Longest Chain (LC) Consensus

Blockchain @ uninsubria

Roberto Vicario

2024/2025

Contents

1 Longest Chain (LC)

…

> **Problem:** *Nodes may disagree on the current state because they have seen different versions of the chain. Is there a way to randomly sample the leader from an unknown set of participants?*

1.1 Sybil Attacks

A *Sybil Attack* occurs when a single entity creates multiple identities to gain disproportionate influence in a network.

> **Theorem:** Given a network of n nodes, each with a distinct hashrate $\mu_1, \mu_2, \dots, \mu_n$. In each round of leader selection, the probability that node i is chosen as the leader is proportional to its hashrate and is given by:

$$\frac{\mu_i}{\sum_{j=1}^{n} \mu_j}$$

To prevent Sybil attacks, the system must ensure that creating multiple identities is costly or requires a significant investment of resources. For this reason, were implemented two main mechanisms: *Proof of Work (PoW)* and *Proof of Stake (PoS)*.

1.1.1 Proof of Stake (PoS)

The chance of being chosen to propose or validate a block generally depends on the amount committed. This approach helps limit the influence of any single participant and discourages the creation of many identities. PoS can be integrated into consensus mechanisms in these ways:

- **PoS + BFT:** The quorum is easily achieved by selecting the nodes with the highest stake.
- **PoS + LC:** The longest chain selects the leader by the depth of the chain, which is proportional to the stake held by the nodes.

1.1.2 Proof of Work (PoW)

In this mechanism, the nodes called miners, compete to solve a cryptographic *Hard Puzzle* by finding a nonce that, when combined with the block's data and hashed, produces a hash value below a specified target:

$PROOF$

Given a cryptographic hash function H, find an input x such that:

$$H(x) \leq \tau$$

where τ is the current difficulty target.

Since H behaves like a random function, the only viable strategy is brute-force search. The difficulty of the puzzle is adjusted by changing τ:

- Smaller $\tau \Rightarrow$ fewer hash outputs satisfy the condition $\Rightarrow$ harder puzzle
- Larger $\tau \Rightarrow$ more outputs satisfy the condition $\Rightarrow$ easier puzzle

PoW can be integrated into consensus mechanisms in these ways:

- **PoW + BFT:** Integrating PoW with BFT consensus can introduce instability. Fluctuations in the network's total computational power may disrupt predictable leader selection, undermining the reliability and security guarantees of BFT protocols. As a result, combining PoW with BFT is generally discouraged.
- **PoW + LC:** Nodes compete to solve computational puzzles, and the chain with the most accumulated proof of work is considered the valid one. This combination forms the basis of *Nakamoto Consensus*.

2 Nakamoto Consensus

> **Partially Synchronous**

Nakamoto Consensus is a consensus mechanism designed for PoW-based blockchains. It elects a leader in each round based on the computational effort expended by nodes, allowing them to agree on a single chain of blocks.

$ALGORITHM$

1. **Puzzle Solving:** Each node attempts to solve a cryptographic puzzle.
2. **Leader Election:** The first node to find such x becomes the leader and broadcasts the block as its proposal for the next block in the chain.
3. **Chain Extension:** Honest nodes always extend the chain with the highest total work, the one requiring the most cumulative PoW effort.

4. **Difficulty Adjustment:** The protocol adjusts τ over time to maintain a stable block production rate and reduce accidental forks. A typical target is one block every fixed time interval.