Peer-to-Peer Networking and Applications (2023) 16:2591–2606
https://doi.org/10.1007/s12083-023-01550-5

A cluster reputation-based hierarchical consensus model in blockchain

Yangyang Jiang[1] · **Yepeng Guan**[1,2,3]

Received: 16 March 2023 / Accepted: 21 August 2023 / Published online: 29 August 2023

Abstract
The blockchain has gained widespread attention due to its decentralized, secure, traceable, and immutable characteristics. However, current consensus protocols face the challenge of limited scalability and inadequate consideration of dynamic node behavior in large-scale networks. They cannot effectively adapt to the increasing number of nodes and the subsequent network traffic surge. Furthermore, their inability to sufficiently address dynamic node behavior poses a risk to the consensus process and compromises the overall reliability of the system. A cluster reputation-based hierarchical consensus model (CRHCM) has been proposed to addresses node dynamics and improves network scalability in this paper. The model introduces a reputation system that updates node reputations based on both current and historical behaviors during the consensus process. Node reputation fluctuations are assessed using a discrete Markov chain, which enables the identification of abnormal nodes and improves overall node reliability. Furthermore, a hierarchical structure has been proposed to improve scalability and reduce the communication complexity of blockchain by assigning nodes to the upper or lower layers through reputation and fluctuation levels. Experimental and theoretical evaluations demonstrate the effectiveness of CRHCM. The model achieves a balanced distribution of reputation values among all nodes and exhibits high scalability. It has excellent performance by comparisons with some other state-of-the-arts.

Keywords Consensus protocol · Reputation model · Markov chain · Hierarchy structure

1 Introduction

As the core technology of Bitcoin, blockchain entered the world stage in 2008 and has attracted extensive attention from the financial industry and society [1–3]. All blockchain nodes hold the ledger records, jointly maintain the ledger data, and eliminate the process of third-party management and verify transactions. It significantly reduces the risk of transactions [4]. Due to decentralization, security, traceability, and non-tampering, blockchain is often used to solve the trust and security issues of transactions. Nowadays, the application scenarios of blockchain have expanded from cryptocurrency to various industries, including healthcare [5], education [6], supply chain [7], the Internet of Things [8], and so on. rom cryptocurrency to various industries, including healthcare [5], education [6], supply chain [7], the Internet of Things [8], and so on.

In the blockchain, the consensus protocol plays an important role [9]. To ensure the security of data in the blockchain and maintain the continuous develoQWpment of the blockchain, nodes use a specific form of consensus mechanism to agree on the state of data in each block and the order in which transactions are executed [10]. Excellent consensus protocols can effectively improve the performance of blockchain, such as transaction throughput, scalability, and security [11]. Various consensus protocols have appeared in existing blockchain platforms, such as Proof of work (Pow) [1] in Bitcoin and Ethereum 1.0, Proof of Stake (POS) [12] in Ethereum 2.0, Practical Byzantine Fault Tolerance (PBFT) [13] and Raft [14] protocol in Hyperledger Fabric, etc. Pow [1] is one of consensus protocols applied in Bitcoin. Its core idea is to allocate block computing rights and block rewards based on the

This article is part of the Topical Collection: *3 - Track on Blockchain*
Guest Editor: Haojin Zhu

✉ Yepeng Guan
ypguan@shu.edu.cn

Yangyang Jiang
jiangyangyang2023@outlook.com

[1] School of Communication and Information Engineering, Shanghai University, Shanghai 200444, China

[2] Key Laboratory of Advanced Display and System Application, Ministry of Education, Shanghai 200072, China

[3] Key Laboratory of Silicate Cultural Relics Conservation (Shanghai University), Ministry of Education, Shanghai 200444, China

computational power of nodes. The message would be broadcasted throughout the distributed network for other nodes to verify after a block is generated [15]. However, the maximum transaction capacity of Bitcoin is 7 transactions per second (TPS), which can be increased to a maximum of 25 transactions without compromising consensus security by adjusting protocol parameters [16]. The Pow [1] protocol has the problem of high computational consumption and low transaction throughput. POS [12] is similar to the improvement of Pow [1]. It uses proof of stake to reward validators and nodes, dramatically reducing the waste of computing resources [17]. However, it does not solve the problem of unfairness, and it will cause the circulation of coins to be blocked. The Pow [1] and POS [12] protocols are proof-based and primarily utilized in public blockchains [18]. PBFT [13] is the preferred core consensus protocol for consortium blockchain for voting-based protocols. It solves the problem of low efficiency of the original Byzantine fault-tolerant protocol, reduces its complexity, and avoids consuming many computing resources [19]. However, PBFT [13] suffers from high communication complexity and low scalability.

Various consensus protocols have been proposed though, trust as an essential factor is often neglected. For an open blockchain, it is easy for some malicious nodes to join the network and damage the system [20]. Therefore, it is critical to address attacks from malicious nodes inside the system and to establish a trust relationship between nodes [21]. Reputation evaluation models can identify and isolate dishonest nodes according to the trust value based on game theory and mathematical calculation methods. It does not require a specific environment, tedious calculation processes, or vast amounts of data [22].

In this paper, A Cluster Reputation Based Hierarchical Consensus Model (CRHCM) has been proposed to address the dynamic nature of nodes in the consensus process and enhance network scalability. The model incorporates both reputation and fluctuation models to handle dynamic node behaviors. Besides a hierarchical structure is introduced to alleviate network congestion and improve scalability. The main contributions of this paper are as follows:

1. A Cluster Reputation Based Hierarchical Consensus Model (CRHCM) has been proposed to effectively addresses node dynamics and improves network scalability.
2. A reputation system has been proposed to update node reputations based on current and historical behavior during the consensus process. Discrete Markov chains has been selected to evaluate node reputation fluctuations. Abnormal nodes can be identified in a timely manner, and the overall reliability of nodes is improved.
3. A hierarchical structure has been proposed to improve scalability and reduce the communication complexity of blockchain by assigning nodes to the upper or lower layers through reputation and fluctuation levels.
4. Experimental and theoretical evaluations demonstrate the effectiveness of CRHCM. The model achieves a balanced distribution of reputation values among all nodes and exhibits high scalability. It has excellent performance by comparisons with some other state-of-the-arts.

The remainder of this paper is organized as follows. Some related work on reputation models is reviewed in Section 2. Cluster Reputation-based Hierarchical Consensus Model (CRHCM) is proposed in Section 3. Results and discussions are described in Section 4, and followed by some conclusions in Section 5.

2 Related work

Consensus mechanism guarantee that all nodes have reached the necessary agreement and are legally added to the jointly maintained ledger in the distributed blockchain. However, nodes may behave arbitrarily, such as disloyalty and malicious behavior. The communication complexity is high, which is not suitable for large-scale distributed networks. A lot of useful work has been done to ensure the efficiency and security of the consensus.

Reputation-based consensus protocol has been proposed called as proof of reputation (PoR) in [23]. In the PoR protocol [23], the reputation is constructed by considering the assets, transaction value, and consensus participation of each distributed node. A reputation mechanism based on game theory has been proposed in [24]. A miner is randomly selected as the pool manager to evaluate the satisfaction of each miner. The reliability of the mechanism cannot be proved by a randomly chosen pool manager in [24]. The reliability of nodes is brought by the reputation incentive model in [24] with limited throughput due to using POW [1] protocol still. To increase throughput and enhance scalability, a high-throughput optimization solution called FabricETP has been proposed in [25]. FabricETP addresses concurrent conflicts through two dimensions: intra-block conflicts and inter-block conflicts. It minimizes the number of conflicting transactions in intra-block conflicts by reordering their execution sequence, while for inter-block conflicts, it incorporates a cache-based mechanism to proactively terminate invalid transactions. A reputation-based incentive method (PORX) has been proposed in [26]. Design reward and punishment factors in the income payment function of reputation. An optimized practical Byzantine fault tolerance consensus protocol has been proposed based on the EigenTrust model (T-PBFT) in [27]. The reliability of nodes is improved through the reputation value though, the communication complexity, however, has not changed in T-PBFT [27]. It is not suitable for application in large-scale networks. A trusted consensus group is evaluated with a higher trust

value to prevent nodes with a lower trust value from participating in consensus. Trust-PBFT consensus protocol has been proposed in [28], which integrates the PeerTrust-based trust computing model and practical Byzantine fault tolerance (PBFT) consensus. It allows a small number of nodes with high trust values to participate in transactions.

In addition, various strategies have been employed in existing studies to address the communication complexity and scalability in consensus networks, such as grouping and hierarchical structures. The possibility of an agent talking to a partner of unknown reliability has been discussed for large distributed network environment in [29]. Agent's reputation is modeled using a form of personal capital called as reputation capital (RC), which is calculated from historical feedback during interactions with other devices. Although the concept of reputation capital has been introduced by [29], the detailed calculation and process to ensure reliable communication between agents is lacking. Two-stage methodology has been proposed to determine the number of groups and reputation threshold for joining the group in [30]. K-means clustering protocol is applied to the reputation capital value of the object to determine the number of groups. Things are dynamically moved from one group to another based on changing individual reputation capital. However, it does not provide a clear correlation between the reputation threshold and the credibility of correctly distinguishing group members. An efficient leaderless Byzantine consensus algorithm (DBFT) has been proposed in [31]. In order to achieve reliable and efficient Byzantine fault tolerance in the blockchain, it optimizes processes such as voting, rounds, message broadcasting, and verification. However, the processing strategy for Byzantine nodes is not discussed in detail. It will limit the effectiveness of the algorithm in dealing with malicious Byzantine nodes and keeping the protocol secure. A hierarchical Byzantine fault tolerant (HBFT) consensus protocol has been proposed in [32]. Node reputation is used to enhance the reliability of the consensus process in [32]. However, it does not consider that nodes with a high reputation may be malicious. When Byzantine nodes with high reputation value do evil, the reliability and security of the protocol will be reduced. A scalable multi-layer PBFT based consensus mechanism has been proposed in [33]. Communication complexity is in turn reduced by hierarchically grouping nodes into different layers and restricting communication within the group. Additionally, a grouped PBFT algorithm (GPBFT) has been proposed in [34], which selects a primary node and forms groups based on trust evaluation. Independent PBFT consensus has been performed by each consensus group to reduce communication complexity. However, it should be noted that the PBFT-based hierarchical structure has introduced higher computational and communication overhead, as well as additional burdens for node management and maintenance.

The dynamic behavior of nodes in the consensus process has been considered in CRHCM. It introduced both reputation and node reputation fluctuation to identify high-reputation Byzantine nodes, which ensures the overall reliability of the system. In addition, CRHCM introduces a two-layer hierarchical structure to reduce the additional burden of node management and maintenance. It can be applied to further enhance scalability and reduce communication complexity.

3 Cluster reputation-based hierarchical consensus model

A framework of the proposed CRHCM is shown as in Fig. 1. Some high-reputation group nodes are selected to be responsible for delivering and verifying messages in CRHCM.

The CRHCM model mainly includes four stages: reputation value assessment, node fluctuation level assessment, cluster construction process, and consensus. Each stage is executed sequentially. Blocks are not created and published until the end of the total node.

The client C initiates a consensus request to the blockchain, and all nodes N participate in the consensus. The node fluctuating state is determined by the change of the node reputation value list $[r_1, r_2,..., r_n]$. The reputation value model and node fluctuation state serve as the basis for consensus. Nodes with high reputation value and low volatility are selected to enter the consensus process and grouped reasonably. Since the nodes have high reliability during the consensus process, the efficiency of the consensus process will be significantly improved after grouping. Finally, evaluate the reputation value of the participating consensus nodes to fully prepare for the following consensus process.

3.1 Reputation value assessment

The proposed reputation value assessment model is described in this section. The reputation evaluation model is divided into in-cluster assessment and global reputation assessment, respectively. It aims to reduce the impact of Byzantine nodes on the blockchain and ensure the reliability of the consensus process. The model adopts nodes with a high reputation as primary ones to participate in the consensus and leads others to actively and effectively participate in the consensus.

3.1.1 In-cluster assessment

Considering that everyone is untrustworthy in a blockchain, nodes with normal behavior are more conducive to the stability of the blockchain. A novel model is proposed, which evaluates the nodes in the cluster by the ones with high reputation value in the cluster. On the other hand, the growth or

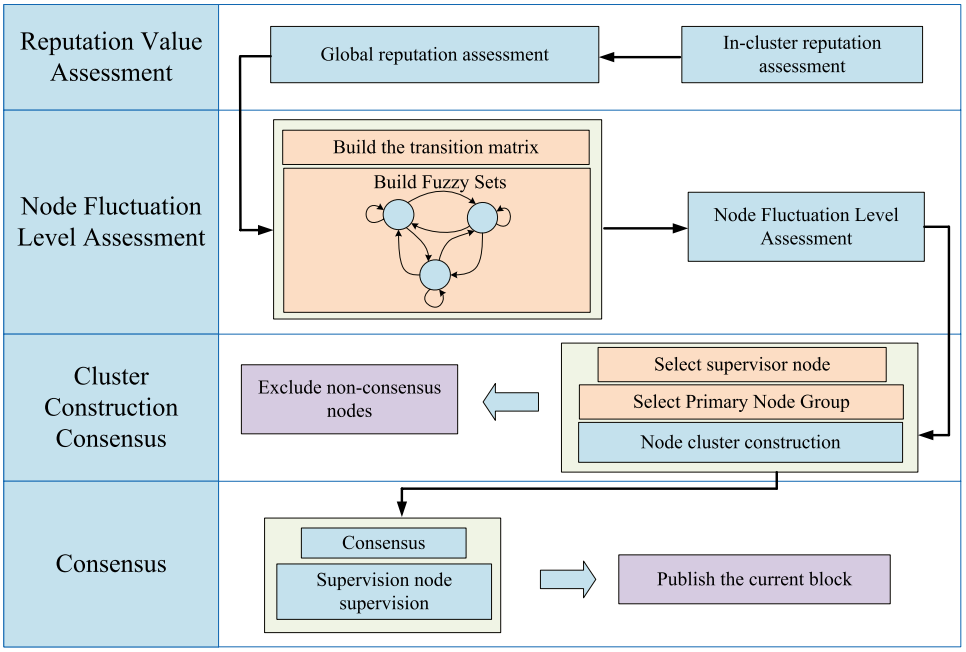

Fig. 1 Framework of the proposed CRHCM model

decline of reputation value can be obtained by quantifying the behavior of nodes.

The node behavior is divided into historical and current ones, respectively. The behavior of the node during the previous consensus process is called historical behavior. Current behavior represents that occurs during the current consensus process. The malicious behavior of the node can be judged according to the current behavior. If a node behaves maliciously in the current consensus, its historical node behavior will be traced back. The reputation value of the node is calculated by combining the historical behavior with the current behavior.

The quantitative standard is based on the historical behavioral impact of nodes in the consensus network. If the impact is enormous, the change in reputation will be correspondingly more significant. The main factors affecting the reputation value of nodes are proposed to quantify the historical behavior of nodes. Each node maintains its own reputation list $[r_1, r_2,..., r_n]$, and the list of reputation values is updated after each consensus. In historical behavior, the number of node participation M_i and the number of nodes participating in consensus and the number of consensus failures B_i are the main objects of concern. A higher number of failures indicates that the node is more unstable. Secondly, the number of timeouts T_i is an indicator that reflects the quality of communication between nodes. The greater the number of timeouts, the greater the impact on the process of quickly reaching a consensus, which in turn indicates the instability of the node. Finally, the number of times a node sends error messages E_i is an essential criterion for judging whether a node is doing evil or not. The more the number of errors, the greater the possibility that the node is a Byzantine node and the greater the threat to the network. Therefore, the historical behavior error rate is expressed as follows:

$$H_i = \frac{\sigma E_i + \beta B_i + \lambda T_i}{M_i} \tag{1}$$

where σ, β and λ are the weights of different behaviors in the historical process, respectively.

For the current consensus, two primary nodes are selected in the cluster as computing nodes for the evaluation model. Computing nodes are mainly responsible for performing reputation evaluation calculations on other nodes in the cluster. Other nodes collect factors that affect mutual communication during the consensus process. And send these factors to the dual primary node as the main basis for judging the reputation value. After the dual primary nodes complete the mutual verification and evaluation, update the node reputation value list.

According to the historical and current behavior characteristics above, calculate the reputation of nodes in the cluster as follows:

$$R_{cluster}(t) = \begin{cases} a, & primary \\ \frac{a}{2}, & replica \\ (W-1)exp\{W + cos(\frac{(1-H_i)\pi}{b})\}, & abnormal \end{cases} \tag{2}$$

where $R_{cluster}(t)$ represents the reputation value within the cluster completed by the current consensus, a represents the

size of the reward reputation value. W represents the number of times that abnormal nodes have abnormal factors in the current consensus process.

In Fig. 2, the node evaluation communication process within the cluster is shown. The node records the current behavior through the evaluation process of the dual master node, which is used to judge the node behavior under the current consensus. R_i represents the replica node, UP represents the upper primary node, and LP represents the lower primary node. When R_i nodes communicate with each other, they evaluate the current behavior of the nodes they communicate with. Then, the evaluated information is sent to the UP node and LP node. Finally, the dual-master nodes mutually verify the received evaluation information, and perform evaluation and distribution of reputation values in the cluster.

3.1.2 Global reputation assessment

Global reputation can provide more lasting momentum and stability for the blockchain and can balance the problem of excessive reputation gap between nodes. In other words, the global reputation is regarded as the total reputation according to the in-cluster reputation value assessment. When calculating the total reputation, the gap and change of the reputation value between nodes and the overall level need to be considered. This approach can prevent nodes with high reputation from doing evil and stimulate the activity of nodes in blockchain.

$$R_g(t) = R_g(t-1) + \omega(t) R_{cluster}(t) \tag{3}$$

The global reputation is defined as the combination of the current in-cluster reputation and the last consensus global reputation. The global reputation is calculated as follows:

$$\omega(t) = \exp\left\{-\eta\left(\frac{R_g(t-1) - \overline{R_g(t-1)}}{\delta}\right)^2\right\} \tag{4}$$

$$\delta = \sqrt{\frac{\sum_{i=1}^{N} \left(R_i(t-1) - \overline{R(t-1)}\right)^2}{N}} \tag{5}$$

where $\overline{R_g(t-1)}$ represents the average level of the previous consensus reputation value. $R_i(t-1)$ represents the $R_{cluster}(t-1)$ of node i. δ represents the standard deviation of the reputation value level. η is an adjustable parameter that specifies the expected range of variation, N represents the total number of participants in this consensus.

3.2 Node fluctuation level assessment

The reputation value model proposed above can increase or decrease the reputation value of the node through the behavior of the node, which improves the reliability of the node. However, it is not convincing to truly reflect the credibility of nodes and judge the credibility of nodes only based on the node reputation value. It is not ruled out that there are nodes with high reputation value doing evil. A node fluctuation state model has been proposed based on Markov chains to further evaluate node fluctuations. It improves the credibility of nodes with high reputation value.

Markov chain is a set of processes with discrete sets and random state spaces [35, 36]. For a random process $\{X_n, n = 0, 1, 2,...\}$, the Markov chain for $n \geq 0$, and state $i, j, i_0, i_1, \ldots, i_{n-1}$ is defined as follows:

$$\begin{aligned} P\{X_{n+1} = j | X_0 = i_0, X_1 = i_1, ...X_{n-1} = i_{n-1}, X_n = i\} \\ = P\{X_{n+1} = j | X_n = i\} \end{aligned} \tag{6}$$

where $X_n = i$ indicates that the process is in state i at time n, and $\{0, 1, 2, ...\}$ is the state space of the process.

The reputation value of each node participating in the consensus will fluctuate after participating in the consensus with different fluctuations. There are three levels for the fluctuation: slight, moderate, and violent. According to the reputation value model of the node, a violent fluctuation means that the node is likely to be an abnormal node or a

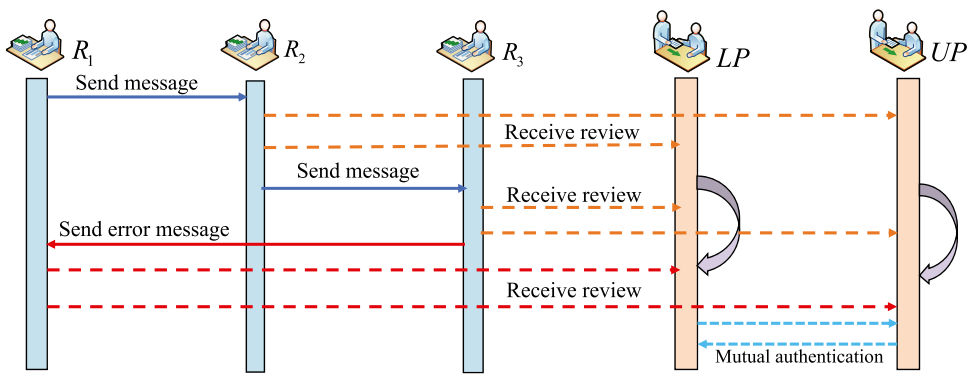

Fig. 2 In-cluster node evaluation communication process

Table 1 The definition of node fluctuation level

Symbol	Fluctuation	Representative meaning
L_s	Slight	The fluctuation of the node is slight, and the risk is small.
L_m	Moderate	Node fluctuations are moderate and may have some risk, but it can be tolerated.
L_v	Violent	Node fluctuates violently, it is abnormal or Byzantine node and must not participate in the consensus.

Byzantine node. The defined node fluctuation level is shown in Table 1.

According to the definition of the node fluctuation state, the representation of the node fluctuation state space L_t is as follows:

$$L_t = |L_s, L_m, L_v| \tag{7}$$

where L_t represents the node fluctuation state at time t. L_t contains L_s, L_m, L_v, which represent different states of the node, respectively.

Multiple random fluctuating states would be generated along with the node state changes as shown in Fig. 3. A node can transition from itself to its own state or its adjacent state. Besides a node in a slight state can also transition to a violent state when the difference in reputation value is too significant.

The reputation value of the node will be evaluated and updated after the nodes complete the consensus as shown in Fig. 4. Each node maintains a reputation list. The reputation difference list $[\Delta r_1, \Delta r_2, ..., \Delta r_t]$ is calculated from the reputation value list. However, since the reputation difference $[\Delta r_1, \Delta r_2, ..., \Delta r_t]$ of nodes changes is within a dynamic range for the actual process, it is difficult to get an accurate level of node fluctuations. In order to solve this problem, the reputation difference of nodes is divided into three levels and set up corresponding fluctuation fuzzy sets [37]. The node is in a state of slight fluctuations $\Delta r \in (-a, a)$, indicating that the behavior of the node is within the normal range of variation with a normal operating state. The node is in a moderately fluctuating state $\Delta r \in (-\tau a, -a)$, indicating that the behavior of the node is in a moderately fluctuating state with a certain weak risk. The node is in a highly volatile state $\Delta r \in (-\infty, -\tau a)$, indicating that the behavior of the node exists serious risks. τ is a selective range parameter, which depends on the determination of reputation parameters.

The state of node fluctuations is further evaluated by the state transition of node fluctuations and the Markov chain. Assume that the number of times the node is n times in the L_i state, and the number of times the node changes from the L_i state to the state L_j is m. The state probability P_{ij} of the node in the blockchain transferred from L_i to state L_j can be obtained with probability $P = m/n$. The Markov state probability transition matrix P_L is as follows:

$$P_L = \begin{pmatrix} P_{ss} & P_{sm} & P_{sv} \\ P_{ms} & P_{mm} & P_{mv} \\ P_{vs} & P_{vm} & P_{vv} \end{pmatrix} \tag{8}$$

where P_{ij} represents the probability of state transition from i to j.

The transfer matrix P_L can help predict the status of node reputation fluctuations. According to the predicted

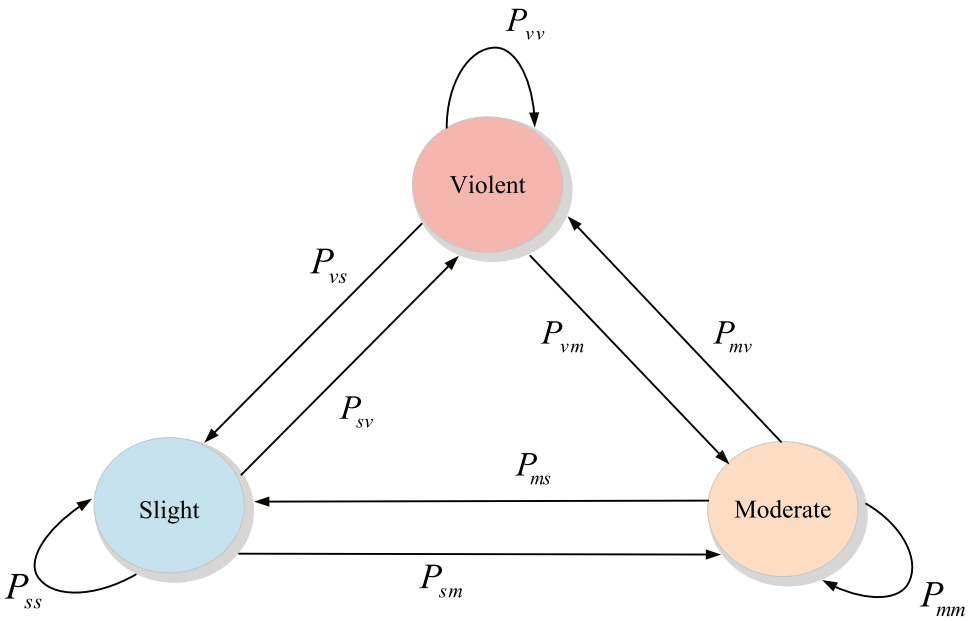

Fig. 3 Node fluctuating state transition

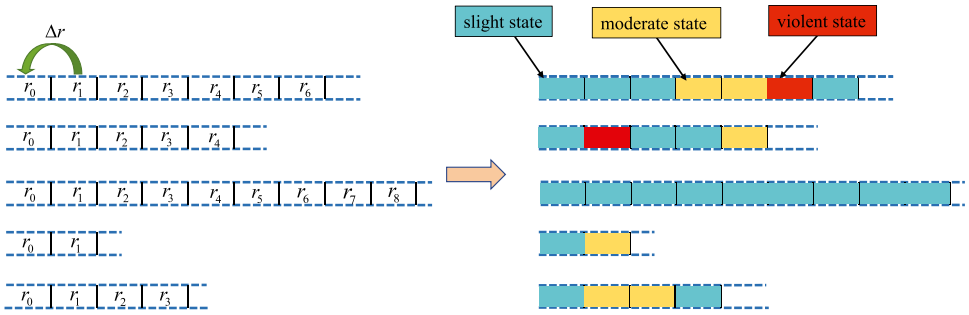

Fig. 4 Node fluctuation state changes

reputation fluctuation state and the previous fluctuation state records, the Byzantine probability of the node can be obtained. The reliability of the node can be judged to improve the security of the blockchain.

3.3 Cluster construction consensus

Since consensus suffers from high communication complexity and low network data throughput, it is not conducive to large-scale network communication at present. There is no suitable method to evaluate the reliability of the nodes, which leads to the evil of the nodes and the collapse of the blockchain.

A new two-layer topology based on reputation value and node fluctuation model is proposed to solve the above problems. The nodes are divided into different layers. The concept of node clusters is proposed as shown in Fig 5. The proposed topology can be used to improve the reliability of nodes and disperse the ability of nodes through reputation value and fluctuation model. Besides, it can reduce the complexity of communication and improve the data throughput of the network.

The nodes participating in the consensus are divided into upper and lower layers. The upper and lower layers are divided into multiple upper layer and lower layer node groups, respectively. Each upper-layer node group includes three nodes with different divisions of labor, namely the upper primary node *UP*, verification node *V* and supervisory node *S*. The upper primary node *UP* is responsible for sending and receiving messages from clients. The verification *V* node is to verify the correctness of the messages of the upper primary node UP. The supervisory node S is mainly responsible for the consistency of the cluster. The lower-level node group includes a low primary node LP and replica nodes *R*. The lower primary node *LP* communicates with the upper primary node *UP*. Replica nodes *R* transmit and send information and add it to their own ledger.

Assuming that nodes are driven by interests, nodes with high reputation value and low volatility are more credible. There is no guarantee that nodes participating in the blockchain are reliable and credible, so not all nodes can participate in consensus. In order to better maintain consensus consistency and reduce the malicious behavior of nodes, a consensus cluster is constructed by selecting nodes with high reputation and low volatility. The characteristics of high reputation and low volatility are possessed by the constructed consensus cluster. It can improve node stability and reduce communication complexity.

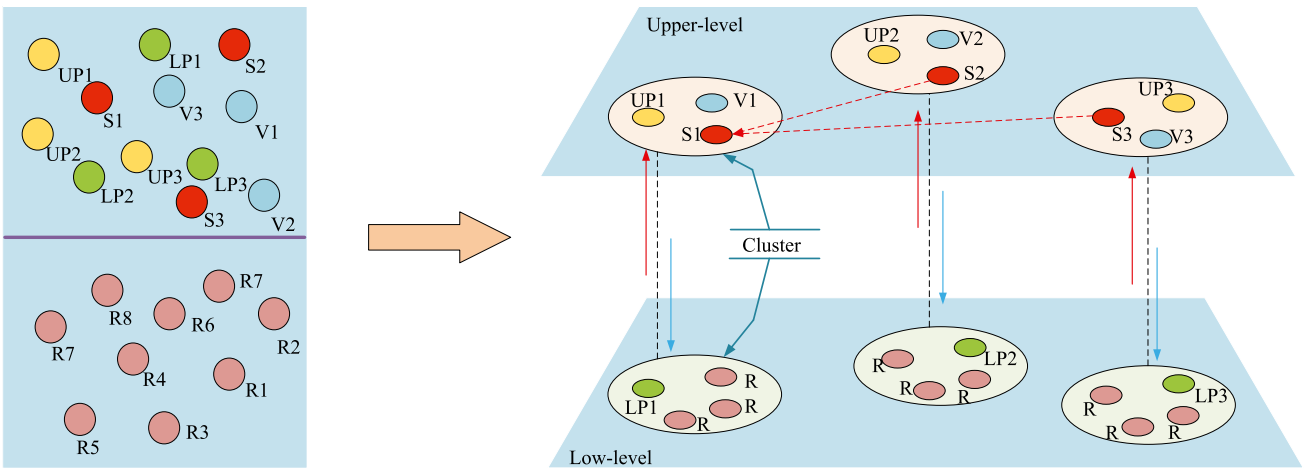

Fig. 5 Cluster construction process

The nodes participating in the consensus are divided according to the reputation value and reputation fluctuation model as shown in Algorithm 1. Untrustworthy nodes are excluded according to the fluctuation of the last consensus and the prediction of the next moment in the Markov chain. Select some nodes with high reputation and low volatility as the upper master nodes by counting the number of participating nodes and the number of clusters. The supervision node is mainly responsible for the abnormal supervision of the upper nodes and the reputation evaluation in the cluster. The node with a high reputation value is selected as the supervision node.

3.4 Consensus

In this section, the proposed CRHCM consensus process will be described in detail. The proposed CRHCM consensus can resist the evil ability of nodes and guarantee eventual consistency.

The blockchain is difficult to expand with low communication efficiency for the large-scale network environment. A CRHCM consensus protocol has been proposed. The protocol first groups nodes to improve the scalability of the network. Node matching strategy has been used to reduce communication complexity. The consensus process of CRHCM is given as Fig. 6 including two different communication situations with Byzantine nodes.

The main processes are described as follows:

Request phase The client plays the role of sending consensus requests and receiving network consensus results, and finally reaching a consensus.

In the request phase, the client *C* sends a request to the primary node *UP*, the verification node *V*, and the supervisory node *S* of the upper cluster at each cluster. The upper primary node *UP*, the verification node *V* and the supervisory node *S* cache the messages sent by the client *C* and verify the signature of the client, respectively. The request message format is:

$$< REQUEST, m, t, c, n, sig_c > \tag{9}$$

where *m* is the message requested by the client, *t* is the timestamp. *c* is the number of the client. *n* is the random number selected by the client. sig_c represents the signature of the client as the view number.

Pre-verify phase The upper-level primary node *UP* at each cluster sends the message received from the client *C* to the lower-level primary node *LP* and the verification node *V*, respectively. The verification node *V* is to verify the correctness of the message sent by the primary node. The format of the pre-verify message is:

$$< PRE - VERIFY, m, t, n, g, sig_{up} > \tag{10}$$

where *g* is the cluster number and sig_up is the signature of the upper primary node.

Pre-prepare phase After the lower-level primary node *LP* receives pre-verify message of the upper-level primary node *UP*, it broadcasts a Pre-Prepare message to all replica nodes in the lower layer. The format of the pre-prepare message is:

$$<< PRE - PREARE, n, d, g, sig_{lp} >, m > \tag{11}$$

where *d* is the digest of the request message, and the node verifies whether the digest information of the message *m* is the same as *d*.

Prepare phase Once all replica nodes have passed the verification process, they proceed to the prepare phase. During this phase, replica nodes send messages to the corresponding receiving replica nodes. However, in the prepare phase, two scenarios may occur if Byzantine nodes are present. A node

Algorithm 1 Cluster Construction Consensus

Input: List of reputation values $[r_1, r_2, ..., r_n]$ of all nodes N and the degree of fluctuation L.
Output: The number of consensus clusters.

```
Consensus Cluster ← ∅
L_t + 1 = predict the fluctuation at time t + 1
for N_i ∈ N do
    if L_t = L_v or L_t + 1 = L_v then
        Exclude N_i from consensus
    else
        Add N_i into Consensus
    end if
end for
Get the number of consensus clusters
Select high reputation and low volatility node
Randomly distribute other nodes
```

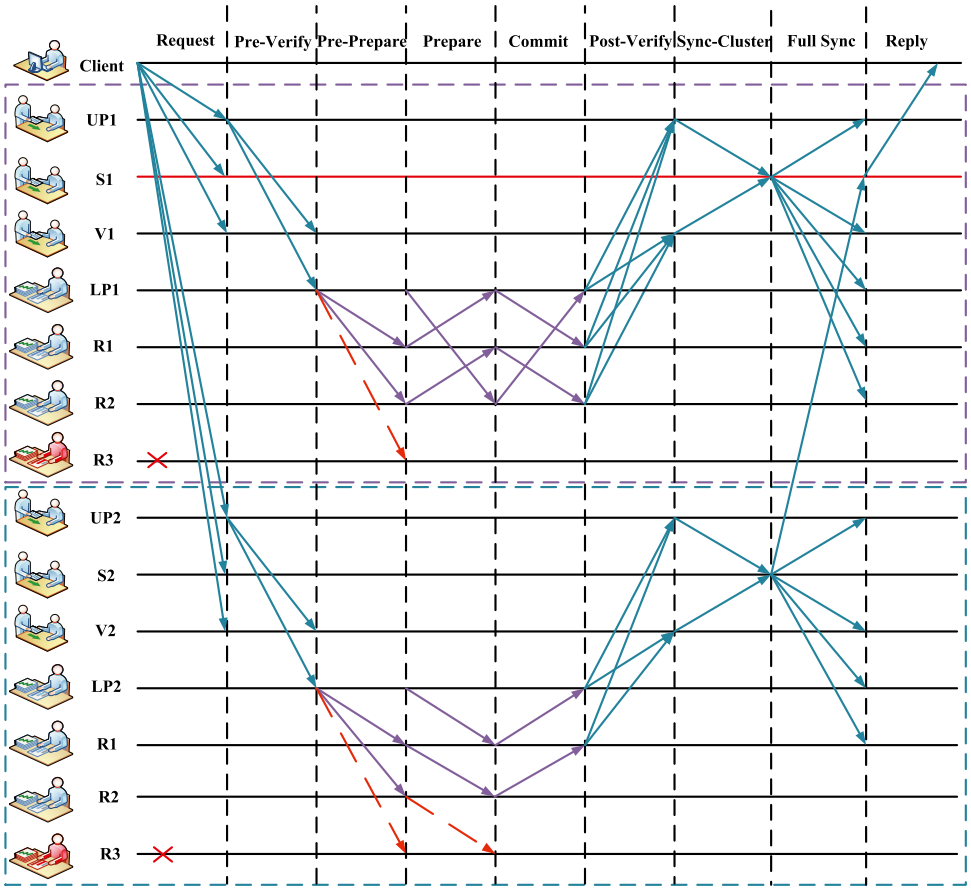

Fig. 6 Consensus process of CRHCM

does not match with a Byzantine node, as depicted in the upper layer of Fig. 6. A node matches with a Byzantine node as shown in the lower layer of Fig. 6. The latter case represents the worst-case scenario in the system. Upon receiving the prepare messages, the respective receiving replica nodes record the messages in their local message logs. The format of the prepare messages is as follows:

$$< PREPARE, n, d, g, sig_{s\_r} > \tag{12}$$

where sig_{s_r} is the signature of the sending replica node in the preparation phase.

Commit phase After receiving the message from the sending replica node in the preparation stage, the matching receiving replica node verifies and saves the message, replies to the corresponding sending replica node. When the sending replica node receives the commit message sent by the receiving replica node, it will verify the correctness and matching of the signature, and verify whether the view number n matches the digest d at the same time. The format of the commit message is:

$$< COMMIT, n, d, g, sig_{r\_r} > \tag{13}$$

where sig_{r_r} is the signature of the receiving replica node.

Post-verify phase When the node completes the commit phase, it will send a message to the upper-level primary node UP and the verification node V to judge the consensus of the lower-layer nodes. The message format is:

$$< POST - VERIFY, n, d, g, verify(), sig_{s\_r} > \tag{14}$$

where $verify()$ function is only used to judge whether the consensus of the lower-level nodes is successful. If it is true, the consensus of the lower-level nodes is successful, otherwise, it fails.

Sync-cluster phase When the upper-layer primary node and the verification node receive the messages in the post-verify phase, the number of messages is more significant than half of that of the lower-layer nodes, they send a cluster consensus reaching message to the supervisory node, respectively. The format of these messages for sync-cluster is as follows:

$$< SYNC - CLUSTER, n, d, g, state(), sig_{up} > \tag{15}$$

$$< SYNC - CLUSTER, n, d, g, state(), sig_{v} > \tag{16}$$

If the number of success messages is more than half of the lower nodes, the state is true, otherwise it is false.

Full-sync phase After receiving the message sent by the primary node and the verification node in each cluster, determine whether the state of the two is the same. If they are the same, the consensus result is considered correct, and the supervisory node at each cluster sends a message to all nodes at the cluster. The message format is:

$$< FULL - SYNC, g, state(), sig_s > \tag{17}$$

At the same time, the slave supervision node will send the consensus status to the master supervision node. The message format is:

$$< FULL - SYNC, g, state(), \mathrm{sig}_{s\_s} > \tag{18}$$

where sig_{s_s} represents the signature of the message sent by the slave supervisory node to the primary supervisory node.

Reply phase When the primary supervisory node receives the consensus results of half of the clusters, it will reply to the client with a message. The format of the reply message is:

$$< REPLY, t, c, n, sig_s > \tag{19}$$

where sig_s represents the signature of the primary supervision node.

4 Results and discussions

In order to evaluate the proposed CRHCM model, an experimental network is constructed in Rust programming language as a prototype implementation of the CRHCM. The CRHCM prototype is deployed on a Linux machine configured with an i5-7300HQ CPU, 2.50GHz, and 32GB RAM.

4.1 Evaluation of CRHCM model

Some parameters for the CRHCM are discussed to get a fair evaluation with some other models as following.

4.1.1 In-cluster weight evaluation

Intra-cluster reputation is an important indicator that affects the reliability of nodes. Historical behavioral parameters are assessed using the Analytic Hierarchy Process (AHP) approach [38]. AHP is defined as a decision-making approach that involves constructing multiple criteria into a hierarchy and assessing the relative importance of these criteria [38].

AHP [38] establishes a pairwise comparison matrix by comparing criteria. It performs eigenvalue decomposition and normalizes the eigenvectors. Simultaneously, AHP [38] computes the Consistency Ratio (CR) to assess the consistency of the pairwise comparison matrix to ensure reliable and consistent decision outcomes. If $CR \leq 0.1$, the consistency of the judgment matrix is considered acceptable. Otherwise, adjustments and revisions need to be made to the matrix to meet the consistency requirements.

In the experiments, 100 rounds of intra-cluster consensus are simulated. Normal nodes can respond in time. Malicious nodes will give corresponding malicious behaviors randomly. Some experimental results are given in Table 2.

It can be seen from Table 2 that the historical behavior parameters are different in AHP decision-making model [38] at different historical behaviors. The CR is minimum at $\sigma = 0.13$, $\beta = 0.20$, $\lambda = 0.67$, respectively. In the experiments, σ, β, λ was set 0.13, 0.20, and 0.67, respectively. They remained the same in the experiments to ensure a balanced consideration of different behavior types.

To get a reasonable value of b in (2), the value of b is changed from 0.6 to 1.4 at an interval of 0.2. And when W=1, the reputation values within exceptional nodes are all 0 and cannot be altered, making it impossible to obtain the precise value of parameter b. Therefore, the experimental parameter W in (2) was set to 2 and remained unchanged in subsequent experiments.Some experimental results are shown as Fig. 7.

One can find from Fig. 7 that b exhibits distinct behavior under varying conditions. In-cluster reputation monotonically increases with the historical anomaly error rate when $b > 1$. In-cluster reputation initially increases with the increase of the historical anomaly error rate when $b < 1$, subsequently decreases with further increments of the error rate. We set $b = 1$, and maintain the same in experiment.

4.1.2 Global reputation weight evaluation

To assess the dispersion of global reputation, a coefficient of variation (CoV), which is taken as a standardized standard deviation [39], is selected as a measure. Considering that nodes may not have sufficient historical behavior data for evaluation at the initial stage, the reputation of each node entering the blockchain is randomly assigned an integer value between 2 and 4. Subsequently, the reputation of the primary

Table 2 Weight of Indicators of Historical Behavior

σ	β	λ	CR
0.125	0.208	0.667	3.83E-16
0.133	**0.200**	**0.667**	**0**
0.190	0.238	0.571	3.83E-16
0.073	0.268	0.659	1.15E-15
0.050	0.200	0.750	1.53E-15

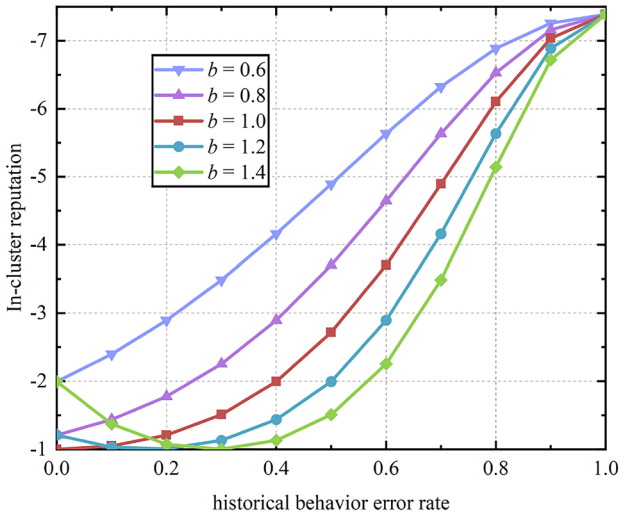

Fig. 7 Function $R_{cluster(t)}$ with different b

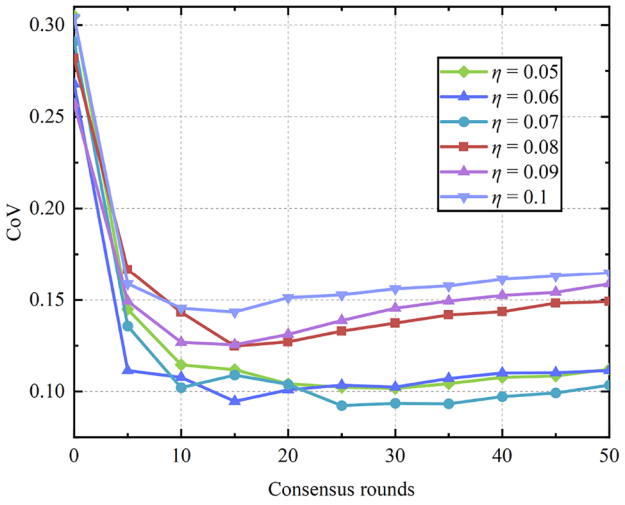

Fig. 8 *CoV* with different η

node within the cluster undergoes a modification with a value of $a = 2$ in (2). The reputation of the replica nodes is adjusted to 1 accordingly. Furthermore, a subset of nodes is selected to participate in the consensus process. To get a reasonable value of η in (4), we change it from 0.05 to 0.1 at interval of 0.01. Some experimental results are shown in Fig. 8.

It can be seen from Fig. 8 that there are different influences on the stability with different η. *CoV* decreases rapidly at first, gradually changes slowly with the increase of consensus rounds, and finally tends to a steady value. This suggests that greater dispersion among nodes leads to a more polarized global reputation trend, which in turn can significantly diminish node enthusiasm. In order to maintain the stability of blockchain and improve the enthusiasm of the nodes, *CoV* should be set less than 0.15 [40]. It can be found from Fig. 8 that when η is set 0.08, *CoV* is close to 0.15. In subsequent experiments, η in (4) is set 0.08 and keep it the same.

4.1.3 Evaluation of node fluctuation status

To evaluate the effectiveness of the node fluctuation mechanism, the consensus process involved the participation of three node clusters. Each cluster comprised a total of 10 nodes. We have simulated a realistic scenario and assessed the impact of node fluctuations on the consensus outcomes in this manner. The initial state of the nodes is a slight fluctuation state. The initial state of the nodes is a slight fluctuation state. There are 10 malicious nodes, and the malicious behavior of the nodes is set randomly. In order to satisfy the fuzzy set condition, if the current number of errors exceeds 2, it is a severe fluctuation state, so τ was set to 20. To visually display the trend, we conducted 5 sets of experiments, some average, maximum and minimum values of node fluctuation states in the consensus process under multiple trials are shown in Fig. 9.

It can be found from Fig. 9 that the node fluctuation state is a slight fluctuation state when entering the consensus. After multiple rounds of consensus, malicious nodes are detected as moderately fluctuating states and violently fluctuating states, respectively. When the state of violent fluctuation is reached, the node cannot participate in the subsequent consensus process and is excluded from the consensus process. Therefore, the number of nodes is constantly decreasing and will stabilize after 50 rounds of consensus. This shows that the node fluctuation model can well identify malicious nodes and eliminate consensus, which is beneficial to the stability of blockchain.

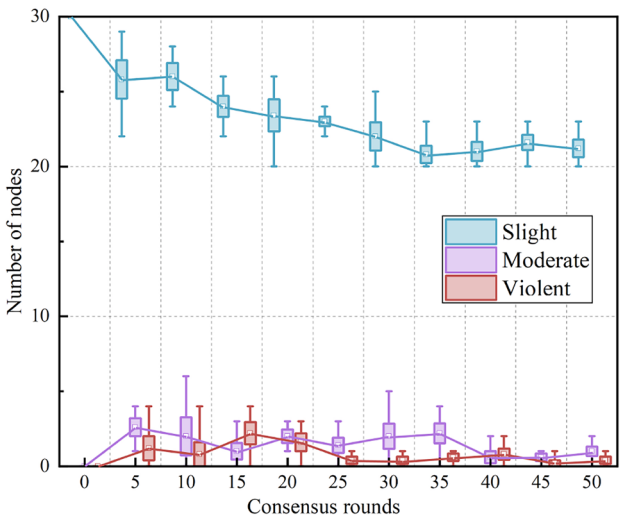

Fig. 9 Node fluctuation state changes in the consensus process

4.2 Evaluation and comparisons with some consensus protocols

In this section, the CRHCM would be evaluated with some consensus protocols in complexity, fault tolerance, and throughput.

4.2.1 Communication complexity

Communication complexity is one of the essential indicators to measure the efficiency of consensus protocols. The communication complexity is $O(N^2)$ for PBFT [13], T-PBFT [27], and GPBFT [34], respectively. They have high communication complexity, which makes the network scalability poor. In our CRHCM, nodes with high volatility and low reputation value will be excluded from the consensus, and other nodes will participate in the consensus. At the same time, the clustering strategy is used to further reduce the number of communications in the consensus process, which can improve the scalability of the network, and be applicable to large-scale networks.

Assume that there are u clusters participating in the consensus process. The lower layer of each cluster is n nodes, the number of lower nodes of each cluster is the same. The entire cluster has a total of $n+3$ nodes. Then the total number of nodes N participating in the consensus is $N = u(n+3)$. The number of communications for all clusters is $u(6+6n)$. The communication complexity is $O(N)$. The number of communications is estimated to be around $2n^2$ in PBFT [13]. The communication times is approximately n^2 in T-PBFT [27] with communication complexity of O(N2). In GPBFT [34], the communication time is $2(u-1)+2un(n-1)$. To further evaluate the communication efficiency among PBFT [13], T-PBFT [27], GPBFT [34], and CRHCM, since GPBFT [34] and CRHCM are layered protocols, the u is set as 3 for convenience in the experiment. Some results are illustrated in Fig. 10.

One can find that CRHCM does not significantly increase the communication time, while PBFT [13], T-PBFT [27] and GPBFT [34]increase with the number of nodes. It highlights that CRHCM has superior communication time performance by comparisons with PBFT [13], T-PBFT [27], and GPBFT [34], respectively.

4.2.2 Byzantine fault tolerance analysis

Byzantine fault tolerance is one of critical standards for consensus, which refers to the scope and probability of allowing errors. The maximum fault tolerance of PBFT [13] is given by $f = (n-1)/3$. However, although GPBFT [34] improves network scalability, it does not enhance the fault tolerance, which remains $f = (n-1)/3$. For the CRHCM protocol, the

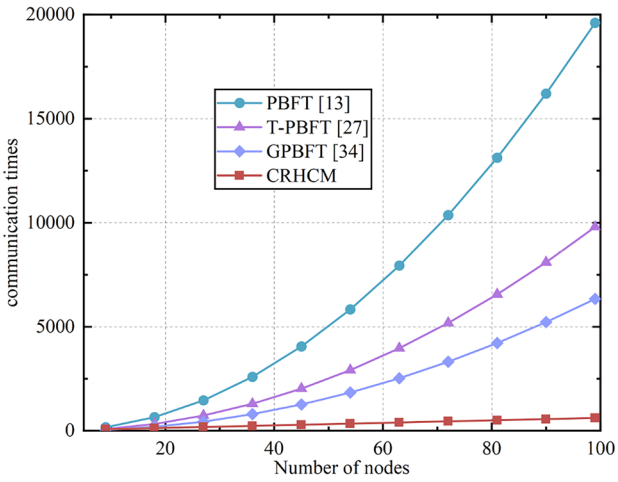

Fig. 10 The communication times under different number of nodes

number of upper nodes in each cluster is always equal to 3 and has a high reputation value. Most of the lower layers are nodes with low reputation values, so the fault tolerance rate of the lower nodes will be analyzed emphatically.

Assume that the lower layer of the cluster has n nodes and the number of nodes is greater than or equal to 4. The node-to-node matching communication method is used in the lower layer, which makes the communication of nodes appear in two situations, as shown in Fig. 6. The first case is when a node fails, there is no node to match. The upper node can receive more than half of the communication from the lower node. The maximum number of fault tolerance in the cluster is $(n-1)/2$. The second case is when a node fails and there are nodes to match. When this happens, it is the worst case. The upper-layer nodes can only receive half of the lower-layer nodes at most to reach a consensus, and the maximum number of fault tolerance in the cluster is $(n-2)/2$. For u clusters, the overall fault tolerance number is $(3un-2u)/4$. The fault tolerance number is $(2n-1)/3$ [32] for T-PBFT [27]. To get fault tolerance rate changes with the number of nodes, a comparison has been made between the Byzantine fault tolerance rate of CRHCM and that of PBFT [13], T-PBFT [27] and GPBFT [34]. Some results are given in Fig. 11.

One can note that the fault tolerance rate gradually increases and tends to a stable state with the increase of the number of nodes. The maximum fault tolerance of PBFT [13] is close to 1/3, while GPBFT [34] does not improve the fault tolerance of PBFT [13]. GPBFT [34] has the same fault tolerance as PBFT [13]. On the other hand, the maximum fault tolerance of T-PBFT [27] is close to 2/3. The maximum fault tolerance of CRHCM is close to 3/4. CRHCM has a fault tolerance rate close to 3/4. It further highlights that CRHCM has the best fault tolerance performance among the investigated consensus protocols.

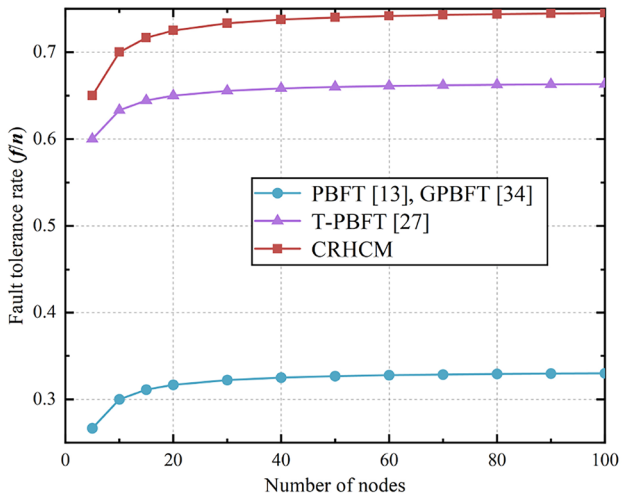

Fig. 11 Fault tolerance rate changes with the number of nodes

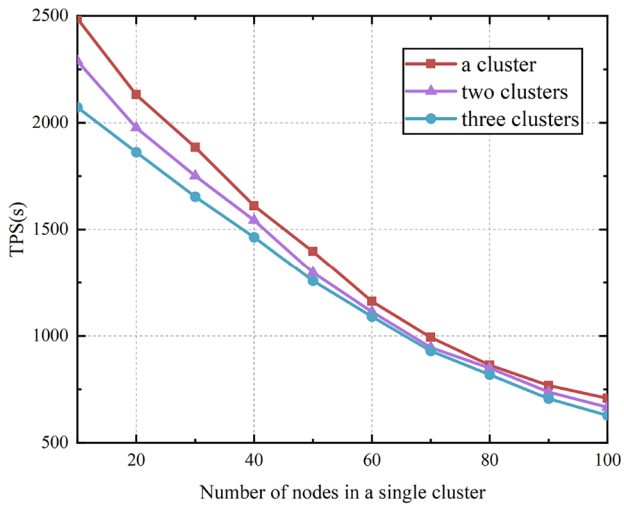

Fig. 12 *TPS* under different number of clusters

4.2.3 Throughput analysis

Throughput is another important indicator to measure the performance of the consensus protocol in the blockchain. It represents the ability to process transactions [41]. The higher the throughput, the higher the efficiency of the consensus mechanism and the stronger the ability to handle transactions. *TPS* in a specific environment is calculated as:

$$T_{tps} = \frac{\sum_{i=1}^{n} T_i}{\Delta t} \tag{20}$$

where Δt is a certain time period, $\sum_{i=1}^{n} T_i$ is the total number of transactions processed within the time period Δt.

In this experiment, the total number of transactions and the number of *TPS* are calculated for a period of one minute. The client continuously sends transactions to the consensus node. The number of tested clusters increases sequentially. The network size within each cluster increases sequentially from 10 to 100 nodes. When the number of clusters increases to 3, the number of nodes is 3 times that of a single cluster. The experimental results are shown in Fig. 12.

It can be seen from Fig. 12 that the overall trend of *TPS* decreases as the number of nodes increases. As the number of clusters increases, *TPS* slightly decreases with data exchange and reputation value processing between nodes. But it is worth noting that as the number of clusters increases, the number of nodes will increase exponentially. Therefore, for a large-scale network environment, *TPS* can be effectively improved in our model.

To further test CRHCM, PBFT [13], T-PBFT [27] and GPBFT [34] are selected to evaluate in *TPS*. Some results are shown in Fig. 13.

It can be found from Fig. 13 that TPS of PBFT [13], T-PBFT [27] and GPBFT [34] is much lower than that of CRHCM. Some reasons are as follows. Three-stage communication method is used in PBFT [13] for communication, there is a large amount of data exchanged during each communication segment. This increases communication complexity. GPBFT [34] utilizes the grouping of PBFT [13] to achieve consensus. The underlying structure of PBFT [13], however, remains unchanged in GPBFT [34]. The feature reputation model is used by T-PBFT [27] to reduce part of the communication complexity. But the communication complexity is still too high to be suitable for large-scale networks. It also does not take into account the situation

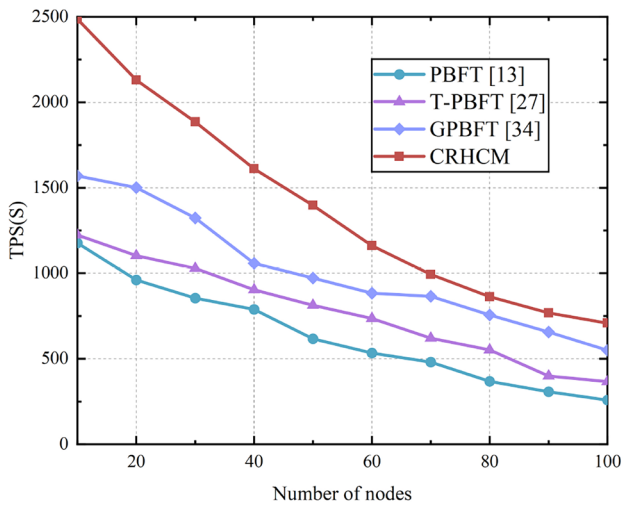

Fig. 13 *TPS* under different number of nodes

Table 3 Comparison with other consensus protocols

	Byzantine fault tolerance	Node reliability	Communication complexity	Primary node	Scalability
PBFT [13]	Yes	Low	$O(N^2)$	Single	Low
DBFT [31]	Yes	Low	$O(N^2)$	Single	High
RAFT [14]	No	Low	$O(N)$	Single	High
T-PBFT [27]	Yes	Medium	$O(N^2)$	Single group	High
GPBFT [34]	Yes	Medium	$O(N^2)$	Multi-group	High
CRHCM (proposed)	**Yes**	**Medium**	$O(N)$	**Multi-group**	**High**

of high reputation value nodes doing evil. In CRHCM, the complexity is reduced through the hierarchical structure, and the lightweight reputation and fluctuation model is used to improve the credibility of nodes while reducing the calculation time of reputation. It significantly reduces the communication complexity of the network and improves *TPS*.

4.2.4 Protocol comparison summary

When compared to typical consensus protocols, the CRHCM stands out for its exceptional performance among the studied protocols. Although RAFT [14] is fault-tolerant, it cannot tolerate Byzantine problems. For PBFT [13], DBFT [31], T-PBFT [27], and GPBFT [34] protocols, the communication complexity is $O(N^2)$ and the scalability is low. Moreover, in the case of T-PBFT[27], the primary node is a single group, whereas in GPBFT[34], the primary node consists of multiple groups. Typically, having multiple groups as primary nodes tends to enhance the security of the system. The CRHCM protocol can reduce the communication complexity to $O(N)$ in a highly decentralized blockchain network. In addition, the proposed reputation value model enables nodes to have higher reliability. The performance results of the above protocols are summarized in Table 3 in order to reflect the differences of the protocols more clearly.

5 Conclusion

A cluster reputation-based hierarchical consensus model (CRHCM) has been proposed to addresses node dynamics and improves network scalability. The model introduces a reputation system that updates node reputations based on both current and historical behaviors during the consensus process. Node reputation fluctuations are assessed using a discrete Markov chain. Abnormal nodes can be identified in a timely manner, and the overall reliability of nodes is improved. Furthermore, a hierarchical structure has been proposed to improve scalability and reduce the communication complexity of blockchain by assigning nodes to the upper or lower layers through reputation and fluctuation levels. Experimental and theoretical evaluations demonstrate the effectiveness of CRHCM. The model achieves a balanced distribution of reputation values among all nodes and exhibits high scalability. CRHCM has excellent performance by comparisons with some other state-of-the-arts.

Author contributions Yangyang Jiang and Yepeng Guan: Conceptualization, Investigation, Software, Methodology, Writing - original draft, Validation, Writing - review & editing.

Funding This work is supported in part by National Key R &D Program of China (Grant No. 2020YFC1523004).

Data availability Not applicable.

Declarations

Ethical approval Yes.

Consent for publication Yes.

Conflict of interest The authors declare that they have no competing interests.

References

1. Nakamoto S (2008) Bitcoin: A peer-to-peer electronic cash system. Decentralized Business Review 21260
2. Zheng Z, Xie S, Dai HN et al (2018) Blockchain challenges and opportunities: A survey. Int J Web Grid Serv 14(4):352–375
3. Bamakan SMH, Motavali A, Bondarti AB (2020) A survey of blockchain consensus algorithms performance evaluation criteria. Expert Syst Appl 154(113):385
4. Mohsenzadeh A, Bidgoly AJ, Farjami Y (2022) A fair consensus model in blockchain based on computational reputation. Expert Syst Appl 204(117):578
5. Tanwar S, Parekh K, Evans R (2020) Blockchain-based electronic healthcare record system for healthcare 4.0 applications. J Inf Secur Appl 50:102407
6. Bhaskar P, Tiwari CK, Joshi A (2021) Blockchain in education management: Present and future applications. Interact Technol Smart Educ 18(1):1–17
7. Lohmer J, Bugert N, Lasch R (2020) Analysis of resilience strategies and ripple effect in blockchain-coordinated supply chains: An agent-based simulation study. Int J Prod Econ 228(107):882
8. Novo O (2018) Blockchain meets IoT: An architecture for scalable access management in IoT. IEEE Internet Things J 5(2):1184–1195

9. Lee Y, Lee KM, Lee SH (2020) Blockchain-based reputation management for custom manufacturing service in the peer-to-peer networking environment. Peer-to-Peer Netw Appl 13:671–683
10. Hu Q, Cheng H, Zhang X et al (2022) Trusted resource allocation based on proof-of-reputation consensus mechanism for edge computing. Peer-to-Peer Netw Appl 1–17
11. Thakkar P, Nathan S, Viswanathan B (2018) Performance benchmarking and optimizing hyperledger fabric blockchain platform. In: 2018 IEEE 26th International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems, IEEE, 264–276
12. Saleh F (2021) Blockchain without waste: Proof-of-stake. Rev Financ Stud 34(3):1156–1190
13. Castro M, Liskov B et al (1999) Practical Byzantine fault tolerance. In: OsDI, 173–186
14. Ongaro D, Ousterhout J (2014) In search of an understandable consensus algorithm. In: 2014 {*USENIX*} Annual Technical Conference ({*USENIX*}{*ATC*} 14), 305–319
15. Wang Y, Zhong M, Cheng T (2022) Research on PBFT consensus algorithm for grouping based on feature trust. Sci Rep 12(1):12515
16. Xiao Y, Zhang N, Lou W et al (2020) A survey of distributed consensus protocols for blockchain networks. IEEE Commun Surv Tutor 22(2):1432–1465
17. Wang T, Hua H, Wei Z et al (2022) Challenges of blockchain in new generation energy systems and future outlooks. Int J Electr Power Energy Syst 135(107):499
18. Khan D, Jung LT, Hashmani MA (2021) Systematic literature review of challenges in blockchain scalability. Appl Sci 11(20):9372
19. Aublin PL, Mokhtar SB, Quéma V (2013) Rbft: Redundant Byzantine fault tolerance. In: 2013 IEEE 33rd International Conference on Distributed Computing Systems, IEEE, 297–306
20. Huang J, Kong L, Chen G et al (2019) Towards secure industrial IoT: Blockchain system with credit-based consensus mechanism. IEEE Trans Industr Inf 15(6):3680–3689
21. Wang EK, Sun R, Chen CM et al (2020) Proof of X-repute blockchain consensus protocol for IoT systems. Comput Secur 95(101):871
22. Bidgoly AJ (2021) Probabilistic analysis of trust based decision making in hostile environments. Knowl-Based Syst 211(106):521
23. Gai F, Wang B, Deng W et al (2018) Proof of reputation: A reputation-based consensus protocol for peer-to-peer network. In: Database Systems for Advanced Applications: 23rd International Conference, DASFAA 2018, Gold Coast, QLD, Australia, May 21-24, 2018, Proceedings, Part II 23, Springer, 666–681
24. Tang C, Wu L, Wen G et al (2019) Incentivizing honest mining in blockchain networks: A reputation approach. IEEE Trans Circuits Syst II Express Briefs 67(1):117–121
25. Wu H, Liu H, Li J (2023) FabricETP: A high-throughput blockchain optimization solution for resolving concurrent conflicting transactions. Peer-to-Peer Netw Appl 1–18
26. Wang EK, Liang Z, Chen CM et al (2020) PoRX: A reputation incentive scheme for blockchain consensus of IIoT. Futur Gener Comput Syst 102:140–151
27. Gao S, Yu T, Zhu J et al (2019) T-PBFT: An eigentrust-based practical Byzantine fault tolerance consensus algorithm. China Commun 16(12):111–123
28. Tong W, Dong X, Zheng J (2019) Trust-pbft: A peertrust-based practical byzantine consensus algorithm. In: 2019 International Conference on Networking and Network Applications, IEEE, 344–349
29. Fortino G, Messina F, Rosaci D et al (2019) Using blockchain in a reputation-based model for grouping agents in the Internet of Things. IEEE Trans Eng Manage 67(4):1231–1243
30. Fortino G, Fotia L, Messina F et al (2021) A blockchain-based group formation strategy for optimizing the social reputation capital of an iot scenario. Simul Model Pract Theory 108(102):261
31. Crain T, Gramoli V, Larrea M et al (2018) Dbft: Efficient leaderless byzantine consensus and its application to blockchains. In: 2018 IEEE 17th International Symposium on Network Computing and Applications, IEEE, 1–8
32. Wang X, Guan Y (2022) A hierarchy Byzantine fault tolerance consensus protocol based on node reputation. Sensors 22(15):5887
33. Li W, Feng C, Zhang L et al (2020) A scalable multi-layer PBFT consensus for blockchain. IEEE Trans Parallel Distrib Syst 32(5):1146–1160
34. Wang Y, Zhong M, Cheng T (2022) Research on PBFT consensus algorithm for grouping based on feature trust. Sci Rep 12(1):12515
35. Khujamatov K, Ahmad K, Reypnazarov E et al (2020) Markov chain based modeling bandwith states of the wireless sensor networks of monitoring system. Int J Adv Sci Technol 29(4):4889–4903
36. Neal RM (2000) Markov chain sampling methods for Dirichlet process mixture models. J Comput Graph Stat 9(2):249–265
37. Liu P, Zhang X, Pedrycz W (2021) A consensus model for hesitant fuzzy linguistic group decision-making in the framework of Dempster-Shafer evidence theory. Knowl-Based Syst 212(106):559
38. Venkanna U, Leela Velusamy R (2016) TEA-CBRP: Distributed cluster head election in MANET by using AHP. Peer-to-Peer Netw Appl 9:159–170
39. Abdi H (2010) Coefficient of variation. Encyclopedia of Research Design 1(5)
40. Block RA, Zakay D (1997) Prospective and retrospective duration judgments: A meta-analytic review. Psychon Bull Rev 4(2):184–197
41. Li Y, Qiao L, Lv Z (2021) An optimized Byzantine fault tolerance algorithm for consortium blockchain. Peer-to-Peer Netw Appl 14:2826–2839

Yangyang Jiang is now working toward the M.S. degree in communication and information system with the school of Communication and Information Engineering, Shanghai University, Shanghai, China. His research interests include Blockchain Technology and Privacy Protection.

Yepeng Guan is currently a full professor at the College of Communication and Information Engineering in Shanghai University, China. He received the B.S. and M.S. degrees in physical geography from the Central South University, Changsha, China, in 1990, 1996, respectively, and the Ph.D. degree in geodetection and information technology from the Central South University, Changsha, China, in 2000. His research interests include Machine Learning, Cloud Computing and Blockchain.