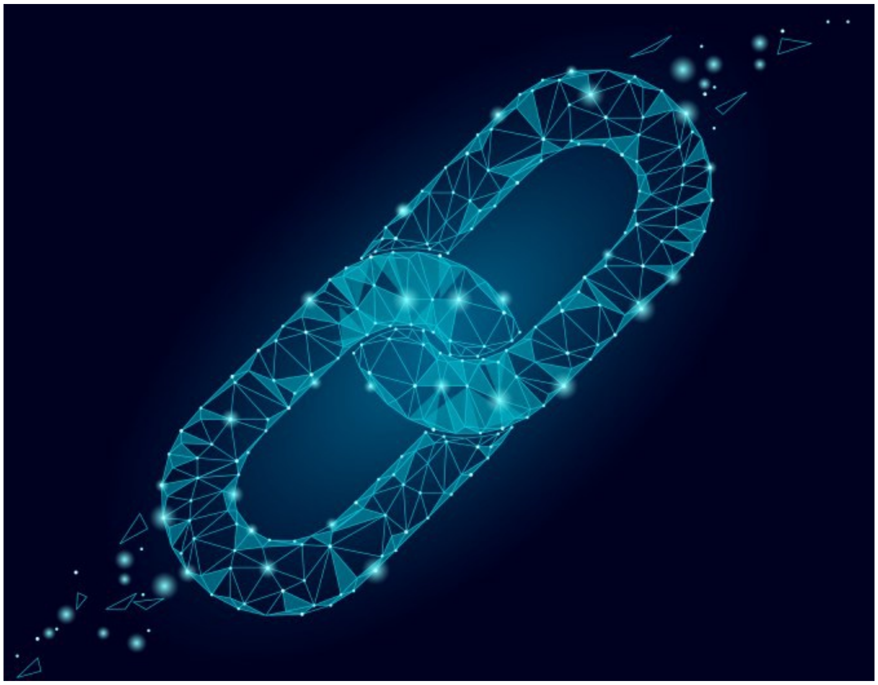
Blockchain Consensus Algorithm: PoW, PoS and Beyond

Justin Chan Follow
Nov 12, 2017 · 9 min read ★



One of the most critical aspects to understand blockchain—its speed, applications, and potentials is consensus algorithms. It determines everything from network security, confirmation speed, to environmental friendliness.

As crucial as it is, few comprehend how such a dynamic concept works in practice; and among the most misunderstood aspects include just how new blocks of information are securely added to the ledger, considering *no* centralized authority is engaged to maintain the integrity of the network.

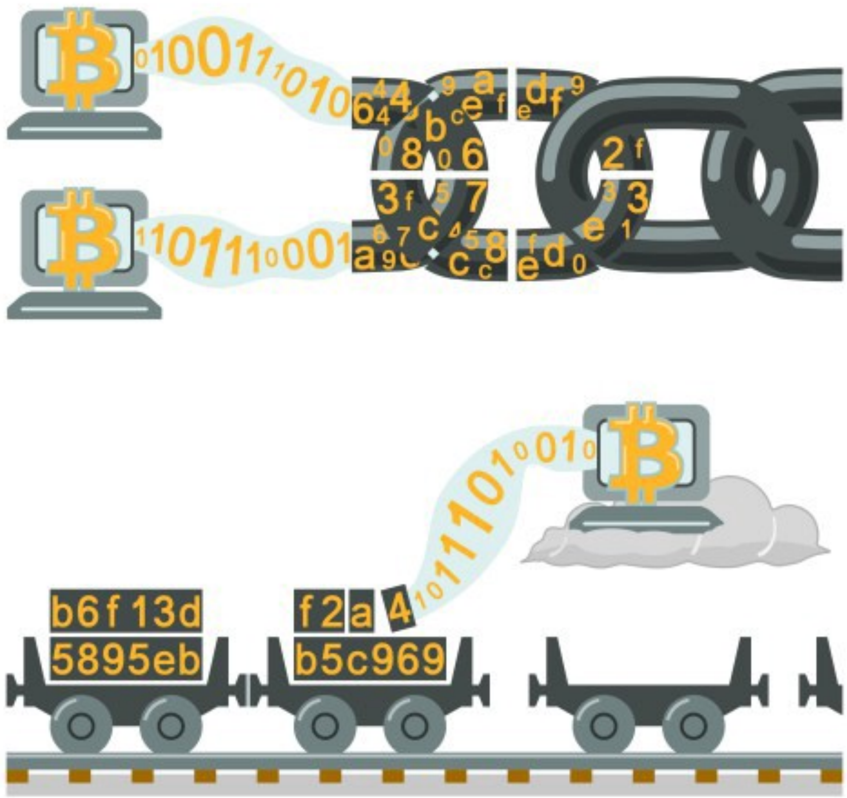
As a recap from an earlier post, blockchain provides a way for transactions to be ordered and verified in a distributed ledger, and ultimately provides a record of truth over a period of time.

Without a central intermediary, the network of participating users that make up this system need to agree on the validity of what's being added to the ledger, using a set of *pre-defined* rules. A ***consensus*** needs be reached for the majority of the nodes in the network. But just how effective it is to implement such a consensus remains a work in progress til this very day….

· · ·

The emergence of Proof-of-Work

The problem of ensuring reliability in decentralized systems was first formalized in the scholarly paper "The Byzantine Generals' Problem", published back in 1982. In the illustration created by the authors, a Byzantine army is attacking a city and has it completely encircled. To proceed, the generals who are dispersed around the city's periphery must agree on a battle plan; but while some generals want to attack further, others want to retreat. And to complicate matters, the generals are so far apart from each other that messengers are required to deliver communications between them, while one or more generals may also be a traitor intending to sabotage the situation.



Given such a scenario can the army execute any strategy?

Yes they can, but it will require a majority (i.e. 51% or more of the generals) to agree on a chosen strategy.

This Byzantine Generals' Problem is analogous to solving consensus issues on the blockchain. The nodes must all agree on a certain set of rules, and be able to move forward by agreeing on a particular assessment of transaction information before it is added onto the chain. This is not easy—thousands of people use the network and must ensure that they agree on the validity of new information that's to be added, on top of preventing bad actors from sabotaging the ledger and rewriting history. So, a specific type of **consensus mechanism/algorithm** must be adopted. This protocol enables a coherent group of network entities to work together in order to update the ledger in a secure fashion.

. . .

In the words of Ethereum founder Vitalik Buterin, the purpose of a consensus algorithm is to *"allow for the secure updating of a state according to some specific state transition rules, where the right to perform the state transitions is distributed among some economic set"*. Buterin identifies this set as one containing users who can "be given the right to collectively perform transitions via some algorithm" and who *"must be securely decentralized—meaning that no single actor, or colluding subsets of actors, can take up the majority of the set, even if the actor has a fairly large amount of capital and financial incentive"*. The consensus algorithm, therefore, ensures that through decentralization, the next block to be added to the chain is the *one and only* version of the truth in the system.

The first consensus protocol to emerge in the crypto era via the Bitcoin blockchain was "Proof-of-Work". PoW requires that each validating user proves that he has performed a computational action as a way to prevent the network from being attacked by such forces as spam and denial of service. Each node tries to solve complex cryptographic problems using their own computational resources—the one who eventually finds the solution can confirm the transactions and write the block onto the chain. This means that the nodes (also known as miners) are competing with each other to create the next block of transactions on the blockchain. In turn, the winning miner receives cryptocurrency tokens as a reward for the exceptional levels of time and energy he spent generating the solution; a Bitcoin miner, for example, will receive Bitcoin as his reward. This reward system incentivices miners to generate the right solution and ensures the network remains secured; while the newly minted cryptocurrency is added to the overall circulating supply of coins on the network.

The common drawbacks of PoW

Given the heavy utilization of computational resources involved in mining, PoW is considered to be costly, wasteful and inefficient. Having thousands of miners working on just one solution each time is an excessive use of resources, especially as the only block that has any value thereafter is the one that is solved. With each new block being mined, therefore, there's a heap of effectively useless bi-product.

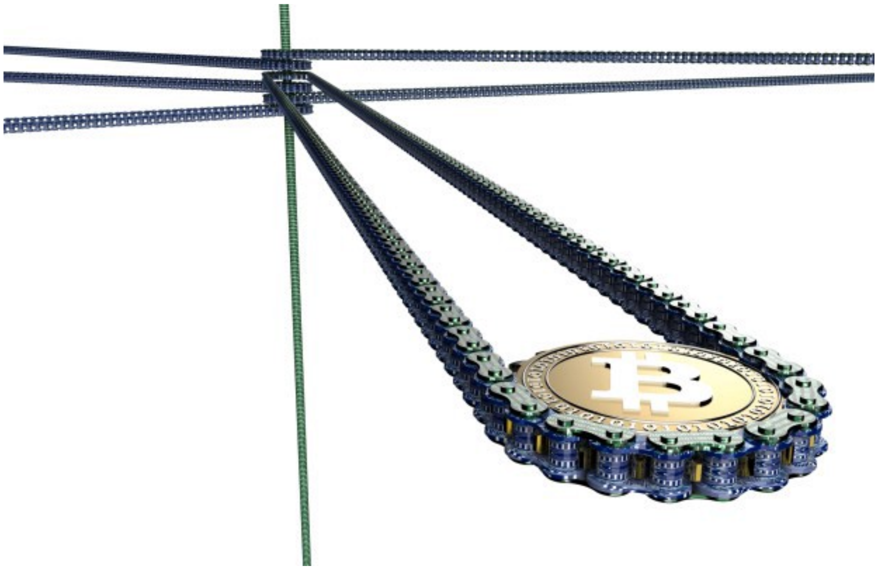
What's more, mining is a costly endeavor. Some estimates put the electricity consumption costs of the entire Bitcoin mining operation in the region of $500 million per year. In fact, one study has equated the entire power consumption of Bitcoin mining to Ireland's average electricity consumption. And that's just for Bitcoin—a whole heap of new cryptocurrencies have emerged that utilize some forms of PoW algorithm.

The hardware used in the mining process, moreover, is usually an advanced and expensive piece of proprietary kit. This has led to miners buying up this hardware, which in turn has spurred massive development on the manufacturing side to create the most advanced mining-specific hardware possible. Take the Application Specific Integrated Circuits (ASICs) as an example, which is a highly specialized piece that has been designed to exclusively mine Bitcoin and other cryptocurrencies. While ASICs significantly improve the efficiency of the mining process, the unique nature of such niche equipment makes it rather expensive. Most of the research and development of ASICs currently originates in China, a country where electricity is also comparatively cheap. So, given that Chinese ASICs manufacturers have considerable incentives to simply mine bitcoins for themselves—and indeed do so—it has led to an estimated 60–70% of Bitcoin's total hashing power congregating in this one country alone.

As such, the problem of excessive *centralization of power* emerges—something that is antithetical to the founding tenets of blockchain. There is little to prevent such closely neighboring miners from colluding in order to control a substantial and influential proportion of the blockchain's processing power, which means they could then nefariously write data onto the ledger that may deviate from the truth. In terms of our Byzantine Generals' analogy, this would be equivalent to a collusion among a faction of the generals.

Proof-of-Stake

The emergence of PoS in the form of the Peercoin project back in 2012 was an attempt to solve the aforementioned problems of cost, efficiency and the susceptibility towards centralization associated with *Proof-of-Work*. The basic premise is that instead of purchasing costly computer equipments to mine blocks ahead of one's competitors, each 'validating' node in the network purchases the coins used in the specific blockchain system.



Under the *PoS* algorithm, tokens are issued to the validating nodes in the network from the very beginning of the network's existence, which means that unlike *PoW*, tokens are not concurrently minted as new blocks are added to the ledger (although some blockchains employ a hybrid *PoW/PoS* algorithm which allows *PoW* minting to take place before switching to *PoS*). A specific node is then selected to commit the new block every few seconds or minutes. But if a node holds more coins, it retains greater power over what is considered to be the *truth* on the ledger. As such, the selection is strongly influenced by those that have the most coins—the more they have invested in the network, the more they have to lose in the event of any mishaps. Another influencing factor is the time period that coins have been held by users, which indicates whether they are invested for the long-term—clearly a more desirable position than someone who just purchased their coins yesterday.

So, those with more stake in the network are deemed to be more trustworthy, and considered less likely to attack the network. Indeed, in order to mount an attack, a user would have to buy 51% of the coin value of the entire network. This would be expensive, as well as nonsensical—there is no incentive for a user to attack a network in which he has so much invested. Attacking oneself is clearly an illogical action to pursue.

In addition, *PoS* is often preferred over *PoW* because it requires considerably less computational work, meaning that it is less laborious and less wasteful. And with less computations, therefore, the cost of executing *PoS* is substantially lower. This removes a major barrier to entry for those wishing to become block validators. The lack of mining requirements simply removes the need for specialist hardware to be developed; instead, the technology can easily run on consumer-grade computers.

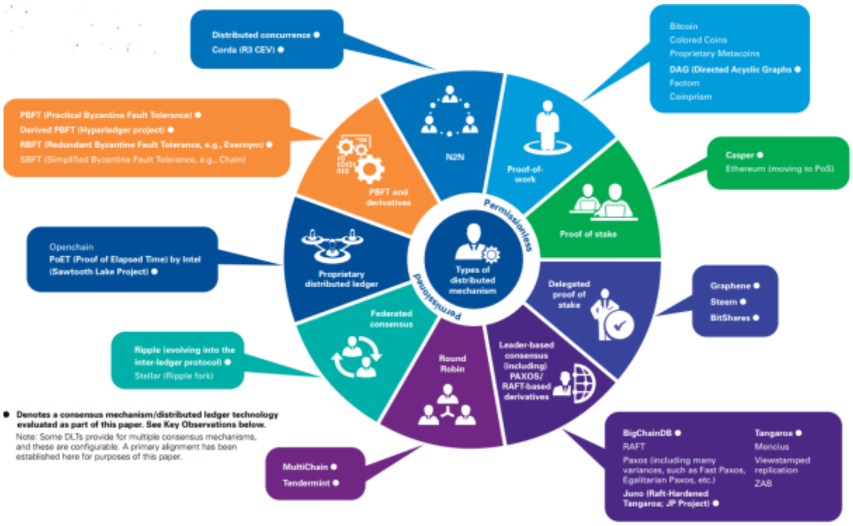
The problem of "Nothing at Stake"

One of the most cited problems with Proof-of-Stake, however, has come to be known as the *"Nothing-at-stake"* problem. Under *PoW* blockchains, there is an incentive to keep on mining the longest chain on the ledger, as this chain will be considered the primary version of the truth and the one that earns the miners their rewards; so miners are clearly incentivized to mine that one single chain.

But with *PoS*, there is little to prevent a miner from mining on numerous *PoS* chains, especially given the fact that there is negligible computational expense to the miner under this algorithm. Hypothetically speaking, therefore, a *PoS* miner operating on various chains could make it difficult for the network to reach consensus, while a bad actor could attempt to rewrite history.

This problem is particularly pertinent when a fork in the chain transpires (as happened with Ethereum, which then produced Ethereum Classic); in such a situation, validators using *PoS* could place stakes on both blockchains which makes it easier to alter the truth in their favor and earn profit. Indeed, partially as a way to solve this problem, Ethereum is in the process of making the transition from *PoW* to a new consensus strategy called Casper, which will employ a *PoS* algorithm. The move is intended to make the processing of new blocks onto the chain significantly more efficient, while a mechanism will be adopted to punish those participants who attempt to stake on two different chains. Further enhancements to the *PoS* model are now coming into play thick and fast among many of the biggest cryptocurrencies.

Among the most popular is the ***Delegated Proof of Stake*** (DPoS) algorithm which was championed by Dan Larimer, the driving force behind several high-profile blockchain projects including BitShares, Steemit and EOS. *DPoS* differs from *PoS* by having the validating process conducted by a group of delegates who are selected by the network users, and who have separate nodes for mining the blocks. Because these delegates are deemed to be trustworthy, this partially solves the nothing-at-stake problems. And the popular Chinese platform NEO employs an algorithm called ***Delegated Byzantine Fault Tolerance***, which employs a delegated voting process in order to prevent the chain from forking in two. Ripple meanwhile, utilizes its own "consensus ledger" which allows the participants and history to define truth in the system, as opposed to the technology. And many more consensus algorithms continue to surface, each solving problems unique to a particular application of blockchain:


Distributed concurrence ●
Corda (R3 CEV) ●
Bitcoin
Colored Coins
Proprietary Metacoins
DAG (Directed Acyclic Graphs ●
Factom
Coinprism
PBFT (Practical Byzantine Fault Tolerance) ●
Derived PBFT (Hyperledger project) ●
RBFT (Redundant Byzantine Fault Tolerance, e.g., Evernym) ●
SBFT (Simplified Byzantine Fault Tolerance, e.g., Chain)
Casper ●
Ethereum (moving to PoS)
N2N
Proof-of-work
PBFT and derivatives
Proof of stake
Permissionless
Types of distributed mechanism
Permissioned
Openchain
PoET (Proof of Elapsed Time) by Intel (Sawtooth Lake Project) ●
Proprietary distributed ledger
Delegated proof of stake
Graphene ●
Steem ●
BitShares ●
Federated consensus
Ripple (evolving into the inter-ledger protocol) ●
Stellar (Ripple fork)
Round Robin
Leader-based consensus (including) PAXOS/ RAFT-based derivatives
● Denotes a consensus mechanism/distributed ledger technology evaluated as part of this paper. See Key Observations below.
Note: Some DLTs provide for multiple consensus mechanisms, and these are configurable. A primary alignment has been established here for purposes of this paper.
MultiChain ●
Tendermint ●
BigChainDB ●
RAFT
Paxos (including many variances, such as Fast Paxos, Egalitarian Paxos, etc.)
Juno (Raft-Hardened Tangaroa; JP Project) ●
Tangaroa ●
Mencius
Viewstamped replication
ZAB

Overview of Distributed Consensus Mechanisms (Source)

Clearly, the search for the ideal consensus mechanism remains an unfinished business at this stage. One expects that it will be an ongoing process to improve such factors as cost, efficiency and scalability, which will lead to some intriguing advancements during the coming months and years. Achieving a balance between the trade-off of decentralization, processing speed and efficiency will likely remain at the heart of further development. And as more use cases for blockchain continue to branch out, it seems highly probably that the range of consensus mechanisms will grow and evolve in tandem. This will also likely spark further novel applications of blockchain.

Comments

. . .

Originally published at www.datadriveninvestor.com on November 12, 2017.