Research Article

Improve PBFT Based on Hash Ring

Wang Zhong , Xiandong Zheng , Wenlong Feng , Mengxing Huang, and Siling Feng

School of Information and Communication Engineering, Hainan University, Haikou 570228, China

Correspondence should be addressed to Wenlong Feng; fwlwl@163.com

Received 10 August 2021; Revised 30 September 2021; Accepted 26 October 2021; Published 9 November 2021

Academic Editor: Mariusz Glabowski

Copyright © 2021 Wang Zhong et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Aiming at the problems of practical Byzantine fault tolerance (PBFT) algorithm, such as high communication complexity, frequent switching views because of Byzantine node become primary nodes and random selection of primary node, HR-PBFT algorithm is proposed. First, the HR-PBFT algorithm uses a hash ring to group nodes, which ensures the randomness and fairness of the grouping. Then, a dual-view mechanism is used in the consensus process, where the first layer node maintains the primary view and the second layer node maintains the secondary view to ensure the proper operation of the algorithm. Finally, the Byzantine node determination mechanism is introduced to evaluate the node status according to the node behavior in the consensus process, improve the reliability of primary node selection, and reduce the frequency of view changes. The experimental results show that the optimized HR-PBFT algorithm can effectively improve the problem of the sharp increase in the number of communications caused by the increase in the number of nodes in the network and prevent frequent view changes.

1. Introduction

Since the birth of Bitcoin [1], digital cryptocurrency has developed rapidly, and its underlying blockchain technology has been widely studied by scholars. Blockchain technology uses multiple technologies such as peer-to-peer communication, consensus algorithms, and cryptography to solve the data monopoly and security problems of existing centralized mechanisms. Blockchain technology is considered to be an emerging technology with the potential to subvert traditional industries [2]; it has broad application prospects in the fields of finance, Internet of Things [3], medical care, and so on.

The core of blockchain technology is the consensus algorithm, which aims to make all nodes maintain the same ledger. The consortium blockchain is a semidecentralized blockchain, and the PBFT algorithm proposed by Castro and Liskov in 1999 [4] is widely used in the consortium blockchain. However, with the rapid development of the consortium blockchain and the number of network nodes growing rapidly, the PBFT algorithm is severely constrained in terms of throughput and scalability. Many scholars have deeply studied and improved the PBFT algorithm to make

it more adaptable to the development of today's consortium blockchain. The communication overhead of the PBFT algorithm is mainly generated by the communication between nodes in the three-phase consensus process. At present, the main improvement ideas are as follows.

In method 1, a grouping approach is used, specifically based on node attributes (node hardware performance, network communication capability, geographic location, etc.) divided into several groups by clustering algorithms, with intragroup consensus first and then intergroup consensus. In [5], an improved PBFT consensus mechanism based on K-medoids is proposed to perform intragroup consensus and then intergroup consensus by clustering and hierarchical division of the characteristics of large-scale network nodes. In [6], nodes are grouped based on their geographical locations, and in [7], a network self-clustering-based approach is proposed to group nodes based on their communication capabilities.

In method 2, digital signature technique is used to aggregate the node voting results to one node, and then the node forwards the statistical results to each node. The SBFT proposed in [8] sets collectors in the blockchain network to col-

lect and forward the votes of each node during the three-phase consensus, so as to avoid direct communication between nodes.

In method 3, a reputation model is used to evaluate the trust of nodes based on their behavior, and nodes with high trust of nodes are selected as master nodes or nodes with high trust value of nodes are selected to form a consensus group instead of all nodes to participate in consensus. In [9], consensus group is formed by electing nodes with high trust degree. In [10], it proposes an optimized PBFT based on feature trust model to improve Byzantine fault tolerance. In [11], it reduces the number of communications by establishing a reputation model, assigning different discourse rights to nodes according to their reputation values and adding a presubmission stage to the PBFT algorithm to improve the algorithm performance and can effectively defend against witch attacks.

In method 4, using a simplified PBFT three-stage protocol, the paper [12] simplifies the three-stage consensus of PBFT to two-stage consensus, which reduces the communication overhead and improves the consensus efficiency.

Although these improved algorithms optimize the traditional PBFT algorithm to some extent, there are problems: grouping according to the node attributes, human intervention is large, and the grouping can be controlled by adjusting node attributes. Although direct communication in the nodes can be avoided, the aggregated signature is too large when the number of network nodes increases, which requires high network performance. The credibility model has many iterations of credibility assessment, and the nodes have to be evaluated for trust at the end of each consensus, which increases the node burden and affects the overall consensus efficiency; reducing the consensus phase will affect the security and consistency of the algorithm.

This paper proposes an improved PBFT algorithm, namely, HC-PBFT algorithm, which adopts the grouping method based on hash ring to avoid human intervention in the grouping process and ensure the randomness and fairness of the grouping. Using hierarchical grouping strategy for consensus, the first layer uses the traditional PBFT algorithm for consensus and the second layer uses the simplified algorithm for consensus, which can greatly reduce the traffic in the consensus process. After the consensus, Byzantine node determination mechanism is executed to mark Byzantine nodes and restrict them from being elected as primary nodes to improve the reliability of master node selection.

The rest of this paper is arranged as follows. The second section introduces the improved design of the HR-PBFT algorithm. The third section presents the experimental comparison and analysis with the traditional PBFT algorithm, and the fourth section is the conclusion of this paper.

2. Design of HR-PBFT Consensus Algorithm

2.1. Overview of PBFT. In 1999, Castro and Liskov first proposed the practical Byzantine fault tolerance (PBFT) algorithm, which reduced the complexity of the algorithm from exponential to polynomial, making the Byzantine fault-tolerant algorithm feasible in practical applications. In the

PBFT algorithm, the Byzantine node that does not exceed one-third of the total number of nodes in the system can be tolerated at most; that is, if more than two-thirds of the nodes are normal, the entire system can work normally. The PBFT algorithm is mainly composed of a three-phase protocol, a view-change protocol, and a garbage collection protocol.

The three-phase protocol in the PBFT algorithm is shown in Figure 1. In a distributed environment, consensus is reached in the network by running this protocol. However, as the network scale expands, the complexity of communication is increasing rapidly in the prepare phase and the confirmation phase, making it difficult to realize large-scale applications.

The PBFT view-change protocol ensures the liveness of the algorithm. In a view, a replica node will be elected as the primary node for consensus. When the primary node makes a mistake or becomes a malicious node, it needs to switch the view. Use the rotation method to select the next backup node as the primary node and enter the next view.

The PBFT garbage collection protocol is to clean up expired client request information.

2.2. Overview of the HR-PBFT Algorithm. The HC-PBFT algorithm consists of four parts: hash ring grouping strategy, consensus process, dual-view mechanism, and Byzantine node determination mechanism.

The hash ring grouping strategy uses a hash ring to group and slice the nodes to ensure the randomness and fairness of the grouping. In the consensus process, the grouping verifies and consensuses the requests from the client, reducing the communication complexity in the consensus process. The dual-view mechanism is that the first layer nodes maintain the main view during the consensus process and the second layer nodes maintain the secondary view to ensure the orderly and normal operation of the consensus. The Byzantine node determination mechanism is to evaluate the behavior of nodes in the consensus process after the consensus is finished, to mark Byzantine nodes, to restrict them from being elected as master nodes, and to improve the reliability of primary node selection. The entire algorithm flow is shown in Figure 2.

Firstly, the unique identifier of each node is calculated through a hash algorithm, arranged to form a hash ring, and then the nodes are divided into groups according to the hash ring. Secondly, according to the primary node election rules, each group elects the primary node of the group to form the first consensus layer and the first layer of consensus layer then selects the first layer of leader nodes, waits for the client request, and performs consensus. Finally, the Byzantine node determination mechanism is executed after the consensus is finished to evaluate the node behavior during the consensus process and update the node status. By marking Byzantine nodes and restricting Byzantine nodes to be elected as primary nodes at the next view cycle, the success rate of consensus can be improved.

2.3. Hash Ring Grouping Strategy. HR-PBFT algorithm proposes a hash grouping strategy based on the principle of

FIGURE 1: The process of PBFT.

FIGURE 2: The process of HR-PBFT.

consistent hash algorithm. Firstly, by hashing the unique identifier of the consensus node, each node obtains a unique hash value. Secondly, all nodes are arranged and combined into a hash ring according to the hash value. Finally, the nodes are grouped according to the hash ring. In the blockchain consensus network, the nodes have unique identifiers and the results of the hash function calculation are unique and random, which can prevent human intervention in the grouping results. The specific node grouping algorithm is as follows,

- (1) Algorithm initialization
- (2) Calculate the hash value of each node, hash (node IP+node public key)
- (3) Sort the calculated node hash values and connect them end to end to form a hash ring
- (4) According to the number of consensus nodes n and the number of groups x , each group has $y = n/x$ nodes. If there is a remainder, just put it in the last group
- (5) The grouping is complete

As shown in Figure 3, assuming that there are 16 nodes in the blockchain network, the n is 16, and take x as 4; that is, divide the blockchain network into 4 groups. After grouping using the grouping algorithm, the result of grouping completion is shown in the left figure in Figure 3, which is divided into 4 groups A, B, C, and D, and there are four nodes in each group. In the first round of view, the initial state of each node is candidate node. The first clockwise node of each group is set as the primary node of this group, and the primary nodes of each group form the first consensus layer. The node network structure diagram in the consensus process can be graphically represented as shown in the right figure in Figure 3, where A0, B0, C0, and D0 form the first layer consensus group and their respective groups form the second layer consensus group.

There are 2 reasons for the consensus failure after grouping. First, the Byzantine nodes are not marked out at the beginning of consensus and more than $1/3$ of the nodes in the first layer are Byzantine nodes in one view cycle, which will definitely cause consensus timeout and lead to consensus failure. Second, the Byzantine nodes are grouped in groups with more than $x/3$ master nodes and the number of Byzantine nodes in each group is greater than $(y+1)/2$. In this case, the Byzantine nodes are in the majority in their own groups and can control the final voting results in the group, so as to control more than $1/3$ nodes in the first layer of master nodes, leading to consensus failure.

2.4. The Consensus Process of HR-PBFT Algorithm. The consensus process of HR-PBFT algorithm is carried out in two parts: the first layer consensus process uses PBFT three-phase consensus algorithm and the second layer consensus uses a simplified consensus algorithm. The consensus process is shown in Figure 4, and the specific consensus process steps are as follows:

- (1) Pre-preparation phase: after receiving the request from the client, the primary node of the first layer verifies and sorts the client request. Then, it broadcasts a pre-preparation message to the nodes of the first layer and each node of the first layer receives the pre-preparation message and forwards it to its own group of nodes of the second layer, at which time all nodes in the network receive the pre-preparation message

FIGURE 3: Node grouping.

FIGURE 4: The consensus process of HC-PBFT.

- (2) Preparation phase: the primary nodes of each group at the first layer count the opinions in the group, agree or reject the pre-preparation message, agree to send 1, refuse to send 0 to the primary, and calculate the statistics by the formula $\text{Vote} = \sum N_i$, where Vote represents the voting result and N_i represents the node's voting node. The opinion of this group is that the voting result of most nodes in the group is that the voting result is greater than $(y+1)/2$. If most nodes in the group agree to the pre-preparation message, the primary nodes of each group broadcast the pre-preparation message to each other. Each group of primary nodes collects preparation messages broadcast by each group of primary nodes and forwards them to the nodes in the group. When a node receives $2f$ opinions that are the same as itself, it means that all nodes in the network have reached a consensus sequence
- (3) Commit phase: when a node receives $2f$ preparation messages consistent with itself, the node sends an agreement to the primary node of the group and it means to enter the confirmation phase. Each group of primary nodes collects opinions in their group; if most of the nodes in the group agree to enter the commit phase, a confirmation message will be broadcast to each group of primary nodes. Then,

each group of primary nodes collects and forwards the confirmation message to the nodes in their group

- (4) Reply phrase: when the node receives $2f$ confirmation messages that are consistent with itself, it means that the nodes in the entire network have reached a consensus on the client's request. The node executes this request, and each group of primary nodes sends a reply message to the client, telling the client that the request has been executed

In the second layer consensus process, the primary node counts the voting results in the group and collects the voting opinions and signatures of the nodes that do not match the final voting results as evidence for evaluating the behavior state of the nodes after the consensus finished. In most improvement schemes, the node evaluation strategy involved is to monitor the behavior of all nodes. Here, this paper adopts the scheme of recording nodes that have different opinions from the final consensus result, aiming to reduce the confirmation of the result record in the node network communication bandwidth and the efficiency of signature verification.

2.5. Dual-View Mechanism. In this paper, a dual-view mechanism is designed for the two-layer consensus process of the HR-PBFT algorithm. The dual-view mechanism consists of

FIGURE 5: The result of view change.

FIGURE 6: Byzantine node determination mechanism.

two view parts. The first part maintains a main view for the first layer nodes, and the first layer nodes run the PBFT three-phase consensus protocol. The second part is that each group of nodes in the second layer maintains a secondary view and each group of nodes runs a simplified consensus protocol.

In the dual-view switching phase, the primary node is selected by taking turns from honest candidate nodes to select the primary node for consensus. Firstly, each group selects the primary node in each group and updates the sub-view of each group. Secondly, the primary nodes of each group constitute the first layer consensus layer and the first layer consensus layer selects the primary nodes of the first layer consensus layer and updates the main view. Finally, the result of the dual-view switching is forwarded to each node and the view change is completed. Suppose that D1 is in the Byzantine node state and the rest of the nodes are in the candidate node state. As shown in Figure 5, it is a diagram for view change. The primary node of the previous view is {A0, B0, C0, and D0}, and A0 is the primary node of the first consensus layer; the primary node changes to {B1, C1, D2, and A1} after the view is switched. B1 is the primary node of the first consensus layer. D1 is restricted from being elected as the master node because it is in the Byzantine node state; the primary node of this group is the D2.

View change will bring high communication complexity, no consensus can be reached during the view change, and no external service will be provided. However, the view does not change for a long time and some nodes remain as the master node, which is not conducive to fairness and the primary node has been in a high-load working state and load balance cannot be guaranteed. In response to these problems, this paper proposes to use the number of blocks produced to trigger a view change mechanism to control the frequency of view switching.

In the consensus process, the consensus layer of the first layer master node maintains a main view and each group of the second layer consensus layer maintains a subview. The nodes in the second layer participate in the consensus process of the first layer primary node consensus layer through this group of primary nodes. If the primary node is a Byzantine node, the group of nodes will not be able to participate in the consensus normally during this view period. For this reason, combine the dual-view mechanism and the hash ring during grouping; when the master node cannot work nor-

mally, the nodes in the group look for the primary node of the next group clockwise and use it as the temporary master node in the current view period and participate in this round through the temporary primary node. Consensus in the view cycle waits for the end of this round of view cycle to reselect the group of primary nodes.

2.6. Byzantine Node Determination Mechanism. This paper proposes a Byzantine node determination mechanism, which divides the node status into four states: primary node, candidate node, error node, and Byzantine node. Among them, the primary node status represents that the node is currently participating in the consensus as the master node, the candidate node status represents that the node is an honest node and has a chance to be elected as the primary node, the wrong node state represents the node downtime behavior, and there is a fault that cannot be elected as the master node. The Byzantine node means that the node has Byzantine behavior and can never be elected as the primary node.

Setting the wrong node status is set for the node that may have a physical failure leading to downtime. If the wrong node behaves normally within the preset view period, it can be reset as a candidate node. However, the malicious node that exhibits Byzantine behavior is set as a Byzantine node; its status can no longer be changed and it will no longer be selected as the primary node.

As shown in Figure 6, the Byzantine node determination mechanism is divided into three parts: check signature, judgment, and update status. Among them, the check signature is to select the abnormal behavior node from the consensus process, the judgment is to analyze and determine the node behavior, and finally update the node status. The Byzantine node determination mechanism can mark the wrong nodes and Byzantine nodes in the node, update them to the corresponding state, and restrict them from being elected as the primary node, which can improve the reliability of master node selection and reduce the probability of frequent switching views.

3. Analysis and Experiment

3.1. Analysis of Consensus Success Rate. After multiple rounds of consensus, the Byzantine node determination

FIGURE 7: Simulation results for success rate.

TABLE 1: Comparison of communication times the three consensus algorithms.

	Pre-prepare	Prepare	Commit	Total
PBFT	$n - 1$	$n * (n - 1)$	$n * (n - 1)$	$2n^2 - n - 1$ (1)
HR-PBFT	$(x - 1) + xy$	$xy + x^2$	$x^2 + 3xy$	$2x^2 + 4xy + x - 1$ (2)
SG-PBFT [13]	$n/2 - 1$	$(n/2 - 1) * (n/2 - 1)$	$n/2 - 1$	$(n/2 - 1) * (n/2 + 1)$ (3)

mechanism can mark Byzantine nodes and error nodes, restrict them from being elected as primary nodes, and improve the reliability of primary node selection, and the success rate of consensus will inevitably increase. Next, design experiments to verify the consensus success rate of the grouped HR-PBFT algorithm and the traditional PBFT algorithm in the initial stage.

In the experiment, the number of nodes x in the first layer is set to 4, representing 4 nodes in the first layer for PBFT consensus and the number of nodes in each group in the second layer is set to 4, representing four nodes in each group in the second layer. If this consensus network runs PBFT algorithm, the network can tolerate a maximum of 6 Byzantine nodes, while running the improved HR-PBFT consensus algorithm in this network can tolerate a maximum of 9 Byzantine nodes. This paper uses the hash ring grouping strategy to group the network nodes. At this time, the consensus process has not been carried out and it is impossible to determine who is the Byzantine node in the network. The node grouping is random, and the primary nodes in the first layer are random election.

As shown in Figure 7, the success rate of consensus in the first view cycle of the PBFT algorithm and the HR-PBFT algorithm is depicted to increase with the number of Byzantine nodes in the network. In the traditional PBFT consensus algorithm, Byzantine nodes elected as primary nodes will inevitably lead to consensus failure and view change. While in the improved algorithm of this paper, only nodes in the first layer with more than $x/3$ nodes are Byzantine nodes or manipulated by Byzantine nodes will lead to consensus failure.

3.2. Analysis of Communication Complexity. This paper uses shard technology and simplifies the consensus process of the second layer to solve the problem of high communication

complexity and poor scalability of PBFT. As shown in Table 1, it is a comparison of the communication complexity of each phrase in the consensus process.

Assuming that the consensus nodes in the network are divided into groups, there are x nodes in the first layer and y nodes in each group of nodes in the second layer. And the total number of nodes in the network is n . The primary node in the main view of the first layer receives a request from the client, verifies and sorts the client request, and enters the pre-preparation phase. After that, the primary node sends the pre-preparation message to other nodes in the first layer. And the nodes receive the pre-preparation message from the primary node and forward it to the nodes in this group in the second layer. This group of nodes has completed the pre-preparation phase, and the communication complexity generated by this process is $(x - 1) + xy$. Entering the preparation phase, the second layer node verifies the request information received in the pre-preparation phase and sends its own opinions to the primary node of the group. Each group primary node counts the opinions of the group and compares the opinions of the group with the first layer node. After the confirmation, the nodes of the first layer will send the results to the nodes in the group. At the end of the preparation phase, the communication complexity generated by this process is $x^2 + xy$. Entering the confirmation phase, the communication behavior between nodes is similar to the preparation phase and the communication complexity generated by this process is $x^2 + 3xy$. The main reason for the high complexity of PBFT communication is the communication between these three-stage nodes. The improved three-stage communication complexity is $x^2 + 4xy + x - 1$. As shown in Table 1, it is a comparison diagram of the communication complexity of each stage of the PBFT algorithm and the improved algorithm.

FIGURE 8: The communication complexity of the three consensus algorithms.

FIGURE 9: Comparison of tolerable Byzantine nodes.

In order to more clearly show the differences between them, the communication complexity is compared at different numbers of network nodes. For example, there are 76 nodes in the network, namely, n is 76. Assuming that each group has 4 nodes, there are 19 groups; that is, the number of nodes in the first layer x is 19 and y is 4. According to the first equation in Table 1, the complexity of PBFT is calculated to be 11475. According to the second equation in Table 1, the complexity of HC-PBFT is 911. And according to the third equation in Table 1, the complexity of SG-PBFT is 1443. As shown in Figure 8, when the number of nodes in the network does not reach 100, the communication complexity of the two consensus algorithms is not much different. However, as the number of nodes increases, the difference between the two increases significantly. The communication complexity of the traditional PBFT algorithm increases rapidly with a square term, while the improved HR-PBFT algorithm increases almost linearly and slowly.

3.3. Fault Tolerance Analysis. The algorithm uses a double-layer verification mechanism. If an error occurs in a certain

link, the entire system will roll back. If the related nodes involved in the subconsensus group reject the operation, the main consensus group can get feedback immediately and feed it back to the corresponding main consensus group node. If a node in the main consensus group rejects the operation, the nodes in the main consensus group will call back the operation failure and the reason to each subconsensus group, forming an upper and lower layer supervision mechanism, which greatly guarantees system security.

In the improved PBFT algorithm, all nodes of the network participate in the consensus process. The first layer still adopts the PBFT three-phase consensus and the first layer tolerates no more than $x/3$ of faulty nodes. The second layer collects the opinions of the nodes in the group through the primary node of each group, and the faulty node tolerated by the second layer does not exceed $(y + 1)/2$.

$$\begin{cases} N = \frac{x}{3} + \frac{x(y+1)}{2}, \\ n = x + xy. \end{cases} \quad (1)$$

The first part of Equation (1) calculates the number of Byzantine nodes that the improved PBFT can tolerate, where x represents the number of nodes in the first layer, y represents the number of nodes in each group in the second layer. The second part of Equation (1) calculates the number of nodes in the network. If this network runs the traditional PBFT consensus algorithm, the maximum number of Byzantine nodes that can be tolerated is $n/3$. This paper compares the improved algorithm tolerance with the original algorithm. In Figure 9, we can see the difference in the number of Byzantine nodes that the HR-PBFT algorithm and the PBFT algorithm can tolerate. As it can be seen from the figure, when the number of nodes in the second-tier group increases, the fault tolerance is better.

3.4. Activity Analysis. The activity of the HR-PBFT algorithm is guaranteed by the dual-view mechanism and the timeout forced view change mechanism. The consensus process of the HR-PBFT algorithm is divided into the consensus between the first layer primary nodes and the second layer consensus among the nodes within the group. For this two-layer consensus mechanism, a dual-view mechanism is proposed, the first consensus layer maintains a main view, and each group of the second layer consensus layer maintains a subview in its own group to ensure the proper operation of the algorithm. In the HR-PBFT algorithm, the primary node is elected in turn among honest candidate nodes. In order to ensure node load balance and fairness, the number of blocks produced is used to trigger view change. At the same time, in order to prevent the Byzantine node in the first layer exceeding $x/3$, the consensus cannot be reached during this view period, this paper proposes a timeout forced view switching mechanism, and the view is forced to switch without reaching a consensus after a predetermined time to ensure the activity of the algorithm.

4. Conclusions

In this paper, an optimization scheme based on hash ring is proposed to solve the problem that the communication complexity increases rapidly with the increase of the number of nodes in the consensus process of the traditional practical Byzantine algorithm and the Byzantine nodes are elected as primary nodes leading to consensus failure and frequent view switching. Firstly, this paper uses a hash ring to group nodes, which ensures the randomness and fairness of the grouping. Secondly, a dual-view mechanism is used in the consensus process, where the first layer nodes maintain the primary view and the second layer nodes maintain the secondary view to ensure the normal operation of the algorithm. Finally, Byzantine node determination mechanism is introduced to evaluate the node state according to the node behavior in the consensus process, improve the reliability of the primary node, and reduce the view switching frequency. In most reputation model improvement schemes, the election of primary nodes is selected from a small number of nodes with high credit values. In this paper, the primary node is elected by turns in the honest candidate nodes. In order to ensure the load balance and fairness of the nodes,

the number of preset blocks is used to trigger the view switching, so as to ensure that each honest candidate node has the opportunity to be elected as the primary node. However, since the primary node election is elected in turn among honest candidate nodes, there is no anonymity may face corrosion attacks. In future work, we will improve and optimize the possible attacks.

Data Availability

The data used to support the finding of this study are included within the article.

Conflicts of Interest

The authors declare that there is no conflict of interest regarding the publication of this paper.

Acknowledgments

This work was supported by the National Key Research and Development Project under Grant (2018YFB1404400) and the National Natural Science Foundation of China under Grant (62062030).

References

- [1] S. Nakamoto, *Bitcoin: A Peer-to-Peer Electronic Cash System*, Bitcoin White Paper, 2008.
- [2] W. L. Chen and Z. B. Zheng, "Blockchain data analysis: a review of status, trends and challenges," *Journal of Computer Research and Development*, vol. 55, no. 9, pp. 1853–1870, 2018.
- [3] O. Novo, "Blockchain meets IoT: an architecture for scalable access management in IoT," *IEEE Internet of Things Journal*, vol. 5, no. 2, pp. 1184–1195, 2018.
- [4] M. Castro and B. Liskov, "Practical Byzantine fault tolerance," in *Proceedings of the Third USENIX Symposium on Operating Systems Design and Implementation (OSDI)*, pp. 173–186, New Orleans, LA, USA, February 1999.
- [5] Z. H. Chen and Q. Li, "Improved PBFT consensus mechanism based on K-medoids," *Computer Science*, vol. 46, no. 12, pp. 101–107, 2019.
- [6] H. Yoo, J. Yim, and S. Kim, "The blockchain for domain based static sharding," in *2018 17th IEEE International Conference on Trust, Security and Privacy in Computing and Communications/12th IEEE International Conference on Big Data Science and Engineering (TrustCom/BigDataSE)*, pp. 1689–1692, New York, NY, USA, August 2018.
- [7] N. Gao, Z. Chuangming, C. Yang, S. Lina, and W. He, "Improvement of PBFT algorithm based on network self-clustering," *Computer Application Research*, vol. 38, no. 11, pp. 1–8, 2021.
- [8] G. Golan Gueta, I. Abraham, S. Grossman et al., "SBFT: a scalable and decentralized trust infrastructure," in *2019 49th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, pp. 568–580, Portland, OR, USA, June 2019.
- [9] X. Zheng, W. Feng, M. Huang, S. Feng et al., "Optimization of PBFT algorithm based on improved C4.5," *Mathematical Problems in Engineering*, vol. 2021, Article ID 5542078, 7 pages, 2021.

- [10] S. Gao, T. Yu, J. Zhu, and W. Cai, "T-PBFT: an EigenTrust-based practical Byzantine fault tolerance consensus algorithm," *China Communications*, vol. 16, no. 12, pp. 111–123, 2019.
- [11] Y. X. Lai, Z. X. Bo, and J. Liu, "Research on sybil attack in defense block chain based on improved PBFT algorithm," *Journal on Communications*, vol. 41, no. 9, pp. 104–117, 2020.
- [12] X. F. Liu, *Research on Performance Improvement of Blockchain Based on Dynamic Fault Authorization of Byzantine Tolerance Consensus Algorithm*, Zhejiang University, Zhejiang, 2017.
- [13] G. Xu, Y. Liu, J. Xing et al., "SG-PBFT: a Secure and Highly Efficient Blockchain PBFT Consensus Algorithm for Internet of Vehicles," *Cryptography and Security*, 2021.