Safe and Efficient Delegated Proof of Stake Consensus Mechanism Based on Dynamic Credit in Electronic Transaction

*Mingjie Zhao, Cheng Dai, Bing Guo**

College of Computer Science, Sichuan University, China
zhaomingjie@stu.scu.edu.cn, daicheng@scu.edu.cn, guobing@scu.edu.cn

Abstract

The sector of electronic transactions based on blockchain technology is expanding quickly thanks to the reliability of the consensus mechanism, which offers a new technical path for developing the financial industry. The consensus algorithms cannot support the safety and efficiency required for electronic transaction scenarios as they are right now. We propose an enhanced DPoS algorithm based on dynamic credit (DC-DPoS) to address the aforementioned issue. It has a system for node classification, dynamic credit evaluation, and a consistency algorithm that combines voting and random selection. According to theoretical research and simulated experiments, our algorithm improves security while achieving reduced latency and greater throughput. It presents a fresh thought for enhancing the functionality of the blockchain system while also satisfying the fundamental technical criteria of electronic transactions.

Keywords: Blockchain, Consensus algorithm, Delegated Proof of Stake, Electronic transaction, Dynamic credit

1 Introduction

In electronic transactions, blockchain technology has become widely employed. Each node in the blockchain is equal, there is no superior-subordinate relationship between nodes, and all nodes cooperate to keep an account book. To ensure the consistency of the account book, a consensus method must be designed in advance; this allows trustworthy transactions to be implemented without the need for outside organizations. Blockchain's distributed architecture decreases system operating and maintenance costs while enhancing security compared to traditional centralized servers. Electronic transactions are becoming more widely recognized and trusted thanks to blockchain security. For instance, the market capitalization of Bitcoin, a typical blockchain digital currency, has surpassed $100 billion, and it is now the digital currency with the largest user base ever. The significance of blockchain to electronic transactions has been strongly endorsed by the International Monetary Fund (IMF).

In 2008, Satoshi Nakamoto first came up with the concept of blockchain and published it in the paper "Bitcoin: A Peer-to-Peer Electronic Cash System" [1]. As its name implies, a blockchain is a chain structure composed of blocks. Each block maintains relevance by storing the previous block's hash value so that the blockchain is a naturally traceable data structure. Take Bitcoin as an example, as shown in Figure 1, each transaction has a hash value, which is stored using the Merkle tree structure, and the root hash value of the Merkle tree is calculated by merging. If the content in the block has been tampered with, that will be detected. Suppose you want to tamper with the contents of the block, and the block header of each block contains its previous hash value. If you're going to be undetected, you must recalculate all the hash values from the current block to the last block of the blockchain, which is almost impossible.

The advantages of blockchain enable it to be used in many fields, such as electronic transactions [2], federal learning [3], the Internet of Things (IoT) [4-8], 5G [9], smart healthcare [10], crowdsensing systems [11], Peer-to-Peer energy trading (P2P-ET) [12], circuit copyright protection [13] and data query [14]. In electronic transactions, Chunchi Liu has designed a three-layer segmented blockchain network. It has established an independent transaction settlement system under the condition of meeting the Internet of Things' e-commerce standards [15]. Meng Li and others have designed an e-commerce privacy protection blockchain system that can be accessed across platforms and rated anonymously [16].

The first main problem with blockchain technology in electronic transactions is transaction security. For example, in Bitcoin, any node can view the transaction history, malicious nodes analyze users' privacy information through transaction history, design heuristic algorithms to launch the transaction users, and even introduce the real identity of users. Once the user's financial privacy information is leaked, it will bring a lot of losses to the user when commercial opponents or lawbreakers use it [17]. Second is the efficiency in the process of the transaction. The electronic transaction requires that the system has high timeliness, the size of a block in the current Bitcoin network is 1MB, which can contain 4,000 transactions. A block is generated every 10 minutes equally, and seven transactions are processed per second, the throughput rate is too low. If you increase the block size, the time it takes for a block to transmit through the network will be longer, and the probability of forks will improve. Therefore, it is unsuitable for application scenarios of real-time transactions and large transaction volumes, such as financial institutions like banks and securities companies.

*Corresponding Author: Bing Guo; E-mail: guobing@scu.edu.cn
DOI: 10.53106/160792642023012401012

The consensus algorithm is the core part of the blockchain system; how to reach an agreement among each node in the blockchain is defined by the consensus algorithm, including selecting block packager nodes and verifying the correctness of blocks. It will directly affect the security, delay, transaction throughput, and other performance of the whole blockchain system. The existing consensus algorithms cannot meet the requirements of high security and timeliness in electronic transaction scenarios. In order to solve these problems, we have improved the consensus algorithm. By designing mechanisms such as selecting block packaging nodes according to node credit classification nodes, combining node credit and a random selection, and voting for the correctness of the generated blocks, our algorithm improves the performance of the security and efficiency so that it can be used in electronic trading scenarios. We propose a consensus algorithm based on dynamic credit: DC-DPoS (Dynamic Credit-Delegated Proof of Stake). The specific contributions of this paper are as follows:

1) First of all, we propose a node authority classification model based on node historical behaviour to evaluate node credit, which ensures the system's security and reduces the number of nodes participating in consensus, thus improving the communication efficiency of the system. It is also designed so that the node can dynamically adjust its authority by increasing or decreasing the investment deposit.

2) A method of randomly selecting block proposer is designed, which has the characteristics of unpredictability, and a single node cannot do evil so that the system can resist targeted malicious block proposer attacks and Self-packing, to improve the security and stability of the system.

3) Establish a mechanism for verifying generated blocks, after the block is packaged, verification nodes will check the validity of generated block, and if it fails, a special block is linked. This mechanism effectively avoids the double-spending attack and maintains the continuity of the block.

The rest of this paper is as follows: The second part introduces some consensus algorithms and their research advances, the third part describes the design of the DC-DPoS consensus protocol in detail, and the fourth part makes a theoretical analysis of the security and efficiency of proposed consensus protocols. The fifth part shows and analyzes the simulation results. The sixth part is the conclusion of this paper.

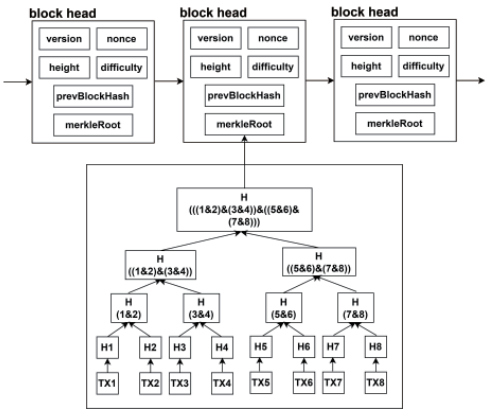

Figure 1. The structure of blockchain

2 Related Work

2.1 PoW

As the earliest consensus algorithm used in blockchain, Proof of Work (PoW) is an indirect consensus algorithm. It does not directly determine the content of the consensus, but through the competition for the right to pack blocks, the competition's winner decides the content of the consensus. The PoW algorithm in Bitcoin continuously calculates the nonce in the block and compares the result of the double SHA256 operation with the difficulty value of the current network. The workload proof is completed if the result is less than the difficulty value. To ensure the finality of the consensus and avoid the fork of the blockchain, the longest chain mechanism is used to solve this problem, and only the chain with a maximum workload is stored locally. Therefore, several blocks need to be identified continuously to maintain the final certainty of the consensus and avoid double-spending attacks. In general, it takes 10 minutes to calculate a block. If you need to confirm six blocks in succession, you must wait 60 minutes. If someone wants to tamper with the block content, he must successfully calculate the mathematical problem of each block from the block where the tamper is located to the current latest block, and the speed must be faster than the longest chain. This situation can only happen if the computing power controlled by the malicious node in the blockchain system exceeds the honest node. Moreover, massive calculations waste a lot of energy. S. Jiang [18] has predicted that the annual energy consumption of China's bitcoin blockchain will peak at 296.59 terawatt-hours in 2024, generating 130.5 million metric tons of carbon emissions without any policy intervention. The consensus algorithm currently used in Ethereum is the Ethash algorithm based on DAG [19], it resists the mining of ASIC special chips by increasing the complexity of the PoW consensus algorithm to increase the memory requirements in the computing process. Litecoin uses the Scrypt algorithm as the consensus algorithm, a memory-dependent algorithm combined with the SHA-256 algorithm [20]. Mostefa Kara et al. [21] proposed a CW-PoW consensus algorithm that can adapt to multiple environments. The algorithm improves the PoW consensus protocol by introducing several rounds of proof, which has a certain degree of robustness against two famous attacks: 51% attack and Sybil attack, and significantly improves energy consumption. Xidi Qu et al. published the first paper that applies federated learning to proof of workload in consensus algorithms, called federated learning proof algorithm (PoFL) [3], which puts the computing miner used to solve mathematical problems in PoW into federated learning and proposes a data transaction verification mechanism based on a reverse game-based mechanism, which effectively protects data privacy.

2.2 PoS & DPoS

The emergence of Proof of Stake (PoS) solves the weakness of the PoW consensus algorithm and competes for the power of packing blocks by owning the amount of coin age (coin age is the product of the number of tokens held by nodes and the holding time), It will not waste energy because of competitive computing power but use the existing equality

to reach a consensus and gain new value. Aggelos Kiayias et al. proposed an Ouroboros consensus algorithm based on PoS, whose security is comparable to PoW [22]. It provides a method based on physical resource proof, which gives the algorithm a qualitative efficiency advantage and proposes a new incentive mechanism to resist attacks such as selfish mining effectively. Bernardo David et al. proposed the Ouroboros Praos algorithm by improving the Ouroboros algorithm [23], which designed a forward secure digital signature and a new anonymous verifiable random function, developed a general combinatorial framework for analyzing semi-synchronous blockchains, and proved the security of the protocol in the random prediction model. Phil Daian et al. proposed a consensus algorithm called Snow White [24], which designs an end-to-end PoS system in a fully distributed and open participation network, which meets the needs of nodes joining dynamically and has good adaptability to the scenarios in which nodes frequently enter or leave in the network environment.

Delegated Proof of Stake (DPoS) is an improved PoS consensus algorithm based on voting. DPoS elects 101 block packaging representative nodes through miners, similar to the representative system. The voting weight of each miner is associated to the number of tokens he holds. The elected block packaging representatives need to pay specific tokens, and the representatives will take turns undertaking the task of packing blocks according to the rules. Block proposers can get a fee from the transaction fee. If the block proposer acts maliciously, the tokens will be removed from the block packing representative, confiscating the tokens.

Jiawen Kang proposed an enhanced DPoS consensus protocol based on DPoS [25], which is divided into two stages: the first stage uses a multi-weighted subjective logic scheme to select block packaging nodes according to credit and efficiency, and the second stage is to improve the enthusiasm of nodes to participate in consensus through incentive mechanism to prevent internal collusion among miners. Later, the efficiency and safety of the protocol are verified on the data set based on the real world. The problem of safe sharing of vehicle data in-vehicle networking is solved; Gang Sun et al. employed a decentralized consensus algorithm based on voting, which significantly improves the system's efficiency, but it does not fully evaluate the credit of nodes [26].

2.3 PBFT

L. Lamport and others put forward the question of Byzantine generals in 1982 [27]. The probable content is that different generals manage different parts of the Byzantine army, the generals achieved the goal of a unified battle by exchanging information with each other, but some generals are spies and always send the wrong message. The crux of the problem is how many spy generals at most in the army can still achieve the goal of unified action. The Byzantine general problem can be compared to exchanging information in the blockchain. Each node in the blockchain is a general, and a malicious node is a spy, so in the case of how many malicious nodes exist, the blockchain can still reach a correct consensus; this is the problem we need to solve. However, the BFT algorithm is too complex and expensive to communicate, so it is difficult to use it in practical issues. In 1999, Miguel Castro et al. proposed a practical Byzantine algorithm [28]. PBFT is a state machine copy replication algorithm based on BFT, which can still reach a consensus when there are less than (total number of nodes-1) / 3 malicious nodes in the system. The algorithm improves the efficiency of solving the Byzantine fault tolerance problem, has high throughput and stability, and can be applied in the actual situation. In recent years, many scholars have done a lot of research on the PBFT algorithm. Xu Yuan et al. proposed an improved PBFT algorithm based on reputation, which effectively improves the system's security, but the scalability of the system is still limited [29]. Wenyu Li et al. introduced an optimal double-layer PBFT and proved that the communication complexity of the algorithm is significantly reduced [30]. D. Data et al. offered for the first time a Byzantine resilient method that multiplies MV (matrix-vector) and CD (Coordinate Descent) and designs a specific coding matrix to resist opponent attacks [31]. Ryerson University's Jelana eliminated a single point of failure in PBFT implementation through a contention solution based on classical CSMA/CA technology [32].

To sum up, the security of PoW is very high, but its mining mechanism is too complex, so the time efficiency is too low. PoS and DPoS are very efficient but with the risk of power centralization and slightly less security. The security and timeliness of the PBFT algorithm are improved, but when a large number of nodes increase, its efficiency will be significantly reduced, so its scalability will be limited. Therefore, they can't meet the needs of some scenarios. We propose a consensus algorithm with improved security and efficiency: DC-DPoS (Dynamic Credit-Delegated Proof of Stake), to solve these problems.

3 Protocol Design

The consensus protocol DC-DPoS designed in this paper will be introduced in this part. Firstly, the protocol is briefly summarized, and then the specific research contents are described.

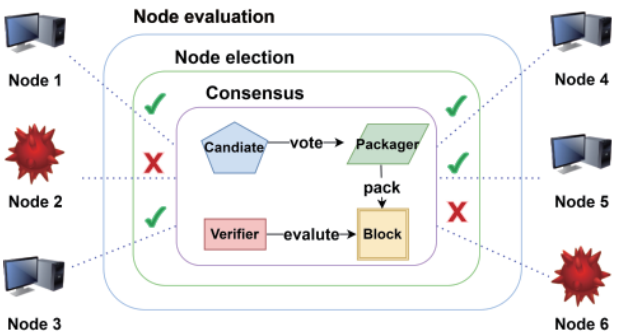

Figure 2. Overall hierarchical structure design

3.1 Overview

As described in the first part, many scholars have made plenty of designs and improvements to the blockchain consensus algorithm in recent years. Not each consensus algorithm is suitable for actual application scenarios because it has some efficiency and safety limitations. We propose a consensus algorithm: DC-DPoS (Dynamic Credit-Delegated

Proof of Stake), which aims to improve the security, fairness, and efficiency of consensus. DC-DPoS consists of three levels; the first is the node evaluation mechanism, according to the historical behaviour of nodes in the past consensus process, calculate its credit value and credit change rate, and get a comprehensive credit degree. The second level is node classification. Through the comprehensive credit degree of nodes, nodes are divided into verification nodes, candidate nodes, and ordinary nodes. Finally, enter the consensus stage, select the block proposer to generate the block through a voting and random selection method, and then verify the validity of the new block. The overall structure is shown in Figure 2.

3.1.1 Concepts

Let us introduce a few concepts that we will use later:

Genesis Block: The earliest built block in the blockchain has a version number that uniquely identifies the block. Except for this block, all other blocks have two identifiers: the previous block's hash value and its version number.

Public Key: In order to avoid transmitting keys directly, blockchain is often used in asymmetric cryptosystems, which are composed of a pair of keys, in which the public key can be obtained as long as it is required. Using the public key to encrypt data is often used to verify digital signatures.

Private Key: A private key corresponds to the public key in a pair of keys. The private key is determined by the public key but cannot be calculated by the public key. The public key can only be decrypted with the corresponding private key, and the public key can verify the content of the private key signature. The information of the private key is confidential.

Node Credit Status List: Each node maintains a list of all nodes' credit history in this article because the node's credit is evaluated based on its historical behaviour. At the end of each epoch, the node's credit is re-evaluated, and the node credit status table is updated.

3.1.2 Assumptions

Assumption 1: The consortium only allows authorized nodes to join, so we think that there are more honest nodes than malicious nodes in the consortium.

Assumption 2: The hash function is one-way calculated, and the hash value can be calculated according to the data, but the data cannot be deduced from the hash value, and the probability of different data outputting the same hash value is close to zero.

Assumption 3: The key system used in this paper is the RSA algorithm, and RSA uses the product of two primes as the public key, so it is complicated to factorize a large number with more than 200bits, so it is assumed that the secret key used in this paper is secure.

3.2 Node Evaluation

In order to ensure the security of the model, it is necessary to evaluate the credit of the nodes participating in the consensus, which is based on the historical behaviour of the nodes. At the same time, according to the performance of the nodes, the reward and punishment mechanism will be set up to change the credit value of the nodes dynamically. Each added node will be assigned an initial credit value, and each node will maintain a credit status table containing all nodes, as shown in Table 1.

The credit evaluation model designed in this paper quantitatively calculates the credit of nodes through many aspects, and the specific calculation methods are as follows:

The first is the rate of participation in consensus: the ratio of the number of times r_i that node i participate in consensus to the total number of times R the system sends consensus over some time: $\tau_i = \frac{r_i}{R} \in [0,1]$, the ratio of node participation in consensus can reflect the enthusiasm of nodes in consensus. Choosing nodes with a high willingness to participate in consensus will help improve the overall consensus's efficiency.

The rate of participating in the consensus and completing the consensus: the ratio of the number of times s_i that the node i successfully reached the consensus to the number of times r_i the node i participated in the consensus over a while: $\varphi_i = \frac{s_i}{r_i} \in [0,1]$. This ratio evaluates the situation in which the node completes the consensus, the node with high completion has high honesty, and the stable network link. Choosing the node with a high completion rate to participate in the consensus can improve the success rate of the consensus.

The rate of malicious behaviour: the ratio of the number s_i of malicious messages sent by a node i over a period of time to the number of times r_i that node i participated in the consensus: $\psi_i = \frac{f_i}{r_i} \in [0,1]$, the credit of the node with malicious behaviour will be reduced because selecting the node with high malicious behaviour ratio will have a negative impact on the security of the system, so such nodes should be removed from the nodes participating in the consensus as soon as possible.

Network environment index: during a time when the node i joins the network, the ratio of network $delay_i$ and the offline time $offline_i$ to online $online_i$: $\sigma_i = -\frac{delay_i + offline_i}{online_i} \in [0,1]$, the network environment of node i will also have a particular impact on the system's consensus, so we also consider the factors of the network environment.

To accurately use various factors to evaluate node credit, this paper allocates the weight of each factor in the credit evaluation of nodes with different credit grades. When the node with a high credit value has negative consensus behaviour, we think it is caused by environmental factors and other objective reasons and will not let its credit value decline sharply. When the nodes with low credit values make positive consensus behaviour, these nodes are more likely to be malicious nodes. Their credit values will not rise rapidly to prevent malicious nodes from being elected as block proposers. The nodes whose credit values are in the middle level are unsure about their nature. In order to speed up the determination of whether they are honest or malicious, we set up that their honest or malicious actions have a more significant impact on the credit value.

Set the weight value: $\vec{w} = [w_1, w_2, w_3, w_4]$, the weight depends on the credit rating of the node. Therefore, the formula for calculating the credit value is

$$P_i = w_1\tau_i + w_2\varphi_i + w_3\psi_i + w_4\sigma_i$$
$$\sigma_i = -\frac{delay_i + offline_i}{online_i} \in [0,1]. \quad (1)$$

In order to encourage the node to make a positive consensus behaviour, the change rate of the credit value of the node is also evaluated, and the formula for calculating the credit change rate of the node is:

$$Q_i = [(P_{i,now} / P_{i,ri})^{1/r_i - 1}. \quad (2)$$

Where $P_{i,now}$ is the current credit value of the node, and $P_{i,ri}$ is the reputation value of the previous r_i consensus of node i.

This paper will comprehensively consider the credit value of the node, the change rate of the credit value and the deposit invested by the node, and quantitatively calculate the comprehensive credit degree of the node, as shown in the following formula:

$$Credit_i = P_i * (1 + Q_i) * \frac{deposit_i}{\sum deposit_i}. \quad (3)$$

Where $deposit_i$ is the deposit paid by the node. The node that delivers a large deposit is less likely to be a malicious node because the return of the node doing evil is lower than the deposit of punishment for doing evil.

The credit will be re-evaluated at the end of each epoch according to the node's performance, as shown in Algorithm 1. The node credit status table will be updated, broadcast to the whole network, and then enter the node classification phase.

Table 1. Credit status table

Attribute	Attribute Tag
NodeID	ID
Public Key	PK
Credit value list	$P_i = [p_1, p_2, ..., p_j]$
Credit value change rate	$Q_i = [q_1, q_2, ..., q_j]$
Comprehensive credit degree	$Credit_i = [c_1, c_2, ..., c_j]$
Total number of system consensus	R
The number of consensus of node participation	r_i
The number of times the node failed to reach a consensus	d_i
The number of times malicious messages were sent	f_i
Network delay	$delay_i$
Node offline time	$offline_i$
Node online time	$online_i$
Deposit paid by node	$deposit_i$

Algorithm 1. Node credit evaluation

Input: Original node status table L_0, node set N

Output: Node credit status table L_N

1 while L_0 do

2 Read (L_0)

3 $\tau_i = \frac{r_i}{R} \in [0,1]$

4 $\varphi_i = \frac{s_i}{r_i} \in [0,1]$

5 $\psi_i = -\frac{f_i}{r_i} \in [0,1]$

6 $\sigma_i = -\frac{delay_i + offline_i}{online_i} \in [0,1]$

7 $\vec{W} = [w_1, w_2, w_3, w_4]$

8 $P_i = w_1\tau_i + w_2\varphi_i + w_3\psi_i + w_4\sigma_i$

9 $Q_i = [(P_{i,now} / P_{i,r_i})^{1/r_i - 1}]$

10 $Credit_i = P_i * (1 + Q_i) * \frac{deposit_i}{\sum deposit_i}$

11 end

12 $L_N = Update(L_0)$

13 return L_N

3.3 Node Classification

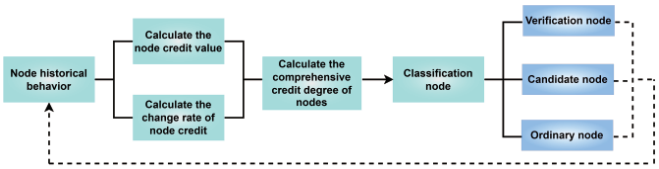

Figure 3. Node classification process

In this part, we will describe the classification process of nodes. As shown in Figure 3, the comprehensive credit degree is obtained by using the credit value and credit value change rate calculated according to the node's historical behaviour. And on this basis, the nodes are divided into three categories: verification nodes, candidate nodes, and ordinary nodes. We will classify nodes according to their credit ranking, as shown in Table 2, and elaborate on the tasks undertaken by different node types.

Table 2. Node classification standard

Comprehensive credit degree	Node types
0%~30%	Verification node
30%~70%	Candidate node
75%~100%	Ordinary node

Verification node: The verification node undertakes the most significant responsibility. To ensure the security of the blockchain system, we must avoid malicious nodes as verification nodes, so we choose the node with the highest credit as the verification node. They are mainly responsible for selecting the block proposers from the candidate nodes and verifying the validity of the blocks generated by the block proposers.

Candidate node: Select the node whose credit degree is in the middle level as the candidate node, which has the opportunity to compete for the right of the packing block but cannot verify the block and select the block proposer.

Ordinary node: This node cannot participate in the competition to become a block proposer nor has the power to verify the block; it is a node downstream of the credit degree. Still, it can improve its credit degree and become a candidate node by adding a deposit.

The roles of these three kinds of nodes are not immutable but will change with the change of their consensus behaviour. If the comprehensive credit degree of the nodes decreases, their roles will also be converted to roles with insufficient privileges. Similarly, if the comprehensive credit degree of the nodes increases, the role will also be upgraded.

To avoid the inaccurate classification of nodes caused by the lack of historical behaviour data in the initial period of consensus. At the beginning of the first round of consensus, we classify nodes according to their deposit ranking. We will reward and punish the deposit of the node according to the performance of the node participation consensus. Nodes can also reduce their deposit according to their wishes, return to roles with low permissions, increase their deposits, and upgrade their permissions. Because the deposit is much higher than the benefits brought by the node's evil, it is safer to do so.

3.4 Vote for Block Proposer

Each verification node votes according to the credit degree of the candidate node, and each verification node must vote for one block proposer, so the number of verification nodes is equal to or redundant with the number of block proposers. We divide each epoch into a fixed number of slots, and the block proposer corresponding to each slot will be randomly selected. The pseudo-code is shown in Algorithm 2, the relevant parameters are shown in Table 3.

All verification nodes encrypt the depth of the next block to be packaged, divide it by $16^{hashlen}$ to get the block proposer' position of their vote, the value must be within the range of the credit degree growth of the block proposer p and will be calculated until the block proposer node with which the credit is satisfied is found, and broadcast with a message $< VoteforBlockProposers, depth, p >_{SK_i}$.

Other verification nodes will verify the validity of the voting message and detect whether the depth is consistent with the depth of the next block to be packaged; decrypt $candidate_i$ using the public key to detect whether the decryption result is equal to depth; recalculation $location_i$ and detect whether p is calculated correctly.

Table 3. DC-DPoS parameter table

Parameter	Explanation
S_v	The sum number of verification nodes
S_c	The sum number of candidate nodes
S_p	The sum number of block proposer
SK_v, PK_v	The private and public key of the verification nodes
SK_P, PK_P	The private and public key of the block proposer nodes
$prehash$	The previous hash of the next block
$depth$	The depth of the next block

Next, to determine the block proposer corresponding to each slot, each block proposer uses the depth of the block to be packed and the previous hash to encrypt to get a result: $proposer_i \leftarrow Encrypt_{SK_P}(depth \| prehash)$.

Use the VRF random function to calculate, sort all the random results, and select the median as the block proposer for this slot, the process of packaging blocks is shown in Figure 4.

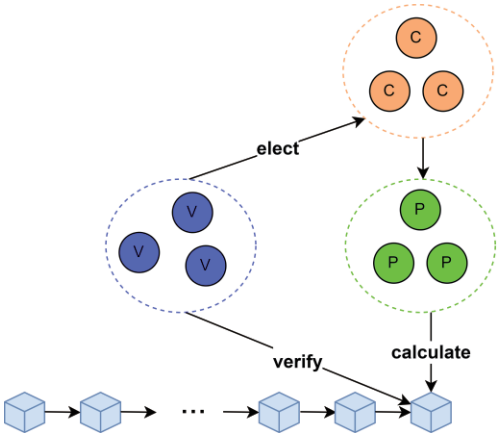

Figure 4. The process of packaging blocks

3.5 Generate New Block

The block proposer will broadcast this message: $< PackingNewBlock, < block >_{SK_P}, Proof_p >_{SK_P}$ when obtaining the corresponding packaged transaction. After receiving the message, other verification nodes will check its validity, such as whether $Proof_p$ includes a certificate corresponding to the selected random number and whether the signature is valid. The block in the message will then be extracted to check whether the timestamp and the block's hash value are correct. If the check passes, the verification node will broadcast a passing message: $< Aggre, < block >_{SK_v}, Proof_p >_{SK_v}$. Only more than half of the verification nodes agree to chain this block; otherwise, a particular block will be chained, and the block proposer's information will be recorded in the block. Use a field to indicate that the block is an error block. If multiple blocks are extracted from the message, it is considered that there is the possibility of multiple consumptions.

In order to avoid double-spending attacks, these blocks will be discarded, and the special blocks mentioned above will be linked.

At the end of each epoch, according to the performance of the participation consensus of the node, its credit degree will be recalculated, and the node will be reclassified. While ensuring security, it also provides a flexible upgrade and downgrade mechanism. The node can come from the consensus of the self-directed participation by increasing or reducing the deposit. As shown in Algorithm 3.

Algorithm 2. Elect for block proposers

1 Input: Node credit status table L_N, $depth$, SK_v, $prehash$, SK_p
2 Output: result //The selected block proposer
3 If an epoch starts
4 reset Credit status table
5 generate verification node
6 generate candidate node
7 generate ordinary node
8 for each verification node v, do
9 $proposer_i \leftarrow Encrypt_{SKp}(depth \| prehash)$
10 $< S_p, Proof_p > \leftarrow VRF_{SK_p}(proposer_i)$
11 $order\_result = rank(S_p)$
12 $result = median(order\_result)$
13 return result
14 end for
15end if

Algorithm 3. Generate a new block

1 If a new block is generated
2 for each verification node v, do
3 if the message is valid
$< PackingNewBlock, < block >_{SK_p}, Proof_p >_{SK_p}$
4 $Broadcast < Aggre, < block >_{SK_v}, Proof_p >_{SK_v}$
5 end if
6 if $valid < Aggre >_{SK_v}$ exceed 50% & only one valid block is extracted.
7 chain the new block
8 else
9 chain the special block
10 end for
11 update the Credit status table
12 end if

4 Safety and Efficiency Analysis

4.1 Safety

Malicious Block Proposer: malicious nodes achieve the packaging process of destroying blocks by pretending to be the block proposer to seek self-interest. The probability that a slot selects malicious nodes:

$$\Pr = \frac{S_p - 1}{S_p}. \tag{4}$$

t slot continuously selects malicious nodes is:

$$\Pr = (\frac{S_p - 1}{S_p})^t. \tag{5}$$

Therefore, when the number of block proposers in the blockchain system increases, the probability of malicious nodes pretending to be the block proposer is significantly reduced.

Double-spending attack: double-spending attack means that by reusing an asset, when the block proposer wants to make a double-spending attack, it will pack more than one block. In the algorithm proposed in this paper, when the verification node checks the message broadcast by the block proposer, it will extract the block. If it finds more than one block, it will discard the block packaged by the block proposer and chain a specially marked block to prevent block proposers from carrying out double-spending attacks. If the verification node and the block proposer collude to attack by sending false voting information, and the number of malicious verification nodes is S_{v_f}, then in the case of

$$\frac{S_v - S_{v\_f}}{S_v} > 0.5. \tag{6}$$

The attack cannot be successful. In the node classification suggested in this paper, the node with a high credit degree is selected as the verification node, so it can be guaranteed that the verification node meets the requirement that the ratio of malicious nodes is less than 0.5.

Self-packing: Self-packing is a process in which block proposers seek self-interest by privately packaging specific blocks or multiple blocks. The DC-DPoS algorithm suggested in this paper uses VRF to do random operations to get the corresponding slot block proposers, Using the previous hash of the packaged block ensures the unpredictability of the result. All the block proposers calculate it, so a single block proposer cannot do evil.

4.2 Efficiency

The consensus algorithm used in this paper does not need mining through massive calculations but selection by voting based on credit. For the functions we use, the same input produces the same output, therefore, the candidate node will only participate once in the competition for each slot block proposer selection. The amount of computation is much less than that of the PoW algorithm. The DC-DPoS algorithm proposed in this paper is mainly divided into three parts. The first part is that the candidates vote for the block proposers, the time complexity is $O(S_v, S_c)$, and the second part is to randomly select each slot block proposers with time complexity of $O(S_v, S_p)$. Finally, the verification node tests the generat-

ed blocks, assuming that a total of S_n blocks are generated, and the time complexity of this step is $O(S_v, S_n)$. The overall complexity of this algorithm is $O(S_v S_c + S_v S_p + S_v S_n)$, which is much less than $O(n^2)$ of the time complexity of the PBFT consensus algorithm.

5 Simulation Results

We will build a model to simulate the algorithm and compare it with the two mainstream algorithms to evaluate the algorithm's performance and prove the theoretical analysis given in the fourth section.

The system test environment is a PC, and the CPU model is AMD A10-8700p Radeon R6 10 Computer Cores 4C+6G 1.80 GHz with the memory of 12GB and the operating system is Ubuntu16.04.

We first set up a DC-DPoS model in the golang language and test the delay of 10000, 15000, 20000 and 25000 transactions when the proportion of verification nodes is 0.1, 0.2, 0.3, 0.4 and 0.5, respectively. We record the probability that malicious nodes are selected as block proposed nodes with increasing consensus time. We also calculate that the credit values of honest and malicious nodes change with the addition of consensus rounds. We test the delay and throughput of DC-DPoS, PBFT and DPoS under 10, 15, 20, 25 and 30 nodes.

5.1 The Influence of Parameters

First, we test the consensus delay of DC-DPoS with different verification node ratios, as shown in Figure 5. The results show that the number of verification nodes affects the consensus delay, but the effect is minimal, and the main impact is the number of transactions.

In order to classify malicious nodes more accurately, we segment the evaluation of nodes. Nodes with low credit will increase the weight of negative behaviour that affects the credit value. As shown in Figure 6, weight1 > weight2 > weight3, we can see that the proportion of malicious nodes selected as verification nodes has decreased after increasing the weight. It can be proved that the way we increase the weight of evil behaviour of nodes with low credit can improve the system's security.

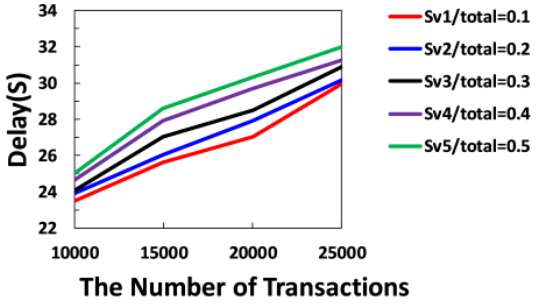

Figure 5. The delay of consensus under the proportion of different verification nodes

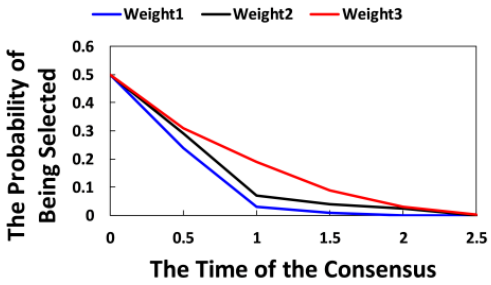

Figure 6. The probability that malicious nodes are selected as block proposed nodes with the increase of consensus time under the different weights

5.2 Safety and Efficiency

First of all, we test the credit value of the nodes after different rounds of consensus to verify the accuracy of our algorithm. From Figure 7, we can see that when the initial credit value is the same, with the increase of consensus rounds, the credit value of malicious nodes decreases, the credit value of honest nodes increases, and the gap increases. It can be proved that our proposed node credit evaluation mechanism is effective.

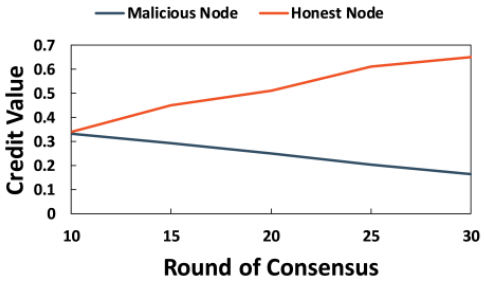

Figure 7. With the increase of consensus rounds, the credit values of malicious nodes and honest nodes change

Delay refers to the time from the initiation of the transaction to the completion of the consensus, which directly affects the transaction speed and is an essential factor in evaluating the performance of the blockchain. The formula is:

$$T_{DELAY} = T_{FINISH} = T_{START}. \quad (7)$$

T_{DELAY} is the delay, T_{FINISH} is the time of completion of the transaction, and T_{START} is the time of initiating the transaction. We compare DC_DPoS with PBFT and DPoS. As shown in Figure 8, we find that the delay of the PBFT algorithm increases obviously with the increase of nodes. In contrast, our algorithm and DPoS are stable, and the delay of DC-DPoS is the lowest, within the acceptable range.

Table 4. Performance comparison with other consensus algorithms

Consensus algorithms	Delay stability evaluation	Throughput stability evaluation	Node reliability evaluation	Reward and punishment	Block proposer election fairness	Suitable for dynamic consensus sets
DC-DPoS	√	√	√	√	√	√
DPoS	√	√	×	√	×	×
PBFT	×	×	×	×	×	×

.

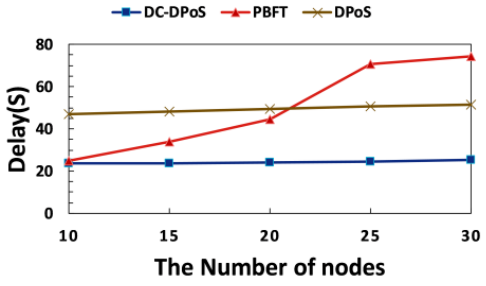

Figure 8. Delay comparison

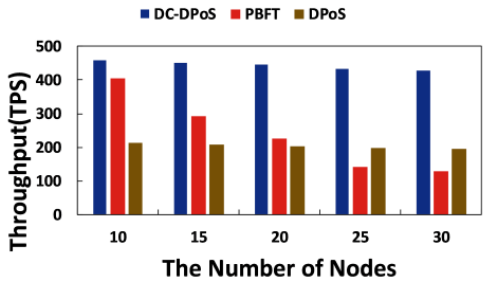

Figure 9. Throughput comparison

Throughput is the total number of transactions processed by the system per second, and it is an important performance of the system's ability to handle transactions. Its formula is:

$$T_{TPS} = \frac{T_{transactions}}{\Delta T}. \tag{8}$$

T_{TPS} is the throughput, $T_{transactions}$ is the total amount of transactions in this period of time, and is the interval of this period of time.

Comparing DC-DPoS with PBFT and DPoS, as shown in Figure 9, the throughput of DC-DPoS is higher than that of the other two standard algorithms, and our algorithm is relatively stable, and the throughput does not decline sharply as the number of nodes increases.

We summarize the characteristics and performance of DC-DPoS, DPoS, and PBFT, as shown in Table 4. The communication times of DC-DPoS and DPoS do not increase with the number of nodes, so the delay and throughput of DC-DPoS and DPoS are stable. DC-DPoS also designs a credit evaluation mechanism, which rewards and punishes according to nodes' behavior, ensuring the security of blockchain consensus. DPoS randomly selects block proposer nodes based on credit, assuring the node election's fairness. In addition, DPoS also provides the function of dynamically joining or exiting nodes so that it can be applied to dynamic networks.

6 Conclusion

Nowadays, there are more and more scenarios for the use of blockchain. In order to meet the needs of practical applications, it is urgent to improve the security and efficiency of a blockchain system. This paper proposes a consensus algorithm DC-DPoS based on dynamic credit. DC-DPoS evaluates the credit of nodes according to the historical behaviour of nodes and classifies nodes on this basis; Experiments show that DC-DPoS can quickly distinguish between malicious and honest nodes to ensure the security of consensus. Through theoretical analysis, we know that DPoS can effectively resist target attacks, double-spending attacks and self-packaging attacks; reward and punish the consensus behaviour of nodes for encouraging nodes to participate in consensus honestly; designs a method to vote for block proposer, which ensures the unpredictability of consensus; and adds a verification link to generate blocks to ensure the effectiveness of blocks. Finally, simulation experiments show that the efficiency of DC-DPoS is significantly improved compared with the traditional consensus algorithm (Time latency is about 50% lower than DPoS, throughput is more than twice that of DPoS, and both are more stable than PBFT), so it can meet the needs of electronic transaction application scenarios.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China under Grant No. 62172061. National Key R&D Program of China under Grant No. 2020YFB1711800 and 2020YFB1707900.

References

[1] S. Nakamoto, Bitcoin: A peer-to-peer electronic cash system, *Decentralized Business Review*, pp. 21260, October, 2008.

[2] Y. Liu, X. T. Liu, L. Zhang, C. J. Tang, H. Y. Kang, An Efficient Strategy to Eliminate Malleability of Bitcoin Transaction, *2017 4th International Conference on Systems and Informatics (Icsai)*, Hangzhou, China, 2017, pp. 960-964.

[3] X. D. Qu, S. L. Wang, Q. Hu, X. Z. Cheng, Proof of Federated Learning: A Novel Energy-Recycling Consensus Algorithm, *IEEE Transactions on Parallel and*

Distributed Systems, Vol. 32, No. 8, pp. 2074-2085, August, 2021.

[4] C. Rodrigues, V. Rocha, Towards Blockchain for Suitable Efficiency and Data Integrity of IoT Ecosystem Transactions, *IEEE Latin America Transactions*, Vol. 19, No. 7, pp. 1199-1206, July, 2021.

[5] Q. Yang, H. Wang, Privacy-Preserving Transactive Energy Management for IoT-Aided Smart Homes via Blockchain, *IEEE Internet of Things Journal*, Vol. 8, No. 14, pp. 11463-11475, July, 2021.

[6] G. Manogaran, M. Alazab, P. M. Shakeel, C. H. Hsu, Blockchain Assisted Secure Data Sharing Model for Internet of Things Based Smart Industries, *IEEE Transactions on Reliability*, Vol. 71, No. 1, pp. 348-358, March, 2022.

[7] X. J. Cai, S. J. Geng, J. B. Zhang, D. Wu, Z. H. Cui, W. S. Zhang, J. J. Chen, A Sharding Scheme-Based Many-Objective Optimization Algorithm for Enhancing Security in Blockchain-Enabled Industrial Internet of Things, *IEEE Transactions on Industrial Informatics*, Vol. 17, No. 11, pp. 7650-7658, November, 2021.

[8] C. Dai, X. G. Liu, L. T. Yang, M. H. Ni, Z. C. Ma, Q. C. Zhang, M. J. Deen, Video Scene Segmentation Using Tensor-Train Faster-RCNN for Multimedia IoT Systems, *IEEE Internet of Things Journal*, Vol. 8, No. 12, pp. 9697-9705, June, 2021.

[9] K. F. Yue, Y. Y. Zhang, Y. R. Chen, Y. Li, L. Zhao, C. M. Rong, L. Y. Chen, A Survey of Decentralizing Applications via Blockchain: The 5G and Beyond Perspective, *IEEE Communications Surveys and Tutorials*, Vol. 23, No. 4, pp. 2191-2217, Fourthquater, 2021.

[10] J. Y. Ren, J. Z. Li, H. X. Liu, T. F. Qin, Task offloading strategy with emergency handling and blockchain security in SDN-empowered and fog-assisted healthcare IoT, *Tsinghua Science and Technology*, Vol. 27, No. 4, pp. 760-776, August, 2022.

[11] C. J. Cai, Y. F. Zheng, Y. F. Du, Z. Qin, C. Wang, Towards Private, Robust, and Verifiable Crowdsensing Systems via Public Blockchains, *IEEE Transactions on Dependable and Secure Computing*, Vol. 18, No. 4, pp. 1893-1907, July-August, 2021.

[12] J. Abdella, Z. Tari, A. Anwar, A. Mahmood, F. L. Han, An Architecture and Performance Evaluation of Blockchain-Based Peer-to-Peer Energy Trading, *IEEE Transactions on Smart Grid*, Vol. 12, No. 4, pp. 3364-3378, July, 2021.

[13] W. Liang, D. F. Zhang, X. Lei, M. D. Tang, K. C. Li, A. Y. Zomaya, Circuit Copyright Blockchain: Blockchain-Based Homomorphic Encryption for IP Circuit Protection, *IEEE Transactions on Emerging Topics in Computing*, Vol. 9, No. 3, pp. 1410-1420, July-September, 2021.

[14] H. T. Wu, Z. Peng, S. T. Guo, Y. Y. Yang, B. Xiao, VQL: Efficient and Verifiable Cloud Query Services for Blockchain Systems, *IEEE Transactions on Parallel and Distributed Systems*, Vol. 33, No. 6, pp. 1393-1406, June, 2022.

[15] C. C. Liu, Y. H. Xiao, V. Javangula, Q. Hu, S. L. Wang, X. Z. Cheng, NormaChain: A Blockchain-Based Normalized Autonomous Transaction Settlement System for IoT-Based E-Commerce, *IEEE Internet of Things Journal*, Vol. 6, No. 3, pp. 4680-4693, June, 2019.

[16] M. Li, L. H. Zhu, Z. J. Zhang, C. Lal, M. Conti, M. Alazab, Anonymous and Verifiable Reputation System for E-Commerce Platforms Based on Blockchain, *IEEE Transactions on Network and Service Management*, Vol. 18, No. 4, pp. 4434-4449, December, 2021.

[17] S. Meiklejohn, C. Orlandi, Privacy-Enhancing Overlays in Bitcoin, *Financial Cryptography and Data Security (Fc 2015)*, San Juan, Puerto Rico, 2015, pp. 127-141.

[18] S. R. Jiang, Y. Z. Li, Q. Y. Lu, Y. M. Hong, D. B. Guan, Y. Xiong, S. Y. Wang, Policy assessments for the carbon emission flows and sustainability of Bitcoin blockchain operation in China, *Nature Communications*, Vol. 12, No. 1, Article No. 1938, April, 2021.

[19] F. M. Bencic, I. P. Zarko, Distributed Ledger Technology: Blockchain Compared to Directed Acyclic Graph, *2018 IEEE 38th International Conference on Distributed Computing Systems (Icdcs)*, Vienna, Austria, 2018, pp. 1569-1570.

[20] M. Padmavathi, R. M. Suresh, Secure P2P Intelligent Network Transaction using Litecoin, *Mobile Networks & Applications*, Vol. 24, No. 2, pp. 318-326, April, 2019.

[21] M. Kara, A. Laouid, M. AlShaikh, M. Hammoudeh, A. Bounceur, R. Euler, A. Amamra, B. Laouid, A Compute and Wait in PoW (CW-PoW) Consensus Algorithm for Preserving Energy Consumption, *Applied Sciences-Basel*, Vol. 11, No. 15, Article No. 6750, August, 2021.

[22] A. Kiayias, A. Russell, B. David, R. Oliynykov, Ouroboros: A Provably Secure Proof-of-Stake Blockchain Protocol, *Advances in Cryptology - Crypto 2017*, Santa Barbara, CA, USA, 2017, pp. 357-388.

[23] B. David, P. Gazi, A. Kiayias, A. Russell, Ouroboros Praos: An Adaptively-Secure, Semi-synchronous Proof-of-Stake Blockchain, *Advances in Cryptology - Eurocrypt 2018*, Tel Aviv, Israel, 2018, pp. 66-98.

[24] P. Daian, R. Pass, E. Shi, Snow White: Robustly Reconfigurable Consensus and Applications to Provably Secure Proof of Stake, *Financial Cryptography and Data Security*, *Fc 2019*, Frigate Bay, St. Kitts and Nevis, 2019, pp. 23-41.

[25] J. Kang, Z. Xiong, D. Niyato, D. Ye, D. I. Kim, J. Zhao, Toward Secure Blockchain-Enabled Internet of Vehicles: Optimizing Consensus Management Using Reputation and Contract Theory, *IEEE Transactions on Vehicular Technology*, Vol. 68, No. 3, pp. 2906-2920, March, 2019.

[26] G. Sun, M. Dai, J. Sun, H. F. Yu, Voting-Based Decentralized Consensus Design for Improving the Efficiency and Security of Consortium Blockchain, *IEEE Internet of Things Journal*, Vol. 8, No. 8, pp. 6257-6272, April, 2021.

[27] L. Lamport, R. Shostak, M. Pease, The Byzantine Generals Problem, *Acm Transactions on Programming Languages and Systems*, Vol. 4, No. 3, pp. 382-401, July, 1982.

[28] M. Castro, B. Liskov, Practical Byzantine fault tolerance, *Usenix Association Proceedings of the Third Symposium on Operating Systems Design and Implementa-*

tion (Osdi '99), New Orleans Louisiana USA, 1999, pp. 173-186.

[29] X. Yuan, F. Luo, M. Z. Haider, Z. K. Chen, Y. C. Li, Efficient Byzantine Consensus Mechanism Based on Reputation in IoT Blockchain, *Wireless Communications & Mobile Computing*, Vol. 2021, Article No. 9952218, May, 2021.

[30] W. Y. Li, C. L. Feng, L. Zhang, H. Xu, B. Cao, M. A. Imran, A Scalable Multi-Layer PBFT Consensus for Blockchain, *IEEE Transactions on Parallel and Distributed Systems*, Vol. 32, No. 5, pp. 1146-1160, May, 2021.

[31] D. Data, L. Q. Song, S. N. Diggavi, Data Encoding for Byzantine-Resilient Distributed Optimization, *IEEE Transactions on Information Theory*, Vol. 67, No. 2, pp. 1117-1140, February, 2021.

[32] J. Misic, V. B. Misic, X. L. Chang, H. Qushtom, Adapting PBFT for Use With Blockchain-Enabled IoT Systems, *IEEE Transactions on Vehicular Technology*, Vol. 70, No. 1, pp. 33-48, January, 2021.

Biographies

Mingjie Zhao is currently pursuing the master degree with the Sichuan University, China. She received the B.E. from Guizhou University, China, in 2020. Her current research interests include big data and blockchain.

Cheng Dai is currently an associate research fellow of Sichuan University, China. He received the Ph.D. from the University of Electronic Science and Technology of China in 2021. His current research interests include human behavior recognition, machine learning, and deep model compression.

Bing Guo is currently a professor of Sichuan University, China, Ph.D. supervisor. He received the Ph.D. from University of Electronic Science and Technology of China in 2002. His current research interests include green computing, personal big data, and blockchain.