


ECBCM: A prestige-based edge computing blockchain security consensus model

Shichang Xuan¹  | Zhiyu Chen¹ | Ilyong Chung² | Haowen Tan² |
Dapeng Man¹ | Xiaojiang Du³ | Wu Yang¹ | Mohsen Guizani⁴

¹Information Security Research Center, Harbin Engineering University, Harbin, China

²Department of Computer Engineering, Chosun University, Gwangju, Korea

³Department of Computer and Information Sciences, Temple University, Philadelphia, Pennsylvania,

⁴Dept. of Computer Science and Engineering, Qatar University, Doha, Qatar

Correspondence

Wu Yang, Information Security Research Center, Harbin Engineering University, Harbin 150001, China.
Email: yangwu@hrbeu.edu.cn

Funding information

China Scholarship Council; Heilongjiang Provincial Science and Technology Department, Grant/Award Number: GX18A008; National Natural Science Foundation of China, Grant/Award Number: 61802086

Abstract

The explosive growth of data in the network has brought huge burdens and challenges to traditional centralized cloud computing data processing. To solve this problem, edge computing technology came into being. Because the edge is closer to the user, processing part of the data at the edge can also bring a faster response to the user and improve their experience. However, the existing edge computing platforms have problems such as data storage security and multiparty data mutual trust. Blockchain technology has become an important means to solve the above data storage and sharing problems due to its excellent characteristics. The core of blockchain technology is consensus, and its speed and security will directly affect the efficiency and stability of the blockchain system. Therefore, this study uses the consensus mechanism as an entry point to reduce the resource consumption of the edge computing blockchain system and improve its security. In order to reduce the resource consumption of traditional consensus algorithms, improve their adaptability in the edge computing environment, and solve the security problem caused by the concentration of node rights, a prestige-based edge computing blockchain security consensus model (ECBCM) is proposed. ECBCM is a general model based on prestige rewards and penalties. It also introduces a node replacement mechanism to ensure the fault tolerance of the consensus process. According to the results of multiple sets of performance comparison experiments and security verification experiments after embedding the existing consensus algorithm, the validity of the consensus model is confirmed.

1 | INTRODUCTION

Edge computing and blockchain technology are in widespread use. Moreover, combining the computing capabilities of edge computing services with the decentralization, data transparency, traceability, and nontamperability of blockchain technology guarantees the security and stability of data in fields that require multiparty collaboration and participation. Indeed, many industries such as Internet of Things (IoTs),^{1,2} smart home,³⁻⁵ vehicular networks,⁶ medicine,⁷⁻¹⁰ finance,¹¹⁻¹³ and manufacturing¹⁴⁻¹⁷ have already combined them based on traditional edge computing model¹⁸ to meet their own needs.

These scenarios generally use slightly modified versions of traditional consensus algorithms that do not account for their high consumption of consensus resources and slow consensus speeds. However, these resources are limited in edge computing environments which means this approach is not practical in scenarios where high consensus speeds are required. To address this, a small number of “good nodes” with a large amount of edge computing resources can be selected as representatives to achieve consensus using the delegated proof-of-stake (DPoS) consensus algorithm.¹⁹ This reduces the consumption of resources and improves efficiency. However, the criteria used to screen good nodes is an important factor that must be carefully considered. If the selection is based solely on the possession of edge computing resources, the power to generate blocks will be concentrated in a small number of nodes. If these nodes become dishonest, then errors will lead to the reselection of good nodes and the frequent generation and repeated execution of blocks will seriously affect the performance of the blockchain consensus system. Furthermore, if a small number of centralized nodes continue to act maliciously, repeated elections of “high-quality nodes” will not help and the system could become paralyzed, which is very dangerous.

In this study, a edge computing-oriented security consensus model (ECBCM) that addresses these issues is proposed. This model introduces prestige evaluation and node replacement mechanisms that effectively prevent the security threats caused by the concentration of node rights. In addition, the ECBCM uses a DPoS consensus algorithm to screen a small number of nodes for consensus, which improves the consensus efficiency and reduces resource consumption. Existing consensus algorithms can be embedded in the ECBCM to form different edge computing-oriented, secure, and efficient consensus algorithms. This provides users with efficient and stable edge blockchain services that are not affected by nodes or security threats caused by the concentration of power. The ECBCM can greatly improve the quality of existing edge blockchain services in various industries and provides a variety of secure and efficient consensus guarantees. To the best of our knowledge, this is the first study to present this kind of solution.

The rest of this paper is organized as follows: Section 2 describes other relevant works and gives a brief introduction to the technologies referenced in this study. Section 3 introduces the application environment for the consensus model and the composition of the network model. It then covers the consensus model design, provides relevant definitions and explanations, and presents the pseudo code for each part of the model. Section 4 presents the experimental part of this study and analyzes the results. Section 5 gives a summary of the work and discusses the prospects for future developments.

2 | RELATED WORK

2.1 | Improvements in edge computing technology

Edge computing technology is an extension of cloud computing technology. It can achieve high-efficiency information acquisition, and use multilevel data processing technology for local preliminary processing of local data to achieve fast information processing to meet user needs. However, data information collection includes multiple forms of information sharing that are associated with data security risks. Therefore, there is significant interest in data protection for edge computing environments. Tian et al proposed a real-time lateral motion detection technique called CloudSEC using an evidence-based reasoning network in the edge-cloud environment. It provides guarantees for attack evidence investigation and real-time attack detection in cloud computing environments.²⁰ Xiao et al proposed a secure collaborative fetch scheme based on reinforcement learning and lightweight authentication for a variety of typical attack models that are vulnerable to mobile edge-computing cache systems; this provides data privacy protection.²¹ Shen et al proposed an efficient graph encryption method based on a tree-based key comparison protocol. This method uses symmetric key primitives and homomorphic encryption to improve the calculation speed during the encryption process. It can be used to encrypt clients and the cloud, and the information between graphs improves the security of user privacy.²² Du et al proposed a new solution to the problem that edge computing networks are vulnerable to severe denial of service attacks due to the low-computing-power edge devices are not suitable to use digital signatures for broadcast verification.²³ Wang et al combined data mining, deep learning, and reinforcement learning technologies and proposed a new technology framework that combines 5G with edge computing IoT.²⁴

2.2 | Consensus algorithm

Since Satoshi Nakamoto published a white paper on Bitcoin there has been extensive research into the underlying blockchain technology. A blockchain is a public, unchangeable, distributed ledger which is guaranteed by a

large number of cryptographic techniques. Data security is provided using consensus algorithms to ensure that the accounting is consistent.²⁵ This consensus mechanism is the core of blockchain technology. The proof-of-work (PoW) algorithm gained attention with the rise of Bitcoin and since then it has been widely studied and improved.²⁶ For example, the proof-of-stake (PoS) consensus algorithm reduces the huge amount of computing resources consumed by the PoW algorithm by introducing the concept of “coin age.” This means that a node's right to vote not only depends on the amount of currency it holds, but also the amount of time it has held it. As each node vote is completed, the time for which it owned the currency will be cleared.²⁷ The DPoS algorithm is an optimized version of the PoS consensus algorithm. By selecting a small number of nodes for consensus, it reduces the consumption of computing and network resources.¹⁹ The Paxos consensus algorithm is a leaderless consensus algorithm,²⁸ but due to the complexity of its consensus process, the Raft consensus algorithm, which favors leadership elections, has emerged as an alternative.²⁹ The practical Byzantine fault tolerance (PBFT) algorithm was originally used to solve the message consistency problem, but it was introduced to the blockchain due to its excellent fault tolerance and efficiency.³⁰

2.3 | Improvements of the consensus algorithm

The speed of the consensus algorithm will seriously affect the working efficiency of blockchain systems, so the improvement of consensus-based algorithms is an important research topic. Wang et al proposed a lightweight consensus algorithm called SM based on the PBFT consensus algorithm. It greatly reduces the complexity of network communications during the consensus process.³¹ EOS is a new commercial distributed blockchain operating system that is committed to improving and innovating the DPoS consensus algorithm.³² Hu et al proposed a packet-based DPoS consensus algorithm to serve educational administration systems.³³ Gueta et al developed a new simplified Byzantine fault tolerance (SBFT) fast consensus algorithm based on the PBFT consensus algorithm. This algorithm provides greater consensus efficiency and error node tolerance.³⁴ Liu et al presented a new lightweight consensus algorithm called SMP, which serves the industrial IoT with weak computing power.³⁵ Han et al proposed a credit-based consensus algorithm called proof-of-credit (PoC), which quantifies the past performance of nodes and uses it as an influence factor in the consensus process. This can directly affect the difficulty factor in the workload proof problem that nodes solve for competing block production rights in the PoW consensus algorithm. Therefore, nodes with a history of dishonesty are at a disadvantage position in the competition. The result of the competition affects the opportunity for nodes to generate blocks.³⁶ Lei et al proposed a reputation-based Byzantine fault tolerance consensus algorithm that gives an honor rating to each participating node. This rating is affected by the consensus behavior of the node until it becomes a master node in the PBFT algorithm. When a node interacts with another node that has a low reputation rating, it will reduce its own reputation. This encourages nodes to interact with nodes that have normal ratings and those with poor ratings become isolated. Therefore, the probability of such nodes acting maliciously is reduced.³⁷

2.4 | Consensus algorithm in edge computing blockchain

In the edge computing model combined with blockchain technology, the following researches have been done on the improvement of consensus algorithms. Xu et al developed a blockchain-based big data edge sharing platform to ensure data sharing security of different stakeholder edge devices, which uses a collaborative proof consensus algorithm PoC, which has low computational complexity degree, which is more suitable for edge computing devices with weak computing capabilities.³⁸ Deepak et al proposed a new consensus algorithm called PoAh to reduce the computational power consumption of PoW consensus algorithms on edge computing devices, and introduced identity in edge computing environments and IoT applications. The verification mechanism makes the blockchain application specialized, ensuring the security of the blockchain client.³⁹ Zhao et al targeted the edge computing environment. Data collected by edge devices may be fraudulent, and security issues such as unauthorized transmission of data by edge devices may also occur. This paper proposes a blockchain-based edge trusted data management scheme, Block-TDM, which uses the elastic consensus based on the PBFT consensus algorithm to ensure the security, availability and efficiency of the blockchain system.⁴⁰

3 | EDGE COMPUTING-ORIENTED SECURITY CONSENSUS MODEL

3.1 | Application environment and network model

The ECBCM was applied to a typical edge computing service model;¹⁸ this only required a small amount of additional content. As shown in Figure 1, there are two types of users in the edge computing service model. The first type of users is willing to participate in the maintenance of the blockchain and contribute the computing resources they own to maintain the operation of the blockchain. They act as edge nodes. The second type of user has no intention to participate in the maintenance of the blockchain, but simply acts as a user of edge computing and blockchain services, and provides data for the blockchain system. The first type of users are edge nodes and are divided into different types of nodes according to the filtering rules. Some nodes are responsible for collecting transactions and propagating them within a specific virtual machine network. Some nodes are responsible for merging transactions to form new blocks. Some nodes are responsible for verifying the correctness of the block to ensure that the transaction is true and correct. Finally, an auditor is added to the edge computing service model. The auditor collects confirmation information for each transaction block that is successfully added to the blockchain and records it in a database for users to query.

Therefore, in the ECBCM, the set of nodes participating in the consensus process is defined as N , and its size is k , ($k \in N^*$). The set N is divided into two subsets: the consensus node set N^C , with size l , ($l \in N^*$, $l < k$), and the transaction node set N^T , with size $(k - l)$. For N^C and N^T , $N^C \cup N^T = N$ and $N^C \cap N^T = \varphi$. A subnetwork composed of the consensus nodes N_i^C is called the consensus network and a subnetwork composed of transaction nodes N_i^T is called the transaction network. Since each consensus changes the nodes in the consensus and the transaction networks, they are not static. The consensus network is determined by the current node identity information and it remains unchanged until the next round of node elections causes the node identities to change.

The consensus node set is further divided into a witness node set N^W , with size m , ($m \in N^*$, $m < l$), and a participant node set N^P , with size $(l - m)$. In the ECBCM blockchain environment, the transaction nodes generate, encrypt, and sign transactions; broadcast the transactions back to the blockchain network; and vote for consensus nodes. The witness nodes are the winners based on the votes. These are the “representative” nodes that are responsible for producing blocks. Participant nodes are responsible for verifying the blocks generated by the witness nodes and all the nodes maintain a complete copy of the blockchain. The classification of nodes is shown in Figure 2.

The resources owned by every user who acts as an edge node will be mortgaged to maintain normal operation of the blockchain system. This part of the user's resources will be used in the traditional DPoS consensus algorithm to determine the amount of equity that decides whether a node becomes a block producer and will not be returned to users at will. A vector $R_i \leq C_i^{\max}, S_i^{\max}, E_i^{\max} >$, ($i \in [1, k]$) is used to represent the total resources owned by a user, where C_i^{\max} is the total number of cores in the central processing unit (CPU), S_i^{\max} is the memory size in kB, and D_i^{\max} is the network bandwidth in kB/s. To protect the normal business operations of users, a certain amount of resources $R_i^U = < C_i^U, S_i^U, E_i^U >$ will be reserved for user N_i according to the amount of resources that have been used.

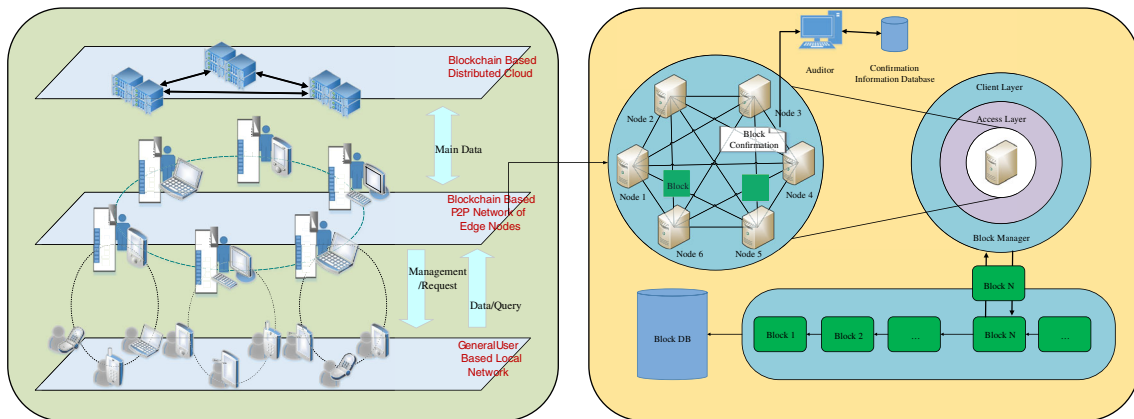


FIGURE 1 Edge computing service model

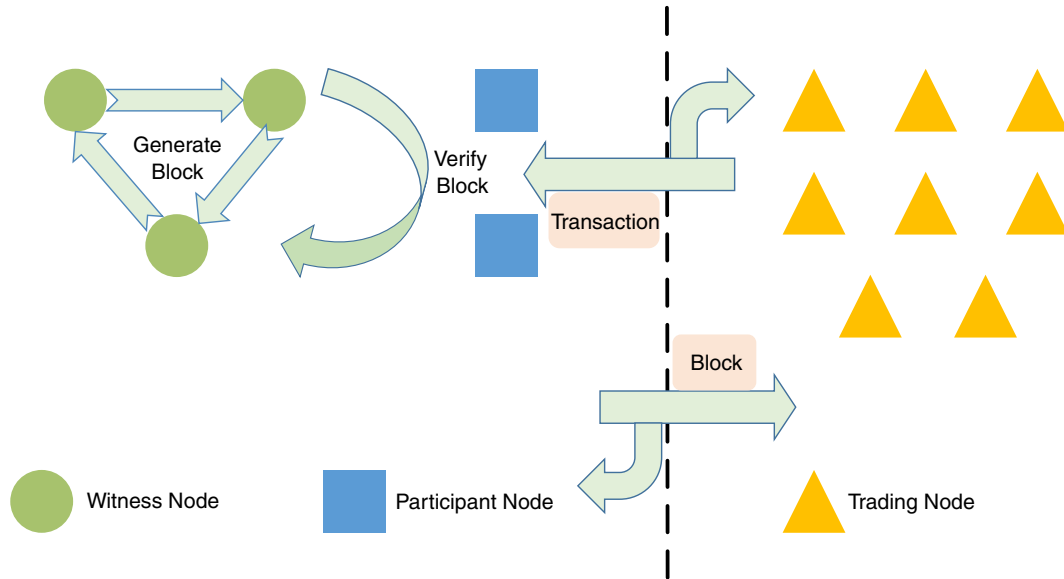


FIGURE 2 Node classification diagram

Each user N_i has a greedy factor $\sigma_i \in (0, 1]$ and the size of σ is set by the user to determine how many idle resources are put into the blockchain consensus process as equity. The formula for the greedy factor σ_i is

$$\sigma_i = w_1 \sigma_{cpu}^i + w_2 \sigma_{mem}^i + w_3 \sigma_{nw}^i, \quad (1)$$

where σ_{cpu}^i , σ_{mem}^i , and σ_{nw}^i represent the CPU, memory, and network bandwidth components corresponding to the greedy factor, respectively; w_1 , w_2 , and w_3 are scaling parameters; and $\sum_{k \in \{1, 2, 3\}} w_k = 1$. The greedy factor ensures that users will not mortgage the same resources as rights and interests, bringing heterogeneity to the consensus voting. Therefore, the equity function is defined as

$$f(R, R^U, \sigma) = \sigma(R - R^U). \quad (2)$$

The equity of user N_i , ($i \in [1, k]$) is defined as $\alpha_i = \alpha_{C_i}, \alpha_{S_i}, \alpha_{E_i}$, where

$$\alpha_{C_i} = \sigma_{cpu}^i (C_i^{\max} - C_i^U). \quad (3)$$

$$\alpha_{S_i} = \sigma_{mem}^i (S_i^{\max} - S_i^U). \quad (4)$$

$$\alpha_{E_i} = \sigma_{nw}^i (E_i^{\max} - E_i^U). \quad (5)$$

Here, α_{C_i} in (3) represents the CPU component corresponding to the equity amount, α_{S_i} in (4) represents the memory component corresponding to the equity amount, and α_{E_i} in (5) represents the network bandwidth component corresponding to the equity.

3.2 | Main idea of the ECBCM

In the ECBCM, the prestige value is defined as the behavior index of the nodes participating in the overall consensus process. The prestige value can directly affect the voting results of good nodes, which decides whether a node is qualified to generate blocks. This value is determined based on the number of resources that the node mortgages to participate in the consensus, the duration of the mortgage, and the behavior of the node during the consensus process. The ECBCM

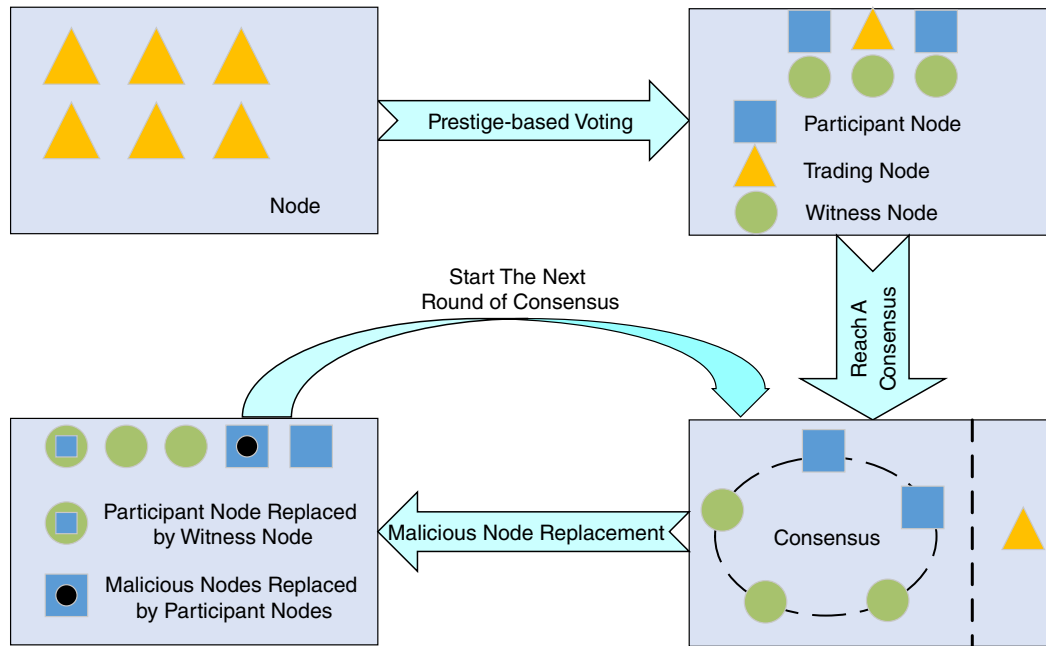


FIGURE 3 Diagram showing the edge computing blockchain security consensus model structure

consensus model is divided into three parts: election based on **prestige**, block consensus based on **embedded algorithms**, and **malicious node replacement**. The overall structure is shown in Figure 3.

The ECBCM includes voting based on prestige and the generation and replacement of malicious nodes. The process can be summarized as follows:

1. In the voting process, the amount of equity staked by users is not the only consideration. Instead, the duration for which users mortgage resources and whether the nodes are normal and actively participating in the blockchain consensus process is considered. The behavioral performance is an influencing factor and the prestige value formed by a combination of these factors is used as the consideration criteria for voting. Therefore, the elected node must be the best node for comprehensive evaluation.
2. Once voting is complete, the behavior of the node is evaluated. Both negative and wrong voting behavior will affect the prestige of the node, which will affect the voting results during the next election.
3. In the consensus process following the last round of elections, nodes with a tendency to commit evil will be punished and deprived of their right to vote. They can continue to participate in the consensus process once their prestige has returned to a certain standard. Nodes will be permanently deprived of their right to participate in elections and negotiations with the edge computing service provider must occur before this right can be restored.

3.3 | Definitions and explanations

In the ECBCM, the prestige value is used as a comprehensive evaluation standard for the nodes participating in the consensus process. During each round of voting, the performance of each node after the previous round will determine the effect on this round of voting; that is, the prestige value of the result. In this model, the prestige value is defined as Rep , where

$$Rep_i = \beta_1 Rep_{stake}^i + \beta_2 Rep_{vote}^i + \beta_3 Rep_{consensus}^i \quad (6)$$

Here, Rep_{stake}^i , Rep_{vote}^i , $Rep_{consensus}^i$ represent the equity component of the user's mortgage, the voting performance, and consensus process performance in prestige, respectively. The terms β_1 , β_2 , and β_3 are the prestige weights and $\sigma_{k \in 1,2,3} \beta_k = 1$. The prestige weight determines the effect that each component has on the prestige value and the specific

size setting should be based on demand. According its working nature, each node N_i may have either the voting performance component $\text{Rep}_{\text{vote}}^i$ or the consensus process performance component $\text{Rep}_{\text{consensus}}^i$ in the prestige calculation. If a node voted and was elected as a witness node in the previous round, and if this identity was maintained in the subsequent consensus process, then during the current round of voting its prestige will lose the $\text{Rep}_{\text{vote}}^i$ and $\text{Rep}_{\text{consensus}}^i$ bonuses.

The equity component $\text{Rep}_{\text{stake}}^i$ is calculated as

$$\text{Rep}_{\text{stake}}^i = \theta_i * |\alpha_i| * \delta_T^i, \quad (7)$$

where θ_i is the prestige coefficient of node N_i and $0 \leq \theta_i \leq 1$. In addition, $|\alpha_i|$ represents the value of user N_i in terms of the amount of equity participating in the consensus mortgage. Finally, δ_T^i represents the duration for which user N_i mortgages the edge computing resources with a value of 1 to 10 minutes. Users can change this value in advance to determine the next mortgage time. Regardless of the type of node a user is determined to be, they are not allowed to exit the consensus process before the next mortgage time. Users who are willing to mortgage resources for a longer time are considered to be more credible in this consensus model. For each node, the value of its equity component $\text{Rep}_{\text{stake}}^i$ will be determined before each round of voting begins; this information is provided by the edge computing service provider and the blockchain system.

In the ECBCM, nodes are categorized according to their prestige coefficient θ_i :

LY(loyal): $\theta \in (0.5, 1]$. Nodes in this state are normal and participate in the election normally.

NL(disloyal): $\theta \in (0, 0.5]$. Nodes in this state are untrustworthy and temporarily lose their right to become witness nodes. Their prestige coefficient will not change during the consensus process after this round of voting, but it can be changed if they actively participate in the consensus process after the next round of voting. MA(malicious): $\theta = 0$. Nodes in this state are malicious and will permanently lose their right to become witness nodes. Their prestige coefficient will not be restored. To cancel this state, they must contact the administrator.

During the voting process, each transaction node N_i^T has three votes in favor and one vote against/abstention. When a node votes for the elected consensus node, or when it votes against a node with a rating of NL or MA, its correct number of votes is increased by one. If a node with a rating of NL or MA is not found, the correct number of votes will be increased by one if it abstains from voting. Other situations, such as incorrect or negative voting, will increase the number of incorrect votes by one. During the voting process, if the node receives a positive vote then the total number of votes it receives is increased by one; if it receives a negative vote, it is decreased by one. Abstentions are not counted in the total votes.

The voting process performance component $\text{Rep}_{\text{vote}}^i$ is calculated using the equations

$$\text{Rep}_{\text{vote}}^i = (\rho_1 P_i^{V_C} - \rho_2 P_i^{V_E}) * B_V. \quad (8)$$

$$P_i^{V_C} = V_{\text{correct}}^i / V_{\text{Total}}^i. \quad (9)$$

$$P_i^{V_E} = V_{\text{error}}^i / V_{\text{Total}}^i. \quad (10)$$

Here $P_i^{V_C}$ and $P_i^{V_E}$ are the probabilities that node N_i voted correctly or incorrectly in the last round of voting, respectively. In addition, ρ_1 and ρ_2 are the voting weights. In general, ρ_2 will be greater than ρ_1 , which means that dishonest nodes are penalized by decreasing their prestige. The factor B_V is the voting weight, which is introduced so that process the performance and equity components $\text{Rep}_{\text{vote}}^i$ and $\text{Rep}_{\text{stake}}^i$ have an equivalent effect on the prestige value Rep_i . The terms V_{correct}^i , V_{error}^i , and V_{Total}^i represent the number of correct, number of incorrect, and total number of votes during the last round of voting, respectively.

The consensus process performance component $\text{Rep}_{\text{consensus}}^i$ its calculated using the equations

$$\text{Rep}_{\text{consensus}}^i = (\varphi_1 P_i^{P_C} - \varphi_2 P_i^{P_E}) * B_C. \quad (11)$$

$$P_i^{P_C} = C_{\text{correct}}^i / C_{\text{Total}}^i. \quad (12)$$

$$P_i^{P_E} = C_{\text{error}}^i / C_{\text{Total}}^i. \quad (13)$$

Here $P_i^{P_C}$ is the probability that a block confirmed by the participant node has successfully joined the blockchain or that an objectionable block has not been added to the blockchain; $P_i^{P_E}$ is the probability that a successful block was opposed or that an objectionable block was confirmed. In addition, φ_1 and φ_2 are the block confirmation weights. In general, φ_2 will be greater than φ_1 , which ensures that nodes which abnormally confirm a block are punished by decreasing their prestige. The factor B_C is the consensus weight, this ensures that the consensus process performance and equity components $\text{Rep}_{\text{consensus}}^i$ and $\text{Rep}_{\text{stake}}^i$ have an equivalent effect on the prestige value Rep_i . The terms C_{correct}^i , C_{error}^i , and C_{Total}^i represent the number of correctly confirmed, number of incorrectly confirmed, and total number of blocks output after the last round of voting, respectively.

The performance components of the voting and consensus processes were added because the node's performance during the consensus process after the previous round of voting will affect its prestige coefficient during the current round. The prestige coefficient cannot be based solely on the performance of the node in the consensus process after the previous round of voting and prestige reward and punishment are directly used in the current round of the voting process. This is achieved using the performance during the voting and consensus processes.

During the voting process, the transaction node N_i^T will receive successive voting requests from many nodes. Once a request is accepted, node N_i will determine whether to vote according to the following inequality:

$$\langle \text{HASH}(\text{HASH}(\text{PreBlockHead}), \text{Round}) \rangle_{\text{LAST16}} \leq \omega_T * \text{Rep}_C^i * M. \quad (14)$$

The first half of the inequality connects the hash value of the latest block in the blockchain with the current voting round value Round, calculates the hash value of the connected value, and obtains the last 16 bits of the hash value. The term ω_T represents the number of seconds that have elapsed since the first voting request was received from the transaction node N_i^T and Rep_C^i is the prestige value of the consensus node N_i^C that sent the request. Finally, M is the difficulty value and its magnitude affects the difficulty of meeting the inequality. It can be seen that the inequality is easier to satisfy when the node's prestige value is higher, and it becomes easier to satisfy over time.

For node N_i , after each round of voting as each block is generated, the size of its prestige coefficient θ will be changed as a reward or punishment. This change will be determined by the formula:

$$\theta_{\text{Round}}^i = \begin{cases} \theta_{\text{Round}-1}^i & \text{if Prewitness behaves loyal} \\ \min((\theta_{\text{Round}-1}^i + P_i^{V_C} * Z), 1) & \text{if } P_i^{V_C} \geq P_i^{V_E} \\ \max((\theta_{\text{Round}-1}^i - P_i^{V_E} * Z), \theta) & \text{if } P_i^{V_C} < P_i^{V_E} \\ \min((\theta_{\text{Round}-1}^i + P_i^{P_C} * Z), 1) & \text{if } P_i^{P_C} \geq P_i^{P_E} \\ \max((\theta_{\text{Round}-1}^i - P_i^{P_E} * Z), \theta) & \text{if } P_i^{P_C} < P_i^{P_E}. \end{cases} \quad (15)$$

Here, Z is the recovery base and its value determines the recovery speed of the prestige coefficient of node N_i . This shows that if node N_i was elected as a witness node in the consensus process after the last round of voting, and if no malicious operations were performed (ie, the block was not produced or the wrong block was produced), then its prestige coefficient will remain unchanged during the consensus process after the current round of voting. The ability of participant and trading nodes to work normally is closely related to the size of their prestige coefficient. Normal work will slowly restore the node's prestige coefficient, but negative or malicious work will reduce it. This is directly related to whether a node can participate in the election process normally. If node N_i^W is elected as the witness node after the current round of voting performs a malicious operation (ie, a block is not produced or the incorrect block is produced) during the consensus process, then the change in its prestige coefficient will be determined by

$$\theta_{\text{Round}}^i = \max((\theta_{\text{Round}}^i - 0.5), 0). \quad (16)$$

The malicious behavior will stop all the prestige rewards/penalties after the current round of voting. This prevents the malicious node from being selected for the consensus node set in the next round of voting.

In the ECBCM, the set of real-time prestige information will be stored as the content of the block, defined here as RXs. This set contains the prestige information RX_{N_i} for each node including the node number N_i , node identity ($N_i^W / N_i^P / N_i^T$),

node prestige coefficient θ_i , malicious flag *Malicious*, the number of correct votes V_{correct}^i , the number of errors or negative votes V_{error}^i , the total number of votes V_{Total}^i , the number of nodes that correctly confirmed the block after this round of voting C_{correct}^i , the number of errors or negative confirmation blocks C_{error}^i , and the total number of blocks C_{Total}^i generated after the current round of voting. The information in the prestige information set forms a prestige tree. The root $\text{MerkleRoot}_{\text{rep}}$ of the prestige tree is stored in the block header for subsequent comparison and confirmation.

3.4 | Design of the ECBCM

3.4.1 | Consensus node election

In the ECBCM, before the consensus operation is performed, polling and election based on prestige is periodically performed to complete the selection of some high-quality nodes. The details of this process are shown in Algorithm 1. This will be executed after the initialization of the blockchain system. During the subsequent operation of the blockchain system, the algorithm will also be executed regularly. The algorithm uses the round information *Round* of the current election and the node set *N* as inputs, then returns the witness node set N^W , participant node set N^P , old node prestige information list List_{old} , new node prestige information list List_{new} , and the list of local node prestige information stored by each node $\text{List}_{\text{node}}$. A brief description of the process in this part is shown in Figure 4. The detailed process is in Algorithm 1.

3.4.2 | Consensus process

When the election is completed, the node sets N^W , N^P , and N^T will be generated and the ECBCM will move to the block consensus section, which will use different consensus algorithms based on the consensus algorithms embedded in the model. This involves the processes of block generation and consensus. The overall framework of this process and a description of this section when the PoW and PBFT consensus algorithms are embedded in the ECBCM are given below.

Main Framework of the Block Consensus Process

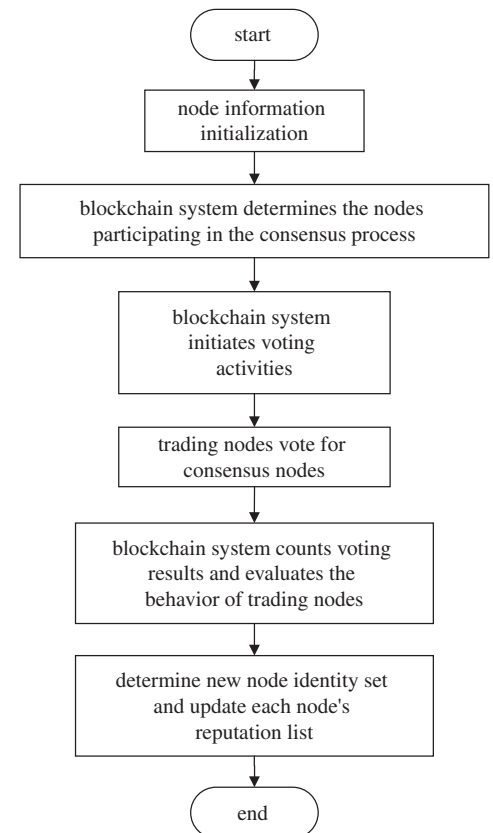


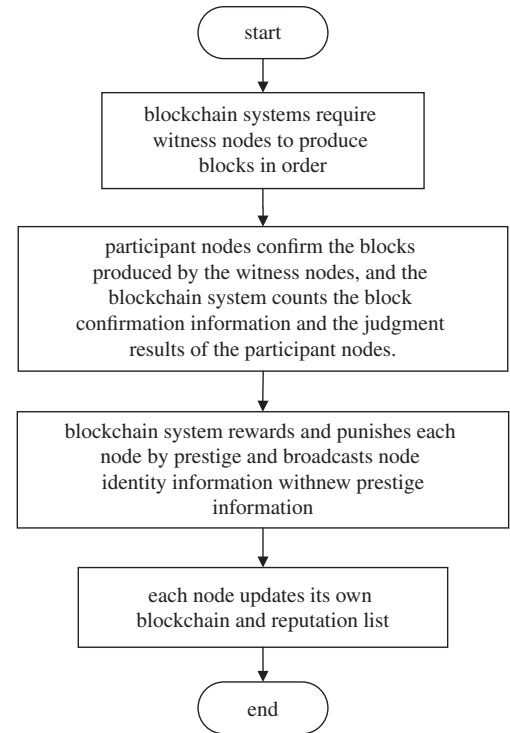
FIGURE 4 Comparison of the consensus algorithms

Algorithm 1. Voting algorithm based on prestige**Input:** Round, N **Output:** N^W , N^P , List_{old}, List_{new}, List_{node}

```

1: start
2: system get RXsnew and create Listold, Listnew
3: system sort stakeholders && Broadcast <HASH(PreBlockHead) MerkleRootRepNew, RXsnew, Round,  $N_i$ > to top  $l$ 
4:  $N_i$  Broadcast  $HASH(HASH(PreBlockHead), MerkleRoot_{Rep}^{New}, Round)N_i$ 
5:  $N_i$  create Listnode $i$  with RXsnew
6: while Count( $HASH(HASH(PreBlockHead), MerkleRoot_{Rep}^{New}, Round, N_i)$ ) <  $l$  do
7:   System Broadcast<HASH(PreBlockHead), MerkleRootRepNew, RXsnew, Round,  $N_i$ > to others in order
8: end while
9:  $N_i^C <- N_i$ 
10: system Broadcast >  $N^C$  <
11:  $N^T <- N - N^C$ 
12: system Broadcast > Vote<
13:  $N_i^C$  Broadcast >  $N_i^C$ , RX $i$  $C$ , Round>
14: some description
15: if Round is right &&  $N_i^T$  still have a vote then
16:   if  $\theta_i^C \leq 0.5$  then
17:     if  $N_i^T$  still have a oppose/abstain vote then
18:        $N_i^T$  vote for  $N_i^C$  with a oppose vote
19:     else  $N_i^T$  ignore message
20:   end if
21: else
22:   if  $N_i^T$  still have a approve vote then
23:     if Vote(RX $i$  $C$ ) then
24:        $N_i^T$  vote for  $N_i^C$  with a approve vote
25:     else
26:        $N_i^T$  ignore message
27:     end if
28:   else
29:     if  $N_i^T$  still have a oppose/abstain vote then
30:        $N_i^T$  vote for  $N_i^C$  with a abstain vote
31:     else
32:        $N_i^T$  ignore message
33:     end if
34:   end if
35: end if
36: else
37:    $N_i^T$  ignore message
38: end if
39:  $V(N_i^C) = \text{Count}(N_i^C)$ 
40: Quicksort( $V(N_i^C)$ )
41:  $N_i^W <- N_i^C (i \in [1, m])$ 
42:  $N_i^P <- N_i^C (i \in [1, l - m])$ 
43: system update Listnew
44: Randsort( $N^W$ )
45: system Broadcast >  $N^W$ ,  $N^P$  <
46: return result

```

FIGURE 5 Comparison of the consensus algorithms

The main framework of the block consensus process is shown in Algorithm 2. This will be executed once the blockchain system has been initialized. The blockchain system will start different block generation and consensus processes according to the consensus algorithms embedded in the consensus model to complete the expansion of the blockchain. The algorithm takes the witness and participant node sets N^W and N^P as inputs and returns the new block judgment information $\text{Block}_{\text{CORRECT}}$ or $\text{Block}_{\text{ERROR}}$, the old node prestige information list List_{old} , the new node prestige information list List_{new} and the list of local node prestige information stored by each node $\text{List}_{\text{node}}$. A brief description of the process in this part is shown in Figure 5. The detailed process is in Algorithm 2.

Algorithm 2. Consensus algorithm based on embedded algorithms

Input: N^W, N^P

Output: $\text{Block}_{\text{CORRECT}}, \text{Block}_{\text{ERROR}}, \text{List}_{\text{old}}, \text{List}_{\text{new}}, \text{List}_{\text{node}}$

```

1: start
2: system Broadcast > message < for produce a Block
3:  $N_i^W$ : produce a Block && Broadcast >  $\text{Block}_{N_i^W}$  <
4:  $N_i^P$ : Check(< $\text{Block}_{N_i^W}$ >)
5: system record  $N_i^P$ 's confirm of COMMIT
6: if Count(COMMIT) satisfy condition then
7:   system Broadcast < $\text{Block}_{\text{CORRECT}}$ >
8:   system judge every  $N_i^P$ 's COMMIT && update  $\text{List}_{\text{new}}$ 
9:   Reward_Punish( $N_i$ )
10:  system update  $\text{List}_{\text{old}}$  &&  $\text{List}_{\text{new}}$ 
11:  system Broadcast > RXs <
12:  every node update  $\text{List}_{\text{node}}$ 
13: else
14:   system Broadcast >  $\text{Block}_{\text{ERROR}}$  <
15:   system change to Algorithm 3
16: end if
17: return result
  
```

Block Consensus Process Embedded in the PoW Consensus Algorithm. The block consensus process embedded in the PoW consensus algorithm can be divided into the following steps:

The system sends a production block message to the witness node containing information such as the round of the election Round, the random number nonce, and the difficulty value D.

Each node performs block generation and broadcast according to the PoW consensus algorithm based on the received information. Each node checks the correctness of the newly generated block and determines the broadcast confirmation message or ignores the new block based on the result.

The system takes $\lfloor (l - m)/2 \rfloor + 1$ as the statistical standard for the COMMIT message in Algorithm 2, and then adopts the general process to complete the consensus process.

Block Consensus Process Embedded in the PBFT Consensus Algorithm The block consensus process embedded in the PBFT consensus algorithm can be divided into the following steps:

The system sends a confirmation message to the witness node that generated the block in this round. If no response is received, it transfers directly to Algorithm 3 for malicious node replacement processing.

Witness nodes that respond generate a block, which is used as the content of the consensus transmitted in the PBFT consensus algorithm. According to the process of the PBFT consensus algorithm, the new block is submitted to the participant nodes for consensus and final confirmation is given. If a new block is found to have problems during the consensus process, the system will replace the witness node that generated the block according to Algorithm 3.

The system takes $2 * \lfloor (l - m - 1)/3 \rfloor + 1$ as the statistical standard of the COMMIT message in Algorithm 2 and adopts the general process to complete the block consensus process.

3.4.3 | Malicious node replacement

When the block generated by the witness node is Block_{ERROR} or no block is generated, the ECBCM will transfer to the malicious node replacement section, which will complete the witness and participant node sets N^W and N^P as well as the conversion between nodes. The details of this process are shown in Algorithm 3. Algorithm 3 is executed when the witness node fails to respond or the witness node does not generate the correct block according to the regulations. After execution, the algorithm returns the updated witness and participant node sets N^W and N^P . A brief description of the process in this part is shown in Figure 6. The detailed process is in Algorithm 3.

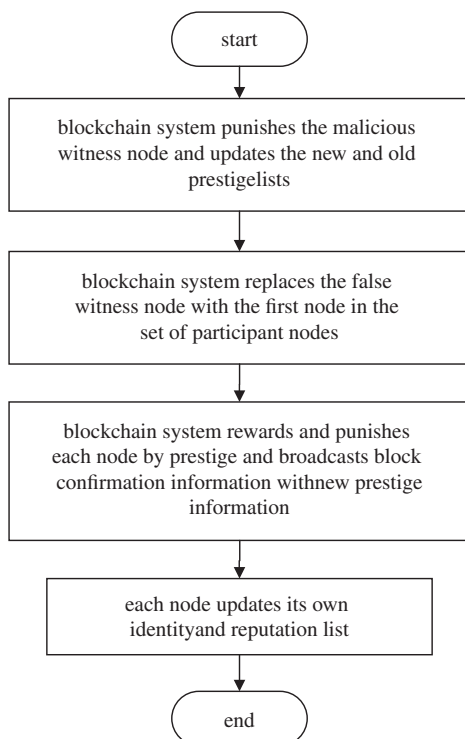


FIGURE 6 Comparison of the consensus algorithms

Algorithm 3. Malicious node replacement algorithm**Input:** $Block_{ERROR}$ **Output:** N^W, N^P

```

start
2:  $Malicious \leftarrow N_i^W$ 
    $\theta_{N_i^W} = Punish(N_i^W)$ 
4: system update  $List_{old}$  &&  $List_{new}$ 
   for  $i \leftarrow (x+1)$  to  $m$  do
6:    $N_{i-1}^W$ 
    $N_m^W \leftarrow N_1^P$ 
8:   for  $i \leftarrow 2$  to  $(l-m)$  do
    $N_{i-1}^P$ 
10:   $N_{l-m}^P \leftarrow N_{malicious}^W$ 
   get  $N^W$  &&  $N^P$ 
12:  Reward_Punish( $N_i$ )
   system update  $List_{old}$  &&  $List_{new}$ 
14:  system Broadcast  $> N^W, N^P, RXs <$ 
   every node update  $List_{node}$ 
16: end for
   end for
18: return result

```

4 | EXPERIMENTAL TESTING OF THE ECBCM

The purpose of these experiments was to test the consensus efficiency of the consensus model, the ability of the prestige evaluation mechanism to punish malicious nodes and decentralize the power of the nodes, and the ability of the consensus model to detect and replace malicious nodes. Some parameters were fixed throughout the experiments, as shown in Table 1.

The simulation program used in these experiments used multithread technology. The distributed multinode environment of the blockchain was simulated by assigning multiple threads with each thread representing a node. When the program began, a corresponding number of nodes were enabled according to the user's initial settings and the blockchain information of each node was initialized. During the operation of the program, the data required for the experiment was output to a log file for statistical analysis. To ensure the accuracy of the results, the PoW and PBFT consensus algorithms were also simulated in this environment so that they could be compared with the consensus model embedded with these algorithms.

This experiment was performed on a server with the following configuration: CPU model, AMD Opteron (tm); processor, 6128 @ 2.0 GHz; operating system, CentOS 7.4.1708; memory, 32G; and network card bandwidth, Gigabit. The simulation experiment program was written in GO and GO compiler was go-1.13.4.

4.1 | ECBCM performance comparison

In the first experiment of this group, the PoW consensus algorithm, PBFT consensus algorithm, ECBCM embedded with the PoW consensus algorithm, and ECBCM embedded with the PBFT consensus algorithm were compared in terms of their block production speed. The number of nodes participating in the consensus was set to 1000 and the size of the

TABLE 1 Experimental parameters

Variable	β_1	β_2	β_3	ρ_1	ρ_2	φ_1	φ_2	B_V	B_C	M	Z
Size	0.6	0.2	0.2	1	3	1	3	3×10^7	3×10^7	1×10^9	0.002

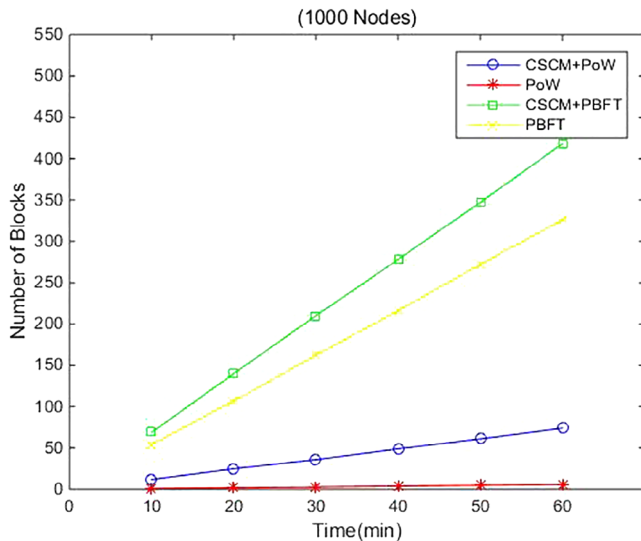


FIGURE 7 Comparison of the consensus algorithms

generated block was 1M. In the ECBCM, the witness node set contained 100 nodes, the participant node set contained 200 nodes, and the election period was 10 minutes. The running time was 60 minutes. The experiments were repeated 10 times and the average was taken to obtain the final result, as shown in Figure 7.

Since the ECBCM consensus model embedded with the PoW consensus algorithm has undergone a node selection at the beginning, the selected witness nodes are relatively high-quality nodes with high computing resources. Therefore, this experiment uses the PoW consensus process in the ECBCM consensus model. The difficulty value is appropriately reduced to increase the block production speed. It can be clearly observed in the Figure 7 that the consensus speed of the PBFT consensus algorithm is much faster than the PoW consensus algorithm. Even the PoW consensus algorithm embedded in the consensus model cannot be compared with the PBFT consensus algorithm in terms of consensus speed. However, through comparison, it is found that the ECBCM consensus model embedded with different types of consensus algorithms is faster than the original consensus algorithm in terms of consensus speed, which indicates that the ECBCM consensus model can improve the consensus efficiency of the embedded algorithm to a certain extent.

In the second experiment of this group, the ECBCM using the embedded PoW consensus algorithm and the ECBCM embedded in the PBFT consensus algorithm are compared with the existing consensus algorithm PoAh applied to the edge computing blockchain. In keeping with its experimental variables, the number of nodes participating in the consensus is set to 100, where the size of the witness node set is set to 5, the participant node set is set to 20, and the size of the generated block is fixed to 35 bytes. In the process of recording the generation of 10 blocks, the time it takes to generate each block is shown in Table 2 with the data of the PoAh consensus algorithm.

The table shows that, because the number of nodes and the block size are greatly reduced, the work efficiency of ECBCM is significantly improved, although the average time for ECBCM embedded PoW consensus algorithm to generate a block is about 27.1 seconds, which is much larger than the PoAh consensus algorithm. 3.4 seconds. But the average time for ECBCM to embed a PBFT consensus algorithm to generate a block is about 2.3 seconds, which is less than the PoAh consensus algorithm. In some scenarios where consensus speed is sought, ECBCM can meet the demand for consensus speed by flexibly changing the embedded consensus algorithm. ECBCM can still be competent in some edge computing environments composed of multiple sets of edge devices with weak data processing capabilities.

4.2 | ECBCM capability testing

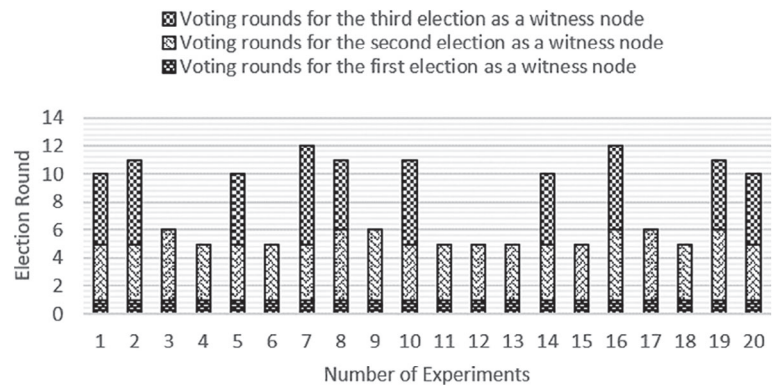
In this set of experiments, the ECBCM embedded with the PBFT consensus algorithm was used. The witness node set contained 10 nodes, the participant node set contained 20 nodes, the node election period was set to 10 minutes, and node 1 was predefined. The equity values consisted of a set of 100 scattered values between 5×10^6 and 6×10^7 , and the initial prestige coefficients of every node was set to 0.7.

In the first experiment of this group, node 1 with the largest equity value was set to generated the block abnormally. The process by which the prestige coefficient of the node was reduced to 0 was recorded. The experiment was repeated 20 times. The experimental results are shown in the histogram in Figure 8.

TABLE 2 Block generation time statistics

Number of Block	Time Taken for Block Validation in Seconds		
	PoAh	ECBCM+PoW	ECBCM+PBFT
1	3.3	26.22	1.76
2	3.4	27.01	2.3
3	2.8	27.3	2.51
4	4.02	27.9	2.14
5	2.9	26.83	1.96
6	3.8	27.3	2.63
7	3.4	27.57	2.5
8	3.42	27.14	2.87
9	3.38	26.72	2.41
10	3.23	27.2	2.15

Abbreviations: ECBCM, edge computing blockchain security consensus model; PBFT, practical Byzantine fault tolerance; PoW, proof-of-work.

FIGURE 8 Working status of abnormal high equity value nodes

According to the histogram, the abnormally working highstakes node was elected as a witness node after the first round of voting in every repetition. However, because of its evilness, the prestige coefficient was reduced and the node was not elected as a witness node in the second round of voting. The prestige coefficient recovered slowly due to normal operation as a trading node, which meant that the node still had the opportunity to be elected as a witness node. However, the continued evil of this node meant that it was elected at most three times before it permanently lost the opportunity to become a witness node. Therefore, the ECBCM has a certain degree of tolerance for occasional evil nodes, giving them the chance to recover the opportunity to become a witness node by actively participating in other work. However, if nodes are continuously evil, they will be permanently deprived of this right. In the second experiment of this group, the program ran for 20 rounds of voting without interruption and the initial equity value of the top 20 nodes was recorded, as shown in Figure 9. The graph shows that the chance of a node being selected as a witness node decreases as the equity value decreases. However, all of the nodes up to the 18th position had the opportunity to be elected as witness nodes, which shows that the ECBCM can reduce the concentration of node power to a certain extent, and that the set of witness nodes is not static.

4.3 | ECBCM malicious node detection and replacement

This group of experiments used the ECBCM embedded with the PBFT consensus algorithm. The number of nodes participating in the consensus was set to 1000 and the size of the generated block was fixed at 1M. The set of witness nodes in the ECBCM contained 100 nodes, the set of participant nodes contained 200 nodes, and the election period was 60 minutes. Over multiple experiments, 8, 12, 16, 20, 24, or 28 anomalous witness nodes were introduced with 10-, 20-, 30-, 40-,

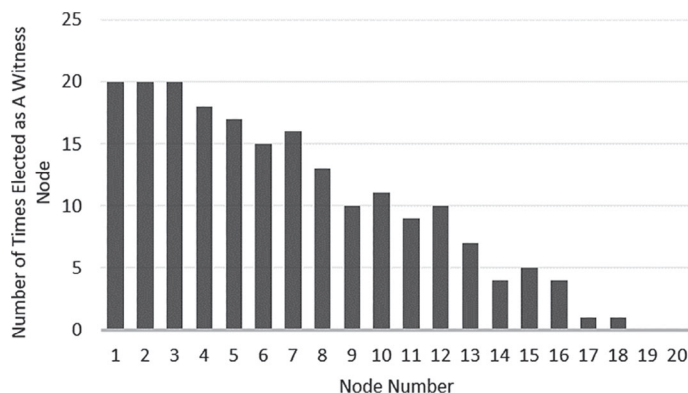


FIGURE 9 Witness node election times

Result Time(minutes)	10	20	30	40	50	60
Malicious node set	8	8	8	8	8	8
Number of discoveries	7	8	8	8	8	8
Malicious node set	12	12	12	12	12	12
Number of discoveries	8	12	12	12	12	12
Malicious node set	16	16	16	16	16	16
Number of discoveries	10	16	16	16	16	16
Malicious node set	20	20	20	20	20	20
Number of discoveries	13	20	20	20	20	20
Malicious node set	24	24	24	24	24	24
Number of discoveries	13	24	24	24	24	24
Malicious node set	28	28	28	28	28	28
Number of discoveries	21	28	28	28	28	28

TABLE 3 Number of abnormal witness nodes found at different times

50-, and 60-minute checkpoints. The number of abnormal witness nodes found at each time was recorded, as shown in Table 3.

As shown, the number of abnormal witness nodes found and replaced within 10 mins was not fixed. However, after 20 minutes, all the abnormal witness nodes were discovered and replaced, regardless of how many there were. Analysis showed that the ECBCM embedded with the PBFT consensus algorithm generated approximately 69 blocks in 10 minutes with 1000 nodes and 100 witness nodes. The sequence number of the abnormal node was based on the specified abnormal node number, a unique random number generated in the witness node sequence set. This makes it difficult to find abnormal witness nodes beyond rank 69, but over time, all of the abnormal witness nodes will be found and replaced. This shows that the ECBCM can accurately find and replace abnormal nodes.

5 | CONCLUSION

This study presents an ECBCM based on prestige. The consensus model introduces the idea of quantifying computing resources and mortgage time into equity, and uses the idea of the DPoS consensus algorithm to elect "quality nodes" for consensus, which reduces the consumption of resources and improves the speed of consensus; The idea of prestige is introduced, nodes behavior will affect the rise and fall of prestige value, which will affect the result of "high-quality node" election, make the node's power more decentralized, curb the chance of continuous evildoing of nodes; The idea of node replacement is introduced, and malicious nodes are replaced with normal nodes of the replacement, which further reduces the impact of node evils on the operation of the blockchain system; This consensus model can embed existing consensus algorithms into it, forming a new type of secure consensus algorithm oriented to the edge computing environment, which is a block in the edge computing environment. The chain system provides more diverse consensus support.

In summary, this study gives a solution to the problems of efficiency, security, and flexibility of the consensus mechanism in the edge computing model using blockchain technology. This is of great significance for promoting the close integration and development of edge computing technology and blockchain technology. Subsequent research will explore the possibility of introducing rewards and punishments to each node via benefit distribution strategies to mobilize the enthusiasm of each node in the consensus algorithm the maximum extent for the benefit of the blockchain system.

ACKNOWLEDGMENTS

This work was supported by the National Natural Science Foundation of China (61802086), Heilongjiang Provincial Science and Technology Department (GX18A008), and the China Scholarship Council(CSC).

ORCID

Shichang Xuan  <https://orcid.org/0000-0003-0332-0686>

REFERENCES

1. Stanciu A. Blockchain based distributed control system for edge computing. Paper presented at: Proceedings of the 2017 21st International Conference on Control Systems and Computer Science (CSCS); 2017:667-671; Bucharest, IEEE.
2. Li R, Song T, Mei B, Li H, Cheng X, Sun L. Blockchain for large-scale internet of things data storage and protection. *IEEE Trans Serv Comput*. 2018;12(5):762-771. <https://doi.org/10.1109/TSC.2018.2853167>.
3. Tantidham T, Aung YN. Emergency service for smart home system using Ethereum blockchain: system and architecture. Paper presented at: Proceedings of the 2019 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops); ; 2019:888-893; Kyoto, Japan, IEEE.
4. Xiong Z, Zhang Y, Niyato D, Wang P, Han Z. When mobile blockchain meets edge computing. *IEEE Commun Mag*. 2018;56(8):33-39. <https://doi.org/10.1109/MCOM.2018.1701095>.
5. Choi JS. A hierarchical distributed energy management agent framework for smart homes, grids, and cities. *IEEE Commun Mag*. 2019;57(7):113-119. <https://doi.org/10.1109/MCOM.2019.1900073>.
6. Yang Y, He D, Wang H, Zhou L. An efficient blockchain-based batch verification scheme for vehicular ad hoc networks. *Trans Emerg Telecommun Technol*. 2019;Early View:1-12.
7. Casado-Vara R, Corchado J. Distributed e-health wide-world accounting ledger via blockchain. *J Intell Fuzzy Syst*. 2019;36(3):2381-2386. <https://doi.org/10.3233/JIFS-169949>.
8. Rahman MA, Rashid MM, Hossain MS, Hassanain E, Alhamid MF, Guizani M. Blockchain and IoT-based cognitive edge framework for sharing economy services in a smart city. *IEEE Access*. 2019;7:18611-18621. <https://doi.org/10.1109/ACCESS.2019.2896065>.
9. Rahman MA, Hossain MS, Loukas G, et al. Blockchain-based mobile edge computing framework for secure therapy applications. *IEEE Access*. 2018;6:72469-72478. <https://doi.org/10.1109/ACCESS.2018.2881246>.
10. Kaur H, Alam MA, Jameel R, Mourya AK, Chang V. A proposed solution and future direction for blockchain-based heterogeneous medicare data in cloud environment. *J Med Syst*. 2018;42(8):156. <https://doi.org/10.1007/s10916-018-1007-5>.
11. Jayasinghe U, Lee GM, MacDermott Á, Rhee WS. TrustChain: a privacy preserving blockchain with edge computing. *Wirel Commun Mob Comput*. 2019;2019:1-17. <https://doi.org/10.1155/2019/2014697>.
12. Muthanna AA, Ateya A, Khakimov A, et al. Secure and reliable IoT networks using fog computing with software-defined networking and blockchain. *J Sens Actuator Netw*. 2019;8(1):15. <https://doi.org/10.3390/jsan8010015>.
13. Cao T, Zhong L, Xiao H, Song C, Yang S, Xu C. Credible and economic multimedia service optimization based on game theoretic in hybrid cloud networks. *Trans Emerg Telecommun Technol*. 2019;Early View:1-20. <https://doi.org/10.1002/ett.3779>.
14. Casado-Vara R, Prieta DF, Prieto J, Corchado JM. Blockchain framework for IoT data quality via edge computing. Paper presented at: Proceedings of the Proceedings of the 1st Workshop on Blockchain-Enabled Networked Sensor Systems. Association for Computing Machinery; 2018:19-24; New York, NY.
15. Xu J, Wang S, Bhargava BK, Yang F. A blockchain-enabled trustless crowd-intelligence ecosystem on mobile edge computing. *IEEE Trans Ind Inform*. 2019;15(6):3538-3547. <https://doi.org/10.1109/TII.2019.2896965>.
16. Ren Y, Zhu F, Qi J, Wang J, Sangaiah AK. Identity management and access control based on blockchain under edge computing for the industrial Internet of Things. *Appl Sci*. 2019;9(10):2058. <https://doi.org/10.3390/app9102058>.
17. Wan J, Li J, Imran M, Li D. A blockchain-based solution for enhancing security and privacy in smart factory. *IEEE Trans Ind Inform*. 2019;15(6):3652-3660. <https://doi.org/10.1109/TII.2019.2894573>.
18. Yang R, Yu FR, Si P, Yang Z, Zhang Y. Integrated blockchain and edge computing systems: a survey, some research issues and challenges. *IEEE Commun Surv Tutor*. 2019;21(2):1508-1532. <https://doi.org/10.1109/COMST.2019.2894727>.
19. Bach L, Mihaljevic B, Zagar M. Comparative analysis of blockchain consensus algorithms. Paper presented at: Proceedings of the 2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO); 2018:1545-1550; IEEE.
20. Tian Z, Shi W, Wang Y, et al. Real-time lateral movement detection based on evidence reasoning network for edge computing environment. *IEEE Trans Ind Inform*. 2019;15(7):4285-4294. <https://doi.org/10.1109/TII.2019.2907754>.

21. Xiao L, Wan X, Dai C, Du X, Chen X, Guizani M. Security in mobile edge caching with reinforcement learning. *IEEE Wirel Commun.* 2018;25(3):116-122. <https://doi.org/10.1109/MWC.2018.1700291>.
22. Shen M, Ma B, Zhu L, Mijumbi R, Du X, Hu J. Cloud-based approximate constrained shortest distance queries over encrypted graphs with privacy protection. *IEEE Trans Inf Forens Sec.* 2017;13(4):940-953. <https://doi.org/10.1109/TIFS.2017.2774451>.
23. Du X, Xiao Y, Guizani M, Chen HH. Defending DoS attacks on broadcast authentication in wireless sensor networks. Paper presented at: Proceedings of the 2008 IEEE International Conference on Communications; 2008:1653-1657; IEEE.
24. Wang D, Chen D, Song B, Guizani N, Yu X, Du X. From IoT to 5G I-IoT: the next generation IoT-based intelligent algorithms and 5G technologies. *IEEE Commun Mag.* 2018;56(10):114-120. <https://doi.org/10.1109/MCOM.2018.1701310>.
25. Nakamoto S, Bitcoin A. A peer-to-peer electronic cash system. *Bitcoin*, 2008;0:1-2. <https://bitcoin.org/bitcoin.pdf>.
26. Song T, Zhao Y. Comparison of blockchain consensus algorithm. *Comput Appl Softw.* 2018;35:1-8.
27. Vasin P. Blackcoin's Proof-of-Stake Protocol v2. *blackcoin.co.* 2014;71:1-2. <https://blackcoin.co/blackcoin-pos-protocol-v2-whitepaper.pdf>.
28. Lamport L. Fast paxos. *Distrib Comput.* 2006;19(2):79-103. <https://doi.org/10.1007/s00446-006-0005-x>.
29. Ding L, Han QL, Ge X, Zhang XM. An overview of recent advances in event-triggered consensus of multiagent systems. *IEEE Trans Cybern.* 2017;48(4):1110-1123. <https://doi.org/10.1109/TCYB.2017.2771560>.
30. Sukhwani H, Martinez JM, Chang X, Trivedi KS, Rindos A. Performance Modeling of PBFT Consensus Process for Permissioned Blockchain Network (Hyperledger Fabric). Paper presented at: Proceedings of the 2017 IEEE 36th Symposium on Reliable Distributed Systems (SRDS); 2017:253-255; IEEE.
31. Wang G, Shi ZJ, Nixon M, Han S. SMChain: a scalable blockchain protocol for secure metering systems in distributed industrial plants. Paper presented at: Proceedings of the Proceedings of the International Conference on Internet of Things Design and Implementation. Association for Computing Machinery; 2019:249-254; New York, NY.
32. IOE. Eosio technical white paper. *EOS. IO*; 2017. <https://github.com/EOSIO/Documentation>. Accessed December 18, 2017.
33. Wang B, Hu Y, Li S, Niu J. A blockchain consensus mechanism for educational administration system. Paper presented at: Proceedings of the 2019 IEEE 2nd International Conference on Electronics Technology (ICET); 2019:603-608; Chengdu, China; IEEE.
34. Gueta GG, Abraham I, Grossman S, et al. SBFT: a scalable and decentralized trust infrastructure. Paper presented at: Proceedings of the 2019 49th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN); 2019:568-580; IEEE.
35. Liu Y, Wang K, Lin Y, Xu W. LightChain: a lightweight blockchain system for industrial Internet of Things. *IEEE Trans Ind Inform.* 2019;15(6):3571-3581. <https://doi.org/10.1109/TII.2019.2904049>.
36. Han X, Yuan Y, Wang FY. A fair blockchain based on proof of credit. *IEEE Trans Comput Soc Syst.* 2019;6(5):922-931. <https://doi.org/10.1109/TCSS.2019.2938841>.
37. Zhuang Q, Liu Y, Chen L, Ai Z. Proof of reputation: a reputation-based consensus protocol for blockchain based systems. Paper presented at: Proceedings of the Proceedings of the 2019 International Electronics Communication Conference. Association for Computing Machinery; 2019:131-138; New York, NY.
38. Xu C, Wang K, Li P, et al. Making big data open in edges: a resource-efficient blockchain-based approach. *IEEE Trans Parall Distrib Syst.* 2019;30(4):870-882. <https://doi.org/10.1109/TPDS.2018.2871449>.
39. Deepak P, Saraju PM, Priyadarsi N, Elias K, Elias K. Proof-of-authentication for scalable blockchain in resource-constrained distributed systems. Paper presented at: Proceedings of the 2019 IEEE International Conference on Consumer Electronics (ICCE); 2019:1-5; Las Vegas, NV, IEEE.
40. Liu M, Yu FR, Teng Y, Leung VC, Song M. Computation offloading and content caching in wireless blockchain networks with mobile edge computing. *IEEE Trans Veh Tech.* 2018;67(11):11008-11021. <https://doi.org/10.1109/TVT.2018.2866365>.

How to cite this article: Xuan S, Chen Z, Chung I, et al. ECBCM: A prestige-based edge computing blockchain security consensus model. *Trans Emerging Tel Tech.* 2020;e4015. <https://doi.org/10.1002/ett.4015>