Proof-of-vote, validators selected by people-vote

Johan Nygren, johanngrn@gmail.com

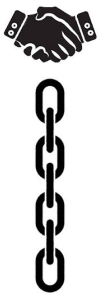
ABSTRACT: Proof-of-vote is a third generation of the Nakamoto consensus. With proof-of-vote, validators compete for people-votes, using proof-of-suffrage given by proof-of-person, and authorize transactions based on authority delegated by the consensus mechanism, just like proof-of-work or proof-of-stake. This logical conclusion of the Nakamoto consensus allows a “nation” of people to secure their own ledger, the equivalent of representative democracy for distributed ledger technology.

Introduction

The Nakamoto consensus is a majority consensus system based on popular vote. What distinguishes the different variations of Nakamoto consensus algorithms (such as proof-of-work, proof-of-stake, or proof-of-vote) is how voting rights are allocated.

Proof-of-stake VS proof-of-vote

Overall, in a proof-of-stake network at any given time, you could remove all coins from all validators, replace those with people-votes in exact same quantity and distribution, and it would operate identically. The only difference is that the people voting get a share of the transaction fees.



The Nakamoto consensus is a majority consensus system based on popular vote. What distinguishes the different variations of Nakamoto consensus algorithms (such as proof-of-work, proof-of-stake, or proof-of-vote) is how voting rights are allocated.

Validator selection in proof-of-vote

Validator selection is done using entities that are atomic, i.e., each entity has the same probability of being selected. This is achieved by using the voters, individual people, as the entities that are selected. Each person is assigned a unique identifier incrementally from 1, and every period (every block in a blockchain), a random number is generated, selecting the person with the ID `1 + randomNumber%population`. Voters give their vote to validators, and the voter that is selected by the randomly generated number authorizes their validator for the next period. To compare with existing infrastructure, the voters are analogous to individual miners in a mining pool, and the validator the mining pool.

The random number generation is done by letting the validators pre-commit hash onions (hash chains that are peeled off gradually) with random numbers, and then submitting their next random number when they are selected for the next period. `if(hash(submittedHash) == hashOnion[validator]) randomNumber ^= submittedHash; hashOnion[validator] = submittedHash;`. Collusion attacks are prevented by pre-committing a large enough number of hashes at once.