CSE 2550: Blockchain Technology I

Lecture 9
Mining and Consensus

Ghada Almashaqbeh
UConn - Fall 2023

Outline

- Mining and consensus algorithms/protocols.
  - Proof-of-stake.
  - Proof-of-space/storage.
  - Proof-of-elapsed time.
  - 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.

Proof-of-Work is 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)?
  - In 2014, researchers showed that ***electricity consumption*** of Bitcoin mining is comparable to some developed countries.
  - Mining is not for free: requires advanced hardware, cooling systems, huge electricity bills, maintenance cost, etc.
- Can we use other forms of resources (storage, bandwidth, etc.) to have a useful mining process?
- Can we reduce the electricity consumption of the mining process?

Proof-of-Work - 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

- 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 proof-of-work).
    - Announcing the election result to others could take place at the beginning or at the end of the round.
- Mining a block requires only validating transactions and adding them to a block, then signing this block.
  - No extensive computations are needed, 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.
- 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 which determines if a user has been selected as a leader based on his/her stake value.
    - The VRF requires a secret key; hence, can produce the output hash other than the key owner.
    - A proof is produced to verify in zero knowledge (without revealing the secret key) that the hash 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).
- Secure single leader election is a very 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. Or elect several leaders per round. (Algorand pioneered this solution.)

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.

Useful Mining

- Many flavors, with the goal of building a mining process with useful outcome.
- 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 take the form of a challenge/response approach, which needs to be implemented in a non-interactive way.
- Usually a miner will put 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 concerns:
  - 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.

Proof-of-Elapsed Time

- Relies on secure/trusted hardware
  - Also called secure enclaves or Trusted Execution Environments (TEE), e.g., Intel SGX.
- Two main flavors:
  - Each miner requests a wait time from its enclave, the miner with the shortest wait time will be the round leader.
  - A variant of useful proof-of-work.
    - The enclaves execute some useful computation.
    - Each instruction cycle is treated as a lottery ticket. If it wins, the enclave owner, i.e., the miner, is authorized to mine a new block.
- In both approaches an irrefutable proof must provided attesting that a miner has indeed won the round.

Proof-of-Elapsed Time Issues

- Requires trusting the secure hadware manfuartcauer.
- Breaking one machine allows the attacker to always win the race and be the leader of every round.
- An attacker may purchase several chips and run the mining on all of them concurrently, use the results of the winning chip.
    - Called stale chip problem.

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].
- Experimental results showed that transactions are confirmed in less than a minute in these protocols.

- *[Example on how this works will be shown on the board in class]*

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).

Hybrid Mining Algorithms

- Combine several mining algorithms together to solve the limitations of using a single algorithm.
- Examples:
  - As mentioned before, usually 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).

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.
- [Kiayias et al., 2017] Kiayias, Aggelos, Alexander Russell, Bernardo David, and Roman Oliynykov. "Ouroboros: A provably secure proof-of-stake blockchain protocol." In Annual International Cryptology Conference, pp. 357-388. Springer, Cham, 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.