BLOCKCHAINS

ARCHITECTURE, DESIGN AND USE CASES

SANDIP CHAKRABORTY
COMPUTER SCIENCE AND ENGINEERING,
IIT KHARAGPUR

PRAVEEN JAYACHANDRAN
IBM RESEARCH,
INDIA

Gilad, Y., Hemo, R., Micali, S., Vlachos, G., & Zeldovich, N. (2017, October). *Algorand: Scaling byzantine agreements for cryptocurrencies*. In *Proceedings of the 26th Symposium on Operating Systems Principles* (pp. 51-68). ACM.

Algorand: Scaling Byzantine Agreements for Cryptocurrencies

Algorand: Architecture

- Select a random user
  - prepare a block
  - propagate block through gossiping
- Select random committee with small number of users (~10k)
  - run Byzantine Agreement on the block
  - digitally sign the result
  - propagate digital signatures
- **Who select the committee??**

Cryptographic Sortition

- Each committee member selects himself according to per-user weights
- Implemented using verifiable random functions (VRFs)

<hash,proof> ← $\text{VRF}_{sk}(x)$

- **x:** input string
- **(pk_i,sk_i):** public/private key pair
- **hash:** hashlenbit-long value that is uniquely determined by sk and x
- **proof:** enables to check the hash indeed corresponds to x

Block Proposal

- Minimizing unnecessary block transmissions
  - **discard** messages **not** having **highest priority** seen by that user so far
  - priority for the block proposal obtained by **hashing** the hash **output** of **VRF** concatenated with the **sub-user index**
- Waiting for block proposals
  - $\boldsymbol{\lambda_{\text{stepvar}} + \lambda_{\text{priority}}}$ time to identify the highest priority (~10 seconds)
  - $\boldsymbol{\lambda_{\text{stepvar}}}$: the variance in how long it takes different users to finish the last step of BA*
  - $\boldsymbol{\lambda_{\text{priority}}}$: the time taken to gossip the priority and proof message

BA*

- **Two phase**:
  - reduces the problem of agreeing on a block to agreement on one of two options - *final consensus* or *tentative consensus*

Strong Synchrony versus Weak Synchrony

- **Strong Synchrony**: Most honest users (say, 95%) can send message that will be received by most other honest users within a known time bound
  - Adversary can not control the network for long
  - Ensures liveness of the protocol

Strong Synchrony versus Weak Synchrony

- **Weak Synchrony**: The network can be asynchronous for long (entirely controlled by adversary) but bounded period of time
  - **There must be a strong synchrony period after a weak synchrony period**
  - Algorand is **safe** under weak synchrony

Final Consensus

- One user reaches final consensus
  - Any other user that reaches final or tentative consensus in the same round must agree on the same block value (**ensures safety**)
  - Confirm a transaction when the block reaches to the final consensus

Tentative Consensus

- One user reaches tentative consensus
  - Other users may have reached consensus on a different (but correct) block
  - Can be in two cases
    - The network is strongly synchronous - adversary may be able to cause BA* to reach tentative consensus on a block - BA* is unable to confirm that the network was strongly synchronous
    - The network was weakly synchronous - BA* can form multiple forks and reach tentative consensus on two different blocks - users are split into groups

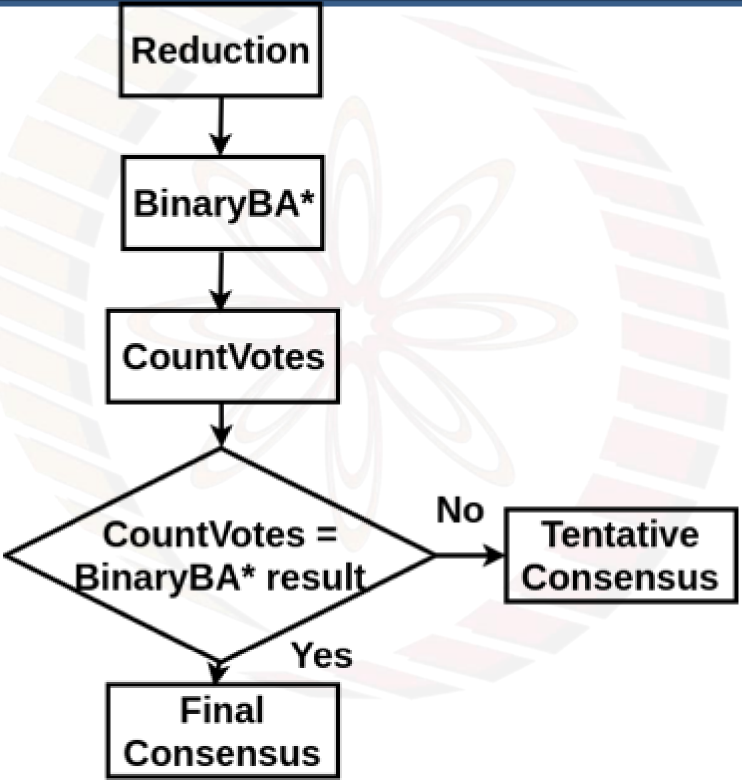
Come out of Tentative Consensus

- Run BA* periodically to come out of tentative consensus - run the next round
  - Network can not be under weak synchrony all the times
  - Cryptographic sortition ensures different committee members at different rounds of the BA*

BA*

- **Two phase**:
  - reduces the problem of agreeing on a block to agreement on one of two options - *final consensus* or *tentative consensus*
  - reaches agreement either agreeing on a proposed block, or agreeing on an empty block

BA*: Overall Procedure



Algorand: Summary

- No forks
- No miners
- No proof-of-work
- No wait for confirmation
- Trivial computation
- Perfect scalability
- Great security

thank you!