Received January 9, 2022, accepted February 16, 2022, date of publication February 22, 2022, date of current version March 22, 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3153701

Improved Fault-Tolerant Consensus Based on the PBFT Algorithm

JIAN YANG, ZHENHONG JIA, RUIGUO SU, XIAOXIONG WU, AND JIWEI QIN
College of Information Science and Engineering, Xinjiang University, Urumqi 830046, China

Corresponding author: Zhenhong Jia (jzhh@xju.edu.cn)

This work was supported by the Major Project of Xinjiang Uygur Autonomous Region under Grant 2020A03001.

ABSTRACT Nowadays Practical Byzantine Fault Tolerance (PBFT) algorithm has become the most extensive consensus algorithm in the alliance chain. However, the PBFT algorithm is usually only applicable to small networks due to high communication complexity and poor scalability. Although there have been many improved algorithms for PBFT in recent years, they ignore fault tolerance and democracy. Therefore, to meet the requirements of a high degree of decentralization and fault tolerance of blockchain-based scenarios. This paper proposes a high fault tolerance consensus algorithm NBFT, which follows the principle of decentralization and democratization of blockchain and ensures the improvement of performance in fault tolerance upper limit and scalability. First, we use the consistent hash algorithm to group the consensus nodes to avoid much communication between nodes, reduce the communication complexity of the network, and improve the scalability of the network. Second, to ensure the fault-tolerant ability of the grouping consensus, the nodal decision broadcast model and threshold vote-counting model are proposed first. Combined with the proposed two models, the joint fault analysis of nodes is carried out, and the fault tolerance upper limit is more than 1/3. Then, the Faulty Number Determined (FND) model is introduced to simulate the experiment, and the results are verified.

INDEX TERMS High fault tolerance, group consensus, scalability.

I. INTRODUCTION

The Byzantine fault tolerance (BFT) algorithm has been a widespread concern since it was proposed by Leslie Lamport, Robert Shostak, and Marshall Pease in 1982 [1]. However, it has not been applied in practice due to its high communication complexity. It was not until Castro *et al.* proposed the practical Byzantine fault tolerance (PBFT) algorithm [2] in 1999, which reduced the complexity of the original Byzantine fault tolerance algorithm from exponential to polynomial, that the Byzantine fault tolerance algorithm was introduced into the engineering field. At present, PBFT algorithm has become the most widely used consensus algorithm in blockchain alliance chain [3]. Because the Byzantine fault-tolerant consensus algorithm can still ensure the correct consensus of data in the distributed network when Byzantine nodes do evil. Therefore, the Byzantine fault-tolerant consensus algorithm is significant for the development of blockchain technology, ensuring the normal consistency of data of each node in the blockchain distributed network and the order of uplink transactions. The communication complexity, scalability, fault tolerance, and performance of the consensus algorithm will directly affect the performance of blockchain-based applications [4].

Although the PBFT algorithm breaks the performance bottleneck of the original POW algorithm, improves throughput, and reduces transaction confirmation delay, due to the problems of high communication complexity, poor scalability, and poor fault tolerance of the PBFT algorithm, it is difficult for the performance of blockchain-related projects to meet the actual business requirements. The PBFT algorithm is usually only suitable for networks with less than 100 network nodes, which is challenging to be used in more extensive networks [5]. It limits the application of blockchain technology in financial services, energy trading, supply chain management, the Internet of things (IoT), and other fields [6]. Therefore, there are many improved algorithms for PBFT in recent years. For example, literature [7]–[10] combines PBFT with a public chain consensus algorithm to improve consensus efficiency.The core idea is to select a certain number of nodes

The associate editor coordinating the review of this manuscript and approving it for publication was Alessandra De Benedictis.

as committees through Proof-of-Work (PoW) [11], Proof-of-Stake (PoS) [12], and other algorithms, and then the committees generate blocks through the PBFT algorithm. This algorithm is usually suitable for token scenarios, but there is no token circulation in the actual alliance chain. Therefore, a series of Byzantine fault-tolerant algorithms based on credit mechanisms have been proposed [9], [10], [13]–[22]. The consensus nodes are screened through the reputation mechanism, and the consensus of the whole network is turned into the participation of some nodes or committees. Generally, these algorithms can effectively reduce the complexity of communication and improve the ability of network extension. However, these methods will have the following shortcomings. First, the original intention of blockchain technology is decentralization and democratization, while node selection based on reputation tends to be centralized. Therefore, applying such a consensus algorithm in the blockchain is contrary to our original intention of decentralization. Secondly, when selecting nodes by reputation value, some nodes with high reputations have many tasks and consume more resources. As a result, nodes in the network may be prone to laziness. Third, this kind of algorithm reduces the cost of doing evil by the nodes with high reputation and increases the risk of joint evil, thus affecting the consensus security of the entire network. Therefore, it is vital to design an efficient and decentralized consensus algorithm to solve the bottleneck of blockchain.

In addition, privacy security protection has become more critical in recent years. In order to curb all kinds of network attacks, many scholars have put forward many reasonable solutions in network security [23]–[26]. Therefore, to curb the joint evil of Byzantine nodes, the destruction of the entire network consensus. The consistency hash algorithm [27] is used to select the master node and the network consensus node. The identity of the consensus node is hidden in advance to restrain the joint evil of Byzantine nodes.

The contributions of this paper are as follows:

a. First, the consistent hash algorithm is introduced to select and group nodes. Thus, each node has uncertainty about the consensus role in advance, achieving the purpose of restraining the evil nodes.
b. Subsequently, the nodal decision broadcast model and threshold vote-counting model are proposed for the first time to ensure the consensus security. The error tolerance interval of the NBFT algorithm is analyzed according to these two models.
c. Evil acts against possible nodes in the consensus are committed. First, combined with the proposed two models, joint fault analysis is performed on the nodes. The results show that the maximum probability of fault tolerance is more substantial than 1/3. Then, the FND model [4] is introduced to further analyze and prove that the fault tolerance upper limit of the NBFT algorithm will be greater than 1/3.

The rest of the article works as follows. Section II: Introduction of related work. Section III shows the establishment of the NBFT algorithm and model. In Section IV, the fault-tolerant upper limit and communication complexity of the NBFT algorithm is analyzed. Section V summarizes the work of this paper.

II. RELATED WORK

In recent years, the PBFT algorithm is mainly been studied through communication complexity and scalability. First of all, scalability is the biggest bottleneck of PBFT, so many solutions are proposed to solve PBFT scalability. For example, in literature [28], [29], a multi-layer PBFT algorithm is adopted to improve the scalability of the network and avoid the problem of increased communication quantity caused by the increase of consensus nodes. Unlike the multi-layer PBFT algorithm, a multi-center PBFT algorithm is proposed in the literature [30], which also effectively improves the scalability of the network. In addition, M. Zamani *et al.* formed many consensus groups by adopting network sharding to enhance the expansion ability of the network [31], [32]. However, increased network scalability often comes at the expense of system security. For example, there may be Byzantine node proliferation through multi-layer or grouped PBFT consensus. When sharding is adopted, the data loss of any single slice will not be able to query the recorded data [33]. Secondly, the communication volume between nodes is significantly reduced by introducing threshold signature in literature [34]. However, because the synthesis and forwarding of threshold signatures of each link depend on the primary node, its performance is largely related to the number of tasks of the primary node.

Although these algorithms improve PBFT performance, they often ignore the fault tolerance of the consensus network. In practical application, the stronger the fault-tolerant capability is, the stronger the algorithm's applicability will be, ensuring the system's regular operation in a more severe environment. Fault tolerance is vital in all fields, and many scholars have conducted in-depth studies on fault tolerance in recent years [35], [36]. Therefore, the fault tolerance of the algorithm is considered in the design of this paper, and the node decision broadcast model and threshold vote-counting model are proposed for the first time. Its purpose is to hope that the entire network can still reach a correct and consistent consensus under the more severe distributed network environment. In addition, existing studies have sacrificed decentralization to reduce communication complexity. However, the degree of decentralization of the alliance chain itself is lower than that of the public chain. Suppose the consensus of the whole network is participated by some nodes to improve the performance. In that case, both the democracy of consensus and the utilization of network computing resources will be insufficient. With the development of computing hardware, we know that the performance of marginal hardware is also outstanding [37]. We hope that all nodes can participate in the consensus of data together when blockchain technology is applied to IoT scenarios such as the Internet of vehicles, finance, supply chain management, and the more extensive network. This can make full use of the computing resources of the entire network but also better follow the original intention of decentralization and democratization of blockchain. To this end, we designed a mechanism for all nodes to participate in consensus. In addition to ensuring decentralization and democratization, the network scalability, communication complexity, and fault tolerance are also improved.

Our proposed approach differs from previous approaches in several way:

a. To prevent the nodes from doing evil things, we hid the identity of the consensus node so that each node did not know the identity chosen by the primary node and each group of representative nodes in advance.
b. To reduce the complexity of communication, we use the idea of convolutional pooling in the neural network to gradually reduce the consensus group. Each group node reaches a local consensus, and then each group of representative nodes reaches the whole network consensus. In addition, to prevent the consensus security from being reduced through the agreement of representative nodes, two models are proposed for the first time to effectively monitor the behavior of each group of representative nodes
c. Different from other algorithms, we improve network scalability and reduce communication complexity and improve the fault-tolerant upper limit of the network, making the consensus of the whole network more secure and more tolerant. Through analysis and experiments, it is proved that the fault-tolerant upper limit will exceed 1/3.

III. NBFT ALGORITHM DESIGN

This paper assumes that the number of nodes in the whole network is n, and the number of nodes in each group is as follows: m ($m = 3f_1 + 1, f_1 = 1, 2, 3 \ldots\ldots$). Therefore, the number of neutrons in the entire network is as follows: $R = [(n-1)/m]$. In addition, according to the conclusion of $n \geq 3f + 1$ [2] (where f represents the maximum number of Byzantine nodes that can be tolerated), each group can tolerate $E = [(m-1)/3]$ Byzantine nodes at most, and the entire network can tolerate $w = [(R-1)/3]$ group consensus abnormalities at most (where $R, E, w$are integers, $R \geq 4$).

A. NBFT CONSISTENCY PROTOCOL

In the NBFT algorithm, the original PBFT algorithm, where all nodes broadcast to each other, is changed to the consensus within the group. Then, the consensus between the groups is broadcast. The consensus in the group does not represent a global consensus. As a result, to avoid communication that is too trivial, the consensus in the group, with the aid of the Hotstuff algorithm [34], depends on behalf of the primary node aggregation and forwards the message. The actual consensus is shown in Figure 1 and consists of the *preprepare1*, *in-prepare1*, *in-prepare2*, *out-prepare*, *commit*, and *preprepare2* phases. Replica0 serves as the primary node, and the client serves as the customer service. In the *in-prepare1* and *in-prepare2* stages, each m consensus node constitutes a subnetwork. For the convenience of description, the last node of each group is taken as the representative primary node of the group. After the client sends a message to the primary node, the entire network will be triggered, and the final agreed message will be written to the blockchain.

B. GROUPING POLICY AND CONSENSUS NODE SELECTION

We adopt the consistent hash algorithm for the selection of each group node and the primary node [27]. The consistency hashing algorithm has properties, such as anti-collision and uniform dispersion. After each node calculates its hash value through $hash(nodeip)$, it can be evenly mapped to the hash ring of $0 \sim 2^{32}$ (where $nodeip$ represents node ip). The primary node is calculated according to the $hash(masterip + previous_{hash} + view_{number})$ after the hash value of the ring clockwise to the nearest node is the current consensus of the primary node (where $masterip$ represents the ip address of the primary node of the consensus of the last round, $previous_{hash}$represents the hash of the last block, and $view_{number}$ represents the view number) after the primary node selection, according to the hash ring clockwise direction starting from the $[view_{number}/node_{number}]$ nodes(where [ ] represents that is rounded, $node_{number}$ represents the number of nodes in the network), which also skips the primary node every m nodes to a consensus group. The representative primary node of each group is selected according to the hash ring formed by the group and combined with$hash(masterip + view_{number} + group_{number})$ to choose the representative primary node of the group in this round consensus (where $group_{number}$ represents the group number). When grouping for the first time, the primary node is the first node in the clockwise direction of the ring, and the initial value of $view_{number}$ is 0.

The consistency algorithm can not only realize the fast selection of each group and consensus node but also ensure the uncertainty of each node's selection of the group and representative primary node in advance, reducing the risk of the joint evil of the Byzantine nodes.

C. NODE DECISION BROADCAST MODEL

In the *in-prepare2* stage, after the consensus within each group is completed, each representative primary node will participate in the intergroup agreement on behalf of each group. However, whether the consensus result within the group is correct and whether the representative primary node is Byzantine may affect the consensus of the whole network. Therefore, we need to build a model with monitoring capability to monitor the consensus results representative primary node. As a result, we design a node decision broadcast model. When the nodes in the group find that the consensus within the group is abnormal, they do not let the group representative primary node participate in the consensus in the *out-prepare* stage, but they broadcast the message to the representative

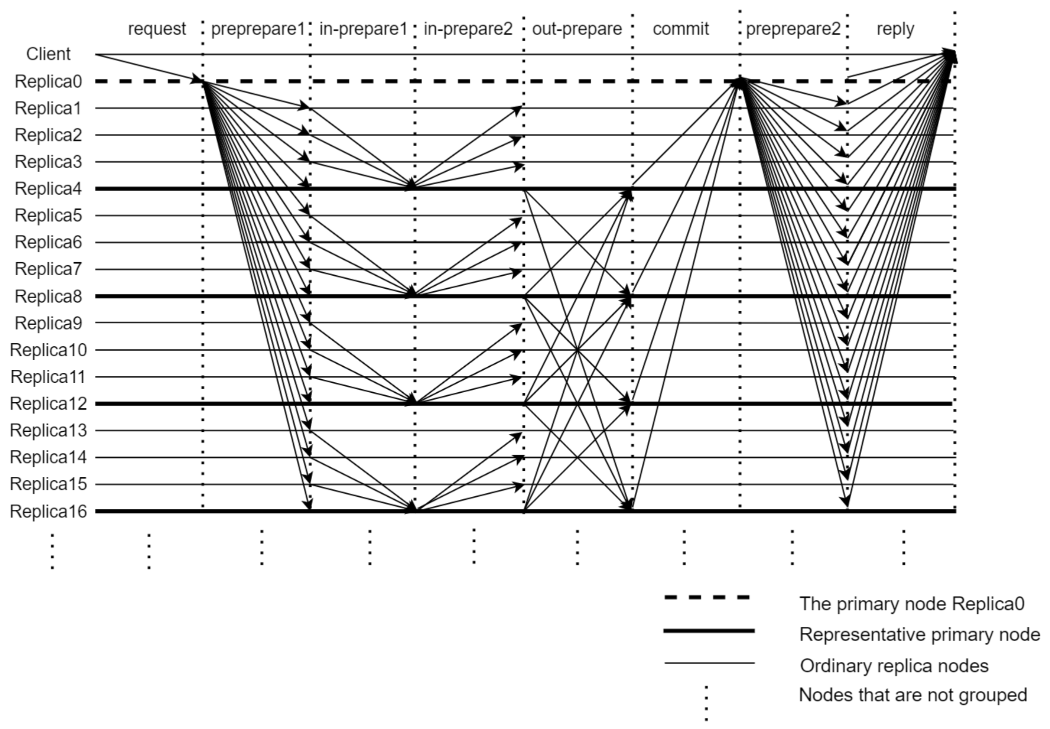

FIGURE 1. NBFT consensus algorithm ($n = 17, m = 4$).

primary node in other groups. Abnormal group consensus includes three situations: (1) The internal nodes of each group discover that the messages representative of the primary node is inconsistent with their messages. (2) The group node fails to receive the message representative of the primary node within the limited time in the *in-prepare2* phase. (3) In the *in-prepare2* phase, the number of aggregated signatures sent by the representative primary node is less than $2E + 1$. In any of the above situations, the nodes in the group will broadcast in the *out-prepare* stage. The pseudocode for the node decision broadcast model is described in Algorithms 1.

D. THRESHOLD VOTE-COUNTING MODEL

The purpose of the node decision broadcast model is to monitor the representative primary node consensus and prevent non-Byzantine valid messages from being delivered to the intergroup consensus. However, after the consensus within the *in-prepare2* stage group, Byzantine nodes may also participate in the broadcast in the *out-prepare* stage. Therefore, to avoid the influence of the Byzantine node vote on the entire network consensus in the *out-prepare* stage, we need to design a vote-counting model to weaken the vote of the Byzantine node. According to the conclusion of $n \geq 3f + 1$ [2], when a network of Byzantine nodes is less than 1/3 of the whole number of network nodes, the Byzantine behavior can actually be tolerated. Therefore, we believe that the number of valid signatures in the group exceeds $m - E$ in the counting process. The consensus votes of the group are counted as m votes; otherwise, the number of valid signatures is calculated as the number of votes. In addition, valid signatures of ordinary replica nodes are counted as 1 vote. Therefore, if all representative primary nodes do not fail in the *out-prepare* stage, some representative primary nodes will reach the consensus threshold votes of the whole network through the threshold vote-counting model. Specific threshold analysis will be introduced in Section IV. Then, after collecting the representative primary nodes that meet the threshold number of votes, all the collected valid signatures consistent with its message are aggregated and forwarded to the primary node. Finally, the primary node will aggregate signature information and statistical vote information as proof of the consensus of the whole network and broadcast the second time to all consensus nodes. Each node verifies the secondary broadcast message from the primary node and sends a reply message to the customer service terminal. When the customer service terminal receives $(n - 1)/2 + 1$ reply to messages from different nodes, it considers that the whole network has reached a consensus.

Therefore, the threshold vote-counting model not only enables the votes of non-Byzantine nodes in each group to obtain good statistics but also weakens the influence of Byzantine nodes within the group on the consensus of the whole network. The pseudocode for the threshold vote-counting model is described in Algorithms 2.

IV. NBFT FAULT TOLERANCE UPPER LIMIT AND COMMUNICATION COMPLEXITY

The models established in Part C and Part D of Section III aim to avoid the influence of Byzantine nodes on the consensus of the whole network. Therefore, to prove that the proposed two

Algorithm 1 The Node Decision Broadcast Model Pseudocode

```
if it is the representative primary node in in-prepare2 phase then
    while the in-prepare2 phase wait timed out do
        Broadcast the aggregation signature messages to the
        representative primary nodes of the other groups in the
        out-prepare phase.
    end while
end if
else then
    if the in-prepare2 phase did not wait for a message
    representative primary node then
        Broadcast messages to representative primary nodes of
        other groups in the out-prepare phase.
    end if
    else then
        if inconsistent with the representative primary node
        message in the group then
            Broadcast messages to representative primary nodes
            of other groups in the out-prepare phase.
        end if
        else then
            if the number of aggregation signatures of the
            representative primary node in the group does not
            exceed $2E + 1$ then
                Broadcast messages to representative primary
                nodes of other groups in the out-prepare phase.
            end if
        end else
    end else
end else
```

Algorithm 2 The Threshold Vote-Counting Mode Pseudocode

```
F: Number of valid signatures
P: Valid votes
H: Total valid votes
if the received signed message is valid then
    if the signature message accepted is from the representative
    primary node then
        if $F \geq m - (m-1)/3$ then
            $P = m$
        end if
        else then
            $P = F$
        end else
    end if
    else then
        $P = 1$
    end else
end if
while $H \geq (R - [(R-1)/3]m)$ do
    All valid messages received in the out-prepare phase are
    aggregated and sent to the primary node
end while
```

models can indeed improve fault tolerance and suppress the influence of Byzantine nodes. So this section combined with node decision broadcast model and threshold vote-counting to analyze NBFT fault tolerance upper limit.

A. ERROR TOLERANCE NUMBER INTERVAL

According to Part A of Section III, we know that in the NBFT consensus algorithm, the final consensus result can be formed only after the intragroup consensus and the intergroup two-round voting consensus. There are R consensus groups in the *out-prepare* stage. To ensure the security of consensus among groups, a maximum of w groups can have abnormal consensus. Therefore, we need to analyze thresholds to ensure overall system safety and activity.

We know that in the PBFT consensus algorithm, the maximum number of Byzantine nodes tolerated is a value, and correct consensus can be guaranteed as long as the number of Byzantine nodes is not greater than 1/3 of the number of consensus nodes in the whole network. However, the error tolerance upper limit of the NBFT algorithm is not a definite value, but it is an interval. In the *out-prepare* stage, if all of the representative primary nodes were Byzantine nodes, the consensus could not be successful. In this case, the minimum error tolerance number is equal to the number of consensus groups R. As seen from the threshold vote-counting model in Part D of Section III, consensus among all $R-w$ groups can be guaranteed when all $R-w$ groups reach the maximum fault-tolerant number. This occurs only when all representative primary nodes cannot fail and the consensus can be successful. Therefore, assuming that all representative primary nodes do not fail, the maximum number of fault-tolerant nodes is the minimum number of the non-Byzantine nodes required to reach the correct consensus. $R-w$ groups have a correct consensus, each group of correct consensus has reached the maximum number of fault tolerances E, and the remaining w groups are all Byzantine nodes. Finally, Formula 1 is obtained, and T represents the maximum number of fault tolerances. Since it is not possible to determine whether a node that is not grouped is a Byzantine node, the formula is simplified and ignores nodes that are not grouped. Finally, we can conclude that the error tolerance interval of the NBFT protocol is $[R, T]$.

$$T = \left[\frac{\left[\frac{n-1}{m}\right] - 1}{3} \right] m + \left[\left[\frac{n-1}{m}\right] - \frac{\left[\frac{n-1}{m}\right] - 1}{3} \right] \times \left[\frac{m-1}{3} \right] \tag{1}$$

B. REPRESENTING THE PROBABILITY THAT ALL PRIMARY NODES ARE BYZANTINE NODES

According to the inference analysis of the NBFT fault tolerance number in Part A of Section IV, when the number of Byzantine nodes in the network exceeds the number of groups in the network, all of the representative primary nodes are Byzantine nodes. Therefore, we conduct a probability

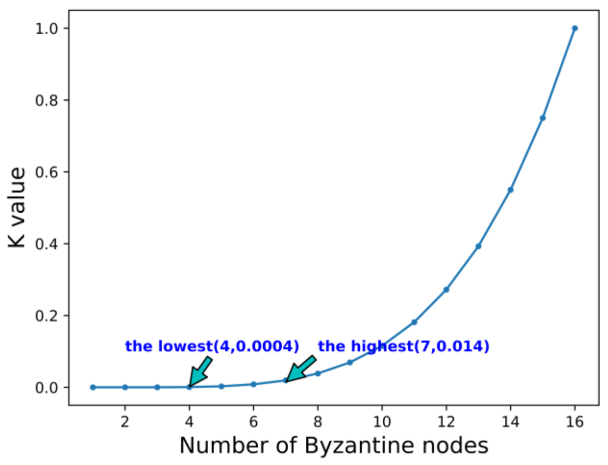

FIGURE 2. Probability of joint evil of principal nodes ($n = 17, m = 4$).

analysis on the situation where all of the main nodes are Byzantine nodes. Assuming that each node is independent of each other, Formula 2 is obtained (Krepresents the probability of failure and i represents the number of Byzantine nodes in the network).C_n^R indicates that when the number of Byzantine nodes exceeds the number of groups, R nodes are randomly selected from n consensus nodes. C_i^R means that the R representative primary nodes are all from the i Byzantine nodes. Finally, the probability obtained is the probability that all representative primary nodes are Byzantine nodes under the current i Byzantine nodes.

$$K = \frac{C_i^R}{C_n^R} \tag{2}$$

According to Formula 2, we took $n = 17$ and $m = 4$ to draw Figure 2. It can be seen that the K value is almost equal to 0 in the interval [4], [7] of the error tolerance number inferred from Part A of Section IV. The probability of all representative primary nodes being Byzantine nodes is very low. We find that as the network grows larger, the probability that all representative primary nodes within the range of fault tolerance numbers are Byzantine nodes tends to be 0. Therefore, we can conclude that when all nodes are in-dependent of each other, R nodes are randomly selected as representatives of the primary nodes within the range of the number of fault tolerances, and the probability that all R nodes are Byzantine nodes is almost 0.

C. P POINTS OF ANALYSIS

According to the interval analysis of the number of fault tolerances in Part A of Section IV, we know that when the number of fault tolerances is maximized, the number of non-Byzantine nodes required to reach correct consensus is the lowest. In other words, if the number of Byzantine nodes in the network exceeds the maximum number of fault tolerances, in the *out-prepare* stage, it is impossible to have a $R - w$ group consensus correctly, and the consensus of the entire network will fail. However, according to the decision broadcast model and the threshold vote-counting model, we know that when the proportion of Byzantine nodes in the network gradually increases, the number of Byzantine nodes does not exceed the maximum number of fault tolerances, and thus, it is also possible to destroy the consensus of the whole network. For example, as long as the Byzantine nodes destroy the $R - w$ group consensus according to a certain grouping mode, each representative primary node in the *out-prepare* stage cannot reach the consensus threshold. Because the minimum cost of destroying the consensus of each group is $E+1$ Byzantine nodes, as long as there are $R-w$ groups and each group has $E + 1$ Byzantine nodes, the total vote count of the whole network cannot exceed $(R - w)m$ votes. As a result, the consensus threshold of the whole network cannot be reached. This phenomenon is called joint evil between groups. Therefore, we hope to deduce the ratio between the minimum number of nodes that cooperate to the number of nodes in the whole network under different networks. We call this ratio the cutoff point P. As long as the ratio of the number of Byzantine nodes is greater than the P value of the current network, the nodes of each group can cooperate in evil.

Before the analysis, we need to explain that the consensus threshold in the *out-prepare* stage requires $R-w$ groups with correct consensus, which does not necessarily mean that there are $R - w$ groups with correct consensus. According to the threshold vote-counting model, the sum of the total votes of all correct consensus groups plus the valid votes of ordinary replica nodes participating in the *out-prepare* stage broadcast is greater than or equal to $(R - w)m$. This is equivalent to the whole network, and at least $R - w$ groups can reach a consensus.

According to the analysis of the P value, it is also impossible to determine whether the replica nodes that are not grouped in the *in-prepare2* phase are Byzantine nodes. Therefore, we still ignore the nodes that cannot be grouped. Assuming that some H groups are damaged, we first deduce the maximum number of damaged groups that the network can tolerate, as shown in Formula 3.

$$\begin{aligned} m([\frac{n-1}{m}] - H) + [m - 1 - [\frac{m-1}{3}]]H \\ \geq [[\frac{n-1}{m}] - \frac{[\frac{n-1}{m}] - 1}{3}]m \end{aligned} \tag{3}$$

In Formula 3, H is the maximum integer, indicating that the maximum number of destructive groups $H_{\max}$ is the amount that the network can tolerate. $H_{\max} + 1$ represents the minimum number of joint evils between groups, and at least $(E+1)(H_{\max}+1)$ Byzantine nodes are required to participate in joint evils between groups. The ratio of this value to n is the value of P at the cutoff point when the current network size is n. When the proportion of Byzantine nodes is greater than that of P, there may be a risk of node unions between groups.

Therefore, we took $m = 4, 7, 10$ to analyze changes of P points under different n values. For convenience, we chose $(n - 1)$ as an integer multiple of m for analysis. The final experimental situation is shown in Figure 3.

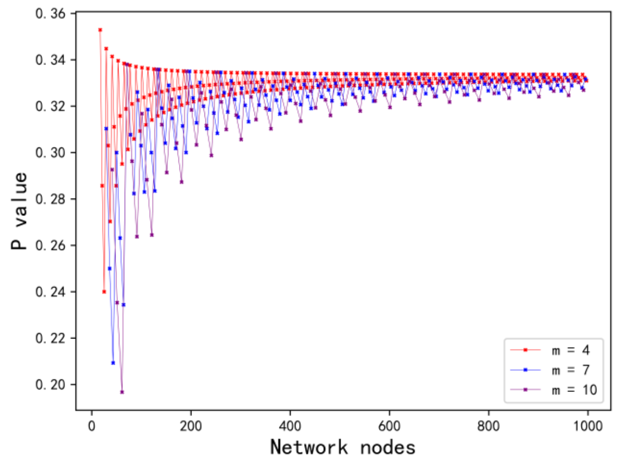

FIGURE 3. ***P*** **points corresponding to different networks ($m = 4, 7, 10$).**

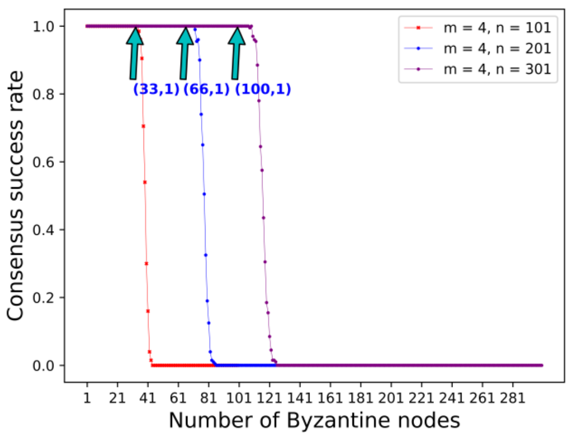

FIGURE 4. Simulated consensus experiment based on the FND model ($m = 4$).

As the original intention of our algorithm design is for the Byzantine fault-tolerant consensus of large networks, we can see from Figure 3 that regardless of how much m is taken, the final P value is approximately 1/3. In other words, in a large network, when the proportion of Byzantine nodes is less than 1/3 of the total number, there will not be a failure of network consensus caused by joint evil between groups, or the probability of failure is very low. Therefore, to further verify that the NBFT algorithm can guarantee the success of consensus when the proportion of Byzantine nodes is less than 1/3, the FND [4] fault-tolerant analysis model is introduced in Part D of Section IV to further analyze the fault-tolerant upper limit of the NBFT algorithm.

D. NBFT SIMULATION CONSENSUS

Based on the analysis of the interval where the fault-tolerant number is obtained in Part A of Section IV and the probability that all of the representative primary nodes are Byzantine nodes in Part B of Section IV we know that the probability that all of the representative primary nodes being Byzantine nodes in the interval where the fault-tolerant number is obtained is almost 0. Then, through the analysis of the P point in Part C of Section IV, we also know that in a large network, the failure of network consensus caused by joint evil between groups will occur only when the proportion of Byzantine nodes exceeds 1/3 of the number of network nodes. Therefore, based on the analysis in the preceding sections, the NBFT algorithm can ensure that when each node is independent of the other, the maximum probability of the network fault tolerance upper limit will exceed 1/3.

So we introduce the FND model [4] and conduct a simulated consensus experiment in combination with the two models established in Part B and Part C of Section IV to analyze the NBFT fault-tolerant upper limit further. The specifications of the system are: intel core i5-9300H, 2.4 GHz processor, 8 GB RAM and 1T storage. When we take $m = 4$ and the network size is 101 201 and 301, respectively, the experiment is carried out, assuming that each node in the experiment is independent of each other. The change in the success rate of the system consensus with the number of Byzantine nodes in the network is analyzed. Under each number of Byzantine nodes, 200 simulation experiments are carried out to calculate the success rate of consensus by counting the times of consensus success.

The final experimental results are shown in Figure 4, and we can see that the fault-tolerant upper limit exceeds 1/3. When the success rate of consensus is less than 1, the corresponding number of Byzantines is greater than 1/3 of the number of consensus nodes in the current network. In addition, as the network expands, the gap between the in-flection point and 1/3 of the point will become increasingly larger. As the network gradually expands when the number of Byzantine nodes is less than 1/3 of the number of network nodes, the probability of forming all representative primary nodes that are Byzantine nodes and the probability of joint sabotage between groups is very low.

We also conducted an experimental analysis on $m = 7$. Finally, as shown in Figure 5, we can still see that the number of Byzantines corresponding to the inflection point is greater than 1/3 of the number of consensus nodes in the current network. As the network increases, the gap between the inflection point and the 1/3 of the number of consensus nodes increases.

Therefore, we can draw another conclusion. When each node is independent of the other, the NBFT fault tolerance upper limit will exceed 1/3. Although when the number of Byzantine nodes is less than 1/3 of the number of network nodes, it is possible that all of the representative primary nodes are Byzantine nodes. However, the probability of this happening is almost 0, as seen through the experiments. In addition, the selection of grouping and consensus nodes adopts the consistent hash algorithm, which ensures the uncertainty of the nodes to the consensus role, which makes it is very difficult for nodes to conduct joint evil. Therefore, the combination of the two models and the consistent hash algorithm can ensure that the fault tolerance upper limit of the NBFT algorithm is greater than 1/3.

In addition, the experimental results also show that when the number of Byzantine nodes exceeds 1/3 of the number

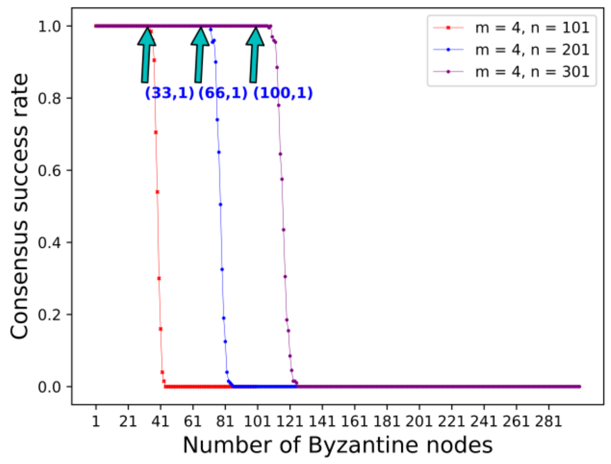

FIGURE 5. Simulated consensus experiment based on the FND model ($m = 7$).

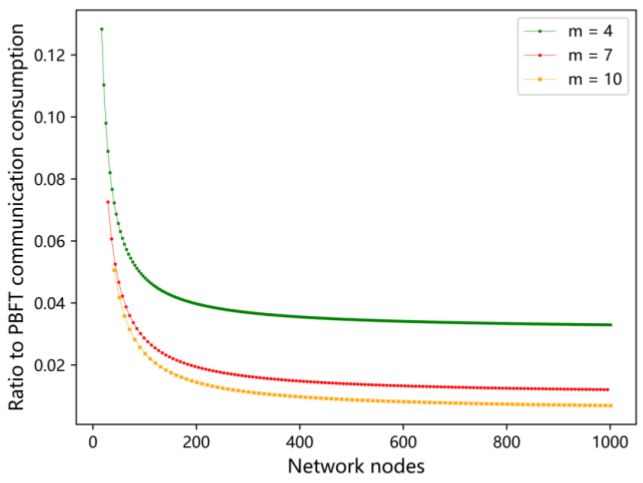

FIGURE 6. The communication consumption ratio of the NBFT and PBFT algorithms under different network sizes.

of network nodes, the consensus of the whole network is guaranteed to be unaffected by Byzantine nodes within a certain range. However, the existing Byzantine fault-tolerant algorithm cannot guarantee the correct consensus when the number of Byzantine nodes exceeds 1/3 of the number of network nodes. Therefore, the fault tolerance performance of our algorithm is better than that of the existing Byzantine fault tolerance algorithm. The reason for this is that the two models we propose weaken the evil ability of some Byzantine nodes and make the whole network more fault-tolerant.

E. NBFT COMMUNICATION COMPLEXITY

According to the overview of NBFT in Part C of Section III, we know that in the *in-prepare2* stage, when consensus anomalies occur among nodes in the group, they will broadcast to consensus nodes in the *out-prepare* stage, resulting in an uncertain number of nodes broadcast in the *out-prepare* stage. Therefore, the communication complexity of the NBFT algorithm is also simplified. It is assumed that only the representative primary nodes of each group are broadcast in the *out-prepare* stage. Based on this assumption, NBFT algorithm traffic is shown in Formula 4, where C represents the total consensus traffic.

$$C = 2(n-1) + 2(m-1)[\frac{n-1}{m}] + [\frac{n-1}{m}]^2 \tag{4}$$

As seen from the expression in Formula 4, the complexity of the NBFT algorithm is $O([(n-1)/m]^2)$. Therefore, we take $m = 4, 7, 10$ to conduct experimental analysis on networks less than 1000 and compare the ratio of communication consumption of the NBFT and PBFT algorithms at different nodes. The experimental results are shown in Figure 6. As seen from Figure 6, compared with the PBFT algorithm, the NBFT algorithm consumes very little traffic regardless of grouping. With the change of the network, different groups can effectively reduce the total traffic and improve the scalability of the network.

Through the experimental analysis and derivation of fault-tolerant communication complexity and expansibility, it was found that the performance of our algorithm has greatly improved compared with the PBFT algorithm. In addition, our algorithm adopts a consistent hashing algorithm for the selection of nodes and the grouping of networks before consensus to hide the identity of nodes in advance without the need for the creation of centralized institutions. In consensus, two models are constructed to ensure the democratization of consensus, meaning that our algorithm has a high degree of decentralization.

Thus, in Table 1, we compare the performance of other mainstream Byzantine fault-tolerant algorithms. Our algorithm has characteristics, such as high fault tolerance and a high degree of decentralization while ensuring that the consensus is decentralized. Although the communication complexity of our algorithm is not optimal, according to the business requirements of the actual scene, we can balance the communication complexity with the number of fault tolerance and the degree of decentralization and adopt different consensus protocols to meet the needs of the actual business. For example, blockchain finance, digital assets, and other financial fields require relatively high levels of security and decentralization, and our algorithm has obvious advantages. We can guarantee that when the number of Byzantine nodes is more significant than 1/3 of the network nodes, the consensus of the entire network can still be correctly consistent within a certain range. In addition, each node can participate in the consensus of the whole network more democratically. However, some alliance chains are not genuinely decentralized. We follow the original intention of decentralization and democratization of blockchain. The original intention of democratization. In the multi-Layer PBFT consensus algorithm, Z is an integer that satisfies Formula 5, $X_{\max}$ satisfies Formula 6, and the communication complexity finally obtained is a dynamic range.

$$Z = 1 + 3 + 3^2 + \ldots\ldots + 3^{\max} \tag{5}$$

$$X_{\max} = \lfloor \log_3 (2Z + 1) \rfloor - 1 \tag{6}$$

TABLE 1. Comparison with other algorithms.

	Byzantine fault tolerance	Fault-tolerant limit	Communication complexity	Degree of decentralization	Scalability
PBFT [2]	Yes	$1/3$	$O(n^2)$	Low	Low
HoneyBFT [38]	Yes	$1/3$	$O(n^2+n^3\log^n)$	Low	Low
Hotstuff [34]	Yes	$1/3$	$O(n)$	Low	High
RAFT [39]	No	$1/2$	$O(n)$	Low	High
Multi-Layer PBFT [4]	Yes	$\le 1/6$	$[O(Z),O(Z^{4/3})]$	Low	High
NBFT (proposed)	Yes	$\ge 1/3$	$O([(n-1)/m]^2)$	High	High

V. CONCLUSION

This paper aims at the problems of lack of democracy and low security in existing alliance chain consensus algorithms. A new fault-tolerant consensus algorithm is proposed, which ensures the decentralized and democratization of consensus nodes in the network and improves the performance of scalability and communication complexity. The experimental results show that the NBFT fault-tolerant consensus algorithm is superior to the existing Byzantine fault-tolerant consensus algorithm. New solutions are proposed for some scenarios with high security and decentralization requirements, such as financial services based on blockchain technology, energy trading, supply chain management, Internet of Things (IoT), and other fields, which contribute to the development of alliance chain. In addition, the algorithm in this paper is mainly for single-layer consensus networks, and its scalability needs to be improved. Therefore, we will further consider the multi-layer consensus network in future research and ensure that the fault-tolerant performance of the network is still outstanding, which will significantly improve the application scope of our algorithm. Promote the application of blockchain alliance chains to more extensive networks and more severe distributed scenarios.

REFERENCES

[1] L. Lamport, R. Shostak, and M. Pease, "The Byzantine generals problem," *ACM Trans. Program. Lang. Syst.*, vol. 4, no. 3, pp. 382–401,1982.

[2] M. Castro, "Practical byzantine fault tolerance," in *Proc. 3rd Symp. Operating Syst. Des. Implement.*, 1999, pp. 173–186.

[3] O. Onireti, L. Zhang, and M. A. Imran, "On the viable area of wireless practical Byzantine fault tolerance (PBFT) blockchain networks," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2019, pp. 1–6.

[4] W. Li, C. Feng, L. Zhang, H. Xu, B. Cao, and M. A. Imran, "A scalable multi-layer PBFT consensus for blockchain," *IEEE Trans. Parallel Distrib. Syst.*, vol. 32, no. 5, pp. 1146–1160, May 2021.

[5] 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 *Proc. IEEE 36th Symp. Reliable Distrib. Syst. (SRDS)*, Sep. 2017, pp. 253–255.

[6] M. H. Miraz and M. Ali, "Applications of blockchain technology beyond cryptocurrency," *Ann. Emerg. Technol. Comput.*, vol. 2, no. 1, pp. 1–6, Jan. 2018.

[7] Y. Meng, Z. Cao, and D. Qu, "A committee-based Byzantine consensus protocol for blockchain," in *Proc. IEEE 9th Int. Conf. Softw. Eng. Service Sci. (ICSESS)*, Nov. 2018, pp. 1–6.

[8] R. Pass and E. Shi, "Hybrid consensus: Efficient consensus in the permissionless model," in *Proc. 31st Int. Symp. Distrib. Comput. (DISC)*, Oct. 2017, p. 39:1-39:16.

[9] G. Golan Gueta, I. Abraham, S. Grossman, D. Malkhi, B. Pinkas, M. Reiter, D.-A. Seredinschi, O. Tamir, and A. Tomescu, "SBFT: A scalable and decentralized trust infrastructure," in *Proc. 49th Annu. IEEE/IFIP Int. Conf. Dependable Syst. Netw. (DSN)*, Jun. 2019, pp. 568–580.

[10] M. M. Jalalzai, C. Busch, and G. G. Richard, "Consistent BFT performance for blockchains," in *Proc. 49th Annu. IEEE/IFIP Int. Conf. Dependable Syst. Netw. (DSN-S)*, Jun. 2019, pp. 17–18.

[11] S. Nakamoto. (2008). *Bitcoin: A Peer-to-Peer Electronic Cash System*. White Paper. [Online]. Available: http://www.bitcoin.org/bitcoin.pdf

[12] P. Vasin. (2014). *Blackcoin's Proof-of-Stake Protocol V2*. White Paper. [Online]. Available: https://blackcoin.co/blackcoin-pos protocol-v2-whitepaper.pdf

[13] X. Zheng, W. Feng, M. Huang, and S. Feng, "Optimization of PBFT algorithm based on improved C4.5," *Math. Problems Eng.*, vol. 2021, Mar. 2021, Art. no. 5542078.

[14] Y. Meng, Z. Cao, and D. Qu, "A committee-based Byzantine consensus protocol for blockchain," in *Proc. IEEE 9th Int. Conf. Softw. Eng. Service Sci. (ICSESS)*, Nov. 2018, pp. 1–6.

[15] R. Pass and E. Shi, "Hybrid consensus: Efficient consensus in the per missionless model," in *Proc. 31st Int. Symp. Distrib. Comput. (DISC)*, Oct. 2017, p. 39.

[16] H. Wang and K. Guo, "Byzantine fault tolerant algorithm based on vote," in *Proc. Int. Conf. Cyber-Enabled Distrib. Comput. Knowl. Discovery (CyberC)*, Oct. 2019, pp. 190–196.

[17] W. Tong, X. Dong, and J. Zheng, "Trust-PBFT: A peertrust-based practical Byzantine consensus algorithm," in *Proc. Int. Conf. Netw. Netw. Appl. (NaNA)*, Oct. 2019, pp. 344–349.

[18] G. Yu, B. Wu, and X. Niu, "Improved blockchain consensus mechanism based on PBFT algorithm," in *Proc. 2nd Int. Conf. Adv. Comput. Technol., Inf. Sci. Commun. (CTISC)*, Mar. 2020, pp. 14–21.

[19] S. Gao, T. Yu, J. Zhu, and W. Cai, "T-PBFT: An eigentrust-based practical Byzantine fault tolerance consensus algorithm," *China Commun.*, vol. 16, no. 12, pp. 111–123, Dec. 2019.

[20] J. Kang, Z. Xiong, D. Ye, D. I. Kim, J. Zhao, and D. Niyato, "Toward secure blockchain-enabled internet of vehicles: Optimizing consensus management using reputation and contract theory," *IEEE Trans. Veh. Technol.*, vol. 68, no. 3, pp. 2906–2920, Mar. 2019.

[21] W. Fang, Z. Wang, H. Song, Y. Wang, and Y. Ding, "An optimized PBFT consensus algorithm for blockchain," *J. Beijing Jiaotong Univ.*, vol. 43, no. 5, pp. 58–64, Oct. 2019.

[22] Y. Wang, S. Cai, C. Lin, Z. Chen, T. Wang, Z. Gao, and C. Zhou, "Study of blockchains's consensus mechanism based on credit," *IEEE Access*, vol. 7, pp. 10224–10231, 2019.

[23] Z. Wu, S. Shen, H. Zhou, H. Li, C. Lu, and D. Zou, "An effective approach for the protection of user commodity viewing privacy in e-commerce website," *Knowl.-Based Syst.*, vol. 220, May 2021, Art. no. 106952.

[24] T. Wang, M. Z. A. Bhuiyan, G. Wang, L. Qi, J. Wu, and T. Hayajneh, "Preserving balance between privacy and data integrity in edge-assisted Internet of Things," *IEEE Internet Things J.*, vol. 7, no. 4, pp. 2679–2689, Apr. 2020.

[25] P. J. Sun, "Research on the tradeoff between privacy and trust in cloud computing," *IEEE Access*, vol. 7, pp. 10428–10441, 2019.

[26] J. A. Alzubi, J. Selvakumar, O. Alzubi, and R. Manikandan, "Decentralized Internet of Things," *Indian J. Public Health Res. Develop.*, vol. 10, no. 2, pp. 251–254, Feb. 2019.

[27] D. Karger, E. Lehman, T. Leighton, R. Panigrahy, M. Levine, and D. Lewin, "Consistent hashing and random trees: Distributed caching protocols for relieving hot spots on the world wide web," in *Proc. 29th Annu. ACM Symp. Theory Comput.*, May 1997, pp. 654–663.

[28] H. Qushtom, J. Misic, X. Chang, and V. B. Misic, "A scalable two-tier PBFT consensus for blockchain-based IoT data recording," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Jun. 2021, pp. 1–6.

[29] W. Lv, X. Zhou, and Z. Yuan, "Design of tree topology based byzantine fault tolerance system," *J. Commun.*, vol. 38, no. Z2, pp. 143–150, 2017.

[30] M. Du, Q. Chen, and X. Ma, "MBFT: A new consensus algorithm for consortium blockchain," *IEEE Access*, vol. 8, pp. 87665–87675, 2020.

[31] M. Zamani, M. Movahedi, and M. Raykova. (2018). *RapidChain: Scaling Blockchain Via Full Sharding Dfinity Palo Alto, CA*. White Paper. [Online]. Available: https://eprint.iacr.org/2018/460.pdf

[32] L. Luu, V. Narayanan, C. Zheng, K. Baweja, S. Gilbert, and P. Saxena, "A secure sharding protocol for open blockchains," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, Oct. 2016, pp. 17–30.

[33] P. J. Sadalage and M. Fowler, *NoSQL Distilled: A Brief Guide to the Emerging World of Polyglot Persistence*. London, U.K.: Pearson, 2013.

[34] M. Yin, D. Malkhi, M. K. Reiter, G. G. Gueta, and I. Abraham, "Hotstuff: BFT consensus with linearity and responsiveness," in *Proc. ACM Symp. Princ. Distrib. Comput.*, 2019, pp. 347–356.

[35] S. Xiao and J. Dong, "Distributed fault-tolerant containment control for linear heterogeneous multiagent systems: A hierarchical design approach," *IEEE Trans. Cybern.*, vol. 52, no. 2, pp. 971–981, Feb. 2022, doi: 10.1109/TCYB.2020.2988092.

[36] S. Xiao and J. Dong, "Cooperative fault-tolerant containment control for nonlinear multiagent systems with switching directed topologies based on hierarchical mechanism," *IEEE Trans. Syst., Man, Cybern., Syst.*, early access, Nov. 22, 2021, doi: 10.1109/TSMC.2021.3125772.

[37] X. Lin, J. Wu, S. Mumtaz, S. Garg, J. Li, and M. Guizani, "Blockchain-based on-demand computing resource trading in IoV-assisted smart city," *IEEE Trans. Emerg. Topics Comput.*, vol. 9, no. 3, pp. 1373–1385, Jul. 2021.

[38] A. Miller, Y. Xia, K. Croman, E. Shi, and D. Song, "The honey badger of BFT protocols," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, Oct. 2016, pp. 31–42.

[39] D. Ongaro and J. Ousterhout, "In search of an understandable consensus algorithm," in *Proc. USENIX Annu. Tech. Conf.*, 2014, pp. 305–320.

JIAN YANG received the B.S. degree in communication engineering from Tarim University, Xinjiang, China, in 2019. From 2017 to 2018, he was an Exchange Student at Lanzhou University, Gansu, China. He is currently pursuing the graduate degree with the School of Information Science and Engineering, Xinjiang University, Xinjiang. His research interests include smart contract and consensus mechanism.

ZHENHONG JIA received the B.S. degree from Beijing Normal University, Beijing, China, in 1987, and the M.S. and Ph.D. degrees from Shanghai Jiao Tong University, Shanghai, China, in 1995. He is currently a Professor with the Signal and Information Processing Laboratory, Xinjiang University, China. His research interests include digital image processing, optical networks, photoelectric information detection, and sensors.

RUIGUO SU received the B.S. degree in communication engineering from the City Institute, Dalian University of Technology, in 2019. He is currently pursuing the master's degree with Xinjiang University. His research interest includes blockchain smart contracts and applications.

XIAOXIONG WU is currently pursuing the degree in information science and engineering with Xinjiang University. His research interests include the development of food traceability systems based on blockchain technology and the research of blockchain fault-tolerant algorithms.

JIWEI QIN received the master's and Ph.D. degrees from the School of Computer Architecture, Xian Jiaotong University, Xi'an, China, in 2008 and 2013, respectively. She is currently a Professor with Xinjiang University. Her research interest includes intelligent networks.