Hindawi
Wireless Communications and Mobile Computing
Volume 2021, Article ID 8455180, 10 pages
https://doi.org/10.1155/2021/8455180

Research Article

Improved Byzantine Fault-Tolerant Algorithm Based on Alliance Chain

Wuqi Gao,[1] **Wubin Mu**,[1] **Shanshan Huang**,[2] **Man Wang**,[2] **and Xiaoyan Li**[2]

[1]*School of Computer Science and Technology, Xi'an Technological University, Xi'an 710021, China*
[2]*School of Electronics and Information Engineering, Xi'an Technological University, Xi'an 710021, China*

Correspondence should be addressed to Wuqi Gao; gaowuqi@xatu.edu.cn

Received 25 August 2021; Revised 7 October 2021; Accepted 15 October 2021; Published 29 October 2021

Academic Editor: Deepak Kumar Jain

Alliance chain is a typical multicenter block chain and is easily implemented, so it is supported by more and more enterprises and governments. This paper analyzes the advantages and disadvantages of the Practical Byzantine Fault Tolerance (PBFT) in the alliance chain application scene. Aiming at the low efficiency of multinode consensus of the PBFT algorithm, the C-Raft-PBFT consensus algorithm is proposed. By integrating the Raft algorithm and the PBFT algorithm with the credit mechanism, designing node credit evaluation and grading protocols, and increasing Byzantine node detection based on feedback mechanism and other methods, the system efficiency is improved. The experiment results show that the improved algorithm has better throughput and lower delay, and the system's fault tolerance is also improved. Among them, the delay is reduced by 1.93 seconds on average; in the case of an increase in system nodes, the number of nodes in the experimental data is between 200 and 225, and the throughput is increased by 6.46% on average.

1. The Introduction

Block chain originates from Bitcoin. Mr. Satoshi Nakamoto puts forward the concept of Bitcoin [1] in 2009. It is designed as a peer-to-peer digital currency [2], using asymmetric encryption algorithm and proof-of-work consensus mechanism to achieve the immutability of the system together [3–5]. Block chain is the underlying technology of Bitcoin [6], which has the characteristics of immutability, traceability, anonymity, openness, and transparency. According to different application scenarios of block chain technology, such as artificial intelligence and Internet of Things [7, 8], it is generally divided into public chain, private chain, and alliance chain [9]. Among them, the alliance chain is the current main direction, which can realize strong cooperation between multiple organizations and promote the healthy and orderly development of the block chain industry.

Consensus algorithm is the core part of block chain and the key to ensure the efficient operation of block chain system [10]. The consensus algorithm can ensure that all nodes of the system can work together without central control and achieve the consistency of system data. Current block chain consensus algorithms can be roughly divided into POX (Proof-of-X) consensus algorithm and Byzantine consensus algorithm [11]. POX series algorithms mainly include Proof of Work (POW) [12], Proof of Interest (POS) [13], and Proof of Entrusted Interest (DPOS) [14]. POX algorithm is mainly applicable to public chains, which increases the cost of service request proposal in the form of tokens to reach a consensus [15, 16]. Private chain mainly adopts RAFT algorithm to reach consensus [17]. RAFT consensus algorithm mainly considers the case of node failure and does not consider the Byzantine fault tolerance problem. At present, most alliance chains adopt Byzantine consensus algorithm, and the most mainstream alliance chain consensus algorithm is PBFT consensus algorithm. PBFT is a general algorithm to solve the consistency problem of distributed systems with Byzantine error nodes. The service of the PBFT algorithm

is deterministic, that is, each consensus node performs an operation in the same state to get the same result, and each consensus node must have the same result when it starts to perform an operation state. PBFT algorithm has fewer starting nodes and higher consensus efficiency, and fault tolerance is close to 1/3 and does not require a lot of computing power. However, PBFT algorithm also has some shortcomings such as the performance of PBFT algorithm decreases sharply with the increase of the number of nodes [18–20].

At present, alliance chain is more suitable for the needs of various application scenarios. The main contribution of this paper is to propose C-RAFT-PBFT consensus algorithm to solve the existing problems of PBFT algorithm used in traditional alliance chain and integrate RAFT algorithm and PBFT algorithm with credit mechanism to design node credit evaluation and grading protocol [21]. The Byzantine node detection method based on feedback mechanism is added to improve the system efficiency, and the improved algorithm is more suitable for the alliance chain with multiple nodes participating in the consensus. Finally, through comparative experiments, it is proved that under the same circumstances, the consensus efficiency and throughput of the optimized alliance chain consensus algorithm are higher, the consensus delay is lower, and the fault tolerance of the system is also improved to a certain extent.

2. Consensus Algorithm Analysis

A consensus algorithm is a protocol through which all the parties of the block chain network come to a common consensus, and the algorithm mainly includes PBFT consensus, view replacement protocol, and checkpoint protocol.

2.1. PBFT Consensus Algorithm Analysis. PBFT (Practical Byzantine Fault Tolerance) is considered to be one of the best algorithms used to solve Byzantine problems. PBFT consensus algorithm can solve the consistency problem of Byzantine error nodes in distributed systems. For PBFT algorithm, when there are f Byzantine nodes in the system, the total node n in the whole network must be less than $3f + 1$. Only in this way can the system run normally. PBFT consensus algorithm mainly includes three parts: consistency protocol, view replacement protocol, and checkpoint protocol.

2.1.1. Conformance Protocol and Problem Analysis. The main purpose of conformance protocol is to complete the information synchronization of the whole system. PBFT consensus algorithm includes three roles: client, master node, and slave node. PBFT is a kind of algorithm based on state machine copy, copy of all the rotation in a view in the process of operation. The main node is determined by the view number and the set of the number of nodes, the main node $p = v \bmod |R|$, where v is the view number, the value of v increases from 0, $|R|$ is the number of nodes, p is the main node number, and the master node is selected by the slave nodes in the system according to their numbers. The master node and the slave node participate in the consensus agreement as the replica node and reach the consensus through three communication stages: preprepare, prepare, and commit.

(1) *Prepreparation Stage*. Client C sends a request to the master node, and the master node broadcasts the request to the slave node after receiving the request

(2) *Preparation Stage*. All the replica nodes receive the preprepare message and judge the message. If the preprepare message is approved, it enters the prepare stage and broadcasts the prepare message to other service nodes

(3) *Confirmation Stage*. After the replica node receives $2F + 1$ prepare message, it verifies and broadcasts the message and gives the response to the client

The process of PBFT consensus algorithm to reach consensus is shown in Figure 1, where "C" is the client, "0" is the master node, "1," "2," and "3" are the three slave nodes, and "1" is the Byzantine error node.

The three-stage broadcast process of PBFT algorithm consistency protocol requires large network transmission and communication overhead. In a system with N network nodes, the time complexity of PBFT algorithm is $O(N^2)$, and the number of message transmission is $2N(N-1)$ after the three-stage broadcast process. Due to the complexity of communication, when the number of network nodes increases to a certain amount, the performance of PBFT will decline sharply.

2.1.2. View Replacement Protocol and Problem Analysis. The view replacement protocol is mainly used to maintain the correctness and stability of the system. When the master node fails, the view replacement protocol will be triggered. The system sets two timeout mechanisms as the trigger conditions for this protocol: no broadcast request is received from the master node within a certain period of time, or no new block is generated within a certain period of time. As soon as one condition is met, the view replacement protocol is triggered, the view number is incremented by 1, and a new master node is replaced. The process of view replacement protocol is as follows:

(1) A copy of the node to other nodes broadcast view change request, into the view $v + 1$, according to the formula of $p = v \bmod |R|$ select new master node

(2) The new master node will broadcast the new view message to other nodes after receiving $2f$ valid view replacement messages

(3) The new master node continues to execute the request that was not completed in the previous view. Starting from the preprepare stage, the consistency agreement is implemented

In the PBFT algorithm, the selection of master nodes is to take turns as master nodes by number, which is relatively random. The selected master nodes are likely to be malicious nodes, and frequent view changes will reduce system efficiency.

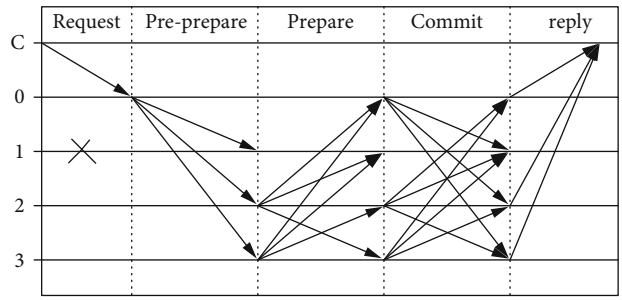

Figure 1: Schematic diagram of conformance protocol.

2.1.3. Checkpoint Protocol Analysis. Ideally, all nodes can complete various interactive tasks in the system in time to maintain consistency. However, in actual situations, some nodes may lag behind other nodes due to their own failures or network problems. At this time, periodic checkpoint protocols are needed to synchronize the entire system to prevent system failures caused by the accumulation of inconsistent nodes. After the checkpoint protocol checks the consistency status, it will clear the relevant certificates of the confirmed blocks to reduce the storage pressure of the nodes.

2.2. RAFT Consensus Algorithm and Problem Analysis. RAFT (The RAFT Consensus Algorithm) is a consistency algorithm used to manage the replicated data logs of the system. The algorithm mainly synchronizes the data of the system by logging. PBFT is a vote-based consensus algorithm. Although its throughput is better than public chain consensus algorithms such as PoW, PoS, and DPoS, PBFT still cannot meet some realistic scenarios with high throughput requirements. However, RAFT consensus algorithm can achieve significantly higher throughput than PBFT in a private chain environment with smaller node size. In addition to high throughput, the RAFT consensus algorithm can also meet the fault tolerance of 1/2. But RAFT algorithm cannot solve the Byzantine problem, therefore, applying the RAFT consensus algorithm to the alliance chain needs to solve the Byzantine node problem.

3. Improved Alliance Chain Consensus Algorithm

This article focuses on the problem that the performance of PBFT will drop sharply when the number of PBFT algorithm network nodes increases to a certain amount, the selected master node is likely to be a malicious node, frequent view replacement will reduce system efficiency, and raft algorithm is more efficient but cannot solve the Byzantine problem, this paper proposed an improved Byzantine fault-tolerant algorithm suitable for the alliance chain. The algorithm integrates the Raft algorithm and the PBFT algorithm with the credit mechanism, design node credit assessment, and grading protocol and uses the Byzantine node detection method based on the feedback mechanism. The new algorithm improves the throughput of the alliance chain system, reduces the delay, and also improves the overall fault tolerance of the system, so that it can meet the needs of richer and more demanding realistic scenarios. Figure 2 depicts the overall flow of the algorithm.

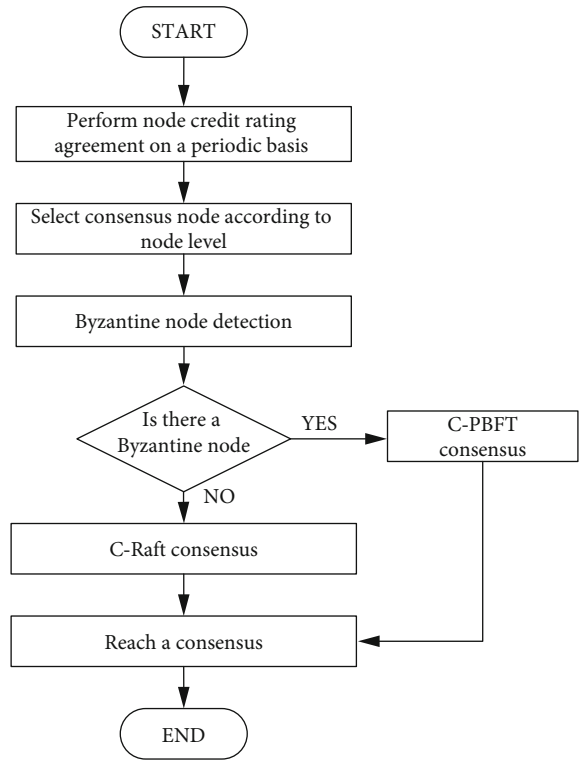

Figure 2: Overall flow chart of system improvement algorithm.

First, the system executes the node credit rating protocol periodically to grade the system nodes. Next, the nodes participating in the consensus are selected according to the node level. Then, Byzantine nodes are detected. If Byzantine nodes exist, the system adopts the improved credit-based PBFT consensus algorithm (C-PBFT). If Byzantine nodes are not detected, the system adopts the improved credit-based RAFT consensus algorithm (C-RAFT).

3.1. Credit-Based Node Evaluation Index. The credit of nodes refers to the comprehensive evaluation of the performance of nodes in the alliance chain network. In order to solve the problem of possible Byzantine nodes in the system, it is necessary to evaluate and test the nodes in the alliance chain network.

The performance of the node itself is an important factor to measure whether the node is safe or not. In addition, the trading ability of nodes and the similarity of node evaluation feedback are other main reference factors considered in this paper. Compared with active nodes with higher transaction success rate, nodes with fewer successful transactions are more vulnerable to attack and become Byzantine nodes due to lack of defense mechanism. In addition, the feedback mechanism of nodes is added in this paper, and the feedback similarity of nodes is also an important reference factor for node credit evaluation.

According to the performance of Byzantine nodes in the block chain system, a feedback-based node credit evaluation index is proposed, as shown in Table 1. According to this index, nodes in the network generate their own node state record table.

Table 1: Node credit evaluation indicators.

Feature	Specific indicators	The weight
Node performance	Network latency	0.1
	Node offline times	0.1
	Offline duration of node	0.1
Node trading capacity	Successfully participate in a valid transaction in the system network	0.2
	The total number of transactions made in the previous unit of time	0.3
Node feedback	Have you ever submitted an invalid transaction	0.1
	Whether or not it was on the last trusted list	0.1

3.2. Node Hierarchy Protocol. This paper classifies the nodes in the network based on credit evaluation indexes and selects active nodes to participate in the system consensus. Definition $C[I]$ represents the credit score of node A, which can be classified into four levels: A, B, C, and D. At the beginning of the system operation, the initial credit value of all nodes in the network is $C[S]$, and the initial integral is defined as 0. With the continuous operation of the system, the credit score of nodes is increased or decreased according to the credit evaluation indexes of nodes.

According to the node evaluation index in the previous section, in terms of node performance, the credit score of nodes with a network delay is reduced by 1, the credit score of nodes with an offline time is reduced by 1, and the credit score of nodes with an offline time greater than the threshold is reduced by 1. In terms of node trading capacity, the credit score of a node successfully participating in a valid transaction in the system network is increased by 2, and the credit score of an invalid transaction submitted by the node is reduced by 1. If the total number of transactions generated within a unit time on the node is equivalent to the total number of transactions generated within a unit time on the system, the credit score of the node is increased by 3. Finally, the node is added to the credit score of the node in the past trusted round. A node's credit score is greater than or equal to 6 is defined as $C[G]$, and a node's credit score less than or equal to -4 is defined as $C[B]$ according to the following formula.

$$C[i] = \begin{cases} C[g] & C[i] \geq 6, \\ C[s] & C[i] = 0, \\ C[b] & C[i] \leq -4. \end{cases} \quad (1)$$

According to the above formula, the credit rating of nodes is carried out, and when the credit score of nodes is $C[I] > C[G]$, node credit rating A; when the credit score of nodes is $C[S] < C[I] < C[G]$, node credit rating B; when the credit score of nodes is $C[b] < C[i] < C[s]$, node credit rating C; when the credit score of nodes is $C[i] < C[b]$, node credit rating D.

System nodes are divided into four categories according to the node credit rating protocol, and nodes with different credit ratings have changed permissions. Among them, class A nodes have the highest credit rating and are preferred to participate in the system consensus; second, B-type nodes. When A-type nodes are selected or not A-type nodes exist, consensus nodes will be selected from B-type nodes. Class C nodes have low credit, but when there are not enough nodes of class A and B, they can still participate in the consensus of the system. D class node has a low credit rating and does not take part in the system consensus.

Table 2: Node permission classification.

Credit rating	Preferred participation consensus	Participate in the consensus	Not participating in the consensus
A	√	√	√
B	×	√	√
C	×	√	√
D	×	×	√

The credit-based node grading protocol can greatly improve the enthusiasm of nodes and effectively reduce the probability of malicious nodes participating in the consensus, thus, improving the system efficiency. Table 2 shows the classification of node permission.

3.3. Byzantine Node Detection Based on the Feedback Mechanism. Nodes that are subject to system failures, network delays, or malicious attacks during a consensus round are called Byzantine nodes. Increase the feedback mechanism between the nodes in this paper, namely, the consensus after the end of each round, each assessment system between nodes, the main evaluation factors for the node a consensus on the success of participation, which is in the last round of the trusted list, after the round of consensus, the system will generate a node feedback table consistency $\mathrm{FT} = \{f1, f2, \cdots \mathrm{Fi} \cdots, \mathrm{fn}\}$, where fi is the feedback information of node I, and FT will be constantly updated as the consensus process progresses. If the feedback opinions of all nodes are consistent, it is considered that there is no Byzantine node in the system.

In addition, after the consensus is ended, the node transaction information table is generated by the client with $\mathrm{RE} = \{r1, r2 \cdots \mathrm{ri} \cdots, \mathrm{rn}\}$, where ri is the transaction information sent by node i to the client, if the client does not receive trading information from all the nodes, the system will also think that the node has produced a wrong behavior, the node that has not sent information is determined to be a

Input: Node feedback consistency table FT, client received information table RE
Output: True/false (true: Byzantine node exists; false: No byzantine node)
//determine the consistency of feedback messages
1 **for** i = 0: N-1
3 **if** (!f_i == f_{i+1})
4 **return** true;
5 **end if**
6 **end for**
//determines whether the client has received node information for all nodes
8**if** (RE.Length!=N)
9 **return** true;
10 **end if**

Algorithm 1: Byzantine node detection algorithm.

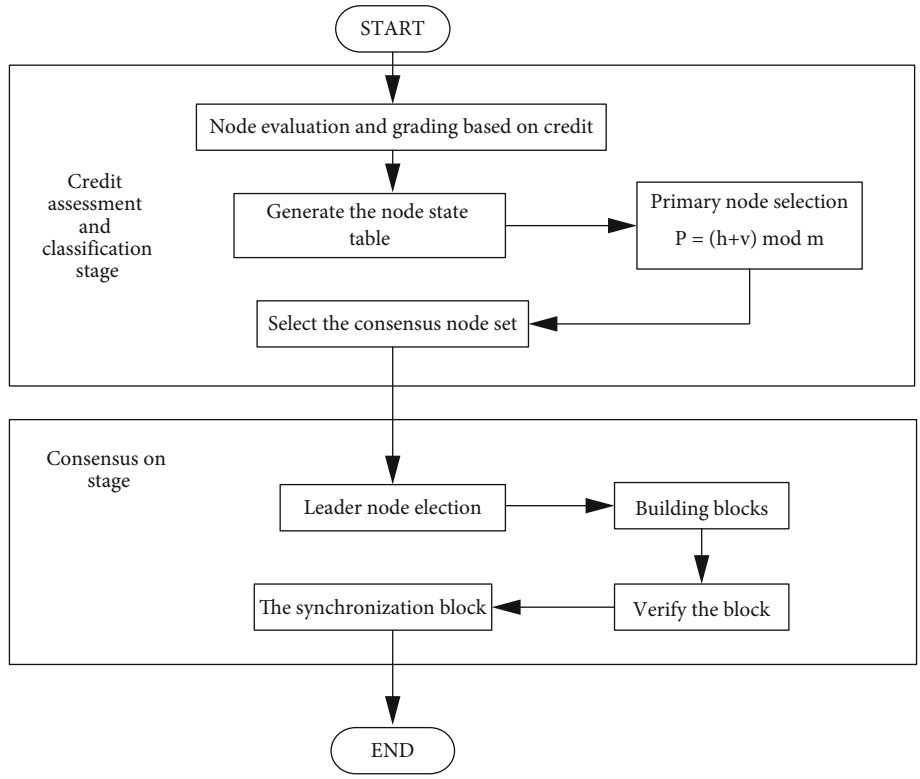

Figure 3: C-RAFT consensus process.

Byzantine node, and the node is removed from the consensus node set, thereby reducing the possibility of Byzantine nodes in the system consensus node and improving system efficiency.

For a consensus set $M\{1, 2, 3 \cdots, N\}$, Byzantine node detection algorithm is shown in Algorithm 1. The input of Algorithm 1 is the node feedback consistency table FT = $\{F1, F2 \cdots \text{Fi} \cdots, \text{fn}\}$ and the information table received by the client RE = $\{r1, r2 \cdots \text{Ri} \cdots, \text{rn})\}$. The output of Algorithm 1 is whether there are Byzantine nodes in the system. If the output is true, it means there are Byzantine nodes in the system; otherwise, it is false, and the system has no Byzantine nodes.

3.4. Improved Algorithm Consensus Process

3.4.1. C-RAFT Consensus Process. C-RAFT consensus algorithm is divided into credit assessment stage and consensus stage. The consensus process is shown in Figure 3, as follows:

(1) First, each node is numbered by the system. Assuming the total number of nodes in the system is m, the number is 1 to m. The nodes in the network are scored and graded according to the credit evaluation index and grading protocol. Then, the serial number of the main node in the consensus process of this round is selected according to the following formula

$$p = (h + v) \bmod m. \quad (2)$$

In Formula (2), M represents the number of nodes, H represents the current block height, and V represents the number of rescreening rounds. The system is initiated by the master node for credit rating. Each node in the network sends its own status records to the master node. The master node takes these information as the input of credit rating and executes node rating protocol and outputs the node rating results.

(2) The second step is the consensus process. The C-RAFT consensus process includes leader node election, leader node election, block building, validation, and synchronization

(1) Leader node election

The nodes that the system participates in the consensus are all honest nodes. During the whole process of reaching the consensus, these nodes are mainly in one of the following three states: leader, candidate, and follower. The interactive transformation relationship of these three situations is shown in Figure 4.

At the beginning of each round of consensus, all the consensus nodes are followers. The follower node is passive, unable to initiate requests and package blocks, and can only respond to the mentor node and candidate nodes. The leader node has unique authority to collect transactions and building blocks, and only one leader node can exist at a time. Candidate is the intermediate state in which followers become leaders. Considering the physical location of nodes in the network, the network environment and other factors caused by the clock are out of sync. Term is used as the logical clock. Terms are numbered with increasing integers, and each node stores the current term number. The current term is exchanged through internode communication and updated to a larger term number. When the candidate or leader node finds that its term number has expired, it will immediately revert to follower.

Using remote procedure calls (RPCs) to communicate between nodes of the system requires only two different types of node RPCs: request vote RPC and append entries RPC. In each consensus round, the lead node uses a heartbeat mechanism to perform RPC communication at 50-millisecond intervals. As Figure 4 shows, each entry starts with a leader node election. Initially, all the consensus nodes are in the follower state, and each node increases its own term number and waits a random 150-300 milliseconds. The node then votes for itself and issues the request vote RPC request in parallel with the other consensus nodes, after which the consensus node is transitional to the candidate state. The candidate node changes its state based on the number of votes received. During the tenure, there will eventually be only one leader node. If no node gets enough votes in one period, it moves on to the next period. Once a node is selected as the leader, that node immediately broadcasts an empty append entries RPC message to the other nodes to prevent the election of a new term.

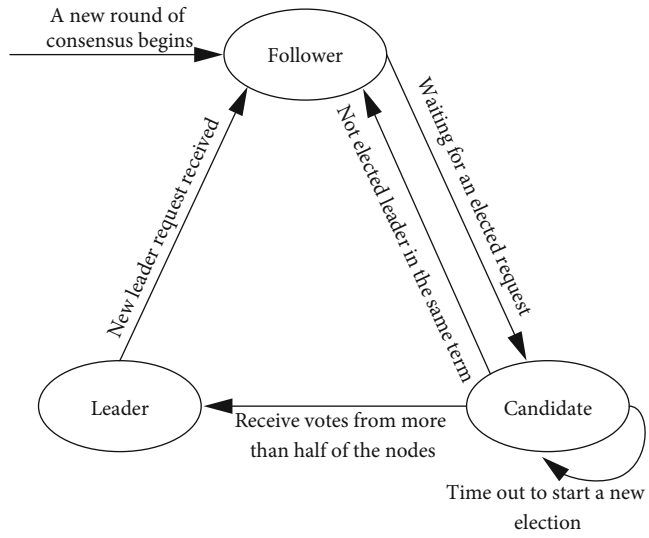

Figure 4: Schematic diagram of three state transitions

(2) Building blocks

The elected leader node will complete as many transactions as possible and authenticate each transaction. The leader then broadcasts an append entries RPC message containing block information and the current term number to the remaining follower nodes.

(3) Verify blocks

Each follower that receives an append entries RPC message verifies the correctness of the block transaction. If the validation is approved, vote to the leader node for confirmation. The leader node will count the number of votes within a certain period of time. When the number of votes exceeds 1/2 of the system consensus node, the system sends an empty append entries RPC message to all follower nodes to indicate that the block has been approved by most nodes. The follower node receives the confirmed append entries RPC and records the block information to the local block chain.

(4) Synchronous blocks

Consensus node initiates synchronous block request to the whole network node, so that the whole network block chain is updated.

3.4.2. C-PBFT Consensus Process. C-PBFT algorithm contains the master node election consensus and consistency, the election of the master node is no longer rotated by the original PBFT algorithm, but in credit rating is chosen as main reference, the node with the highest credit is selected as the master node, so the probability that the master node is the Byzantine nodes can be effectively reduced. At the same time, the system overhead can be reduced, and consensus efficiency can be improved. The consensus process is shown in Figure 5.

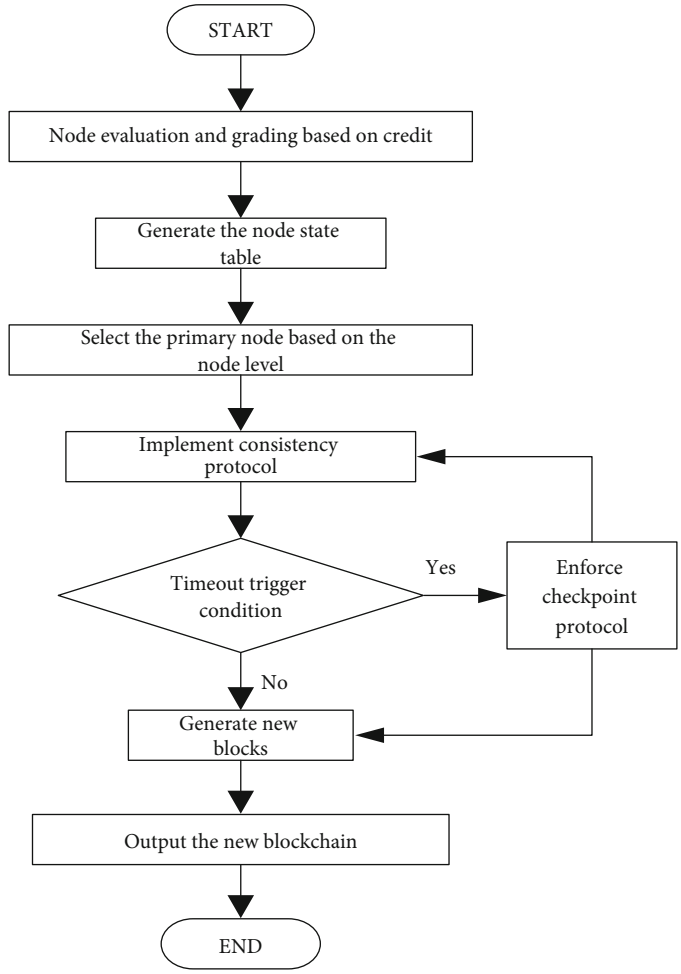

Figure 5: C-PBFT consensus process.

The specific process of C-PBFT algorithm is as follows:

(1) First, the credit value of the nodes is calculated and graded according to the credit evaluation and node grading protocol, and then the node with the highest credit value is selected as the main node among the nodes of class A level

(2) Next, the master node participates in the conformance protocol and determines whether to trigger the timeout trigger condition set in the view replacement protocol

In case of timeout, the checkpoint protocol is implemented and the master node is replaced to enter a new consistency protocol. Otherwise, a new block is generated and a new block chain is output.

3.5. Experimental Analysis. Docker technology was used in the experiment to simulate and build a multinode block chain environment. The specific experimental environment is shown in Table 3. In this paper, the performance of the improved algorithm is verified by building a prototype of the alliance chain, and the experimental results are analyzed.

Table 3: Experimental operation environment.

Experimental configuration	Configuration parameters
CPU	Intel(R)Xeon(R) W-2123 CPU@ 3.60 GHz
Memory	32.0 GB
The operating system	Ubuntu16.04
The Docker version	18.03.0-ce
Network bandwidth	200 Mbps

3.5.1. Ductility Analysis. Delay refers to the time required for system consensus algorithm to reach consensus and complete data synchronization. The system's lower latency allows transactions to be confirmed more quickly. Run the improved consensus algorithm designed in this paper and then continuously add the number of nodes participating in the consensus to check the system delay under different number of nodes. After testing the delay of the improved algorithm in this paper, the delay of the traditional PBFT consensus algorithm of the alliance chain is tested in the same network environment on the premise that the number

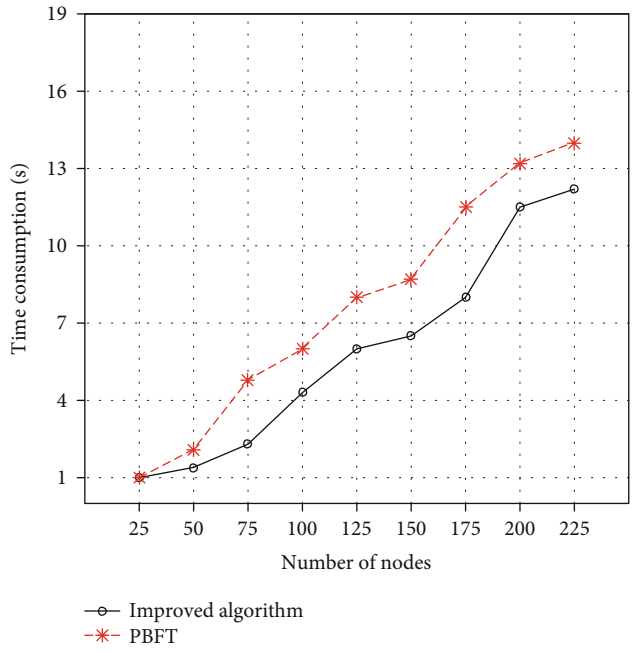

Figure 6: Consensus delay comparison diagram.

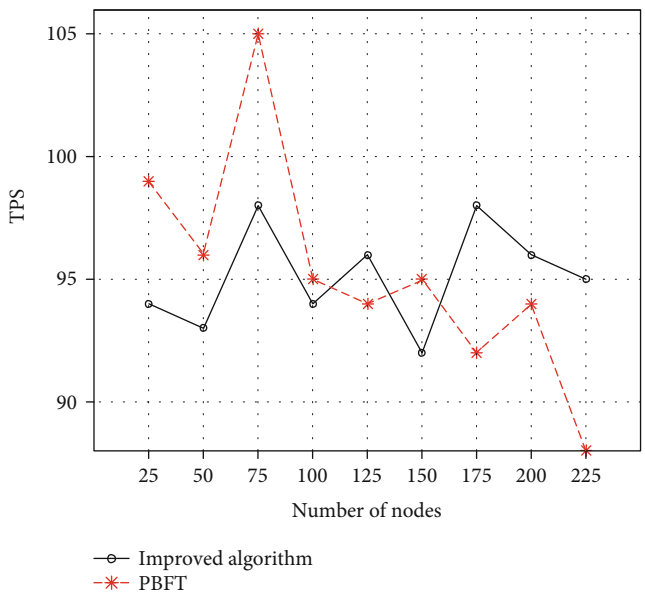

Figure 7: Throughput comparison diagram.

of nodes is the same and the size of the transmitted transaction information data is the same. The number of nodes selected is shown on the horizontal axis in Figure 6. After 30 experiments, the average value is taken as the result, and the ductility test result is shown in Figure 6.

As can be seen from Figure 6, compared with the PBFT mechanism of the traditional consensus algorithm of the alliance chain, the improved consensus algorithm takes less time to complete a certain number of data synchronization processes under the same number of nodes, and the delay

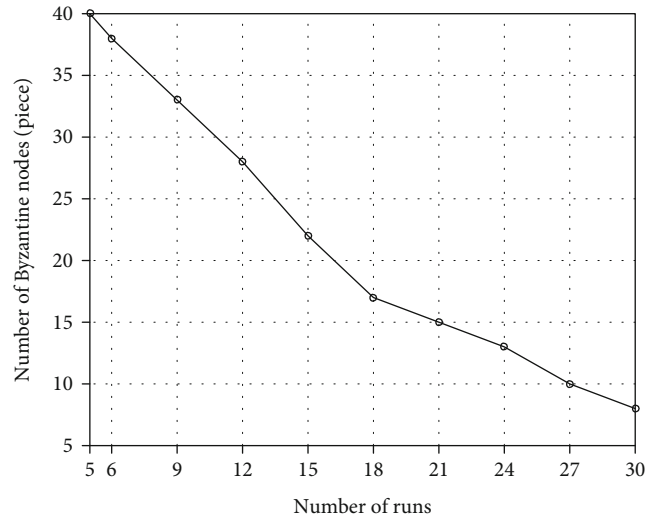

Figure 8: Byzantine node number of the improved algorithm.

of the improved algorithm is calculated by the experimental results to be 1.93 seconds lower on average than that of the PBFT algorithm. This is because when broadcasting new block information, the system is based on the credit model to reduce the probability of Byzantine error in the consensus node. The whole system is in a benign cycle, and RAFT consensus algorithm is more used. The two-stage submission is more efficient than PBFT three-stage full-network broadcast, so it is better than the traditional PBFT consensus algorithm in time delay performance.

3.5.2. Throughput Analysis. Throughput is expressed in TPS, which represents the number of requests a system can handle per second. This index can evaluate the concurrency capability of the system. In the throughput analysis, it mainly tests the maximum number of block data synchronization completed by the two consensus algorithms within the same time when the number of nodes gradually increases. After 30 experiments, the average value is taken as the result, as shown in Figure 7.

Figure 7 shows that chain PBFT consensus algorithm for traditional alliance; as the number of nodes increases to a certain number, the system throughput will drop significantly, improve after the algorithm in the node number is less when throughput is less than traditional league chain consensus algorithm, but with the operation of the system, node number more, in the node number about 200, the throughput of PBFT algorithm shows a significant downward trend, but the improved algorithm still has good throughput, which is higher than the traditional PBFT consensus algorithm. According to the calculation of experimental data, the throughput of the improved algorithm is 6.46% higher than that of the traditional PBFT when the number of nodes is between 200 and 225.

3.5.3. Fault Tolerance Analysis. The PBFT consensus algorithm requires that the maximum number of Byzantine nodes that the system can tolerate is not more than 1/3 of the total network nodes, which has a low fault-tolerant ability and has a great impact on the system performance. The fault-tolerant performance of the improved algorithm is

greater than 33%. As shown in Figure 8, in a block chain system with 200 nodes, there are 53 Byzantine nodes. When the consensus algorithm that improved 30 times is run, the number of Byzantine nodes in the system is reduced to 8. It can be seen that with the operation of the system, the whole system will enter a virtuous cycle, the system will be more secure and reliable, the probability of Byzantine nodes will be reduced, the system will adopt RAFT consensus algorithm more, and the RAFT consensus algorithm requires that the maximum number of Byzantine nodes that the system can tolerate should not exceed 1/2 of the total network nodes. Accordingly, the fault-tolerant sex of whole system also had certain rise.

4. Conclusion

There are strong application scenarios for alliance chain in the future, but there is no consensus mechanism specifically for alliance chain at present. In this paper, an improved consensus algorithm is proposed based on the characteristics of alliance chain of summarizing and comparing the shortcomings of current mainstream consensus algorithm. The new consensus algorithm integrates RAFT algorithm and PBFT algorithm with credit mechanism, designs node credit evaluation and grading protocol, and adds Byzantine node detection method based on feedback mechanism, which improves system efficiency and ensures that the system can detect errors in time. Consensus algorithm, as the core and key component of block chain application, has become the bottleneck of the development of block chain technology. Due to the variety of application scenarios, each consensus algorithm cannot cover all aspects. The consensus proposed in this paper is only applicable to the situation of high trust, and how to combine the block chain with the real economy in the future still deserves everyone's attention.

Data Availability

The processed data required to reproduce these findings cannot be shared at this time as the data also forms part of an ongoing study.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by National Natural Science Foundation of China (52072293, 62171360), Teaching Research Foundation of Shaanxi Province (Shan Teaching[2020]198), Teaching Research Foundation of Xi'an Technological University (0203-302011908).

References

[1] S. W. Stornetta, "Bitcoin: a peer-to-peer electronic cash system," *Journal of Cryptology*, vol. 32, 2009.

[2] J. A. Donet, C. Pérez-Solà, and J. Herrera-Joancomartí, "The bitcoin P2P network," *International Conference on Financial Cryptography and Data Security*, vol. 8438, no. 2, pp. 87–102, 2014.

[3] W. Stallings, "Cryptography and network security: principles and practice," *Int'l Annals of Criminology*, vol. 46, no. 4, pp. 121–136, 1999.

[4] J. Mattila, *The Blockchain Phenomenon-The Disruptive Potential of Distributed Consensus ArchitecturesEB/OL*, ETLA Working Papers, 2016, http://hdl.handle.net/10419/201253.

[5] P. Wang, F. Huitong, X. Li, J. Guo, Z. Lv, and R. Di, "Multi-feature fusion tracking algorithm based on generative compression network," *Future Generation Computer Systems*, vol. 124, pp. 206–214, 2021.

[6] S. H. E. N. Xin, P. E. I. Qing-Qi, and L. I. U. Xue-Feng, "Survey of block chain," *Chinese Journal of Network and Information Security*, vol. 2, no. 11, pp. 11–20, 2016.

[7] S. Yang, B. Deng, J. Wang et al., "Scalable digital neuromorphic architecture for large-scale biophysically meaningful neural network with multi-compartment neurons," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 31, no. 1, pp. 148–162.

[8] S. Yang, J. Wang, B. Deng et al., "Real-time neuromorphic system for large-scale conductance-based spiking neural networks," *IEEE Trans Cybern*, vol. 49, no. 7, pp. 2490–2503, 2019.

[9] V. Morabito, *Business Innovation through Blockchain*, Springer International Publishing, Cham, 2017.

[10] V. Gramoli, "From blockchain consensus back to byzantine consensus," *Future Gener Comput Syst*, vol. 107, pp. 760–769, 2020.

[11] S. Pahlajani, A. Kshirsagar, and V. Pachghare, "Survey on private blockchain consensus algorithms," in *In: 2019 1st international conference on innovations in information and communication technology (ICIICT)*, pp. 1–6, Chennai, India, 2019.

[12] M. Jakobsson and A. Juels, "Proofs of work and bread pudding protocols," in *Secure Information Networks*, pp. 258–272, Springer, Boston, 1999.

[13] S. King and S. Nadal, *Ppcoin: Peer-to-Peer Crypto-Currency with Proof-of-Stake*, Self-published paper, 2012.

[14] D. Larimer, *Delegated Proof-of-Stake Consensus*, 2017, http://bit-shares.org. https://bitshares.org/technology/delegating-proof-of-stake-consensus.

[15] L. S. Sankar, M. Sindhu, and M. Sethumadhavan, "Survey of consensus protocols on blockchain applications," in *In: 2017 4th international conference on advanced computing and communication systems (ICACCS)*, pp. 1–5, Coimbatore, India, 2017.

[16] D. Mingxiao, M. Xiaofeng, Z. Zhe, W. Xiangwei, and C. Qijun, "A review on consensus algorithm of blockchain," in *In: 2017 IEEE international conference on systems, man, and cybernetics(SMC)*, pp. 2567–2572, IEEE, 2017.

[17] D. Huang, X. Ma, and S. Zhang, "Performance analysis of the raft consensus algorithm for private blockchains," *IEEE Trans Sys Man Cybern Sys*, vol. 50, no. 1, pp. 172–181, 2019.

[18] H. Sukhwani, J. M. Martinez, X. Chang, K. S. Trivedi, and A. Rindos, "Performance modeling of PBFT consensus process for permissioned blockchain network (hyperledger fabric)," in *In: 2017 IEEE 36th Symposium on Reliable Distributed Systems (SRDS)*, pp. 253–255, IEEE, 2017.

[19] P. Wang, M. Sun, H. Wang, X. Li, and Y. Yang, "Convolution operators for visual tracking based on spatial-temporal regularization," *Netural Computing and Applications.*, vol. 32, no. 10, pp. 5339–5351, 2020.

[20] L. Zhang and Q. Li, "Research on consensus efficiency based on practical byzantine fault tolerance," in *In: 2018 10th international conference on modelling, identification and control (ICMIC)*, pp. 1–6, IEEE, 2018.

[21] C. Wang, "Dynamic improved PBFT algorithm based on credit coefficient," *Electronic Production*, vol. 8, 2019.