Abstract—Though voting-based consensus algorithms in blockchain outperform proof-based ones in energy- and transaction-efficiency, they are prone to incur wrong elections and bribery elections. The former originates from the uncertainties of candidates’ capability and availability, and the latter comes from the egoism of voters and candidates. Hence, in this paper, we propose an uncertainty- and collusion-proof voting consensus mechanism, including the selection pressure-based voting algorithm and the trustworthiness evaluation algorithm. The first algorithm can decrease the side effects of candidates’ uncertainties, lowering wrong elections while trading off the balance between efficiency and fairness in voting miners. The second algorithm adopts an incentive-compatible scoring rule to evaluate the trustworthiness of voting, motivating voters to report true beliefs on candidates by making egoism consistent with altruism so as to avoid bribery elections. A salient feature of our work is theoretically analyzing the proposed voting consensus mechanism by the large deviation theory. Our analysis provides not only the voting failure rate of a candidate but also its decay speed. The voting failure rate measures the incompetence of any candidate from a personal perspective by voting, based on which the concepts of the effective selection value and the effective expectation of merit are introduced to help the system designer determine the optimal voting standard and guide a candidate to behave in an optimal way for lowering the voting failure rate.

Index Terms—Blockchain, consensus mechanism, large deviation theory.

I. INTRODUCTION

FEATURED with decentralization, transparency, immutability, and global inclusiveness, blockchain [1], [2] is giving impetus to extensive novel applications and fields, pulling us from the Internet of information to Internet of value. So far, the market size of the blockchain is more than USD 4.9 billion, which is estimated to reach USD 67.4 billion by 2026, at a compound annual growth rate of 68.4% during 2021-2026 [3]. Essentially, blockchain is an event-driven deterministic state machine, which calls for consensus algorithms to agree on the order of deterministic events and screen out invalid events. The mainstream consensus algorithms in blockchain are Proof of Work (PoW) [4] and Proof of Stake (PoS) [5].

In PoW, blockchain participators (miners) solve cryptographic puzzles in the form of a hash computation by brute-forcing, thus competing the accounting right and gaining rewards. Obviously, this leads to a serious low-efficiency issue due to the immense scale of electricity wasting and long transaction confirmation delay. Yet, such a way of sacrificing efficiency does not bring true fairness even though fairness is the original design intention of PoW. This is because mining one block is so hard that many individual miners tend to gather their hashing powers to form mining pools for seeking the solution of PoW puzzles together. The accounting rights and rewards are gradually concentrated in several super mining pools, leading a solo miner hardly to have the opportunity to participate in the decision-making of blockchain. This unfairness makes complete decentralization cannot be realized through PoW.

As a power-saving alternative to PoW, PoS relies on stake instead of mining power to decide who will get the chance to mine, effectively addressing the problem of electricity wasting. The stake is reliable as the proof because any participant who owns more stake is more willing to maintain the security of blockchain. Also, PoS outperforms PoW in resisting 51% attack as participants with more stakes lose more when attacked. However, PoS still has a limitation in improving efficiency since it does not significantly lower the transaction confirmation delay. Not only that, PoS leads to more unfairness in that the rich miners are bound to be dominant in the blockchain network.

Different from PoW and PoS where all nodes participate in the decision-making of blockchain, Delegated Proof of Stake (DPoS) [6] leverages the voting power of stakeholders (voters) to select qualified nodes to mine, who will receive rewards

1When blockchain is applied in the financial sector, the transaction is a typical event.

2Everyone has an equal opportunity to compete for the accounting right.
when creating correct and timely blocks. Since no competition among miners in mathematical computations, DPoS outperforms PoW and PoS in energy- and transaction-efficiency. However, DPoS suffers from two issues: 1) wrong election. Due to the information asymmetry, a voter is hardly fully aware of a candidate’s behavior. When the candidate mines precariously or behaves maliciously, he will be voted out at the end of one round. However, such an ex-post countermeasure cannot compensate for the loss of the voters caused by their wrong selection in this round; 2) bribery election. Though the voting mechanism seems democratic, it can easily ruin justice in practice since voters are also stakeholders, which makes room for the voters and candidates to collude, leading to bribery elections. For instance, EOS, one of the popular DPoS-based blockchains, discovered collusion is the most likely potential risk [7, 8]. Even though some block producers (e.g., Huobi, Starteos) have been accused of vote buying and collusion with others, they denied and were still voted as block producers, indicating the bribery election is still an ongoing issue to be addressed [9]. Not only in EOS, many other application scenarios of DPoS have also encountered this kind of problem [10].

In this paper, we propose an uncertainty- and collusion-proof voting consensus mechanism, which inherits the advantages of DPoS in terms of energy- and transaction-efficiency, but has no trouble in the wrong and bribery elections. It is challenging to realize the proposed voting consensus mechanism in that 1) the uncertainty leading to wrong elections is dual: one from unknown candidates and the other from familiar ones. In detail, lacking knowledge on the mining ability of unknown candidates, a voter may miss professional miners or mistakenly vote incompetent candidates as qualified ones. Even, the voter may still make wrong decisions when facing candidates who had been selected before due to their uncertain behaviors, such as becoming unavailable because of electricity shortage or malicious attacks; 2) the real intention of voting (egotism or altruism) is the private information of a voter, which makes it easy to hide collusion between voters and candidates, implying that bribery elections are prone to take place.

To tackle the first challenge, we propose a selection pressure-based voting algorithm. In ecology, the selection pressure is used to characterize the advantage that a creature’s traits are chosen to survive. We introduce this concept in the proposed voting consensus algorithm to depict the urgency of a candidate being chosen. The selection pressure is designed to vote miners with high availability and low frequency of being chosen before. Such a design has two merits: 1) not only can it reduce side effects from uncertain behaviors of known candidates but also explore unfamiliar or even completely unknown candidates, decreasing wrong elections due to low cognitive level of global information; 2) it can trade off the balance between efficiency and fairness in electing miners. Our algorithm provides a way to vote for miners with high historical contribution and availability effectively while considering the candidates’ historical frequency of being chosen before in order to give consideration to fairness.

To address the second challenge, we adopt a scoring rule for evaluating the trustworthiness of voting. Such a rule is subject to the incentive compatibility (IC) constraint so that a rational and intelligent voter finds she can obtain a higher score when telling the truth rather than lying. Thus, the motivation of collusion between the voter and the candidate is naturally eliminated, implying that bribery elections can be prevented. An attractive nature of the trustworthiness evaluation algorithm is that there is no need to submit any private information. Instead, a voter only needs to report her opinion of any other random peer voter’s prior and posterior beliefs on candidates. The reason behind such a design is each voter’s subjective belief comes from the same blockchain system, which affects every voter in the same way. Hence, in the opinion of any voter, her subjective belief is symmetric with regard to that of any other voter.

A salient feature of our work is theoretically analyzing the proposed voting consensus mechanism by taking advantage of the large deviation theory. The main contributions and results of this part are summarized as follows:

- We provide not only the voting failure rate of a candidate but also its decay speed. Such a fine-grained analysis reveals that to slow down the growth of the voting failure rate, lowering the selection criteria is more effective than increasing the selection pressure of a candidate.
- A concept of the effective selection valve is introduced to describe the lowest selection criteria that can guarantee the voting failure rate of a candidate no more than a given tolerance degree. Through solving the effective selection valve, the optimal number of super nodes (i.e., the winners in the voting) can be determined for the system designer to meet the requirement of the voting failure rate.
- A concept of the effective expectation of merit is introduced as the minimum expected merit a candidate brought to the voter, which makes his voting failure rate no more than a given tolerance degree. The value of this metric quantitatively answers a question: how much effort should a candidate devote to become a super node?

The remainder of the paper proceeds as follows: In Section II, we summarize the related works for consensus mechanism in blockchain. Then we present an overview of our proposed voting consensus mechanism in Section III. The main part of the mechanism, named selection pressure-based voting consensus algorithm, is elaborated in Section IV, which is further enhanced with the voting trustworthiness evaluation in Section V. We conduct an experimental evaluation in Section VI to illustrate the effectiveness of our mechanism. The whole paper is concluded in Section VII.

II. RELATED WORK

Generally speaking, we classified the consensus algorithms into two categories: the miner-based and the ledger-based. In the former, the reliability of ledger (block) mainly depends on the credibility of the miner, while in the later, it mainly

3In this paper, we use “he” and “she” to differentiate the candidate and the voter.
depends on the consistency of ledger veriﬁed by most blockchain participants.

Speciﬁcally, the miner-based algorithms select credible miners through different forms of competition, according to which, it can be further divided as: the proof-based, the voting-based and the probability-based.

In the proof-based consensus algorithms, only when the nodes prove to be credible, can they gain the accounting right. PoW [4] and PoS [5] are two classic proof-based consensus algorithms, which uses computational power and stakes respectively. Taking advantage of the traits of PoW and PoS, researchers proposed several variants, such as the Proof of Activity (PoA) [11], the Proof of Capacity (PoC) [12], and the Proof of Elapsed Time (PoET) [13]. In PoA, a mined block needs to be signed by \( N \) miners to become valid. Thus, the difﬁculty of mining a block depends on the fraction of online stakeholders instead of the number of coins in PoS. Besides, stakes can also be replaced by capacity in PoC, where miners allocate large capacity in the local hard drive for mining, indicating their interest in mining the block. In PoET, the leader is elected through a lottery-based election model, which is required to be run in a trusted execution environment (TEE). In detail, each validator takes advantages of TEE to generate a waiting time randomly, where the one with the shortest waiting time becomes the leader, ensuring that the leader is fairly selected.

The voting-based consensus algorithms determine who has the accounting right through voting. DPoS [6] is a representative one, where stakeholders (voters) vote to select delegates (super nodes) for block generation. DPoS has many variants. For instance, Roll-DPoS [14] brings in the randomness of block producers from a voted candidate pool for high throughput. Delegated Proof-of-Private-Stake (DPoPS) [15] deals with the privacy problem of stake and the randomness of the next block producer. Delegated Proof-of-Stake with downgrade (DDPoS) [16] quickly replaces malicious delegates for security improvement.

However, the voting-based algorithms have the issues of bribery election and wrong election [17], [18], which motivate some solutions [10], [19], [20]. For instance, Kang et al. [10] replaced the stake with reputation as the voting basis and evaluated the reputation of miner candidates based on the voter’s local opinion and the recommend opinions from other peer voters. DT-DPoS (DPoS with dynamic trust) [20] selects the witness nodes for block generation based on the stake-based vote and a trust value, which is simply determined by the transaction history between any two nodes in the past. Su et al. [19] proposed a credit-based DPoS consensus protocol, which uses a recorded credit value of the node as the stake and permits voting both for the trusted node and against the untrusted node. However, these mechanisms have a premise that the voters are well-behaved and reliable, fundamentally different from our scheme.

Unlike the above two subtypes, in the random probability-based consensus algorithms, the proportion of a node’s resources in the whole network is taken as the probability that it is selected as the block producer. In Algorand [21], nodes are randomly selected to generate new blocks by veriﬁable random function (VRF) and whether the blocks are ﬁnally conﬁrmed depends on the veriﬁcation process based on designed BA* algorithm. Ouroboros [22] focuses on selecting the nodes for block generation randomly by the seed \( P \) of the sampling function (e.g., follow-the-Satoshi), which is stored in advance in the ﬁrst block of each epoch (called genesis) with all potential stakeholders (candidates). The probability of receiving the right is based on the proportion of each candidate’s stake. Ouroboros and its improvements (e.g. Ouroboros Praos [23] and Ouroboros Genesis [24]) use the proportion of each candidate’s stake as the probability of receiving the right, also importing the randomness of the leader election process, which makes the leaders unpredictable in advance. Some improvements followed this idea subsequently. However, our mechanism proposes a reasonable scheme to encourage all stakeholders (voters) to choose qualiﬁed candidates honestly for the aim of democracy, rather than selecting randomly.

In the ledger-based consensus algorithms, all the nodes in the network should verify the transactions or the blocks together. These algorithms focus on preventing the faults caused by some crashed or malicious nodes in the network that would damage the consistency of ledgers. Based on different assumptions, it can be classiﬁed into three types: partial synchrony-based, asynchrony-based, and synchrony-based. Most ledger-based consensus algorithms [25], [26], [27], [28], [29], [30] have a partial synchronization assumption on network communication. It supposes that there is a special event called Global Stabilization Time (GST), which eventually happens after some unknown ﬁnite time. The system behaves asynchronously till GST and synchronously after GST. The asynchrony-based consensus algorithms [31], [32] assume that there is no bound on the time to deliver a message, but each message will eventually be delivered. These algorithms often introduce a random number generator to prevent adversaries from predicting. The synchrony-based consensus algorithms [33], [34], [35] have a strong assumption on network communication. That is, all messages are delivered within a given time duration. These algorithms often utilize synchronization assumptions to improve the BFT threshold or to design speciﬁc protocols. In summary, most classical ledger-based consensus algorithms often adopt the all-to-all broadcast protocol and have a high communication complexity, which limits the number of participants. Recent studies (e.g., HotStuff [27], FastBFT [30]) have reduced communication complexity signiﬁcantly by changing the way of exchanging messages. However, they have to limit the number of malicious nodes due to the trust assumption of Byzantine fault tolerance (BFT) and the need to avoid Sybil attacks. Therefore, the nodes should be known and adjustable, which limits the freedom of nodes to join and withdraw from blockchain system. Thus, such algorithms are usually restrictedly applied in permissioned blockchain.

III. OVERVIEW OF OUR VOTING CONSENSUS MECHANISM

In this section, an overview of our proposed uncertainty- and collusion-proof voting consensus mechanism is introduced.
As shown in Fig. 1, the consensus process includes two main steps, the voting process that our paper is trying to improve and the block generation and synchronization process that follows the original DPoS.

A. Consensus Process

The first step is the focus of our research, where \( N \) voters (i.e., all the available nodes in the whole network) are required to select \( K \) super nodes from \( M \) candidates as representatives to mine. Candidates are potential super nodes who are expected to generate blocks once being voted successfully. Since a candidate obtains the accounting right through the voting rather than the competition with others in terms of computing power, the proposed mechanism inherits the advantage of DPoS in saving energy and speeding up transactions. However, such a voting-based consensus mechanism may suffer the issues of wrong and bribery elections as mentioned in Section I, pressing a need to find solutions.

To avoid bribery and wrong elections, we employ score to evaluate the capability of any candidate to be a super node. K nodes with the highest scores are selected as super nodes. The score \( S^k_j \) of any candidate \( j \) in the \( k^{th} \) round of voting can be calculated as

\[
S^k_j = \sum_{i=1}^{N} s_i \times t^k_{ij} \times c^k_{ij}, \quad j = 1, 2, \ldots, M. \tag{1}
\]

In (1), \( c^k_{ij} \in \{0, 1\} \) is an indicator identifying whether voter \( i \) chooses candidate \( j \) as a super node in round \( k \), satisfying \( \sum_{j=1}^{M} c^k_{ij} = K \), which is a key parameter needed to be controlled strategically to refrain from wrong selection. \( t^k_{ij} \) represents the trustworthiness of votes that voter \( i \) casts to \( j \) in round \( k \), reflecting the degree that voter \( i \) chooses candidate \( j \) in light of his block mining capability rather than other factors, which is designed to avoid bribery election. The larger the value of \( t^k_{ij} \), the more credible the vote that \( i \) votes \( j \). Finally, \( s_i \) is the stake of voter \( i \), indicating her voting weight or speaking power in the voting process.

In the proposed uncertainty- and collusion-proof voting consensus mechanism, as shown in Fig. 2, to reduce wrong selections, we propose the selection pressure-based voting algorithm to determine the optimal \( c^k_{ij} \) (\( i = 1, 2, \ldots, N; \ j = 1, 2, \ldots, M \)), combined with which, the voting trustworthiness \( t^k_{ij} \) can be calculated through the evaluation algorithm to hold back bribery elections. We will detail the above two algorithms in the following sections.

The motivation behind the definition of \( S^k_j \) is to make the selection of voters more reliable. We respectively constrain the voting trustworthiness from direct and indirect evaluations. Direct trustworthiness refers to our proposed specific evaluation algorithm, which assesses whether voter votes in accord with her true belief. And the indirect trustworthiness is related to the number of stakes a voter owns, because it reflects her concern about the stability of blockchain to some extent. The more stakes \( s_i \) she has, the more she hopes the system to be steady in order to prevent the loss caused by the instability of system. Thus, \( s_i \) can be seen as an indirect trustworthiness evaluation.

The second step of consensus mechanism (i.e., the block generation and synchronization process) can follow the original DPoS [36]. After being selected, the top \( K \) super nodes will generate blocks in turn according to a randomly predetermined sequence. The sequence is randomly determined according to specific rules. If any of super nodes fails, it will be replaced by the one ranking after \( K \) at the end of a round. A round represents the time during which all super nodes generate blocks in turn. The rules for determining random sequence and the replacement of failed or malicious nodes can follow other variants of DPoS [14], [15], [16] for security improvement. Once a block is generated, it will be broadcasted to all the other super nodes. If more than two-thirds of the super nodes verify the block successfully, the block will be appended to the blockchain.

In addition, voting process is not conducted every time after the generation of \( K \) blocks for efficiency. A new voting process may be invoked after several rounds, which is determined according to different system conditions.

B. Threat Model

Our purpose is to motivate rational voters to make the right choice when voting. In other words, voters will not make wrong or undemocratic voting because of their insufficient knowledge about candidates or bribery driven by economic gains. In our voting process, the main participants include two roles: voters and candidates. We assume that both roles are rational, implying they are profit-driven when making decisions. This is obviously a practical assumption, based on which, we define a threat model that illustrates potential threats and malicious behaviors as follows:

**Intended bribery attacks from candidates.** The candidates can bribe voters to get more votes for being elected as super nodes. This collusion is beneficial for both candidates and voters. They can share rewards and give priority to their preferred transactions, thus ruining fairness. So rational profit-driven voters will choose to cooperate with these candidates.

**Unintended wrong election of voters.** Voters have a lack of perception and responsiveness to potential misbehaving candidates, because they lack information from unknown candidates.
and cannot predict the uncertain behavior of familiar ones. On one hand, a shortage of historical information will damage the interests of honest voters when they blindly vote to an unfamiliar or even completely unknown candidate. On the other hand, the uncertain behaviors of a familiar candidate, like becoming unavailable due to electricity shortage or being malicious attacked, also bring side effects on the final voting results and block generation.

IV. SELECTION PRESSURE-BASED VOTING ALGORITHM AND ANALYSIS

In this section, we first introduce the proposed selection pressure-based voting algorithm, which helps voters to vote the top \( K \) candidates. Then we take advantage of the large deviation theory [37] to deduce the optimal \( K \) when the voting failure rate threshold is given.

A. Selection Pressure-Based Voting Algorithm

To guarantee efficiency as well as fairness, the selection pressure \( F_{ij}^t \) of each candidate \( j \) (\( j = 1, 2, \cdots, M \)) in the \( t^{th} \) round of voting is formulated as follows:

\[
F_{ij}^t = M_{ij}^{t-1} - C_{ij}^{t-1}, \quad t = 2, 3, \ldots
\]  

(2)

In (2), \( M_{ij}^{t-1} = m_{ij}^1 + m_{ij}^2 + \cdots + m_{ij}^{t-1} \) and \( C_{ij}^{t-1} = c_{ij}^1 + c_{ij}^2 + \cdots + c_{ij}^{t-1} \), where \( m_{ij}^t \geq 0 \) is the merit of candidate \( j \) to voter \( i \) in round \( k \) (\( k = 1, 2, \cdots, t-1 \)) and \( c_{ij}^t \in \{0, 1\} \) is whether candidate \( j \) is selected by voter \( i \) in round \( k \) as mentioned above.

Based on our proposed consensus algorithm, voter \( i \) will vote candidates by setting \( c_{ij}^1 = 1 \) where \( j \) is one of the top \( K \) candidates with the highest \( F_{ij}^t \). It is a trade-off of the balance between efficiency and fairness in voting super nodes. In detail, the bigger \( F_{ij}^t \) implies the higher \( M_{ij}^{t-1} \) and the lower \( C_{ij}^{t-1} \). The greater the merit \( M_{ij}^{t-1} \) of candidate \( j \), the more the contribution he had made to voter \( i \) until round \( t-1 \). Selecting candidates with the higher \( M_{ij}^{t-1} \) can improve the profit of voter \( i \) in the future, thus ensuring the efficiency. In addition, the lower \( C_{ij}^{t-1} \) means the fewer historical selections are made on candidate \( j \). Voting candidates with the lower \( C_{ij}^{t-1} \) can achieve the fairness in the voting process. That is, a candidate who was selected less before (lower \( C_{ij}^{t-1} \)) often has a higher chance to be voted this round. This helps voters explore unfamiliar or completely unknown candidates instead of always choosing those candidates they are familiar with. Thus, through trying to have a more comprehensive understanding of the overall information about candidates, voters can make rational and intelligent elections.

In the following, we will detail how to calculate \( m_{ij}^k \) (\( i = 1, 2, \cdots, N; \quad j = 1, 2, \cdots, M; \quad k = 1, 2, \cdots \)) in (2). Since \( m_{ij}^k \) means the merit of candidate \( j \) to voter \( i \) in round \( k \), it is affected by two factors: the availability of candidate \( j \) and the profit he brought to voter \( i \) in round \( k \), indicated by \( R_{ij}^k \). Obviously, a candidate with high historical contribution and low unavailability is more valuable to a voter. Hence, \( m_{ij}^k \) can be calculated as

\[
m_{ij}^k = \rho d(u_j) + R_{ij}^k.
\]  

(3)

In (3), \( \rho \) is a scaling parameter. \( d(u_j) \) is a function indicating the availability of candidate \( j \), where \( u_j \in [0, 1] \) is the probability that the candidate \( j \) is unavailable at any given round, called the unavailable probability. The larger \( u_j \), the smaller \( d(u_j) \). In practice, \( u_j \) can be estimated by the total number of unavailable times divided by the total number of times that candidate \( j \) was voted. Obviously, the method using the availability of a candidate as an important metric for evaluating his merit can effectively reduce the negative impact of the candidate’s uncertain behaviors on voting, lowering the probability of the wrong election.

In our mechanism, each voter \( i \) calculates the accumulative merit \( M_{ij}^{t-1} \) of any candidate \( j \) and the times \( C_{ij}^{t-1} \) that voter \( i \) votes candidate \( j \) within the last \( t-1 \) rounds. According to the selection pressure-based voting algorithm, the voting choice \( c_{ij}^t \) to any candidate \( j \) in the current round \( k \) is determined by each voter \( i \). These parameters are calculated and updated locally by each voter \( i \) to guide the voting.

B. Analysis Based on the Large Deviation Theory

In this subsection, we take advantage of the large deviation theory to analyze the performance of the proposed voting algorithm. To that aim, we create a virtual queue \( Q_{ij}^t \) (\( i = 1, 2, \cdots, N; \quad j = 1, 2, \cdots, M \)) as follows:

\[
Q_{ij}^t = C_{ij}^{t-1} - M_{ij}^{t-1}, \quad t = 2, 3, \cdots
\]  

(4)

It is easy to find that \( Q_{ij}^t = -F_{ij}^t \), implying that the larger \( F_{ij}^t \), the smaller \( Q_{ij}^t \). Since the proposed voting consensus algorithm votes the top \( K \) nodes with the biggest \( F_{ij}^t \), \( Q_{ij}^t \) actually characterizes the ranking of candidate \( j \) for being voted. A candidate with a large \( F_{ij}^t \) has a high urgency to be voted, thus he ranks high, implying he has short virtual queue length. Hence, we name the virtual queue \( Q_{ij}^t \) the ranking queue. A node with short length of ranking queue has high chance to be voted.

Whether a candidate \( j \) being voted by voter \( i \) in each round is denoted by \( c_{ij}^k \). If we use \( p \) to represent the probability of \( c_{ij}^k = 1 \), then the probability of \( c_{ij}^k = 0 \) is \( 1-p \). So \( c_{ij}^k \) subjects to a Bernoulli distribution. According to \( C_{ij}^{t-1} = c_{ij}^1 + c_{ij}^2 + \cdots + c_{ij}^{t-1} \), \( C_{ij}^{t-1} \) can be assumed as a Poisson random variable when \( t \) is large enough and \( p \) is small. Similarly, the merit of candidate \( j \) to voter \( i \) in each round depends on whether he is voted, which is an event obeying the Bernoulli distribution as just mentioned. Meanwhile, the voter’s decisions in successive rounds are independent, because his choice changes every time after receiving the result of the last round. Thus, \( M_{ij}^{t-1} \) can also be viewed as a Poisson random variable.

We assume that \( \Lambda \) and \( \Lambda \) are respectively the expectations of \( M_{ij}^{t-1} \) and \( C_{ij}^{t-1} \). To guarantee the stability of the virtual queue \( Q_{ij}^t \), \( \Lambda > \Lambda \) should hold.

For any candidate, his main concern is under what conditions will he fail in the election. This is also the key to
analyzing a voting algorithm. To that aim, we study the metric $L_{ij} = \sup_{t \geq 2} Q^t_{ij}$, which is the longest ranking queue length of candidate $j$ when $t \to \infty$, reflecting his historic lowest selection pressure. Considering each voter can select up to $K$ super nodes in our proposed voting consensus algorithm, we have the following definition:

**Definition 4.1 (Selection Valve):** The selection valve $L$ is the ranking queue length of a candidate whose selection pressure is ranked $K^{th}$.

Obviously, when $L_{ij} > L$, candidate $j$ may fail in the election. Otherwise, he will be voted by voter $i$. To analyze such an event, we define the following concept:

**Definition 4.2 (Voting Failure Rate):** The voting failure rate indicates the probability that any candidate $j$ is not voted by voter $i$ when his historical longest ranking queue $L_{ij}$ is longer than the selection valve $L$, i.e., $P(L_{ij} > L)$.

In the following, we will analyze the distribution of the event $L_{ij} > L$. Let $L = lb$ for $b > 0$. In light of Cramer’s theorem [37], which is an example of a large deviations principle (LDP) and is used to analyze the distribution of the event, when $t \to \infty$, we have

$$
\lim_{t \to \infty} \frac{1}{t} \log P(L_{ij} > lb) = -I(b),
$$

with

$$
I(b) = \inf_{i \geq 2} t \psi^*(\frac{b}{t}),
$$

where

$$
\psi^*(x) = \sup_{\theta \in \mathbb{R}} \{ \theta x - \psi(\theta) \}.
$$

In (7), $\psi^*(x)$ is the convex conjugate or the Legendre transformation of $\psi(\theta)$ which is the cumulant generating function of $Q^t_{ij}$ and can be expressed as

$$
\psi(\theta) = \lim_{t \to \infty} \frac{1}{t} \log \mathbb{E}[e^{\theta Q^t_{ij}}] = \lim_{t \to \infty} \frac{1}{t} \log \mathbb{E}[e^{\theta(C^t_{ij} - M^1_{ij})}] = \lim_{t \to \infty} \frac{1}{t} \log \mathbb{E}[e^{\theta C^t_{ij} - \log(1 - e^{-\theta M^1_{ij}})}] = \psi_{C^t_{ij}}(\theta) + \psi_{M^1_{ij}}^*(-\theta).
$$

In the above equation, $\psi_{C^t_{ij}}(\cdot)$ and $\psi_{M^1_{ij}}^*(\cdot)$ are respectively the cumulant generating functions of $C^t_{ij}$ and $M^1_{ij}$. Because the cumulant generating function for any Poisson process $X$ can be calculated as $\log \mathbb{E}[e^{\theta X}] = \mathbb{E}X(e^\theta - 1)$, we have

$$
\psi_{C^t_{ij}}(\theta) = \Lambda(e^\theta - 1),
$$

and

$$
\psi_{M^1_{ij}}^*(-\theta) = \lambda(e^{-\theta} - 1).
$$

Thus, (7) can be transferred to

$$
\psi^*(x) = \sup_{\theta \in \mathbb{R}} \{ \theta x - \Lambda(e^\theta - 1) - \lambda(e^{-\theta} - 1) \}.
$$

In the following, we will discuss how to solve (10) so as to obtain $I(b)$ in (5). In fact, we only need to consider the case of $x > 0$ in (10) because of $\frac{d}{dx} \psi(\theta) = \lambda e^{-\theta} - \lambda e^{\theta} < 0$, $\theta^*$ that maximizes $\psi(\theta)$ can be calculated with the condition $\frac{\partial \psi(\theta)}{\partial \theta} = 0$. So,

$$
\theta^* = \log \frac{x + \sqrt{x^2 + 4\lambda \Lambda}}{2\Lambda}.
$$

Hence, we have

$$
\psi^*(x) = x \log \frac{x + \sqrt{x^2 + 4\lambda \Lambda}}{2\Lambda} - \frac{2\lambda}{x + \sqrt{x^2 + 4\lambda \Lambda}} + \lambda + \Lambda.
$$

Let $x = \frac{b}{t}$ in (12), we can obtain $\inf_{t \geq 2} t \psi^*(\frac{b}{t})$ at $t = b/(\lambda - \lambda)$.

Thus, we have

$$
I(b) = b \log \frac{\lambda}{\Lambda},
$$

(5) shows that $P(L_{ij} > L)$ slumps exponentially with the increase of $I(b)$ given selection valve $L$, so $I(b)$ is the rate function to indicate the decay speed of the voting failure rate. The larger $I(b)$ is, the slower the growth rate of the voting failure rate. Otherwise, the growth rate of the voting failure rate will increase, suggesting that candidate $j$ should improve his availability as much as possible to increase his selection pressure so as to shorten $L_{ij}$ for decreasing $P(L_{ij} > L)$.

Fig. 3 illustrates the impacts of $b$ and $\lambda - \Lambda$ on $I(b)$. The decay rate of the voting failure probability $I(b)$ increases when either $b$ or $\lambda - \Lambda$ increases. Because $L = lb$ is positively correlated with $b$ and $\lambda - \Lambda$ is the expectation of $F^t_{ij}$, we can draw the first conclusion: no matter lifting the selection valve $L$ or enhancing the selection pressure $F^t_{ij}$ of a candidate $j$, $I(b)$ goes up and accordingly his voting failure rate will decrease. Furthermore, it can be found that the decay rate $I(b)$ increases linearly with $b$ while creeping up logarithmically with $\lambda - \Lambda$. Hence, the second conclusion is that the voting failure rate is more sensitive to the selection valve rather than the selection pressure. Hence, to reduce the voting failure rate, the most effective way is to increase $L$ so as to increase the vacancies of super nodes. However, more super nodes incur more risks of their credibility. Therefore, it is worth studying how to determine a proper selection valve $L$ to trade off the balance of the voting failure rate and the risks of super nodes’ credibility.

To that aim, we introduce the following concept:

**Definition 4.3 (Effective Selection Valve):** The effective selection valve $L^*(\epsilon)$ is the minimum selection valve that makes the voting failure rate of a candidate $j$ no more than a given threshold $\epsilon \in (0, 1]$, which is called the voting failure tolerance degree. That is

$$
L^*(\epsilon) = \min \{ L : P(L_{ij} > L) \leq \epsilon \}.
$$

Facing any candidate $j$ whose expected merit to voter $i$ and expected number of times being voted by voter $i$ are respectively $\lambda$ and $\Lambda$, meaning that his expected selection pressure is $\lambda - \Lambda$, the effective selection valve of voter $i$ can be calculated by taking advantage of the following theorem:
his voting failure rate. i.e.,
cannot be changed, a direct way for candidate
value of miner) of the whole system. Thus, the deduction of the optimal
efficiency (high merit) and fairness (equal opportunity to be a
of selection valve
the calculation of selection pressure and the determination
is an empirical value obtained from the qualitative analysis
also can make sure there are enough qualified candidates
to be super nodes. The selection valve L
also can make sure there are enough qualified candidates
in our paper, we deduce the optimal value of
and changeable according to the system condition. However,
K
in the current typical DPoS-based system (e.g., EOS),
K
is determined by considering the decentral-
(21) is determined by considering the decentral-
the reason behind this fact is that the merit of a candidate
includes two parts: his availability and his profit brought to the
even if the availability may reduce, the profit that
the candidate can bring to the voter cannot be decreased due
to his fixed expectation of the number of times being voted
(i.e., \( \Lambda \)). Thus, the effective expectation of merit \( \lambda^* (\epsilon) \) will not reduce to zero but stabilize to the expectation of profit
brought to the voter.

V. OUR VOTING CONSSENSUS FRAMEWORK WITH VOTING
TRUSTWORTHINESS EVALUATION
In this paper, we employ the idea of the private-prior peer
prediction [38], [39] to evaluate the trustworthiness of votes
that voter \( i \) casts to any candidate \( j \) so as to determine \( t^i_{ij} \)
\( i = 1, 2, \ldots, N; \ j = 1, 2, \ldots, M; \ k = 1, 2, \ldots \) in (1). Our
voting consensus framework with trustworthiness evaluation is
shown in Fig. 6 and Fig. 7, which is detailed in the following.
According to the application scenarios with different network
scales and node connectivity, we provide two schemes
for the calculation of the voting-related parameters, which can be
deployed on a selected node or all the available voters in
the whole network. In detail, in a small-scale network with
high node connectivity, most of the voters in the network
can receive the information from other normally-working peer
nodes within the maximum time delay \( T_{max} \). In this case,
according to the private-prior peer prediction, each voter \( i \) will
not be forced to directly report her true beliefs on candidate
\( j \). Instead, she will be asked to provide her opinion of her
random peer \( r \)'s evaluation of \( j \), which will reflect her true
beliefs to some extent. So in the beginning of round \( k \), any
voter \( i \) \( (i = 1, 2, \ldots, N) \) is required to broadcast her prior

---

Theorem 4.1: \( L^*(\epsilon) = -\log \epsilon / \log \frac{\Lambda}{\epsilon} \), given \( \lambda > \Lambda > 0 \) and \( \epsilon \in (0, 1] \).

Proof: We have \( \log P(L_{ij} > L) ~ -I(b/\epsilon) \) when \( l \to \infty \).
In light of \( P(L_{ij} > L) \leq \epsilon \), we have \( I(b) \geq -\log \epsilon \). Combined with \( I(b) = b \log \frac{\Lambda}{\epsilon} \), we can derive the value of \( L^*(\epsilon) \).

Fig. 5 illustrates that the effective selection valve \( L^*(\epsilon) \)
decreases logarithmically with the increase of the voting failure
tolerance degree \( \epsilon \), implying that the reduced voting failure rate requirement makes voters \( i \) have more vacancies of super nodes. When \( \epsilon \) is fixed, as the expectation of the selection pressure \( \lambda - \Lambda \) rises up, \( L^*(\epsilon) \) decreases logarithmically, which also can make sure there are enough qualified candidates to be super nodes. The selection valve \( L \) is defined as the ranking queue length of a candidate whose selection pressure is ranked \( K^{th} \), where \( K \) is the number of super nodes that voters should vote. Therefore, there exists an optimal value of \( K \), namely \( K^*(\epsilon) \) corresponding to \( L^*(\epsilon) \), which can be actually used to guarantee the voting failure rate of a candidate is not higher than the voting failure tolerance degree. So \( K^*(\epsilon) \) is a valuable parameter for the voting consensus algorithm designer to guarantee the requirement of the voting failure rate of candidates.

In the current typical DPoS-based system (e.g., EOS), a fixed \( K \) (e.g., 21) is determined by considering the decentral-
ization degree and the efficiency of block generation, which is an empirical value obtained from the qualitative analysis and the historical experiment. Actually, \( K \) can be flexible and changeable according to the system condition. However, in our paper, we deduce the optimal value of \( K \) through the calculation of selection pressure and the determination of selection valve \( L \), which balance the trade-off between efficiency (high merit) and fairness (equal opportunity to be a miner) of the whole system. Thus, the deduction of the optimal value of \( K \) in our paper is a quantitative calculation for system optimization.

However, when \( K \) is fixed, i.e., the vacancies of super nodes cannot be changed, a direct way for candidate \( j \) to reduce his voting failure rate is to improve his competitiveness, i.e.,
Fig. 6. Our voting consensus framework with trustworthiness evaluation in small-scale network.

belief \( y^i_{r,j}(k) \in [0, 1] \) about how likely her random peer voter \( r \) \((r = 1, 2, \ldots, i - 1, i + 1, \ldots, N) \) will vote for candidate \( j \) \((j = 1, 2, \ldots, M) \). The random peer voter \( r \) can be selected by a random sampling function, e.g., follow-the-satoshi \([22] \), Verifiable Random Function(VRF) \([40] \). At the same time, voter \( i \) will also receive the prior belief \( y^i_{r,j}(k) \in [0, 1] \) from all other nodes in the blockchain system, where \( i \neq j \) is also a voter. When \( k = 1 \), voters have to exchange the estimation of prior beliefs on selections without any prior knowledge. When \( k > 1 \), the prior belief \( y^i_{r,j}(k) \) can be calculated as:

\[
y^i_{r,j}(k) = P(c^i_{r,j} = 1|c^i_{ij} = 1)P(c^k_{ij} = 1) + P(c^i_{r,j} = 1|c^i_{ij} = 0)P(c^k_{ij} = 0). \tag{14}
\]

In (14), \( P(c^i_{r,j} = 1|c^i_{ij} = 1) \) and \( P(c^k_{ij} = 0) \) are respectively the probabilities that voter \( i \) selects or doesn’t select candidate \( j \) in round \( k \), which can be obtained from her subjective prior belief on selecting this candidate. \( c^i_{r,j} \) \((c^i_{ij}) \) indicates whether voter \( i \) \((r) \) picks candidate \( j \) in the long term. Hence, \( P(c^i_{r,j} = 1|c^i_{ij} = 1) \) and \( P(c^i_{r,j} = 1|c^i_{ij} = 0) \) are the conditional probabilities that voter \( r \) selects candidate \( j \) when voter \( i \) makes the same and different decisions respectively, both of which can be calculated according to the historic results of previous \( k - 1 \) rounds. In detail, when \( k > 1 \), we have

\[
P(c^i_{r,j} = 1|c^i_{ij} = 1) = \frac{P(c^i_{r,j} = 1|c^i_{ij} = 1)}{P(c^i_{ij} = 1)} = \sum_{v=0}^{k-1} c^v_{ij} / \sum_{v=0}^{k-1} c^v_{ij},
\]

\[
P(c^i_{r,j} = 1|c^i_{ij} = 0) = 0 = \frac{P(c^i_{r,j} = 1|c^i_{ij} = 0)}{P(c^i_{ij} = 0)} = \sum_{v=0}^{k-1} c^v_{ij} / \sum_{v=0}^{k-1} (1 - c^v_{ij}).
\]

Then, each voter \( i \) updates: a) the unavailable times of each candidate \( j \) before round \( k \), so as to deduce his unavailable probability \( u_j \); b) the selection times \( C^k_{ij} \) of candidate \( j \) before \( k \), based on which the profit \( R^k_{ij} \) that candidate \( j \) devoted to voter \( i \) can be obtained; employing the above information, the selection pressure \( F^k_{ij} \) of each candidate \( j \) in round \( k \) can be calculated according to (2), so that the voting choice \( c^k_{ij} \) \((i = 1, 2, \ldots, N; \ j = 1, 2, \ldots, M) \) can be obtained through the proposed selection pressure-based voting algorithm.

After updating and calculating the above key parameters, the belief of voter \( i \) on each candidate iterates, which makes voter \( i \) have the posterior belief \( y^i_{r,j}(k) \in [0, 1] \) about the possibility that the random peer voter \( r \) \((r = 1, 2, \ldots, i - 1, i + 1, \ldots, N) \) will vote for any candidate \( j \) \((j = 1, 2, \ldots, M) \). Then, voter \( i \) will notify his choice \( c^k_{ij} \) as well as the posterior belief \( y^i_{r,j}(k) \) to all other nodes, and receive these two parameters from each of them within \( T_{max} \).

We use \( V_j \in \{h, l\} \) to indicate the mining capability of candidate \( j \). If candidate \( j \) is qualified enough to be voted as a super node, \( V_j = h \), and otherwise \( V_j = l \). Let \( e^i_j \in \{0, 1\} \) represent whether candidate \( j \) is voted as a super node in round \( k \). Since the election result and performance of super nodes in the \( k - 1 \) rounds before are recorded, \( P(V_j = h) \) can be estimated by \( \frac{\sum_{k=1}^{k-1} e^i_j}{k-1} \) before the super nodes are voted in round \( k \). Obviously, \( P(V_j = l) = \frac{\sum_{k=1}^{k-1} (1-e^i_j)}{k-1} \) in this case. Thus, the posterior belief \( y^i_{r,j}(k) \) when \( k > 1 \) can be calculated as

\[
y^i_{r,j}(k) = \begin{cases} 
P(c^i_{r,j} = 1|V_j = h)P(e^i_j = 1|c^k_{ij} = 1) + P(c^i_{r,j} = 1|V_j = l)P(e^i_j = 0|c^k_{ij} = 1), & c^k_{ij} = 1 \\
\quad + P(c^i_{r,j} = 1|V_j = h)P(e^i_j = 0|c^k_{ij} = 0), & c^k_{ij} = 0
\end{cases}
\]

where \( P(c^i_{r,j} = 1|V_j = h) = \frac{P(c^i_{r,j} = 1|V_j = h)}{P(V_j = h)} = \sum_{v=0}^{k-1} c^v_{ij} \), \( c^k_{ij} = \sum_{v=0}^{k-1} c^v_{ij} \) and \( P(c^i_{r,j} = 1|V_j = l) = \frac{P(c^i_{r,j} = 1|V_j = l)}{P(V_j = l)} = \sum_{v=0}^{k-1} c^v_{ij} / [(k-1) - \sum_{v=0}^{k-1} c^v_{ij}], P(e^i_j = 1|c^k_{ij} = 1) \) and \( P(e^i_j = 0|c^k_{ij} = 0) \) (\( P(e^i_j = 1|c^k_{ij} = 0 \) and \( P(e^i_j = 0|c^k_{ij} = 0) \)) respectively represent the probabilities that candidate \( j \) will be voted or not in round \( k \) when voter \( i \) picks (does not choose) candidate \( j \) in this round. Because in this stage, whether candidate \( j \) is finally voted as a super node is not determined yet. \( P(e^i_j = 1|c^k_{ij} = 1) \) and \( P(e^i_j = 0|c^k_{ij} = 1) \) (\( P(e^i_j = 1|c^k_{ij} = 0 \) and \( P(e^i_j = 0|c^k_{ij} = 0) \)) need to be predicted by voter \( i \), thus reflecting her real thoughts on whether candidate \( j \) is qualified enough to be voted.

\( ^3 \)When \( k = 1 \), the posterior belief \( y^i_{r,j}(1) \) is an estimated value due to no knowledge on each candidate. 

---

**References:**

[22] Follow-the-satoshi (2018). Verifiable Random Function (VRF).
[40] Structured hash functions and random oracles (2018).

---
Based on the prior and posterior beliefs, i.e., $y_{ij}^t(k)$ and $y_{ij}^{r'}(k)$, as well as the voting choice $c_{ij}^k$ received, the trustworthiness $t_{ij}^k$ of voter $i$’s choice $c_{ij}^k$ in round $k$ can be calculated according to the following scoring rule [38]:

$$
t_{ij}^k = \alpha W\left( y_{ij}^t(k), c_{ij}^k \right) + \beta. \tag{15}
$$

In (15), $\alpha \in [0, 1]$ is a scaling parameter, and $W$ can be the logarithmic form, namely,

$$
\begin{align*}
W(y, c = 1) &= \ln(y), \\
W(y, c = 0) &= \ln(1 - y), \tag{16}
\end{align*}
$$

or the quadratic form, i.e.,

$$
\begin{align*}
W(y, c = 1) &= 2y - y^2, \\
W(y, c = 0) &= 1 - y^2, \tag{17}
\end{align*}
$$

which is strictly proper and has been proved in [41]. And

$$
\beta = -\frac{1}{N} \sum_{i=1}^{N} \left[ \alpha W\left( y_{ij}^t(k), c_{ij}^k \right) + \left( 1 - \alpha \right) W\left( y_{ij}^{r'}(k), c_{ij}^k \right) \right]. \tag{18}
$$

According to (18), $\beta$ is designed in (15) to help with the measurement of how far the first two items deviate from the average value. Thus, $t_{ij}^k > 0$ represents trustworthy voting. The bigger the $t_{ij}^k$ is, the higher the credibility of the voting. On the contrary, $t_{ij}^k < 0$ reveals an unreliable voting. The smaller the $t_{ij}^k$, the worse the voting credibility.

After calculating $t_{ij}^k$ and combining with $c_{ij}^k$ obtained in the last section, as well as the stake of voter $i$, i.e., $s_i$, we can obtain the score $S_{ij}^k$ of each candidate in round $k$ according to (1). Most voters can reach a consensus on the final voting results (i.e., the top $K$ nodes with the highest scores $S_{ij}^k$) when receiving the same information. While if two-thirds of the voters cannot receive all the information from peer nodes within the given maximum time delay $T_{max}$, no consensus can be reached on the super nodes within $T_{max}$, then the voters of the whole network will rebroadcast their voting information until confirming the final results. The top $K$ nodes with the highest scores $S_{ij}^k$ can be voted as super nodes and generate blocks in round $k$ finally. The overall process is shown in Fig. 6.

In the case of a large-scale network with low node connectivity, we can depend on a randomly selected node to collect voting-related parameters, including $c_{ij}^k$, $y_{ij}^t(k)$ and $y_{ij}^{r'}(k)$ from all voters, to prevent the uncertainty of the network which may lead to the failure of timely super node determination and consensus. This node who collects these voting-related parameters can be selected by using a random sampling function [21], [22] in the voting process each time. Then it calculates the voting trustworthiness $t_{ij}^k$ of votes $c_{ij}^k$ and the scores of the candidates $S_{ij}^k$, so as to determine the winning super nodes of this round. The overall process is shown in Fig. 7.

Our algorithm can be deployed on the smart contract, and the system or all the voters in the whole network trigger the smart contract to conduct the calculation after receiving the relevant information. And these voting information can be stored in blockchain for verification.

**Theorem 5.1.** The method for evaluating trustworthiness is incentive compatible.

**Proof:** Because of the randomness of $r$, we analyze the expectation of $t_{ij}^k$, represented by $\mathbb{E}(t_{ij}^k)$. It can be calculated as

$$
\mathbb{E}(t_{ij}^k) = \mathbb{E}[\alpha W(y_{ij}^t(k), c_{ij}^k)] + \mathbb{E}[(1 - \alpha)W(y_{ij}^{r'}(k), c_{ij}^k)] + \mathbb{E}[\beta] = \alpha \left( 1 - \frac{1}{N} \right) \mathbb{E}[W(y_{ij}^t(k), c_{ij}^k)]
$$

$$
+ (1 - \alpha) \left( 1 - \frac{1}{N} \right) \mathbb{E}[W(y_{ij}^{r'}(k), c_{ij}^k)] - \frac{1}{N} \sum_{n=1, n \neq i}^{N} [\alpha W(y_{nj}^n(k), c_{nj}^k)] + (1 - \alpha) W(y_{nj}^{n'}(k), c_{nj}^k)].
$$

Employing the logarithmic form of $W$, we have

$$
\mathbb{E}(t_{ij}^k) = \alpha \left( 1 - \frac{1}{N} \right) [p_1 \ln y_{ij}^t(k) + (1 - p_1) \ln(1 - y_{ij}^t(k))] + (1 - \alpha) \left( 1 - \frac{1}{N} \right) [p_2 \ln y_{ij}^{r'}(k) + (1 - p_2) \ln(1 - y_{ij}^{r'}(k))]
$$

$$
- \frac{1}{N} \sum_{n=1, n \neq i}^{N} [\alpha W(y_{nj}^n(k), c_{nj}^k)] + (1 - \alpha) W(y_{nj}^{n'}(k), c_{nj}^k)].
$$

where $p_1 = P(c_{ij} = 1)$, $p_2 = P(c_{ij} = 1 | c_{ij} = 0)$ when $c_{ij} = 0$ or $p_2 = P(c_{ij} = 1 | c_{ij} = 1)$ when $c_{ij} = 1$.

In light of

$$
\frac{\partial \mathbb{E}(t_{ij}^k)}{\partial y_{ij}^t(k)} = \alpha \left( 1 - \frac{1}{N} \right) \frac{p_1 - y_{ij}^t(k)}{y_{ij}^t(k)(1 - y_{ij}^t(k))} = 0,
$$

$$
\frac{\partial \mathbb{E}(t_{ij}^k)}{\partial y_{ij}^{r'}(k)} = \alpha \left( 1 - \frac{1}{N} \right) \frac{p_2 - y_{ij}^{r'}(k)}{y_{ij}^{r'}(k)(1 - y_{ij}^{r'}(k))} = 0,
$$

Authorized licensed use limited to the terms of the applicable license agreement with IEEE. Restrictions apply.
we can deduce when \( y_{ij}^1(k) = p_1 \) and \( y_{ij}^1(k) = p_2 \), the above equations are satisfied.

Since
\[
\frac{\partial \mathbb{E}^2[k_{ij}]}{\partial y_{ij}^1(k)} \bigg|_{y_{ij}^1(k)=p_1} = \alpha(1 - \frac{1}{N}) \frac{y_{ij}^1(k)(y_{ij}^1(k) - 1)}{y_{ij}^2(k)(1 - y_{ij}^2(k))} < 0,
\]
\[
\frac{\partial \mathbb{E}^2[k_{ij}]}{\partial y_{ij}^1(k)} \bigg|_{y_{ij}^1(k)=p_2} = \alpha(1 - \frac{1}{N}) \frac{y_{ij}^1(k)(y_{ij}^1(k) - 1)}{y_{ij}^2(k)(1 - y_{ij}^2(k))} < 0,
\]
\( y_{ij}^1(k) = p_1 \) and \( y_{ij}^1(k) = p_2 \) can maximize \( \mathbb{E}[k_{ij}] \). Because \( p_1 \) and \( p_2 \) represent respectively the prior and posterior probabilities that an arbitrary voter supports candidate \( j \), reflecting his objective mining capacity. Hence, \( y_{ij}^1(k) = p_1 \) and \( y_{ij}^1(k) = p_2 \) indicate the subjective beliefs tally with the objective reality, implying the voter reports the truth in this case. In another word, only a voter behaves honestly, can she maximize her trustworthiness of voting, meaning the method for evaluating trustworthiness is incentive compatible. When we use the quadratic form to calculate \( W \), the same conclusion can be drawn.

VI. EXPERIMENTAL EVALUATION

In this section, we analyze the impacts of some key parameters on the reward of voters and testify the effectiveness of our proposed voting mechanism. We conduct a large number of simulations under different parameter settings, but only report partial results derived in a few parameter settings as follows, because other parameter settings present similar trends and thus we omit them to avoid redundancy.

Basically, we consider there are 50 candidate miners in total and set the reward of generating a new block as 12.5 which is in line with the current setting of Bitcoin and will be shared by all voters of the winning candidate, i.e., \( \sum_i R_{ij}^t = 12.5 \). Also, we assume that voters own different numbers of stakes indicating their different weights during the voting process, where these stakes are assigned to them arbitrarily \( s_i \in \{1, 2, 3, 4\} \). Besides, we define that all voters can give each candidate up to one vote, and each voter needs to choose the top \( K \) super nodes that she considers reliable to give one vote.

A. Availability Function

First, in order to study the effect of unavailable probability on the voter’s merit, we calculate the cumulative reward of any voter until round \( t \) with different availability functions in Fig. 8, as well as different numbers of stake for the voter. As shown in Fig. 8(a), we present the cumulative reward of the voter when her stake varies and the availability function changes, as well as the case of not considering the unavailability factor, till the round \( t = 100 \). While Fig. 8(b) illustrates the three specific availability functions, i.e., \( d_1 \), \( d_2 \), and \( d_3 \), varying with unavailable probability \( u_j \in [0, 1] \). The three types of functions represent power function, exponential function and linear function respectively. And the scaling parameter in (3) is \( \rho = 5 \), which satisfies the requirement to guarantee the stability of virtual queue.\(^6\)

\(^6\)Other values that meet the requirements can be implemented in a similar way.

**Fig. 8.** Cumulative reward of the voter with different availability functions and numbers of stake.

**Fig. 9.** Evolution of the voter’s cumulative reward with different numbers of stake.

In the light of Fig. 8(a), we can observe that with the increase of the voter’s stake, the cumulative reward of it increases correspondingly. Furthermore, we compare two versions of our algorithm: one utilizing unavailability (with three different availability functions: \( d_1 \), \( d_2 \), \( d_3 \), respectively) and the other without it. We can find that the selection pressure with considering the unavailability can affect the cumulative reward of the voter significantly. The results under three different availability functions have no obvious difference, even though they have different decline speeds, as shown in Fig. 8(b). So the specific formula that defines availability has no distinct effect on the voter’s reward. Also, with the increasing number of stakes, the influence of involving unavailability becomes more significant, which can be seen from the increasingly large gap between the results.

B. Evolution of the Reward

Next, we investigate the evolution of the voter’s reward over time under different numbers of stakes and different availability functions, where the experimental results are presented in Figs. 9 and 10, respectively.

According to Fig. 9, one can easily observe that the more stake the voter voting to the winning candidates, the higher her cumulative reward is, which is consistent with the general expectation. In detail, the stake of voter \( i \), i.e., \( s_i \), is used as the voting weight, which affects her reward obtained from the winning candidate. Considering the case where the voter votes the same candidates as super nodes, she can receive the reward from the successfully voted candidates she supported, which is positively related to her voting weights as the super nodes divide reward according to the proportion of their supporters’ votes, i.e., \( \sum_{k=1}^{s} \frac{s_i \times c_{ik}}{s_i \times c_{ik} + 1} = \frac{s_i \times c_{ik}}{S_i} \). Thus, the greater the weight, the more the reward. Besides, because the voter will never suffer from any economic loss during the voting process, her total reward is increasing as the number of rounds increases.
In Fig. 10, we plot the evolution of the voter’s reward with different unavailability functions $d_1$, $d_2$, and $d_3$. It is clear that all these three functions make the voter’s reward present the same evolution trend. That is, the decline rate of availability function does not have a significant impact on the changing rate of the voter’s reward. This is in accordance with the requirement that the difference of availability functions should not affect whether the candidate can be selected as super nodes or not, which further determines that the voter’s reward will not be influenced.

### C. Effectiveness of Our Proposed Voting Consensus Algorithm

In this subsection, we investigate the effectiveness of our proposed voting consensus mechanism by comparing the voting results without trustworthiness evaluation (an example result of DPoS algorithm) and the voting results with trustworthiness evaluation in one round of voting. The number of super nodes that need to be voted is set to $K = 5$ or $21$, where the corresponding voting results are reported in Figs. 11 and 12, respectively. Note that in the case of $K = 5$, as only a few candidates with higher capability to be voted as super nodes are among the first 10 candidates, we report the voting results of them in Fig. 11 for better presentation even there are 50 candidates in total.

Before conducting the simulation experiment, we rank the capability of candidate $j$, i.e., $V_j$, in the current round according to their unavailability probability, successful mining rate and community rewards so as to obtain the objective ranking results in the absence of collusion, bribery, inertia and other negative factors, which are reported in the second column of TABLE I and that of TABLE II. And the voting results without trustworthiness evaluation in Figs. 11(a) and 12(a) are numerically presented in the third column of TABLE I and that of TABLE II, respectively. Similarly, the voting results with trustworthiness evaluation are shown in Figs. 11(b) and 12(b) are also reported in the fourth column of TABLE I and that of TABLE II, respectively.

According to Fig. 11(a), in the case of selecting 5 super nodes, the malicious candidate No. 4 bribes some voters to make his total number of votes greater than that of No. 7 in the voting process without trustworthiness evaluation. While after trustworthiness assessment of each voter based on our proposed algorithm, it can be seen that the malicious competition behavior of No. 4 has been successfully suppressed as shown in Fig. 11(b), and the ultimately-voted super nodes are exactly what we expect as reported in the fourth column of TABLE I. When $K = 21$, we assume that candidates No. 1 to No. 4 bribe voters in this round of voting. In fact, it is unnecessary for candidate No. 3 to bribe any voter as he has the highest real capability to be voted. Through comparing Figs. 12(a) and 12(b), one can find that the voting results with trustworthiness evaluation can definitely meet the basic requirement of selecting super nodes with higher capability; and candidates No. 1, No. 2, and No. 4 are successfully suppressed. The final 21 super nodes are presented in the fourth column in TABLE II. All the above experimental results demonstrate that our proposed mechanism can successfully prohibit malicious competition behaviors, such as the wrong election and bribery election, in the voting process. And the underlying reason is that with the help of trustworthiness evaluation for each voter, our proposed voting consensus mechanism can frustrate candidates to bribe and eliminate wrong behaviors of voters.

### TABLE I

| No. of Super Nodes | Capability Ranking | Voting without Trustworthiness Evaluation | Voting with Trustworthiness Evaluation |
|--------------------|-------------------|----------------------------------------|---------------------------------------|
| 1                  | 3                 | 3                                      | 3                                     |
| 2                  | 8                 | 8                                      | 8                                     |
| 3                  | 2                 | 2                                      | 6                                     |
| 4                  | 6                 | 6                                      | 2                                     |
| 5                  | 7                 | 7                                      | 7                                     |

### TABLE II

| No. of Super Nodes | Capability Ranking | Voting without Trustworthiness Evaluation | Voting with Trustworthiness Evaluation |
|--------------------|-------------------|----------------------------------------|---------------------------------------|
| 1                  | 3                 | 3                                      | 3                                     |
| 2                  | 23                | 1                                      | 23                                    |
| 3                  | 5                 | 2                                      | 5                                     |
| 4                  | 28                | 4                                      | 28                                    |
| 5                  | 9                 | 5                                      | 9                                     |
| 6                  | 32                | 6                                      | 32                                    |
| 7                  | 6                 | 9                                      | 6                                     |
| 8                  | 44                | 11                                     | 44                                    |
| 9                  | 14                | 13                                     | 14                                    |
| 10                 | 25                | 14                                     | 25                                    |
| 11                 | 38                | 15                                     | 38                                    |
| 12                 | 15                | 16                                     | 15                                    |
| 13                 | 11                | 23                                     | 11                                    |
| 14                 | 40                | 25                                     | 40                                    |
| 15                 | 26                | 26                                     | 26                                    |
| 16                 | 33                | 27                                     | 33                                    |
| 17                 | 29                | 28                                     | 29                                    |
| 18                 | 27                | 29                                     | 27                                    |
| 19                 | 30                | 30                                     | 30                                    |
| 20                 | 16                | 32                                     | 16                                    |
| 21                 | 13                | 33                                     | 13                                    |

![Fig. 11](image-url) Effectiveness of the voting consensus algorithm when $K = 5$.
In reality, we have conducted the simulation experiments with 516 candidates and 21 super nodes to trace the latest EOS setting. Due to the simulation results in different network settings are similar, we only show the ones in the simulation (i.e., 50 candidates and 21 super nodes) for avoiding redundant descriptions.

VII. CONCLUSION AND FUTURE WORK

In this paper, we propose an uncertainty- and collusion-proof voting consensus mechanism, which involves the selection pressure-based voting algorithm to deter wrong elections and adopts an incentive-compatible scoring rule for evaluating the trustworthiness of voting so as to avoid bribery elections. In addition, we take advantage of the large deviation theory to theoretically analyze the proposed voting consensus mechanism, based on which we can draw the following conclusions: 1) the growth rate of the voting failure rate can fall off with the increase of the selection valve more sensitively than with that of the selection pressure; 2) the effective selection valve decreases logarithmically with the increase of the voting failure tolerance degree and the decrease of the selection pressure; 3) the increase of either the voting failure tolerance degree or the selection valve will lead to the drop of the effective expectation of merit, which will remain stable after declining to a certain level. Extensive simulations verify the effectiveness of the proposed voting consensus mechanism.

Our selection pressure-based voting algorithm reflects the idea of exploration and exploitation and endows them with the same degree of importance. In future work, we will investigate effective voter’s voting choices by dynamically adjusting the importance of exploration and exploitation. In addition, we leave the update method of super nodes and candidates as an interesting topic for future research, which does not need to replace the super nodes after each round of block generation. A well-designed method can reduce the voting time and communication consumption, and can achieve faster and more effective transaction packaging on the premise of ensuring the reliability of the generator.

REFERENCES

[1] Y. Xiao, N. Zhang, W. Lou, and Y. T. Hou, “A survey of distributed consensus protocols for blockchain networks,” IEEE Commun. Surveys Tuts., vol. 22, no. 2, pp. 1432–1465, 2nd Quart., 2020.
[2] W. Wang et al., “A survey on consensus mechanisms and mining strategy management in blockchain networks,” IEEE Access, vol. 7, pp. 22328–22370, 2019.
[3] MarketsandMarkets, (2021), Blockchain Market by Component (Platform and Services), Provider (Application, Middleware, and Infrastructure), Type (Private, Public, and Hybrid), Organization Size, Application Area (BFSI, Government, it & Telecom), and Region—Global Forecast to 2025. [Online]. Available: https://www.marketsandmarkets.com/Market-Reports/blockchain-technology-market-90100890.html
[4] S. Squarepants. (2008). Bitcoin: A Peer-to-Peer Electronic Cash System. [Online]. Available: https://bitcoin.org/en.pdf
[5] S. King and S. Nadal, “PPCoin: Peer-to-peer crypto-currency with proof-of-stake,”Peercoin, Hague, The Netherlands, Tech. Rep., Aug. 2012, vol. 19, no. 1, pp. 1–6.
[6] D. Larimer. (2018). DPoS Consensus Algorithm—The Missing White Paper. Steemit. [Online]. Available: https://steemit.com/dpos/dpos-consensus-algorithm-the-missing-white-paper
[7] B. David, P. Gaži, A. Kiayias, and A. Russell, “Ouroboros Praos: Ouroboros Consensus Algorithm—The Missing White Paper,” 2014. In: Theor. Comput. Sci.
[8] (2018). EOS Voter Collusion Scandal Highlights Issues of Trust and Governance. [Online]. Available: https://www.investing.com/analysis/how-serious-is-the-eos-voting-scandal-200347581
[9] A. Ware. (2020). EOS Core Report. [Online]. Available: https://www.cryptoeq.io/coreReports/eos-.abridged
[10] J. Kang, Z. Xiong, D. Ye, D. I. Kim, J. Zhao, and D. Niyato, “Toward secure blockchain-enabled Internet of Vehicles: Optimizing consensus management using reputation and contract theory,” IEEE Trans. Veh. Technol., vol. 68, no. 3, pp. 2906–2920, Mar. 2019.
[11] I. Bentov, C. Lee, A. Mizrahi, and M. Rosenfeld, “Proof of activity: Extending Bitcoin’s proof of work via proof of stake,” IACR Cryptol. ePrint Arch., vol. 2014, no. 452, Dec. 2014.
[12] (2014). Burstcoin. [Online]. Available: https://www.burst-coin.org/proof-of-capacity
[13] Intel. (2017). Hyperledger Sawtooth. [Online]. Available: https://sawtooth.hyperledger.org/docs/core/releases/1.0/architecture/po.html
[14] X. Fan and Q. Chai, “Roll-DPoS: A randomized delegated proof of stake scheme for scalable blockchain-based Internet of Things systems,” in Proc. 15th EAI Int. Conf. Mobile Ubiquitous Syst., Comput., Netw. Services, Nov. 2018, pp. 482–484.
[15] G. Chaumont, P. Bugnot, Z. Hildreth, and B. Giraux, “DPoS: Delegated proof-of-private-stake, a DPoS implementation under X-cash, a Monero based hybrid-privacy coin,” X-Cash, Irvine, CA, USA, Tech. Rep., 2019, pp. 1–46.
[16] F. Yang, W. Zhou, Q. Wu, R. Long, N. N. Xiong, and M. Zhou, “Delegated proof of stake with downgrade: A secure and efficient blockchain consensus algorithm with downgrade mechanism,” IEEE Access, vol. 7, pp. 118541–118555, 2019.
[17] P. Zhang and M. Zhou, “Security and trust in blockchains: Architecture, key technologies, and open issues,” IEEE Trans. Computat. Social Syst., vol. 7, no. 3, pp. 790–801, Jun. 2020.
[18] J. Liu, W. Zheng, D. Lu, J. Wu, and Z. Zheng, “Understanding the decentralization of DPoS: Perspectives from data-driven analysis on EOSIO,” 2022, arXiv:2201.06187.
[19] Z. Su, Y. Wang, Q. Xu, and N. Zhang, “LBVS: Lightweight vehicular blockchain for secure data sharing in disaster rescue,” IEEE Trans. Dependable Secure Comput., vol. 19, no. 1, pp. 19–32, Jan. 2022.
[20] Y. Sun, B. Yan, Y. Yao, and J. Yu, “DT-DPoS: A delegated proof of stake consensus algorithm with dynamic trust,” Proc. Comput. Sci., vol. 187, pp. 371–376, Jan. 2021.
[21] J. Chen and S. Micalli, “Algorand: A secure and efficient distributed ledger,” Theor. Comput. Sci., vol. 777, pp. 155–183, Jul. 2019.
[22] A. Kiayias, A. Russell, B. David, and R. Olfatykov, “Ouroboros: A practically secure proof-of-stake blockchain,” in Proc. Ann. Int. Cryptol. Conf. Cham, Switzerland: Springer, 2017, pp. 357–388.
[23] B. David, P. Gaži, A. Kiayias, and A. Russell, “Ouroboros Praos: An adaptively-secure, semi-synchronous proof-of-stake blockchain,” in Proc. Ann. Int. Conf. Theory Appl. Cryptograph. Techn. Cham, Switzerland: Springer, 2018, pp. 66–98.
[24] C. Badertscher, P. Gaži, A. Kiayias, A. Russell, and V. Zikas, “Ouroboros Genesis: Composable proof-of-stake blockchains with dynamic availability,” in Proc. ACM SIGSAC Conf. Comput. Commun. Secur., Oct. 2018, pp. 913–930.
[25] M. Castro and B. Liskov, “Practical Byzantine fault tolerance,” in Proc. OSDI, vol. 99, Feb. 1999, pp. 173–186.
[26] J. Kwon, “Tendermint: Consensus without mining,” Draft Fall, vol. 1, p. 11, Aug. 2014.
[27] M. Yin, D. Malkhi, M. K. Reiter, G. G. Gueta, and I. Abraham, “HotStuff: BFT consensus with linearity and responsiveness,” in Proc. ACM Symp. Princ. Distrib. Comput., Jul. 2019, pp. 347–356.
[28] G. S. Veronese, M. Correia, A. N. Bessani, L. C. Lung, and P. Verissimo, “Efficient Byzantine fault-tolerance,” IEEE Trans. Comput., vol. 62, no. 1, pp. 16–30, Jan. 2013.

[29] R. Kapitza et al., “CheapBFT: Resource-efficient Byzantine fault tolerance,” in Proc. 7th ACM Eur. Conf. Comput. Syst., Apr. 2012, pp. 295–308.

[30] J. Liu, W. Li, G. O. Karame, and N. Asokan, “Scalable Byzantine consensus via hardware-assisted secret sharing,” IEEE Trans. Comput., vol. 68, no. 1, pp. 139–151, Jan. 2019.

[31] A. Miller, Y. Xia, K. Croman, E. Shi, and D. Song, “The honey badger of BFT protocols,” in Proc. ACM SIGSAC Conf. Comput. Commun. Secur., Oct. 2016, pp. 31–42.

[32] B. Guo, Z. Lu, Q. Tang, J. Xu, and Z. Zhang, “Dumbo: Faster asynchronous BFT protocols,” in Proc. ACM SIGSAC Conf. Comput. Commun. Secur., Oct. 2020, pp. 803–818.

[33] D. Schwartz et al., “The ripple protocol consensus algorithm,” Ripple Labs, San Francisco, CA, USA, Tech. Rep., 2014, vol. 5, no. 8, pp. 151–158.

[34] D. Mazieres, “The stellar consensus protocol: A federated model for internet-level consensus,” Stellar Develop. Found., vol. 32, pp. 1–45, Aug. 2015.

[35] I. Abraham, S. Devadas, D. Dolev, K. Nayak, and L. Ren, “Efficient synchronous Byzantine consensus,” 2017, arXiv:1704.02397.

[36] G.-T. Nguyen and K. Kim, “A survey about consensus algorithms used in blockchain,” J. Inf. Process. Syst., vol. 14, no. 1, pp. 101–128, 2018.

[37] A. J. Ganesh, N. O’Connell, and D. J. Wischik, Big Queues. Cham, Switzerland: Springer, 2004.

[38] J. Du, E. Gelenbe, C. Jiang, H. Zhang, Y. Ren, and H. V. Poor, “Peer prediction-based trustworthiness evaluation and trustworthy service rating in social networks,” IEEE Trans. Inf. Forensics Security, vol. 14, no. 6, pp. 1582–1594, Jun. 2019.

[39] N. Miller, P. Resnick, and R. Zeckhauser, “Eliciting informative feedback: The peer-prediction method,” Manage. Sci., vol. 51, no. 9, pp. 1359–1373, 2005.

[40] S. Micali, M. Rabin, and S. Vadhan, “Verifiable random functions,” in Proc. 40th Annu. Symp. Found. Comput. Sci., 1999, pp. 120–130.

[41] R. Selten, “Axiomatic characterization of the quadratic scoring rule,” Exp. Econ., vol. 1, no. 1, pp. 43–61, 1998.

Shengling Wang (Senior Member, IEEE) received the Ph.D. degree from Xi’an Jiaotong University in 2008. After that, she did her Post-Doctoral Research with the Department of Computer Science and Technology, Tsinghua University. Then, she worked as an Assistant and Associate Professor with the Institute of Computing Technology, Chinese Academy of Sciences, from 2010 to 2013. She is currently a Professor with the School of Artificial Intelligence, Beijing Normal University. Her research interests include mobile/wireless networks, game theory, and crowdsourcing.

Xidi Qu received the B.S. degree in computer science from Beijing Normal University, Beijing, China, in 2018, where she is currently pursuing the degree with the School of Artificial Intelligence. Her research interests include blockchain, consensus mechanism, and privacy preservation.

Qin Hu received the Ph.D. degree in computer science from The George Washington University in 2019. She is currently an Assistant Professor with the Department of Computer and Information Sciences, Indiana University–Purdue University Indianapolis. Her research interests include wireless and mobile security, blockchain, the Internet of Things, and crowdsourcing/crowdsensing.

Xia Wang received the Ph.D. degree from Wuhan University, China, in July 2006. Then, she began to work with the Faculty of Science, Beijing University of Technology, Beijing. Her research interests include stochastic processes and the limit theory. Her research work is mainly supported by the National Natural Science Foundation and the Beijing Natural Science Foundation.

Xiuzhen Cheng (Fellow, IEEE) received the M.S. and Ph.D. degrees in computer science from the University of Minnesota, Twin Cities, in 2000 and 2002, respectively. She is currently a Professor with the School of Computer Science and Technology, Shandong University. Her current research interests include privacy-aware computing, wireless and mobile security, smart cities, and algorithm design and analysis. She has served on the editorial boards of several technical publications and the technical program committees of various professional conferences/workshops. She has also chaired several international conferences. She worked as the Program Director for the U.S. National Science Foundation (NSF) from April 2006 to October 2006 (full-time), and from April 2008 to May 2010 (part-time).