other nodes. Otherwise, A initiate's impeachment of the production node and attaches the transaction to the impeachment as evidence.

The production node maliciously tampers with the verification results of single or multiple transactions in the block.

The exchange initiator can be any node. Every transaction in the block should be verified by voting, and every node participating in the voting will leave their signature on the voting result. If the node receiving the block finds that there are traces of forgery or tampering in the voting result, it should Initiate impeachment of the production node, and attach the tampered block to the impeachment as evidence.

The replacement of the production node includes the following steps:First, a node initiates impeachment, requesting the replacement of the production node. Then, the other nodes decide whether to approve the impeachment based on the sufficiency of the impeachment information and send their decision with signature to the superior node. Finally, the superior node broadcasts the voting results to all notes. When the number of yes-votes is greater than two thirds of the total number of nodes, the production node will be replaced, namely, a new production node election will be performed.

Superior Node Replacement
The superior node is replaced when the following situation occurs:

a) The term of the superior node ends.
b) The superior node maliciously initiated the replacement of the production node.

When most other nodes object to the replacement, the replacement is considered illegal, and the superior node that initiated the replacement will be punished.

c) Publish fake voting results.

Due to the lack of valid signatures of other nodes in the forged voting results, this behavior will not affect the normal production node election. The upper node that issued the voting result will be impeached, and the fake voting result will be attached to the impeachment information as evidence.

The replacement of the superior node includes the following steps: First, a certain node initiates impeachment, requesting a superior node replacement; then, other nodes check the evidence in the impeachment information to decide whether to support the impeachment, and send their judgment with signature to other nodes. If more than two-thirds of the nodes support the impeachment, the impeached node loses the status of the superior node. A new superior node will be elected.

3.7 Node Penalty Mechanism

In Sect. 3.6, we listed the malfeasance and malicious behavior of the superior node/ production node and formulated the corresponding supervision and punishment mechanism. In addition to the superior node/ production node, ordinary nodes will also be punished when the following behaviors occur:

1) During the transaction confirmation process, the judgment is inconsistent with the final voting result.
2) During the election process, the judgment is inconsistent with the final voting result.

The reputation value of punished nodes will be reduced according to the rules described in Sect. 3.3, which will reduce their chances of becoming superior nodes/ production nodes in the election. Nodes with a reputation value less than 0 after being punished will be forced to move out of the network.

4 Experiment Analysis

4.1 Fault Tolerance

The fault tolerance performance of a consensus mechanism can be measured by the ratio of the number of malicious nodes allowed to normal nodes when the system is running normally. In the RC-VPBFT algorithm, the maximum number of malicious nodes that can be tolerated is:

$$\frac{m}{2} - 1, m \leq N \tag{3}$$

where N is the total number of nodes in the system, and m is the number of nodes participating in verification in each consensus, which is an adjustable variable. In practical applications, when the security requirements of the system are high, the maximum value of m can be set to N, at this time, the number of tolerable failure nodes in the system reaches the maximum value is:

$$\frac{N}{2} - 1 \tag{4}$$

In other words, when the number of malicious nodes in the system is f, the total number of nodes can be restricted as:

$$\mathrm{N} \geq 2 * \mathrm{f} + 1 \tag{5}$$

In the traditional PBFT algorithm, when the number of malicious nodes in the system is f, the total number of nodes in the system is not less than [9]:

$$\mathrm{N} \geq 3 * \mathrm{f} + 1 \tag{6}$$

Through comparison, it can be found that when the number of malicious nodes in the system is the same, the RC-VPBFT algorithm requires fewer total nodes in the system than the PBFT algorithm, that is, the RC-VPBFT algorithm has better fault tolerance performance.

4.2 Message Complexity

In the blockchain network, every broadcast message needs to consume a certain amount of network bandwidth, causing time delays. The number of messages required in a complete consensus process using RC-VPBFT algorithm can be expressed as:

$$T = 1 + N + N + N \tag{7}$$

where, N is the number of nodes currently participating in the consensus. As can be seen from the (7), with the increase of N, the number of messages required for a single consensus process in the network increases linearly.

In the PBFT algorithm, broadcast messages exist in the three stages of pre-preparation, preparation and confirmation, and the number of messages is N, N^2 and N^2, respectively. The number of messages required in a complete consensus process in the PBFT algorithm can be written as:

$$T = 1 + N + N^2 + N^2 + N \tag{8}$$

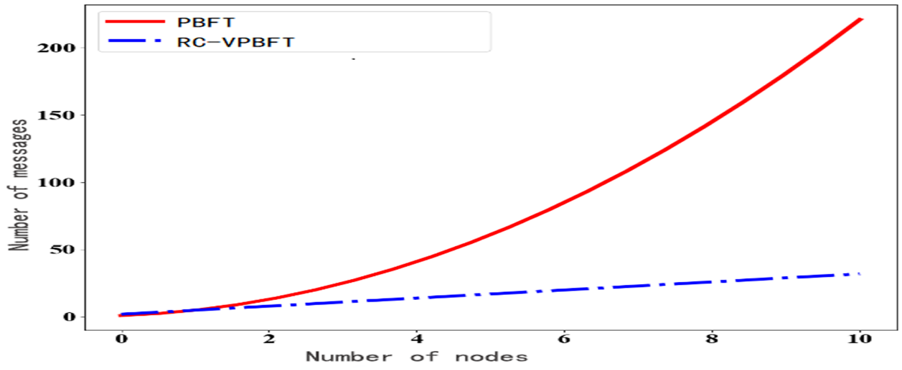
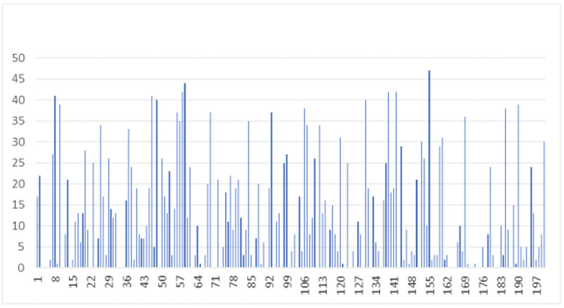


Fig. 9. Comparison of message complexity

As shown in Fig. 9, the number of messages required to reach consensus in the RC-VPBFT algorithm is much less than the PBFT algorithm.

4.3 System Centralization Trend

The core advantage of blockchain is decentralization, so it is necessary to consider the impact of consensus algorithms on the degree of system decentralization. According to the VPBFT algorithm [8] and the RC-VPBFT algorithm, we simulated 50 votes in a 200-node system. the times that each node is elected as a production node is shown in Fig. 10:



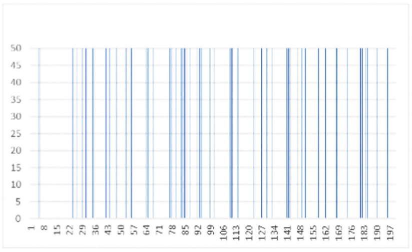


Fig. 10. Comparison of the frequency each node becomes a production node in the VPBFT and RC-VPBFT algorithms

In VPBFT, several nodes are elected as production nodes in 50 votes, and other nodes have little chance of being elected as production nodes, making the entire system tend to be a centralized system. In RC-VPBFT, each node can become a production node. Through comparison, we can find that RC-VPBFT allows more nodes to participate in the block production activities and better maintain the decentralized characteristics of the system.

5 Conclusion

In this paper, we propose a PBFT consensus mechanism based on dynamic clustering reputation value voting. By introducing random parameters in the node recommendation algorithm, the reputation value is no longer used as the only criterion for node recommendation, which avoids the problem of the accumulation of the reputation value existing in some nodes caused by the Matthew effect and the tendency of the system to be centralized due to this problem. By introducing the ISODATA algorithm and simplifying the PBFT algorithm, the complexity of the message is reduced from O(nˆ2) to O(n) with enough security, so that the system can accommodate more nodes under the same hardware and network condition, which improve the security of the system. By designing the network dynamic expansion protocol, the dynamic joining and exiting of nodes are partially realized, new nodes can join in existing network freely.

References

1. Khan, F.A., Asif, M., Ahmad, A., et al.: Blockchain technology, improvement suggestions, security challenges on smart grid and its application in healthcare for sustainable development. Sustain. Cities Soc. **55**, 102018 (2020)
2. Casino, F., Dasaklis, T., Patsakis, C.: A systematic literature review of blockchain-based applications: current status, classification and open issues. Telemat. Inf. **36**, 55–81 (2019)
3. Frizzo-Barker, J., Chow-White, P.A., Adams, P.R., et al.: Blockchain as a disruptive technology for business: a systematic review. Int.J. Inf. Manag. **51** (2020)
4. Xue, T., Yuan, Y., Ahmed, Z., et al.: Proof of contribution: a modification of proof of work to increase mining efficiency. In: 2018 IEEE 42nd Annual Computer Software and Applications Conference (COMPSAC), pp. 636–644 (2018)
5. King, S., Nadal, S.: PPcoin: peer-to-peer crypto-currency with proof-of-stake. Engineering (2012)
6. Nguyen, C.T., Hoang, D.T., Nguyen, D.N., et al.: Proof-of-stake consensus mechanisms for future blockchain networks: fundamentals, applications and opportunities. IEEE Access **7**, 85727–85745 (2019)
7. Lamport, L.: Brief announcement: leaderless byzantine paxos. In: Peleg, D. (ed.) DISC 2011. LNCS, vol. 6950, pp. 141–142. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-24100-0_10
8. Ongaro, D., Ousterhout, J.: In search of an understandable consensus algorithm. In: USENIX, pp. 305–320 (2014)
9. Castro, M., Liskov, B.: Practical Byzantine fault tolerance. In: Proceedings of the Third USENIX Symposium on Operating Systems Design and Implementation (OSDI), pp. 173–186 (1999)
10. Bugday, A., Ozsoy, A., Öztaner, S.M., et al.: Creating consensus group using online learning-based reputation in blockchain networks. Pervasive Mob. Comput. **59** (2019)
11. Vincent, G.: From blockchain consensus back to Byzantine consensus. Future Gener. Comput. Syst. **107**, 760–769 (2020)
12. Wang, Y., et al.: Study of blockchains's consensus mechanism based on credit. IEEE Access **7**, 10224–10231 (2019)
13. Wang, Y., Song, Z., Cheng, T.: Improvement research of PBFT consensus algorithm based on credit. In: Zheng, Z., Dai, H.-N., Tang, M., Chen, X. (eds.) BlockSys 2019. CCIS, vol. 1156, pp. 47–59. Springer, Singapore (2020). https://doi.org/10.1007/978-981-15-2777-7_4
14. Lv, S., Li, H., Wang, H., Wang, X.: CoT: a secure consensus of trust with delegation mechanism in blockchains. In: Si, X., Jin, H., Sun, Y., Zhu, J., Zhu, L., Song, X., Lu, Z. (eds.) CBCC 2019. CCIS, vol. 1176, pp. 104–120. Springer, Singapore (2020). https://doi.org/10.1007/978-981-15-3278-8_7
15. Veronese, G.S., Correia, M., Bessani, A.N., et al.: Efficient byzantine fault-tolerance. IEEE Trans. Comput. **62**(1), 16–30 (2013)
16. Wang, H., Guo, K., Pan, Q.: Byzantine fault tolerance consensus algorithm based on voting mechanism. J. Comput. Appl. **39**(06), 1766–1771 (2019)
17. Wang, H., Guo, K.: Byzantine fault tolerant algorithm based on vote. In: 2019 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC), Guilin, China, pp. 190–196 (2019)
18. Li, K., Li, H., Wang, H., et al.: PoV: an efficient voting-based consensus algorithm for consortium blockchains. Front. Blockchain **3**, 11 (2020)
19. Ball, G.H., Hall, D.J.A.: Clustering technique for summarizing multivariate data. Syst. Res. Behave Sci. **12**(2), 153–155 (1967)

20. Likas, A., Vlassis, M., Verbeek, J.: The global K-means clusteringalgorithm. Pattern Recogn. **36**(2), 451–461 (2003)
21. Pattabiraman, V., Parvathi, R., Nedunchezian, R., et al.: A novel spatial clustering with obstacles and facilitators constraint based on edge detection and K-medoids. In: International Conference on Computer Technology and Development, pp. 402–406 (2009)
22. Tang, C., Wu, L., Wen, G., et al.: Incentivizing honest mining in blockchain networks: a reputation approach. IEEE Trans. Circ. Syst. II: Express Brief. **67**(1), 117–121 (2020)