electronics

MDPI

Article

An Optimized Byzantine Fault Tolerance Algorithm for Medical Data Security

Gang Xu [1,3], Tengkai Yao [1], Kejia Zhang [2,4,*], Xiangfei Meng [1], Xin Liu [5], Ke Xiao [1] and Xiubo Chen [6]

[1] School of Information Science and Technology, North China University of Technology, Beijing 100144, China; gx@ncut.edu.cn (G.X.); yaotengkai@163.com (T.Y.); mengxiangfei_ncut@163.com (X.M.); xiaoke@ncut.edu.cn (K.X.)
[2] School of Mathematical Science, Heilongjiang University, Harbin 150080, China
[3] Yunnan Key Laboratory of Blockchain Application Technology, Kunming 650233, China
[4] State Key Laboratory of Public Big Data, Guizhou University, Guiyang 550000, China
[5] School of Information Engineering, Inner Mongolia University of Science and Technology, Baotou 014010, China; lx2001lx@imust.edu.cn
[6] Information Security Center, State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing 100876, China; xb_chen@bupt.edu.cn
* Correspondence: zhangkejia@hlju.edu.cn

Abstract: Medical data are an intangible asset and an important resource for the entire society. The mining and application of medical data can generate enormous value. Currently, medical data management is mostly centralized and heavily relies on central servers, which are prone to malfunctions or malicious attacks, making it difficult to form a consensus among multiple parties and achieve secure sharing. Blockchain technology offers a solution to enhance medical data security. However, in medical data security sharing schemes based on blockchain, the widely adopted Practical Byzantine Fault-Tolerant (PBFT) algorithm encounters challenges, including intricate communication, limited scalability, and the inability to dynamically add or remove nodes. These issues make it challenging to address practical requirements effectively. In this paper, we implement an efficient and scalable consensus algorithm based on the PBFT consensus algorithm, referred to as Me-PBFT, which is more suitable for the field of medical data security. First, we design a reputation evaluation model to select more trusted nodes to participate in the system consensus, which is implemented based on a sigmoid function with adjustable difficulty. Second, we implement the division of node roles to construct a dual consensus layer structure. Finally, we design a node dynamic join and exit mechanism on the overall framework of the algorithm. Analysis shows that compared to PBFT and RAFT, ME-PBFT can reduce communication complexity, improve fault tolerance, and have good scalability. It can meet the need for consensus and secure sharing of medical data among multiple parties.

Keywords: blockchain; consensus algorithm; PBFT; medical

Citation: Xu, G.; Yao, T.; Zhang, K.; Meng, X.; Liu, X.; Xiao, K.; Chen, X. An Optimized Byzantine Fault Tolerance Algorithm for Medical Data Security. *Electronics* **2023**, *12*, 5045. https://doi.org/10.3390/electronics12245045

Academic Editor: Alberto Fernandez Hilario

Received: 28 September 2023
Revised: 25 November 2023
Accepted: 29 November 2023
Published: 18 December 2023

1. Introduction

The Bitcoin System [1], as described by Satoshi Nakamoto in 2008, utilizes blockchain as its foundational technology. Blockchain is a distributed database system that is decentralized, cannot be tampered with, and allows for traceability [2]. Blockchain technology integrates a variety of computer technologies, such as P2P network protocol [3], blockchain structure, consensus algorithm, asymmetric encryption [4], smart contracts [5], etc. Blockchain technology bridges secure communication between parties that cannot be trusted with each other. In terms of blockchain deployment, it can be categorized into two types: public chain and permissioned chain. The permissioned chain further consists of a consortium chain and a private chain [6,7]. The public chain is the most decentralized and has no special restrictions on the operation of nodes accessing the network, such

as Bitcoin [1], Ethereum [8], etc. The consortium blockchain will authenticate the nodes participating in the network, and only the nodes that meet the requirements can access the network, such as Hyperledger [9]. The consortium blockchain can be regarded as a multicenter system, while the private blockchain is controlled by a node. Due to these transparent, reliable, and secure characteristics, blockchain technology can provide new solutions for medical data security [10].

The consensus algorithm serves as the fundamental mechanism of the blockchain system. The consensus algorithm is utilized to guarantee the uniformity of data across every node within the blockchain, promote the consensus of medical data among medical institutions, and promote the secure sharing of data [11,12]. Currently, numerous consensus algorithms exist, including RAFT [13], Proof-of-Work (PoW) [1], Proof-of-Stake (PoS) [14], Delegated Proof-of-Stake (DPoS) [15], and Practical Byzantine Fault Tolerance (PBFT) [16], among others. RAFT is primarily employed in private blockchain networks. It does not take into account the presence of malicious nodes within the system, thus it fails to address the issue of byzantine fault tolerance. The PoW consensus algorithm has strong stability, high security, and strong fault tolerance, but nodes compete for the master node through computing resources, resulting in a large amount of power consumption. The PoS consensus algorithm uses the coin age to compete, which reduces the waste of resources, but may cause the problem of decentralization reduction due to the coin age concentration. The DPoS consensus algorithm elects representative nodes to participate in data packaging, which reduces the confirmation time and improves the consensus efficiency. However, DPoS still has the risk of centralization, and DPoS introduces 21 super nodes, which increases the starting cost of the medical data management system. The PBFT consensus algorithm does not need resource competition, does not have an equity mortgage, has a large data processing capacity, and has a low starting cost. It only needs more than four nodes to start, which is more suitable as the consensus algorithm of a medical data security system. However, the PBFT algorithm also has many shortcomings. (1) When the number of nodes increases, the number of messages passing through the PBFT algorithm increases dramatically, reducing the efficiency of system consensus. (2) The primary node is selected randomly. If the primary node selected continuously is malicious, it will greatly affect the security of the system. (3) It cannot join or exit nodes dynamically.

The emergence of blockchain technology has opened up possibilities for the secure sharing of medical data, offering a means to prevent privacy breaches and data tampering. In existing blockchain systems designed for securing medical data, Azaria et al. [17] implemented the MedRec platform using the Ethereum blockchain. MedRec utilized PoW as its consensus algorithm, requiring token incentives and resulting in a wasteful consumption of computational resources. Teng-Fei et al. [18] proposed the Medical Data Sharing Mechanism (MDSM), combining Medical Institution Federation Servers (MIFS) and Audit Federation Servers (AFS). MDSM adopted an improved DPOS consensus mechanism, necessitating a committee of at least 21 nodes for the audit federation server group, thereby raising the system's initial costs. ModelChain [19] integrated online machine learning with blockchain to protect medical data and implement predictive modeling. ModelChain utilized the Proof-of-Information (POI) algorithm as its consensus mechanism, combining machine learning with Proof-of-Work, requiring substantial computational support. Qu [20] employed a practical Byzantine fault-tolerant algorithm to construct a consortium medical blockchain with lower initiation costs and reduced computational consumption. However, it did not optimize the performance of the Practical Byzantine Fault Tolerance (PBFT) algorithm, making it challenging to meet the demands of large-scale dynamic sharing of medical data. Designing an efficient, reliable, and well-matched consensus algorithm to facilitate the secure sharing of medical data among multiple parties is a critical research focus.

This paper introduces a new optimized Byzantine Fault Tolerance consensus algorithm that is more suitable for the medical field, with higher efficiency and better scalability, referred to as ME-PBFT. The key contributions of this paper are as follows:

- We propose a reputation evaluation model that incorporates a sigmoid function to determine the reputation value of nodes. This approach allows us to adjust the difficulty level, thereby reducing the number of consensus nodes involved and enhancing consensus efficiency.
- We propose the double-consensus layer structure of the ME-PBFT algorithm, namely, the large consensus layer and the small consensus layer, and introduce the construction process of the consensus layer and the selection method of the primary node.
- Based on the ME-PBFT framework, we design the dynamic node joining and exiting mechanism to improve the efficiency of node joining and exiting and the flexibility and scalability of the system.

The rest of this paper is structured as follows: In Section 2, we present the related work. Section 3 provides an overview of ME-PBFT. Section 4 outlines the system framework of ME-PBFT. Section 5 explains the construction process, consensus process, and the mechanism for node dynamic joining and exiting in ME-PBFT. Section 6 presents an analysis and comparison. Finally, Section 7 concludes the paper.

2. Related Work

In blockchain consensus algorithms, two widely applied categories are Proof-of-X (POX) and Byzantine Fault Tolerance (BFT) [21] algorithms.

POX algorithms typically involve reaching a consensus based on the competition for certain resources. PoW achieves a consensus through computational competition, ensuring the security of the Bitcoin system. However, the intense computational competition in PoW results in a significant consumption of electrical resources. Ateniese et al. [22] introduced Proof-of-Space (PoSpace), which utilizes storage space as a competitive resource. The larger the data occupying the space, the higher the probability of obtaining the right to record transactions. Ball et al. [23] proposed Proof-of-Useful Work (PoUW), which leverages the hashrate in PoW to solve meaningful problems. To solve the problem of waste of power resources in PoW, PoS uses coin age as equity instead of computing power competition, which to some extent solves the problem of power resource consumption, but also faces the risk of reduced decentralization. DPoS is an extension of PoS that introduces 21 super nodes, enhancing the scalability of PoS while reducing decentralization.

The BFT consensus algorithm is one of the important solutions for consensus mechanisms in medical data security systems based on blockchain. In 1982, Leslie Lamport, Robert Shostak, and Marshall Pease formally introduced the Byzantine General problem [24] and gave solutions based on verbal messages and signed messages, which pioneered the research of BFT algorithms. In 1999, Castro and Liskov introduced the Practical Byzantine Fault-Tolerant algorithm, commonly known as PBFT. PBFT significantly lowers the complexity of the Byzantine protocol, shifting it from an exponential level to a polynomial one. This advancement enhances the practicality and feasibility of implementing Byzantine Fault-Tolerant algorithms in engineering applications. However, the PBFT algorithm still faces the problems of poor dynamics, poor scalability, and high communication complexity, which restrict the application of PBFT in practice. As a result, several researchers have introduced numerous enhancements and advancements to the BFT algorithm.

In terms of innovation in the consensus method of the BFT consensus algorithm, [25] proposed Zyzzyva to reduce communication complexity by adopting the main request and the client's immediate response. Miller et al. [26] proposed HoneyBadgerBFT, which was mainly used to solve the problem of reaching a consensus in an asynchronous network. Its process involves atomic broadcast and asynchronous common subsets, resulting in a high degree of communication complexity. Biryukov et al. [27] proposed a reputation model to select consensus groups according to the results of consensus rounds. Ref. [28] proposed Egalitarian Practical Byzantine Fault Tolerance (ePBFT), improved the selection strategy of a primary node in PBFT, optimized the consensus process of blockchain, improved the efficiency of data verification, and accelerated the consensus process. In the work by Gao et al. [29], they introduced T-PBFT, which employs a feature trust model for the

purpose of selecting high-caliber nodes within the network to establish consensus groups. This approach effectively trims down the count of consensus nodes and mitigates communication complexity. DBFT, as described in [30], introduces a dual-response mechanism and a self-conflict checking mechanism, effectively addressing issues associated with view change. Meanwhile, Wang et al. proposed vBFT [31], which employs a voting-based approach to categorize network nodes into three groups. This method enhances decentralization to a certain degree, and it offers the advantage of not requiring system restarts when the number of nodes changes, thus conserving system resources. Additionally, Li et al. [32] devised a hierarchical structure to enhance scalability and consensus efficiency. Zhan et al. [33] introduced the DRBFT consensus protocol, which is a Byzantine fault-tolerant protocol that utilizes a commissioned randomization approach. They also developed an RS random selection algorithm in conjunction with a voting mechanism. This algorithm helps to decrease the number of nodes involved in the consensus process, thereby enhancing the efficiency and reliability of the overall consensus program.

In terms of innovation in combination with specific application scenarios regarding the BFT consensus algorithm, Lao et al. [34] proposed G-PBFT, which is mainly aimed at the blockchain for the Internet of Things. It is a scalable consensus protocol based on location, reducing network overhead and improving consensus efficiency and scalability. Xu et al. [35] proposed SG-PBFT, suitable for the Internet of Vehicles, which used a score grouping mechanism to improve consensus efficiency. In reference [36], a consensus algorithm known as sc-PBFT (node-state-checkable Practical Byzantine Fault Tolerance) was introduced. This algorithm safeguards against the infiltration of Byzantine nodes into the alliance chain system by continually monitoring node statuses. Any malicious nodes detected are appropriately flagged and isolated within a designated area. Ref. [37] proposed a consensus algorithm combining PoS and PBFT in the Internet of Things environment, combined the initial equity and voting mechanism, and analyzed the impact of the number of nodes with different voting behaviors on the consensus probability.

It can be seen that the purpose of researchers is to design an efficient, scalable, and secure consensus mechanism.

3. Preliminaries

In this section, we will offer a summary of the core elements of our algorithm optimization, with a primary focus on PBFT and the reputation evaluation model that incorporates adjustable difficulty based on the sigmoid function.

3.1. Overview of PBFT

PBFT is a state machine replication algorithm that addresses the Byzantine general problem by ensuring consistency among nodes in the presence of malicious nodes. It divides all nodes into two categories: a primary node and multiple replica nodes, both of which participate in the consensus process. The primary node receives client requests and initiates voting. The replica node is responsible for verifying voting information and participating in the specific voting process, promoting a consensus among all nodes.

In PBFT, the primary node selection formula is as follows:

$$P = v \bmod N \tag{1}$$

P is the number of the primary node, v is the number of the view, and N is the total number of nodes.

The selection of the primary node in PBFT is based on a random selection method, resulting in an equal probability for each node to be elected as the primary node. This means that even Byzantine nodes have a chance to become primary nodes. When a Byzantine node becomes the primary node, the view-switching protocol is activated to choose a new primary node. This leads to a slower consensus process and adds complexity to the algorithm.

The consensus process of PBFT consists of five stages, which are request, pre-prepare, prepare, commit, and reply, as depicted in Figure 1. For more details, see [16].

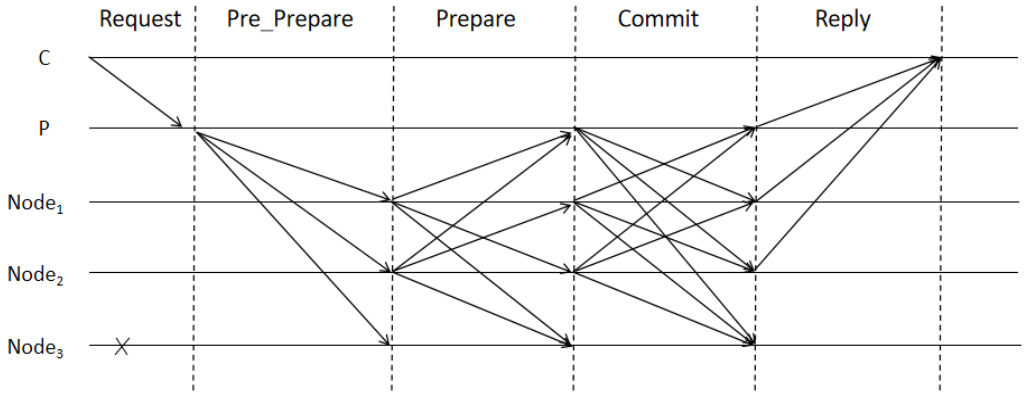

Figure 1. The consensus process of PBFT.

(1) In the request phase, the client node transmits a request message to the primary node.

(2) In the pre-prepare phase, the primary node examines and handles the message upon receiving it, and subsequently disseminates a pre-prepare message to the replica nodes. Each replica node receives and validates the message. Upon successful validation, the replica node acknowledges the request and proceeds to the prepare phase.

(3) In the preparation phase, replica nodes exchange and receive prepared messages from other different replica nodes. The prepare phase concludes when each replica node has received $2f + 1$ valid prepared messages from distinct replica nodes, where f represents the number of Byzantine nodes in the system.

(4) In the commit phase, each node transmits a commit message to the other replica nodes for verification. When each node has received $2f + 1$ commit messages that are consistent with the prepared messages, the commit phase is achieved.

(5) In the reply phase, each node sends reply messages to the client node. Upon receiving $f + 1$ reply messages from distinct nodes, the request is successfully executed, and a consensus is achieved.

3.2. The Reputation Evaluation Model with Adjustable Difficulty Based on the Sigmoid Function

A node's reputation directly mirrors its behavioral inclination. Nodes with good reputations tend to exhibit honest behavior, whereas nodes with poor reputations are more inclined to engage in Byzantine behavior. Therefore, we propose a reputation evaluation model to evaluate node reputation and select nodes with higher reputations to participate in the consensus process. In the model, reputation is reflected by the degree of participation of a node in the system. If a node actively participates in transactions with other nodes, it indicates that the node has a high reputation and other nodes are more willing to conduct transactions with it; otherwise, it indicates that the node has a poor reputation. We divide the transaction between nodes into two types, one is a direct transaction and the other is an indirect transaction, which is reflected in the degree of direct participation and the degree of indirect participation in the reputation evaluation model: (1) the degree of direct participation (DP): evaluation is carried out among nodes of direct transaction. (2) the degree of indirect participation (IP): It is evaluated between nodes without direct transactions and is related to DP. Therefore, the global degree of participation of node i can be expressed as $T_i = DP_i + IP_i$. The specific calculation process of the degree of participation is shown in Section 5.

The sigmoid function [38] is an S-type function, which has a good threshold value and can more conveniently screen nodes with higher reputation values. The Sigmoid function

can suppress the good and bad extremes and ensure the stability of system nodes. The formula of the sigmoid function adopted in this paper is:

$$f(x) = \frac{1}{1 + e^{-b(x-a)}} \tag{2}$$

The function when $a = 0$ is shown in Figure 2. By adjusting parameter b of the sigmoid function, we can change the difficulty of reputation evaluation and better control the quantity and quality of nodes participating in consensus. Parameter a can control the function center point, $f(a) = 0.5$. Let the node's initial reputation value $R' = a$.

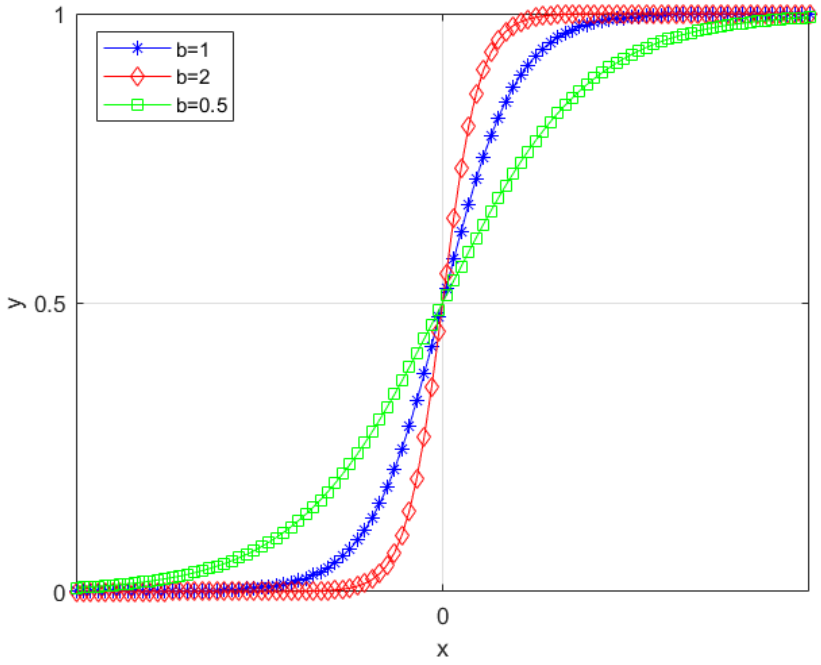

Figure 2. Sigmoid image when $a = 0$.

Finally, according to the global degree of participation and initial reputation value of the node, the reputation value of the node can be obtained by using the Sigmoid function. The reputation value of the node i is expressed as:

$$R_i = sigmoid(R' + T_i) \tag{3}$$

4. System Overview

Due to its anonymity, non-tampering, traceability, and other characteristics, blockchain provides a new solution for information security storage and security sharing, especially for medical data that involve more privacy. Medical data themselves have great value. Making full use of medical data can provide better medical services for patients. However, medical data involve a lot of personal privacy and cannot be stored and shared at will. Blockchain provides a solution for information security. However, it also faces the attack of Byzantine nodes. Reducing the presence of Byzantine nodes and improving consensus speed is a crucial area of research in blockchain technology. To address this, we introduce a novel consensus algorithm called ME-PBFT, which is based on the Practical Byzantine Fault Tolerance (PBFT) algorithm. The overall framework of our proposed algorithm is depicted in Figure 3.

As evident from Figure 3, our scheme is mainly divided into three parts: reputation evaluation model, consensus layer construction, and consensus process. Firstly, we introduce a reputation evaluation model based on the Sigmoid function to quantify node reputation value through the degree of participation. Then, different identities of nodes are

determined according to the reputation value of nodes, and different consensus layers are constructed from nodes with different identities. The structure of the consensus layer is shown in Figure 4. After the construction of the consensus layer structure, the consensus process begins.

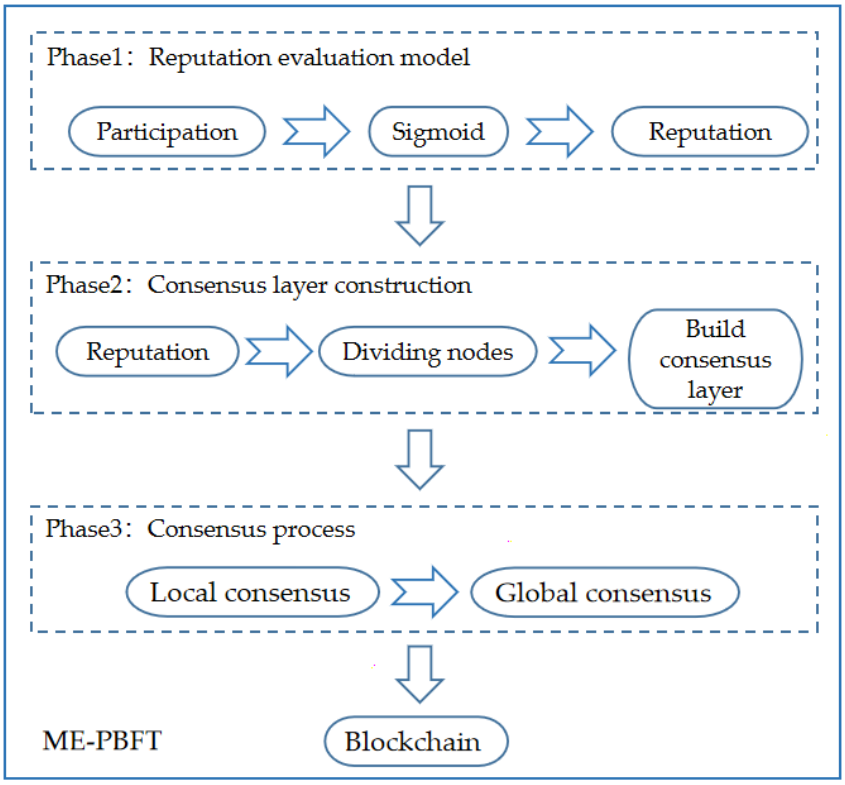

Figure 3. ME-PBFT framework.

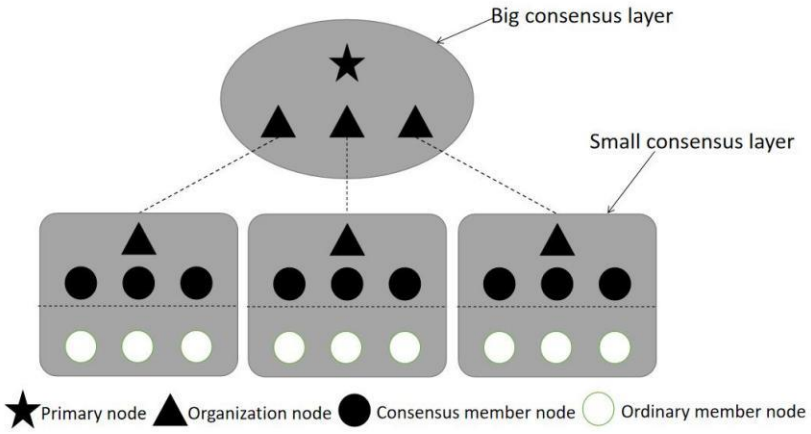

Figure 4. Structure of consensus layer.

We divided the consensus structure into two layers. There are two distinct consensus layers: the large consensus layer and the small consensus layer. The large consensus layer comprises each organization node, while the small consensus layer consists of the organization node and its associated member nodes. The two-layer consensus structure reduces the process of member node consensus among institutions, reduces the number of message propagations during the consensus process, and reduces algorithm complexity.

Our scheme defines four types of nodes, which are the primary node, organization node, consensus member node, and ordinary member node. A brief description of each node follows, and details are described in Section 5.

1. Primary node: The selection of the primary node is a random process carried out within the primary node group, which consists of organization nodes. The primary node is tasked with receiving client requests and initiating voting procedures.
2. Organization node: The organization node is formed by consensus member nodes that actively engage in the large consensus layer's consensus process, serving as the primary node group. Within the small consensus layer, the organization node assumes the role of the primary node.
3. Consensus member node: Member nodes with high reputation value participate in the small consensus.
4. Ordinary member node: Ordinary member nodes do not participate in the consensus.

In PBFT, all nodes participate in the consensus process. However, in our ME-PBFT scheme, the consensus process primarily involves the primary node, organization node, and consensus member node. By reducing the number of participating nodes, we are able to speed up the consensus process and simplify the algorithm. Moreover, these nodes are selected based on their high reputation value according to our reputation evaluation model, which helps in reducing the count of Byzantine nodes participating in the consensus and enhances the system's resilience. We have also improved the selection of primary nodes by having them randomly generated by the primary node group. If a primary node exhibits Byzantine behavior, the next node in the primary node group takes over as the primary node instead of triggering the view-switching protocol. This optimization helps in reducing the additional overhead caused by the view-switching protocol.

5. The Process of ME-PBFT

In this chapter, we mainly introduce the working process of ME-PBFT, including node reputation evaluation and consensus layer construction, the consensus process, and node dynamic entry and exit.

5.1. Node Reputation Evaluation and Consensus Layer Construction

In this part, we first evaluate the reputation of nodes, and then determine the identity of nodes according to the evaluation results so as to build the consensus layer.

5.1.1. Node Reputation Evaluation

Let us assume there are n nodes in this blockchain network. At the beginning of reputation evaluation, let $R' = a$, $T_i = 0$ for each node. Then, the reputation value $R_i = sigmoid(R' + T_i) = 0.5$. Thus, regardless of the specific difficulty parameter b, when the reputation evaluation commences, each node's reputation value is positioned at the midpoint of the reputation value range.

Algorithm 1 divides nodes based on whether there are direct transactions between nodes in ME-PBFT, in order to calculate the degree of direct and indirect participation of nodes in the next step. For a $node_i$, those that have direct transactions with $node_i$ are divided into the direct transaction node set $TxSets_i$ of $node_i$; otherwise, they are divided into the non-direct transaction node set $NonTxSets_i$.

Algorithm 1: Divide nodes

Input: $node_i$, all nodes
Output: $TxSets_i$, $NonTxSets_i$
$TxSets_i \leftarrow \emptyset$, $NonTxSets_i \leftarrow \emptyset$;
for $node_j \in$ *all nodes* **do**
  if $node_j$ transactions with $node_i$ **then**
    $TxSets_i \leftarrow node_j$;
  else
    $NonTxSets_i \leftarrow node_j$;
  end
end

Algorithm 2 describes the calculation process of the degree of direct participation, measuring the performance of nodes directly engaging in transactions in the algorithm. For $node_i$, if the direct transaction node set $TxSets_i$ is not an empty set, we inquire about the count of successful transactions and unsuccessful transactions between the entities $node_i$ and $node_j$. The degree of mutual direct participation DP_{ij} between $node_i$ and $node_j$ is calculated according to the query results. Finally, the degree of direct participation DP_i of $node_i$ is equal to the sum of all the degrees of mutual direct participation of $node_i$.

Algorithm 2: Calculate direct participation

Input: $node_i$,$TxSets_i$
Output: DP_i
$DP_i \leftarrow 0$, $DP_{ij} \leftarrow 0$;
if $TxSets_i = \emptyset$ **then**
  $DP_i = 0$
else
  for $node_j \in TxSets_i$ **do**
    $DP_{ij} = success(i,j) - fail(i,j)$;
  end
end
$DP_i = \sum_x DP_{ix}$

Algorithm 3 describes the degree of the indirect participation calculation process, measuring the performance of nodes in indirect transactions in the algorithm. When calculating the degree of indirect participation, we mainly consider nodes whose transaction path with $node_i$ is 2. Nodes with transaction paths greater than 2 are considered to have less impact on $node_i$'s reputation evaluation. First, the indirect participation IP_i is equal to zero if the direct transaction node set of $node_i$ is an empty set. If the direct transaction node set of $node_i$ is not an empty set, we find whether there exists an intermediate node that belongs to both the direct transaction node set of $node_i$ and the direct transaction node set of $node_j$. If it exists, the degree of mutual indirect participation $IP_{ij} = 1/n$, and n is the total number of nodes. Finally, for $node_i$, the degree of indirect participation IP_i is equal to the sum of all the degrees of mutual indirect participation of $node_i$.

Algorithm 3: Calculate indirect participation

Input: $node_i$,$TxSets_i$,$NonTxSets_i$
Output: IP_i
$IP_i \leftarrow 0$;
if $TxSets_i = \emptyset$ **then**
  $IP_i = 0$
else
  for $node_j \in NonTxSets_i$ **do**
    if $node_k \in TxSets_i$&$node_k \in TxSets_j$ **then**
      $IP_{ij} = 1/n$
    else
      Compute IP_{ij} iteratively;
    end
  end
end
$IP_i = \sum_x IP_{ix}$

Algorithm 4 describes the calculation method of the global reputation value, which measures the reputation level of nodes in the algorithm, in order to select higher quality nodes in the next step. Firstly, the degree of global participation T_i of $node_i$ is calculated according to its degree of direct and indirect participation. Then, the global reputation

value R_i can be obtained by obtaining the sigmoid function of $node_i'$s initial reputation value R' and the degree of global participation T_i.

Algorithm 4: Calc reputation value

```
1 Input: node_i, all nodes
2 Output: R_i
3 R_i ← 0;
4 for node_i ∈ all nodes do
5       T_i = DP_i + IP_i;
6       R_i = sigmoid(R' + T_i);
7 end
```

5.1.2. Construction of Consensus Layer

There are two consensus layers in our algorithm, namely the small consensus layer and the large consensus layer. The small consensus layer is mainly composed of consensus member nodes, while the large consensus layer is mainly composed of organization nodes and the primary node. The construction process of the consensus layer is mainly to determine the identity of nodes according to reputation value. Firstly, we use Algorithm 5 to divide nodes belonging to the same institution into the same node set, in order to meet the distribution characteristics of actual medical data and accelerate the consensus speed of the ME-PBFT algorithm. If $node_i$ belongs to k institution, put $node_i$ into the node set $InstNodes_k$.

Algorithm 5: Divide instituion

```
1 Input: node_i, all nodes
2 Output: InstNodes_k
3 InstNodes_k ← Ø;
4 for node_i ∈ all nodes do
5     if node_i ∈ InstNodes_k then
6         InstNodes_k ← node_i;
7     end
8 end
```

We describe the determination of consensus member nodes and organization nodes in Algorithm 6 and Algorithm 7, respectively. Algorithm 6 describes the process of determining consensus member nodes, reducing the number of nodes participating in the consensus and ensuring that high-reputation nodes participate in the consensus. At the same time, the small consensus layer is composed of consensus member nodes. In Algorithm 6,R_i is the reputation value of $node_i$. If $R_i \geq 0.8$, $node_i$ will be divided into $ConNodes_k$, the consensus member node set of its affiliated institution. Otherwise, $node_i$ will be divided into $NonConNodes_k$, the non-consensus node set. Algorithm 7 describes the process of determining organization nodes, selecting the node with the highest reputation value within the institution as the organization node to participate in the consensus of the big consensus layer. In Algorithm 7, we first rank all consensus member nodes $ConNodes_k$ of institution k according to reputation value. If the reputation value of $node_i$ ranks first, we define it as the organization node of the institution k.

The consensus layer Is constructed by dividing the identities of nodes, and the large consensus layer is constructed by the organization nodes, in which the primary node will be taken turns by the organization nodes. Apart from the primary node and the organization nodes, the member nodes participating in the consensus together constitute the small consensus layer.

Algorithm 6: Identify consensus nodes

1 **Input**: $node_i$, $InstNodes_k$, R_i
2 **Output**: $ConNodes_k$, $NonConNodes_k$
3 $ConNodes_k \leftarrow \emptyset$, $NonConNodes_k \leftarrow \emptyset$;
4 **for** $node_i \in InstNodes_k$ **do**
5 **if** $R_i \geq 0.8$ **then**
6 $ConNodes_k \leftarrow node_i$;
7 **else**
8 $NonConNodes_k \leftarrow node_i$;
9 **end**
10 **end**

Algorithm 7: Identify organization nodes

1 **Input**: $ConNodes_k$, Reputation set R in $Instituion_k$
2 **Output**: $OrgNodes_k$
3 $OrgNodes_k \leftarrow \emptyset$;
4 **Sort** $ConNodes_k$ by R
5 **for** $node_i \in ConNodes_k$ **do**
6 **if** R_i is in the top1 **then**
7 $OrgNodes_k \leftarrow node_i$;
8 **end**
9 **end**

5.2. Consensus Process

Our algorithm consensus process mainly includes two parts, namely the large consensus process and the small consensus process. The two consensus processes add up to a total of 9 consensus phases. The consensus process is depicted in Figure 5, and here are the specific details of the consensus:

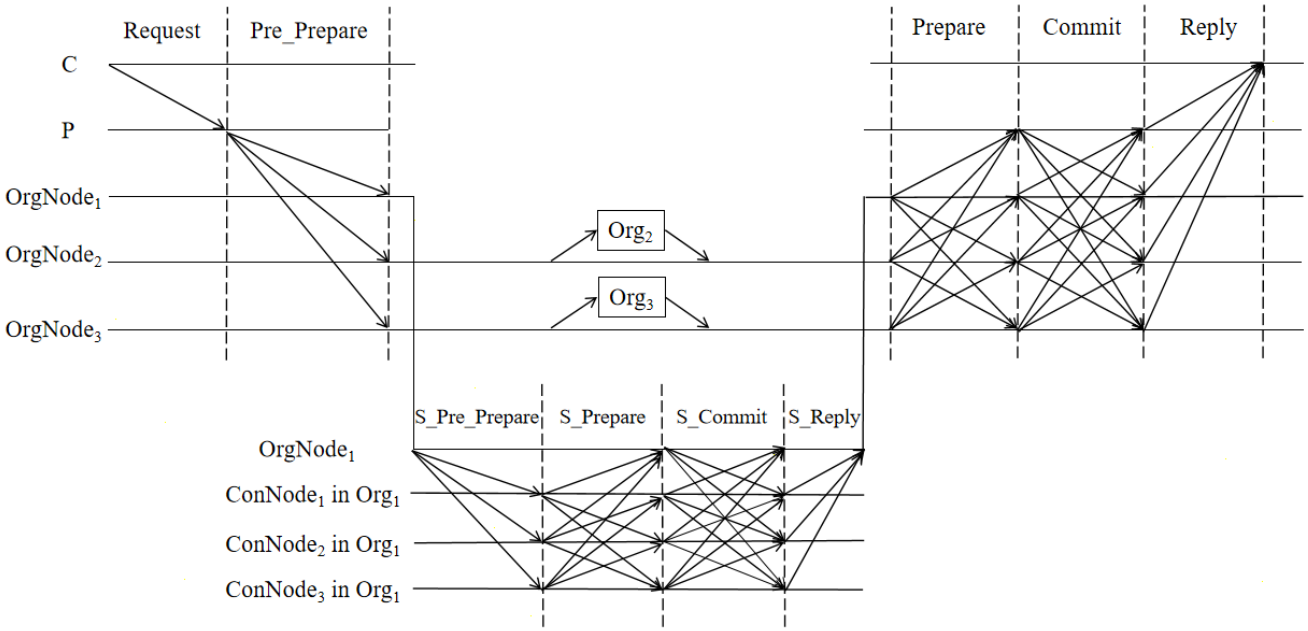

Figure 5. Algorithm consensus process.

(1) In the request phase of the large consensus layer, the client transmits a message $\langle REQUEST, o, t, c \rangle_{\sigma_m}$ to the primary node. Here, o represents a state machine operation, and t signifies the current timestamp, c is the client identity, and σ_m is the signature of the message m.

(2) In the pre_prepare phase of the large consensus layer, the primary node checks the received information, and then broadcasts a message $\left\langle \langle PRE\_PREPARE, v, n, d \rangle_{\sigma_p}, m \right\rangle$ to the organization node, where v is the view number, n is the serial number assigned by the primary node to the request, d is the hashed value of m, and σ_p is the signature of the primary node. Each organization node receives the message and verifies it.

(3) In the s_pre_prepare phase of the small consensus layer, the organization node broadcasts the a message $\left\langle \langle S\_PRE\_PREPARE, v, n, d, i \rangle_{\sigma_i}, m \right\rangle$ to the consensus member node of the organization, where i represents the identifier of the current node, and σ_i is the signature of the current node. Consensus member nodes of the small consensus layer will verify the message after receiving it. Once the verification passes, consensus member nodes will enter the s_prepare phase.

(4) During the s_prepare phase of the small consensus layer, the consensus member nodes broadcast a message $\langle S\_PREPARE, v, n, d, i \rangle_{\sigma_i}$ to each other and receive the S_PREPARE message from other consensus member nodes of the organization. The s_prepare phase is completed when each consensus member node obtains $2f_1 + 1$ valid S_PREPARE messages. Here, f_1 represents the number of Byzantine nodes within the organization.

(5) In the s_commit phase of the small consensus layer, each consensus member node will broadcast a S_COMMIT message $\langle S\_COMMIT, v, n, i \rangle_{\sigma_i}$ to other consensus member nodes of the organization for verification. When each consensus member node has received $2f_1 + 1$ S_COMMIT messages that are consistent with the S_PRE_PREPARE messages, the s_commit phase ends.

(6) In the s_reply phase of the small consensus layer, each consensus member node transmits an S_REPLY message denoted as $\langle S\_REPLY, v, n, d, i \rangle_{\sigma_i}$ to the organization node of its own organization. When the organization node receives the same S_REPLY message sent by $f_1 + 1$ different consensus member nodes, the small consensus phase is completed and the prepare phase of the large consensus layer is entered.

(7) During the prepare phase of the large consensus layer, each organization node broadcasts a PREPARE message $\langle PREPARE, v, n, d, i \rangle_{\sigma_i}$ to other organization nodes and receives the PREPARE message from other organization nodes. The prepare phase concludes when every organization node acquires $2f_2 + 1$ valid PREPARE messages, where f_2 is the count of Byzantine nodes in the large consensus layer.

(8) In the commit phase of the large consensus layer, each organization node broadcasts a COMMIT message denoted as $\langle COMMIT, v, n, i \rangle_{\sigma_i}$ to other organization nodes for verification. When each node has received $2f_2 + 1$ COMMIT messages that match with the PRE_PREPARE messages, the commit phase of the large consensus layer is complete.

(9) In the reply phase of the large consensus layer, the organization nodes send REPLY messages $\langle REPLY, v, t, c, i, r \rangle_{\sigma_i}$ to the client. Here, r stands for the result of the operation. When the client receives the same REPLY messages sent by $f_2 + 1$ different nodes, the request is successfully executed and the final consensus is reached.

5.3. Dynamic Join and Exit

The procedure of node joining and exiting in this paper is based on the overall architecture of this algorithm. When a node joins or exits, it first selects its own organization, and then completes the joining request among the consensus member nodes within the organization. It does not need to verify the whole network nodes, which greatly reduces the communication complexity when nodes join or exit. The specific node joining process is shown in Figure 6, and the details are as follows.

(1) During the join_request phase, the new node transmits a JOIN_REQUEST message $\langle JOIN\_REQUEST, t, j, PK \rangle_{\sigma_j}$ to all participating nodes in the organization's consensus. Here, t stands for the current timestamp, j represents the identifier of the new node, PK is the public key of the new node, and σ_j signifies the signature of the new node.

(2) During the join phase, upon receipt of the JOIN_REQUEST message, the consensus-participating nodes within the organization perform identity verification on the new node.

Once the new node's identity is confirmed, these consensus nodes broadcast a JOIN message $\langle JOIN, v, h, C, j, i \rangle_{\sigma_i}$, where v corresponds to the view number, h represents the number of the nearest stable checkpoint known to the node, C is a set containing valid checkpoint information, i is the identifier of the current node, and σ_i is the signature of the current node.

(3) During the view_change phase, when the organization node receives $2f + 1$ valid JOIN messages, the organization node transmits a VIEW_CHANGE message in the large consensus layer to trigger the view conversion.

(4) During the new_view phase, the organization node broadcasts to the new node and other consensus member nodes $\langle NEW\_VIEW, v + 1, D, O, j, i \rangle_{\sigma_i}$, where D represents a collection of $2f + 1$ valid VIEW_CHANGE messages and O is a set of PRE_PREPARE messages. After the consensus member node in the organization passes the verification, the new node will be added to the system.

(5) During the join_reply phase, all nodes engaged in a consensus within the organization transmit JOIN_REPLY messages $\langle JOIN\_REPLY, v + 1, p, t, j, i \rangle_{\sigma_i}$ to the new node. These messages include set p, which is a set of client requests related to the message in O, enabling $node_j$ to handle these requests in the new view. Once $node_j$ receives $f + 1$ valid JOIN_REPLY message, it can commence participation in the consensus.

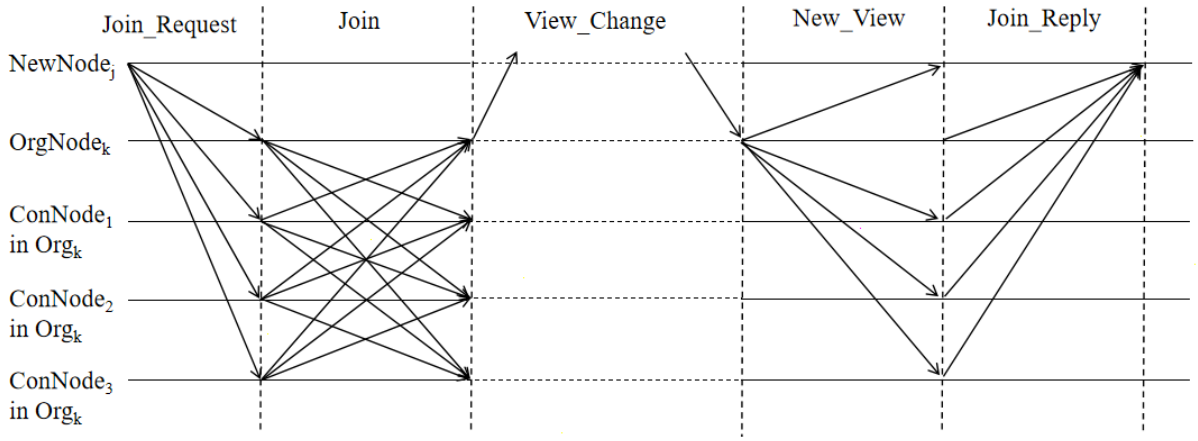

Figure 6. New node joining process.

6. Analysis and Comparison

In this section, we initially performed a theoretical analysis of our scheme, focusing on three key aspects: communication complexity, fault tolerance, and scalability. Then, in Section 6.4, we compared our scheme with PBFT and RAFT algorithms through experiments.

6.1. Communication Complexity

We study the communication complexity of algorithms by analyzing the number of transmitted messages. In PBFT, all nodes participate in a consensus and propagate information. Its algorithm complexity is $O(n^2)$, where n denotes the overall number of nodes in the system. In our scheme, a two-layer consensus model was established, and through a reputation evaluation model, nodes with high reputation value were selected to participate in the system consensus. Therefore, the number of nodes involved in the consensus is fewer than n. In addition, we introduce the sigmoid function with adjustable parameters to adjust the difficulty of reputation evaluation and control the number of nodes entering the consensus.

Conclusion: The communication complexity of ME-PBFT is better than that of PBFT.

Proof. Suppose there are a total of K institutions in the system, and the number of nodes excluding the institutional nodes in each institution is $M_1, M_2, \ldots, M_k$, respectively.

The proportional coefficients of nodes participating in consensus in each institution are $X_1, X_2, \ldots, X_k$, $0 < (X_1, X_2, \ldots, X_k) < 1$.

The total number of nodes:

$$N = M_1 + M_2 + \ldots + M_k + K \tag{4}$$

The algorithm complexity of PBFT is:

$$A = N^2 = (M_1 + M_2 + \ldots + M_k + K)^2 \tag{5}$$

ME-PBFT can be regarded as $K + 1$ PBFT groups, and the algorithm complexity of ME-PBFT is:

$$B = (X_1 M_1)^2 + (X_2 M_2)^2 + \ldots + (X_k M_k)^2 + K^2 \tag{6}$$

When $B < A$, the communication complexity of ME-PBFT is better than that of PBFT. □

6.2. Fault Tolerance

The goal of consensus algorithms is to enable distributed nodes to quickly reach consensus. In actual systems, there may be network errors, malicious attacks, Byzantine nodes, and other situations that affect system consensus. Therefore, the fault tolerance capability of algorithms has a direct impact on the security of consensus algorithms. In PBFT, all nodes will participate in a consensus, and any malicious behavior of nodes will cause a decrease in the system's fault tolerance. In our scheme, the fault tolerance ability of the algorithm has been optimized. We employ a reputation assessment model to carefully choose a subset of trustworthy nodes from the entire system to engage in the consensus process.

Conclusion: ME-PBFT exhibits superior fault tolerance compared to PBFT.

Proof. Suppose there are a total of K institutions in the system, and the count of nodes excluding the institutional nodes in each institution is $M_1, M_2, \ldots, M_k$, respectively.

The proportional coefficients of nodes participating in a consensus in each institution are $X_1, X_2, \ldots, X_k$, $0 < (X_1, X_2, \ldots, X_k) < 1$.

The fault tolerance capability of PBFT is:

$$C = \frac{N-1}{3} = \frac{M_1 + M_2 + \ldots + M_k + K - 1}{3} \tag{7}$$

ME-PBFT can be regarded as $K + 1$ independent PBFT groups. The fault tolerance capability of ME-PBFT is:

$$\begin{aligned} D &= \tfrac{X_1 M_1 + X_2 M_2 + \ldots + X_k M_k + K - 1}{3} + (1 - X_1) M_1 + (1 - X_2) M_2 + \ldots + (1 - X_k) M_k \\ &= \tfrac{3M_1 - 2X_1 M_1 + 3M_2 - 2X_2 M_2 + \ldots + 3M_k - 2X_k M_k + K - 1}{3} \end{aligned} \tag{8}$$

$(1 - X_k) M_k$ indicates the count of nodes in organization K that do not participate in the consensus, which does not impact the fault tolerance capability of the system.

$$3M_k - 2X_k M_k > M_k, D > C. \tag{9}$$

ME-PBFT exhibits superior fault tolerance compared to PBFT. □

6.3. Scalability

In the PBFT algorithm, if the number of nodes becomes excessively large, it significantly degrades the algorithm's performance, imposing limitations on the practicality of blockchain systems employing PBFT. In our proposed approach, we implement a reputation assessment model based on the Sigmoid function to select nodes with higher reputation

values for participation in the consensus process. This effectively reduces the number of nodes actively involved in a consensus. Moreover, as the number of nodes continues to grow, we can adjust the difficulty parameter of the Sigmoid function to fine-tune the reputation evaluation process, enabling better control over the number of nodes engaged in the consensus. Consequently, our solution can accommodate a larger number of nodes without the need for arbitrary restrictions on the blockchain system's node count. Furthermore, we have developed a dynamic node enrollment and exit mechanism to facilitate the onboarding of high-quality nodes and the removal of Byzantine nodes. This program enhances system scalability and provides assistance in maintaining the network's integrity and performance.

6.4. Comparison

To demonstrate the performance of the ME-PBFT algorithm, we conducted comparative simulation experiments in the same environment with a classical RAFT algorithm and the commonly used consensus algorithm PBFT, focusing on communication times and data throughput. The simulation environment included Windows 10, AMD Ryzen 7 5800 H with Radeon Graphics 3.20 GHz CPU, and 16.0 GB RAM. We developed a prototype system for medical data sharing based on ME-PBFT. Monitoring points were set for each network node to observe performance changes. We compared and analyzed the differences in consensus efficiency and throughput among the three algorithms.

1. Communication times: Communication times refers to the amount of communication needed by nodes during the consensus process. It serves as an indicator of the consensus algorithm's efficiency. Figure 7 illustrates the communication times for the RAFT, PBFT, and ME-PBFT algorithms. As the number of nodes increases, the communication times for all three algorithms gradually rises. Notably, RAFT exhibits the most moderate increase in communication latency. Conversely, the PBFT algorithm experiences the fastest increase due to each node having to communicate in three phases. Our approach mitigates this by reducing the number of nodes actively participating in a consensus and employing a two-layer consensus structure to effectively minimize communication volume. As a result, ME-PBFT demonstrates significantly slower growth in communication times compared to PBFT.

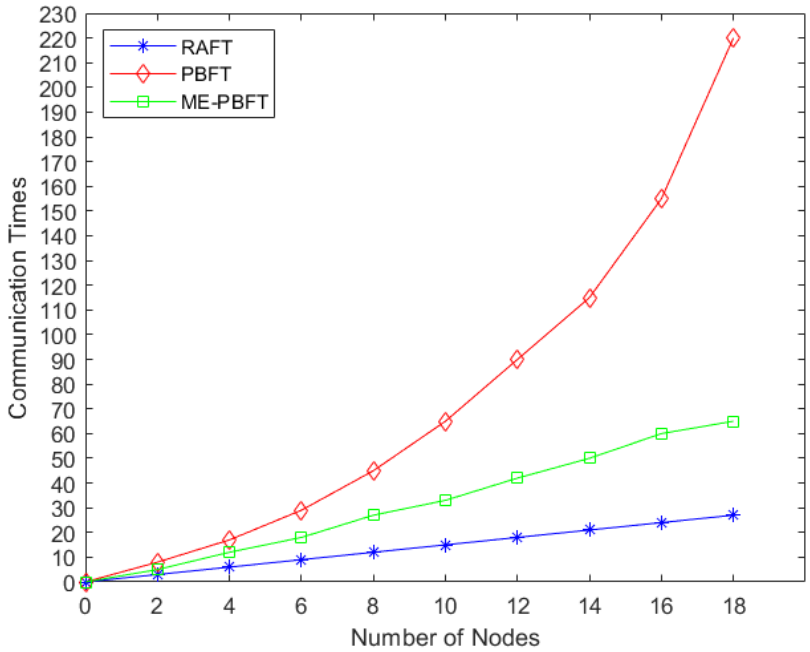

Figure 7. Comparison of the communication times.

2. Data Throughput: Data throughput is commonly considered a critical indicator of algorithm performance, and this holds true for blockchain consensus algorithms as well. In the context of blockchain, data throughput can be defined as the quantity of transactions processed within a specific time frame, typically measured in terms of TPS (Transactions Per Second):

$$TPS = \frac{transactions}{\Delta t} \tag{10}$$

Figure 8 illustrates the trend of the number of packaged transactions as the running time increases for the three algorithms. According to the TPS formula, the tangent line's slope corresponds to data throughput. It was observed that the TPS of ME-PBFT was greater than that of PBFT and RAFT with increasing time.

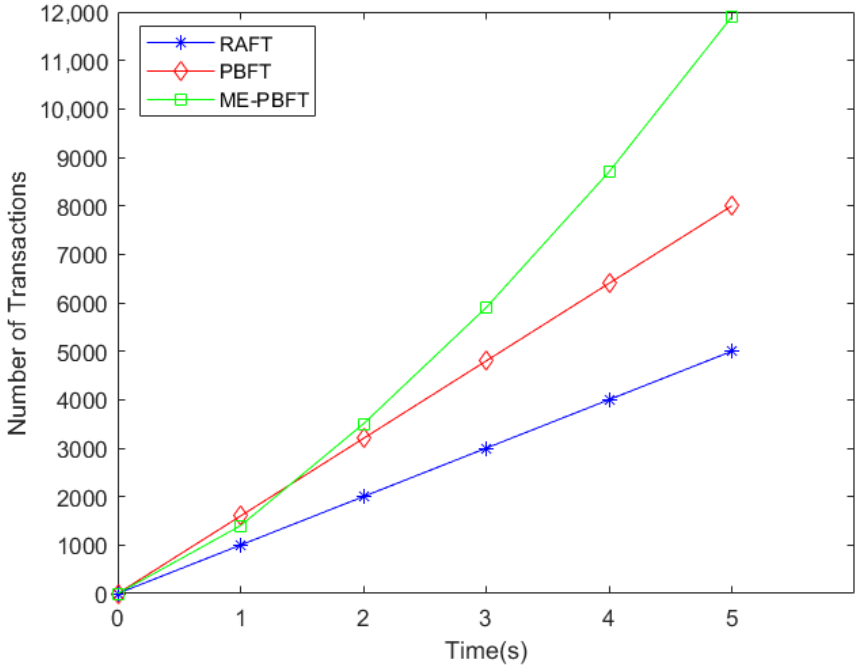

Figure 8. Comparison of data throughput.

7. Conclusions

This paper introduces ME-PBFT, a new consensus algorithm based on PBFT, specifically designed for medical data security. The algorithm proposes a reputation evaluation model using the sigmoid function to select high-quality nodes for consensus, reducing the number of nodes involved and improving communication efficiency. The paper also presents a double-consensus layer structure to speed up the consensus process and achieve global consensus through local consensus. Additionally, a node dynamic join and exit mechanism is introduced to allow for the inclusion of new nodes and the removal of malicious nodes, further enhancing system security and scalability. The analysis demonstrates that ME-PBFT improves consensus efficiency, security, and scalability, making it suitable for medical data security scenarios. Future work will focus on optimizing the system model to make it applicable to a wider range of scenarios.

Author Contributions: Conceptualization, G.X. and T.Y.; methodology, G.X., T.Y. and K.Z.; software, X.M.; validation, X.L.; formal analysis, K.X.; investigation, G.X., T.Y. and K.Z.; re-sources, X.C.; data curation, X.M.; writing—original draft preparation, T.Y.; writing—review and editing, G.X.; visualization, K.Z.; supervision, G.X.; project administration, X.C.; funding acquisition, G.X. and K.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by Yunnan Key Laboratory of Blockchain Application Technology (202105AG070005) & YNB202301, the Project of Science and Technology Major Project of Yunnan Province (202303AC100003), the National Natural Science Foundation of China under Grant No. 62271234, the Open Foundation of State Key Laboratory of Public Big Data (Guizhou University) under Grant No. PBD2022-16, and the Double First-Class Project for Collaborative Innovation Achievements in Disciplines Construction in Heilongjiang Province under Grant No. LJGXCG2022-054. The work of Kejia Zhang was also supported by the Advanced Programs of Heilongjiang Province for the overseas scholars and the Fundamental Research Funds for Heilongjiang Universities.

Data Availability Statement: No data were used for the research described in the article.

Conflicts of Interest: The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

1. Nakamoto, S. Bitcoin: A Peer-to-Peer Electronic Cash System. Available online: http://bitcoin.org/bitcoin.pdf (accessed on 1 April 2022).
2. Michael, M.; Mills, S. The Missing Links in the Chains? Mutual Distributed Ledger (aka Blockchain) Standards. *Soc. Sci. Electron. Publ.* **2016**, *8*, 11–15.
3. Fox, G. Peer-to-peer Networks. *Comput. Sci. Eng.* **2001**, *3*, 75–77. [CrossRef]
4. Diffie, W.; Hellman, M. New Directions in Cryptography. *IEEE Trans. Inf. Theory* **2006**, *22*, 644–654. [CrossRef]
5. Watanabe, H.; Fujimura, S.; Nakadaira, A. Blockchain contract: Securing a blockchain applied to smart contracts. In Proceedings of the 2016 IEEE International Conference on Consumer Electronics (ICCE), Las Vegas, NV, USA, 7–11 January 2016; pp. 467–468.
6. Yang, R.; Wakefield, R.; Lyu, S. Public and private blockchain in construction business process and information integration. *Autom. Constr.* **2020**, *118*, 103276. [CrossRef]
7. Zheng, Z.; Xie, S.; Dai, H.; Chen, X.; Wang, H. An Overview of Blockchain Technology: Architecture, Consensus, and Future Trends. In Proceedings of the 2017 IEEE International Congress on Big Data (Big Data Congress), Honolulu, HI, USA, 25–30 June 2017; pp. 557–564.
8. Wood, G. Ethereum: A secure decentralised generalised transaction ledger. *Ethereum Proj. Yellow Pap.* **2014**, *151*, 1–32.
9. Andola, N.; Gogoi, M.; Venkatesan, S. Vulnerabilities on hyperledger fabric. *Pervasive Mob. Comput.* **2019**, *59*, 101050. [CrossRef]
10. Abeywardena, K.Y.; Attanayaka, B.; Periyasamy, K. Blockchain based Patients' detail management System. In Proceedings of the 2020 2nd International Conference on Advancements in Computing (ICAC), Malabe, Sri Lanka, 10–11 December 2020; pp. 458–463.
11. Wang, W.; Hoang, D.T.; Hu, P. A survey on consensus mechanisms and mining strategy management in blockchain networks. *IEEE Access* **2019**, *7*, 22328–22370. [CrossRef]
12. Jin, S.; Zhang, X.; Ge, J. Overview of blockchain consensus algorithm. *J. Inf. Secur.* **2021**, *6*, 85–100.
13. Huang, D.; Ma, X.; Zhang, S. Performance analysis of the raft consensus algorithm for private blockchains. *IEEE Trans. Syst. Man Cybern. Syst.* **2019**, *50*, 172–181. [CrossRef]
14. Thin, W.Y.M.M.; Dong, N.; Bai, G. Formal Analysis of a Proof-of-Stake Blockchain. In Proceedings of the 2018 23rd International Conference on Engineering of Complex Computer Systems (ICECCS), Melbourne, VIC, Australia, 12–14 December 2018; pp. 197–200.
15. Snider, M.; Samani, K.; Jain, T. Delegated proof of stake: Features & tradeoffs. *Multicoin Cap.* **2018**, *19*, 1–19.
16. Castro, M.; Liskov, B. Practical byzantine fault tolerance and proactive recovery. *ACM Trans. Comput. Syst. (TOCS)* **2002**, *20*, 398–461. [CrossRef]
17. Azaria, A.; Ekblaw, A.; Vieira, T.; Lippman, A. MedRec: Using blockchain for medical data access and permission management. In Proceedings of the 2nd International Conference on Open and Big Data (OBD), Vienna, Austria, 22–24 August 2016; pp. 25–30.
18. Teng-Fei, X.; Qun-Chao, F.; Cong, W.; Xin-Yan, W. A medical data sharing model via blockchain. *Acta Autom. Sin.* **2017**, *43*, 1555–1562.
19. Kuo, T.T.; Ohno-Machado, L. Modelchain: Decentralized privacy-preserving healthcare predictive modeling framework on private blockchain networks. *arXiv* **2018**, arXiv:1802.01746.
20. Qu, J. Blockchain in medical informatics. *J. Ind. Inf. Integr.* **2022**, *25*, 100258. [CrossRef]
21. Distler, T. Byzantine fault-tolerant state-machine replication from a systems perspective. *ACM Comput. Surv. (CSUR)* **2021**, *54*, 1–38. [CrossRef]
22. Ateniese, G.; Bonacina, I.; Faonio, A.; Galesi, N. Proofs of space: When space is of the essence. In Proceedings of the Security and Cryptography for Networks: 9th International Conference, Amalfi, Italy, 3–5 September 2014; pp. 538–557.
23. Ball, M.; Rosen, A.; Sabin, M.; Vasudevan, P.N. Proofs of useful work. Available online: https://eprint.iacr.org/2017/203 (accessed on 1 March 2017).
24. Lamport, L.; Shostak, R.; Pease, M. The Byzantine Generals Problem. *ACM Trans. Program. Lang. Syst.* **1982**, *4*, 382–401. [CrossRef]
25. Kotla, R.; Alvisi, L.; Dahlin, M. Zyzzyva: Speculative byzantine fault tolerance. *ACM Trans. Comput. Syst. (TOCS)* **2010**, *27*, 1–39. [CrossRef]

26. Miller, A.; Xia, Y.; Croman, K. The honey badger of BFT protocols. In Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, New York, NY, USA, 24 October 2016; pp. 31–42.
27. Biryukov, A.; Feher, D.; Khovratovich, D. Guru: Universal reputation module for distributed consensus protocols. *Cryptol. Eprint Arch.* **2017**, *671*, 1–20.
28. He, L.; Hou, Z. An Improvement of Consensus Fault Tolerant Algorithm Applied to Alliance Chain. In Proceedings of the 2019 IEEE 9th International Conference on Electronics Information and Emergency Communication (ICEIEC), Beijing, China, 12–14 July 2019; pp. 1–4.
29. Gao, S.; Yu, T.; Zhu, J.; Cai, W. T-PBFT: An EigenTrust-Based Practical Byzantine Fault Tolerance Consensus Algorithm. *China Commun.* **2019**, *16*, 111–123. [CrossRef]
30. Zhang, J.; Rong, Y.; Cao, J.; Rong, C.; Bian, J.; Wu, W. DBFT: A Byzantine Fault Tolerance Protocol With Graceful Performance Degradation. *IEEE Trans. Dependable Secur. Comput.* **2021**, *19*, 3387–3400. [CrossRef]
31. Wang, H.; Guo, K. Byzantine Fault Tolerant Algorithm Based on Vote. In Proceedings of the 2019 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC), Guilin, China, 17–19 October 2019; pp. 190–196.
32. Li, Y.; Qiao, L.; Lv, Z. An Optimized Byzantine Fault Tolerance Algorithm for Consortium Blockchain. *Peer-Peer Netw. Appl.* **2021**, *14*, 2826–2839. [CrossRef]
33. Zhan, Y.; Wang, B.; Lu, R. DRBFT: Delegated randomization Byzantine fault tolerance consensus protocol for blockchains. *Inf. Sci.* **2021**, *559*, 8–21. [CrossRef]
34. Lao, L.; Dai, X.; Xiao, B. G-PBFT: A Location-based and Scalable Consensus Protocol for IoT-Blockchain Applications. In Proceedings of the 2020 IEEE International Parallel and Distributed Processing Symposium (IPDPS), New Orleans, LA, USA, 18–22 May 2020; pp. 664–673.
35. Xu, G.; Bai, H.; Xing, J. SG-PBFT: A secure and highly efficient distributed blockchain PBFT consensus algorithm for intelligent Internet of vehicles. *J. Parallel Distrib. Comput.* **2022**, *164*, 1–11. [CrossRef]
36. Pang, Z.; Yao, Y.; Li, Q. Electronic Health Records Sharing Model Based on Blockchain With Checkable State PBFT Consensus Algorithm. *IEEE Access* **2022**, *10*, 87803–87815. [CrossRef]
37. Mišić, J.; Mišić, V.B.; Chang, X. Design of Proof-of-Stake PBFT Algorithm for IoT Environments. *IEEE Trans. Veh. Technol.* **2022**, *72*, 2497–2510. [CrossRef]
38. Ezeafulukwe, U.A.; Darus, M.; Fadipe-Joseph, O. On analytic properties of a sigmoid function. *Int. J. Math. Comput. Sci.* **2018**, *13*, 171–178.