ISSN 1330-3651 (Print), ISSN 1848-6339 (Online)

https://doi.org/10.17559/TV-20231119001117

Original scientific paper

MS-PBFT: A Master-Slave Multichain Consensus Mechanism Based on Pre-Broadcast and Threshold Signatures

Bohan KANG*, Yunzhi CHEN, Jianming ZHU

Abstract: This paper addresses scalability limitations in blockchain systems for supporting multi-asset transactions by proposing a master-slave multi-chain architecture. The underlying consensus protocol, called MS-PBFT, utilizes pre-broadcast and threshold signatures to reduce communication overhead. Experimental results demonstrate that MS-PBFT achieves substantially higher transaction throughput and lower latency compared to PBFT consensus, showcasing efficiency gains. This work provides a useful blockchain structure and consensus approach to enable classification of digital assets and concurrent processing across chains.

Key words: blockchain; consensus algorithm; master-slave multichain; PBFT

1 INTRODUCTION

As a new service model to realize the optimal allocation of resources, the data transaction system has attracted wide attention [1-3]. At present, most information systems adopt centralized storage architecture, and the data assets are mostly hosted by the third-party organizations with potential security threats such as data privacy, data integrity, and data access control [4-6]. The emergence of blockchain technology provides a new idea to solve these problems. The application scenarios of the mainstream consensus algorithm under the blockchain are all single-chain models, which are flexible to maintain the consistency of the distributed ledger. However, it is difficult to expand the complex and diversified transaction capabilities in parallel. This paper proposes a master-slave multichain data sharing architecture and deeply analyzes the data sharing mechanism under multichain from the aspects of system logical structure and consensus protocol [7]. Different from the traditional blockchain, this paper designs the two-layer blockchain under the data sharing consortium scenario, aiming to deal with the highly dynamic and real-time digital data elements sharing under the organizations. Each layer contains an independent and complete blockchain, which meets the needs of the complex scene inside and outside of data sharing under multiple stakeholders [8, 9].

1.1 Consensus Applied to Blockchain

Blockchain represented by Bitcoin and Ethereum pioneered the decentralized ledger, achieving P2P trusted value transmission on the premise of not relying on third-party trusted institutions, while traditional consensuses such as PoW (Proof of Work), PoS (Proof of Stake), and DPoS (Delegated Proof of Stake) are not able to meet the application requirements of multi-scene collaboration in social production, as well as with low efficiency and strictly energy consumption problems.

Practical Byzantine Fault Tolerance (PBFT) is a classical consistency consensus based on state machine replication. With the expansion of the application scenario scale of the blockchain, the scalability of PBFT has been growing increasingly. In the blockchain system with n consensus nodes, the communication complexity of PBFT is $O(n^2)$. Many studies improve the PBFT from different perspectives to promote its performance. Some scholars combine the PBFT with the public blockchain consensus to construct committee-based PBFT algorithm, such as Tendermint and ByzCoin [10, 11], while these consensuses still remain the inherent poor scalability of PBFT and the shortcomings of public blockchain consensus. In addition, MinBFT [12], CheapBFT [13], and FastBFT [14] use the secure hardware to build trusted environments to reduce the communication stages. Based on the aggregate signature and gossip protocol, The BFT algorithm [15] continuously combines and propagates parts of the aggregated signatures to reduce the complexity of communication. Given the low performance of existing consensus, it is difficult to support the concurrent processing of digital assets in multiple scenarios and achieve multichain consensus [16].

1.2 Multichain Models of Blockchain

Blockchain has been applied in various scenarios to solve problems. However, with the increasing complexity of the data sharing scenarios in terms of transaction collaboration, scalability, and performance, which further limit the application scenarios and application scale of the blockchain [17, 18], researchers have developed a series of multichain models, which can be divided into parallel models according to the relationship in blockchains. Ma et al. [19]. propose a new paradigm of blockchain scheme, dividing the blockchain into the master-slave layer. Deng et al. [20] propose a lightweight blockchain architecture suitable for the small and medium-sized applications. However, these schemes above do not focus on the consensus interaction in blockchain.

When the off-blockchain services need to interact with multiple blockchains due to the inevitable coupling between services, the off-chain services have to maintain the certificates of nodes in multichain. As for the master-slave multichain, the master-chain is responsible for data storage while the slave-chain for the logical operation. Although master-slave multichain can improve overall performance and adapt to different business scenarios, the constraints of master-chain are the key to the overall performance.

1.3 Contributions and Organizations

To cope with the problems of long-time complexity and low efficiency with diverse asset types in single-blockchain and traditional PBFT, this paper proposes the MS-PBFT based on the master-slave multichain to reduce the bandwidth of primary node and the communication complexity of the consensus protocol. Each subject in master-chain maintains its slave-chain, which stores the transaction content of the organization. All of the subjects collectively maintain the global master-chain, which stores the transaction digest without the double-spending problem to ensure that the transaction cannot be tampered with. In addition, The MS-PBFT consensus algorithm with multiple owners is proposed, which adds the *Pre-Broadcast* module compared to the traditional PBFT protocol to broadcast the data proposal in advance. MS-PBFT act as an efficient method to complete the data verification. By theoretical and experiment analysis, our proposed method and architecture can realize a secure and trusted digital asset trading environment.

The rest of the paper is arranged as follows. The second section introduces the related work of consensus mechanisms. The third section proposes the improved design of the MS-PBFT algorithm. The fourth and fifth sections show theoretical analysis and experimental comparison with the traditional PBFT consensus. The last section is the conclusion of this paper.

2 RELATED WORK

Variant consensus algorithms have been designed for blockchain to provide trustless environment with different performances. As a classic consensus algorithm, the PBFT is used to achieve data transaction consistency in the distributed ledger through three core phases of communication (*Pre-prepare*, *Prepare,* and *Commit* stage). Existing blockchain-based data sharing solutions are mostly based on PBFT as underlying consensus, due to the communication complexity in the consensus process reaching $O(n^2)$, the exponential growth of node communication is observed as the number of nodes increases. When dealing with a large number of nodes, the bandwidth capacity of the nodes in PBFT becomes a crucial bottleneck. The consensus protocol of the PBFT completely includes the following five completely communication phases: *Request*, *Pre-prepare*, *Prepare*, *Commit*, and *Reply* phases, as shown in Fig. 1.

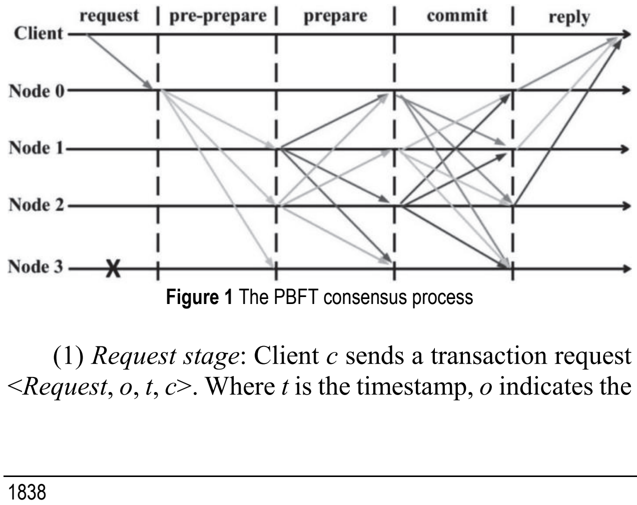

Figure 1 The PBFT consensus process

(1) *Request stage*: Client *c* sends a transaction request <*Request*, *o*, *t*, *c*>. Where *t* is the timestamp, *o* indicates the operation, Request contains the message content *m* and the message digest *d*(*m*).

(2) Pre-prepare stage: The master node accepts the request to client c, dropping the error request and sorting the correct request, and then assigns the number n. The master node then broadcasts a message < *Pre-Prepare*, *v*, *n*, *d* >, *m* > to the slave nodes in the system. Where *v* is the view number, *n* is the sequence number of the request assigned sequence number, and *d* is the summary of message *m*.

(3) Prepare stage: The nodes judge the message received from Pre-prepare stage, and enter the Prepare stage, broadcasting the message < *Prepare*, *v*, *n*, *d*, *i* > to other nodes (including the primary node), where *i* is the node number.

(4) Commit stage: When the node receives $2f$ verified message, it enters the Commit stage, the node sends < *Commit*, *v*, *n*, *d*, *i* > messages to other nodes, which receives $2f$ confirmation messages from other nodes, writes the received messages to the log if they are passed, executes the request, and replies with the message <*REPLY*, *v*, *t*, *c*, *i*, *r*> to the client, where *r* is a response of node *i*.

(5) Reply stage: The result is valid when the client receives at least $f + 1$ identical response from nodes.

In 2016, Tsai [21] divided the traditional single-layer blockchain into two architectures: Account Blockchain (ABC) and Transaction Blockchain (TBC). ABC is responsible for query and storage accounts, while TBC is set for building blocks and perform transactions. Although ABC and TBC can achieve load equilibrium, the architecture is still reluctant to reach the classification of diversified digital assets. Feng et al. [22] proposed the SDMA-PBFT consensus for the consortium blockchain, introducing classification and agent mechanism to divide the whole network into several sub-domains. Gao et al. [23] proposed the T-PBFT algorithm based on trust nodes with high credit values that enhance the fault tolerance rate of the PBFT. However, its communication complexity is still as high as $O(n^2)$. Laphou et al. [24] proposed G-PBFT consensus, by selecting several robust and reliable IOT devices as consensus nodes; it can prevent Sybil attacks effectively. However, the G-PBFT also has the problem of privacy disclosure.

The idea of the master-slave multichain was formally proposed in 2018, including a master-chain and multiple slave-chains [25]. Each slave-chain can be regarded as an extension of the main-block layer, and the block hash in slave-chain is anchored to the latest block in master-chain in the time slice. The master-slave multichain was first applied in the diversified digital asset trading business. Each slave-chain serves as the channel for its digital asset transaction, ensuring the privacy and concurrency of its transactions in different channels [26]. The primary node in the slave-chain guarantees the transaction consistency of channels through the consensus, all the primary nodes among different slave-chains are jointly responsible for maintaining the consistency of master-chain.

Currently, the academic research of master-slave multichain data transactions are still in early development. Some limited researches have primarily focused on the innovative applications in areas such as digital assets trading, energy services, and agricultural information management. In this paper, the architecture based on master-slave multichain contains the following features and advantages:

(1) Establishing a multi-party distributed digital identity management mechanism.

The sharing and supervision of data elements need to be supported by entity institutions. Trusted digital identity is the premise of effective data management of all stakeholders. The existing data sharing and management interaction platform still adopts centralized coordination mechanism; the authentication and management of user identity will have a strong dependence on a single central organization. This paper manages the digital identity information of the decentralized master-slave multichain network through the DID digital identity. By issuing distributed digital certificates, the master-slave multichain system builds the asymmetric key encryption and CA authentication as the basic signature algorithm, ensuring effective data sharing in the domain and automatically realizing the cross-chain data exchange outside the domain.

(2)Making an efficient and secure architecture for distributed datasets verification.

In the master-slave multichain architecture proposed in this paper, the master-chain saves the transaction hash value, while the slave-chain stores the transaction contents in its channel, and the master-slave layer connects each other with hash anchor. If members in channel conspired to tamper with the slave-chain transaction in the channel, it is of necessity to tamper with the master-chain transaction stored by the whole network members at the same time. So, the cost of tampering is relatively large. Moreover, in the MS-PBFT consensus, the signature generated by group member in threshold way (threshold value $T = 2f + 1$) to the channel, each copy of the threshold signature reaches consensus by proposal voting of the parties. The global block contains the threshold signature of each channel, representing the quorum willingness of voting results. All nodes in slave-chains only need to verify the legitimacy of threshold signature after receiving the global block.

(3) Building a stable and reliable diversified datasets sharing environment network.

The master-slave multichain architecture proposed in this paper makes breakthrough in the service function and performance constraints compared with traditional single blockchain, which achieves data isolation and transaction processing through channel technology. Each node in the channel is responsible for verifying the consistency of the data. Moreover, MS-PBFT ensures that the system can function well when there are at least 2/3 good network communication nodes in each group. Therefore, the system can realize the classification and processing of different digital assets while taking into account the data protection, meeting the diversified business needs of enterprises.

3 PROPOSED METHOD

3.1 The Architecture of Master-Slave Blockchain

This section proposes the master-slave blockchain scheme and MS-PBFT consensus from the perspective of reducing the export bandwidth of the primary node in the multichain and optimizing the communication complexity of the consensus protocol. Each channel represents a kind of slave-chain, which stores the transaction digest through the hashkey. All the nodes in the system depend on DID distributed digital identities to securely manage and verify the digital identity information of the decentralized blockchain network. The topological map of the master-slave multichain model is shown in Fig. 2.

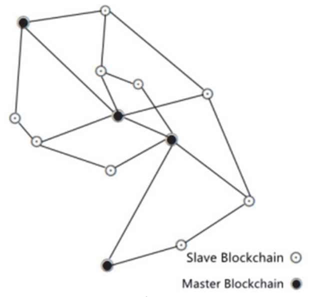

Figure 2 The topological map of the master-slave multichain model

The master-chain stores the hash value of double flower transactions in all channels, ensuring that the digital assets are globally consistent and untampered with, and jointly maintained by all members in the system. The transaction is only effective when the hash value of the blockchain transaction is written into the master-chain, the format of which is:

MasterBlock = <*pre_hash*, *cur_hash*, *height*, *timestamp*, *SC_height*, *Merkleroot*>

where *pre_hash* is the hash value of the previous block, *cur_hash* is the hash value of the current block, *height* represents the block height of the master chain, *SC_height* means the slave-chain identification for the digital asset transaction, which is used to quickly locate to the corresponding block height. *Merkleroot* is a data structure that organizes the hash value of a transaction in a *Merkletree* manner. Meanwhile, the format structure of slave-chain is:

SlaveBlock = <*pre_hash*, *cur_hash*, *timestamp*, *MC_height*, *Merkleroot*>

where *pre_hash* is the hash value of the previous block in the slave-chain, *MC_height* is the block height of the master chain, aiming for rapid positioning to the block height where the corresponding master chain is located. *Merkleroot* is a data structure that organizes the hash value of a transaction in a *Merkletree* manner.

The underlying architecture of the data sharing scheme is a credible decentralized data interaction service based on blockchain, which is divided into data layer, network layer, consensus layer, participation incentive layer and system application layer.

(1) Data layer.

The data layer is responsible for realizing the underlying operation support of the collection, calculation, and storage of data. The hash and keyword index of encryption data under the master-slave multichain are stored in the slaveblock, and the transaction sheet of the block is repeatedly stored in the *Merkleroot* of the block header for rapid induction and verifying the existence and integrity of the data. From the slave-chain block header, the front *pre_hash*, *cur_hash*, *timestamp*, r*andom number*, *Merkleroot* are stored. The master-chain is responsible for storing the block hash of the corresponding slave-chain, and the hash anchor of the master-slave multichain is realized through the cross-chain exchange node.

(2) Network layer.

The network layer is composed of institutions at all levels, all nodes in the whole network distribute the new information to other nodes. However, under the service mode of the institution, there are many participants. If the single-chain data storage and circulation mode is adopted with the traditional blockchain, there are bottlenecks in performance, capacity, privacy, and expansion, which cannot meet the full realization of the service application scenarios of multi-party participation. Therefore, this paper adopts the master-slave multichain structure to realize P2P interaction. The master-chain is responsible for data sending and receiving, the slave-chains are responsible for data uploading and reading, and the cloud server is responsible for storing encrypted data, and recording data index and keywords from the blockchain. Through the functional positioning of the master chain, the transmission mechanism of data under the service is optimized.

(3) Consensus layer.

The consensus layer is the core technology of master-slave multichain. Its main function is to efficiently reach a consensus on the effectiveness of block data in the decentralized network to solve the trust problem between nodes. In the master-slave multichain architecture, data elements can not only be traded within the channel, but also be queried and shared among blockchains. This paper designs an improved consensus based on PBFT in master-slave multichain, effectively solving the problems of low throughput and high transaction delay. The *Pre-Broadcast* module and *threshold signature* are introduced to reduce the communication complexity of the protocol as well as reduce the export bandwidth of the primary node.

(4) Incentive layer.

Based on the concerns of privacy leakage, lack of incentive mechanism and other problems, the multi-party institutions at all levels cannot be effectively aggregated. To promote institutions at all levels to actively participate in the sharing and improving the value of data assets, an incentive mechanism is established to participate in the organization under master-slave multichain. The data owners can develop pricing strategies for the use of data, and provide database interface services to scientific research institutions and enterprises in need. At the same time, under the operation supervision and macro-control of government departments, the value appreciation of their respective data assets can be realized through digital currency transactions and settlements, prompting all parties to share the stored data assets.

(5) Application and contract layer.

The application and contract layer specifies the application scope of the blockchain. This layer is mainly responsible for service practices such as data assets storage and data query, encapsulating various application scenarios and cases of the data transaction service, and automatically executed by smart contract. For example, the contract layer encapsulates various types of script codes, algorithms, and smart contracts for specific application scenarios. It automatically triggers execution when the smart contract reaches its constraints. In the future, more practical case scenarios of data assets visualization and service management can be developed according to the development needs of participants.

3.2 The MS-PBFT Consensus

Compared with traditional PBFT, MS-PBFT adds the *Pre-Broadcast* module to broadcast data proposal in advance, so that the primary node only needs to package the hash value of the data proposal and distribute it to other nodes. In addition, the threshold signature is introduced based on three-stage communication protocol to realize batch verification and signature aggregation, completing verification through the cooperation of the master-slave multichain consensus.

Pre-Broadcast module is a part of additional phrase to simplify the communication complexity and overhead of the MS-PBFT consensus process. In the *Pre-Broadcast* module, the client firstly sends the request to any node in the slave-chain, and actively forwards it to other nodes connected with it. After a round of system traversal, a complete list of transactions will be maintained in the transaction pool of all consensus nodes, which acts as the functions of caching transactions and packaging blocks. Since the *Pre-Broadcast* module broadcasts the data proposal in advance, the primary node only needs to package the hash value of the data proposal and distribute it to other nodes in three-stage communication, which can reduce the export bandwidth of the primary node in the master-slave multichain and communication complexity of the consensus protocol.

(t, n) threshold signature indicates that the signature key of a group is shared by n members. In the threshold signature based on BFT consensus, the primary node distributes the group of signature rights to other ledgers of nodes, which vote in a threshold way. Only when the number of voting reaches threshold t can generate effective signature resolution. In this way, it can ensure the consensus that most of the results of agreement in the blockchain, as well as in the minimum connectivity in the network environment to achieve low latency, high robust consensus algorithm. Assuming that each member has the group signature private key of own channel, the group public key is available in the master-slave multichain system. TheMS-PBFT process is shown in Fig. 3.

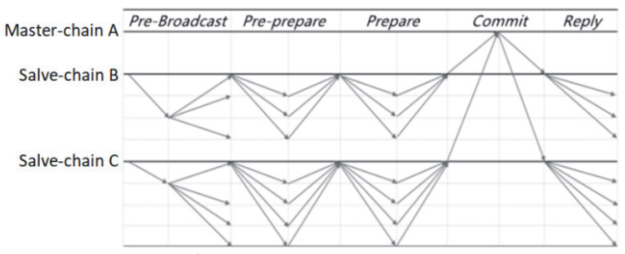

Figure 3 MS-PBFTprocess in master-slave multichain

(1) The paper sets the total number of nodes of participants in each slave-chain channel as $3f + 1$, where f represents the number of Byzantine nodes. There are at most f Byzantine nodes in each slave-chain. In the *Pre-Broadcast* stage, the client selects the slave node i randomly in time slice t, storing data in the local memory pool, and then actively broadcasts the $hash_{proposal}$ of the data to all nodes in the slave-chain, which is verified and stored in the memory pool. Finally, each node returns a response to the primary node I.

(2) In the *Pre-prepare* stage, the primary node I packages all the $hash_{proposal}$ collected within time t into blocks. After signed by primary node I itself, the Message < sig_I ($hash_{proposal}$, v_I, h_I, $hash(block_I)$), $block_I$) > broadcasts to all nodes in the channel, where sig_I represents the digital signature of I. v_I and h_I refer to the view number and block height of the slave-chain, respectively. $hash(block_I)$) is the block digest of $block_I$. After receiving the $hash_{proposal}$ from primary node, the slave-chain node checks the signature and invokes the $hash_{proposal}$ in its own memory pool for comparison verification. When successful, the slave-chain node votes based on its threshold signature *PartSigi*, and returns the voting result as well as threshold signature sub-key to the primary node in slave-chain.

(3) In the *Prepare* stage, after collecting all the voting results in slave-chain, primary node firstly needs to check whether the threshold signature is correct as well as the view number v_I is consistent. If the primary node collects at least $2f$ threshold votes, it will vote for $block_I$ together with its own threshold signatures. When $T = 2f + 1$, $ThresholdSig_I = ThreSig(PartSig_1, PartSig_2, \ldots, PartSig_T)$. Each primary node (such as B and C) sends the voting result to the master-chain (such as A), and broadcasts the $hash(block_I)$ to the system for verification.

(4) In the *Commit* stage, after taking the verification of v_I, h_I and voting result, primary node in master-chain generates the new $block_{global} = (hash(block_1), hash(block_2), ..., hash(block_I))$, and executes the consensus in master-slave multichain system. The primary node of the slave-chain broadcasts the new block in its channel, all nodes in the slave-chain update the ledger after checking its signature, and write the new block into local distributed ledgers.

(5) In the *Reply* stage, the primary node in master chain writes $hash(block_{global})$ into the global ledger. Through data consistency protocol of the master-slave multichain, the asynchronous data sharing is realized. The consensus execution process is shown in algorithm 3-1.

Algorithm 3-1: Master-Slave Multichain Consensus Algorithm

```
Input: produced data in time slice $t$
Output: $block_{global}$ in time slice $t$
    function GenerateGlobalBlock ( )
      for $i, j \in Mpi$
        if Time( ) - Start $t < MaxDelay$
          Broadcast Replica ($i$, $hash(proposal)_{[dpi]}$)→MTxPool($j$, $hash(proposal)_{[dpi]}$)
        Generate pre-parepare
        Broadcast pre-parepare
    else
        return Timeout
    if primary(Mpi) check(Timestamp) & check (signature ($h_{block}$)) is true
      Broadcast <$sig_{MP}(hash(proposal), v_{MP}, h_{MP}, hash(block_{MP})), block_{MP}$)>
    for $i \in Mpi$
      if replica check(MTxPool($i$, $hash(proposal)_{[dpi]}$) == ($hash(proposal)_{[dpi]}$)
        return $PartSig_i$ and $sk_i$ to primary($i$)
        Generate prepare
        Broadcast prepare
        if primary($i$) check($PartSig_i$) & check ($v_{MP}$) is true
          if $T = 2f_i + 1$
            $ThresholdSig_{Mp} = ThresholdSig(PartSig_{Mp1}, PartSig_{Mp2}, \ldots, PartSig_{MpT})$
          Broadcast $ThresholdSig_{Mp}$ & $hash(block_{MP})$) & $v_{MP}$ & $h_{MP}$ to MasterChain
      for $k \in M_p$
          if primary($M_k$) check($ThresholdSig_{Mp}$) & check ($v_{MP}$) is true
            Generate $block_{global} = (hash(block_{M1}), hash(block_{M2}) \ldots hash(block_{Mp}))$
            Generate commit
            Broadcast commit
        Broadcast $block_{global}$ to Mpi
             Committed $block_{global}$ on the slave-chain
        return true
        Committed $block_{global}$ on the master-chain
```

4 SECURITY ANALYSIS
4.1 The Master-Slave Multichain is Tamper-Proof

The consensus mechanism under the master-slave multichain uses threshold signature as the consistency verification criterion, setting the total number of nodes to $3f + 1$ if the Byzantine node is f. In the Prepare stage of the consensus, the primary node can perform the aggregation operation of the threshold signature only when the threshold value is at least $T = 2f + 1$. This ensures that in the worst case of at most f Byzantine nodes in the channel, an honest node with an $f + 1$ threshold can still occupy the majority vote of the threshold signature. In each stage of the messaging process of the consensus mechanism, the P2P communication is endorsed by the digital signature and DID of the sender, and the master-slave multichain is anchored by the block key to ensure that the information is traceable and tamper-proof. When validating the data, the receiver first checks whether the signature is correct and the corresponding public key address is in the license list so that it can effectively guard against witch attacks.

4.2 The Master-Slave Multichain Architecture does not have Fork Problems

Since the blockchain stores information through the linked storage structure, when a new block appears at a certain height h of the blockchain, then the block $h + 1$ may appear as the fork of two chains. Assuming that a certain slave-chain is at the height of block h in time t, two new $block_1$ and $block_2$ simultaneously appear on the master-slave multichain. Due to at least $2f + 1$ threshold signature from the primary node in blockchain, the vote is required to perform the aggregation of threshold signature. Therefore, in the case of fork success, the primary nodes of the two broadcast new blocks need to obtain at least $4f + 2$ voting results. The honest node can only vote once for all blocks, the Byzantine node can vote at most f times for each new block. In conclusion, in the case of two block bifurcations, the total number of votes for all nodes in the channel is at most $2f + (2f + 1) = 4f + 1 < 4f + 2$ times, so there will be no bifurcation.

4.3 The Master-Slave Multichain Architecture is Credible and Reliable

Under the master-slave multichain proposed in this paper, the number of Byzantine nodes f does not exceed 1/3 of all nodes $3f + 1$, and there are at least $f + 1$ honest nodes in each channel, which means that even in the worst case of the consensus voting process, the number of honest nodes can still occupy more than 50%. In addition, due to the scattered geographical location of participating nodes and uncertain network transmission, the master-slave multichain adapts threshold signature in the consensus process. The signature key of the channel is shared by all participating nodes in a threshold way. When the number of votes collected by the primary node in the channel is the legal threshold, the threshold signature is aggregated upwards without requiring responses from all nodes. Through the *Pre-Broadcast* module and threshold signature voting in MS-PBFT, the inter-communication between nodes in the traditional PBFT is avoided, the communication complexity is reduced from $O(n^2)$ to $O(n)$.

5 EXPERIMENTS

In terms of the system performance, this paper takes experiments for the proposed master-slave multichain architecture. Throughput (i.e. the number of transactions processed per unit of time) and block confirmation delay (i.e. the time of consumption until data is packaged into the block and confirmed to be irreversible) are important indicators to measure the scalability of blockchain. Based on the Docker container in the Hyperledger Fabric, this paper builds the consensus module and constructs three different slave channels of nodes based on MS-PBFT to simulate three slave-chains. The master chain is composed of three nodes to conduct experiments. The parameters of the nodes are shown in Tab. 1.

Table 1 Parameters of blockchain nodes for the master-slave multichain system

	Channel = 2	Channel = 3	Channel = 4
Number of slave-chain nodes i	100	150	200
Legal threshold number of votes T	60	90	120
Number of master-chain nodes p	2	3	4

As shown in Fig. 4, when taking the different numbers of the slave-chains (the number of channels is set to 2, 3, 4) access to the master chain, the system throughput (TPS) is positively correlated with the number of accessed slave-chains, which show that the more slave-chains access to the system, the larger data transaction throughput will be computed. In contrast to the traditional PBFT, the system performance in MS-PBFT experiences a significant decline as the number of participating nodes surpasses a certain threshold, which highlights its ability to handle concurrent consensus processes for transactions through a master-slave multichain structure.

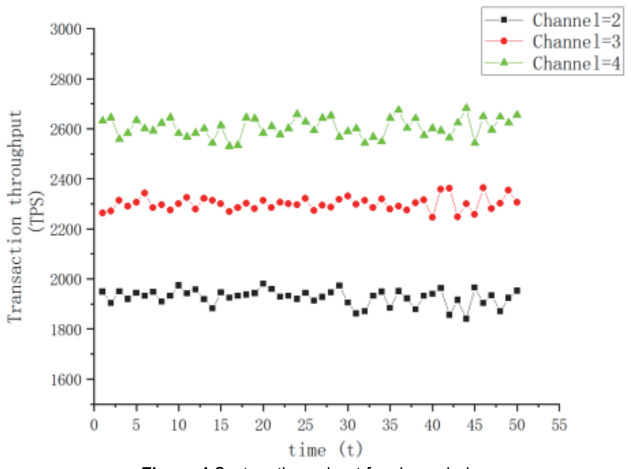

Figure 4 System throughput for slave-chains

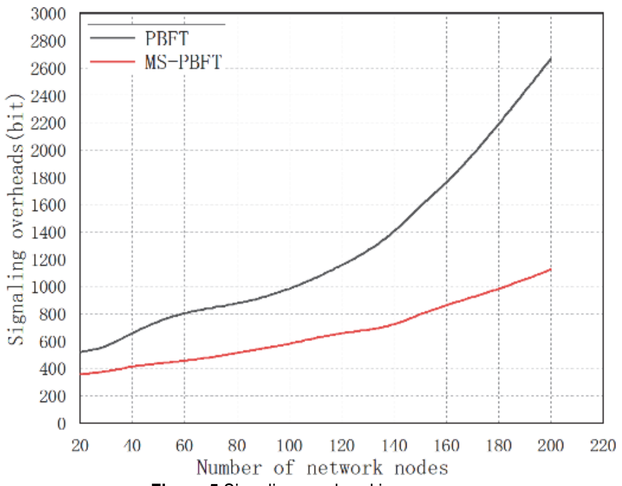

Figure 5 Signaling overhead in consensus

Fig. 5 shows the signaling overhead of different blockchain structures in the consensus respectively, i.e. master-slave multichain and traditional consortium blockchain, signaling overhead refers to the number of signaling sent by the whole network nodes in the consensus process. It can be seen that with the growth of the number of nodes in the whole network, the signaling overhead of the PBFT consensus algorithm in the traditional consortium chain increases exponentially by $O(n^2)$. The MS-PBFT in the master-slave multichain decouples the consensus process into two stages while maintaining a unified consensus on the entire network level for the data itself. The signaling overhead is $O(n - m)$ at the slave-chain level and $O(m^2)$ at the master chain. Due to the significantly smaller number of m compared to n, this results in a significant reduction of signaling overhead. In addition, the paper tests the average delay and transaction throughput between MS-PBFT and PBFT with different numbers of nodes, and the experimental results are shown in Fig. 6 and Fig. 7.

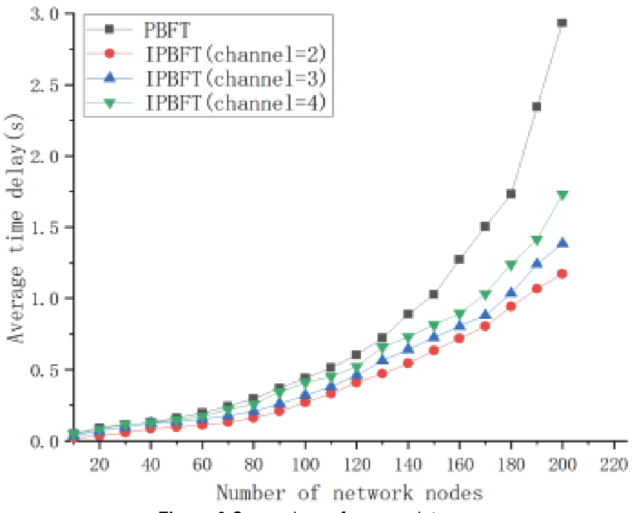

Figure 6 Comparison of average latency

Due to the requirement of two rounds of inter-node voting in the three-stage consensus implementation process, PBFT exhibits a time complexity of $O(n^2)$, thereby limiting its applicability to systems with a small number of nodes and constraining scalability. MS-PBFT performs the consensus process concurrently through the isolation channel constructed by the master-slave multichain system, which simplifies the communication steps of node interaction in PBFT and greatly reduces the communication and computing overhead, making the time complexity of the consensus become $O(n)$.

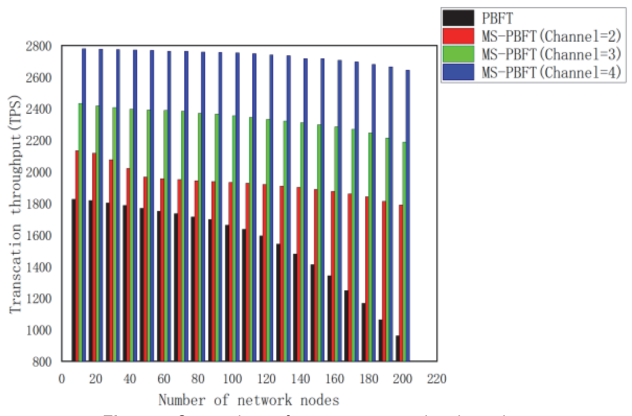

Figure 7 Comparison of average transaction throughput

Although the MS-PBFT also exhibits an increase in resource consumption with the number of nodes *n*, it significantly optimizes traditional PBFT consensus, showcasing the adaptability of the proposed architecture in large-scale networks. In addition, in the throughput test index of transaction transactions, MS-PBFT remains in a stable range with the growth in the number of nodes and can still maintain high throughput even under a large number of nodes, showing good scalability.

Our work cannot able to focus on the view of the access rights of different role users to take transaction at different locations in blockchain. In the future, we will study the identity-based and attribute-based access control models to ensure the secure sharing of transaction ledger in the master-slave multichain. In addition, based on our proposed multichain architecture, we will continuously design a hierarchical tree-based key management method to solve the problem of key leakage and cancellation.

6 CONCLUSION

In conclusion, this paper presented an innovative master-slave blockchain framework and associated MS-PBFT consensus protocol to overcome scalability bottlenecks of traditional blockchain. By distributing transactions across channels with isolated consensus while maintaining global data integrity, substantial performance gains are realized as validated experimentally. This offers a practical solution for blockchain-based systems to support multi-asset transactions by exploiting parallelization. Future enhancements can focus on hierarchical tree-based key management method, and identity-based and attribute-based access control methods.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (62372493) and Research on Blockchain Security Cross-Chain & Supervision Prototype System Oriented to Fusion Application (LD22F020002). We appreciate their supports very much. The work presented in these studies remains the sole responsibility of the authors.

7 REFERENCES

[1] Shao, Q. F., Jin, C. Q., Zhang, Z., et al. (2018). Blockchain: Architecture and research progress. *Chinese Journal of Computers, 41*(5), 969-988.
[2] Efanov, D. & Roschin, P. (2018). The all-pervasiveness of the blockchain technology. *Procedia Computer Science, 123*, 116-121. https://doi.org/10.1016/j.procs.2018.01.019
[3] Nakamoto, S. (2008). Bitcoin: A peer-to-peer electronic cash system. from https://bitcoin.org/bitcoin. Accessed October 2023.
[4] Tsai, W. D., Yu, L., Wang, R., et al. (2017). Blockchain application development techniques. *Journal of Software, 28*(6), 1474-1487.
[5] Yuan, Y. & Wang, F. Y. (2016). Current situation and prospect of blockchain technology. *Acta Automatica Sinica, 42*(4), 3-16.
[6] Krichen, M., Ammi, M., Mihoub, A., & Mutiq, A. (2022). Blockchain for modern applications: A survey. *Sensors, 22*(14), 5244. https://doi.org/10.3390/s22145274
[7] Chai, H. Y., Leng, S. P., Chen, Y. J., & Zhang, K. (2021). A Hierarchical Blockchain-Enabled Federated Learning Algorithm for Knowledge Sharing in Internet of Vehicles. *IEEE Transactions on Intelligent Transportation Systems, 22*(7), 3975-3986. https://doi.org/10.1109/TITS.2020.3002712
[8] Pass, R. & Shi, E. (2018). Thunderella: Blockchains with optimistic instant confirmation. *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. New York, USA, 3-33. https://doi.org/10.1007/978-3-319-78375-8_1
[9] Yao, S., Lei, Z., Gang, F., Bowen, Y., Bin, C., & Muhammad, A. I. (2019). Blockchain-enabled Wireless Internet of Things: Performance Analysis and Optimal Communication Node Deployment. *IEEE Internet of Things Journal, 6*(3), 5791-5802. https://doi.org/10.1109/JIOT.2019.2905743
[10] Pass, R. & Elaine, S. (2017). Hybrid consensus: Efficient consensus in the permissionless model. *31st International Symposium on Distributed Computing*. Vienna, Austria, 39-55.
[11] Kogias, E. K., Jovanovic, P., Gailly, N., et al. (2016). Enhancing bitcoin security and performance with strong consistency via collective signing. *25th USENIX Security Symposium*. Austin, USA, 279-296.
[12] Veronese, G., Correia, M., Bessani, A., Lau, C. L., & Paulo, V. (2013). Efficient Byzantine fault-tolerance. *IEEE Transactions on Computers*, 62(1), 16-30. https://doi.org/10.1109/TC.2011.221
[13] Kapitza, R., Behl, J., Cachin, C., Tobias, D., Simon, K., Seyed, V. M., Wolfgang, S. P., & Klaus, S. (2012). CheapBFT: resource-efficient Byzantine fault tolerance. *Proceedings of the 7th ACM European conference on Computer Systems*, Switzerland, 295-308. https://doi.org/10.1145/2168836.2168866
[14] Liu, J., Li, W., Karame, G. O., & Asokan, N (2018). Scalable Byzantine consensus via hardware-assisted secret sharing. *IEEE Transactions on Computers*, 68(1), 139-151, https://doi.org/10.1109/TC.2018.2860009
[15] Long, J. & Wei, R. (2019). Scalable BFT consensus mechanism through aggregated signature gossip. *IEEE International Conference on Blockchain and Cryptocurrency*, Seoul, South Korea, 360-367. https://doi.org/10.1109/BLOC.2019.8751327
[16] Fan, X. (2018). Scalable practical Byzantine fault tolerance with short-lived signature schemes. *Proceedings of the 28th*

Annual International Conference on Computer Science and Software Engineering, Markham, Canada, 245-256.

[17] Thai, Q. T., Yim, J. C., Yoo, T. W., Yoo, H. K., Kwak, J. Y., & Kim, S. M. (2019). Hierarchical Byzantine fault-tolerance protocol for permissioned blockchain systems. *The Journal of Supercomputing, 75*(11), 7337-7365. https://doi.org/10.1007/s11227-019-02939-x

[18] He, H. W., Yan, A., & Chen, Z. H. (2018). Summary of smart contract technology and application based on blockchain, *Journal of Computer Research and Development, 55*(11), 2452-2466.

[19] Ma, Z., Huang, W., Bi, W., Hongmin, G., & Zhen, W. (2018). A master-slave blockchain paradigm and application in digital rights management. *China Communications, 15*(8), 174-188. https://doi.org/10.1109/cc.2018.8438282

[20] Deng, X. H., Zhu, N. H., Huang, L., et al. (2021). LBA: lightweight blockchain architecture. *Application Research of Computers, 38*(10), 2904-2908.

[21] Tsai, W. T., Blower, R., Zhu, Y., & Yu, L. (2016). A system view of financial blockchains. *IEEE Symposium on Service-Oriented System Engineering*, Oxford, UK, 450-457. https://doi.org/10.1109/SOSE.2016.66

[22] Feng, L. B., Zhang, H., Chen, Y., & Lou, L. (2018). Scalable dynamic multi-agent practical Byzantine fault-tolerant consensus in permissioned blockchain. *Applied Sciences, 8*(10), 1919. https://doi.org/10.3390/app8101919

[23] Gao, S., Yu, T. Y., Zhu, J. M., & Cai, W. (2019). T-PBFT: An EigenTrust-based practical Byzantine fault tolerance consensus algorithm. *China Communications, 16*(12), 111-123. https://doi.org/10.23919/JCC.2019.12.008

[24] Lao,L., Dai, X., Bin, X., & Songtao, G. (2020). G-PBFT: a location-based and scalable consensus protocol for iot-blockchain applications. *IEEE International Parallel and Distributed Processing Symposium*, New Orleans, USA, 664-673. https://doi.org/10.1109/IPDPS47924.2020.00074

[25] Min, X. P., Li, Q. Z., Kong, L. J., et al. (2018). Permissioned blockchain dynamic consensus mechanism based multi-centers. *Chinese Journal of Computers, 41*(5), 1005-1020.

[26] Zhang, W. F., Sun, H. F., Zhang, Y. D., et al. (2022). A Consensus Algorithm for Consortium Chain with Tree Based Master-Slave Multi-Chain Architecture. *Acta Electronica Sinica, 50*(2), 257-266.

Contact information:

Bohan KANG, PhD Candidate
(Corresponding author)
School of Information, Central University of Finance and Economics,
Beijing, 100081, China
E-mail: kangbh.postgra@qq.com

Yunzhi CHEN, Professor
College of Information Engineering, Hangzhou Vocational & Technical College,
Hangzhou, 310018, China
E-mail: 2006010022@hzvtc.edu.cn

Jianming ZHU, Professor
School of Information, Central University of Finance and Economics,
Beijing, 100081, China
E-mail: zjm@cufe.edu.cn