Received October 27, 2020, accepted November 8, 2020, date of publication November 16, 2020, date of current version November 25, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3037694

CRSM: An Effective Blockchain Consensus Resource Slicing Model for Real-Time Distributed Energy Trading

MENG HU[1,2], TAO SHEN[1,2], (Member, IEEE), JINBAO MEN[3], ZHUO YU[3], AND YINGLI LIU[1,2]

[1]Faculty of Information Engineering and Automation, Kunming University of Science and Technology, Kunming 650500, China
[2]Computer Technology Application Key Laboratory of Yunnan Province, Kunming University of Science and Technology, Kunming 650500, China
[3]Beijing Zhongdian Puhua Information Technology Company, Ltd., Beijing 100000, China

Corresponding author: Tao Shen (shentao@kmust.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61671225, in part by the Yunnan Applied Basic Research Projects under Grant 2018FA034, in part by the Yunnan Reserve Talents of Young and Middle-aged Academic and Technical Leaders (Shen Tao, 2018), and in part by the Yunnan Young Top Talents of Ten Thousands Plan (Shen Tao, Yunren Social Development No. 2018 73).

ABSTRACT Distributed energy trading has become an essential part of the energy trading market and provides a useful supplement to traditional centralized energy trading, but there are still problems such as opaque trading information and asymmetric user data. The blockchain technology has the advantages of traceability, trade openness, and data transparency, which is naturally suitable for distributed energy transactions. The electricity information data transmission represented by distributed energy transaction has the characteristics of real-time, which has a high-efficiency requirement on the selected blockchain technology. The consensus algorithm is the core of blockchain technology and affects the efficiency of the blockchain system. The efficiency of the existing consensus algorithms for energy transaction-oriented blockchain still needs to be improved. In this paper, a consensus resource slicing model(CRSM) is designed to meet the requirements of consensus efficiency in energy trading scenarios. Specifically, CRSM divides consensus nodes into different consensus domains for concurrent consensus, and the storage domain only stores block information without consensus. By building an experimental platform, the efficiency of CRSM was verified, the communication pressure of the blockchain system was reduced, and the consensus speed was effectively improved.

INDEX TERMS Consensus mechanism, multi-consensus domain, blockchain, distributed energy trading.

I. INTRODUCTION

The traditional centralized energy trading market relies on third-party institutions to maintain the stability of the trading market, resulting in additional costs and potential problems of poor efficiency [1]. The distributed energy transaction provides a beneficial supplement to the traditional centralized energy transaction with the advantages of local consumption and improved transaction efficiency. Nevertheless, we still need to face the challenges of user information asymmetry, opaque transactions, and regulatory issues. When the distributed energy trading market is not yet mature enough, it is an excellent choice to introduce new technical means to avoid potential risks [2]. Nowadays, blockchain technology [3], with its characteristics of decentralization, traceability, openness, and transparency, provides a new way of thinking for distributed energy transactions to deal with these challenges. Blockchain technology is essentially a non-tamperable distributed database, which establishes multi-party trust without any central organization. In the energy trading market, there are a large number of entities participating in transactions. If a public chain is used, each transaction needs to be verified by the vast majority of nodes in the entire network. Furthermore, the transaction delay is too high, and it is challenging to meet the needs of real-time transactions on the energy Internet. Due to the strict access and clearance, openness,

The associate editor coordinating the review of this manuscript and approving it for publication was Srinivas Sampalli.

transparency, and decentralization characteristics of alliance blockchain, adequate supervision can be achieved within a specific range, and it naturally fits with the distributed energy trading [4]–[7].

The electric information data transmission represented by distributed energy transaction has the characteristics of real-time, which has a high transmission speed requirement for the selected blockchain technology. Because of its distributed architecture design, blockchain needs a consensus algorithm to coordinate the data consistency of all nodes in the whole network [8]. The consensus algorithm is the core technology of blockchain, which not only guarantees the security and stability of the blockchain but also is the critical factor affecting the operational efficiency of the whole blockchain system. At present, consensus mechanisms commonly used in alliance chains include PBFT, Raft, and Paxos, as well as consensus algorithms improved based on these algorithms, including Reputation-based Byzantine Fault Tolerance algorithm, (RBFT) Credit-delegated Byzantine Fault Tolerance algorithm(CDBFT), and Byzantine Raft algorithm(BRaft), etc. Although the above consensus algorithm has made some improvements, it still cannot meet the requirements of real-time electric data transmission in terms of consensus speed, and the high-performance blockchain consensus mechanism still needs further research [9]–[13].

This paper aims to study the consensus algorithm to optimize the blockchain of the alliance, combining the business of the regionally integrated energy trading system and existing technology, slice the consensus resource for the actual business scenario, optimize the consensus process and conduct verification. The main contributions of this paper are as follows.

1. This paper proposes a consensus model applicable to comprehensive energy trading, slices consensus resources, and divides consensus domains to make multiple consensus domains concur, improve the global block generation efficiency and optimize the consensus speed.

2. This paper optimizes the consistency protocol in the consensus process and reduces the communication complexity to $O(n)$.

3. Given the strict entry and exit mechanism of the alliance chain and the hardware environment of the alliance chain members, This paper also creatively separates the consensus unit and storage unit in the blockchain system, and apportions the communication pressure of the blockchain system, effectively improving the consensus speed.

4. By building an experimental platform, the resource slicing method proposed in this paper is verified and analyzed.

Section I and section II of this article mainly discuss the related work of blockchain technology and consensus mechanism research. Section III proposes a consensus resource slicing model(CRSM) and optimizes the communication protocol of the consensus algorithm. Section IVverifies the proposed consensus resource slicing method through experiments. Moreover, a conclusion is drawn in section V.

II. RELATED WORKS

The blockchain is essentially a distributed database that can not be tampered with and stored by multiple nodes. The data is written into blocks after being confirmed by nodes, and the generated blocks are connected into a chain in chronological order. From a technical perspective, the core of the blockchain is a block and chain data storage structure. A block is a data structure that stores transaction data. It maps the occurrence of a transaction. A block is a data structure that stores transaction data, which maps the occurrence of a transaction. Transaction data is generated and stored in units of blocks connected in chronological order into a chain. What's important is that each node in the blockchain system records all block information, and the blocks in the blockchain are closely linked. If a malicious node wants to change a block data, it needs to change all the blockchain data of most nodes, so that the income brought by changing the data is far less than the cost, thus ensuring the data storage security of the blockchain. Due to the tamper-proof nature of blockchain technology, more and more industries use blockchain as the underlying technology to develop business systems. At the same time, as a platform that can build trust, blockchain is the core technology in the future development of the information Internet to the value Internet [14]–[16].

However, the current consensus algorithm has become a bottleneck in the performance of the entire blockchain system. The purpose of the consensus algorithm is to efficiently achieve consistency and correctness between each node of a zero-trust distributed system. Proof-of-work, proof of equity, and voting-based consensus mechanisms all have their respective advantages in different application scenarios. For different application scenarios of blockchain technology, how to combine the strengths of different consensus algorithms is the most valuable research in improving consensus algorithms [17].

In view of the above content, there have been relevant studies. Reference [18] focuses on the energy transaction process between electric vehicles and the distribution network (DN) based on the Byzantine blockchain consensus framework. Reference [19] developed an optimization model and a blockchain-based architecture to manage the operation of crowdsourced energy systems(CESs) with peer-to-peer energy transaction transactions(ETTs). Reference [20] proposes an SDN-based blockchain energy Internet distributed energy transaction scheme. Reference [21] proposed a scalable dynamic multi-agent hierarchical PBFT algorithm(SDMA-PBFT), which reduces the communication overhead from $O(n^2)$ to $O(nk * log(nk))$. Reference [22] proposed A voting reward and punishment scheme and its corresponding credit evaluation scheme and PBFT-based consistency and checkpoint protocol. Reference [23] proposed an algorithm based on skip Hash and dynamic weighted sharding and introduced an asynchronous consensus group mechanism to process cross-shard transactions effectively. Reference [24] proposed a

	Advantage	Disadvantage	Purpose
PoW	Resist 51% of the attack force. Completely decentralized.	Large computing power consumption. Slow block generation. Dependent on tokens.	The first generation consensus mechanism, the foundation of Bitcoin.
PoS	Energy-saving. The block time is relatively short.	Dependent on tokens. Easy to split.	Lower computational power consumption.
PBFT	Energy-saving. And do not rely on tokens.	Tolerates 1/3 malicious nodes. Slow block generation.	Able to tolerate Byzantine errors.
RAFT	Easy to implement. Energy-saving. Not rely on tokens.	Malicious nodes cannot be tolerated. Poor performance in high concurrency scenarios. Low throughput.	Solve the consistency problem of multiple copies in a distributed system.
SDMA-PBFT	Low communication consumption. No token dependence.	Low throughput. Poor fault tolerance.	Reduce the communication bottleneck.
DPoS	Energy-saving. Decentralization. Relatively high efficiency.	Low throughput. Dependent on tokens.	Solve the problem of the traditional POW mechanism and POS mechanism.
DPBFT	Primary node security selection. The throughput is relatively improved.	Low throughput. The primary node is easy to centralize.	The efficiency of the system is improved by reducing the probability of nodes doing evil.

FIGURE 1. Analysis of consensus mechanism.

competition-based proof of stake(CPoS) consensus mechanism, which can quickly remove forks under the premise of ensuring decentralization. Reference [25] proposed a consensus method based on the Practical Byzantine Fault Tolerance(PBFT) algorithm for multi-energy interactive entities. Reference [26] proposed a blockchain consensus protocol based on the quality of service(QoS). Reference [27] proposed Jointgraph, a DAG-based Byzantine fault-tolerant consensus algorithm for consortium blockchains. Reference [28] proposes a distributed new energy information interconnection model based on the blockchain consensus mechanism. Reference [29] proposed a DPBFT algorithm suitable for the dynamic reputation of energy blockchain. The current research and analysis of the consensus mechanism are shown in Figure 1.

Although the above research solves the problems of low participation and low throughput of nodes in the consensus process, the problem of high transaction delay has not been completely solved and cannot meet the needs of large-scale energy transactions. Therefore, the existing market urgently needs to strengthen the consensus performance of the mechanism.

III. SYSTEM MODEL

A. CRSM RELATED ROLE DEFINITION

In CRSM, the consensus nodes are divided into different consensus domains. Each consensus domain votes the data in a prepared block. The primary node collects the voting information of all consensus domains and generates formal blocks. Then, the primary node broadcasts the formal blocks to the consensus node and storage node for storage. In this paper, the nodes are divided into four categories: monitoring node, primary node, consensus node, and storage node. Also, there is a consensus domain which divides the consensus resource slice and storage domain that stores block information. The relevant definitions are as follows:

1) MONITORING NODE (*Mon_Node*)

Introduced a supervisor in the CRSM consensus algorithm, which is called the monitoring node in this article, which is specially designed for the admission and exit of members of the alliance chain, and is suitable for those who need a supervisor (such as government and other official institutions), the monitoring node can significantly improve the efficiency of the blockchain system. It is primarily responsible for overseeing the actions of blockchain system members, including:

(1) Scanning node status: Receive heartbeat information periodically sent by other nodes, judge the node status from the heartbeat information, and randomly select the primary node from nodes with normal node status;

(2) Entry and exit of members of the blockchain system: The newly added member nodes of the blockchain system first register with the monitoring node, and can participate in consensus and broadcast after the monitoring node has approved;

(3) Assign node roles: Monitoring nodes can be configured with all node roles, such as consensus nodes, storage nodes, and the consensus domain and storage domain to which they belong.

(4) Query the list of consensus domains: scan the status of the consensus nodes in all consensus domains. If the status of the nodes in the consensus domain is reasonable, the consensus domain is the consensus domain where consensus can be performed;

(5) Start a new round of consensus: After verifying that the status of the storage nodes in the storage domain is consistent, notify the primary node to start a new round;

(6) Data consistency: After the storage nodes in the storage domain are synchronized, the monitoring nodes verify that they are synchronized to the latest height.

2) PRIMARY NODE (*Pri_Node*)

The primary node is randomly selected by the monitoring node from the consensus nodes in a healthy state. The primary node is responsible for counting the voting data that needs to be put into the preparatory block, and then broadcast to the consensus node; collect the voting results of the consensus node, generate a formal block, and send it to the storage node in the storage domain.

3) CONSENSUS NODE (*Con_Node*)

Votes the content of the received preparatory block and sends the voting result to the primary node. Then store the official blocks broadcast by the primary node.

4) STORAGE NODE (*Sto_Node*)

It does not undertake the task of voting during the consensus process. It is mainly responsible for storing the formal blocks

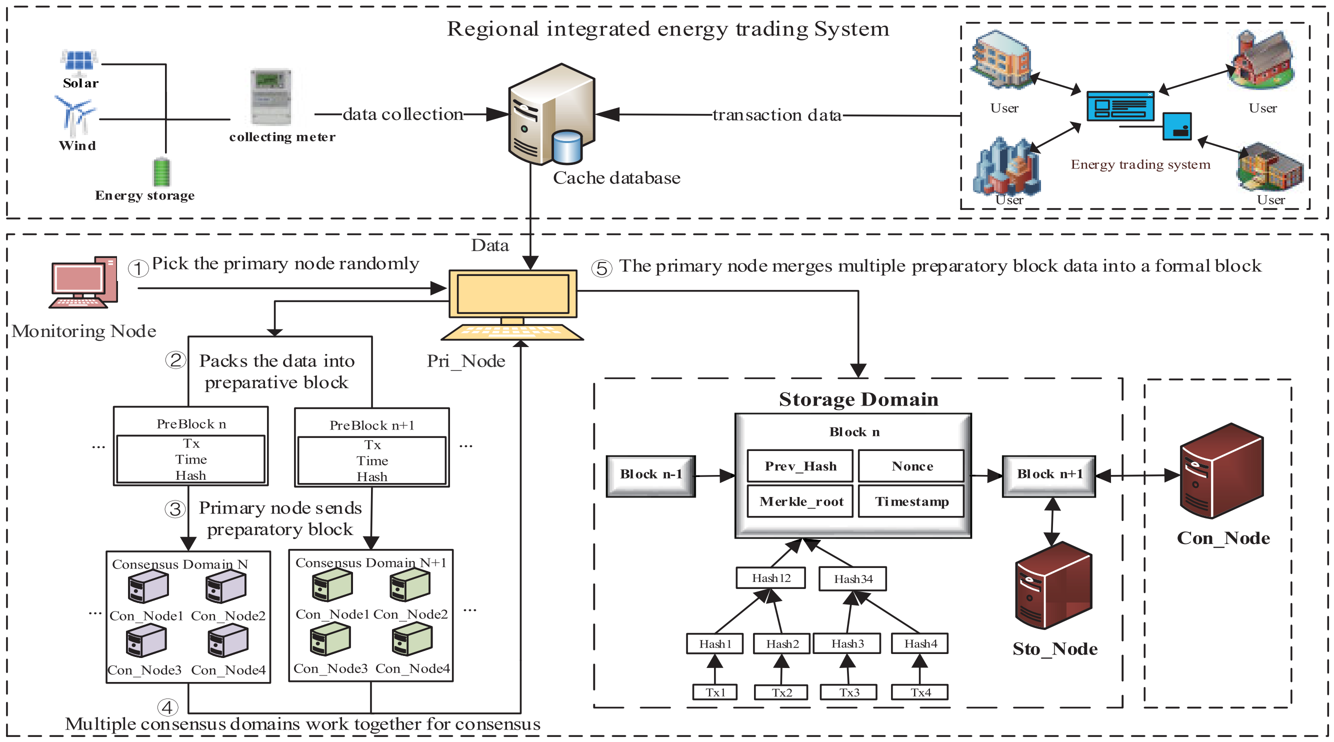


FIGURE 2. System model.

generated by the primary node after the consensus node has reached a consensus.

5) CONSENSUS DOMAIN (*Con_Domain*)

In order to improve the efficiency of consensus, this article proposes the concept of consensus domain. A consensus domain contains multiple consensus nodes, and multiple consensus domains can perform consensus concurrently, which effectively reduces the waste of consensus resources.

6) STORAGE DOMAIN (*Sto_Domain*)

For the consortium chain, this paper proposes the concept of storage domain. It is a collection of storage nodes. It is responsible for storing the official blocks sent by the primary node. Only the nodes in the storage domain need to participate in storage, which significantly reduces the communication resource consumption of other nodes.

B. CONSENSUS RESOURCE SLICING MODEL

In the energy trading market based on blockchain, users are not only consumers but also act like producers by managing their own distributed generation equipment (mainly photovoltaic and wind power generation), distributed energy storage facilities, and distributed loads. Users can use this generated electricity for trading and generate smart contracts automatically after the transaction is concluded. The smart contract should have the attributes of party identity, quota, price, and transaction time. At the same time, both parties sign with the private key to ensure the validity of the contract. After CRSM consensus, the contract is broadcast to the blockchain network for storage on the chain.

Due to the strict admission and exit characteristics of the alliance chain and the regional energy trading scenarios, this paper proposes a consensus resource slice model(CRSM) according to the characteristics of the alliance chain and the requirements of the energy blockchain, as shown in Figure 2, and then proposes the concepts of consensus domain and regulator to improve the efficiency of the blockchain system. In the CRSM, the consensus domain is a different consensus unit. A blockchain system can have multiple consensus domains, and each consensus domain can independently conduct consensus, and they do not affect each other. The supervisor is specially designed for the supervisory agencies in the alliance chain (such as government, official agencies, etc.).It can supervise the status of all nodes, configure consensus domains and storage domains, etc. Suppose the system has N nodes, D consensus domains, a storage domain, and a monitoring node. The monitoring node can allocate N nodes to D consensus domains, and then randomly select a primary node, and one of the nodes can belong to both the consensus domain and the storage domain. The energy trading system puts θ transaction data and power generation data that need to be linked into the Redis cache database. The primary node packs these data into k preparatory blocks, which are verified by D consensus domains($D = k$). Each consensus domain

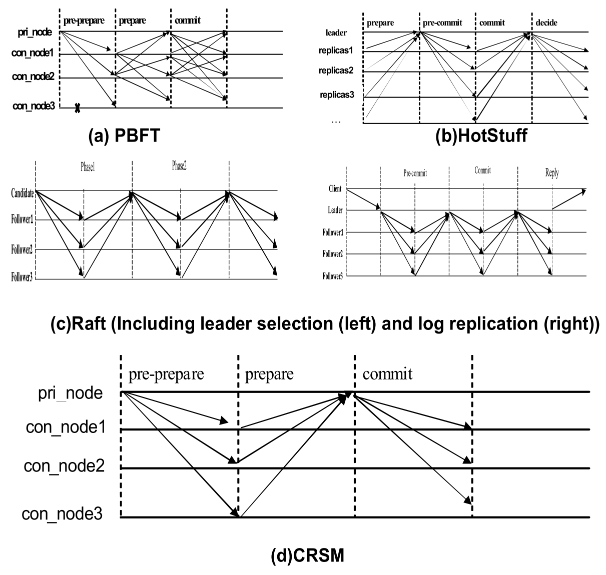


FIGURE 3. Comparison of conformance protocols.

only needs $\lceil\theta/k\rceil$ pieces of data. For voting, the consensus domain δ needs to vote on $(\delta-1)\lceil\theta/k\rceil \rightarrow \delta\lceil\theta/k\rceil$ pieces of data.

C. CONFORMANCE PROTOCOL ANALYSIS

The transaction data is packaged into the block after the consensus of the blockchain system, and the consensus node uses a consensus protocol to ensure the correctness of the storage record block of all nodes. There are two roles in the consensus protocol of the PBFT algorithm, Raft algorithm, HotStuff protocol, and CRSM algorithm: primary node (leader) and consensus node (replica node).

In the PBFT algorithm, the formula node not only needs to communicate with the primary node, but also with other consensus nodes, so the complexity is $O(n^2)$, and the consensus protocol process is shown in Figure 3(a). The specific process is as follows:

Pre-prepare: The primary node *pri_node* broadcasts a pre-prepared message to the consensus node *con_node*. The message should include blocks, timestamps, etc. Then the primary node *pri_node* enters the prepare state.

Prepare: The consensus node *con_node* enters the prepare state and broadcasts a prepare message to all nodes.

Commit: The consensus node *con_node* receives messages from other nodes and verifies them. If the transactions in the block, the correctness of the information, and the block are highly recognized, it sends approval feedback. After the node receives $(2f+1)$ acknowledgment feedback (counting itself), it indicates that the block has been officially chained and entered the commit state.

For the Raft algorithm, the completion of the consensus protocol requires completion of two stages: leader election and log replication. Although the complexity is $O(n)$, there is still room for optimization. The consensus protocol process is shown in Figure 3(c). The specific process is as follows:

Leader election: In the Raft algorithm, a node can only be in the leader, candidate, and follower at any time. At the same time, a strong leader is emphasized to simplify the whole process. Therefore, its log data stream can only be copied from the leader to the follower, and the log can't be transmitted between the followers.

Log replication: The client sends the request to the cluster, and if it is received by followers, it will forward to the leader, who will handle it uniformly. The leader will schedule these requests, inform all followers in order to ensure the consistency of the state of all nodes.

Reference [30] proposed HotStuff, a leader-based Byzantine fault-tolerant replication protocol. It only takes one stage to complete the consensus protocol, making the communication complexity and the number of replicas has a linear relationship, reaching $O(n)$. The consensus protocol process is as follows Shown in Figure 3(b). The specific process is as follows:

Prepare:$(n-f)$ the replica node sends the proposal to the leader.

Pre-commit: leaders receive $(n-f)$ current proposals, merge them into *prepareQC*, and broadcast *prepareQC* to $(n-f)$ replica nodes.

Commit: the replica node promises the proposal in *prepareQC* and sends it to the leader.

Decide: when leaders receive $(n-f)$ commitments, they will be merged into *commitQC* and sent to the replica node.

Different from HotStuff, CRSM data comes to the business layer, so there is no need for other nodes to send data to the master node during the preparation phase. At the same time, in the primary node collection votes stage, the primary node generates formal blocks after receiving $\frac{4}{5}n$ node's approval without waiting for the remaining nodes to vote, which not only ensures security but also avoids the possibility of network problems of the remaining nodes and slows down the process of consistency. Hence, the linear coefficient of the CRSM communication complexity and the number of nodes is smaller than that of the HotStuff protocol and the Raft algorithm. As shown in Figure 3(d),assuming there are N consensus nodes, the specific content is as follows:

Prepare: The primary node broadcasts a prepare message to N consensus nodes. The message should include the prepared block, timestamp, etc. Then the primary node enters the Confirm state.

Confirm: The N consensus nodes verify the prepared block, including verifying the parent hash and block height of the prepared block, and then sends the voting information to the primary node.

Commit: The primary node receives voting messages from consensus nodes, verifies them, and counts the voting results. After receiving $\frac{4}{5}N$ affirmative votes (including itself), the node indicates that it has passed the authentication, enters the commit state, and then the primary node will broadcast the generated formal block to all other nodes for record storage.

D. SAFETY ANALYSIS AND PROOF

We propose the CRSM rather than a perfect consensus protocol, so the security and decentralization of the original consensus protocol have not changed remarkably. The main purpose of introducing the slice model is to make efficient use of the existing nodes to improve the system performance, and to a large extent solve the problems related to the consensus efficiency of the current blockchain system.

1) COMPREHENSIVE ANALYSIS OF NODE VOTING PASS RATE

CRSM is a consensus algorithm optimized on the basis of the alliance chain. It has a strict access and exit mechanism, and the probability of nodes doing evil is small. Because the blockchain has a ternary paradox, it means that the blockchain system can have at most two of the three attributes of decentralization, security, and high performance. In order to ensure that the comprehensive index of the security and performance of the improved blockchain remains unchanged, we have increased the proportion of votes that need to be collected when voting.

The specific analysis is as follows: Suppose the total number of nodes is N, the number of consensus domains is D, the computing power of each node to verify data is δ, the safety factor to ensure security is β, and the number of nodes in each domain is $\lceil N/D \rceil$,x is the pass rate of node voting after the improvement. Assuming that the security of node voting is a positive trend, the computing power spent is a negative trend, S is a comprehensive index of safety and performance, and Sshould be similar before and after the algorithm is improved.

$$\begin{aligned} S_{before} = \beta^{N\frac{2}{3}} - N\delta \approx \beta^{x\frac{N}{D}} - \delta\frac{N}{D} = S_{after} \\ \beta^{x\frac{N}{D}} \approx \beta^{N\frac{2}{3}} - N\delta + \delta\frac{N}{D} \\ x\frac{N}{D} \approx log_{\beta}(\beta^{N\frac{2}{3}} + \delta\frac{N}{D}) \\ x \approx \frac{D}{N} log_{\beta}(\beta^{N\frac{2}{3}} + \delta\frac{N}{D}) \end{aligned} \quad (1)$$

Based on comprehensive considerations, in CRSM, $x = \frac{4}{5}$, which not only ensures that performance is improved in sharding mode, but also does not reduce the security of node verification too much.

2) MALICIOUS NODES TAMPERING WITH BLOCK INFORMATION WILL NOT TAKE EFFECT

Each node in the blockchain system records all the block information. Blocks in a blockchain are closely linked. In CRSM, the monitoring node is set up for the supervision department, so the malpractice of the monitoring node is not considered.

If a node's block data is tampered with, the node block information validation will not pass and will be kicked out of the consensus. The analysis is as follows: suppose a blockchain has α blocks, and there are N nodes and D consensus domains in the blockchain system. The computational power consumed to modify a block is γ. If a malicious node wants to tamper with the ρ block.

It can be seen from the above that the CRSM will generate A formal block as long as it collects $\frac{4}{5}\lceil N/D \rceil$ votes of approval. In other words, without considering the tampering of monitoring nodes, each consensus domain can only tolerate the tampering of $\frac{1}{5}\lfloor N/D \rfloor$ nodes, and CRSM can tolerate at most $(D-1)$ consensus domain tampering. Therefore, the maximum computing power that CRSM can tolerate tampering is f.

$$f = [\frac{1}{5}\lfloor N/D \rfloor * (D-1)]^{\gamma*(\alpha-\rho)} \quad (2)$$

In CRSM, more than 80% of the nodes agree to generate new blocks, and the security of blocks is higher. Before each round of consensus, the block of each node will be verified. If the verification fails, it will not be able to participate in the subsequent consensus. If the calculation force of tampering node exceeds f, the approval vote collected will be less than $\frac{4}{5}$, and no new block will be generated. This makes the blocks of the blockchain system hard to tamper with.

3) THE CRSM PROPOSED IN THIS PAPER CAN EFFECTIVELY PREVENT REPLAY ATTACK AND BIFURCATION

In the preparatory block phase, the primary node generates multiple preparatory blocks, then signs them, and then sends them to the corresponding consensus domain. Nodes in the consensus domain validate the signature of the preparatory block and vote on it if the validation passes. In the formal block phase, although there are multiple reserve blocks, the primary node consolidates the data passed in the multiple reserve blocks into a formal block according to the voting information, and then sends it to the consensus node for verification. If the verification passes, the other nodes store the block on the chain. The signature and validation between nodes are as follows.

1. Node signature

Assuming that the transmitted message is the k-th preparation block $M = [Pre\_Block_k, < Fa\_Hash, Height, Timestamp, Tx >]$ and the node $Node\_A$ needs to sign it, the element $g = e(P_1, P_{pub-1})$ in G_T is first calculated, and the random number $r = [1, N-1]$ is selected, then $w = g^r$ and the integer $h = H_2(M||w, N)$ is calculated. After that, the integer $l = (r-h)modN$ is calculated. If $l = 0$, the random number is selected again. When $l \neq 0$, $S = [l]ds_{Node\_A}$ is finally calculated, and the signature of the message M can be obtained (h, S).(Where Fa_Hash is the father hash of the formal block, $Height$ is the block height of the formal block, Tx is the data in the block, and $Timestamp$ is the time stamp).

2. Node verification signature Suppose that node $Node\_B$ receives the message M', and its signature is (h', S'). If the signature needs to be verified, first verify $h' \in [1, N-1]$ and $S' \in G_1$. if both are true, then calculate the element $g = e(P_1, P_{pub-1})$ in the group G_T, then calculate the element $t = g^{h'}$ and integer $h_1 = H_1(ID_{Node\_A}||hid, N)$ in the group G_T,

then calculate the element $P = [h_1]P_2 + P_{pub-s}$ in G_2 and the element $u = e(S', P)$ in G_T, then calculate the element $w' = u * t$ in G_T, and finally calculate $h_2 = H_2(M'||w', N)$, and compare with h'. if consistent, the verification is passed.

When the node receives the block, its signature will be strictly verified according to the signature algorithm, and only after all information is verified will it be stored on the chain. Besides, each message has a unique hash, based on which the node determines whether the message has been repeatedly accepted. If a duplicate message is found, it will refuse to receive the message unless the same piece of data is received in the same microsecond, which is not possible in a small energy trading scenario. Therefore, there is no bifurcation problem in CRSM, and replay attacks can be effectively prevented.

4) THE CRSM CONSTRUCTED IN THIS PAPER CAN EFFECTIVELY PREVENT SYBIL AND ECLIPSE ATTACKS

In blockchain networks, witch attacks act as multiple nodes to conduct malicious ticket brushing and counterfeiting. An eclipse attack modifies the node's configuration file so that it can only connect to evil nodes.

The blockchain built by CRSM is a licensing chain, so each node and entity must be approved to join the blockchain system, and the nodes all know each other. Each node and entity has public and private keys that are uniquely identified and assigned, and any broadcast and authentication blocks, transactions, and messages must be signed and authenticated using public and private keys; When a node receives any message, transaction, or block, it first uses the public and private keys to verify that the signature is correct and that the node broadcasting the message is in the license list. The node information interaction does not take effect if the signature verification does not pass or is not in the node list. The process of generating the public and private keys is as follows.

1.Randomly select two unequal prime numbers p and q.

2.Calculate the product n of p and q. $(n = pq)$

3.Calculating the Euler function $\varphi(n)$ of n. $(\varphi(n) = (p-1)(q-1))$

4.Randomly select an integer e. $(1 < e < \varphi(n),\ e\ and\ \varphi(n)\ are\ prime\ numbers\ for\ each\ other)$

5.The inverse element d of e for $\varphi(n)$ is calculated. $ed \equiv 1(mod\varphi(n)))$

6.Encapsulate n and e into public keys, n and d into private keys.

With nodes in the alliance chain knowing each other, it is difficult for an attacker to crack the RSA algorithm's public-private key verification mechanism. Besides, due to the existence of monitoring nodes, once the node's configuration file is changed, the link with the monitoring node will be lost, and it will be deemed as dropped and cannot participate in the consensus. Based on the above mechanism, CRSM can effectively prevent attacks such as Sybil and Eclipse.

5) IT CAN EFFECTIVELY PREVENT THE PRIMARY NODE FROM ABNORMAL

It can be seen from 1) that it is difficult for malicious nodes to tamper with block information, and other nodes will also verify information such as hash and block height when receiving block messages broadcast by the primary node. If the primary node is down or offline, the data and block information cannot be received during this period. In this case, the system will reselect the primary node and continue the previous operations. In CRSM, monitoring nodes are set up for government departments, official regulators, so that they can supervise the trading market. Moreover, the monitoring nodes do not directly participate in the consensus process, so this paper assumes that the monitoring nodes are reliable. Also, the function of the primary node in this paper is similar to that of the leader node in the PBFT algorithm and the Raft algorithm, so it will not increase centralization.

6) ACCORDING TO 1), THE CONSENSUS DOMAINS DOES NOT INCREASE THE ADDITIONAL RISK

Each round of the monitoring node selects one node from the consensus node N of all consensus domain D as the primary node of the current consensus. From 1) it can be seen that the node passing rate in each consensus domain needs to exceed 80% to generate a formal block. Therefore, the consensus domain mechanism will not bring additional risks on the basis of improving the consensus efficiency.

7) THE CRSM CONSTRUCTED IN THIS PAPER IS RELIABLE AND STABLE

Because CRSM is implemented in a local area network, it is physical aggregation. The network in the model is in good condition and can eliminate the phenomenon of network congestion. The nodes in each domain can maintain security for the negotiation process. The primary node sends the messages in each consensus domain. After receiving the messages, the consensus node will validate it. After passing the verification, the consensus is carried out concurrently, which can eliminate the hidden dangers brought by subdomains.

Furthermore, in a CRSM based license chain, each participant knows each other. Moreover, they have no incentive to attack the system, because they will be easily caught and get nothing. Once a participant is found to have a malicious attack, he will be expelled from the alliance by the monitor node (the monitor node can remove this malicious node from the node list), and all honest participants will no longer communicate with him. The total number of malicious participants is assumed to be no more than one-third. So, in the worst-case scenario, all the malicious actors are kicked out, and the system can function and stay active

E. CRSM ALGORITHM ROUND OF CONSENSUS PROCESS

The consensus domain design of the CRSM algorithm theoretically requires that each consensus domain is the same, has the same function and has the same status, and is composed of logically independent consensus nodes. All consensus domains work in parallel, and the storage domain is only responsible for storing block data. The detailed flow of the CRSM algorithm is as follows.

S1. Verify Block: The monitoring node scans the block of the consensus node $\{P_1, P_2, P_3, \ldots P_N\}$ and judges whether they are consistent. If they are not consistent, they cannot participate in the consensus process. They must be synchronized to participate in the consensus. The specific algorithm is shown in Algorithm 1.

S2. Configure the Consensus Domain: The monitoring node reads the number of common domains in the configuration file. When the consensus node $\{P_1, P_2, P_3, \ldots P_N\}$ joins the blockchain system, it is allocated to each consensus domain in turn. In addition, the administrator can also modify the configuration in the background database and assign it manually.

Algorithm 1 Block Verification

```
Input: Nodes list $\{P_1, P_2, P_3, \ldots P_N\}$
Output: verification result
1. each $\{P_1, P_2, P_3, \ldots P_N\}$
2.     get($P_{N-}$ block) $\rightarrow$ $Domain_N$
3. if ($P_N - block = \{P_1, P_2, P_3, \ldots P_{\mathrm{N}}\}_{block}$)
4 .     return true
5.else
6.synchronization block
```

*S3. Generate Preparatory Blocks:*Primary node fetches θ transaction information that requires consensus from the Redis cache database, and Generate k preparatory block $[Pre\_Block_k, < Fa\_Hash, Height, Timestamp, Tx >]$, which is consensus by $ConDomain_D$ consensus domains $(D = k)$. Each consensus domain only needs to vote for $\lceil\theta/k\rceil$ data. Consensus domains δ need to vote on $(\delta-1)\lceil\theta/k\rceil \rightarrow \delta\lceil\theta/k\rceil$ data.

S4. Data Synchronization: The primary node sends the generated k preparatory blocks $[Pre\_Block_k, < Fa\_Hash, Height, Timestamp, Tx >]$ to the $ConDomain_D$ consensus domain$(D = k)$, respectively. The consensus nodes in the consensus domain verify the received preparatory blocks, respectively. The specific algorithm is shown in Algorithm 2.

Algorithm 2 Data Synchronization

```
Input : Preparation block
Output: Sync results
1. for ($k = 1; k <= pre\_len; k++$)
2.     Send ($P_1\_height$) $\rightarrow$ $Domain_D$
3. Nodes $\{P_1, P_2, P_3, \ldots P_N\}$ verification
5. if ($P_N - height == Lastblock\_height + 1$)
6.     return true
7.else return false
```

S5. Voting: The consensus domain $ConDomain_D$ votes on the transaction data in the preparatory block $[Pre\_Block_k, < Fa\_Hash, Height, Timestamp, Tx >]$,$(D = k)$, votes for the data of typical transactions, and votes for the data of abnormal transactions. For example, inconsistent block heights, inconsistent block hashes, etc.

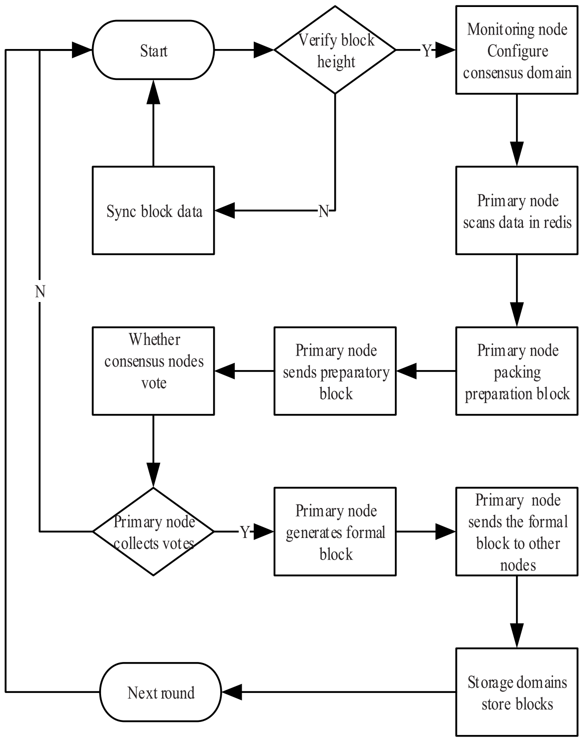

Start
Verify block height
Y
Monitoring node Configure consensus domain
Sync block data
N
Primary node scans data in redis
N
Whether consensus nodes vote
Primary node sends preparatory block
Primary node packing preparation block
Primary node collects votes
Y
Primary node generates formal block
Primary node sends the formal block to other nodes
Next round
Storage domains store blocks

FIGURE 4. **Algorithm flow. Errors in every link of S1-S7 will cause this round of consensus to fail. Data in this round without consensus in the consensus domain will be put into the next round to continue consensus.**

S6. Generate a Formal Block: each consensus domain sends the voting result to the primary node, and the primary node is responsible for statistics. After receiving $\frac{4}{5}$ of the node approval votes, the formal block is generated. And the data passed by each consensus domain is packaged to form a formal block.

S7. Formal Block Storage: The primary node broadcasts the generated formal block to the consensus node and all storage nodes in the storage domain, and each node performs storage. The detailed algorithm flow is shown in Figure 4.

IV. EXPERIMENT AND ANALYSIS

A. EXPERIMENTAL PLATFORM AND ENVIRONMENT

In order to test and analyze the performance of the CRSM algorithm proposed in this paper, we establish a CRSM algorithm experimental platform using the Java language. The underlying technology of the experimental platform includes distributed data storage, gRPC communication mechanism, CRSM consensus algorithm, and the Spring framework of the Java language. It consists of three parts: business domain, consensus domain, and storage domain. The business domain provides business data. In this platform, it is responsible for sending unformatted general-purpose on-chain transaction data to the consensus domain. The consensus domain is based on the CRSM consensus algorithm set out in the present article. Furthermore, the storage domain is responsible for storing the formal blocks generated by the consensus domain to the database.

In order to ensure smooth network communication and reduce the interference of network problems to this experiment, our experimental platform is deployed in a small LAN without access to the Internet. Affected by the number of hosts in the laboratory, 20 nodes were enabled in this experiment. The CPU frequency of the monitoring node is 3.20GH, and the memory size is 8G, and the CPU frequency of the other nodes is 2.20GH, and the memory size is 8G. Block generation interval time and node sending heartbeat time are reasonable values, and the communication between nodes is based on gRPC technology.

B. TPS COMPARATIVE ANALYSIS

Transaction throughput is an important performance index to measure the distributed system. The throughput of a system is usually determined by the number of concurrent transactions and transactions per second (TPS). We use this to test and compare the performance of several algorithms.

$$TPS = \frac{Tran\_len}{\Delta T} \tag{3}$$

Where $Tran\_len$ refers to the number of transactions contained in a block; ΔT refers to the generation time of a block, which generally includes three stages: prepared block generation, data consensus, and formal block generation.

$$\Delta T = Time_{pre\_block} + Time_{consensus} + Time_{for\_block} \tag{4}$$

Due to the difference between consensus algorithms, the interval between blocks is different. To ensure the accuracy of the experiment and the stability of the system, we send 1000 simulated transaction data to the cache database every second to ensure that the blockchain system can get enough data for consensus each time. After the test, the total number of transactions and block data is calculated uniformly. In the case of the different number of nodes, we tested and analyzed the time to generate 20 consecutive blocks by the Raft algorithm, the PBFT algorithm, the improved Hybrid algorithm of reference [31], and the CRSM in this paper. The analysis results are shown in Figures 5 and 6. It can be seen in figure 5 and Figure 6 that the peak and average throughput of CRSM is significantly improved compared with other consensus algorithms under the different number of nodes. This is because a complete consensus includes three stages: preparation block generation, data consensus, and formal block generation. We test the processing process of 1000 energy transaction data by four nodes. We find that the three stages of reserve block generation, consensus generation, and formal block generation share 103ms. The longest time is the consensus process 76ms, followed by reserve block generation 15 ms, and finally, formal block generation 12ms. The consensus process of transaction takes 73.7% of the whole block generation cycle. The CRSM in this paper

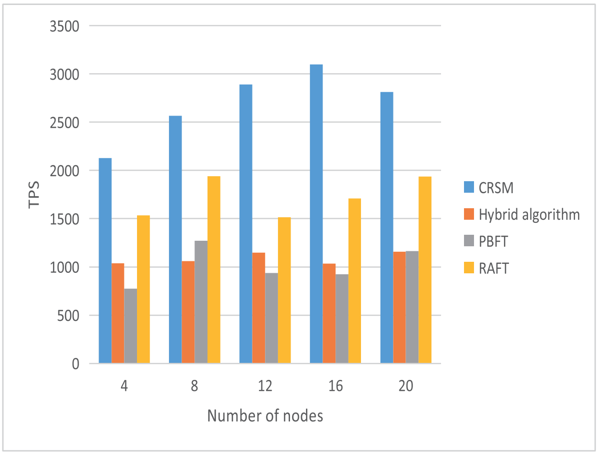


FIGURE 5. TPS comparison under different number of nodes.

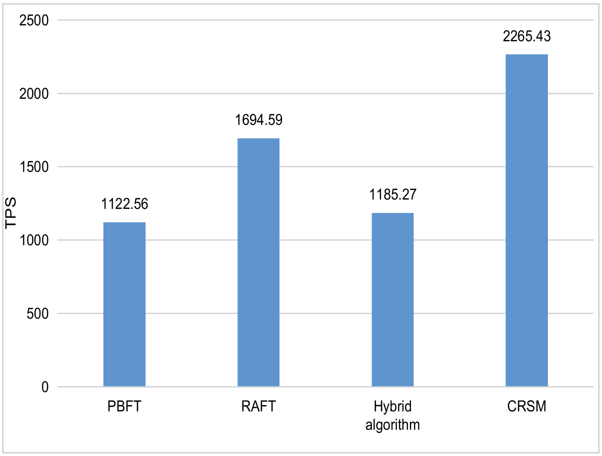


FIGURE 6. Comparison of average TPS of different consensus algorithms.

dramatically optimizes the consensus phase so that TPS will be significantly improved.

C. VERTICAL SCALABILITY ANALYSIS

The common problem from the PBFT algorithm to the consensus mechanism based on its improvement is that the consensus process requires a large number of inter-node communication. The CRSM algorithm reduces the number of consensus node communications in a consensus process, but the inter-node communication still consumes many resources. To test the impact of the number of nodes on the performance of the blockchain system, we tested the TPS changes of the number of consensus nodes in the consensus domain under different block sizes when the number of consensus domains is constant, as shown in Figure 7, where the abscissa represents the number of rounds, and the ordinate represents the TPS. It can be found that when the number of domains is fixed, as the number of nodes increases, the TPS of CRSM is lower. This is because, in the CRSM, when the number of domains is fixed, each additional node in the domain will increase the communication of threads in the stage when the primary node sends the preparatory block

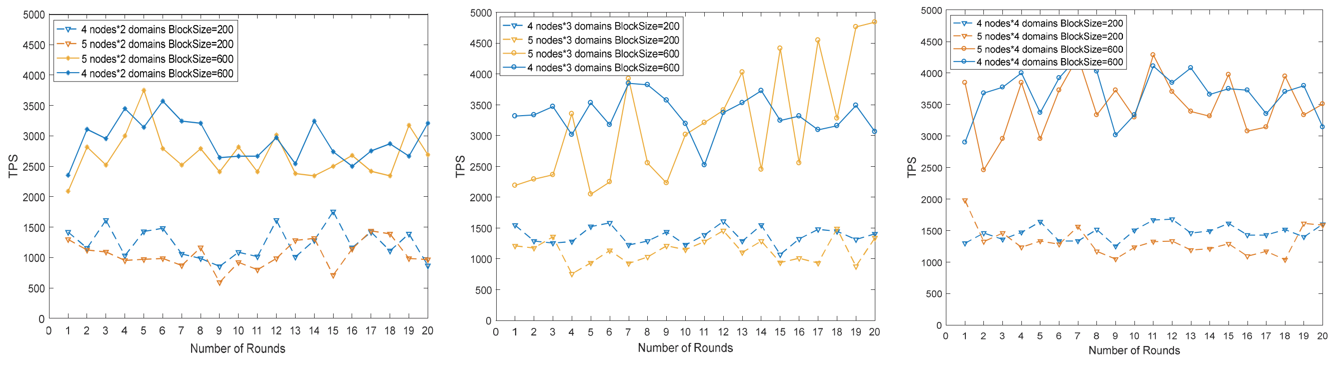


FIGURE 7. TPS comparison of 2(left),3(middle),4(right)consensus domains.

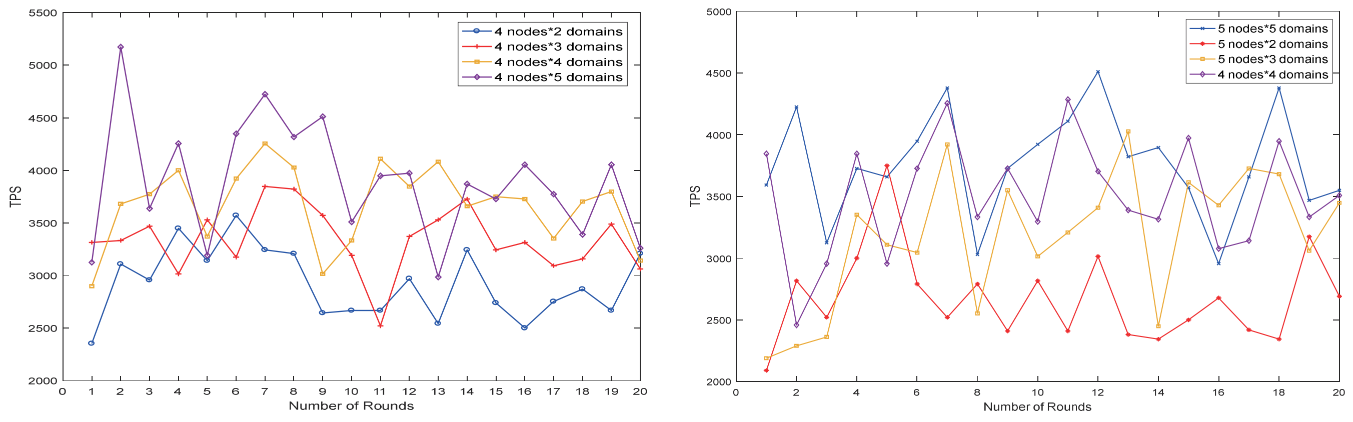


FIGURE 8. TPS comparison of 4(left),5(right) nodes in domain.

to each consensus node and collects the voting information. During the consensus phase, the longer it takes to maintain consistency across nodes. So the higher the number of nodes in the domain, the higher the communication consumption with the primary node, the more time the consensus process takes, and the lower the system throughput.

D. HORIZONTAL SCALABILITY ANALYSIS

In this paper, CRSM improves the performance of blockchain systems by partitioning consistent resources. In order to test the horizontal scalability of CRSM, we tested the relationship between the number of consensus domains and TPS under different node Numbers when the block size was 600. As shown in Figure 8, the abscissa represents the number of rounds, and the ordinate represents TPS. It can be found that with the increase of domain number, the data processing speed of CRSM increases, while TPS increases. However, the increasing speed becomes slower and slower, and TPS tends to be stable. This is because while keeping the number of nodes in the domain unchanged, each additional consensus domain will increase a concurrent consensus process to process the data, thus improving the consensus speed of the blockchain system in the consensus phase. But as the consensus domain increases, so does the time it takes for the master node to distribute and merge the reserve blocks, so the increase in TPS becomes smaller and smaller until it flattens out. Therefore, when the amount of consensus data remains unchanged, as the number of consensus domains increases, the overall consensus time cost decreases and the TPS of the CRSM system increases until it becomes stable.

E. BLOCK SCALABILITY ANALYSIS

In the CRSM algorithm, the number of data bars in a block (namely the block size) can be configured according to business needs to optimize the efficiency of the blockchain system. To study the impact of block size on the processing speed of the blockchain system, the block scalability is tested and analyzed. Specifically, we tested the processing speed of different transaction quantities in a block under different conditions. Under different conditions, the size of each block was tested for 20 rounds. TPS was averaged, and the value of the error bar was the standard deviation value of throughput of 20 blocks, as shown in Figure 9. The abscissa represents the different number of nodes and domains, and the ordinate represents TPS.

It can be found that in different cases, the TPS of a block with 600 transactions is much larger than 200 transactions in a block, but not much higher than 500 transactions in

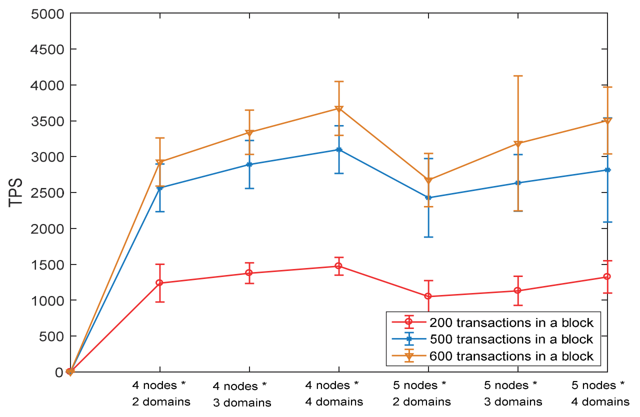


FIGURE 9. TPS comparison of different transaction numbers in a block.

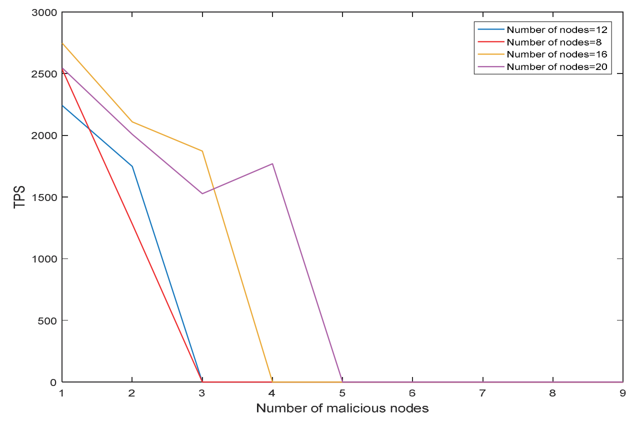


FIGURE 10. TPS trend under different number of malicious nodes.

a block. This is because a node's data processing speed is fixed. Before the number of transactions in the block is just at the critical value of node performance, TPS will gradually increase. When it approaches the critical value of node performance, the processing speed will gradually become stable and will not increase anymore. Even when the amount of data exceeds the node performance threshold, the BLOCK chain system TPS is reduced due to the blocking problem caused by the large amount of data.

F. SAFETY ANALYSIS

According to the theory and algorithm design in this paper, once the node data and configuration files are tampered with, they will be kicked out of the consensus and regarded as invalid nodes. To test the fault tolerance of the system, we separately tested the random disconnection of some common nodes when the total number of nodes N is 8,12,16,20, and the change of system throughput, as shown in Figure 10. The abscissa represents the number of malicious nodes and the ordinate represents TPS.

It can be seen from Figure 10, when the number of malicious nodes is greater than about one-third of the total number of nodes, the system throughput is 0, which means that the system no longer carries out consensus, does not generate new blocks, and prevents too many malicious nodes from tampering with the data before the system. Also, we can also see that the throughput of the system decreases as the number of malicious nodes increases. This is because, in CRSM, the limit time will be set, after which the node feedback information is not received, the node will be judged to be invalid. If the vote of the remaining node $V \geq \frac{4}{5}N$ is collected, the formal block will continue to be generated; otherwise, the consensus will fails.

V. CONCLUSION AND FUTURE WORK

Recently, the influence of blockchain technology in China has reached a new peak and has been applied in various fields. However, a single consensus mechanism cannot adapt to all application scenarios. In the field of distributed energy trading, the particularity of electric data has higher requirements on data transmission efficiency, and the existing consensus algorithm cannot adequately meet such requirements. In this paper, CRSM is proposed to solve this phenomenon. The simulation results show that CRSM can effectively improve the efficiency of the blockchain system and meet the requirements of real-time in distributed energy transactions.

In addition, the applicability of CRSM in other fields still needs further study, but it can provide a reference for future research on other application scenarios. For CRSM, further work will be done on security and efficiency, such as partitioning nodes based on game theory to ensure consistency of nodes within each consensus domain. And study alternative monitoring nodes to deal with emergencies.

REFERENCES

[1] P. Siano, G. De Marco, A. Rolan, and V. Loia, "A survey and evaluation of the potentials of distributed ledger technology for Peer-to-Peer transactive energy exchanges in local energy markets," *IEEE Syst. J.*, vol. 13, no. 3, pp. 3454–3466, Sep. 2019, doi: 10.1109/JSYST.2019.2903172.

[2] N. Ul Hassan, C. Yuen, and D. Niyato, "Blockchain technologies for smart energy systems: Fundamentals, challenges, and solutions," *IEEE Ind. Electron. Mag.*, vol. 13, no. 4, pp. 106–118, Dec. 2019, doi: 10.1109/MIE.2019.2940335.

[3] M. Swan, *Blockchain: Blueprint for a New Economy*. Newton, MA, USA: O'Reilly Media, 2015.

[4] D. Xia, "ETSB: Energy Trading System Based on Blockchain," in International Conference on *Proc. Secur. Intell. Comput. Big-data Services*, 2018, pp. 673–684, doi: 10.1007/978-3-030-16946-6_55.

[5] S. Zhu, "The development of energy blockch ain and its implications for China's energy sector," *Resour. Policy*, vol. 66, Dec. 2020, Art. no. 101595.

[6] I. Dimobi, M. Pipattanasomporn, and S. Rahman, "A transactive grid with microgrids using blockchain for the energy Internet," in *Proc. IEEE Power Energy Soc. Innov. Smart Grid Technol. Conf. (ISGT)*, Washington, DC, USA, Feb. 2020, pp. 1–5, doi: 10.1109/ISGT45199.2020.9087739.

[7] Y. Yuan, "Development status and prospect of blockchain consensus algorithm," *Acta Automatica Sinica*, vol. 44, no. 11, pp. 2011–2022, 2018.

[8] Q. Shao, "Block chain technology: Architecture and progress," *J. Comput. Sci.*, vol. 41, no. 5, pp. 969–988, 2008.

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

[10] B. Wang, M. Dabbaghjamanesh, A. Kavousi-Fard, and S. Mehraeen, "Cybersecurity enhancement of power trading within the networked microgrids based on blockchain and directed acyclic graph approach," *IEEE Trans. Ind. Appl.*, vol. 55, no. 6, pp. 7300–7309, Nov. 2019, doi: 10.1109/TIA.2019.2919820.

[11] M. A. Ferrag, "Blockchain technologies for the Internet of Things: Research issues and challenges," in *IEEE Internet Things J.*, vol. 6, no. 2, pp. 2188–2204, Apr. 2019, doi: 10.1109/JIOT.2018.2882794.

[12] M. J. Ashley and M. S. Johnson, "Establishing a secure, transparent, and autonomous blockchain of custody for renewable energy credits and carbon credits," *IEEE Eng. Manag. Rev.*, vol. 46, no. 4, pp. 100–102, Dec. 2018, doi: 10.1109/EMR.2018.2874967.

[13] Z. Dong, F. Luo, and G. Liang, "Blockchain: A secure, decentralized, trusted cyber infrastructure solution for future energy systems," *J. Modern Power Syst. Clean Energy*, vol. 6, no. 5, pp. 958–967, Sep. 2018, doi: 10.1007/s40565-018-0418-0.

[14] T. M. Fernandez-Carames and P. Fraga-Lamas, "Towards post-quantum blockchain: A review on blockchain cryptography resistant to quantum computing attacks," *IEEE Access*, vol. 8, pp. 21091–21116, 2020, doi: 10.1109/ACCESS.2020.2968985.

[15] A. Hafid, A. S. Hafid, and M. Samih, "New mathematical model to analyze security of sharding-based blockchain protocols," *IEEE Access*, vol. 7, pp. 185447–185457, 2019, doi: 10.1109/ACCESS.2019.2961065.

[16] M. Zhaofeng, W. Xiaochang, D. K. Jain, H. Khan, G. Hongmin, and W. Zhen, "A blockchain-based trusted data management scheme in edge computing," *IEEE Trans. Ind. Informat.*, vol. 16, no. 3, pp. 2013–2021, Mar. 2020, doi: 10.1109/TII.2019.2933482.

[17] C. T. Nguyen, D. T. Hoang, D. N. Nguyen, D. Niyato, H. T. Nguyen, and E. Dutkiewicz, "Proof-of-Stake consensus mechanisms for future blockchain networks: Fundamentals, applications and opportunities," *IEEE Access*, vol. 7, pp. 85727–85745, 2019, doi: 10.1109/ACCESS.2019.2925010.

[18] A. Sheikh, V. Kamuni, A. Urooj, S. Wagh, N. Singh, and D. Patel, "Secured energy trading using byzantine-based blockchain consensus," *IEEE Access*, vol. 8, pp. 8554–8571, 2020, doi: 10.1109/ACCESS.2019.2963325.

[19] S. Wang, A. F. Taha, J. Wang, K. Kvaternik, and A. Hahn, "Energy crowdsourcing and Peer-to-Peer energy trading in blockchain-enabled smart grids," *IEEE Trans. Syst., Man, Cybern. Syst.*, vol. 49, no. 8, pp. 1612–1623, Aug. 2019, doi: 10.1109/TSMC.2019.2916565.

[20] X. Lu, "Blockchain-based distributed energy trading in energy Internet: An SDN approach," *IEEE Access*, vol. 7, pp. 173817–173826, 2019, doi: 10.1109/ACCESS.2019.2957211.

[21] L. Feng, H. Zhang, Y. Chen, and L. Lou, "Scalable dynamic multi-agent practical byzantine fault-tolerant consensus in permissioned blockchain," *Appl. Sci.*, vol. 8, no. 10, p. 1919, Oct. 2018.

[22] Y. Wang, S. Cai, C. Lin, Z. Chen, T. Wang, Z. Gao, and C. Zhou, "Study of Blockchains's consensus mechanism based on credit," *IEEE Access*, vol. 7, pp. 10224–10231, 2019, doi: 10.1109/ACCESS.2019.2891065.

[23] J. Pan, "Blockchain dynamic sharding model based on jump hash and asynchronous consensus group," *Comput. Sci.*, vol. 47, no. 3, pp. 237–280, 2020.

[24] Y. Song, "A blockchain consensus mechanism based on voting rights competition [J / OL]," *J. Shandong Univ.* , vol. 5, pp. 1–8, Dec. 2019.

[25] D. Wang, "Multi-energy interaction subject consensus mechanis m based on practical Byzantine fault-tolerant algorithm," *Autom. Electr. Power Syst.*, vol. 43, no. 9, pp. 41–49, 2019.

[26] B. Yu, "Proof-of-QoS: QoS based blockchain consensus protocol," *Comput. Secur.*, vol. 87, Nov. 2019, Art. no. 101580, doi: 10.1016/j.cose.2019.101580.

[27] F. Xiang, W. Huaimin, S. Peichang, O. Xue, and Z. Xunhui, "Jointgraph: A DAG-based efficient consensus algorithm for consortium blockchains," *Softw., Pract. Exper.*, pp. 1–13, 2019, doi: 10.1002/spe.2748.

[28] X. Fang, "Research on block chain consensus mechanism under di stributed new energy access," *Zhejiang Electr. Power*, vol. 7, pp. 1–6, Dec. 2016.

[29] W. Cai, W. Jiang, K. Xie, Y. Zhu, and T. Shen, "Dynamic reputation–based consensus mechanism: Real-time transactions for energy blockchain," *Int. J. Distrib. Sensor Netw.*, vol. 16, no. 3, 2020, Art. no. 155014772090733.

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

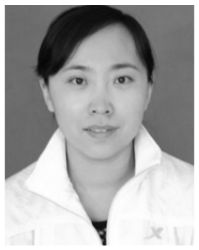
[31] Y. Wu, P. Song, and F. Wang, "Hybrid consensus algorithm optimization: A mathematical method based on POS and PBFT and its application in blockchain," *Math. Problems Eng.*, vol. 2020, pp. 1–13, Apr. 2020, doi: 10.1155/2020/7270624.

MENG HU is currently pursuing the master's degree with the Kunming University of Science and Technology. His research areas include blockchain, peer-to-peer transactions, and the power Internet of Things.

TAO SHEN (Member, IEEE) received the Ph.D. degree from the Illinois Institute of Technology, in 2013. He is currently a Professor and a Deputy Dean of the College of Information Engineering and Automation, Kunming University of Science and Technology. His research interests include blockchain technology, artificial intelligent, and the Internet of Energy.

JINBAO MEN received the master's degree from the University of Coventry, in 2016. He is currently an Intermediate Engineer with Beijing Zhongdian Puhua Information Technology Company, Ltd. His research areas include blockchain technology, artificial intelligence, and the Internet of things.

ZHUO YU received the Ph.D. degree from the Beijing University of Aeronautics and Astronautics, China, in 2011. He is currently a Senior Engineer with Beijing Zhongdian Puhua Information Technology Company, Ltd. His research directions include blockchain, artificial intelligence, VR/AR, GIS, BPM, and information consulting.



YINGLI LIU received the Ph.D. degree from the Kunming University of Science and Technology, in 2017. She is currently a Lecturer with the College of Information Engineering and Automation, Kunming University of Science and Technology. Her research areas include machine learning, natural language processing, and materials genome projects.