CSE 2550/5000: Blockchain Technology

Lecture 8
Mining and Consensus

Ghada Almashaqbeh
UConn - Spring 2026

Outline

- Mining and consensus algorithms/protocols.
  - Proof-of-stake.
  - Proof-of-space/storage.
  - Byzantine fault tolerant.
  - Hybrid mining algorithms.

Why Proof-of-Work?

- Defending against Sybil attacks.
  - Creating fake identities is expensive; fake miners with no resources (computation, bandwidth, etc.) cannot participate in adding blocks to the blockchain.
- Securing the blockchain.
  - Expensive to rewrite or alter the history.
- Providing a natural way of distributing block generation among miners in a random way.
  - I.e., selecting the round leader in a randomized way.
- Implicitly synchronizing network operation.
  - The difficulty of the mining puzzle controls the average block generation rate.

PoW is Wasteful/Unuseful

- Miners perform a repeated hashing process that is not useful for anything beyond mining a new block.
- Is not this a ***computation waste*** (which is translated into resource waste)?
  - Researchers showed that ***electricity consumption*** of Bitcoin mining is comparable to some countries' consumptions.
  - Mining is not for free: requires advanced hardware, cooling systems, huge electricity bills, maintenance cost, etc.
- Can we use other types of resources (storage, bandwidth, etc.) to establish Sybil resistance identities and so:
  - Reduce waste of mining?
  - Or even have a useful mining process?

PoW - More Issues

- How about ***transaction throughput***?
  - Can cryptocurrencies replace other, high throughput, payment systems anytime soon?
- How long does it take to ***confirm a transaction***?
  - The recent history of the blockchain may have several, temporary versions. Several blocks could be mined at the same time.
  - How about applications that require instant settlement, can they afford waiting for an hour or so for a transaction to be confirmed?
- How about ***wealth distribution***? Does the mining process make the wealthy wealthier?
- Not to mention the ***tendency toward centralization*** due to the concept of centralized mining pools.

Potential Solutions

- Several mining algorithms were proposed to optimize the following:
  - Resource consumption, e.g., proof-of-stake.
  - Usefulness, e.g., proof-of-storage.
  - Throughput and confirmation time, e.g., Fault Tolerant Byzantine Agreement based protocols.
- Some systems adopted hybrid solutions that combine several protocols together.

Proof of Stake (PoS)

Proof-of-Stake (PoS)

- Goal: reduce resource consumption.
  - Leader election is based on the amount of currency, or stake, a miner owns.
  - The leader is elected first, then mining takes place (opposite order compared to PoW).
    - No computationally-extensive puzzle solving is needed to mine a block, which saves both hardware cost and electricity.

PoS - Basic Mechanism

- At the beginning of each round (i.e., period during which a new block will be mined):
  - Define the set of miners and the stake share of each one of them.
  - Randomly select a miner to be the round leader.
    - Each miner will be selected with a probability proportional to the amount of currency it stakes in the system.
  - Mine a block by forming a candidate block (containing a set of valid transactions), signing this block, and then announcing it to the network.
- Other miners accept the block if it is valid and created by an elected leader (requires a proof of leader election).

PoS - Leader Election I

- Through a randomized, unpredictable process, which requires a cryptographic lottery.
- The i^{th} miner will be selected with a probability $p_i = s_i / \sum s_i$ for all i, where s_i is the stake of the i^{th} miner.
- Several lottery implementations, an example:
  - Verifiable random function (VRF)-based, used in Algorand [Gilad et al., 2017].
    - A random public seed is selected per round, used as input for the VRF; if the output is larger than a threshold, then the miner is elected a block proposer.
    - The VRF requires a secret key; hence, no one can produce the output other than the key owner.
      - The output also includes a proof that the VRF output is correct.

PoS - Leader Election II

- Depending on the protocol, a set of block proposers (potential leaders) is selected, then a consensus protocol is run to select one block as winner (and a winner leader).
  - In Algorand (based on the academic paper), a miner's VRF output is tested against another threshold to determine if it is a voter. A block with the highest number of votes will be selected as the winner to extend the blockchain.
- Secure single leader election is an active research area
  - It simplifies protocol design,
  - increases throughput,
  - and reduces blockchain forks.

PoS - Issues I

- Initial stake distribution
  - How to distribute the currency among the miners to have stake and participate in mining?
    - Several options, starts with PoW and then switch to pure PoS. Or have a stake allocation phase during which miners can buy coins.
- Targeted attacks and denial of service (DoS)
  - If leader election is public, attackers may attack the leader to prevent mining a new block.
    - Potential solutions: implement a private leader election process, leader is know after announcing a block, and/or elect several leaders per round. (Algorand pioneered this direction.)

PoS - Issues II

- Nothing-at-stake attack.
  - A miner, once selected as the round leader, may extend several forks at the same time.
    - Mining a block on each branch requires only a signature!
  - Even worse, the leaders of the past rounds may collude to rewrite the blocks they mined as they want.
  - Some proposed solutions:
    - Financial punishments (the miner who is detected doing this attack will lose its stake).
    - Checkpoints to prevent rewriting the chain by colluding miners.

PoS - Issues III

- Wealth distribution.
  - The miner with the highest stake will be selected more frequently to mine new blocks, and hence, collect mining rewards.
    - The wealthy becomes wealthier!
  - This makes 51% attack easier.
  - Potential solutions:
    - select an appropriate mining reward function to smooth out wealth distribution,
    - develop leader election algorithms that exclude recently elected miners, etc.

PoS in Ethereum I

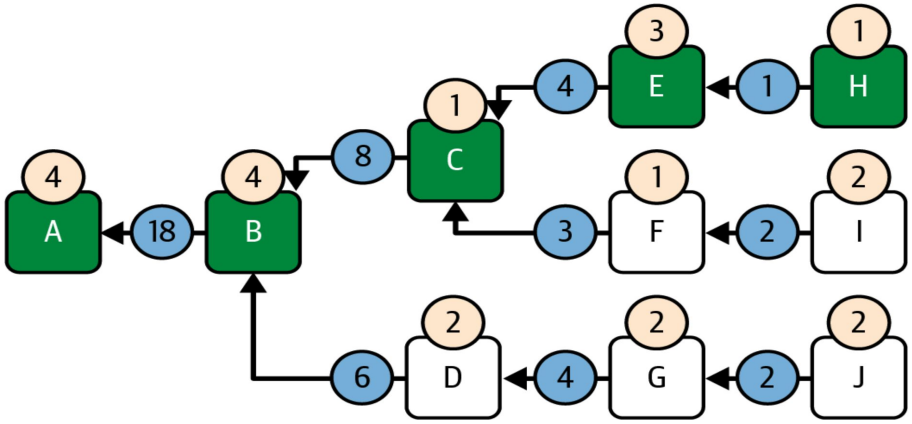
- Gasper: Combines the fork choice rule (LMD-GHOST) and finality gadget (Casper FFG).
- Anyone who stakes 32 ETH can become a validator (aka miner).
  - Validators propose and vote on blocks.
  - Following the protocol leads to staking rewards, while misbehaviors leads to stake slashing as a penalty.
- The set of validators is known.
- The system operates in slots (each slot is 12 sec) and epochs (each epoch is 32 slots).
- Using RANDAO, for each epoch a committee (of at least 128 validators) is elected, and a single proposer from that committee per slot is elected.
  - Results are known at the beginning of an epoch.

PoS in Ethereum II

- One validator proposes a block in a slot.
- However, due to propagation delays, the view of mined blocks could be different.
- Gasper aims to address this issue based on block voting, i.e., BFT agreement.

PoS in Ethereum - LMD-GHOST

- LMD-GHOST stands for latest message driven and greediest heaviest observed subtree.
- Each validator votes on the blockchain head based on its local view, and the score (total votes) of a block is propagated to its parents in the branch.
- The subtree with the heaviest votes is selected.
- If a validator votes on two different heads, the latest vote only is counted.



*From chapter 15, Mastering Ethereum

PoS in Ethereum - Casper FFG

- Finality means that once a transaction is final, it is permanent, i.e., it cannot be removed from the blockchain (so it is strongly confirmed).
- Casper FFG simply follows a BFT style for finality—once ⅔ of the validator committee votes on a block, it is considered final.
- Validators vote once per epoch, happening during the first slot of an epoch, which is called a checkpoint.

Useful Mining

Useful Mining

- Many flavors, with the goal of building a mining process with a useful outcome.
  - So the mining not only secures the system but also offers a service.
- Usually relies on utilizing the miners to provide a distributed service.
  - Such as storage service of archival data, content distribution, computation outsourcing, etc.
- The probability of selecting a miner as a round leader is tied to the amount of service a miner puts in the system.
- Several challenges:
  - How to prove that a miner provided a correct service?
    - Requires deploying additional protocols to produce such proofs.
  - How to use this knowledge to select the round leader?
    - Similar approaches to proof-of-stake can be used.

Proof-of-Space/Storage

- Miners store files for others, prove periodically that they still hold the file.
  - Examples: Spacemint, Spacemesh, Filecoin, Storj, PermaCoin.
- The larger the dedicated storage space, the higher the probability of being selected as a leader.
- Usually create a storage market; beside collecting mining rewards, miners are paid for the storage by the customers.

Proof-of-Storage Issues I

- Cryptographic proofs for storing files (checking the references is optional):
  - proof-of-space [Dziembowski et al., 2015],
  - proof-of-spacetime [Moran et al., 2016],
  - proof-of-retrievability [Miller et al., 2014].
- Mainly takes the form of a challenge/response approach, which needs to be implemented in a non-interactive way.
- Usually a miner will put in some stake, like a penalty deposit, in order to participate.
  - If proofs are not submitted, part of this deposit is revoked, this besides not being paid by the customer (if such payments are involved).
    - How to determine the value of the financial punishment?

Proof-of-Storage Issues II

- Several issues:
  - Trade-off between computation/storage [Moran et al., 2016].
    - Either generate a file on the fly or have it already stored.
    - The construction is about a randomly generated file; is this particularly useful?
  - Outsourcing; store files somewhere else and retrieve when needed.
    - Adding timing bound on a miner's response could be useful in this case.
  - Claim to store several copies of a file.
    - For redundancy reasons, one may ask for storing several copies of a file.
    - Proof-of-replication (a modified version of proof-of-storage) is used to mitigate this issue, e.g., used in Filecoin.

Byzantine Fault Tolerant (BFT)-based

Byzantine Agreement Based I

- Or Byzantine Fault Tolerant (BFT)-based consensus.
- Goal: “Agree faster.”
  - Speeds up transaction confirmation, increases throughput, and reduces the probability of forking the blockchain.
- Based on the classic Byzantine general problem in distributed systems.
  - The failure of one or more components prevents the system from reaching consensus.
- It was shown that a system of $3t+1$ parties can tolerate up to t failures, and hence, reach consensus.
- The Practical Byzantine Fault Tolerance (PBFT) algorithm [Castro et al., 1999] was the first efficient solution that works in weakly synchronous environments such as the Internet.

Byzantine Agreement Based II

- For each round, a committee will be elected to decide the next mined block through a PBFT protocol.
- Committee election could be based on the previous algorithms we studied:
  - Based on PoW, Byzcoin [Kogias et al., 2016].
  - Based on PoS and VRFs, Algorand [Gilad et al., 2017].
    - We covered it before.

- *We will explore Byzcoin*

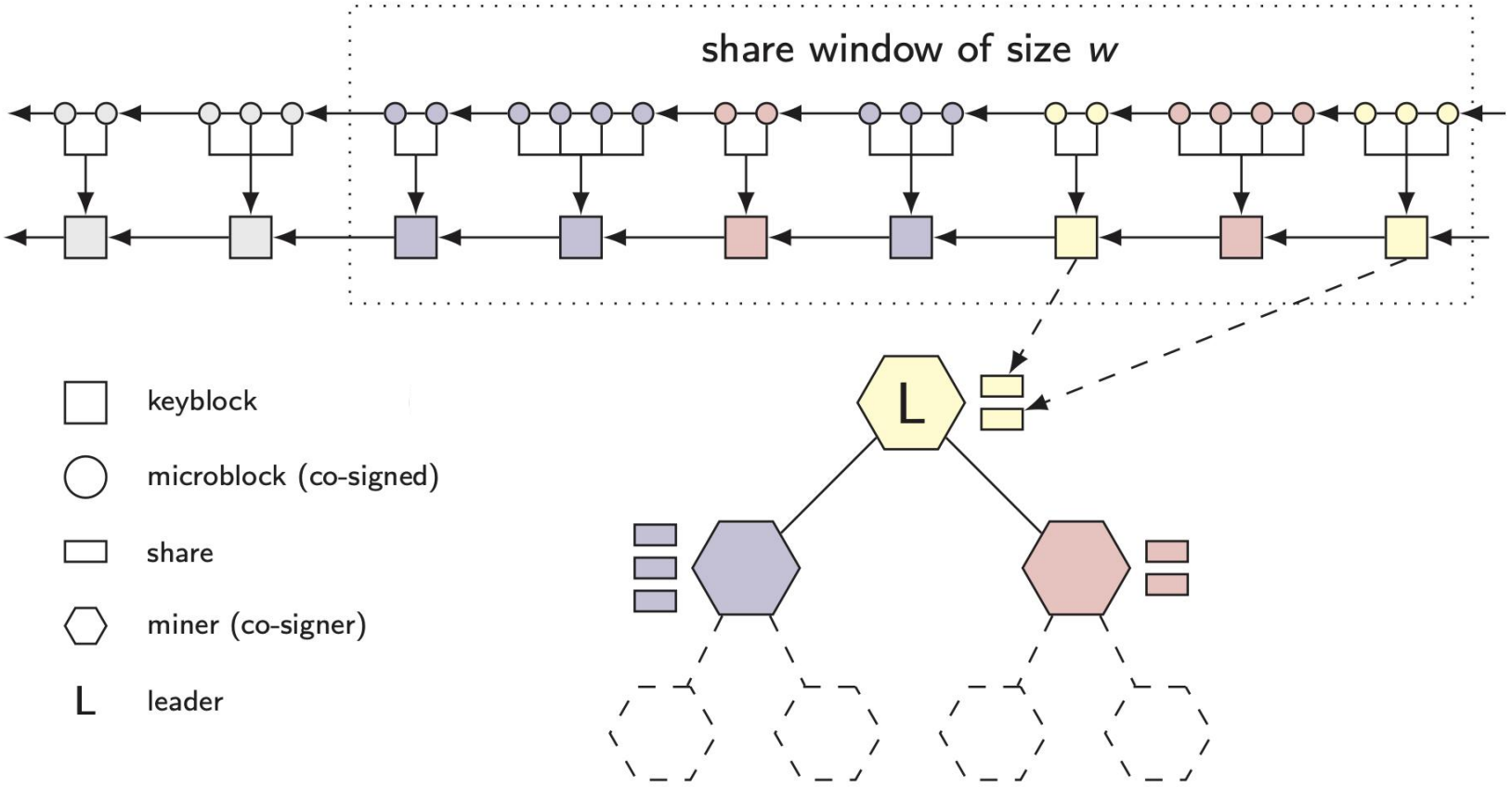
Byzcoin I

- Dynamic (rotating) committee election based on PoW.
  - Decouples transaction verification from mining.
    - Maintains two parallel blockchains; one contains microblocks (each contains a set of verified transactions mined using BFT), the other contains keyblocks (groups several microblocks together in the header and mined through PoW).
  - Once a transaction appears in a microblock, it is confirmed.
  - For every epoch, Keyblocks are used to elect the BFT committee (and its leader) who will agree on the microblocks.
    - Needed to prevent Sybil attacks in open-committee BFT.
    - A sliding window based approach, the miners of the last n keyblocks are the current committee members, and the miner of the latest keyblock is leader.

Byzcoin II

- For microblocks, a leader initiates a PBFT protocol to collectively agree (and sign) new blocks (assuming the committee of size 3t+1):
  - Leader proposes a new microblock and announces it to the rest of the committee. This is called a pre-prepare message.
  - Each committee member validates the block and broadcasts a prepare message indicating accepting the block.
  - Once each member receives at least 2t + 1 prepare messages, they acknowledge that by broadcasting a commit message.
- All responses are authenticated using collective signature, CoSi (or basically multi-signature that allows several parties to sign a message and produce a single signature instead of many).

Byzcoin Pictorially



Mining rewards are distributed in proportion to the number of shares.

**From [Kogias et al., 2016]*

BFT Consensus - Issues

- Network connectivity/synchrony assumptions.
- ⅓ of the mining power can be malicious.
  - Less than Bitcoin tolerance level.
- Scalability (i.e. number of miners).
  - Adding more miners to the system does not boost throughput; still one committee is elected per epoch to process transactions.
- Byzcoin-style, specifically, is susceptible to targeted attacks.
  - Committee/leader are known in advance.
  - How did Algorand address this issue?

Hybrid Mining

Hybrid Mining Algorithms

- Combine several mining algorithms together to solve the limitations of using a single algorithm.
- Examples:
  - Proof-of-stake and proof-of-work are combined together. Proof-of-work is used for initial distribution the currency in the system, and then the network continues using proof-of-stake only.
  - Or combine PoW or PoS with Byzantine agreement based protocols (PoW/PoS are utilized in the committee election process as seen previously in Algorand and Byzcoin).

References

- [O'Dwyer et al., 2014] O'Dwyer, Karl J., and David Malone. "Bitcoin mining and its energy footprint." (2014): 280-285.
- [Gilad et al., 2017] Gilad, Yossi, Rotem Hemo, Silvio Micali, Georgios Vlachos, and Nickolai Zeldovich. "Algorand: Scaling byzantine agreements for cryptocurrencies." In In Proceedings of the 26th ACM Symposium on Operating Systems Principles (SOSP). 2017.
- [Dziembowski et al., 2015] Dziembowski, Stefan, Sebastian Faust, Vladimir Kolmogorov, and Krzysztof Pietrzak. "Proofs of space." In Annual Cryptology Conference, pp. 585-605. Springer, Berlin, Heidelberg, 2015.
- [Moran et al., 2016] Moran, Tal, and Ilan Orlov. "Proofs of Space-Time and Rational Proofs of Storage." IACR Cryptology ePrint Archive 2016 (2016): 35.
- [Miller et al., 2014] Miller, Andrew, Ari Juels, Elaine Shi, Bryan Parno, and Jonathan Katz. "Permacoin: Repurposing bitcoin work for data preservation." In Security and Privacy (SP), 2014 IEEE Symposium on, pp. 475-490. IEEE, 2014.
- [Kogias et al., 2016] Kogias, Eleftherios Kokoris, Philipp Jovanovic, Nicolas Gailly, Ismail Khoffi, Linus Gasser, and Bryan Ford. "Enhancing bitcoin security and performance with strong consistency via collective signing." In USENIX Security 16, 2016.
- [Castro et al., 1999] Castro, Miguel, and Barbara Liskov. "Practical Byzantine fault tolerance." In OSDI, vol. 99, no. 1999, pp. 173-186. 1999.