Simplex Consensus: A Simple and Fast Consensus Protocol [CP23][1]

Iniyan Joseph

2026-11-12

Purdue University

iniyanijoseph@gmail.com

[1] CS 592 - Special Topics in Blockchains

Recap on Consensus

- Consensus algorithms are used for a network of machines to agree on some ground truth.
- We should
  - Agree on the ordering of transactions. (Consensus)
  - Make progress (Liveness)

Goals

- Fairness
  - Each player should have a fair chance at proposing a block
- Low Latency
  - Fast transaction confirmation
  - Some block proposers may be faulty
- Easy to Understand

Setting

- *Partially-synchronous* setting (same as PBFT [CLo99])
  - Before GST, messages take arbitrarily long
  - After GST, messages are delivered in at most Δ time
- Proof-of-Stake
- The set of n players is known ahead of time.
- $f < \frac{n}{3}$ byzantine faults

Prior Work

- The prior state-of-the-art requires multiple honest leaders in-a-row to confirm blocks, which hurts latency
  - It may be hard to achieve this if we have some bad leaders

Main Idea

- Design a protocol with an easy liveness proof

Preliminaries

- Each block is a tuple of the form

$$b_h = (h, \text{Hash}(b_1 ... b_{h-1}), \text{transactions})$$

- The blockchain may contain "dummy" blocks

$$b_h = (h, X, X)$$

- A player i votes from a block b_h by signing the message "vote b_h"
- If $> \frac{2n}{3}$ people vote for this block, then we can confirm/"notarize" it

Iteration

- Proceed in iterations $h = 1, 2, 3, \ldots$
- Move to the next iteration when you have seen a notarized blockchain of length h and sent it to everyone else

Constructing Notarized Blocks

- Each iteration has a leader chosen randomly
- In iteration h
  1. If i is the new leader, i chooses notarized blockchain of $h-1$, extends with a new block b_h, and sends everyone a signed message
  2. After seeing the **first** valid proposal from the leader, player i sends everyone a signed message vote b_h
- Everyone adds iteration when they see $\frac{2}{3}$ (Quorum!)

But what about when things go wrong?

Faulty Iterations

Scenario: Faulty Leader

- Sends different block proposals to different players to force honest players to split their vote and form less than a quorum.
- Question to the crowd:
  - What breaks? liveness, or consistency?

Solution

- Liveness! Because no new blocks can be added without a quorum, no progress is made by the system.
- But dummy blocks give an easy solution. If more than 3Δ time passes without approving a block, a player i sends everyone a signed message vote dummy_h
- But now there is a problem: there may be both a notarized dummy block and a notarized block proposal for h in the view of honest players

Finalization

- If
  - i enters iteration $h + 1$
  - i did not time out and consequently vote for dummy_h
- Send a signed "finalize h" message
- At any point, after seeing $\frac{2}{3}$ "finalize h" messages, i now has a quorum and can finalize h

Questions?

Proving Security (Consistency)

- Let's assume that Anne and Blythe are two honest players
- Anne's blockchain is `LOG`, and is at most as long as Blythe's blockchain `LOG'`
- For consistency, we must show that `LOG` is a prefix or equal to `LOG'`

Consistency

- **Claim:** There can only be one notarized blockchain of length h across all honest nodes

Consistency

- **Claim:** At most one block proposal from h can be notarized in an honest view
  - Proof: Each player votes for at most one proposal. By quorum intersection, only one of the two can win if there are at least $\frac{2n}{3}$ honest players
- **Claim:** A dummy block at h cannot be finalized
  - Proof: Honest players only vote for finalizing h or dummy. Quorum intersection

Proving Security (Liveness)

- **Claim:** If the network is good (after GST), an honest leader can always get its block proposal notarized, and then finalized
  - If the honest leader enters h by time $t >$ GST, then every honest player enters h by time $t + \delta$

Liveness

- Every honest node will see a notarization for some block of height h by time $t + 2\delta$
  - Leader enters at time t on iteration h and proposes b_h
  - Every honest player enters iteration h and sees the proposal at time $t + \delta$
    - Either they send a vote or someone already entered iteration $h + 1$
  - Every honest player seems some notarized block of height h at time $t + 2\delta$
- The dummy block of height h cannot be notarized in any honest view before time $t + 2\delta$

Thus, every honest player finalizes the leader's block proposal by time $t + 3\delta$

Liveness

- **Claim:** If the network is after GST, any iteration will conclude after $3\Delta + \delta$
- At t, every honest player has entered
- At $t + 3\Delta$, every honest timer for h has fired (i.e. voted for dummy_h), or some honest process entered iteration $h + 1$ already
- At $t + 3\Delta$, every honest timer for h has fired, or some honest process entered iteration $h + 1$ already
- At $t + 3\Delta + \delta$, every honest player enters iteration $h + 1$

Questions?

That’s All Folks!

References

Bibliography

[CP23] B. Y. Chan and R. Pass, "Simplex consensus: A simple and fast consensus protocol," in *Theory of Cryptography Conference*, 2023, pp. 452–479.
[CLo99] M. Castro, B. Liskov, and others, "Practical byzantine fault tolerance," in *OsDI*, 1999, pp. 173–186.