electronics

Article

Fibonacci Group Consensus Algorithm Based on Node Evaluation Mechanisms

Xueli Shen and Xinru Li *

School of Software, Liaoning Technical University, Huludao 125105, China; shenxueli@lntu.edu.cn
* Correspondence: lxr1999123@163.com

Abstract: In response to challenges posed by the Practical Byzantine Fault Tolerance (PBFT) consensus algorithm, where all nodes are involved in the consensus process, issues such as high communication overhead in the three-phase procedure, random selection of master nodes, and the absence of reward and penalty mechanisms are addressed. This leads to the proposal of a Speculative Practical Byzantine Fault Tolerance (SP-PBFT) consensus algorithm based on a node evaluation mechanism for speculative grouping. Firstly, the consensus protocol was optimized, and a timeout mechanism was proposed to divide the consensus process into an optimistic mode and a pessimistic mode, which reduced the communication overhead under the premise of resisting Byzantine node attacks. Secondly, a node evaluation mechanism was proposed to calculate the node reputation value according to the node's historical behavior and give the corresponding reward or punishment, which reduced the possibility of malicious nodes participating in the consensus process. Finally, the Fibonacci grouping mechanism was used to reduce the number of nodes participating in the consensus process, fundamentally improving the consensus efficiency, avoiding the problem of centralization of the consensus process caused by the cumulative reputation value of nodes, and improving the enthusiasm of consensus nodes. Simulation experiments using Docker containers to simulate multiple nodes show that the SP-PBFT consensus algorithm proposed in this paper has better performance than the PBFT consensus algorithm and other improved algorithms in terms of consensus delay, throughput, fault tolerance and communication complexity.

Keywords: blockchain; consensual algorithm; PBFT algorithm; node evaluation mechanism; Fibonacci grouping

check for updates

Citation: Shen, X.; Li, X. Fibonacci Group Consensus Algorithm Based on Node Evaluation Mechanisms. *Electronics* **2023**, *12*, 3592. https://doi.org/10.3390/electronics12173592

Academic Editor: Andrei Kelarev

Received: 26 July 2023
Revised: 20 August 2023
Accepted: 24 August 2023
Published: 25 August 2023

Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

In recent years, blockchain technology has gained significant research attention as a decentralized distributed ledger system [1]. Initially, its primary purpose was to digitally record transactions and achieve the goal of replacing physical cash with digital currency. However, as blockchain technology continues to evolve, it has found extensive applications in the financial services sector [2], as well as in public services, the Internet of Things (IoT) [3], reputation systems, federated learning [4], crowdsourcing platforms, and security services. By incorporating blockchain technology, it becomes possible to establish a trustworthy distributed environment. As a result, blockchain technology holds promising prospects for a wide range of applications [5].

Consensus algorithms are a crucial component of blockchain technology, addressing the issue of distrust among nodes in a distributed system and ensuring all nodes reach consensus on the state of transactions and blocks. However, traditional consensus algorithms can present issues such as resource wastage, insufficient decentralization, and security concerns [6]. Consequently, research and application of consensus algorithms tailored to different scenarios [7] constitute a prominent area of study in blockchain technology. The goal of consensus algorithms is to achieve agreement among multiple nodes regarding a specific transaction without the need for a central authority. In distributed systems, due to factors like network latency and node failures, nodes may have differing views and states, necessitating consensus algorithms to resolve data consistency issues. Moreover, consensus algorithms play a critical role in verifying and updating blockchain ledger data [8]. They ensure that nodes can communicate, validate each other, and ultimately reach unanimous decisions throughout the system without relying on a centralized entity. They form the core component that directly shapes system behavior and achievable performance. Different application domains of cryptocurrencies [9] require corresponding consensus algorithms to meet their unique requirements. This fact not only drives the demand for adapting existing consensus algorithms to new settings but also necessitates continuous innovation in consensus algorithms. In summary, consensus algorithms continuously evolve alongside research and advancements in distributed systems. Currently, many different consensus algorithms have been proposed and applied. Below are some representative consensus algorithms. Proof of Work (PoW) [10]: A computational puzzle-based algorithm where miners compete to solve complex mathematical problems to validate transactions and add new blocks to the blockchain. Proof of Stake (PoS) [11]: Validators are chosen based on the number of cryptocurrency tokens they hold and are willing to "stake" as collateral. This reduces the need for energy-intensive computations. Delegated Proof of Stake (DPoS) [12]: Token holders vote for delegates who are responsible for validating transactions and maintaining the blockchain's integrity. Practical Byzantine Fault Tolerance (PBFT) [13]: This is a classical consensus algorithm ensuring consensus in a distributed network with a known and limited number of faulty nodes. Among these algorithms, the PBFT algorithm is considered a consensus mechanism that can tolerate Byzantine faults, prevent forks, and provide high performance and fast confirmation. From this point of view, the PBFT algorithm is an ideal consensus algorithm, but the PBFT algorithm still has shortcomings, such as large communication overhead in the three-phase process, all nodes participate in the consensus process, and it is not suitable for dynamic networks, etc. Therefore, many experts and scholars have improved PBFT [14–18].

PBFT algorithm is divided into three protocols: consensus protocol, view switching protocol and checkpoint protocol. Many improved algorithms are based on the optimization of the above three aspects. Wang [14] proposed an improved PBFT consensus algorithm with random selection of master nodes, which optimized the three-stage process to reduce communication complexity. At the same time, nodes could dynamically join and exit to improve system availability, but the random selection of master nodes made the consensus process lack security and easily triggered the view-switching protocol. Fang [15] proposed an improved PBFT algorithm using ring signature, which automatically formed rings when the nodes were signed to increase the dynamics of consensus and optimize the efficiency of view switching. Zhang [16] proposed an improved Byzantine fault-tolerant consensus algorithm of the recommended trust model, which added a reward and punishment mechanism and did not randomly select the master node and consensus node to improve the security of consensus, but the problem of large communication overhead still existed. Ren [17] also proposed the PBFT consensus algorithm based on the trust evaluation model. Although the consensus nodes were grouped to improve the consensus efficiency, the consensus protocol process did not change, and the communication complexity was only slightly less than O (n^2). Qin [18] proposes an improved PBFT consensus algorithm that classifies nodes via the clustering method. According to the response situation of nodes in the consensus process as the data dimension, K-means++ clustering algorithm is combined with it to reduce the number of nodes participating in the consensus process and improve the quality of consensus nodes, but the number of communications in the three stages is not greatly improved.

The blockchain field has widely applied many mathematical tools and technologies. Caldarola [19] proposed elliptic curve cryptography, which uses points on the elliptic curve to achieve encryption and decryption. Shorter key lengths can be used to provide the same security, thereby improving the efficiency of encryption and decryption. It is widely used in fields such as protecting digital signatures, identity verification, data encryption, and secure communication. Patel [20] combines blockchain technology with convolutional neural networks (CNNs) to construct a new intelligent framework, such as BlockCrime. Blockchain technology can be used to store detected crime scene locations and alert the nearest law enforcement agencies, thereby improving public safety. Meanwhile, CNNs can be used to detect malicious activities, such as the Xception model used in BlockCrime. This model uses convolutional and pooling layers to extract image features and uses fully connected layers for classification. By using deep learning and blockchain technology, the BlockCrime framework can automatically detect malicious behavior and improve public security. Various mathematical tools, including deep learning, convolutional neural networks, gradient descent algorithms, Adam optimizers, cryptography [21], and exponentially weighted moving average, all provide great inspiration for this article. So, as a recursively defined sequence, the Fibonacci function has the following characteristics: The value of the Fibonacci function increases exponentially with the increase in n, so it can be used for node grouping so that the number of nodes in each group does not differ significantly, thereby reducing the difference in node reputation values and reducing the possibility of nodes with high reputation values becoming the main node. The definition of the Fibonacci function has recursive properties, so it can be used for grouping, making the nodes in each group have a certain degree of correlation, thereby improving the security of consensus algorithms. The definition of the Fibonacci function has simple properties, is easy to implement and calculate, and can, therefore, be used for node grouping, improving the efficiency of consensus algorithms.

In order to address the challenges posed by the improved PBFT algorithm, which still exhibits a significant communication overhead, employs random selection of the master node, faces certain limitations in achieving consensus within dynamic networks, and encounters difficulty in identifying Byzantine nodes within the system, this paper introduces a Fibonacci group consensus algorithm based on a node evaluation mechanism (SP-PBFT). This algorithm is grounded in an overarching network perspective and involves the adaptive selection of consensus nodes. The approach employs a speculative method that takes into account the network environment of the system [22]. It tailors its steps to different network conditions to facilitate consensus and enhance the consensus protocol. Concurrently, it enhances the node evaluation mechanism and organizes nodes according to the Fibonacci function [23]. This approach reduces the count of consensus nodes and contributes to improved efficiency. Furthermore, the participation of nodes in the consensus process potentially leads to greater reliability, streamlining the view-switching protocol.

2. Design of SP-PBFT Consensus Algorithm

In view of the general situation, the traditional PBFT algorithm is rarely attacked by malicious nodes in the consensus, and the Byzantine consensus algorithm will only work when malicious nodes interfere with the consensus process. Therefore, the three-stage communication process of the PBFT algorithm can be simplified, and the optimization idea of a speculative algorithm can be used to divide the PBFT algorithm into two modes. (1) In the optimistic mode, nodes behave without delay and hardly apply any protection against malicious nodes. The emphasis on consensus efficiency makes the algorithm in the optimistic mode competitive with the centralized system in terms of speed. (2) Pessimistic mode, the optimistic mode triggered by timeout and security measures against malicious nodes are added. The only goal is to successfully reach consensus even if there are malicious nodes.

According to the optimization idea shown in the form of the speculative PBFT algorithm, it is found that this consensus algorithm can quickly complete the consensus and improve the efficiency of consensus. In summary, the characteristics of the speculative consensus algorithm are shown in Table 1.

Table 1. Characteristics of speculative consensus algorithms.

Classification	Characteristic	Advantage
Optimistic mode	Node behavior does not apply any protective measures to resist malicious node attacks without delay and with the main node secure	Quickly reach consensus and reduce communication complexity
Pessimistic mode	When there is a delay in node behavior or when the main node is a malicious node, protective measures are added to the algorithm to resist malicious node attacks	Being able to identify Byzantine nodes in the system and improve the security of the consensus process

2.1. Algorithm Improvement Proposal

The best case is network synchronization (all messages are delivered within the maximum delay d), d is less than the timeout threshold t_1, simplifying the acknowledgement phase to reduce node interactions in traditional PBFT algorithms, thus reducing the communication overhead in the consistency protocol, generating blocks as fast as possible, and the SP-PBFT algorithm in the optimistic mode, as shown in Figure 1.

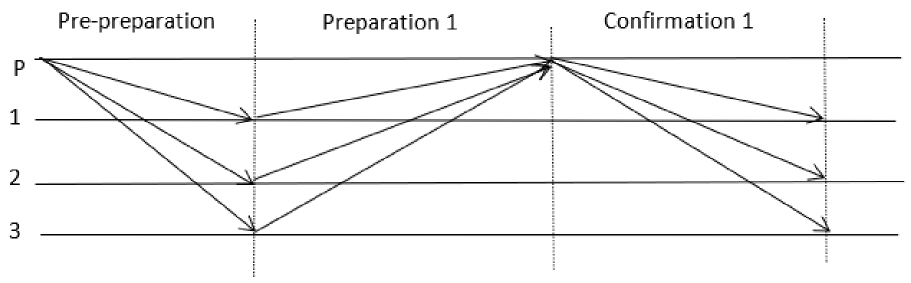

Figure 1. Consensus Process of SP-PBFT Algorithm under Optimistic Mode.

(1) Pre-preparation: the master node sends <<Pre-prepare,v,h,H(B),P>,B> message to all the backup nodes, and the backup nodes receive the provided Pre-prepare message.
(2) Preparation 1: The backup node i returns the <<Prepare-1,v,h,H(B),i>,Vote$_t$(B^h,i)> message to the master node after receiving the Prepare message, and the master node receives the provided Prepare-1 message.
(3) Confirmation 1: If the master node receives 2f + 1 consistent Prepare-1 messages from different backup nodes, i.e., the master node has 2f + 1 (possibly including its own) affirmative votes on block B^h, it can immediately synchronize block B^h and broadcast the message <<Commit-1,v,h,H(B),p>,Cert(B^h)>> to all the backup nodes, and if the backup node receives the Commit-1 message before t_1, it can synchronize block B^h, and the backup node receives the provided Commit-1 message.

In the optimistic mode, the consensus timeout is transferred to the pessimistic mode; the worst-case scenario is that while guaranteeing security, f out of 3f + 1 nodes are malicious nodes and refuse to generate blocks, the SP-PBFT algorithm still outperforms the PBFT algorithm in the pessimistic mode, adding an additional confirmation phase to enable the system to succeed in the consensus, and at the same time, adding the node evaluation mechanism, which reduces the possibility of the malicious node to become the master node again, effectively reducing view switching so that the consensus efficiency is improved, and the consensus process is patched up, as shown in Figure 2.

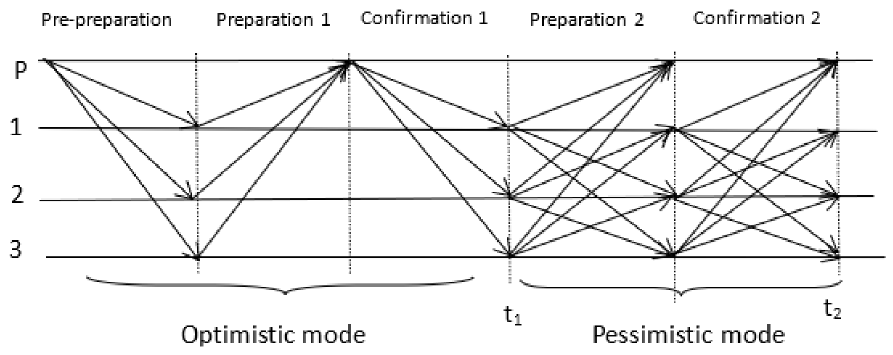

Figure 2. Overall consensus process of SP-PBFT algorithm.

(4) Preparation 2: If backup node i does not receive a Commit-1 message before t_1, it broadcasts a <<Prepare-2,v,h,H(B),i>,Vote$_t$(B^h,i)> message to all other consensus nodes, which receive the provided Prepare-2 message.

(5) Confirm 2: If backup node i receives a Prepare-2 message and a Commit-1 message, it broadcasts a <<Commit-2,v,h,H(B),i>,Cert(B^h)> message to all other backups, otherwise it waits until it receives at least 2f + 1 (possibly including its own) consistent Prepare-2 messages, then broadcasts a <<Commit-2,v,h,H(B),i>,VoteSet(B^h,i)> message to all other consensus nodes, and if the backup receives a Commit-2 message containing the block certificate or receives at least 2f + 1 consistent Commit-2 messages containing the vote set prior to t_2, they can synchronize the block B^h, and the backup node receives the provided Commit-2 message.

The SP-PBFT consensus algorithm is designed against the background of consortium blockchain. In the consortium blockchain, most of the nodes are honest, so the SP-PBFT algorithm will generally execute the optimization PBFT protocol in an optimistic mode. When a Byzantine node appears in the master node and executes the optimized consensus protocol, this node re-enters the node evaluation mechanism, and its score is reduced. The node with a high reputation value in a certain range is regarded as the new consensus node, which ensures the honesty of the consensus node and reduces the attack of malicious nodes. The SP-PBFT algorithm will continue to execute the optimized consistency protocol to complete the next consensus operation.

The specific improvement scheme is divided into the following points:

Firstly, the optimization idea of speculation was proposed, and the algorithm was divided into an optimistic mode and a pessimistic mode. According to the network environment of the system, the speculative method was adopted, and different steps were applied to different network environments to reach consensus so as to optimize the consensus protocol. Adding the restriction that a set of nodes cannot participate in consensus twice in a row eliminates forks.

Secondly, a node evaluation mechanism is proposed, which rewards or punishes nodes based on their historical behavior so that the nodes with higher reputation value are more likely to be the master nodes, and the nodes with poor performance cannot be selected. At the same time, the Byzantine nodes in the system are identified to optimize the view-switching protocol.

Finally, the Fibonacci function was used to group the nodes according to their reputation value, and half of the high reputation value of each group was taken as the consensus node, which reduced the number of nodes participating in the consensus and fundamentally improved the efficiency. At the same time, it prevents the centralization of the consensus process due to the accumulation of reputation value, improves the enthusiasm of nodes, conforms to the decentralized characteristics of blockchain, and makes the consensus process more secure and fair.

2.2. Algorithm Notation

(1) The algorithm contains a total of n nodes; the set of consensus nodes is denoted by R, including master and slave nodes; the ith consensus node is denoted by R_i; and the set of candidate nodes is denoted by s.
(2) The algorithm has certain requirements on the number of nodes; if there are f Byzantine nodes in the system, the total number of all the nodes must be made larger than 3f + 1 to accomplish the consensus, which is usually made equal to 3f + 1 under experimental conditions to reduce the load on the system.
(3) Meanings of the symbols in the system: v: view number; p: master node; h: block height; B: block; H(B): summary of block B; B^h: block B has height h; Vote(B^h,i): votes on block B^h generated by node i; VoteSet(B^h,i): node i receives a set of votes on block B^h; Cert(B^h): a block certificate, which contains on block B^h at least 2f + 1 favorable votes, can be used to verify the validity of block B^h.

2.3. Node Evaluation Mechanism

To enhance system efficiency, a reduction in the frequency of master node view replacements becomes imperative. This reduction facilitates the selection of non-malicious and secure nodes as master nodes. The SP-PBFT algorithm employs a node evaluation mechanism that categorizes nodes into two distinct types based on their reputation values. The first category comprises consensus nodes tasked with active participation in the consensus process, denoted by N-f in number. The second category consists of candidate nodes, denoted by f in number, which do not partake in the consensus process but respond to consensus outcomes. Within the SP-PBFT algorithm, every node upholds a node evaluation mechanism and designates the master node based on evaluations. A high reputation value indicates a credible node that performs effectively during the consensus process. In the algorithm's initial state, the SP-PBFT system identifies a random number. The time taken by a node to identify this random number, the degree of broadcast, transaction engagement, and computed trust score collectively serve as metrics for the node evaluation mechanism. The pseudocode for initializing node reputation values is outlined as follows (Algorithm 1), where C represents the threshold for consensus node reputation value and times denotes the consensus count.

Algorithm 1 Node Reputation Value Initialization.

```
Input: Time, Report, Trade, Credit, C, times, R, S;
1: Begin
2:     set R = {$R_i$ | $0 \le i \le |R| - 1$}
3:     for $R_i \in R$
4:         $R_i$. Time = 0;
5:         $R_i$. Report = 0;
6:         $R_i$. Trade = 0;
7:         $R_i$. Credit = 0;
8:         Value = Report + Trade + Credit − Time;
9:     end for
10:    set S = {$S_i$ | $0 \le i \le |S| - 1$}
11:    for $S_i \in S$
12:        $S_i$. Time = 0;
13:        $S_i$. Report = 0;
14:        $S_i$. Trade = 0;
15:        $S_i$. Credit = 0;
16:        Value = Report + Trade + Credit − Time;
17:    end for
18:    set C = 0;
19:    set times = 0;
20: end
Output: R, S, Value, V, times;
```

Definition 1. *Finding a random number time. Each node searches for the random number based on the SHA256 hash function [24], and the time of searching for the random number is positively correlated with the computing power of the node. The higher the computing power of the node, the faster it searches for the random number. When a node finds a random number, it can broadcast it to the whole network. In order to reduce the difficulty and improve the efficiency, it takes the last four digits of the hash value of the previous block as the next random number and takes the time of finding the random number as an important indicator of the node evaluation mechanism, Time is the ratio of the time it takes for a node to find a random number to the time it takes for the last node to find a random number.*

Definition 2. *Broadcast degree and transaction degree of a node. After a node completes a transaction, it will broadcast to the whole network. After completing a consensus round, it will record the number of node broadcasts, the number of transactions involved, the total number of broadcasts and the total number of transactions in this consensus. In the consensus process, the broadcast degree of a node is represented by the ratio of the number of broadcasts that the node participates into the total number of broadcasts, which is denoted as Report. Similarly, the transaction degree of a node is represented by the ratio of the number of transactions involved in the node to the total number of transactions, which is denoted as Trade.*

Definition 3. *Node trust score. In this paper, the regression model logistic is used as the calculation method of trust score. The historical behavior of nodes can be used as the judgment index, the voting reputation value and accounting reputation value can be used as parameters, and the dynamic calculation of the trust value (Credit) of nodes can be performed according to the logistic regression function. The formula is as follows.*

$$\mathrm{Credit} = \frac{1}{1 + e^{-(\mathrm{Credit}_{\mathrm{vote}} + \mathrm{Credit}_{\mathrm{account}})}}$$

where $Credit_{vote}$ and $Credit_{account}$ denote the voting reputation function and accounting reputation function, respectively, thus defining the voting reputation value and accounting reputation value.

Definition 4. *The voting reputation function. In the Byzantine fault tolerance algorithm (PBFT), the number of Byzantine nodes cannot exceed 1/3 of the total number of nodes, and consensus can be successful only when the number of normal nodes is greater than 2/3. In order to make as many nodes as possible participate in the voting, the more nodes participate in the voting, the more nodes are set to increase the reputation value of nodes, and the voting reputation function is as follows.*

$$\mathrm{Credit}_{\mathrm{vote}} = \begin{cases} \sum_{i=1}^{k} X_i - \sum_{i=1}^{n} Y_i & , m \geq \frac{2}{3}n \\ (n - m)/n & , m < \frac{2}{3}n \end{cases}$$

In the voting reputation function, suppose the number of voting nodes is m, the total number of nodes is n, the consensus round is k, and the number of nodes voting i round is X_i; before electing the accounting node, multiple votes can be conducted in the group and the number of nodes not voting in the i round is Y_i. The voting flow chart of the SP-PBFT algorithm is shown in Figure 3.

Definition 5. *Accounting reputation function: In order to avoid malicious nodes from participating in the consensus, the bookkeeping reputation function is defined, and the historical behavior of whether the bookkeeping is successful is recorded in the next round of the bookkeeping reputation value as the evaluation index of the bookkeeping reputation function, the bookkeeping reputation function is as follows.*

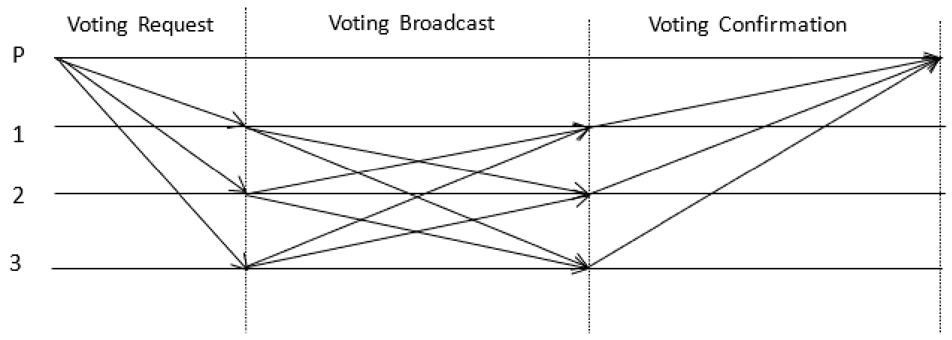

Figure 3. Voting Process Diagram of SP-PBFT Algorithm.

$$\mathrm{Credit}_{\mathrm{account}} = \begin{cases} 1 & \text{Non accounting nodes} \\ \mathrm{M} \;/\; \mathrm{T}_{\mathrm{i}-1,\mathrm{i}} & \text{Accounting node} \end{cases}$$

In the accounting reputation function, only one block is generated in each round of consensus, so the accounting reputation value of the non-accounting node is set to be 1; if the node is successful in accounting, the M value is 1; if the accounting node is not successful in accounting, the m value is 0, and the block time in the *i*th round of consensus is $\mathrm{T}_{\mathrm{i}-1}$ (min).

Definition 6. *Node evaluation mechanism: Nodes can find the random number through the SHA256 hash function, broadcast and record, and calculate the node reputation value (Value) through the four coefficients of Time, Report, Trade and Credit, the formula for Value, as follows.*

$$\mathrm{Value} = \mathrm{Report} + \mathrm{Trade} + \mathrm{Credit} - \mathrm{Time}$$

Compared to the traditional PBFT algorithm, the reward and punishment mechanism of this algorithm can reduce the likelihood of malicious nodes becoming master nodes. The specific reasons are as follows: firstly, due to the monitoring and evaluation of node behavior by this algorithm, the reputation value of malicious nodes will decrease, ranking behind consensus nodes and master nodes, reducing the likelihood of becoming the master node. Secondly, the algorithm adopts Fibonacci function rules for node grouping, with the first half of nodes in each group forming consensus nodes, thereby reducing the possibility of malicious nodes becoming consensus nodes. Because malicious nodes have a lower reputation value and are ranked behind consensus nodes, which are composed of nodes with higher reputation values, the probability of malicious nodes becoming consensus nodes is relatively low. Finally, the algorithm adopts a voting mechanism to select the main node, and consensus nodes vote on candidate main nodes, selecting the node with the highest number of votes as the main node. Due to the low reputation value of malicious nodes, other consensus nodes are less likely to vote for them, making it less likely that malicious nodes will become the primary node. In summary, the reputation reward and punishment mechanism introduced by this algorithm can reduce the possibility of malicious nodes becoming the main node and make it more difficult for malicious nodes to become the main node. Each node saves the block generated by the reputation value of the node evaluation mechanism to the end of the blockchain, and each node broadcasts the result of the node evaluation mechanism.

The pseudo-code for selecting consensus nodes based on their reputation values is as follows (Algorithm 2):

Algorithm 2 Consensus Node Selection.

```
Input: Time, Report, Trade, Credit, C, times, R, S;
1: Begin
2:    for $R_i \in R$
3:        Value = Report + Trade + Credit − Time;
4:    end for
5:    if ($R_i$. Value < C) | | ($R_i$ == S&&$R_i \notin R$)
      //Determine whether the reputation value of $R_i$ node exceeds the consensus node reputation value
threshold
6:        set temp = $R_i$;
7:        set $R_i$ = S. max;
8:        S. max = temp;
9:        times = times + 1;
10:       for S. max ∈ S
      //Place the eliminated consensus nodes in the candidate node set and sort them by reputation value
11:           if(S[i] < S[i − 1])
12:               S [0] = S[i];
13:               for (j = i − 1; S [0] < S[j]; --j)
14:                   S[j + 1] = S[j];
15:                   S[j + 1] = S [0];
16:               end for
17:           end if
18:       end for
19:   end if
20: end
Output: R, S, Value, times;
```

The flow chart of the SP-PBFT algorithm is shown in Figure 4.

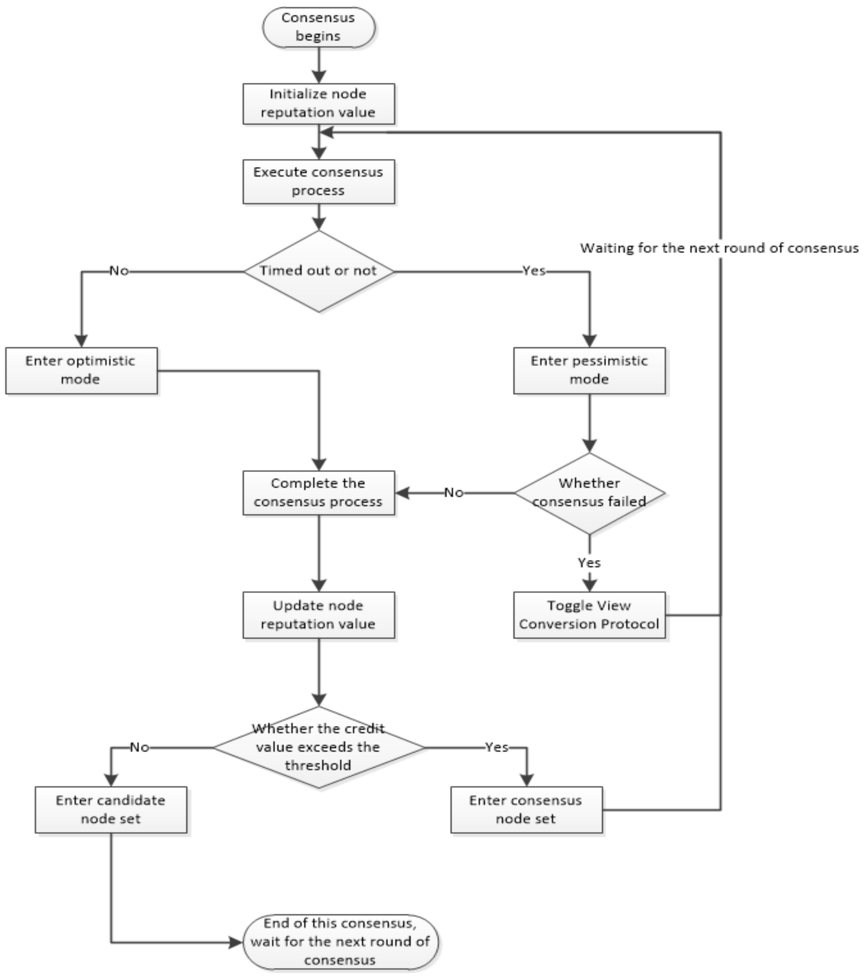
Consensus begins
Initialize node reputation value
Execute consensus process
Timed out or not
No
Yes
Waiting for the next round of consensus
Enter optimistic mode
Enter pessimistic mode
Whether consensus failed
No
Yes
Complete the consensus process
Toggle View Conversion Protocol
Update node reputation value
Whether the credit value exceeds the threshold
No
Yes
Enter candidate node set
Enter consensus node set
End of this consensus, wait for the next round of consensus

Figure 4. Flowchart of SP-PBFT algorithm.

2.4. Fibonacci Grouping

Among various methods of grouping in multiple mathematical models, this paper has chosen the Fibonacci function as the grouping function for its advantages in the following aspects:

(1) Enthusiasm: The Fibonacci function ensures that every node has the opportunity to become a consensus node, rather than only nodes with high reputation values having the opportunity to participate in consensus. This can improve the enthusiasm of nodes to become consensus nodes.

(2) Mathematical proof: The properties of the Fibonacci sequence may make the analysis and proof of the algorithm simpler, allowing for fast computation and transmission between nodes, thereby reducing communication complexity and improving consensus efficiency.

(3) Performance balance: Fibonacci sequences have certain mathematical characteristics that can provide appropriate thresholds while pursuing performance balance. In consensus algorithms, performance balancing is an important consideration to ensure high throughput and low latency.

(4) Threshold adjustment: The Fibonacci sequence can be used to dynamically adjust the number of nodes participating in consensus. This is beneficial for adapting to different network conditions, node failure rates, and security requirements.

(5) Simplicity: The Fibonacci sequence can simplify algorithm implementation to a certain extent, as it provides a simple way to adjust the number of voting rounds and confirmation stage.

The Fibonacci function may be a useful tool in some cases; therefore, it is considered a more suitable function for node grouping.

2.4.1. Consensus Node Selection

The consensus algorithm proposed in this paper uses the Fibonacci function to group nodes. The specific grouping process is as follows: First, all nodes are arranged in descending order according to their reputation values. Then, according to the Fibonacci function rules, the nodes are grouped into several groups, with the number of nodes in each group being a certain number in the Fibonacci sequence. Specifically, assuming there are n nodes in total, and the kth number in the Fibonacci sequence is F(k), the nodes will be divided into F(k) groups, with the number of nodes in each group being F(k − 1). The expression for the Fibonacci function is as follows:

$$\begin{cases} \mathrm{F}(1) = 1 \\ \mathrm{F}(2) = 1 \\ \mathrm{F}(\mathrm{n}) = \mathrm{F}(\mathrm{n}-1) + \mathrm{F}(\mathrm{n}-2)\ (\mathrm{n} \geq 2, \mathrm{n} \in \mathrm{N}^*) \end{cases}$$

Finally, use the rounding-up function to determine the consensus node composed of the first half of the nodes in each group. Due to the small difference in reputation values among nodes in each group and the same probability of nodes being selected as consensus nodes in groups with low reputation values as those in groups with high reputation values, this solves the problem of node reputation accumulation in reputation mechanism-based consensus algorithms, reduces the degree of system centralization, and enhances the enthusiasm of nodes to participate in consensus node selection. Additionally, it reduces the difference in node reputation values, preventing the most reputable nodes from overworking. In summary, the consensus algorithm proposed in this article uses the Fibonacci function for node grouping, which can improve the security and efficiency of the consensus algorithm. The Fibonacci node structure diagram is shown in Figure 5.

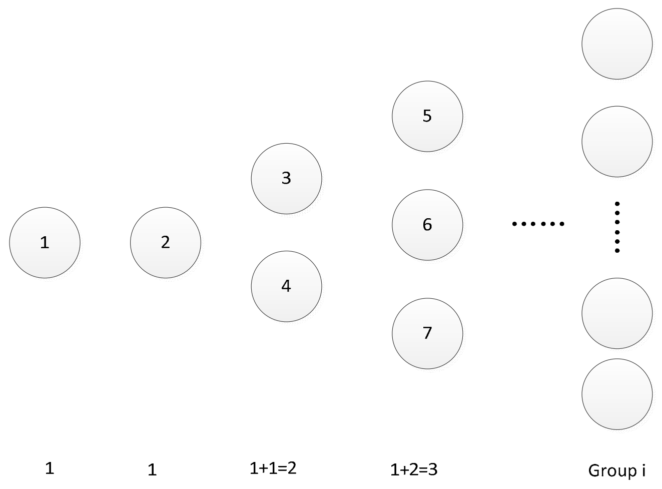

Figure 5. Node Fibonacci group structure diagram.

2.4.2. Master Node Selection

View switching refers to the scenario where, if the master node turns out to be a Byzantine node and fails to dispatch a consensus request message or times out within the specified duration, the remaining slave nodes take the initiative to initiate a request for view replacement. This process entails displacing the current master node and re-selecting a trusted master node for participation. In case the previously designated master node is identified as being overthrown, the new view is once again switched, leading to the discarding of blocks that had achieved consensus in the preceding phase. Once the new master node is elected, the consensus process is re-established for the previously discarded blocks. Commencing a new view involves synchronizing and promptly inspecting global nodes to guarantee data integrity and consistency. Subsequently, the system awaits the forthcoming round of consensus. In the traditional PBFT algorithm, the determination of the view-switching master node adheres to the principles laid out in Equation (1).

$$\mathrm{p} = \mathrm{v} \bmod |\mathrm{R}| \tag{1}$$

In this formula, p indicates the master node number for new elections, v indicates the view number, |R| indicates the number of the nodes in copy right now that use polling to switch the traditional view to be consistent with the agreement. Here, there is a risk that a large random probability may turn malicious nodes into a master node. In order to solve this problem, this paper uses the Fibonacci function to group the nodes, and the nodes in each group vote with each other, in which the weight of the vote is positively correlated with the Value value. The master node starts from the first group of nodes, and the node generated by the second group of voting will be selected if the selection fails, and so on. Because the Fibonacci grouping is implemented according to the order of reputation value from high to low, the node with high reputation value is in the front of the group and has the priority to serve as the master node, which improves the security and stability of the system. This process can quickly select the master node to prevent the slow selection of the master node from causing system delay and stagnation.

3. Analysis of Theoretical and Experimental Results

This paper validates the proposed SP-PBFT algorithm through two aspects: theoretical analysis and comparative experimental implementation. The theoretical analysis compares the algorithm's security, consensus efficiency, and feasibility. The experimental implementation compares SP-PBFT with other improved PBFT algorithms in terms of throughput, latency, Byzantine fault tolerance, and communication complexity, among other experimental results.

3.1. Theoretical Analysis

This paper accomplishes theoretical analysis by conducting security analysis, consensus efficiency analysis, and exploring the feasibility of enhancing consensus efficiency by reducing the number of consensus nodes.

3.1.1. Security Analysis

In the traditional PBFT consensus algorithm, the selection of the master node is carried out through polling, with each node serving as the master node in order of numbering. Due to the possibility of Byzantine nodes sending incorrect messages, if a Byzantine node is selected as the primary node, it may lead to errors in the consensus process. If there are f Byzantine nodes in the consensus node, then in the PBFT algorithm, if the selection of the main node is random, there is a $1/(f + 1)$ probability that the Byzantine node will be selected as the main node. Therefore, if there are Byzantine nodes in the consensus nodes, traditional PBFT algorithms pose certain security risks in selecting the main node, and even in subsequent master node selections, the aforementioned method does not inherently reduce risks. To tackle this challenge, the SP-PBFT algorithm, introduced within this paper, embraces a novel approach. It incorporates a node evaluation mechanism that calculates the reputation value for each node based on its historical performance. Subsequently, the algorithm organizes nodes into sorted groups contingent on their reputation values. This dynamic grouping amplifies the probability of nodes with higher reputation values assuming the role of master nodes. It is important to note that the SP-PBFT algorithm diverges from conventional enhanced algorithms, which primarily introduce reward and punishment mechanisms. Following the Fibonacci grouping, the SP-PBFT algorithm selectively designates only half of the nodes with elevated reputation values within each group. Notably, this selection is not purely sequenced by reputation value. This strategic decision serves to thwart potential malicious nodes from targeting nodes with high reputation values, a scenario that could trigger consensus failures. As a result, this approach bolsters the security of the consensus process while promoting its overall robustness.

3.1.2. Consensus Efficiency Analysis

The traditional PBFT algorithm has a three-stage communication process and requires all nodes to participate in the consensus process. The nodes interact with each other in pairs, which makes the consensus efficiency low and consumes a lot of resources. The proposed SP-PBFT algorithm improves the consensus efficiency from the following three aspects:

(1) The consensus protocol is optimized by using the idea of speculation. When the system has no delay, only the optimistic mode is executed to reduce the communication complexity.
(2) Secondly, a node evaluation mechanism is proposed to calculate the reputation value according to the historical behavior of the node so that the consensus nodes are well-behaved nodes and the probability of view switching is reduced.
(3) The Fibonacci group consensus is proposed, and only half of the nodes in each group are selected to participate in the consensus, which reduces the number of nodes participating in the consensus and fundamentally improves the efficiency of consensus.

3.1.3. Feasibility Analysis

Reducing the number of nodes participating in consensus is a possible optimization direction when optimizing the PBFT consensus algorithm. The advantages are as follows.

(1) Reduce latency: The performance of PBFT is affected by the latency of network communication and message delivery. Reducing the number of nodes participating in consensus can reduce the load on message delivery and communication, thereby reducing the overall consensus latency.

(2) Reduce computing costs: Consensus algorithms require participating nodes to perform calculations, signatures, and other operations. Reducing the number of nodes can reduce computational costs and resource consumption.
(3) Simplify coordination: A smaller number of nodes may simplify the coordination process of consensus protocols and reduce complexity.

However, there are still considerations to consider:

(1) Security considerations: The PBFT algorithm is a fault-tolerant consensus algorithm that can tolerate a certain number of Byzantine errors (malicious behavior). Reducing the number of nodes may reduce the system's ability to resist attacks. However, this article not only has a pessimistic mode but also proposes a node evaluation mechanism, which scores based on the historical behavior of nodes. The malicious nodes detected using this method cannot successfully participate in consensus and can only react to the consensus process. Therefore, reducing the number of nodes can ensure that the system is still safe enough to tolerate a certain number of malicious nodes.
(2) Distributed: One of the advantages of PBFT is its distributed nature, which can achieve consensus in a decentralized environment. Reducing the number of nodes may lead to centralization risk and reduce the distributed nature of the system. However, this article proposes an improved PBFT consensus algorithm based on the node-grouping reputation model. By arranging all nodes in descending order of reputation values and grouping them according to the Fibonacci function, consensus nodes are selected within the group, thus solving the problem of node reputation accumulation in reputation-mechanism-based consensus algorithms. Each participating node is not the same and will be evaluated based on historical behavior. This reduces the degree of system centralization and enhances the enthusiasm of nodes to become consensus nodes.

3.2. Analysis of Experimental Results

This experiment uses Docker container simulation multiple nodes to compare the traditional PBFT algorithm and its improved algorithm and analyzes the advantages and disadvantages of the improved algorithm. The experimental environment is the Ubuntu 18.04 operating system with 16 GB of system memory; the CPU is an Intel Core i7 processor with a Docker container engine. This experiment uses Docker containers for simulation experiments. Firstly, install the Docker container and create a Dockerfile to define the image of PBFT nodes. Specify the required operating system, software dependencies, and node configuration. Use the constructed image to run a container of 100 nodes, assign different ports to each container so that they can communicate with each other, and configure node parameters in each container, such as node ID, network address, etc. These parameters will affect communication and consensus between nodes. According to the PBFT consensus algorithm, if there are f Byzantine nodes, the total number of nodes is not less than 3f + 1. Using the four aspects of throughput, delay, fault tolerance and communication complexity, the PBFT algorithm, PBFT consensus algorithm based on the Trust evaluation model in the literature [17] (trust-based Practical Byzantine Fault Tolerance, T-PBFT), an improved PBFT algorithm [14] (Random Practical Byzantine Fault Tolerance, RPBFT) based on the random selection of master nodes, as referenced, and the SP-PBFT algorithm of this paper are compared. The performance of the four algorithms is analyzed using the experimental results.

3.2.1. Throughput

Throughput is defined as the number of transactions that the system processes per request in a unit of time. The higher the throughput, the greater the number of transactions that will be processed. Set the four consensus algorithms PBFT, T-PBFT, RPBFT, and SP-PBFT to complete the same number of transactions in 10, 20, 30, 40, 50, 60, 70, 80, 90 and 100 nodes, respectively, and the number of transactions completed per second will be calculated. TPS (Transaction Per Second) represents the system throughput, and the calculation formula is given in (2).

$$\text{TPS} = \text{Transactions}/\Delta\text{t} \quad (2)$$

where Δt represents the time from the transaction information being issued to the completion of block storage, namely, the block generation time, and Transactions represents the number of transactions completed in each block within the time interval of block generation. The throughput of the system is also related to the network environment and block size. The larger the block is, the more transactions it can carry, and the corresponding network load will increase. The number of transactions in one second with different numbers of nodes is counted and all of the results are averaged. The final result is shown in Figure 6 below.

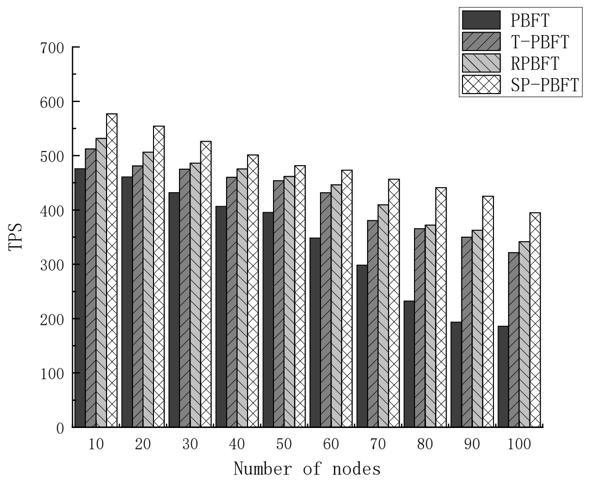

Figure 6. Comparison of throughput between SP-PBFT and other algorithms.

Through an analysis of the experimental results and a comparative study of the four algorithms, it becomes evident that the throughput of all four algorithms diminishes as the number of nodes increases. However, notably, the SP-PBFT algorithm outperforms the other three algorithms in terms of throughput. The conventional PBFT algorithm's reliance on universal node participation in the consensus process leads to a decline in throughput. The RPBFT algorithm, while introducing alterations to the consensus process and protocol optimization, demonstrates only a relative enhancement. Although the T-PBFT algorithm implements consensus grouping, it fails to reduce the number of participating nodes in the consensus process, resulting in a suboptimal efficiency for the optimized consensus protocol, which contrasts with the superior efficiency observed in the SP-PBFT algorithm. Upon rigorous testing, the enhanced SP-PBFT algorithm showcases a substantial enhancement in throughput.

3.2.2. Time Delay Analysis

Consensus latency refers to the time from the submission of a request by a client to the completion of confirmation, which is the time required for communication in the node network from the submission of a request to the completion of confirmation. The throughput of the blockchain is mainly affected by the delay. Under the premise of consistent other aspects, the smaller the delay, the higher the throughput of the blockchain system. In addition, lower delay can also improve the speed of blockchain consensus confirmation and the efficiency of the system consensus. Set PBFT, T-PBFT, RPBFT and SP-PBFT four consensus algorithms to complete the same number of transactions in 10, 20, 30, 40, 50, 60, 70, 80, 90 and 100 nodes, respectively, and calculate the average delay of the transaction. In this paper, the delay is defined as the time it takes for a block to complete the consensus, as shown in Equation (3).

$$\text{DelayTime} = \text{Time}_{\text{reply}} - \text{Time}_{\text{req}} \tag{3}$$

$\text{Time}_{\text{reply}}$ represents the time of consensus confirmation completion; Time_{req} represents the time of the consensus request and takes the average value of multiple tests using the Caliper tool. The results are shown in Figure 7 below.

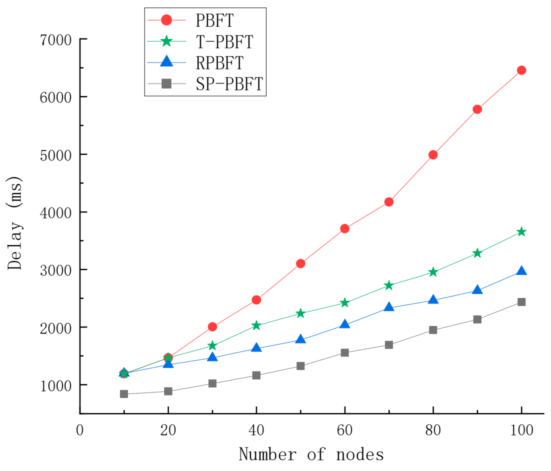

Figure 7. Comparison of latency between SP-PBFT and other algorithms.

Through an analysis of the experimental results and a comparative assessment of the four algorithms, it becomes evident that as the number of participating nodes increases, the delay time also lengthens. Initially, with a small number of nodes, the distinctions among the four algorithms are relatively minor. However, as the number of nodes rises, significant discrepancies become apparent. The conventional PBFT algorithm mandates universal node participation in consensus, resulting in prolonged delays. In contrast, the RPBFT algorithm optimizes the consensus protocol, diminishing communication complexity and consequently reducing delay times. Although the T-PBFT consensus grouping approach does not curtail the number of consensus nodes and maintains a substantial number of broadcasts, the improvement in delay time is only marginal. The SP-PBFT algorithm demonstrates notable advancements in both the consensus protocol and the number of consensus nodes. Following comprehensive testing, the enhanced SP-PBFT algorithm exhibits lower delay times in comparison to the other three algorithms. This superior delay performance enhances the algorithm's stability and boosts consensus efficiency.

3.2.3. Fault Tolerance

Byzantine fault tolerance is a fault tolerance ability of the distributed network; that is, in the case of malicious nodes, honest nodes can still reach consensus. Byzantine nodes are added to the experiment to test the fault tolerance of the consensus algorithm. The proportion of Byzantine nodes is set to 5%, 10%, 15%, 20%, 25%, 30%, and the total number of nodes is 100. Comparing the delay and throughput of PBFT algorithm, T-PBFT algorithm, RPBFT algorithm and SP-PBFT algorithm with different number of Byzantine nodes to complete the same number of transactions, the results are shown in Figures 8 and 9 below.

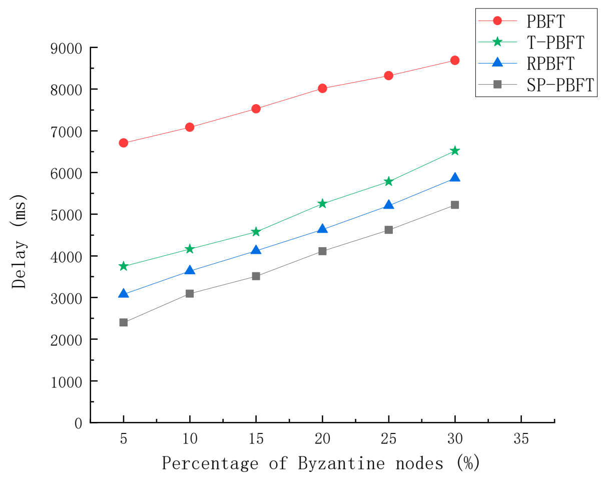

Figure 8. Comparison of latency between four algorithms under different numbers of Byzantine nodes.

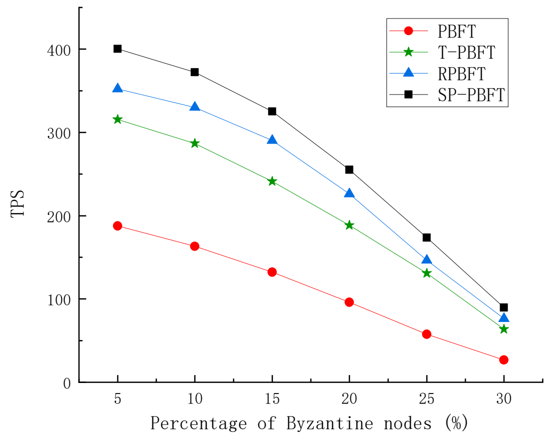

Figure 9. Comparison of throughput between four algorithms under different numbers of Byzantine nodes.

The experimental results show that the four algorithms have certain fault tolerance. When the number of Byzantine nodes is 33, it does not exceed 1/3 of the number of nodes, and blocks can be normally generated. But when the number of Byzantine nodes is 34, it exceeds 1/3 of the number of nodes, so the block cannot be successfully generated. It can be seen from Figure 8 that the delay of the four algorithms becomes longer with the increase in the number of Byzantine nodes, but the delay of the SP-PBFT algorithm is lower than that of the other three algorithms. It can be seen in Figure 9 that the throughput of the algorithm is lower when the number of Byzantine nodes is larger, but the throughput of the SP-PBFT algorithm is always higher than that of the other three algorithms, so the SP-PBFT algorithm is better.

3.2.4. Communication Complexity

The communication complexity of the consensus algorithm represents the interaction of each node in the system to complete the consensus process, which can also be called the number of broadcasts. The higher the communication complexity, the lower the efficiency of the consensus and the greater the consumption of resources. In the pre-preparation process of the traditional PBFT algorithm, the master node broadcasts the pre-preparation message to all nodes except itself, and the number of communications is $N-1$. In the preparation process, the received message was verified, the preparation message was sent, and the number of communications in this process was $(N-1)^2$. Entering the confirmation phase, the verification prepares the message and sends the confirmation message to all nodes, and the number of communications is N(N − 1). Combining the above three processes is the number of messages sent by the traditional PBFT algorithm, $N - 1 + (N - 1)^2 + N(N - 1) = 2N^2 - 2N$ messages are sent, and the communication complexity is $O(n^2)$.

In the SP-PBFT algorithm of this paper, the number of messages sent in the pre-preparation, preparation and acknowledgement phases in the optimistic mode is N − 1; that is, in the whole process of the optimistic mode, $N - 1 + N - 1 + N - 1 = 3N - 3$ messages are sent, and the communication complexity is O(n). In the pessimistic mode, the number of messages sent in the preparation phase 2 and the confirmation phase 2 in the optimistic mode are both $(N - 1)^2$, and a total of $3N - 3 + 2(N - 1)^2 = 2N^2 - N - 1$ messages are sent. The communication complexity is $O(n^2)$, which is consistent with the traditional PBFT algorithm, but after the screening of the node evaluation mechanism, most of them enter the optimistic mode to complete the consensus and no longer go through the consensus process of the pessimistic mode. It can be seen that the communication complexity of the improved SP-PBFT algorithm is reduced, the application scope of the algorithm is also wider, and the consensus efficiency is higher, which is optimized.

3.3. Comparison of Related Consensus Algorithms

Table 2 shows a comparison of traditional PBFT algorithms, T-PBFT algorithms, RPBFT algorithms and SP-PBFT algorithms.

Table 2. Comparison of related consensus algorithms.

Comparison of Consensus Algorithms	Speculation	Reduce the Number of Consensus Nodes	Communication Complexity
PBFT	×	×	$O(n^2)$
T-PBFT	×	×	$<O(n^2)$
RPBFT	×	×	O(n)
SP-PBFT	√	√	Optimistic mode O(n) or Pessimistic mode $O(n^2)$

It is evident that the SP-PBFT algorithm possesses distinct advantages over other consensus algorithms, as outlined below:

(1) The SP-PBFT algorithm introduces a node evaluation mechanism, which applies rewards or penalties based on historical node behavior. This mechanism serves to enhance the security of the consensus process.
(2) The SP-PBFT algorithm offers the capability to identify Byzantine nodes, consequently reducing the likelihood of these malicious nodes participating in the consensus process. This contributes to the system's robustness and optimization of the view-switching protocol, ultimately leading to enhanced consensus speed.
(3) The SP-PBFT algorithm reduces the communication overhead in the network, adopts the speculative method to optimize the consensus protocol, and the communication complexity required to reach consensus in the optimistic mode is reduced from $O(n^2)$ to O(n).
(4) The SP-PBFT algorithm introduces Fibonacci grouping, which, when combined with the reputation value within the node evaluation mechanism, facilitates the selection of master and consensus nodes. This strategic combination reduces the volume of nodes involved in the consensus process, leading to a fundamental enhancement in consensus efficiency.

4. Conclusions and Outlook

This paper introduces the SP-PBFT consensus algorithms that adopts a speculative approach based on the network environment of the system, employing distinct steps to achieve consensus under various network conditions. In an optimistic mode, when the network is secure and stable, the confirmation phase is streamlined, resulting in reduced communication overhead and quicker consensus attainment. Conversely, when the consensus timeout detects potential system insecurity, such as the presence of malicious node attacks, the pessimistic mode is activated. In this mode, an additional confirmation phase is introduced to counteract malicious node attacks. To enhance system resilience and security, a node evaluation mechanism is implemented to reward or penalize nodes based on their historical behavior. This mechanism enables the system to identify Byzantine nodes, ensuring security and optimizing the view-switching protocol. Simultaneously, the Fibonacci function is utilized for node grouping, and in conjunction with the reputation value from the node evaluation mechanism, appropriate master and consensus nodes are selected. This approach reduces the number of consensus nodes and fundamentally enhances consensus efficiency. Experimental results demonstrate that the SP-PBFT algorithm outperforms both the traditional PBFT algorithm and its improved versions in terms of throughput, latency, fault tolerance, and communication complexity. Particularly in scenarios involving honest master nodes and stable networks, the advantages of the SP-PBFT algorithm become even more pronounced. Therefore, the SP-PBFT algorithm is suitable for consortium chains with stable networks and more trusted nodes. This algorithm still has some shortcomings. For example, in the case of an unstable network and more malicious nodes, the consensus efficiency of the pessimistic mode needs to be further improved. Future work will focus on optimizing the pessimistic mode and discuss how to quickly reach a consensus in the face of many malicious nodes and maximize the consensus efficiency without affecting security and dynamism.

Author Contributions: Conceptualization, X.S. and X.L.; methodology, X.S.; software, X.L.; validation, X.S. and X.L.; formal analysis, X.S.; investigation, X.S.; resources, X.L.; data curation, X.L.; writing, original draft preparation, X.L.; writing, review and editing, X.L.; visualization, X.S.; supervision, X.S.; project administration, X.S.; funding acquisition, X.S. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Natural Science Foundation of China, grant number 62173171.

Data Availability Statement: The data used to support the findings of this study are included within the article.

Acknowledgments: The authors would like to thank the editors and the reviewers for their valuable time and constructive comments.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Etemadi, N.; Van Gelder, P.; Strozzi, F. An ISM Modeling of Barriers for Blockchain/Distributed Ledger Technology Adoption in Supply Chains towards Cybersecurity. *Sustainability* **2021**, *13*, 4672. [CrossRef]
2. Ali, O.; Ally, M.; Dwivedi, Y. The state of play of blockchain technology in the financial services sector: A systematic literature review. *Int. J. Inf. Manag.* **2020**, *54*, 102199. [CrossRef]
3. Balamurugan, S.; Ayyasamy, A.; Joseph, K.S. IoT-Blockchain driven traceability techniques for improved safety measures in food supply chain. *Int. J. Inf. Technol.* **2021**, *14*, 1087–1098. [CrossRef]
4. Feng, C.; Liu, B.; Yu, K.; Goudos, S.K.; Wan, S. Blockchain-Empowered Decentralized Horizontal Federated Learning for 5G-Enabled UAVs. *IEEE Trans. Ind. Inform.* **2022**, *18*, 3582–3592. [CrossRef]
5. Wang, T.; Hua, H.; Wei, Z.; Cao, J. Challenges of blockchain in new generation energy systems and future outlooks. *Int. J. Electr. Power Energy Syst.* **2022**, *135*, 107499. [CrossRef]
6. Leng, J.; Zhou, M.; Zhao, J.L.; Huang, Y.; Bian, Y. Blockchain Security: A Survey of Techniques and Research Directions. *IEEE Trans. Serv. Comput.* **2022**, *15*, 2490–2510. [CrossRef]
7. Xiong, H.; Chen, M.; Wu, C.; Zhao, Y.; Yi, W. Research on Progress of Blockchain Consensus Algorithm: A Review on Recent Progress of Blockchain Consensus Algorithms. *Future Internet* **2022**, *14*, 47. [CrossRef]
8. Suripeddi, M.K.S.; Purandare, P. Blockchain and GDPR—A Study on Compatibility Issues of the Distributed Ledger Technology with GDPR Data Processing. *J. Phys. Conf. Ser.* **2021**, *1964*, 042005. [CrossRef]
9. Li, Y.; Yang, G.; Susilo, W.; Yu, Y.; Au, M.H.; Liu, D. Traceable Monero: Anonymous Cryptocurrency with Enhanced Accountability. *IEEE Trans. Dependable Secur. Comput.* **2021**, *18*, 679–691. [CrossRef]

10. Kim, H. Adjusting the Block Interval in PoW Consensus by Block Interval Process Improvement. *Electronics* **2021**, *10*, 2135. [CrossRef]
11. Saad, M.; Qin, Z.; Ren, K.; Nyang, D.; Mohaisen, D. e-PoS: Making Proof-of-Stake Decentralized and Fair. *IEEE Trans. Parallel Distrib. Syst.* **2021**, *32*, 1961–1973. [CrossRef]
12. Xu, H. Consensus protocol based on DPOS and aggregate signature. In Proceedings of the 2022 3rd International Conference on Computer Vision, Image and Deep Learning & International Conference on Computer Engineering and Applications (CVIDL & ICCEA), Changchun, China, 20–22 May 2022; pp. 1028–1035.
13. Yang, J.; Jia, Z.; Su, R.; Wu, X.; Qin, J. Improved Fault-Tolerant Consensus Based on the PBFT Algorithm. *IEEE Access* **2022**, *10*, 30274–30283. [CrossRef]
14. Wang, S.; Li, Z.; Jia, Z. Improved PBFT consensus algorithm for randomly selecting main nodes. *Comput. Appl. Softw.* **2022**, *39*, 299–306.
15. Fang, Y.; Deng, J.Q.; Cong, L.H.; Liu, C.Y. An improved PBFT blockchain consensus algorithm based on ring signature. *Comput. Eng.* **2019**, *45*, 32–36.
16. Zhang, M.; Wang, B. Improving Byzantine fault tolerance consensus algorithm based on recommended trust model. *Comput. Appl. Res.* **2023**, *40*, 667–670.
17. Ren, X.; Tong, X.; Zhang, W. PBFT consensus algorithm based on trust evaluation model. *J. Shanxi Univ. (Nat. Sci. Ed.)* **2023**, *46*, 108–118.
18. Qin, W. PBFT Consensus Algorithm Based on Node Partition Clustering. *Inf. Technol. Informatiz.* **2023**, *274*, 65–69.
19. Caldarola, F.; d'Atri, G.; Zanardo, E. Neural Fairness Blockchain Protocol Using an Elliptic Curves Lottery. *Mathematics* **2022**, *10*, 3040. [CrossRef]
20. Patel, D.; Sanghvi, H.; Jadav, N.; Gupta, R.; Tanwar, S.; Florea, B.; Taralunga, D.; Altameem, A.; Altameem, T.; Sharma, R. BlockCrime: Blockchain and Deep Learning-Based Collaborative Intelligence Framework to Detect Malicious Activities for Public Safety. *Mathematics* **2022**, *10*, 3195. [CrossRef]
21. Apostu, S.; Panait, M.; Vasa, L.; Mihaescu, C.; Dobrowolski, Z. NFTs and Cryptocurrencies—The Metamorphosis of the Economy under the Sign of Blockchain: A Time Series Approach. *Mathematics* **2022**, *10*, 3218. [CrossRef]
22. Leng, J.; Zhou, M.; Zhao, J.L.; Huang, Y.; Bian, Y. Progress in Blockchain BFT Consensus Algorithm Research. *Comput. Sci.* **2022**, *49*, 329–339.
23. Xiao, B.; Li, Z.; Li, X. Consensus algorithm based on the importance of Fibonacci grouping. *Comput. Digit. Eng.* **2021**, *49*, 2509–2513+2525.
24. APPEL, A.W. Verification of a Cryptographic Primitive: SHA-256. *Proc. Soc. Inf. Biliol.* **2015**, *37*, 1–31. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.