Consensus algorithms for peer-to-peer data storage networks

https://doi.org/10.31713/MCIT.2020.28

Artur Joshi
National University of Water and Environmental Engineering
Institute of Automatics, Cybernetics and Computer Engineering
Rivne, Ukraine

Abstract—**Recently blockchain applications became a standard for peer-to-peer (P2P) data storage networks for its security, immutability and decentralization. There are huge number of applications built on blockchain basis from digital currencies to electronic voting and healthcare. As of anonymous nature of blockchain, it has a problem of trust of the nodes of which networks consists. Consensus algorithms are aimed to solve this issue providing protocols which make network data being protected by every single node in a system.**

Keywords—distributed applications, blockchain, consensus algorithms, data storage.

I. Introduction

Blockchain technologies attract huge amount of attention nowadays. This distributed immutable network is expected to change commerce, government and healthcare spheres drastically as of its secured, fault-tolerant and anonymous nature [1]. The biggest application of blockchain technology is Bitcoin – cryptocurrency exchange network which allows users to swap goods with no third party in between of seller and buyer. This network already proved its value by growing Bitcoin exchange rate in 70 times from 2015 until 2020 [2].

Another application is smart contracts which allow people writing piece of code which only executes if all conditions met before certain point of time. Typically condition means sending some amount of money which allows this technology to take its place in crowdfunding, science projects investments or any other field which requires guarantees from both sides of commercial process [5].

All these systems require its nodes being consistent at every moment of system working. Consistency typically means data of the system to be not self-controversial [3]. In peer-to-peer system it can also mean that system can identify system's state validity by itself by provided consensus algorithms. Consensus algorithms are aimed to solve concurrency problems which typically arise in distributed systems, and malicious attacks inside the system.

These algorithms help system to be in operational state at high rate providing good uptime keeping users data and values safe and accessible.

II. Blockchain Networks

A blockchain is distributed database of records which can be understood as chain of transaction which have been processed in a system. Each transaction should be verified by consensus of majority of the participants of the system. Once information is committed to a database it cannot be erased or changed. The blockchain contains a certain and verifiable record of every single transaction ever made in a system [1].

The main algorithm can be described as follows. Every node can perform any of defined system actions changing system's state. Each of change is called as a transaction. One or more transactions can be grouped into a block. Each block always consists of data to be stored and some service data. Service data always include hash of the current block and hash of the previous one (1).

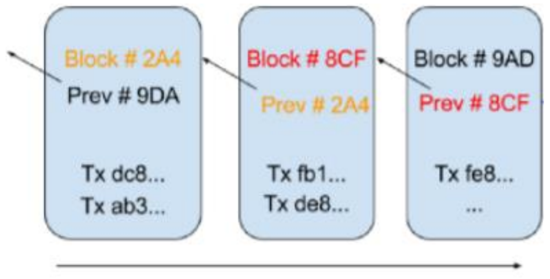

Figure 1. Blockchain block structure

Resulting structure makes block immutable once it is added. Consider the case when malicious attacker tries to change value of the transaction of block #8CF. In this case hash of this block is changed which makes it invalid. Even if attacker tries to change hash value of this block it does not become valid as the next block #9AD contains expected hash value. That said attacker should overwrite the whole chain after target block. Bitcoin network is built on the principle that each block can be added not sooner than each 10 minutes. This is achieved by adding complexity to block hash calculation which is called as Proof-of-Work (PoW). Proof-of-Work is one of the consensus algorithms which makes all nodes in the system working on finding next block's hash value. This solution works perfectly for Bitcoin network as security and consistency is a key feature of digital money system.

However not every system can afford to wait 10 minutes to commit data state change so there are needed advanced consensus algorithms which are fast but keep trust and safety in a system.

III. Consensus Algorithms

As showed before, one of the most popular consensus algorithms is Proof-of-Work (PoW). This algorithm is based on the idea of adding complexity to block's hash calculation making this process time-consuming enough to protect the system from being brute forced. The main disadvantage of this algorithm is vast computational resources expenses which usually are not rewarded as only one node of the system gets block generation fee [6].

Another option is to use Proof-of-Stake (PoS) algorithm. This algorithm is based on choosing verifying node on its stake, which is amount of digital money of the node. System chooses verifier randomly in respect of their money or age. Each node which wants being chosen as a verifier should lock some amount of money as an insurance which is automatically withdraws from account in case of fraud. This algorithm is more energy efficient and requires from attacker having a half of the whole network money to perform attack. The main disadvantage of the system is motivation to concentrate money in one account to have chance of verifying higher [7].

There is also Proof-of-Stake algorithm improvement which is called Delegated Proof-of-Stake (DPoF) which allows people with high stakes choose trustees to have right of signing block. This makes system extends range of verifiers without concentrating money in specific accounts.

Leased Proof-of-Stake (LPoS) is a modification of Proof-of-Stake algorithm. Currently this algorithm is only supported by Waves company. This algorithm allows every user to hand over balance as a lease to mining nodes. Miners should share a part of the profit with users. Thus, this consensus algorithm allows users gaining mining income without actual mining.

Proof-of-Capacity algorithm works as follows. Each miner picks big data volume, which must be written to the filesystem. For each new block, miner reads small data set from the whole saved data and returns deadline result as an elapsed time from last block creation. Miner got minimal deadline time sign the block and gains reward for a transaction.

Proof-of-Importance algorithm is based on nodes importance. Importance defines as value of a balance and number of signed transactions. Unlike PoS, PoI also considers users activity in a network. This approach forces users not only get money grow, but also actively use them.

The description of the algorithm was published in 2014 as a potentially new and more reliable algorithm for Bitcoin. The authors of the PoA algorithm tried to combine the two most popular algorithms, such as Proof-of-Work and Proof-of-Stake, in order to increase the level of protection against potential attacks (51% attack, Denial-of-Service attacks (DoS). the algorithm is as follows: each miner of the blockchain network tries to generate an empty block header, which includes the hash of the previous block, the public address of the miner, the index of the current block in the blockchain and a nonce; after generating an empty block header that meets the current complexity requirements, the node sends this header to the blockchain network; all nodes in the network consider the header of such a block as data received from pseudo-random owners. Stakeholders are selected using the hash of the sent out block header and the hash of the previous block + N presets using the follow-the-satoshi algorithm; each online stakeholder checks the received, empty block header for its correctness. During the check, each received header checks whether it is one of the first N-1 stakeholders of the "lucky ones" of this block, and in this case signs the header of the empty block with its secret key and sends it to the blockchain network; when the N-th stakeholder sees that he should become a signer of this block, he, in addition to the header of the empty block, adds a block with included transactions (he chooses the number of included transactions), all N-1 signatures from other stakeholders and signs the block; stakeholder N sends out a new prepared block. Nodes receive this block, make sure it is legal and add this block to the blockchain. The reward for transactions received by the N-stakeholder is distributed between the miner and the N "lucky" stakeholders.

The Intel Proof-of-Elapsed-Time (PoET) Consensus Algorithm is currently used as the consensus algorithm in the Hyperledger Sawtooth project. Essentially, the PoET algorithm is similar to the PoW algorithm, but without using the significant amount of power needed to operate the PoW algorithm. The equipment used to generate the block creates a block and then switches to other tasks not related to the generation of blocks until its time comes in order to generate the next block. The built-in mechanism ensures that each of the network nodes is sure that the time is chosen randomly: it is initially not less than determined for all nodes of the blockchain network, and also that the winning node expected the necessary time.

The Practical Byzantine Fault Tolerance (pBFT) algorithm was proposed back in 1999 as a mechanism ensuring the distributed networks integrity. During its operation, pBFT intensively uses the network to exchange messages between network nodes to ensure a coherent network operation. pBFT is best suited for work in trusted networks, such as intra-corporate or inter-organization blockchains [8]. The principles of the algorithm are: the node that receives the transaction sends it to all nodes in the current network. The content of the transaction is not verified a priori; each node that receives data from all other network nodes, checks them and if more than 2/3 of the "votes" are received for the transaction and accepts it (2).

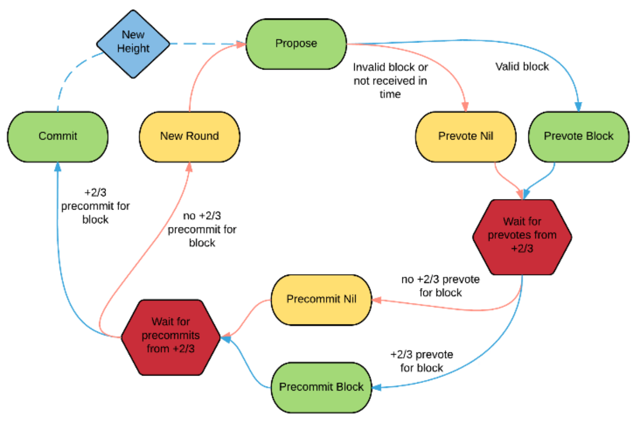
New Height
Propose
Invalid block or not received in time
Valid block
Commit
New Round
Prevote Nil
Prevote Block
+2/3 precommit for block
no +2/3 precommit for block
Wait for prevotes from +2/3
no +2/3 prevote for block
Precommit Nil
Wait for precommits from +2/3
+2/3 prevote for block
Precommit Block

Figure 2. Byzantine Fault tolerance algorithm

The main advantage of this algorithm is allowing nodes to not spend the significant costs of calculations, as is the case with PoW and high performance with a relatively small number of nodes of the blockchain network.

However, this algorithm has such disadvantages as high overhead costs for interactions between network nodes, significant limitation in the size of the network, as a sufficiently large network will be “blocked” by high network costs between nodes. Also, algorithm has a significant vulnerability in the construction of small networks (<20 nodes), as well as networks with one controlling (managing) person, for example, within the same organization.

PBFT algorithm is also widely used recently as of popularity of Tendermint – software that securely and consistently replicates application over multiple machines. Security means that up to 1/3 of the machines fail does not fail the system [4].

Unlike Proof-of-Work or Proof-of-Stake, where anyone can become a miner at any time, in BCA only so-called validators can take part in the formation of the blockchain.

The way of how an ordinary network participant becomes a validator depends on the specific implementation. In the simplest case, validators are declared in the genesis block and their list does not change in the future (the main thing is that there should be strictly less than 1/3 in the initial list of design validators). In the Tendermint, it is easy to implement the rotation of validators. To do this, it is enough to indicate in the protocol a special transaction that will be sent by the participant if he wants to run. Additionally, it is possible, as inside the Lisk, to enter a vote for candidates, or choose them in accordance with some already existing parameters.

In the Tendermint implementation, participants can always get an exact list of validators for any block. They are identified by their public keys, and during the voting process they sign messages sent to other validators and ordinary network participants with the corresponding private keys. Thus, it is always possible to identify the author of a vote and be sure that no one from the outside network can take part in building consensus.

An accountable BFT algorithm is one that can identify all Byzantine validators when there is a violation of safety. Traditional BFT algorithms do not have this property and provide no guarantees in the event safety is compromised. Of course, accountability can only apply when between one-third and two-thirds of validators are Byzantine. If more than two-thirds are Byzantine, they can completely dominate the protocol, and we have no guarantee that a correct validator will receive any evidence of their misdeeds. Furthermore, accountability can be at best eventual in asynchronous networks - following a violation of safety, the delayed delivery of critical messages may make it impossible to determine which validators were Byzantine until sometime after the safety violation is detected. In fact, if correct processes can receive evidence of Byzantine behavior, but fail irreversibly before they are able to gossip it, there may be cases where accountability is permanently compromised, though in practice such situations should be surmountable with advanced backup solutions. By enumerating the possible ways in which a violation of safety can occur, and showing that in each case, the Byzantine validators are identifiable, a protocol can be shown to be accountable. Tendermint’s simplicity affords it a much simpler analysis than protocols which must manage leadership elections [9].

The simplest scheme of pBFT algorithm work is a chain of events (1)

NEWHEIGHT -> (PROPOSE -> PREVOTE -> PRECOMMIT)+ -> COMMIT -> NEWHEIGHT ->... REFERENCES (1)

If the block is fully valid and managed to reach all validators. Suppose A is still malicious, so this time he will try to prevent the creation of the block.

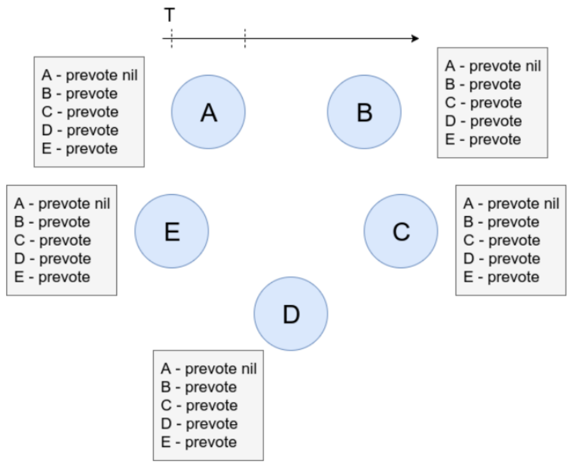

Figure 3. Pre-vote state of pBFT algorithm

All validators have enough pre-vote messages to send pre-commit messages. Consider sender A is malicious and it sends nil message.

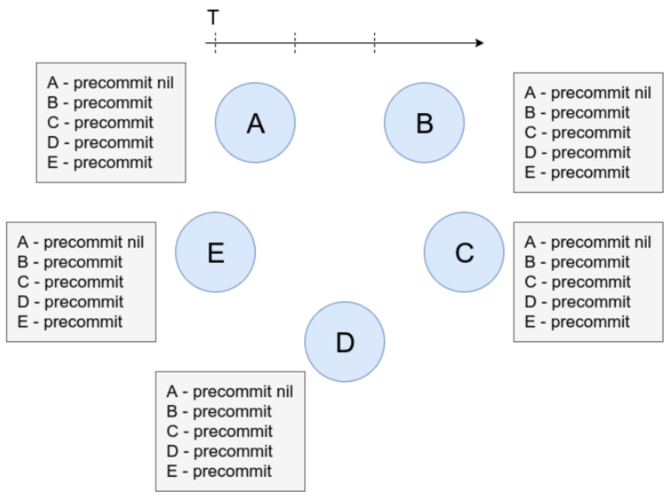

Figure 4. Pre-commit state of pBFT algorithm

Anyway, that this did not create issues for other participants, as they have 2/3 pre-commit messages in order to create a new block. The theory of BFT is decades old, but software implementations have only became popular recently, due to the success of "blockchain technology" like Bitcoin and Ethereum. In practice, the blockchain data structure optimizes BFT design. Tendermint consists of two chief technical components: a blockchain consensus engine and a generic application interface. The consensus engine, called Tendermint Core, ensures that the same transactions are recorded on every machine in the same order. The application interface, called the Application BlockChain Interface (ABCI), enables the transactions to be processed in any programming language [4]. Unlike other blockchain and consensus solutions, which come pre-packaged with built in state machines (like a key-value store, or scripting language), developers can use Tendermint for BFT state machine replication of applications written in any programming language and development environment is right for them.

Byzantine Fault Tolerant consensus provides a rich basis upon which to build services that do not depend on centralized, trusted parties, and which may be adopted by society to manage critical components of socioeconomic infrastructure. Tendermint, as presented in this thesis, was designed to meet the needs of such systems, and to do so in a way that is understandably secure and easily high performance, and which allows arbitrary systems to have transactions ordered by the consensus protocol, with minimal fuss [9]. Careful considerations are necessary when deploying a distributed consensus system, especially one without an agreed upon central authority to mediate potential disputes and reset the system in the event of a crisis. Tendermint seeks to address such problems using explicit governance modules and accountability guarantees, enabling integration of Tendermint deployments into modern legal and economic infrastructure.

REFERENCES

[1] Underwood S. News: Blockchain beyond bitcoin. *Communications of the ACM Volume 59, Number 11. 2016. pp. 15-17.*

[2] Ciaian P., Rajcaniova M., Kancs d’A. The digital agenda of virtual currencies: Can BitCoin become a global currency? *Information Systems and e-Business Management, Number 14. 2016. pp. 883-919.*

[3] Tanenbaum A.S., Van Steen M. Distributed Systems: Principles and Paradigms, 2nd ed., *Upper Saddle River. 2007. pp. 306-315.*

[4] Kwon J., Tendermin: Consensus without mining. *2014. pp. 2-9*

[5] Wood G. Ethereum: A secure decentralized generalized transaction ledger. *2014. pp. 15-28.*

[6] Bentov I., Gabizon A., Mizrahi A. Cryptocurrencies without Proof of Work. *2017. pp. 3-15.*

[7] King S., Nadal S. PPCoin: Peer-to-Peer Crypto-Currency with Proof-of-Stake. *2012. pp. 1-4.*

[8] Veronese G.S., Correia M., Bessani A.N., Lung L.C., Verissimo P. Efficient Byzantine Fault Tolerance. *2011. pp. 4-10.*

[9] Buchman E. Tendermint: Byzantine Fault Tolerance in the Age of Blockchains. *2016. pp. 31-33, 88.*