ALGORITHM OF REACHING CONSENSUS FOR LARGE BLOCKCHAIN NETWORKS

a@sumus.team, k@sumus.team, [rr@sumus.team](mailto:rr@sumus.team)

March 3rd, 2018

**Abstract**

We propose a new algorithm for reaching consensus – stake distributed Byzantine Fault Tolerant (sdBFT), that allows to increase the number of nodes of the network involved in consensus building by several orders in comparison with existing algorithms of BFT’s class and significantly increase the speed of transactions.

**1 Introduction**

When creating a Bitcoin coin and blockchain technology Satoshi Nakamoto used the Proof-of-Work algorithm as a consensus algorithm so-called PoW [1]. With the growing popularity of blockchain technology the features of the PoW algorithm associated with low transaction speed and, accordingly, their high cost, have been revealed.

Many experts from the IT industry have created new consensus algorithms or adapted already existing ones, eliminating weaknesses of PoW. As a result, there appeared consensus algorithms PoS, DPoS, LPoS, PoE, PoIT, pBFT. By this article, we tried to contribute to the development of consensus algorithms.

The need to establish our own consensus algorithm appeared when trying to create a blockchain, satisfying the following requirements for the blockchain network:

1. Time to create a new block is no more than 1 minute.
2. The blockchain network type is a corporate network.
3. The total number of nodes, that can participate in the consensus-building, can vary from to .
4. High transaction speed – at least transactions per second.
5. Implementation of the algorithm does not require significant energy costs.
6. Implementation of the blockchain algorithm should not require significant computing power, in contrast with PoW blockchains.

Previously proposed algorithms of consensus are recognized as unsatisfactory for the above requirements by authors for the following reasons.

Consensus PoW – the very first type of consensus implemented in the Bitcoin currency’s blockchain. Consensus is characterized by a low speed of the block closing and low transaction speed, according to the data [2] the transaction speed of Bitcoin currency is only 7 transactions per second. Formation of the blockchain block at PoW consensus requires significant computing resources. And the more a consensus participant has computing resources, the higher the probability for it to form a block, what leads to an unproductive computing race between the consensus participants.

Proof-of-Stake (PoS) consensus and its variations DPoS, LPoS were proposed to solve the problems of PoW consensus, connected with high computational costs and a low speed of closing of transaction block [3]. Despite the high speed of block closing and low requirements for hardware resources, algorithm PoS has limitations. The problem with PoS is centralization of coins (system resources). Blockchain user, having the maximum amount of system resources, receives even more coins for the provision services for the confirmation of the block. Consequently, the number of nodes, participating in a consensus, will evolutionarily diminish, what will lead to non-compliance with the requirements for the blockchain outlined above.

pBFT consensus – another alternative to the PoW consensus algorithm. Several pBFT consensus realizations have been proposed in the world, one of them – «Honey Badger» [4]. As shown in Fig. 1 the implementation of pBFT consensus works all the better, the smaller the number of nodes participating in the consensus.



Fig.1 Graph of block forming delay time dependence from the speed of incoming transactions

Figure 1 shows the graph of block forming delay time dependence from the speed of incoming transactions, where Nodes/Tolerance – the relation of the total number of nodes reaching consensus, and the number of nodes that have not reached consensus.

Curve 1 shows the time variation of block closing in relation to the speed of incoming transactions for 32 nodes, curve 2 – for 40 knots, curves 3-6 – for 48, 56, 64 and 104 nodes respectively.

Consensus works most effectively with a number of nodes not exceeding 40, the transaction speed for such a number of nodes reaches 2∙ transactions per second, while the closing time of the block does not exceed 40 seconds.

If the number of nodes exceeds 60, curves 5 and 6, then the transaction speed for such a number of nodes does not exceed 0.5∙ transactions per second, while the closing time of the block exceeds 100 seconds. The time of consensus for 104 nodes reached 6 minutes.

The authors suggest a different approach to solving the problem of performance loss in the pBFT algorithm implemented in the stake distributed Byzantine Fault Tolerant (sdBFT) algorithm of consensus.

**2 Main provisions and assumptions of sdBFT**

We proceed from the premise that the task of achieving consensus in the first approximation reduces to obtaining an agreed solution by participants of the distributed system in the event that some of their number did not participate in the coordination of the decision. This can happen for the following reasons:

1. An error occurred during the transmission of the message about the decision by one of the participants.
2. Too slow transmission of a message about the decision by one of the participants.
3. A failure in the work of the participant in the system.
4. Misleading when making a decision in the system, both intentional and unintentional.

If we consider the reasons 1-3 of non-participation in the development of consensus, then to make a decision it is sufficient that condition is to be carried out [5], where the number of participants who did not participate in the consensus, and number of participants who took the decision. In the event that participants appear in the system, not participating in consensus for the fourth reason, the task is reduced to the task of Byzantine generals, which has a solution when



(1)

If instead of generals we consider blockchain nodes participating in the development of consensus, it is obvious that this task is similar to the task of Byzantine generals. To solve the problem of increasing the time for reaching consensus it is proposed from a finite set of nodes , the cardinality of this set is , allocate a subset ( ) , and proceeding from the assumption about uniformity of properties distribution of nodes in all blockchain network, solve the problem not on but on under *n* ≫ *n*' . The total number of nodes in the entire blockchain network is set equal to . Denote the set of these nodes as .

Let there be given a function:

(2)

where – independent variable, corresponding to the current time. Assume that the value of function (2) corresponds to the current state at the moment .

Let there be given a function:

(3)

where – finite subset of the set of natural numbers of capacity .

Let subset be randomly selected from , where is given. Then get

(4)

Assume that – set of values of all , сorresponding to all current states .

Let the function maps to the set . Assume that the number of nodes obtained by such a mapping is , . Suppose that – number of the assigned master node, . If – forming block, in relation to which at some point in time set of nodes tend to reach consensus, then the hash function SHA-3 [6] above this block, denote as , and its values are denoted by . Then the result of calculating the digital signature, e.g., by algorithm EdDSA [7] with elliptic curve parameters edwards25519[7] will be equal to .

# 3 Description of the algorithm

1. Suppose that at time node with number records into blockchain .
2. Select all , including using the function . Consensus building is carried out on a half-open interval .
3. If the master node recognizes inclusion of the record as valid in block , the master node passes this record to all nodes from the for testing and inclusion in the block . Otherwise, the record is denied without notice.
4. The new record is included in the block before the moment . The master node sends the message to the same nodes about fixing block . All nodes from calculate the value of hash function equal assumed .
5. Each node computes an digital signature:

(5)

and passes it to the node

1. The master node waits digital signatures for the time after the moment . In the moment a tuple is generated on the master node

(6)

The master node checks each signature from (6) and counts the number of valid signatures. Some node signatures from may be voting "against" or may be incorrect in that case, when the node with some number will appear in , which  
a) recognizes record as incorrect;

b) at the time has a blockchain state , different from the state for the node ;

c) distort record when forming a blockchain block.

1. The master node calculates the number of valid signatures and verifies fulfillment of inequality:   
    (7)  
   If (7) is not implemented, then the master node makes a conclusion, that no consensus has been reached, otherwise, the number is made up for block :  
    (8)  
   for which and , which is the digital signature of the node with the number , are calculated.
2. Denote number  
    (9)  
   as a new closed block, which will be considered a new blockchain state , relevant to moment . The master node sends (9) to all nodes of the set .
3. Suppose that at each node of with the number , verification of and (9) is carried out. If the verification is passed, then the block is added in the node’s blockchain number and blockchain on the node goes into state . If this node does not receive the above signatures for verification in the time interval , where – data transmission delay time, then the node with the number will consider the consensus unreached and will choose a new set on the basis of the old state *d*, applying (3).

# 4 Building of function

Compute the double hash of (9), denoting the resulting number . Construct a pseudo-random bit sequence of the form:

(10)

Get the next bit record:

(11)

Divide sequentially bit record (11) without omissions and overlaps on the tuples by bits in each, of which build set of numbers of nodes from , when , as a result, the master node will always be a node, the number of which is formed first. If already received number is accidentally repeated then the re-obtained number is skipped.

# 5 Conclusions

According to the authors the proposed sdBFT algorithm should have a higher speed-in-action compared with BFT algorithms. Changing the capacity of the set , it will be possible to control the speed of new blocks creation, in other words – algorithm speed.

Potentially large number of consensus participants complicates the preliminary collusion, when a group of voting nodes forms a new block, managing the composition of the block at its discretion, since at the next consensus setting another set of voting nodes will be selected.

Pseudo-random selection of a plurality of voting nodes according to the authors will not allow to have a significant influence on the choice of nodes on the next vote.

In the future, the authors propose to obtain experimental confirmation of theoretical positions, outlined in this article, evaluate the speed of the sdBFT algorithm, explore possible blockchain network locks.

# References

[1]  Satoshi Nakamoto (2009). “Bitcoin: A Peer-to-Peer Electronic Cash System”. www.bitcoin.org .

[2]  (2018.01.10). “Transactions Speeds: How Do Cryptocurrencies Stack Up To Visa or PayPal?”. https://howmuch.net/articles/crypto-transaction-speeds-compared.

[3]  BitFury Group (2015.09.13). “Proof of Stake versus Proof of Work”. http://bitfury.com/content/5-white-papers- research/pos-vs-pow-1.0.2.pdf.

[4]  Andrew Miller, Yu Xia, Kyle Croman, Elaine Shi, Dawn Song (2016). “The Honey Badger of BFT Protocols”. https://eprint.iacr.org/2016/199.pdf.

[5]  Leslie Lamport, Robert Shostak, Marshall Pease (1982). “The Byzantine Generals Problem”. ACM Transactions on Programming Languages and Systems. T.4, 3: 382–401.

[6]  National Institute of Standards and Technology. SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions. <https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.202.pdf>.

[7] S.Josefsson, I.Liusvaara. “Edwards-Curve Digital Signature Algorithm (EdDSA)”. IETF RFC. https://tools.ietf.org/html/rfc8032.