# PoAS: A Stake-Based Double-Voting Blockchain Consensus Protocol

Yj1190590[*1]

**Abstract.** This study proposes a stake-based blockchain voting consensus protocol, where a voting process is used to accumulate stakes and to compensate for the main defects of proof of stake. The voting process, competition mechanism, and strategy for selecting branches are presented to introduce the main structure of a chain-based proof of stake system with a deterministic finality. In addition, other features, such as the decentration of development and scalability solution, are included herein.

KEY WORDS

1. Proof of Accumulative Stakes (PoAS). 2. Vote. 3. Finality. 4. Reward distribution.

## 1. Introduction

This study primarily improves the chain-based proof of stake (PoS) consensus protocol to make it more suitable for the requirements of cryptocurrencies.

In the aspect of consensus mechanisms, blockchain protocols can be broadly divided into two categories, namely chain-based protocol and Byzantine Fault Tolerant (BFT)–based protocol. We chose the chain-based mechanism because it has more advantages in terms of code complexity, degree of decentralization, and objectivity, which are more important for currency functions that require high security and robustness. Chain-based consensus has lower performance in comparing with BFT protocols because it has more validators to be synchronized. Multichain solutions such as side-chain or sharding are improved in our protocol, which helps to compensate for this shortage better.

In the aspect of competitive resource, there are two mainstream protocols, namely proof of work (PoW) and PoS. In comparison with PoW, PoS protocol does not consume a large amount of energy and does not pose the risk of centralized mining pools. However, it has two major problems, which are nothing at stake (NaS) problems and wealth concentration issue. If these two problems are resolved, PoS will have a great advantage over PoW.

NaS problem has existed since the birth of PoS because PoS miners do not need to pay any costs for reward competition in comparison with PoW miners. However, the PoS's competitive resources, the total amount of stake, is fixed, which is an attribute that can be used to build the "Finality" characteristic in a very convenient way. "finality" is a feature that most chain-based protocols usually don't have and it helps avoiding history attack[1], which is one of the major problems in the NaS. Concerning the double-voting problem[2], which is the other problem caused by NaS, it is resolved using a public voting mechanism in the protocol.

The centralizing tendency of wealth redistribution is also an inherent problem of PoS protocols. According to PoS logic, whenever miners spend the same amount of time, richer miners earn more; therefore, they tend to spend more time working. This implies that rich miners continue to become richer; this process will continue until only the richest users remain in the system. Furthermore, wealth distribution usually obeys the Pareto's law, which means most wealth is acquired by a few users; therefore, the poor users who will eventually disappear are much more than we thought. This problem can be solved using a method called

[1]Yj1190590 (3171228@qq.com)

"accumulating stakes" to reduce the work cost of a large number of users with small stakes and ensure that more users can participate in the wealth redistribution without reducing the overall competitive intensity.

"Accumulating stakes" means that the stakeholders assign their stakes to a group of people in an active or passive way. The ones who receive these stakes will work for the stakeholders to make the network active. The primary method to accumulate stakes is using the ability of "network dispersity." Regarding the definition of "network dispersity," consider the following fact: The greater the number of dispersed online terminals works together, the faster they can acquire the randomly distributed information in the network resulting from network latency. This type of "ability" is referred to as "network dispersity." Utilizing network dispersity to accumulate stakes can provide an objective and fair competitive environment.

A consensus protocol using the capabilities of stake and accumulating processes is called as proof of accumulative stakes (PoAS).

## 2. Scenario and Characters

The entire network can be considered as a canvassing and voting scenario that involves three types of characters according to a node's functions.

*Stakeholder*. As the owner of the currency, they are the subject of every transaction in the network. Therefore, the broadcast of every transaction is initiated by them. They are primarily responsible for answering the "canvass" requests from gatherers and publishing transactions along with their votes. A stake held by a stakeholder is considered as "votes" in this scenario, and the number of stakes voted to a miner determines his power in the mining competition.

*Miner*. They usually have gatherers that canvass for "votes" throughout the network, and they use these "votes" to compete in block generation. They are responsible for determining the main branch and verifying blocks and transactions.

*Gatherer*. They are affiliated to the miners who accumulate stakes with network dispersity. They detect nearby stakeholders and send a "canvass" request as quickly as possible. They primarily run on network terminals by being embedded in client apps or web sources.

The network scheme is shown in Figure 1.

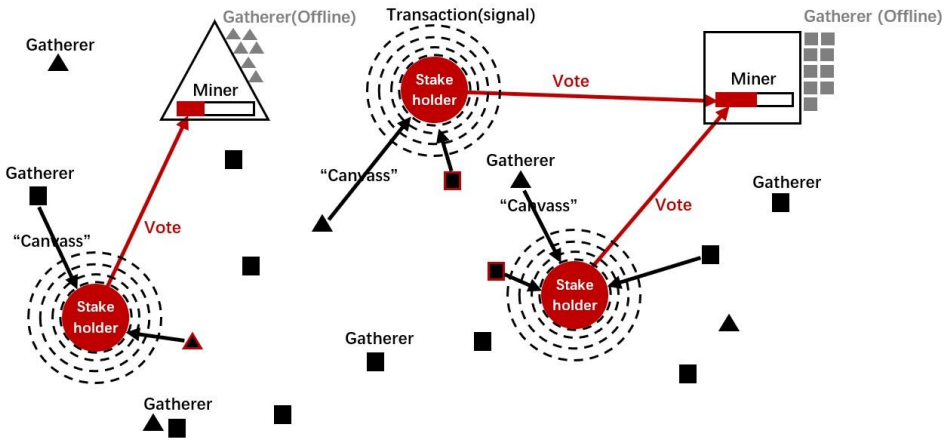


Fig. 1. Network scheme

Miners with more dispersed online gatherers have a greater likelihood of winning votes (i.e., stakes), which helps miners win mining competitions later.[3]

## 3. Consensus Process

We rely on two types of votes, namely the vote from stakeholders for miners and the vote from the miners for the current main branch, to reach a consensus that will be introduced below.

### 3.1 Vote for miners

The purpose of voting for miners is to accumulate stakes, especially those that are dispersed at the hands of stakeholders with lower stakes. The specific steps of the voting phase are as follows:

(1) Before each transaction is created, the stakeholder sends a broadcast signal and gatherers nearby send a "canvass" request to the stakeholder after receiving the broadcast signal.

(2) After answering the first request, the stakeholder waits the corresponding gatherer to feed back to its owner—the miner—to acquire his signature on the voting information, which includes both accounts of the two voting sides and time stamp.

(3) The stakeholder writes the signed voting information into the transaction structure and broadcasts the transaction.

A transaction with voting information is referred to as a "vote." Voting stakes are assigned to the miner for two purposes: competition for the rights to generate blocks and confirmation of the main branch. Therefore, during counting, the stakes of the two purposes are counted separately and recorded as x stake (generating blocks) and y stake (main branch). The two types of stakes are initially equal to the existing currency amount in each stakeholder. The specific steps of the vote-counting phase are as follows:

(1) Assign the x and y stakes of every vote within the last voting cycle (e.g., 6000 blocks) to the miners they voted for. If a stakeholder has multiple votes on one block, then they will be equally divided on each vote;

(2) Record the set of all votes in the last voting cycle as $T$, find the relevant votes of each miner from set $T$ and discard those before his last time of block generation, and collect the rest of them recording as set $T'$, where $T' \subseteq T$.[4]

(3) In set $T'$, collect x stakes acquired by each miner in Step (1) and record them as set $X$.

(4) In set $T$, collect y stakes acquired by each miner in Step (1) and record them as set $Y$.

According to the steps listed above, the stakes are accumulated in the hands of the miners through voting and the results are recorded as sets $X$ and $Y$. For the counting results of any time point, such as the time at block $a$, refers to the statistical results of one voting cycle ahead of block $a$ and are recorded $X_a$ and $Y_a$.

#### 3.1.1 Block competition

The purpose of the competition is to determine the miner that generates the next block; the miner operates as follows:

(1) The miner performs a mathematical operation based on constants such as timestamps and personal signatures. If the expected result meets the requirements of block generation, it is recorded as $hashProof() < target * d * effective(x)$, where $target$ refers to the goal; $d$ refers to the difficulty-adjustment parameter [5]; $x$ refers to the number of x stakes $(x \in X)$ acquired by the current miner. The $effective()$ function calculates the valid part of $x$, which will be introduced in detail in section 3.2.2.

(2) After the requirements of block generation are met, the miner packs the transactions received, generates the block, and publishes it. The earnings of each part involved in the process and other calculation parameters are packed into the block header simultaneously.

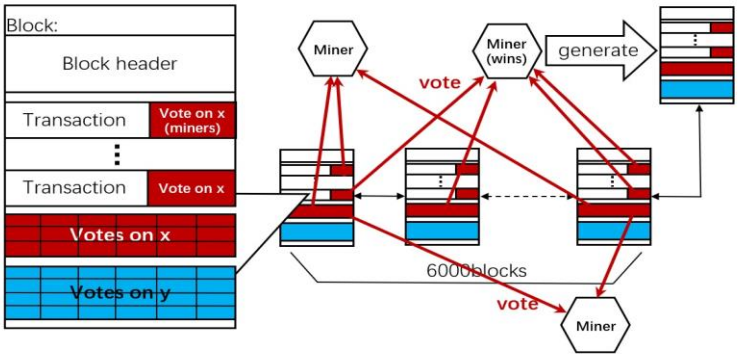


Fig. 2. Process of block competition
The system counts votes in existing blocks, and the miner with the maximum number of votes has the greatest chance to win the competition.

### 3.1.2 Parametric control

To control the voting frequency, the voting ability (x and y stakes) of stakeholders must be adjusted separately. For x stake, the time passed since the last vote of each account is considered as the adjustment coefficient. The stake starts at zero and increases by a certain proportion for every time unit $t$ (e.g., 10 blocks) that passes. The stake reaches its maximum value when a voting cycle (e.g., ~100 h for 6000 blocks) passes. For the y stake, $t$ is equal to the voting cycle, which implies that y stake comes into effect only once in an interval of a voting cycle. Stakes are also affected by the transfer frequency of currency. The stakes represented by moving currency are linearly increasing with the transfer interval in the same way.

Voting transactions without additional trading information should be needed to increase the flexibility of voting methods. However, such a voting transaction does not have a transaction fee to reward the miners. Therefore, it is necessary to provide a solution that can act as an incentive. We consider the total valid stakes of all voting transactions in the block as a parameter, which will affect the next block generation interval. As the stake increases, the calculation period will be reduced, for instance, if the calculation period is floating between 0.9 and 1.1s, this would directly affect the generation speed of the next block. In this case, it would be better to pack maximum votes. This parameter should be occasionally adjusted in terms of the average value.

## 3.2 Vote for main branch

In a PoAS protocol, the priority of branches does not depend on their length, but instead on their weight, which is not equal to the number of blocks but the votes obtained by the block. With reference to the description of the GHOST protocol, we define the weight of a PoAS branch as the sum of votes obtained by all the descendants of the root of the branch.

To determine the main branch, the miners cyclically sign the top block of the current main branch and publish the data to the network (for example, 60 blocks refers to one cycle recorded as “cycle2” to distinguish it from the other voting cycle.). These data are the vote for the main branch, which will be packaged into the block [6]. For these votes, the miners collect as many votes as possible into the blocks. Miners collecting voting information aligns with their own interests because collecting the votes on the current branch can increase the branch’s weight; hence, collecting votes on the other branches can reduce the competitive power of the corresponding miners in the current branch; this will be elaborated further later in section 3.2.2. For the blocks generated in the other branches, the miners would also try to collect as many references (block headers) as possible, which also align with their own interest, because doing so will significantly reduce the competitive power of the miners who generate these

blocks; this will also be elaborated further later in section 3.2.2. As a result, the blocks of each branch essentially contain the blocks and voting status of all branches; therefore, each chain only need to observe them from the perspective of itself.

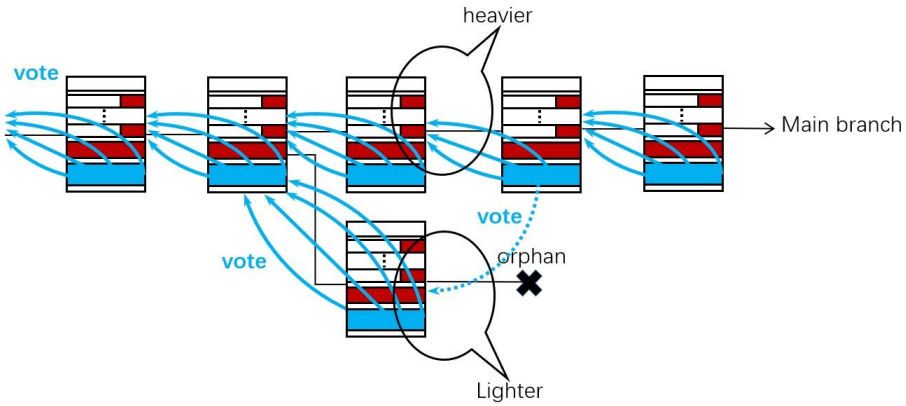


Fig. 3. Process of main branch selection

When a fork occurs, the miners vote between the branches and the votes are recorded in the next block. This determines the number of votes (stakes) for both branches. This also determines the weight of branch. The heaviest branch is the main branch.

### 3.2.1 Main branch and Save point

Suppose that the root of a branch to be weighted is $c$, $d$ represents a descendant of $c$, and the length of the chain $c \rightarrow d$ is shorter than one voting cycle2. If we need to calculate the weight of this branch from the perspective of $c \rightarrow d$, then the calculation method is as follows:

(1) First, count $Y_d$, and then equally distribute the y stakes of every miner in $Y_d$ to each vote of their that was recorded in $c \rightarrow d$;

(2) Count the y stakes distributed to the votes that were voted for $c$ or descendants of $c$, then add up all the results, which is the weight of this branch.

If the length of $c \rightarrow d$ is longer than one voting cycle2, it needs to be separated into small segments using one voting cycle2 as the unit and then calculated by segments. According to the calculation method, each stake can be voted only once for any branch shorter than one voting cycle2. Therefore, if a branch acquires more than half of the total stakes of the whole system within a voting cycle2, it is impossible to have a competitive branch then. We denote the branch as "finalized," and the block at the root of the branch as a "save point." All the blocks before the save point are irreplaceable, and all the blocks after the save points must be their respective descendants. The genesis block is the first save point, and the remaining save points will be established based on the previous save point. The method is as follows:

(1) Assuming $p$ as the latest save point and $b_i$ as the newly generated block, i.e., $p$'s descendant, we can compare the priorities of $p \rightarrow b_i$ to determine the current main branch;[7]

(2) Assuming the new block on the head of the main branch is $b$, $p_j$ is its ancestor and the length of $p_j \rightarrow b$ is shorter than one voting cycle2. Calculate the weight of branch with $p_j$ as the root successively; when it exceeds 2/3 (ensure an appropriate fault-tolerance) of the total stakes, $p_j$ becomes the new save point. Set $p = p_j$ and return to Step (1).

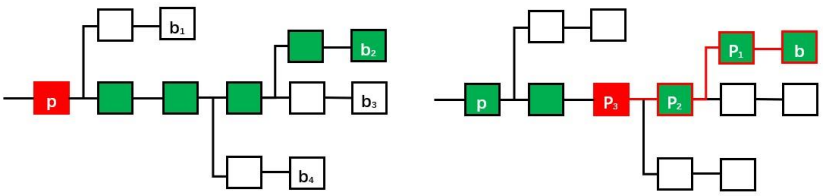


Fig. 4 Process of determining the main branch (1) and save point generation (2)

### 3.2.2 Incentives and penalties

To ensure perspective consistency across all chains, it is necessary to motivate the miners to actively record the votes and blocks from all branches. To achieve this, we consider the following incentives and penalties:

(1) If a miner generates a block in another branch and is referenced, then the miner is also considered to have generated a block in the current branch; therefore, his x stake's statistics are immediately zeroed out. In other words, as long as a miner generates blocks, he will be temporarily unable to participate in the competition for generating blocks in all branches, which is an incentive for other miners to ensure that all blocks can be referred to as far as possible.

(2) The *effective()* function used in section 3.1.1 is intended to motivate the miners to vote, especially for the right branches. The current miner's *x* stakes are distributed evenly to each of his votes within the last voting cycle2. Next, the stakes distributed to the votes that point to the current chain are counted as the result of *effective(x)*. The *effective()* function makes the miners' ability to generate blocks directly proportional to their correct votes. As a result, recording the wrong votes (pointing to the other branches) of other miners will reduce their ability to generate blocks, which is an incentive for miners to ensure that all wrong votes, as far as possible, are packed into the block. For the correct votes, as mentioned in section 3.2, they can increase the weight of the current branch and ensure to be recorded too.

(3) If miners aim all their votes for blocks that are relatively mature to ensure the correctness of the votes, it will have an impact on the identification of the current main branch. In this case, we stipulate that any vote that points to the current chain must point to the previous block.

### 3.3 Summary

According to the abovementioned consensus process, stakeholders can hand over their work to miners via the accumulating process. Users with lower stakes only need to vote once in a voting cycle to ensure their stakes do not miss the mining activities. This can compensate for the wealth concentration issue. Since the cost of stakeholders participating in network maintenance has been lowered, we can reduce the mining reward and avoid serious inflation.

Regarding the problem of multiple voting by miners, since the votes are already saved in the blocks before they are counted, the choices that anyone has ever made will be shown publicly; therefore, there will be no hidden competition branches and miners cannot make multiple votes. For the historical attacks that construct new branches, the proposed protocol provides two layers of protection against such. The first is the save point. Any historical attacks before the save point will be rejected. The second is the branch priority decision strategy. The priority of branches in PoAS is determined by the number of votes they have won. The branches with higher online stakes will be definitely heavier. Therefore, unless the historical stakes that an attacker possesses exceed all the online stakes of the current main

branch, it will not succeed. Thus, the NaS problems existing presently can be prevented using the proposed protocol.

However, since the miner–gatherer form can be embedded in apps and websites, many of those applications could accumulate stakes with network dispersity as miners, which would reduce the threshold for mining, thereby improving the sustainability of chains.

## 4. Reward Distribution

Mining activities are awarded from two parts:

(1) Transaction fees: Users need to pay service fees for every transaction to compensate for the resources consumed by miners. The transactions' service fees in PoAS will be payed to the miners they voted for and distributed along with block reward when the miners generate blocks, unlike Bitcoin where the service fees of all transactions in blocks are obtained directly by the block generators. However, for the purpose of security and encouraging higher service fees, we use a small portion of the service fee, e.g., 20%, as a reward to the miners who generated the current block.

(2) Block reward: For encouraging more users to participate in network maintenance, the system will issue another amount of currency as the mining reward in addition to service fees. Block reward is shared proportionally by the miner, the wallet account (introduced in section 6.1), and stakeholders. Considering the case wherein the number of participating stakeholders may be large, we divide the block reward into several parts and raffle among the voting records several times. The proportion of the stakes of the vote is the same as the probability of winning the lottery. Each part of the earning will be shared by the participants in the voting record that wins it.

## 5. Security

Stake accumulation resolves the participation issue of low-stake users, but some users, especially high-stake users, may have no intention of sharing their income with other parts. More importantly, the system can never avoid the possibility of being controlled by colluding groups if the users are all working cooperatively. Therefore, an individual mining style is introduced as distinguished from the stake-accumulating one, specifically as follows:

(1) Stakeholders add a label named "mining style" into the vote structures while they create them, representing two mining styles as "stake-accumulating" and "individual mining" denoted as "type-A" and "type-B," respectively.

(2) Type-A mining works as the process introduced above.

(3) Type-B mining removes the signal casting and signing phases before publishing the vote. The stakeholders will replace the miners doing everything the miners should do and take all the income.

Therefore, type-B miners can work individually as in common PoS systems. As long as the proportion of type-B stakes reaches a satisfactory level, the security will be ensured. We could accomplish that by dynamically adjusting the block rewards as follows:

Calculate the block rewards of the two mining styles separately and dynamically according to the voting data in the last voting cycle. By performing a certain calculation, we can increase the block reward of type-B mining if there are not enough active type-B stakes. The more stakes are lacked, the more reward will be provided especially when type-B stakes are less than type-A stakes, the reward should be increased further. Therefore, proportion of type-B stakes is guaranteed and the security could be ensured. Similarly, type-A block reward should be increased when the proportion of type-A stakes doesn't meet up the expectation, therefore, ensuring that enough resource is available for the stake-accumulating miners to work with.

## 6. Fairness

In this section, we will introduce how to keep fairness under the environment of network-dispersity competition.

As concluded before, security can be ensured even if all the type-A stakes are controlled by malicious groups. We can assume that all the roles of participation in type-A mining are acting for the purposes of benefits other than the purpose of damaging.

According to type-A mining process, we can see three interested parties participating in the process, namely stakeholders, miners, and provider of wallet applications. The fairness could be protected if all of them are abiding the protocol, but they would not in practice. To maximize their self-interests, they would affect the fairness by exhibiting the following behaviors:

(1) Wallet apps could replace the miners or collude with them (or to say bribed) to increase their benefit.

(2) Stakeholders could collude with the miners to increase their benefit.

(3) Miners could increase their chance of winning votes by bribing wallet providers or stakeholders.

We will discuss the responses to all these behaviors.

### 6.1 Wallet app replaces the miners

Wallet applications controlling the activities of all users are the foundation of the competition environment. If wallet apps cannot be rewarded enough, they could work as miner themselves controlling the stakeholders to vote to them and get the reward as miners. As a result, a "wallet account" field is added to the voting structure and a wallet sharing part is added to the reward distribution. However, it cannot prevent the wallet from replacing miners and enjoying both rewards. Some new rules are required to change the best strategy of wallet providers.

If we want the wallets to distribute the votes fairly and not distributing them to a few miners with a purpose, we must be able to distinguish between the two behaviors. Therefore, the concept of "sign density" is introduced. "Sign" means the wallet account field in the structure of vote, which represents different wallet providers, and "density" means the stake proportion of the votes with a certain sign. Ideally, the users of a certain wallet application should be randomly distributed over the network, and the miners who win the votes from those users should be dispersed too. Therefore, the sign densities in the votes of an honest miner should approach the densities of those signs in all votes. Although it will be affected by factors such as local languages, from the perspective of probability, the closer a sign density of a miner to the ideal value, the more likely the miner and wallet of the sign to be honest. Based on this, the following rules are established:

(1) Assuming there are $n$ signs, the sum of stakes of a certain sign $i$ is $S_i$ [8,9], then the density of $i$ is

$$D_i = \frac{S_i}{\sum_{k=1}^{n} S_k}$$

(2) Assuming that the sign $i$ is in a voting record that wins a part of the block reward from the miner $m$, the density of $i$ in the total votes is $D$, density of $i$ in the votes of $m$ is $d$, the miner $m$ and corresponding wallet provider of $i$ will then be considered as honest if $0<d\leq D$ and would be fully rewarded according to the sharing proportion. They will be considered as "have the possibility of cheating" when $d>D$ and their reward will be discounted to a certain extent. Assuming that $R$ and $R'$ is the reward before and after the discount, the discounting formula is as follows

$$R' = R \times \frac{D}{d}$$

(3) The reward of a wallet should always be higher than that of the corresponding miner when we set the sharing proportion, e.g., wallet for 26%, miner for 24% and stakeholder for the rest 50% of the total reward.

Following the above rules, the best strategy of the wallet providers cannot be anything but to distribute the vote with no interventions but making sure that all the rules are based on the assumption that more than half (by stake) of the wallet applications are honest. If the assumption is invalid, we have to use subjective methods, such as a full-stake vote to abandon the dishonest participants. This will not be examined in this article.

### 6.2 Bribing

Miners exchanging their interests with stakeholders or wallet providers to obtain an unfair voting result are both kinds of bribing activities. Colluding will be much more difficult under the control of "sign density" because miners will have to bribe almost every wallet at once, which is impossible as long as there are enough honest wallet applications.

### 6.3 Cheating means of miners

(1) Simulate gatherers. Creating a large number of gatherer nodes via simulation and adding those nodes to the P2P network to increase the success probability of vote canvassing.

To deal with this situation, we can control the process of creating P2P links, e.g., each node only builds connections with a certain number of nodes with the fastest response speed.

(2) Super gatherers: If a gatherer is located near the backbone network, it will win more votes because of network latency. If a large number of miners are using these super gatherers, the miners who rely on the number of network terminals to compete will lose competitiveness.

Although a variety of countermeasures can be added in the voting network as well as the fact that he rate of ROI of this behavior is not good, however, the possibility will always exist, which can also be seen as a potential threat to the fairness of network dispersity. Even at that, this does not cause a great impact on the consensus mechanism itself. Since the competitive mechanism will stay fair if the miners are all competing with the ability and quantity of super gathers. It will not consume many power resources either and will not violate the original intention of our design.

## 7. Incentive for developers

Current multichain solutions have no clear and simple profit model, which causes a lack of incentive for the developers to create expanding projects. In PoAS, wallet applications play an important part in the mining process and that makes them able to directly profit from the system. Developers will then be more willing to build expanding projects which helps to resolve the scalability issue. Besides, developers are encouraged to make better wallet applications, which avoids the risk of centralized development for client applications.

## 8. Conclusion

In comparison with the other existing protocols, the proposed protocol has the following advantages:

(1) No hash power competition and no high-energy consumption.

(2) No such problems as multiple voting and history attack caused by NaS.

(3) Motivates sufficient competitive strength to maintain the security of network without falling into the problem of wealth centralization and inflation.
(4) It has deterministic finality, objectivity, and complete decentralization but does not require any special node or extra expenses.
(5) Provides an incentive to the wallet developers, which helps to solve the scaling problem better.

## Conflict of Interest

Patent NO.: CN201811633256.0

## Notes

[1] History attack: the attacker constructs a new branch from a historical block and tries to replace the original one.
[2] Double-voting problem: a strategy for the miners to vote for each fork of the chain.
[3] The theoretical winning probability is directly proportional to the number of online gatherers.
[4] After a miner generates a block, the x stakes obtained will be emptied, which will prevent attacks such as stake grinding; the y stakes always maintain the counting interval of one voting cycle.
[5] Since this protocol may have a precise and objective counting on the active stakes of the current branch, it is not necessary to rely on the recent speed of block generation to adjust the difficulty, and parameter d can be set directly according to the sum of online stakes.
[6] To compress the voting information, the voting miners' public keys should be denoted as serial numbers of transactions within the previous 6000 blocks, and succinct proofs such as the MGS multi-signature scheme proposed in [6] should be used to replace individual signatures.
[7] Comparing the priorities of two chains begins with the forked position, which is their last common block, and count the weights of the two branches in their respective perspectives. The heavier chain has a higher priority.
[8] The $S_i$ of total votes is the sum of all stakes of the votes with sign $i$ in the last voting cycle; the $S_i$ of a miner is the sum of stakes of the votes with sign $i$ that he won and were recorded in set $X$.
[9] To exclude the occasional situation that very-high-stake users vote, the vote with highest stakes should be removed from the statistics of a miner's $S_i$.

## References


1. Yj1190590 (/yj1190590). "PoAS:A Stake-Based Double-Voting Blockchain Consensus Protocol." Github (accessed 29 April 2018) https://github.com/yj1190590/PoAS
2. Paul Firth. "Proof that Proof of Stake is either extremely vulnerable or totally centralised." BitcoinTalk.org (accessed 1 March 2016) https://bitcointalk.org/index.php?topic=1382241.0
3. Vitalik Buterin. "Long-Range Attacks: The Serious Problem With Adaptive Proof of Work." blog.ethereum.org (accessed 15 May 2014) https://blog.ethereum.org/2014/05/15/long-range-attacks-the-serious-problem-with-adaptive-proof-of-work/
4. Yonatan Sompolinsky and Aviv Zohar. "Secure High-Rate Transaction Processing in Bitcoin" No Publisher (2013) https://eprint.iacr.org/2013/881.pdf
5. Husam Ibrahim."A Next-Generation Smart Contract and Decentralized Application Platform" Github (2018) https://github.com/ethereum/wiki/wiki/White-Paper#modified-ghost-implementation
6. A. Boldyreva. "Threshold signatures, multisignatures and blind signatures based on the gap-Diffie-Hellman-group signature scheme." In Y. Desmedt, editor, PKC 2003, volume 2567 of LNCS, pages 31–46. Springer, Heidelberg, Jan. 2003