Foundations of Blockchain

Trust and Consensus

Matteo Nardelli

October, 2023

Mining

Bitcoin: Mining

- *Mining* is the mechanism by which transactions are validated and cleared;
- Miners validate new transactions and record them on the global ledger;
- A new block, containing transactions that occurred since the last block, is "mined" every 10 minutes on average;
- Transactions added to the blockchain are considered "confirmed";
- Miners receive two types of rewards:
  - new coins created with each new block;
  - transaction fees from all the transactions included in the block.
- The process is called mining because the new coin generation is designed to simulate diminishing returns;

Bitcoin: Consensus through Proof of Work

- Satoshi Nakamoto's main invention is the decentralized mechanism for emergent consensus;
    - Emergent: no explicitly achieved consensus among nodes; but emerges as nodes follow simple rules.
- Four processes that occur independently on nodes across the network:
    - Independent verification of each transaction by every full node;
    - Independent aggregation of those transactions into new blocks by mining nodes, coupled with demonstrated computation through a Proof-of-Work algorithm;
    - Independent verification of the new blocks by every node;
    - Independent selection, by every node, of the chain with the most cumulative computation demonstrated through Proof-of-Work;

Bitcoin: Aggregating Transactions into Blocks

- After validating transactions, a bitcoin node will add them to the memory pool (or, transaction pool);
- Miner node:
    - Each miner is listening for transactions, trying to mine a new block and also listening for blocks discovered by other nodes.
    - As soon as a block is added to the blockchain, a miner will start working on the next block:
    - The miner aggregates all the transactions from the memory pool into a candidate block;
- The first transaction in any block is a special transaction, called a coinbase transaction.

Bitcoin: Coinbase Transaction

- This transaction is constructed by the miner;
- It contains his reward for the mining effort;
- The total amount of reward is the sum of:
  - The coinbase reward (new bitcoin — 50 bitcoin, halved every 210,000 blocks; currently, 6.25 BTC);
  - the transaction fees from all the transactions included in the block.
- Unlike regular transactions, the coinbase transaction does not consume (spend) UTXO as inputs.

Consensus Mechanisms

Decentralization Requires Consensus Mechanisms

- The lack of a central authority is one of the main attractions of blockchains;
    - Censorship resist, lack of permission to access information.
- Without a trusted arbitrator, consensus algorithms are the mechanism used to arrive at a common state, while maintaining decentralization.
    - Consensus algorithms ensure safety and liveness of the blockchain;
    - **Safety**: basically, bad things do not happen;
    - **Liveness**: basically, good things do happen;
- A consensus mechanism requires fault-tolerance:
    - It should continue to work even in presence of faults;
    - **Crash fault-tolerance** (CFT): can tolerate only crash (benign) faults (e.g., Paxos, RAFT);
    - **Byzantine fault-tolerance** (BFT): can tolerate arbitrary (even malicious) behaviors (e.g., PBFT).

Safety and Liveness

- **Safety**: nothing bad happens. It encompasses three properties:
    - **Agreement**: The agreement process requires that no two processes decide on different values;
    - **Validity**: If a process has decided a value, that value muse have been proposed by a process.
    - **Integrity**: A process must decide only once.
- **Liveness**: something good *eventually* happens.
    - Each honest node must eventually decide on a value.

Fault-tolerance

- **Redundancy** is the main technique to mask errors;
- **Replication** (i.e., physical redundancy) allows to build fault-tolerant systems;
  - If some of the nodes become faulty, the overall system remains available due to the data being available on multiple nodes.
  - Active replication: a state update is performed by all the replicas;
  - Passive replication: there is only one server (called primary) that processes the state update request. After processing, the primary updates the state on the other (backup) replicas.
- Minimum number of processors needed to solve consensus
  - CFT: With F faults, at least $2F + 1$ nodes are required;
  - BFT: With F Byzantine faults, at least $3F + 1$ nodes are required.

Timing Assumptions

- **Synchrony**: there is a known upper bound on communication and processor delays;
- **Asynchrony**: No upper bound on delays; algorithms are designed to run without any timing assumptions;
    - This scenario is common in large-scale geographically distributed systems.
- **Partial synchrony**: the upper bound on delays exists, but it is not known. It means that the system becomes synchronous after an instance of time called global stabilization time (GST).

FLP impossibility

- In an *asynchronous* environment, the *deterministic* consensus is impossible, even when only one process is faulty.
    - Fischer, Lynch, Paterson, “Impossibility of distributed consensus with one faulty process”. Journal of the ACM, 32.2 (1985);
    - Meaning: a deterministic consensus algorithm cannot satisfy agreement, validity, termination, and fault tolerance in an asynchronous system.
- Some techniques have been proposed to *circumvent* the FLP impossibility:
    - **Failure detectors**;
    - **Randomized algorithms**: provide probabilistic termination guarantee;
    - **Synchrony assumptions**: additional synchrony and timing assumptions ensure progress and termination.

Consensus

- Additional properties of consensus protocols in blockchains:
    - Scalability: its efficiency as the system scale (and workload) increases.
        - Metrics: transaction throughput and transaction confirmation latency.
    - Decentralization: helps to avoid corruption and collusion, and to build a fairer system.
- Two broad categories:
    - **Voting-based** consensus (or, committee-based): traditional research results from distributed systems community (e.g., Paxos, PBFT);
        - Despite significant progress in the research of distributed consensus algorithms, the design of a secure and efficient BFT blockchain consensus protocol *remains a critical and challenging task*;
    - **Lottery-based** consensus (or, proof-based): firstly introduced with Bitcoin (Nakamoto consensus);
- A systemization of knowledge on consensus in blockchain [GK20].

Finality

- Definition: If a transaction has been committed to the blockchain, it will not be revoked or rolled back.
- Two types of finality:
  - Probabilistic: the property that a committed transaction cannot be rolled back builds over time;
    - As the chain grows, the block containing the transaction goes deeper, which increasingly ensures that the transaction will not be rolled back;
    - This approach is quite slow (e.g., 6 blocks for Bitcoin—an hour);
    - This delay is not acceptable in permissioned blockchain.

Finality

- Definition: If a transaction has been committed to the blockchain, it will not be revoked or rolled back.
- Two types of finality:
  - Probabilistic: the property that a committed transaction cannot be rolled back builds over time;
    - As the chain grows, the block containing the transaction goes deeper, which increasingly ensures that the transaction will not be rolled back;
    - This approach is quite slow (e.g., 6 blocks for Bitcoin—an hour);
    - This delay is not acceptable in permissioned blockchain.
  - Deterministic: absolute finality guarantee for a transaction as soon as it is committed in a block.
    - No forks or rollbacks;
    - Finality provided by committee-based algorithms (e.g., PBFT).

State Machine Replication (SMR)

- Blockchains need to solve the problem of State Machine Replication (SMR):
    - Clients send a sequence of transactions;
    - Miners have to agree on an order in which to implement those transactions;
- SMR protocols usually assume that a Public Key Infrastructure exists;
- Participants can be uniquely identified;
- Proof-based consensus mechanisms aims to achieve consensus without explicit knowing who is providing the proof.
- Blockchains fostered the development of consensus mechanisms and protocols.

Consensus Mechanisms

Practical Byzantine Fault Tolerant

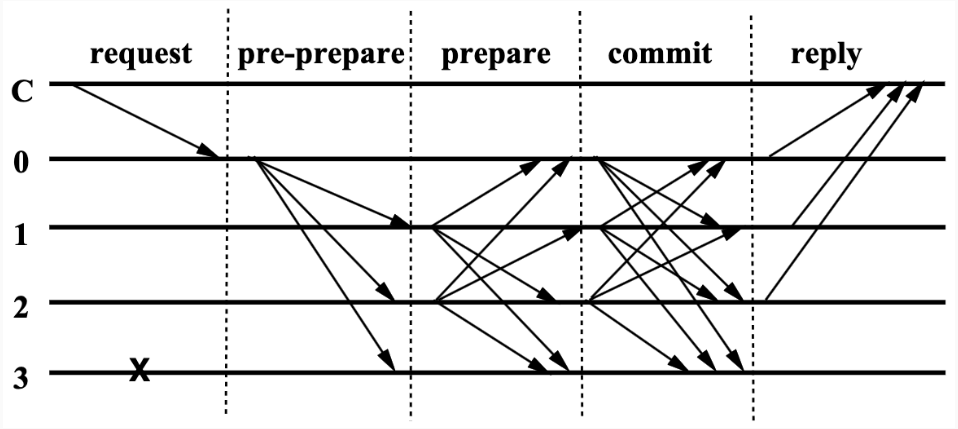
Practical Byzantine Fault Tolerant (PBFT)

- Developed in 1999 by Castro and Liskov [CL+99];
- Ensures fault-tolerance to F Byzantine faults, with N nodes: $N \geq 3F + 1$;
- Two roles: leader, follower;
- Three sub-protocols:
  - Normal operation: executed when everything is running normally and no errors are in the system;
  - View change: executed when a faulty leader node is detected in the system;
  - Checkpointing: Used to discard the old data from the system.

PBFT: Normal Operation

- The protocol runs in rounds;
- In each round, an elected leader node handles the communication with the client; Participants are called replicas;
- Each replica maintains a local state:
    - Service state;
    - A message log;
    - A number representing that replica’s current view.
        - View updated on view-change, i.e., when a leader suspected to be faulty is oust.
- The protocol progresses through three phases: pre-prepare, prepare, and commit.

PBFT: Normal Operation



PBFT: Normal Operation

- Pre-prepare: informs replicas of the update request;
    - The primary receives a request from the client;
    - It assigns a sequence number and sends the **pre-prepare** message to all replicas;
    - The replica checks the message, accepts it by updating its local store, and send a *prepare* message.
- Prepare: replicas are ready to execute the request;
    - Each replica waits for at least $2F + 1$ valid prepare;
    - A prepare is valid if it contains a valid view, sequence number, and message digest;
    - Each replica updates its local state and sends a *commit* message;

PBFT: Normal Operation

- Commit: replicas execute the request;
  - Each replica waits for at least $2F + 1$ valid commit;
  - A prepare is valid if it contains a valid view, sequence number, and message digest;
  - Each replica executes the request;

View-change

- Executed when a primary is suspected faulty;
- This sub-protocol ensures protocol progress;
- View change allows replicas to select new primary (and update the view number);
- Triggered when no progress within a time-frame for a pre-prepare message;

Practical Byzantine Fault Tolerant

- Pros:
    - Immediate and deterministic transaction finality;
    - Energy efficient compared to proof of work;
- Cons:
    - Not very scalable;
    - Sybil attacks can be carried out on a PBFT network:
        - A single entity can control many identities to influence voting.

Istanbul Byzantine Fault Tolerant

- PBFT is not blockchain-specific; Istanbul BFT is a protocol specifically tailored for blockchains [Mon20];
- In Istanbul, two types of nodes: nodes and validators (participating to consensus);
- Very similar to PBFT, but different view change mechanism;
- In each phase (pre-prepare, prepare, commit), if a timeout expires or no majority reached, a *round change* process starts;
- Process change: Validators need to wait for $2F + 1$ round change messages to arrive for the newly proposed round number.

Tendermint

- Tendermint is another variant of PBFT [Kwo14];
- Tendermint works similarly to PBFT: three phases are required to achieve a decision;
- New termination mechanism: Instead of having two sub-protocols for normal and view-change mode, Tendermint terminates without any additional communication costs;
    - On timer expiration, a pre-vote/pre-commit *nil* is propagated;
- A round is complete, a new round starts with three phases and terminates when a decision is reached.

Consensus Mechanisms

Proof of Work

Proof of Work

- Permissionless blockchains are vulnerable to Sybil attacks
    - An adversary may pretend to be multiple nodes simultaneously to take advantage in the leader election.
- Nakamoto consensus (i.e., proof-of-work) is resistant to Sybil attacks;
- Proof-of-work (PoW) consists in solving a cryptographic puzzle:
    - Solving the puzzle requires huge computing power (Sybil attack becomes inconvenient);
    - Verifying the puzzle solution is easy;
    - It is based on computing hash values;
    - Incentive: newly minted currency;
    - Punishment: the cost of energy required to participate in mining;
- Ethash, the Ethereum's PoW algorithm, is slightly different from the PoW by Bitcoin.

Proof of Work

- The puzzle is to compute a **nonce** that satisfy the condition:

$$H(h_{i-1}, nonce, tx, h) < \text{Target}$$

  where:
  - h_{i-1} is the previous block hash;
  - tx is the set of validated transactions;
  - h other block header information;
  - Target indicates the puzzle difficulty; periodically adjusted to preserve the expected block generation intervals around ten minutes;
    - Expressed as the number of leading 0 bit;
    - Increasing the difficulty by 1 bit causes a doubling in the time it takes to find a solution.
- If the hash is not less than the target, the miner will modify the nonce (usually just incrementing it by one) and try again.

Proof of Work: Target Representation in Bitcoin

- The block contains the target in a “target bits” notation:
  - the first two hexadecimal digits represents the exponent;
  - the next six hex digits are the coefficient.
- Example: *`0x1903a30c`*;
  - target = coefficient $\cdot 2^{8\cdot(\text{exponent}-3)}$ = 0x03a30c $\cdot 2^{0x08(0x19-0x03)}$ = $238\,348 \cdot 2^{176}$
  - target = 0x0000000000000003A30C00000000000000000000000000000000000000000000

Selecting Chains of Blocks

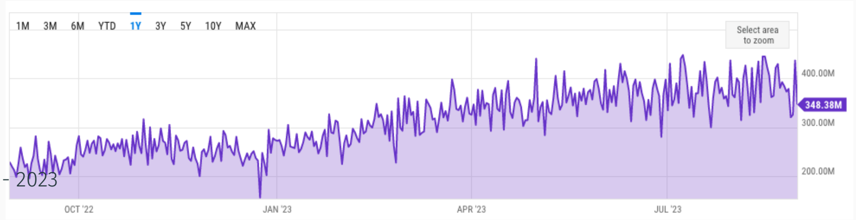
- The “main chain” is the **valid** chain with the **most cumulative PoW associated**;
    - By selecting the greatest-cumulative-work valid chain, all nodes **eventually achieve** network-wide consensus;
- Temporary discrepancies are resolved as more work is added;
    - Sometimes, a new block may happen to extend a chain that is not the main chain, but a secondary chain;
    - If the secondary chain has more cumulative work than the main chain, the node will select the secondary chain as its new main chain;
- Orphan block: If a valid block is received and no parent is found in the existing chains;
    - e.g., two blocks mined within a short time frame;
    - It is saved in the orphan block pool;
    - It will be linked into the chain, once the parent is received and linked.

Proof of Work

- Pros:
    - Scalable;
    - Resistant to Sybil attack;
- Cons:
    - Probabilistic transaction finality;
    - Not energy efficient (by definition).

The 51% Attack

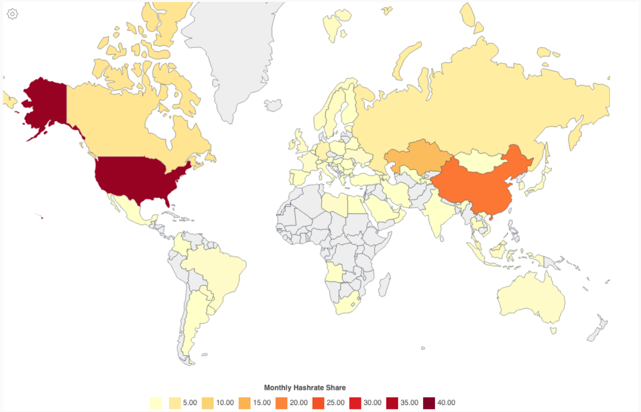
- The confirmation of transactions depends only by the consensus.
  - The total computational power of a decentralized PoW system is the sum of the computational power of the nodes.
  - Larger computational power increases the chance to win the mining reward for each new block mined;
  - This creates an incentive to accumulate clusters of mining nodes (named *mining pools*);
- Any mining pool that achieves 51% hashing power can effectively force its version of events (e.g., including alternative and double transactions).



Hitting the 51% of Hash Rate

Notable event:

- In 2014, the Ghash.io pool obtained 51% hashing power in Bitcoin. The pool voluntarily capped their hashing power at 39.99% to restore trust in the network.



Source

Consensus Mechanisms

Proof of Stake

Proof of Stake (PoS)

- The stake represents the number of coins in the consensus protocol staked by a participant;
    - If someone has a stake in the system, then they will not try to sabotage the system.
- The blockchain keeps track of a set of validators;
    - Anyone who holds the blockchain’s cryptocurrency can become a validator by sending a special type of transaction that locks up funds into a deposit;
- The validators take turns proposing and voting on the next valid block;
- The weight of each validator’s vote depends on the size of its stake;
    - A validator risks losing their deposit if the block they staked it on is rejected by the majority of validators.
    - Conversely, validators earn a small reward, proportional to their deposited stake, for every block that is accepted by the majority.

Variants of PoS

Different types of PoS:

- Chain-based: a modification of PoW, where difficulty depends on the validator's stake;
- Committee-based: involves the election of a committee of validators using verifiable random function (VRF) with probabilities of being elected higher with higher stake;
    - VRF is a public-key pseudorandom function that provides proofs that its outputs were calculated correctly.
- Delegated PoS: two-stage process;
    - Stakeholders elect a validation committee;
    - The committee proposes blocks and achieves consensus using BFT-like algorithms.

Ethereum’s PoS

- Ethereum switched on its proof-of-stake mechanism in 2022;
    - more secure, less energy-intensive, and better for implementing new scaling solutions compared to the previous PoW;
- Key ideas:
    - Validators explicitly stake capital (32 ETH) into a smart contract;
    - The validator is responsible for **checking** that new blocks are valid;
    - Occasionally, the validator creates new blocks;
    - If validators misbehave, some or all of their staked capital can be destroyed.

Ethereum’s PoS

- On depositing the stake, validators join an activation queue;
- Once activated, they receive new blocks from the network:
    - Transactions are validated and re-executed, changes to Ethereum’s state and the block signature are checked;
    - The validator then sends a **vote** in favor of that block across the network;
- Time is divided into slots (12 seconds) and epochs (32 slots).
    - In every slot, one validator is randomly selected to be a block proposer;
        - RANDAO, a DApp for random number generator, is used;
    - In every slot, a committee of validators is randomly chosen: their votes determine the block validity;
    - Every active validator attests in every epoch, **but not in every slot**.
    - Checkpoints occur at the start of each epoch.

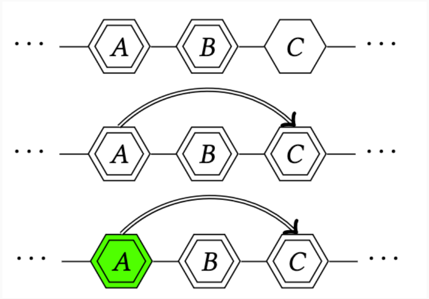
Ethereum’s PoS: Checkpointing

- Checkpoints exist because only a subset of active validators attest in each slot, but all active validators attest across each epoch;
- A checkpoint is a pair block-epoch (b, e) where b is the block of the first slot of epoch e;
- Two types of votes: the block vote and the checkpoint vote:
    - The block vote is used to determine the candidate chain;
    - The checkpoint vote is used to finalize blocks to grow the chain;
- The finality gadget is the mechanism that aims at finalizing blocks;
    - It works at the level of epochs;
    - It grows the finalized chain disregarding the block production;
    - This decoupling permits block availability even when the finalizing process is slowed down.

Ethereum’s PoS: Finality Gadget

- Finalization is achieved through justification, which occurs thanks to checkpoint votes:
    - A checkpoint vote contains a pair of checkpoints: the checkpoint **source** and the checkpoint **target**.
    - If validators controlling more than **two-thirds** of the stake make the same checkpoint vote, then we say there is a **supermajority link** from the checkpoint source to the checkpoint target.
    - The checkpoint target of a supermajority link is said to be justified.
- A checkpoint can be finalized at least in two epochs;
    - To become finalized, a checkpoint needs to be the source of a supermajority link between **justified** checkpoints;
    - Once **finalized**, all the blocks leading to it become finalized;

Ethereum’s PoS: Finality Gadget



- Hexagons represent checkpoints;
- A justified checkpoint has a double hexagon;
- A finalized checkpoint is a double hexagon colored in green;
- The arrow between two checkpoints indicates a supermajority link;
- Formal analysis of Ethereum PoS [PAGTP22]

Attacks to PoS: Nothing at Stake

- Nothing at stake: a **theoretical** security hole in PoS systems;
- Anytime there is a fork, it is in the best self-interest of all of the miners to continue mining both chains;
    - There is no cost to mining;
    - Mining all of the forks ensures that the miner will get their reward no matter which fork wins.
- Double-spend attacks is more feasible.
- Not yet occurred in the real world;

How addressed in Ethereum? Casper relies on security-deposit:

- Who wish to validate transactions must place a security deposit;
- On validator misbehaving, a portion or all of its deposit is forfeited;
- As well as their ability to continue participating in consensus.

Attacks to PoS: Long Range Attack

- Attack:
  - An attacker goes back to the genesis block and forks the blockchain;
  - The new branch will be populated with a completely (or partially) different history of the main chain;
  - Once the newly crafted branch becomes longer than the main chain, it will overtake it.
- Three main attack instances to produce blocks faster than the main chain:
  1. **Simple**: attacker can forge timestamps ahead of time. This does not work if nodes check the block timestamp;
  2. **Posterior Corruption**: attacker uses the private key of a *retired* validator (who used to sign blocks, but currently no stake locked) to rewrite previous blocks;
  3. **Stake Bleeding** [GKR18]: every time the attacker is elected as a validator in the main-chain, he skips her turn, stalling the main chain, while growing an alternative branch.

Mitigations to Long Range Attack

- **Moving Checkpoints**: checkpoints imposes that only a small set of the latest blocks can be reorganized. Almost all PoS protocols use check-pointing.
- **Key-Evolving Cryptography**: a slot leader signs a block and immediately destroys the used key, without recovery capability. *Experimental idea.*
- **Context-Aware Transactions**: Include the hash of a previous block inside a transaction, to avoid adversaries copying transactions from the main chain.
    - This does not eliminate the attack, but introduces an obstacle.
- **Plenitude Rule**: Analyze the frequency of slot leaders with respect to their stake. If a leader is much more frequent than its stake, possible malicious actions have been performed.

Rewriting history: a brief introduction to Long Range attacks.

Proof of Stake

- Pros:
    - Scalable;
    - Ease participation (w.r.t. PoW);
    - Resistant to Sybil attack;
    - Energy efficient compared to proof of work;
- Cons:
    - Probabilistic transaction finality;
    - Harder to implement (w.r.t. PoW);
    - Leads to centralization as it favours users who have a large amount of cryptocurrency.

References (1)

- Miguel Castro, Barbara Liskov, et al., *Practical byzantine fault tolerance*, OsDI, vol. 99, 1999, pp. 173–186.
- Juan Garay and Aggelos Kiayias, *SoK: A consensus taxonomy in the blockchain era*, Topics in Cryptology–CT-RSA 2020, Springer, 2020, pp. 284–318.
- Peter Gaži, Aggelos Kiayias, and Alexander Russell, *Stake-bleeding attacks on proof-of-stake blockchains*, 2018 Crypto Valley conference on Blockchain technology (CVCBT), IEEE, 2018, pp. 85–92.
- Jae Kwon, *Tendermint: Consensus without mining*, Draft v. 0.6, fall **1** (2014), no. 11, 1–11.

References (2)

Henrique Moniz, *The istanbul bft consensus algorithm*, arXiv preprint arXiv:2002.03613 (2020).

Ulysse Pavloff, Yackolley Amoussou-Guenou, and Sara Tucci-Piergiovanni, *Ethereum Proof-of-Stake under Scrutiny (Extended Version)*, Technical report, CEA DILS, October 2022.

Matteo Nardelli