Peer-to-Peer Networking and Applications
https://doi.org/10.1007/s12083-022-01373-w

MANDALA: A scalable blockchain model with mesh-and-spoke network and H-PBFT consensus algorithm

Jinze Li[1,2] · Xiaofeng Li[1,2] · He Zhao[1] · Bin Yu[4] · Tong Zhou[3,4] · Haotian Cheng[1] · Nianzu Sheng[3,4]

Received: 9 March 2022 / Accepted: 19 August 2022

Abstract
With the rapid development of blockchain technology, the scale of its participants continues to expand. The network structure and the PBFT consensus algorithm of the blockchain have problems such as low transmission efficiency and high communication overhead, resulting in poor scalability. To solve these problems, we aim to improve the scalability of the blockchain so that it can support large-scale nodes for efficient transmission and communication. In this study, we propose a model named MANDALA with Mesh-and-Spoke Network and H-PBFT Consensus. The Mesh-and-Spoke network groups nodes into different layers and regulates communication rules among groups, which improves the network transmission efficiency. Then, we propose the Hierarchical Practical Byzantine Fault Tolerance (H-PBFT) consensus algorithm. It divides the consensus of the whole network into several sub-layers, which achieves lower communication complexity and improves fault tolerance. We simulated the model to validate its performance and security. The results indicated that it reduced the communication overhead and improved the effective transmission rate and throughput under the premise of ensuring security. Compared with other blockchain optimization schemes, our model features better consensus efficiency, security, and scalability.

Keywords Blockchain · Network structure · Consensus · Scalability

1 Introduction

Blockchain has received a lot of attention from academia and industry, and is developing towards a wider range of fields. There are many applications in digital assets [1], AI [2], IoT [3], information sharing [4], medical [5], social networking [6], and other fields. The essence of blockchain is a decentralized distributed ledger, which is immutable, traceable, transparent, and reliable. It effectively reduces the cost of trust and is becoming an important foundation for future information technology and the Internet.

Although blockchain technology improves the reliability of data, it suffers from poor scalability in practical use, mainly in performance and node expansion [7]. With the rapid increase in the number of transactions on chain, the performance of the blockchain cannot keep up with high-frequency operations. For example, the throughput of Bitcoin is about 7 transactions per second (tps) [8], and that of Ethereum is 10 ~ 30 tps [9]. Such performance cannot meet the demand for timely confirmation of a large number of transactions. Meanwhile, the problems of redundant data transmission and low transmission efficiency also widely exist in the blockchain network [10], which will aggravate

✉ He Zhao
zhaoh@hfcas.ac.cn

✉ Tong Zhou
tzhou@hfcas.ac.cn

Jinze Li
lijinze@mail.ustc.edu.cn

Xiaofeng Li
xfli@hfcas.ac.cn

Bin Yu
yub@hfcas.ac.cn

Haotian Cheng
htcheng@hfcas.ac.cn

Nianzu Sheng
shengnianzu@163.com

[1] Hefei Institutes of Physical Science, Chinese Academy of Sciences, Hefei 230031, People's Republic of China

[2] University of Science and Technology of China, Hefei 230026, People's Republic of China

[3] Blockchain and Perceptual Computing Engineering and Technology R & D Center, CAS(Hefei) Institute of Technology Innovation, Hefei, China

[4] Anhui ZhongKeJingGe Technology Co., Ltd., Hefei, China

Published online: 04 October 2022

network congestion and seriously degrade the performance of the blockchain. Moreover, as the number of nodes increases, the consensus efficiency tends to drop rapidly, restricting its application in larger node scales. For example, the Byzantine Fault Tolerant (BFT) algorithm [11] is inefficient in systems with large-scale nodes. Therefore, it is mainly used in consortium chains with fewer nodes.

The network structure of the blockchain affects the network transmission efficiency, which is one of the reasons for the scalability problem. Bitcoin [1] adopts an unstructured network. The flooding mechanism can easily cause problems such as broadcast storms and information redundancy, which reduces the transmission efficiency of the network [12]. Ethereum [13] uses a structured network, and network fluctuations caused by frequent node changes will greatly increase the maintenance cost [14]. In addition, the consensus algorithm is a fundamental part of the blockchain, which has a great impact on the scalability of the blockchain. Common consensus algorithms include Proof-of-Work (PoW) [15], Proof-of-Stake (PoS) [16], Practical Byzantine Fault Tolerance (PBFT) [17]. In a blockchain with PoW consensus, members use their computing power to compete for hash operations, but it consumes lots of computing resources and generally has low throughput [18]. PoS consensus favors the participants with more tokens, which increases the risk of monopoly and centralization [19]. PBFT is more performant and doesn't consume many computing resources, but its communication complexity is as high as $O(N^2)$ [20]. With the expansion of node scale, its communication overhead will increase rapidly, and performance such as consensus efficiency and throughput will deteriorate significantly.

According to the above analysis, the network structure and consensus algorithm limit the performance and node scale of the blockchain. We propose a model named MANDALA to improve the scalability of blockchain.

The main contributions of this paper are as follows:

1. We propose a Mesh-and-Spoke network structure, which divides the blockchain network into multiple layers. The advantage is that it can improve the scalability and data transmission efficiency of nodes. Moreover, we set up isolation mechanism, redundancy mechanism, reorganization mechanism, incentive and punishment mechanism. These improve the security and prevent consensus from being monopolized by a few nodes.
2. We propose a consensus algorithm suitable for the Mesh-and-Spoke network, called Hierarchical Practical Byzantine Fault Tolerance (H-PBFT). It divides the consensus into several sub-layers based on the Mesh-and-Spoke network, which reduces the communication complexity and improves fault tolerance. With the increase in the number of ordinary nodes, the communication efficiency and throughput of H-PBFT won't degrade as rapidly as PBFT.
3. We implemented a prototype of MANDALA and compared it with mainstream blockchain improvement models. The experiments demonstrate that MANDALA has good scalability, and its performance and security are better in scenarios with a large node scale.

The rest of this paper is organized as follows. In Sect. 2, we introduce related research on blockchain network structure and PBFT. In Sect. 3, we introduce the framework of MANDALA. In Sect. 4, we introduce the design of the model, mainly including the details of the Mesh-and-Spoke network and H-PBFT. In Sect. 6, we build a complete model and describe the specific process of the model. In Sect. 6.2.2, we analyze the performance and security of MANDALA and compare it with other blockchain systems. In Sect. 7, we conclude this paper and outline future work.

2 Related work

2.1 Research on network structure

The network adopted by the blockchain mainly has four structures: centralized, unstructured, structured, and hybrid [10]. The centralized network adopts a Hub-and-Spoke topology, which is easy to maintain and has high query efficiency. But it is prone to a single point of failure, and the network lacks scalability. The typical researches include Napster [21], Fastpass [22], and IMS [23], etc. The unstructured network adopts the organization method of random graph to form a mesh network, which is robust but has a high network overhead. The main researches include Bitcoin [1], R2Trust [24], and ColChain [25], etc. The structured network uses Distributed Hash Table to manage nodes, and the related schemes include Ethereum [13], IoT Scalable Model [26], Tree Graphs [27], etc. They reduce query latency and make the network more stable, but increase the cost of maintaining the structure, which affects the utilization of the network. The hybrid network selects nodes with higher performance as supernodes to handle search tasks. The related applications include EOS [28], RCANE [29], etc. This network structure is more dependent on supernodes, which affects transmission efficiency and security.

Most of the above network structures suffer from low transmission efficiency. Nodes are vulnerable to attacks, and their security will be affected. In this paper, We combine the advantages of centralized and unstructured networks and propose a Mesh-and-Spoke network, which can improve the effective transmission rate and the security of consensus nodes.

2.2 Research on PBFT improvement

The research on PBFT is highly active in the blockchain field. BFT-DPoS [30] and DBFT [31] apply DPoS [32] to the BFT, improving the throughput, but their communication complexity is still $O(N^2)$. Schemes such as CheapBFT [33] and FastBFT [34] use secure hardware to build a trusted execution environment, reducing the communication phase. However, in large-scale node scenarios, the probability of system errors increases greatly, and its performance declines significantly. Tendermint [35] merges view change with the normal process, reducing the communication complexity of view change. HotStuff [36] further reduces the communication complexity to linear. However, the network topology based on Leader nodes is prone to the single point of failure, which affects security and performance. Lyu et al. [37] proposed a Byzantine system with the tree network structure, which improved the scalability to a certain extent, but the application scenarios are limited. Li et al. [38] further improved the PBFT of the tree network structure and optimized the communication complexity. However, this scheme reduces the fault tolerance of consensus and has a higher delay.

Although the schemes proposed above have their advantages, they cannot significantly improve the scalability of PBFT. The communication complexity of most schemes is still related to the node scale of the entire network, which cannot support more nodes. H-PBFT decouples the communication complexity from the scale of ordinary nodes in the Mesh-and-Spoke network, achieving higher scalability and more suitable for application in large networks.

3 Model architecture

MANDALA is mainly composed of Mesh-and-Spoke network and H-PBFT consensus algorithm, and its overall architecture is shown in Fig. 1.

Poor scalability is a universal and outstanding problem in the current blockchain networks. Any error that occurs in consensus nodes will exert a great impact on the whole system. Therefore, the reliability of the nodes participating in consensus should be improved. In this model, the reliability of the nodes is evaluated by voting, and nodes are divided into three types: core nodes, shadow nodes, and ordinary nodes, all of which constitute the Mesh-and-Spoke network. Shadow nodes and core nodes form the core network, which runs the H-PBFT to generate blocks and reach consensus. Ordinary nodes form the ordinary network, which is responsible for sending transactions to the core network and receiving new blocks that have reached consensus.

H-PBFT divides the network-wide consensus tasks into two-layer subnets based on Mesh-and-Spoke network. It greatly reduces the communication complexity and efficiently solves the data consistency problem of nodes in the network.

4 Model design

4.1 Symbol definition

The symbols involved in this paper mainly focus on Mesh-and-Spoke network and H-PBFT consensus algorithm.

The symbols and meanings about the Mesh-and-Spoke network are shown in Table 1:

The symbols and meanings about H-PBFT are shown in Table 2.

4.2 Mesh-and-spoke network

4.2.1 Node type

In this paper, the nodes of the blockchain system are divided into three types by voting: ordinary nodes, shadow nodes, and core nodes. The logical relationship between different

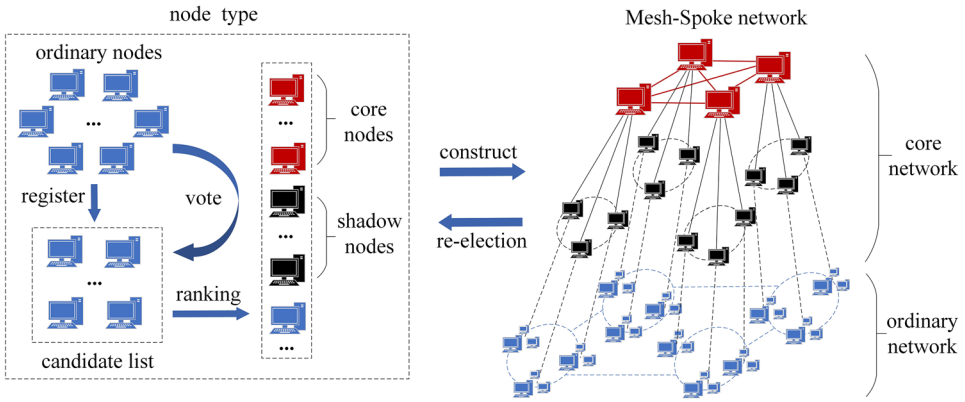


Fig. 1 Model architecture

Table 1 Symbols of Mesh-and-Spoke network

symbol	Description
N	The number of nodes in the core network
K	The number of nodes in the ordinary network
y	The number of core nodes
x	Number of shadow nodes in each core group
C_i	The core node numbered i
G_i	The core group numbered i
S_{ij}	The shadow node numbered j in G_i
N_{ij}	The ordinary node numbered j belonging to G_i
f_1	The upper limit of Byzantine nodes in core nodes
f_2	The upper limit of Byzantine nodes in a core group
l_i	List of shadow nodes in G_i

types of nodes is shown in Fig. 2, and the specific definitions and functions are as follows:

Ordinary nodes: In the Mesh-and-Spoke network, except for core nodes and shadow nodes, other nodes are ordinary nodes, which do not participate in the consensus, but need to accept the consensus results. When an ordinary node joins the Mesh-and-Spoke network, it first calculates the index i of the core group to be joined according to $i = y \bmod id$, where id is the unique identity number of the ordinary node. Then, it connects several shadow nodes in G_i (the specific number can still be adjusted). This method can ensure that the ordinary nodes are relatively evenly distributed in each core group and balance the network load. As shown in Fig. 3,

Table 2 Symbols of H-PBFT consensus algorithm

symbol	Description
$M_{REQUEST}$	Request message
$M_{PRE-PREPARE}$	Pre-prepare message
$M_{PREPARE}$	Prepare message
M_{COMMIT}	Commit message
M_{REPLY}	Reply message
$M_{VIEW-CHANGE}$	View change message
$M_{VIEW-CHANGE-ACK}$	View change acknowledgment message
$M_{NEW-VIEW}$	New view message
M_{UPDATE}	Update message
$< M >$	The signature of the message M
o	Committed transaction
t	Timestamp
c	The ID of the client
d	Block data
v	View number
n	The number assigned to the request
s	Summary data for d
e_i	Summary data for l_i
T	The time interval for generating blocks

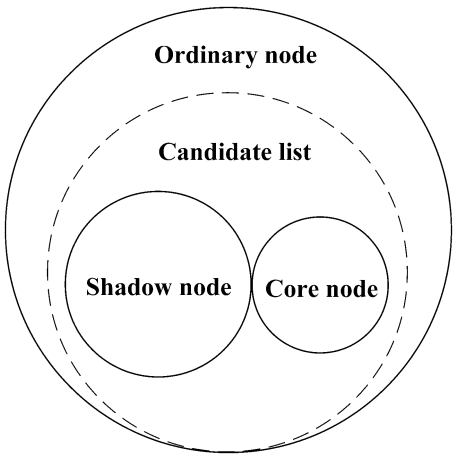


Fig. 2 Logical relationship between nodes

$\{N_{1i}|i = 1, 2, ..., k\}$ are ordinary nodes belonging to G_1. The main functions of ordinary nodes are as follows:

1. Submit the transaction to the corresponding core group.
2. After receiving the latest block, forward it to other ordinary nodes.

Shadow nodes: Each core node has x shadow nodes, which can communicate directly with that core node. As shown in Fig. 3, $\{S_{1i}|i = 1, 2, ..., x\}$ is the shadow nodes of the core node C_1. The main functions of shadow nodes are as follows:

1. Pass messages between core nodes and normal nodes. Shadow nodes collect transactions from ordinary nodes and forward them to core nodes, and also propagate blocks that have reached consensus in the core network to ordinary nodes.
2. Participate in the consensus and act as a candidate for the core node. During the consensus process, if a core node is a Byzantine node, which triggers the view-change protocol, it will be replaced by an eligible shadow node as the new core node.

Core nodes: The y nodes with the highest number of votes. Such as the node set $\{C_i|i = 1, 2, ..., y\}$ in Fig. 3. The main functions of the core node are as follows:

1. Receive and verify transactions from shadow nodes
2. Run H-PBFT consensus algorithm to generate blocks and reach consensus.

Core group: The core node and its corresponding shadow nodes constitute the core group, denoted as G_i $(i = 1, 2, ..., y)$ and i is the index of C_i. Core groups are

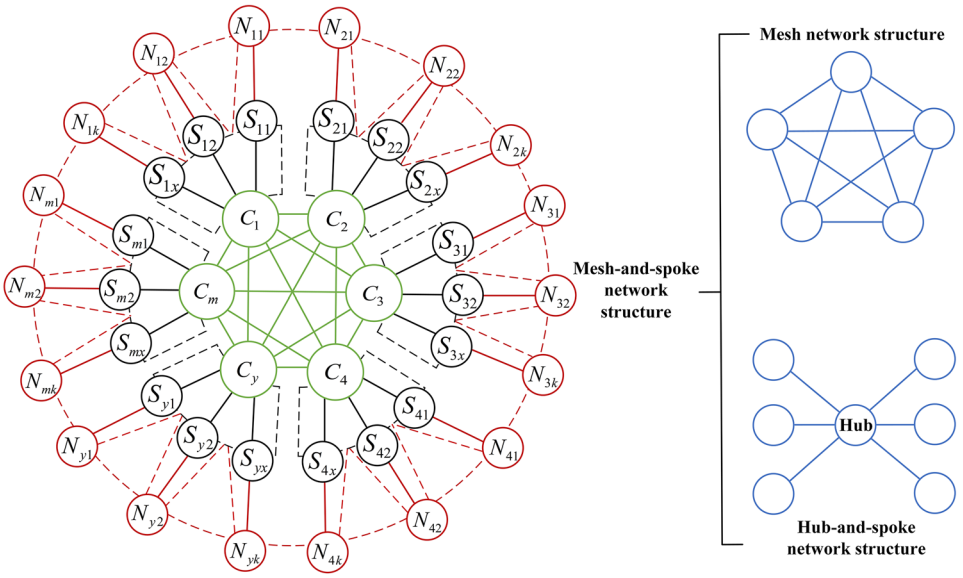


Fig. 3 Mesh-and-Spoke network structure, which resembles some images of mandala flower, hence the model's name

independent of each other and are not affected by other groups, which can achieve better privacy isolation. If a core group behaves abnormally, it will not affect other groups. In Fig. 3, the core group G_1 includes the core node C_1 and the shadow nodes $\{S_{1i}|i = 1, 2, ..., x\}$.

4.2.2 Structural design

The Mesh-and-Spoke network structure is a virtual logical structure and does not change the actual network structure. As shown in Fig. 3, the structure divides the blockchain network into two layers: the core network is composed of core nodes and shadow nodes, and the ordinary network is composed of ordinary nodes. In addition, we set up isolation mechanism, redundancy mechanism, reorganization mechanism, incentive and punishment mechanism. The definitions are as follows:

Core network: It consists of core groups, including core nodes and shadow nodes. The core nodes are connected to each other, and the shadow nodes are only connected to the nodes in the group. In Fig. 3, the core groups $\{G_i|i = 1, 2, ..., y\}$ form the core network, in which G_i includes C_i and $\{S_{ij}|j = 1, 2, ..., x\}$.

Ordinary network: It consists of ordinary nodes. The ordinary nodes connected to the shadow nodes and the original connection relationship between ordinary nodes constitute the ordinary network. In Fig. 3, the ordinary nodes $\{N_{1i}|i = 1, 2, ..., k\}$ of G_1 constitute a local ordinary network, and each local ordinary network constitutes the ordinary network according to the connectivity between ordinary nodes.

Isolation mechanism: Core nodes are isolated from ordinary nodes and cannot communicate with each other. These two types of nodes perform indirect data interaction through shadow nodes. For example, the core node C_1 can communicate with shadow nodes in G_1 and other core nodes, but it cannot communicate with ordinary nodes. This mechanism can prevent malicious ordinary nodes from directly attacking the core nodes, thereby ensuring the security of the core nodes.

• Redundancy mechanism: Mutual redundant connections between ordinary nodes and shadow nodes. For example, the ordinary node $N_{mj}(j = 1, 2, ..., k)$ connects $r(r \leq x)$ shadow nodes in $G_m(1 \leq m \leq y)$. This mechanism ensures stable data transmission between the core network and the ordinary network, and improves the degree of communication concurrency.

• Reorganization mechanism: To prevent the monopoly of core nodes, we have designed a reorganization mechanism. During each consensus period, nodes elected to the core network cannot withdraw from the candidate list. Other nodes can join the candidate list by paying a deposit, or they can withdraw and get their deposit back. Moreover, nodes can change their voting choices at any time, so the candidate list and the ranking of votes are dynamically changed. Before the start of each consensus epoch, the system redetermines the core nodes and shadow nodes according to the latest vote rankings in the candidate list, and then starts the consensus epoch. Under this mechanism, the core network is periodically reorganized and the nodes take turns to generate blocks, which improves security and fairness.

• Incentive and punishment mechanism: We give different degrees of rewards to the members of the core network and the nodes participating in the voting, so as to encourage more nodes to actively participate in the consensus and voting. If a member of the core network fails to complete the task, its deposit will be deducted, and it cannot join the candidate list in the next consensus epoch, which greatly increases the cost and difficulty of malicious acts. Most nodes in the network

are rational. They actively join the candidate list and vote to get more compensation and do not act maliciously. However, there are still a small number of malicious nodes, or abnormal nodes due to hardware errors, disconnection, etc. These nodes are called Byzantine nodes.

4.2.3 The process of mesh-and-spoke network formation

As shown in Fig. 4, nodes in the blockchain that are willing to participate in the consensus can join the candidate list by paying a deposit, and other nodes are ordinary nodes. Ordinary nodes vote on the nodes in the candidate list, ranking them according to the number of votes received. N nodes participate in the consensus, and we elect these nodes from the candidate list. Nodes ranked in the interval $[1, y]$ are elected as core nodes, and nodes ranked in the interval $[y+1, N]$ are elected as shadow nodes. Each core node is randomly assigned x shadow nodes, and the remaining nodes in the candidate list continue to be ordinary nodes. N is initially set to 10% of the total number of nodes in the candidate list, and satisfies $N = \lceil y + xy \rceil$ ($x \geq 3, y \geq 4$). Subsequently, nodes can make proposals and vote to adjust N.

After the node types are divided, the core network and ordinary network are formed according to the rules in Sect. 4.2.2. First, the core nodes form a mesh network. Then each core node and y shadow nodes form a hub-and-spoke network. The mesh network is used as a hub, and the corresponding shadow nodes are used as spoke nodes. It forms the core network, which is a Mesh-and-Spoke network structure. Similarly, the core network is similar to a hub, and ordinary nodes are similar to spoke nodes. In addition to connecting the corresponding shadow nodes, ordinary nodes also maintain the original connection relationship, which constitutes an ordinary network. Finally, the core network and the ordinary network constitute the final Mesh-and-Spoke network.

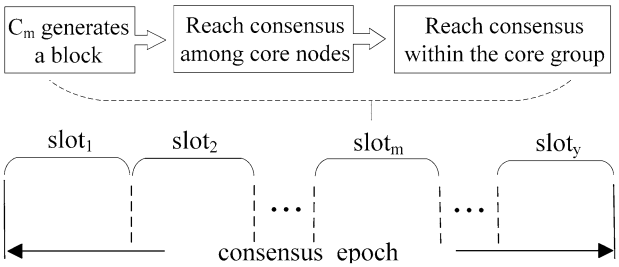


Fig. 5 Composition of a consensus epoch

4.3 H-PBFT consensus algorithm

PBFT has a great contribution to solving Byzantine Generals Problem [11], but its communication complexity up to $O(N^2)$ limits the efficiency of consensus. As the number of nodes increases, its network traffic grows rapidly, which causes great pressure on bandwidth. Furthermore, it is not compatible with the Mesh-and-Spoke network. To tackle the above problems, we propose the H-PBFT consensus algorithm, which reduces the communication complexity and improves the scalability, consensus efficiency, and fault tolerance of nodes.

4.3.1 Consensus epoch

The time when y core nodes take turns to generate blocks and reach a consensus in the core network is called *consensus epoch*, which is divided into y time intervals, and each interval is called a *slot*. As shown in Fig. 5, a block is generated by the core node during each slot period. After the block reaches a consensus at the core node layer and the inner layers of each core group, the slot ends. Until y slots are executed, the consensus epoch ends.

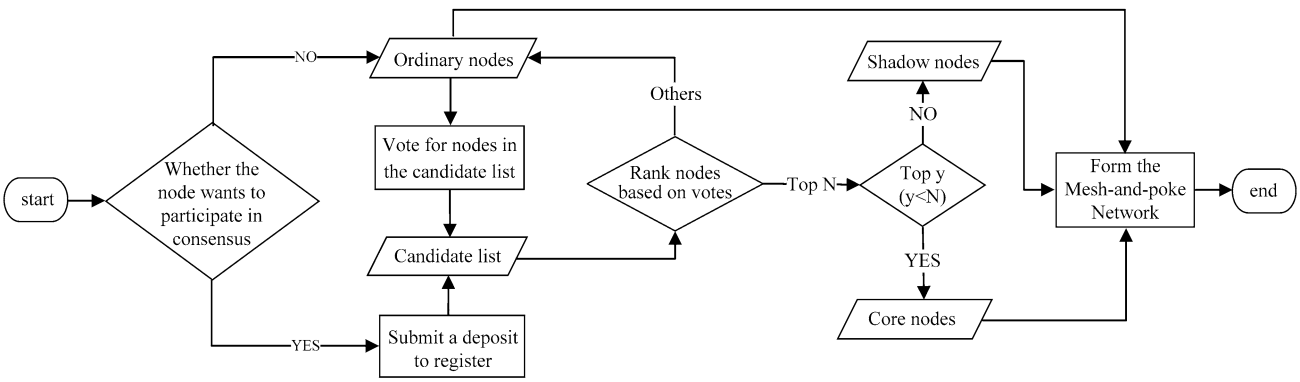


Fig. 4 The process of Mesh-and-Spoke network formation

4.3.2 Normal-case operation

H-PBFT is responsible for organizing various groups within the core network, so that nodes can reach a consensus on the block to be generated. During each slot period, a certain number of transactions will be packaged into the block by the primary node, and the correctness and consistency of the block will be guaranteed through a two-layer consensus protocol. As shown in Fig. 6, the consensus process of $slot_m(1 < m < y)$ in a consensus epoch is described:

1. **Core node layer protocol**

C_m acts as the primary node, $y-1$ core nodes $\{C_i | i = 1, 2, .., m-1, m+1, ..., y\}$ act as replica nodes, and the upper limit of the Byzantine node is $f_1(f_1 < \frac{y}{3})$. The process is as follows:

1. **Request:** As in the *Request* phase of the PBFT, the Client sends $M_{REQUEST}$ to C_m, and the message format is $< REQUEST, o, t, c >$.
2. **Pre-prepare$_c$:** C_m packs the correct transaction into a block and broadcasts $<< PRE-PREPARE, v, n, s, e_m >, d, l_m >$ to replicas.
3. **Prepare$_c$:** After receiving the $M_{PRE-PREPARE}$ message, C_i performs a validity check:

   Check whether the received digest s, e_m is consistent with the digest generated locally for d, l_m. Check whether v is consistent with the current view. Check if messages with the same d and n but different digests have been received before. Check that the transactions in the block are all valid.

   After the verification is passed, C_i saves d and l_m locally, and broadcasts $< PREPARE, v, n, s, e_i, i >$ to other core nodes, where i is the ID of C_i.
4. **Commit$_c$:** After the node receives $2f_1 + 1$ verified $M_{PREPARE}$ messages, it will enter the $commit_c$ phase, and then broadcast $< COMMIT, v, n, s, e_i, i >$ to the core nodes. Similarly, the node receives and verifies the message, and after receiving $2f_1 + 1$ verified M_{COMMIT} messages, it considers that the block has reached a consensus at the core node layer.

2. **Core group layer protocol**

After C_i receives $2f_1 + 1$ verified M_{COMMIT} messages, it determines that the core node layer has reached a consensus. G_i starts the consensus within the group. Taking the core group G_m in Fig. 6 as an example, the core node C_m is the primary, and the shadow nodes $\{S_{mj} | j = 1, 2, ..., x\}$ are the replicas. The view number in the group is v_m, and the upper limit of the Byzantine nodes within the group is $f_2(f_2 < \frac{x+1}{3})$. Each core group does consensus in parallel, and the consensus process within G_m is as follows:

1. **Pre-prepare$_s$:** C_m broadcasts $<< PRE-PREPARE, v_m, s, \varphi >, d >$ to the nodes in the core group, where φ is the $2f_1 + 1$ valid M_{COMMIT} messages received at the core node layer, which is the proof that the block has reached a consensus at the core node layer.
2. **Prepare$_s$:** Similar to the process in the $prepare_c$ phase, S_{mj} sends $< PREPARE, v_m, s, j >$ to other nodes in the core group.
3. **Commit$_s$:** A node in G_m enters the $commit_s$ phase after receiving $2f_2 + 1$ verified $M_{PREPARE}$ messages, and broadcasts $< COMMIT, v_m, s, j >$ in the core group. If the core node in G_m has received $2f_2 + 1$ verified M_{COMMIT} messages, it will determine that the received block has reached a consensus in G_m.

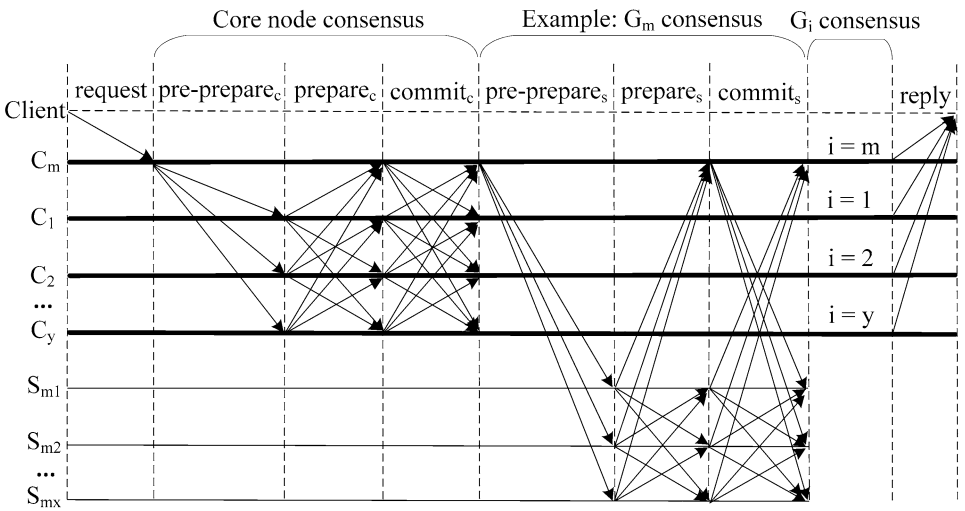


Fig. 6 Normal-Case Operation

When other core groups complete intra-group consensus according to the above process, the core nodes of each core group send M_{REPLY} messages to C_m. When C_m receives no less than $f_1 + 1$ consistent M_{REPLY}, the block generated by it reaches a consensus in the core network, and $slot_m$ finishes.

4.3.3 View change

The view-change protocol is responsible for confirming the new primary to provide liveness when the current primary fails. If the primary is a Byzantine node, the replica initiates a view change and re-selects a new primary. Since H-PBFT is divided into two layers, each layer has the possibility of view change, so it is divided into two cases for discussion. Taking the $slot_m(1 < m < y)$ period as an example, C_m acts as the primary, and the view change process is as follows.

- **View-change protocol at the core node layer**

The view number of the core node layer is v. When the replicas fail to reach a consensus after the time interval $2^{v+1} \cdot T$ or receive an erroneous block, a view change will be triggered. The original view v is still valid until the view change is complete, thus avoiding unnecessary view changes due to occasional network delays.

The new primary of the core node layer is denoted as p_c, which is generated from the shadow nodes in G_m, and the formula is $p_c = (v + 1) \bmod |x|$. The process is shown in Fig. 7:

1. **View-change$_c$:** When a view change occurs, the replica C_i will enter the new view $v+1$ and request to change the primary. It broadcasts $< VIEW - CHANGE, v + 1, h, \delta, P, i >$ to all nodes in the group, where h is the height of the latest stable checkpoint, δ is the set of checkpoints that it has executed, P is the set of messages that it has reached the *prepared* state, and i is its ID.
2. **View-change-ack$_c$:** The shadow nodes in the core group check each received $M_{VIEW-CHANGE}$ message, and if the verification passes, they will send $< VIEW-CHANGE-ACK, v + 1, i, j, u >$ to p_c. Among them, i is the ID of the node sending $M_{VIEW-CHANGE-ACK}$, j is the ID of the node sending $M_{VIEW-CHANGE}$ message, and u is the abstract of $M_{VIEW-CHANGE}$.
3. **New-view$_c$:** When p_c receives an $M_{VIEW-CHANGE}$ and $2f_2 - 1$ corresponding $M_{VIEW-CHANGE-ACK}$ messages, the $M_{VIEW-CHANGE}$ is valid. Until it receives $f_1 + 1$ valid $M_{VIEW-CHANGE}$ messages, which proves that most of the non-Byzantine nodes initiated the view change. Then, p_c broadcasts $< NEW - VIEW, v + 1, V, O >$ to other core nodes, where V contains $f_1 + 1$ $M_{VIEW-CHANGE}$ messages that have passed the verification, which is used to confirm the validity of this view change. O is the unfinished set of $M_{PREPARE}$ that it re-initiated.
4. **Update-member$_c$:** After the $new\text{-}view_c$ phase, p_c becomes the new core node in G_m. It broadcasts $< UPDATE, v_m, V, i >$ to other shadow nodes in the group to inform them that the core node has changed. Among them, v_m is used to keep the view in the group unchanged, V is used to prove the legitimacy of p_c as a new core node, and i is the ID of p_c.

So far, p_c becomes the new core node in G_m, and the original core node C_m is replaced with a shadow node.

- **View-change protocol within the core group**

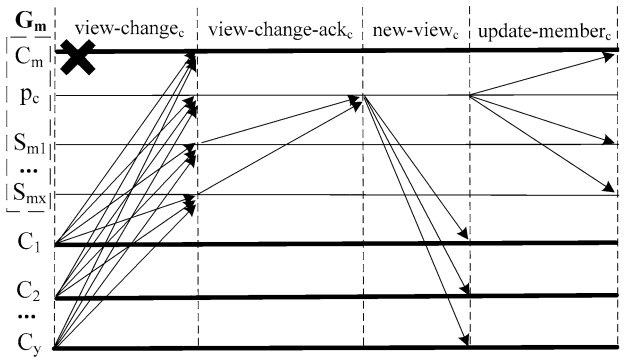


Fig. 7 View change process of the core node layer

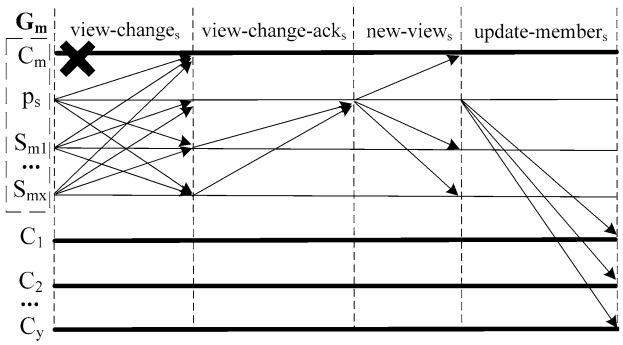


Fig. 8 View change process within the core group

When the replica S_{mi} fails to reach a consensus after the time interval $(2^{v+1} + 2^{v_m+1}) \cdot T$, or receives an erroneous block, it requests a view change within the core group. At this time, the view number in G_m is v_m, and the new primary is recorded as p_s, which is generated from the shadow nodes in G_m, and the calculation formula is $p_s = (v_m + 1) \bmod |x|$. As shown in Fig. 8, it is divided into *view-change*$_s$, *view-change-ack*$_s$, *new-view*$_s$, and *update-member*$_s$. The specific process is similar to the view change process of the core node layer, except that the first three phases are message interaction within the core group, and the last phase is to broadcast M_{UPDATE} to other core nodes, and the content of the specific message is replaced by the corresponding value within the core group.

5 Model construction

5.1 Building a mesh-and-spoke network

Before the start of each consensus epoch, the system determines the recommended value $N*$ of the number of core network nodes by voting. According to Formula (1), it is necessary to determine the optimal configuration of the number of shadow nodes x in each core group and the number of core nodes y, so that H-PBFT can achieve the minimum communication complexity. The formula is as follows (see Sect. 6.3.2 for the derivation process):

$$\begin{cases} N* = y(1+x), (x \geq 3, y \geq 4) \\ N* = \frac{(1+x)^3}{2}, (3 \leq x \leq \frac{N-3}{3}) \end{cases} \tag{1}$$

After obtaining the real solution $x*$ and $y*$ of Formula (1), the integers closest to the real solution are x and y. Then, according to the ranking of the votes in the candidate list, y core nodes and xy shadow nodes are determined to form a core network with $N(N = \lceil y + xy \rceil)$ nodes, and N is the integer value closest to $N*$. Other nodes are composed of the ordinary network. The construction of the Mesh-and-Spoke network is completed, and the consensus epoch begins.

5.2 Consensus in the core network

In the current consensus epoch, ordinary nodes send transactions to the shadow nodes, and then the shadow nodes collect transactions and forward them to the core nodes in the group. The core node verifies the transactions and broadcasts them to other core nodes. The consensus epoch consists of $\{slot_i | i = 1, 2, ..., y\}$. During the $slot_i$ period, the core node C_i acts as the primary. It generates blocks according to H-PBFT and reaches a consensus. First, a consensus is reached at the core node layer. The pseudocode is as follows:

Algorithm 1 Core node layer protocol

```
Input: M_REQUEST from Client
Output: slot_i Finished or Error
Broadcast ( M_PRE-PREPARE , CoreNodes)
RecievedM = Listen()
while Verify(RecievedM) == True then
    switch RecievedM do
        case M_PRE-PREPARE
            Broadcast ( M_PREPARE , CoreNodes)
        case M_PREPARE
            M_prepareSet.append(RecievedM)
            if M_prepareSet.size() ≥ 2f1 + 1 then
                Broadcast( M_COMMIT , CoreNodes)
            end if
        case M_COMMIT
            M_commitSet.append(RecievedM)
            if M_commitSet.size() ≥ 2f1 + 1 then
                execute Algorithm2
            end if
        case M_REPLY
            M_replySet.append(RecievedM)
            if M_replySet.size() ≥ f1 + 1 then
                output( slot_i ,Finished)
            end if
        otherwise
            output( slot_i ,Error)
    endsw
end while
```

When a core node receives more than $2f_1 + 1$ valid M_{COMMIT} messages, it can start the consensus within the group. The core node acts as the primary, and the shadow node within the group acts as the replica. The pseudocode is as follows:

Algorithm 2 Core group layer protocol

Input: M_{COMMIT} from core nodes
Output: M_{REPLY}
Broadcast ($M_{PRE-PREPARE}$, CoreGroup)
RecievedM = Listen()
while Verify(RecievedM) == True then
  switch RecievedM **do**
    case $M_{PRE-PREPARE}$
      Broadcast ($M_{PREPARE}$, CoreGroup)
    case $M_{PREPARE}$
      M_prepareSet.append(RecievedM)
      if M_prepareSet.size() $\geq 2f_2+1$ **then**
        Broadcast(M_{COMMIT} ,CoreGroup)
      end if
    case M_{COMMIT}
      M_commitSet.append(RecievedM)
      if M_commitSet.size() $\geq 2f_2+1$ **then**
        if node.type == core **then**
          Broadcast(M_{REPLY} , C_i)
        else if node.type == shadow **then**
          execute Algorithm3
        break
      end if
  endsw
end while

$slot_i$ finishes when C_i receives no less than f_1+1 M_{REPLY} messages. $slot_{i+1}$ is then executed. If the primary is identified as a Byzantine node during the $slot_i$, it will be replaced according to the view-change protocol of the corresponding level. The unfinished consensus process will then move on. Finally, y slots are executed according to the above algorithm, and the current consensus epoch is completed.

5.3 Synchronize data to the ordinary network

After the core network reaches a consensus during each slot period, the shadow nodes synchronize the block to the ordinary network. Since the Gossip algorithm [39] has many advantages such as simplicity, efficiency, and strong fault tolerance, it is used for data synchronization. The main objective here is to achieve synchronization with ordinary nodes, therefore the *push* mode of the Gossip algorithm is adopted. Taking the shadow node in G_i to synchronize data to the ordinary node as an example, the pseudocode is as follows:

Algorithm 3 Synchronized Data Algorithm

Input: d
Output: M_d
foreach node $\in G_i$ **do**
  if Select(node, P) == True **then**
    AddMessage(d , M_d)
    Broadcast(M_d , OrdinaryNodes)
  end if
end

d is the block data, and M_d the message containing the block. $P=\frac{r}{K}$, r is the number of target sending nodes, which is a configurable constant, and K is the total number of ordinary nodes belonging to G_i. When an ordinary node obtains the latest block, the above steps are repeated, and other ordinary nodes are selected with the probability P to send messages. The messages will continuously spread until the ordinary network receives the message, and the entire network data synchronization is completed.

6 Experiment and analysis

6.1 Experimental design

The model is tested and evaluated from the aspects of throughput, delay, communication times, effective transmission rate, and security by combining theoretical proof and simulation experiment. This experiment is built on the open-source blockchain project Hyperledger Fabric [40], which is a modular and extensible open-source system. We use Docker technology to simulate a multi-node network environment, which runs distributed on multiple hosts, and their parameter configurations are shown in Table 3.

6.2 Performance analysis

6.2.1 Throughput and latency

Throughput and latency are important criteria for measuring blockchain performance. Throughput refers to the number of

Table 3 Host parameters

Number of hosts	Operating system	CPU	RAM
2	Linux- Ubuntu 18.04	AMD Ryzen 7 3700X 8-Core	32G
1	Linux- Ubuntu 16.04	Intel(R) Core(TM) i7-9750H	32G

transactions that the blockchain system can process in unit time, and latency refers to the time it takes for a transaction to be submitted to completion. In this paper, under the same network environment, 16 nodes participating in the consensus are set up, the total number of transactions is fixed at 10,000, and the transaction rate (the number of transactions sent per second) is 500 ~ 2500. Comparing this model with Fabric, the throughput and latency results are shown in Fig. 9.

In terms of throughput, with the increase of transaction rate, the throughput of Fabric and MANDALA both show an upward trend within the processing capacity of the node, but the throughput of this model is higher. Continuing to increase the transaction rate, Fabric reaches the bottleneck first, and then the throughput decreases significantly, while the throughput of our model keeps growing, reaching more than 1000 tps. Therefore, MANDALA can maintain a high level of throughput and exhibit better scalability.

In terms of latency, the latency of Fabric increases sharply with the increase of transaction rate, which is caused by the high communication complexity. As the transaction rate increases, so does the number of messages that nodes broadcast and process, which takes a lot of time. MANDALA has better communication complexity and greatly reduces communication times. Although its hierarchical and multi-stage consensus makes the early latency higher than that of Fabric, the latency remains basically stable as the number of transactions increases. Therefore, MANDALA has better scalability and is more suitable for large-scale networks.

6.2.2 Communication times

The communication times represent the traffic generated by nodes in the process of consensus [41]. The main problem of PBFT and its derived consensus mechanism is the high communication complexity, which will reduce the efficiency of the system. Our main goal in designing H-PBFT is to reduce communication complexity. The communication times of the consensus process and view change are analyzed below.

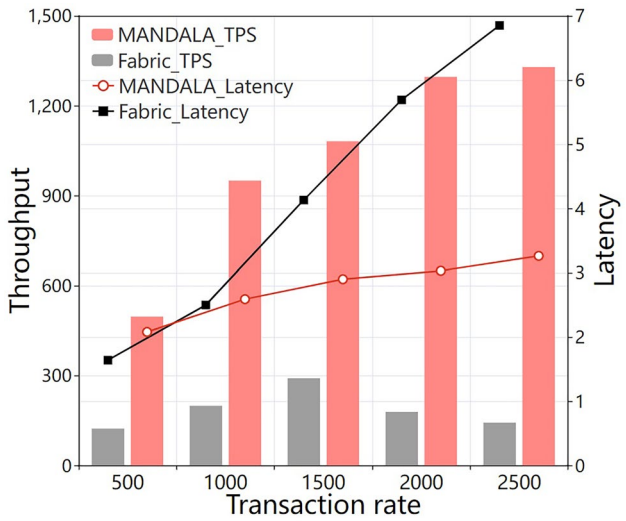


Fig. 9 Throughput and Latency

- **Communication times in the consensus process**

According to the design in Sect. 4.3.2, H-PBFT has two phases. The first phase is for y core nodes to achieve consensus, and the communication complexity is y^2, denoted as C_1. In the second phase, the consensus is carried out for $1+x$ nodes in each core group, and a total of y core groups execute this process in parallel. The communication complexity of this part is $y(1+x)^2$, denoted as C_2. Then the total communication complexity of the two phases is $C_1 + C_2 = y^2 + y(1+x)^2$, denoted as C. And the relationship between N,y,x satisfies $N = y(1+x)(x \geq 3, y \geq 4)$ and the communication complexity C is simplified to get Formula (2):

$$C = \frac{N^2}{1+x^2+2x} + Nx + N, (x \geq 3) \tag{2}$$

where N is a constant, so the dependent variable is C and the independent variable is x. The first and second derivatives of Formula (2) are calculated as follows:

$$\begin{cases} C' = N - \frac{2N^2}{(1+x)^3} \\ C'' = \frac{6N^2}{(1+x)^4} \end{cases} \tag{3}$$

According to Formula (3), the second-order derivative C'' is always greater than 0, so the first-order derivative C' is an increasing function and has a zero point, at which the communication complexity C can reach the minimum value. Let $C' = 0$ and simplify to get Formula (4):

$$N = \frac{(1+x)^3}{2}, \ (3 \leq x \leq \frac{N-3}{3}) \tag{4}$$

When x is the integer closest to the real root of Formula (4), the minimum communication complexity can be obtained. Then calculate y according to $N = y(1+x)$, and the core network with the minimum communication complexity can be determined. If x and y have reached the optimal configuration, calculate the ratio [38] of C and N from Formulas (2) and (4), and simplify to obtain Formula (5):

$$C = \frac{3\sqrt[3]{2}}{2} \cdot N^{\frac{4}{3}} \tag{5}$$

$\frac{3\sqrt[3]{2}}{2}$ is a constant, so when x and y reaches the optimal configuration, the communication complexity of H-PBFT is $O(N^{\frac{4}{3}})$. To evaluate the performance of H-PBFT, we simulate different numbers of nodes participating in the

consensus. Figure 10 shows the communication times of PBFT [17], Tendermint [34], Tree structure PBFT [38] and H-PBFT. PBFT is the most famous practical BFT algorithm, Tendermint is a typical simplified process and scalable BFT algorithm, Tree structure PBFT is an efficient BFT algorithm based on tree network structure.

As shown in Fig. 10, as the number of nodes continues to increase, PBFT and Tendermint need to complete the consensus through the communication volume of complexity $O(N^2)$, which limits the scalability of the node scale. When the number of nodes is too large, PBFT cannot be practically deployed due to the excessive communication overhead. The main goal of Tree structure PBFT and H-PBFT is to reduce the communication complexity. They both significantly reduce the network traffic, while H-PBFT further optimizes the communication complexity and generates fewer communications. In addition, the communication overhead of H-PBFT has nothing to do with the scale of ordinary nodes. As long as the number of nodes in the core network remains relatively stable, the consensus efficiency of H-PBFT will not decrease rapidly with the expansion of the node scale. Therefore, H-PBFT can be applied to scenarios with a larger node scale, effectively improving the scalability.

- **Communication times during view change**

The view-change protocol of H-PBFT is divided into two cases, which are independent of each other, and their communication times are discussed separately.

The view change occurs at the core node layer. The first three phases are similar to the view change process of PBFT, and the communication complexity is $O(xy^2)$. When the new primary is elected, it also goes through the $update\text{-}member_c$ phase, which will bring additional traffic, and the communication complexity of this part is $O(x)$. Therefore, the communication complexity of the view-change protocol at the core node layer is $O(xy^2 + x)$, which is represented by H-PBFT_CN in Fig. 11.

When the view-change protocol is triggered within the core group, the specific process is still similar to PBFT, and the communication complexity of this part is $O(x^3)$. When a new primary is elected, similarly, the $update\text{-}member_s$ phase has to go through, and the communication complexity is $O(y)$. Therefore, the communication complexity of the view-change protocol within a core group is $O(x^3 + y)$. In the worst case, view changes occur in all core groups, then the maximum communication complexity is $O(x^3y + y^2)$, which is represented by H-PBFT_CG in Fig. 11.

Figure 11 shows the number of communications generated during view change for PBFT, Tendermint, Tree structure PBFT, and H-PBFT.

As shown in Fig. 11, under the condition of the same number of nodes, the communication times during the PBFT view change process are the most. Tendermint merges the view change with the normal process, reducing the number of communications to the order of $O(N^2)$, and the traffic of Tree structure PBFT is slightly lower than it. The communication times of H-PBFT are significantly lower than those algorithms, and it has stronger robustness. When the primary is abnormal, H-PBFT can change it at a lower cost, and the consensus process will not be blocked for a long time, which ensures that the system remains liveness.

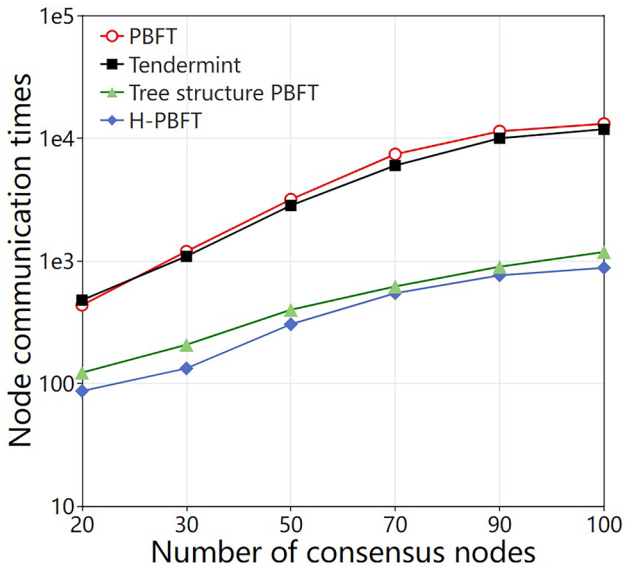


Fig. 10 Node communication times for consensus process

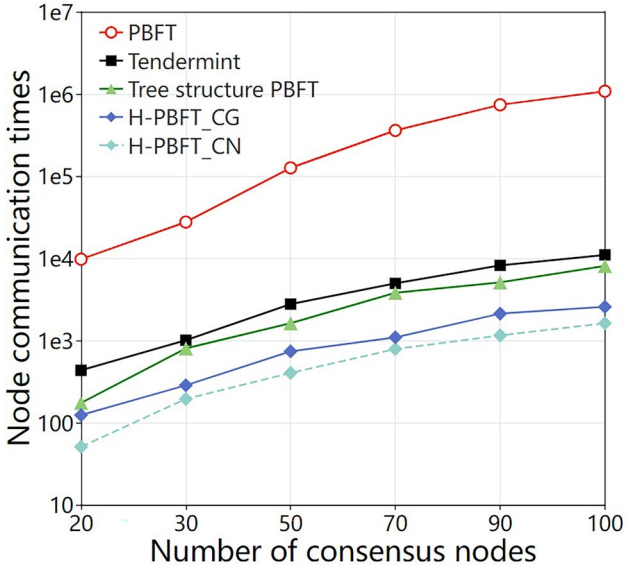


Fig. 11 Node communication times for view change

6.2.3 Effective transmission rate

The efficiency of a Mesh-and-Spoke network can be measured by the effective transmission rate (ETR) of the P2P network [10]. Valid transmission refers to transmitting data to a node that does not have the data, and sending data to a node that has received the data is an invalid transmission. The effective transmission rate is the percentage of effective transmissions to the total transmissions.

The total number of nodes in the network is TN. Assuming that x and y in the Mesh-and-Spoke network satisfy the optimal configuration, the number of transmissions required for the block to be transmitted from the core node to all nodes is $F = N^{\frac{4}{3}} + K \cdot r$, where the communication times of the core network are about $\left\lceil N^{\frac{4}{3}} \right\rceil$, and r is the number of redundant transmission nodes. The commonly used network structures in the blockchain are unstructured networks, structured networks, and hybrid networks, and the representative projects are Bitcoin [1], Ethereum [13], and EOS [28], and their nodes send messages to all neighboring nodes without restriction [10].

In the ideal case of data transmission, the node only needs to send data to the target node once, and the target node can receive the data without additional transmission. The number of adjacent nodes of each node is 10% of the total number of nodes. The number of nodes in the core network is 10% of the total number of nodes, and the number of redundant transmission nodes $r = 3$. The transmission times and effective transmission rates of the two networks are shown in Table 4:

As shown in Table 4, we compare the number of transmissions and ETR of different network structures. With the increase in the number of nodes, the transmission times of those unoptimized P2P networks increase rapidly, and their ETR is lower than 7% and is still decreasing. The number of transmissions in the Mesh-and-Spoke network is stable, and the ETR is maintained at 30%. With the same number of nodes, the transmission times of the Mesh-and-Spoke network are reduced by 90%, and the average ETR is increased by 30%. The results show that the Mesh-and-Spoke network reduces the number of transmissions and improves the efficiency of transmission.

6.3 Security analysis

The security of this model is mainly analyzed from two aspects. The first aspect is to prove the fault tolerance and liveness of H-PBFT, which is the ability of the model to maintain the normal consensus under abnormal conditions. The second is robustness, which refers to the ability of the model to resist attacks from malicious nodes and to recover after being attacked.

6.3.1 Fault tolerance

The proportion of Byzantine nodes in consensus nodes is the factor that has the greatest impact on security. The number of malicious nodes that a consensus algorithm can tolerate is called fault tolerance. Once the upper limit of fault tolerance is exceeded, consensus cannot be reached.

In the consensus process of this model, the upper limit of Byzantine nodes at the core node layer is $f_1 (f_1 < \frac{y}{3})$, and the upper limit of Byzantine nodes per core group is f_2 $(f_2 < \frac{x+1}{3})$. The upper limit of Byzantine nodes for the entire core network is $f * = f_1 + yf_2 (f * < \frac{N+y}{3})$. In the case of the same number of nodes N, the upper limit of Byzantine nodes of PBFT is $f (f < \frac{N}{3})$.

In a special case, if Byzantine nodes are all distributed in each core group, the upper limit of Byzantine nodes of this model degenerates to f, which is the same as the fault tolerance of PBFT. If the Byzantine nodes are all distributed in core nodes, the view-change protocol will be triggered, which will re-determine the new core nodes and execute the consensus process again. Therefore, this situation has no decisive impact on the success of the consensus, and the maximum fault tolerance of H-PBFT remains unchanged.

As shown in Table 5, as the number of nodes participating in the consensus increases, the fault tolerance upper limit of both consensus algorithms increases, but H-PBFT is always higher than PBFT. Moreover, according to the incentive and punishment mechanism, rational nodes usually act honestly,

Table 4 Network transmission efficiency

	Number of transmissions				ETR			
TN	Mesh-and-Spoke	unstructured	structured	hybrid	Mesh-and-Spoke	unstructured	structured	hybrid
200	594	4298	3231	3664	33.5%	4.6%	6.1%	5.4%
300	903	9656	7658	8379	33.1%	3.1%	3.9%	3.5%
400	1216	17,367	10,388	14,886	32.8%	2.3%	3.8%	2.7%
500	1534	26,428	16,640	20,975	32.5%	1.9%	2.9%	2.4%
600	1855	38,950	27,211	33,487	32.3%	1.5%	2.2%	1.8%

they are stable in most cases and do not behave maliciously, so the probability of Byzantine nodes in the core network is very low. Based on the analysis of these two aspects, it can be concluded that the fault tolerance of this model is higher than that of PBFT.

6.3.2 Liveness

The liveness of BFT consensus means that when the primary cannot execute the request normally, or the replica detects that the primary has errors, the system can change to a new and stable view, and a consensus can still be reached in the end [34]. Let's analyze the activity of H-PBFT consensus:

Theorem 1 After the view-change protocol is triggered during the consensus process, if at least $f + 1$ replicas request the view change, then v will be changed to the stable view. (f is either f_1 or f_2, depending on which layer the view change happens).

Proof of theorem 1 Assuming that at least $f + 1$ correct replicas request view changes, there are two cases.

1. The replica receives at least $f + 1$ valid $M_{VIEW-CHANGE}$, and elects a new primary, other replicas all receive valid $M_{NEW-VIEW}$ messages, and all nodes in this layer successfully change to the new view.
2. The replica receives less than $f + 1$ valid $M_{VIEW-CHANGE}$ messages. In this case, the view change mechanism for this round is stalled, waiting for other $M_{VIEW-CHANGE}$. Due to the timeout retransmission mechanism, other replicas will retransmit $M_{VIEW-CHANGE}$. Under the weak synchronization assumption, the message is guaranteed to be delivered in polynomial time. Therefore, the system will eventually reach case (1), which is a stable view.

Theorem 2 *Within each slot, the blocks generated by the correct primary will eventually reach a consensus.*

Table 5 Fault tolerance

Number of nodes	Upper limit of Byzantine nodes	
	PBFT	H-PBFT
20	6	8
30	10	12
50	15	20
70	22	28
90	29	35
100	33	39

Proof of theorem 2 In the stable view, blocks generated by the correct primary will reach a consensus. If the view is unstable, there are two situations:

1. At least $f + 1$ correct replicas request a view change. The view will eventually become stable. (Theorem 1)
2. Fewer than $f + 1$ correct replicas request a view change. If the view change threshold is not met, all replicas ignore the view change request and proceed with the consensus process, leaving the view unchanged. Otherwise, the consensus is not completed within the specified time, and all replicas will request the view change, returning to case (1).

According to Theorem 1 and Theorem 2, all replicas will eventually enter a stable view, and the block generated by the correct primary will eventually reach a consensus, so H-PBFT can guarantee liveness.

6.3.3 Adaptive attack

Models that use a BFT-like consensus mechanism expose the identity of the primary, which is vulnerable to attacks such as distributed denial of service (DDoS) that disrupt the consensus process, called adaptive attacks [42]. Therefore, in the above model, after the attacker knows the current primary, the consensus can be easily disrupted with an adaptive attack.

MANDALA adopts a Mesh-and-Spoke network structure, and the probability of Byzantine nodes in the core network is very low. If a primary is found to act maliciously, it will be replaced by the view change mechanism. As a result, the consensus progress is not much affected. However, the nodes in the ordinary network can join or withdraw at will, which has low stability and a high probability of malicious nodes. Therefore, the risks of adaptive attacks should be one of the main concerns only when ordinary nodes attack the primary. The isolation mechanism cuts off the path for attackers to directly attack the core node. Malicious ordinary nodes cannot directly attack the core node, but can only attack its shadow nodes.

The probability of a single consensus node being offline in the network is p, and the probability that $r(r \leq x)$ shadow nodes in a core group are offline in a grouped network structure is $p*$. The calculation formula [43] is shown in Formula (6):

$$p* = \binom{x}{r} p^r (1-p)^{x-r} \tag{6}$$

When the offline probability of a node is 0.5, 0.4, 0.3, 0.1 [44], we analyze the core group with 10, 12, 14 nodes, and the probability of r nodes in the group being attacked offline is shown in Fig. 12:

Overall, the probability that r nodes in the core group are offline is lower than the probability that a single node is offline under the same conditions, and as r increases, the probability that they are all offline decreases rapidly. Even in the worst case, all nodes in the core group are maliciously attacked and become offline, which will not affect the consensus. The core node can still generate blocks at the core node layer, and then propagate to the ordinary network through other core nodes and shadow nodes, and finally reach a consensus of the whole network, so it is difficult for adaptive attacks to cause a consensus failure of the Mesh-and-Spoke network.

6.3.4 Byzantine node recovery

If the nodes in the model passively become Byzantine nodes due to disconnection, delay, etc., some correct block data will be missing. If it wants to synchronize missing data after returning to normal, it can request data from other normal nodes in the core group. However, the node was in an abnormal state before, and it cannot determine which nodes are normal within the group, therefore preventing it from obtaining correct data. We propose a solution:

The node randomly selects r nodes from the core group and requests M_{COMMIT} in the latest ended slot. If the node receives no less than $\lceil r \cdot p \rceil (p \geq \frac{2}{3})$ consistent M_{COMMIT} messages, it trusts these nodes and requests its missing block data from them. The following analyzes how to determine r to ensure that more than $\lceil r \cdot p \rceil$ consistent messages are received.

Assuming that there are X nodes in the core group except this node, there are at most $f(f < \frac{X}{3})$ Byzantine nodes. Then the calculation formula [45] of the probability PR that the node can get the correct result is shown in Formula (7):

$$PR = \frac{\sum_{i=pr}^{r} C_{X-f}^{i} C_{f}^{r-i}}{C_{X}^{r}} \tag{7}$$

For quantitative analysis, X is taken as 9, 12, 15, 18, and f is 2, 3, 4, 5. The results are shown in Fig. 13:

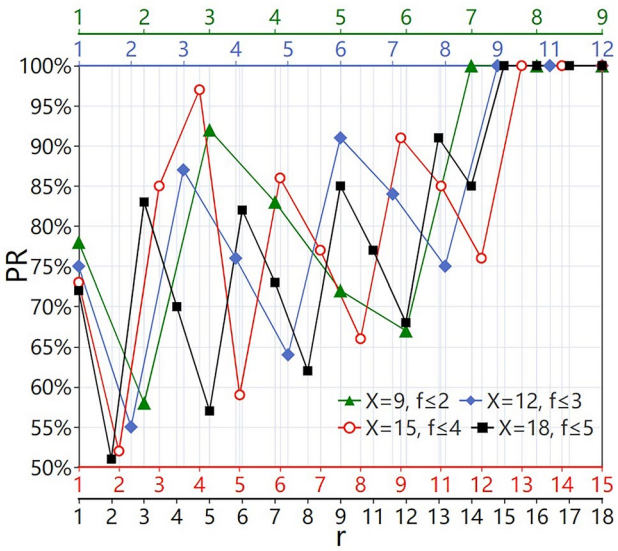


Fig. 13 Probability of correct result

As can be seen from Fig. 13, with the increase of node number r, probability PR generally presents an upward trend. Moreover, the node can receive more than $\lceil r \cdot p \rceil$ consistent messages without requesting all nodes in the group. It requests at most 73% of the nodes in the group to make the probability PR reach 100%, which reduces the traffic by about 30%. That is, if the node requests M_{COMMIT} messages from $\lceil 0.73 \cdot X \rceil$ nodes in the core group, it can receive more than $\lceil r \cdot p \rceil$ consistent replies. Then, the node can request the correct data from these nodes and continue to participate in the subsequent consensus process.

6.4 Scheme comparison

Table 6 lists the comparison of this model with other blockchain optimization schemes.

The model [17] with PBFT consensus generally uses mesh networks, which have lower performance and scalability. The BFT-DPoS consensus model [30] based on the hybrid network improves the throughput, but the

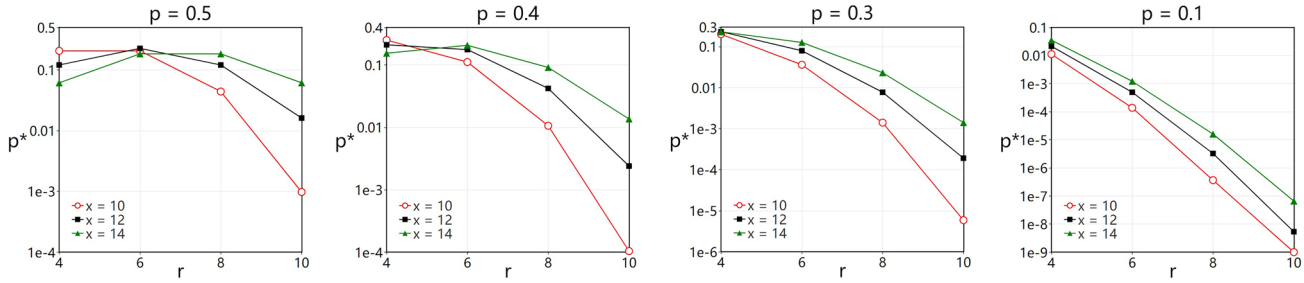


Fig. 12 Offline probability of nodes in the core group

Table 6 Comparison of different blockchain optimization models

Blockchain model							
Consensus	Network structure	Fault tolerance	Communication complexity	Average ETR	Latency	Scalability	Applicable scene
PBFT	Mesh	f	$O(N^2)$	Low	Small scale network: Low Large scale network: High	Low	Permissioned
BFT-DPoS	Hybrid	f	$O(N^2)$	Low	Low	Low	Permissionless
Tendermint	Mesh	f	$O(N^2)$	Low	Medium	Medium	Permissioned
Hotstuff	Star	f	$O(N)$	High	Medium	Medium	Permissioned
Tree structure PBFT	Tree	$< f$	$O(N^{\frac{4}{3}})$	Low	High	High	Permissioned & Permissionless
H-PBFT	Mesh-and-Spoke	$f + \frac{y}{N}f$	$O(N^{\frac{4}{3}})$	High	Medium	High	Permissioned & Permissionless

performance and scalability don't improve much. The model [35] that adopts Tendermint as the consensus mainly makes improvements to PBFT, which simplifies the view change process. However, it waits for a large network delay to ensure liveness, and the application scenario is still limited to the permission chain only. The above three schemes are only improved at the consensus, and the unoptimized P2P network structure is used, resulting in a low effective transmission rate of the network, which greatly restricts the overall performance of the blockchain model.

Tree structure PBFT model [38] improves PBFT on the basis of tree network structure. But it has the disadvantage of communication delay and lower security. As the number of nodes increases, the latency becomes larger, which affects its performance in large-scale networks. Although the process of H-PBFT goes through multiple layers, the overall time delay has increased. With the expansion of the node scale, the delay of MANDALA remains stable, which indicates that it has better scalability. Moreover, MANDALA has lower communication times and stronger fault tolerance. The model [36] with HotStuff and star network structure [46] achieves linear communication complexity based on threshold signature. However, its network structure increases the risk of centralization. As the scale of the network expands, the leader node will broadcast a large number of messages. In a weak network environment, it is prone to a single point of failure and has poor stability, resulting in high node change overhead. It is only suitable for permissioned chains with a good network environment. Therefore, Its application scenarios are fewer. Compared with HotStuff, MANDALA improves the scalability and robustness, and the application scenarios are more extensive.

Different from those models, MANDALA improves the scalability of nodes, and its application scenarios are more extensive, which can be applied in both permissionless and permissioned chains. For permissionless chain scenarios, it is also called public blockchain, which is characterized by large node scale and a complex network environment. MANDALA can be well adapted to this environment, it has good scalability, can be applied to large-scale networks, and its communication overhead is lower, and it can maintain good communication performance. For the permissioned chain, it mainly refers to the consortium blockchain, which has a relatively small number of nodes and a good network environment. In this environment, MANDALA can give full play to its performance advantages, greatly reducing the communication overhead and transmission times, and has better performance and security. MANDALA is not only widely used in the field of blockchain, but also can be applied to other fields such as Software Defined Network (SDN), Internet of Things (IoT), etc. They are essentially distributed networks, and the network structure and consensus algorithm of the MANDALA can be applied to improve performance and scalability.

7 Conclusion and future work

This paper proposes a scalable blockchain model called MANDALA, which can be used in various blockchain scenarios such as large-scale permissionless or permissioned chains. The model uses the H-PBFT algorithm to achieve consensus based on the proposed Mesh-and-Spoke network. Simulation results show that this model can reduce

communication overhead, and improve the throughput and transmission efficiency. Moreover, it decouples the consensus efficiency from the scale of ordinary nodes, which effectively improves the scalability of the blockchain. Without reducing security, the robustness of the consensus algorithm is ensured and the ability of the model to resist malicious attacks can be guaranteed. Compared with other blockchain optimization schemes, MANDALA has better performance, security and scalability.

Future work includes the following directions: (1) Aggregated signature technology can be adopted to further reduce the communication complexity of the Hotstuff scheme and simplify the consensus process. (2) An improved data synchronization algorithm will be developed to enhance transmission efficiency.

Funding The work is supported by the National Key R&D Program of China (No. 2021YFB2700800).

Declarations

Conflict of interest We declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper. We declare that there is no financial interest/personal relationship which may be considered as potential competing interests.

References

1. Nakamoto S (2008) Bitcoin: A peer-to-peer electronic cash system. Decent Bus Rev 21260
2. Salah K, Rehman MHU, Nizamuddin N, Al-Fuqaha A (2019) Blockchain for AI: Review and open research challenges. IEEE Access 7:10127–10149
3. Bandara E, Tosh D, Foytik P, Shetty S, Ranasinghe N, De Zoysa K (2021) Tikiri—Towards a lightweight blockchain for IoT. Futur Gener Comput Syst 119:154–165
4. Chowdhury MJM, Colman A, Kabir MA, Han J, Sarda P (2018) Blockchain as a notarization service for data sharing with personal data store. In 2018 17th IEEE International Conference on Trust, Security and Privacy in Computing and Communications/12th IEEE International Conference on Big Data Science and Engineering (TrustCom/BigDataSE), pp. 1330–1335. IEEE
5. Xia Q, Sifah EB, Asamoah KO, Gao J, Du X, Guizani M (2017) Medshare: Trust-less medical data sharing among cloud service providers via blockchain. IEEE Access 5:14757–14767
6. Li C, Palanisamy B (2019) Incentivized blockchain-based social media platforms: A case study of steemit. In Proceedings of the 10th ACM Conference on Web Science, pp. 145–154
7. Nasir MH, Arshad J, Khan MM, Fatima M, Salah K, Jayaraman R (2022) Scalable blockchains—A systematic review. Futur Gener Comput Syst 126:136–162
8. Xie J, Yu FR, Huang T, Xie R, Liu J, Liu Y (2019) A survey on the scalability of blockchain systems. IEEE Network 33(5):166–173
9. Wang R, Ye K, Meng T, Xu C-Z (2020) Performance Evaluation on Blockchain Systems: A Case Study on Ethereum, Fabric, Sawtooth and Fisco-Bcos. In: International Conference on Services Computing, pp. 120–134. Springer
10. Yu B, Li X, Zhao H, Zhou T (2021) A scalable blockchain network model with transmission paths and neighbor node subareas. Computing 1–25
11. Lamport L, Shostak R, Pease M (2019) The Byzantine generals problem. In Concurrency: The Works of Leslie Lamport, pp. 203–226
12. Li J (2018) Data transmission scheme considering node failure for blockchain. Wireless Pers Commun 103(1):179–194
13. Wood G et al (2014) Ethereum: A secure decentralised generalised transaction ledger. Ethereum Project Yellow Paper 151(2014):1–32
14. Cherbal S, Boukerram A, Boubetra A (2016) A survey of dht solutions in fixed and mobile networks. Int J Commun Netw Distrib Syst 17(1):14–42
15. Meneghetti A, Sala M, Taufer D (2020) A survey on pow-based consensus. Ann Emerg Technol Comput (AETiC), Print ISSN 2516–0281
16. Saleh F (2021) Blockchain without waste: Proof-of-stake. Rev Financ Stud 34(3):1156–1190
17. Castro M, Liskov B et al (1999) Practical byzantine fault tolerance. In OsDI 99:173–186
18. Cao B, Zhang Z, Feng D, Zhang S, Zhang L, Peng M, Li Y (2020) Performance analysis and comparison of PoW, PoS and DAG based blockchains. Digit Commun Netw 6(4):480–485
19. Fu X, Wang H, Shi P (2021) A survey of blockchain consensus algorithms: mechanism, design and applications. Sci China Inf Sci 64(2):1–15
20. Li W, He M (2021) EBFT: A hierarchical and group-based byzantine fault tolerant consensus algorithm. IEEE Int Conf Softw Eng Serv Sci (ICSESS) 32–37. IEEE
21. Carlsson B, Gustavsson R (2001) The rise and fall of napster-an evolutionary approach. Int Comput Sci Conf Active Media Technol 347–354. Springer
22. Perry B, Ousterhout J, Balakrishnan A, Shah H, Fugal D, Fastpass H (2014) A centralized" zero-queue" datacenter network. Proc ACM Conf SIGCOMM 307–318
23. Hwang J-H, Kim N-P, Ji Y-H, Ahn T-H (2010) Ims centralized network architecture towards convergence services. Int Conf Syst Netw Commun 283–288. IEEE
24. Tian C, Yang B (2011) R2trust, a reputation and risk based trust management framework for large-scale, fully decentralized overlay networks. Futur Gener Comput Syst 27(8):1135–1141
25. Aebeloe C, Montoya G, Hose K (2021) Colchain: Collaborative linked data networks. Proc Web Conf 1385–1396
26. Kamel MB, Crispo B, Ligeti P (2019) A decentralized and scalable model for resource discovery in iot network. Int Conf Wirel Mob Comput Netw Commun (WiMob) 1–4. IEEE
27. Jiang Y, Kouzoupis D, Yin H, Diehl M, Houska B (2021) Decentralized optimization over tree graphs. J Optim Theory Appl 189(2):384–407
28. Xu B, Luthra D, Cole Z, Blakely N (2018) Eos: An architectural, performance, and economic analysis. Retrieved 11 Jun 2019
29. Van Toan N, Park U, Ryu G (2018) Rcane: Semi-centralized network of parallel blockchain and apos. IEEE Int Conf Parall Distrib Syst (ICPADS) 1–6. IEEE
30. Lee D, Lee DH (2019) Push and pull: Manipulating a production schedule and maximizing rewards on the eosio blockchain. In Proceedings of the Third ACM Workshop on Blockchains, Cryptocurrencies and Contracts, pp. 11–21
31. Crain T, Gramoli V, Larrea M, Raynal M (2018) DBFT: Efficient leaderless Byzantine consensus and its application to blockchains. IEEE Int Symp Netw Comput Appl (NCA) 1–8. IEEE
32. Luo Y, Chen Y, Chen Q, Liang Q (2018) A new election algorithm for DPos consensus mechanism in blockchain. Int Conf Digit Home (ICDH) 116–120. IEEE

33. Kapitza R, Behl J, Cachin C, Distler T, Kuhnle S, Mohammadi SV, Schröder-Preikschat, W, Stengel K (2012) CheapBFT: Resource-efficient Byzantine fault tolerance. In Proceedings of the 7th ACM European Conference on Computer Systems, pp. 295–308
34. Liu J, Li W, Karame GO, Asokan N (2018) Scalable byzantine consensus via hardware-assisted secret sharing. IEEE Trans Comput 68(1):139–151
35. Cason D, Fynn E, Milosevic N, Milosevic Z, Buchman E, Pedone F (2021) The design, architecture and performance of the tendermint blockchain network. In 2021 40th International Symposium on Reliable Distributed Systems (SRDS), pp. 23–33. IEEE
36. Yin M, Malkhi D, Reiter MK, Gueta GG, Abraham I (2019) HotStuff: BFT consensus with linearity and responsiveness. In Proceedings of the 2019 ACM Symposium on Principles of Distributed Computing, pp. 347–356
37. Lyu W-D, Zhou X-G, Yuan Z-M (2017) Design of tree topology based byzantine fault tolerance system. J Commun 38(Z2):139
38. Li W, Feng C, Zhang L, Xu H, Cao B, Imran MA (2020) A scalable multi-layer PBFT consensus for blockchain. IEEE Trans Parallel Distrib Syst 32(5):1146–1160
39. He X, Cui Y, Jiang Y (2019) An improved gossip algorithm based on semi-distributed blockchain network. In 2019 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC), pp. 24–27. IEEE
40. Androulaki E, Barger A, Bortnikov V, Cachin C, Christidis K, De Caro A, Enyeart D, Ferris C, Laventman G, Manevich Y et al (2018) Hyperledger fabric: a distributed operating system for permissioned blockchains. In Proceedings of the Thirteenth EuroSys Conference, pp. 1–15
41. Li Y, Qiao L, Lv Z (2021) An optimized byzantine fault tolerance algorithm for consortium blockchain. Peer Peer Netw Appl 14(5):2826–2839
42. Li P, Wang G, Chen X, Xu W (2018) Gosig: Scalable byzantine consensus on adversarial wide area network for blockchains. arXiv preprint arXiv:1802.01315
43. Wilkinson S, Boshevski T, Brandoff J, Buterin V (2014) Storj a peer-to-peer cloud storage network
44. Yu B, Li X, Zhao H (2020) Virtual block group: A scalable blockchain model with partial node storage and distributed hash table. Comput J 63(10):1524–1536
45. Tang H, Sun Y, Ouyang J (2020) Excellent practical byzantine fault tolerance. J Cybersecur 2(4):167
46. Qing S-D, Zhang Y-H, Liu H-N, He T, Yang B-X, Wei K (2019) Technical evaluation and impact analysis of libra. In International Conference on Smart Blockchain, pp. 87–96. Springer

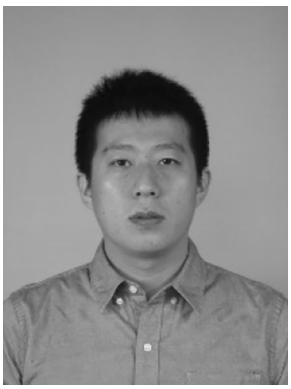
Jinze Li is a master student at the University of Science and Technology of China. His research interests covers blockchain scalability, P2P networks and consensus algorithm. He aims to contribute to research on the intersection of P2P networks and blockchain.

Xiaofeng Li is a research professor of Hefei Institutes of Physical Science, Chinese Academy of Sciences, and a doctoral supervisor at the University of Science and Technology of China. He is the director of Information Center at Hefei Institutes of Physical Science, Chinese Academy of Sciences. His current research interests focus on blockchain technology, computer applied technology and measurement and control technology.

He Zhao is currently a senior engineer and the deputy director of Information Center at Hefei Institutes of Physical Science, Chinese Academy of Sciences. His research interests include computer networking, blockchain technology and software architecture.

Bin Yu received the PhD degree from the University of Science and Technology of China in 2021. His research interests covers blockchain scalability.

Haotian Cheng is a master student at the Hefei Institute of Physical Science, Chinese Academy of Sciences. His research interests covers blockchain technology, privacy-preserving and supervision techniques.



Tong Zhou is the vice president of technology at Anhui Zhongke Jingge Technologies Co., Ltd. His research interests covers blockchain technology and consensus algorithm.

Nianzu Sheng is the deputy director of blockchain lab at Anhui Zhongke Jingge Technologies Co., Ltd. His research interests covers blockchain technology, network security.