Narwhal and Tusk:

A DAG-based Mempool and Efficient BFT Consensus

Presenter: Aditya, Howard, Karamjeet, and Divyanshu

The slides closely follow the format of Alberto Sonnino's presentation slides in ConsensusDays 21.

Introduction

- Distributed Systems are increasing in ubiquity following the introduction of Bitcoin
- However, Bitcoin is mired by a lack of transaction speed due to its computationally intense proof of work algorithm
- Due to the dissatisfaction with the transaction throughput of bitcoin, committee based consensus protocols were crafted.

Introduction Part 2

- Hotstuff stood at the zenith of these algorithms for a time, but its over reliance on the leader node to coordinate consensus lead to a bottleneck effect.
- Furthermore, these consensus algorithms were “monolithic” and solely focused on improving the “theoretical message complexity”

What the paper proposes and assumes

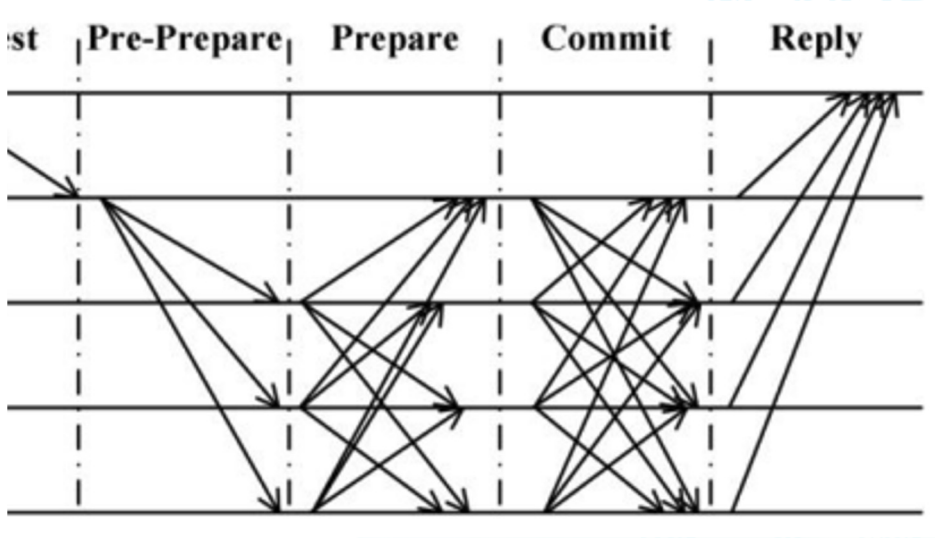
- The paper proposes Narwhal, a DAG based Mempool abstraction
- It also proposes Tusk, a high-throughput zero overhead consensus protocol.”
- The paper assumes 3f+1 configuration where f is max number of potential Byzantine Nodes

Raison d'Etre of Narwhal and Tusk

- The raison d'etre of Narwhal and Tusk is to "separate the task of reliable transaction dissemination from transaction ordering" so as to increase performance of distributed systems, factoring in the presence of Byzantine Nodes.
- The prognosis that the authors make in this paper is that "a better Mempool that reliably distributes transactions is the key enabler of a high-performance ledger." The only job of consensus, "should be to sequence a very small amount of metadata."

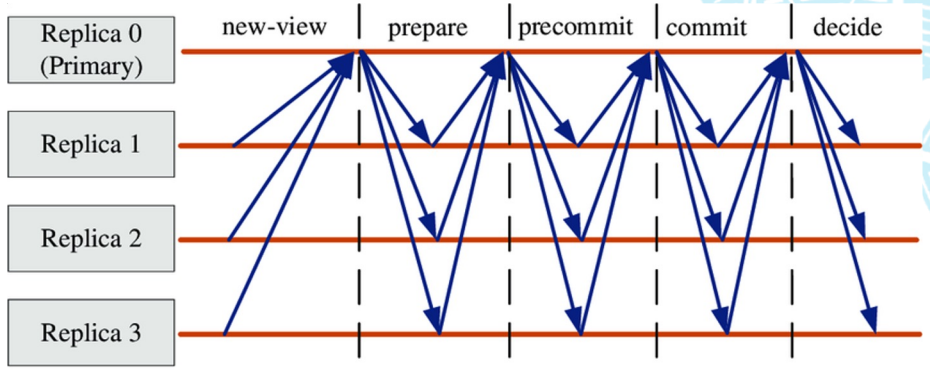
PBFT Recap

- Let’s do a brief recap on PBFT as it is the foundation upon which many consensus algorithms are built. PBFT is a protocol that was developed to help Distributed Systems to mitigate the effects of Byzantine Nodes in a partially synchronous environment.
- It has four phases: Pre-prepare, Prepare, Commit, and Reply.



Recap of Hotstuff

- The protocol, HotStuff, is based of PBFT, and it “allow the distributed system to achieve consensus via a Leader Node and can handle malicious nodes, for both the primary and replicas.”
- It has a linear time complexity as opposed to PBFT’s Quadratic Time complexity

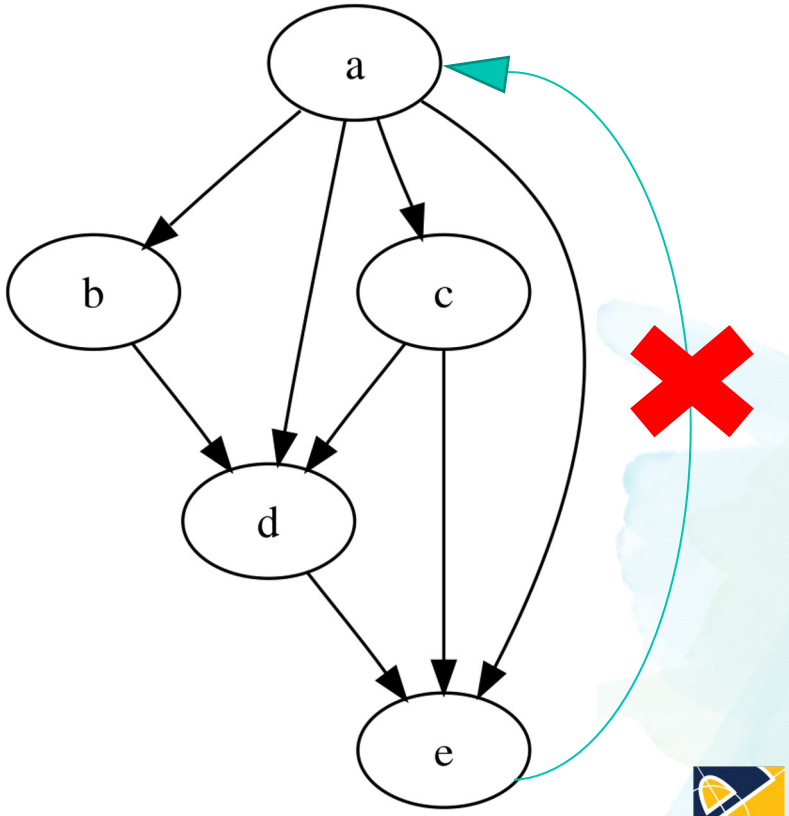


DAG - Directed Acyclic Graph

Narwhal is a DAG-based Mempool

DAG - Directed Acyclic Graph

A Directed Acyclic Graph (DAG) is a particular data structure

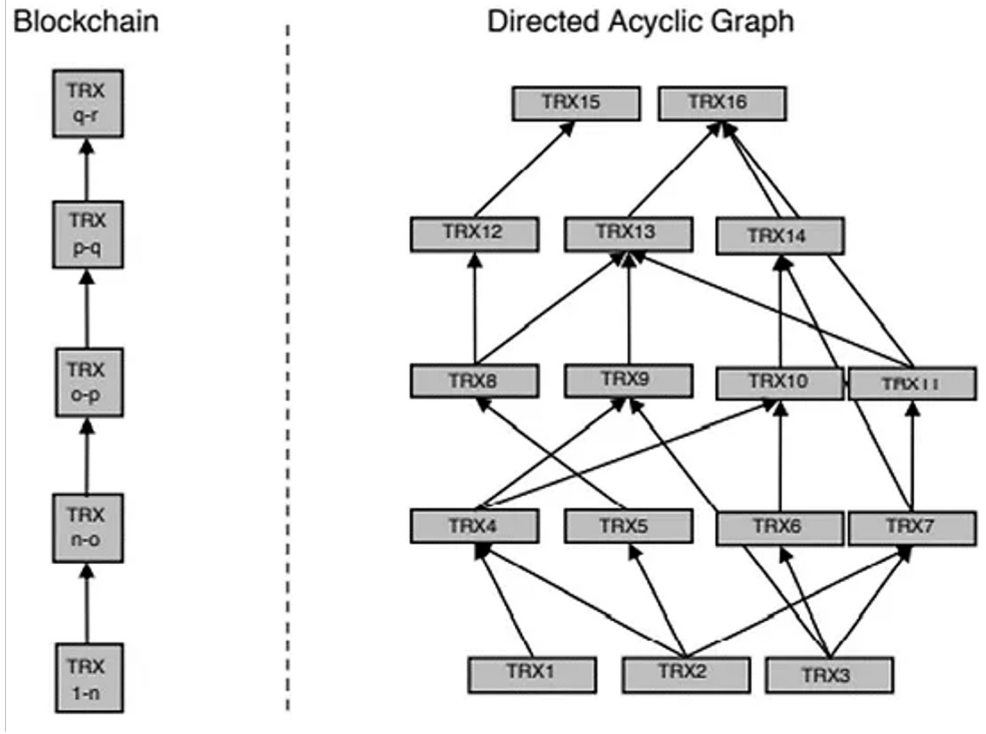


Directed: Each edge in a DAG has a direction, indicating the flow or relationship between nodes.
Acyclic: There are no cycles or loops in the graph.

DAG Protocol

Uses DAGs to create an asynchronous Byzantine Fault-Tolerant consensus.

Several transactions in a block



One transaction in one node

Links = References

Parallelized -> Higher throughput

UCDAVIS COLLEGE OF ENGINEERING

DAG Advantages

High Scalability

- DAG can process more transactions compared to a traditional blockchain

No Block Time

- High transaction speed, allowing users to broadcast transactions and receive confirmations simultaneously

Block time: A typical blockchain network has a time-lapse or waiting period between when a transaction is logged and when it is confirmed.

DAG Potential Issue

Complexity in Consensus

- Coordinating the verification of transactions in a graph structure can be challenging and may require sophisticated consensus mechanisms --> Tusk

Centralization Risks

- DAG implementations might rely on centralized entities during the bootstrapping phase

Security Concerns

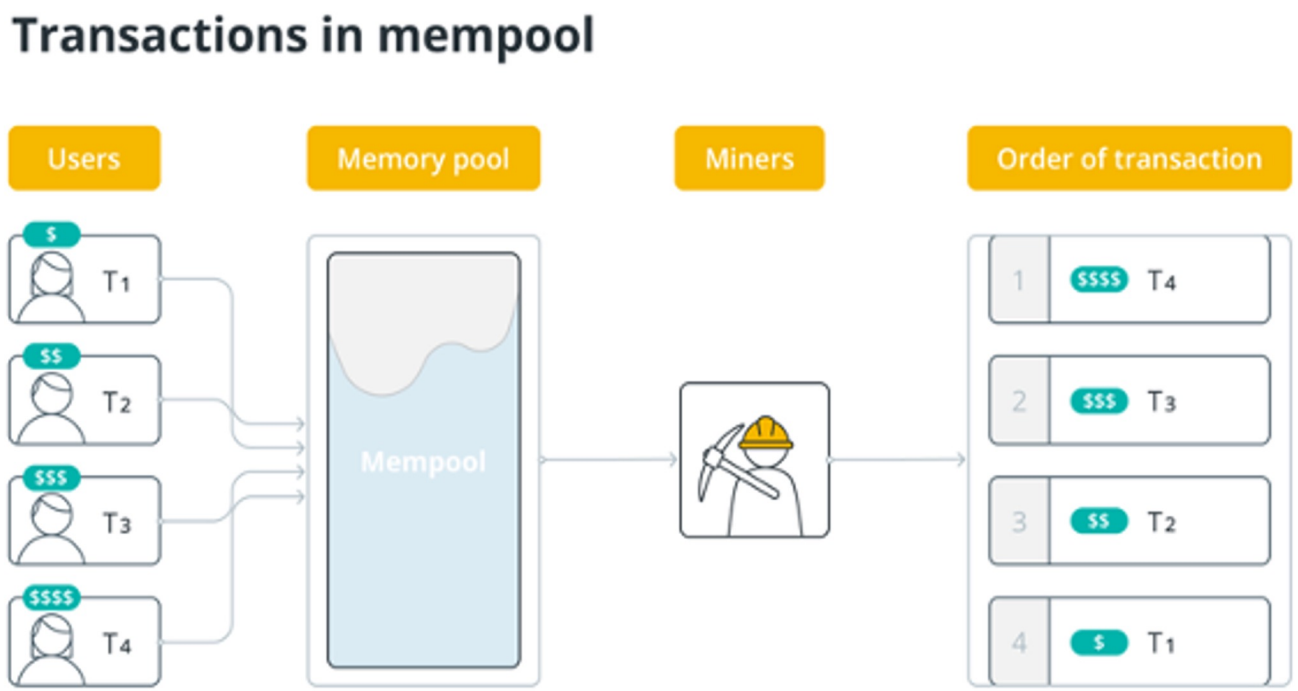
- These third parties generate centralization and can therefore restrict the system's overall security.

Mempool

Mempool

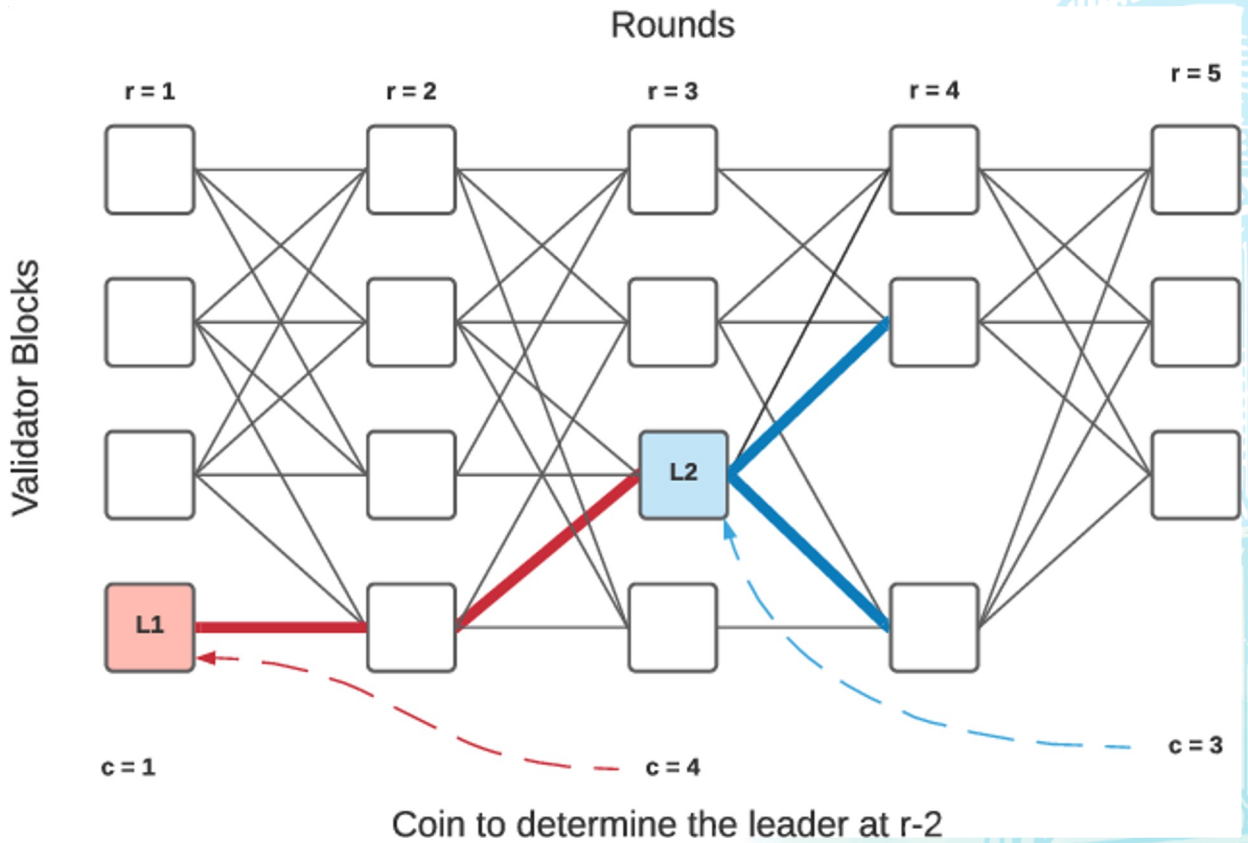
Understanding Mempool in Blockchain:

- Mempool (Memory Pool), serving as a holding area for all pending transactions.
- It acts as a buffer, storing unconfirmed transactions until they are picked up by miners or validators for inclusion in a block.



DAG Based Mempool

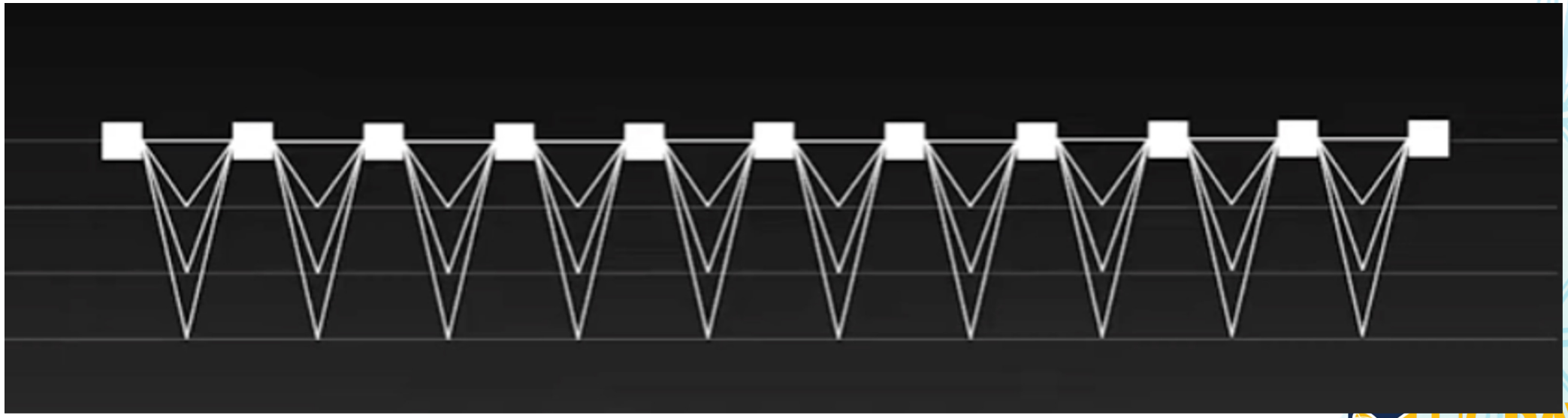
- DAG-based Mempool combines Directed Acyclic Graph (DAG) structures with traditional Mempool mechanisms.
- Increased throughput, faster validation, and improved scalability



Traditional Protocol

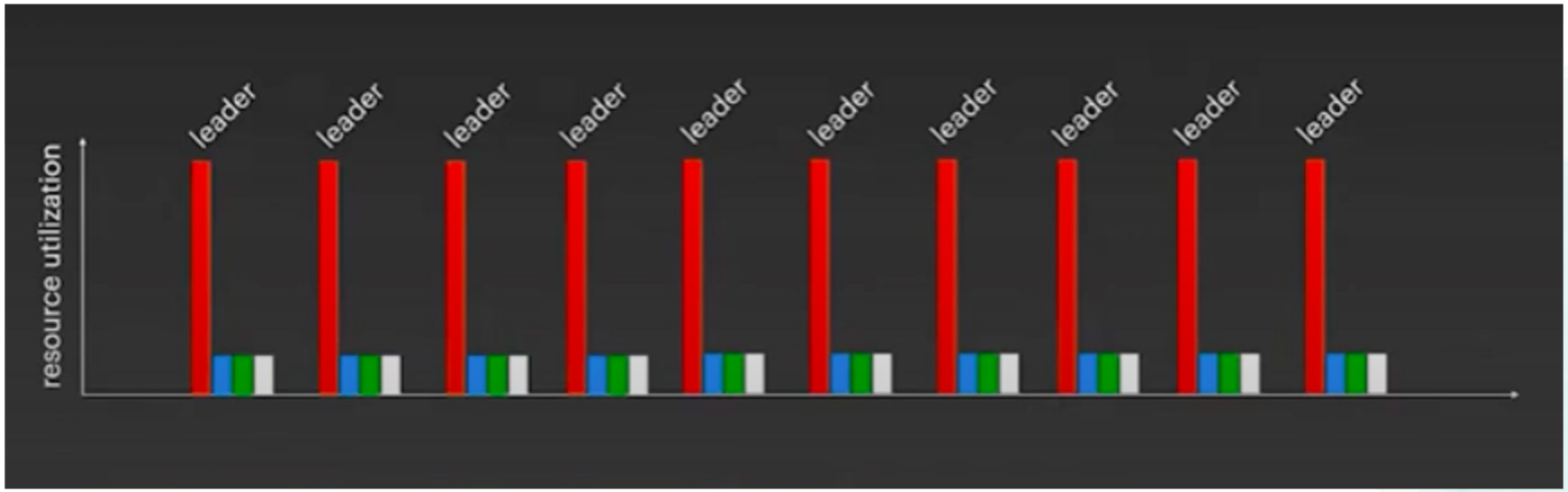
Traditional blockchain protocols often follow a **linear, monolithic design**, leading to bottlenecks in transaction processing.

Focused more on reducing overall **message complexity.**



Traditional Protocol

- Traditional protocols often lead to imbalanced resource utilization, with the **leader node heavily burdened** while **other nodes remain underutilized.**



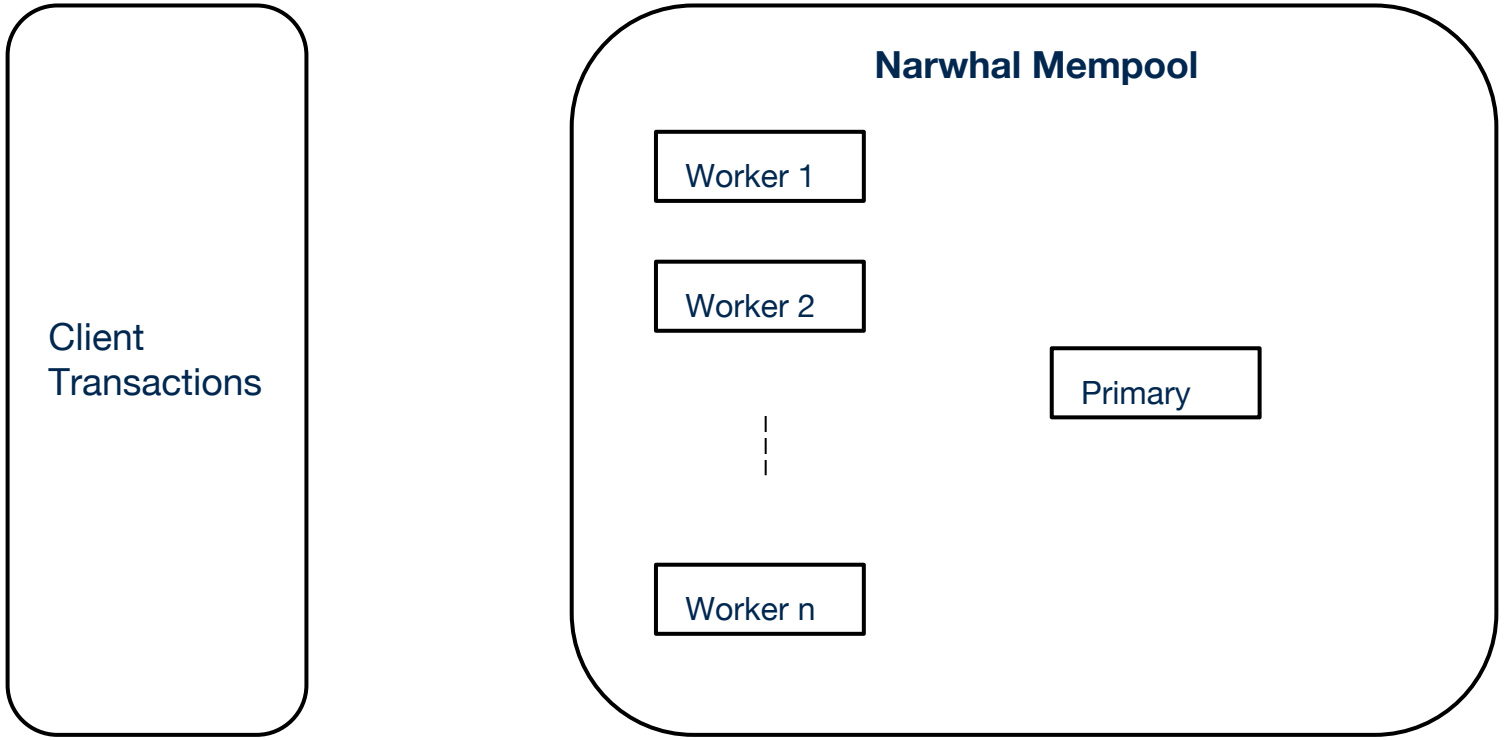
Why not fully utilise miner resources?

Mempool is the Key

Narwhal - A DAG based Mempool

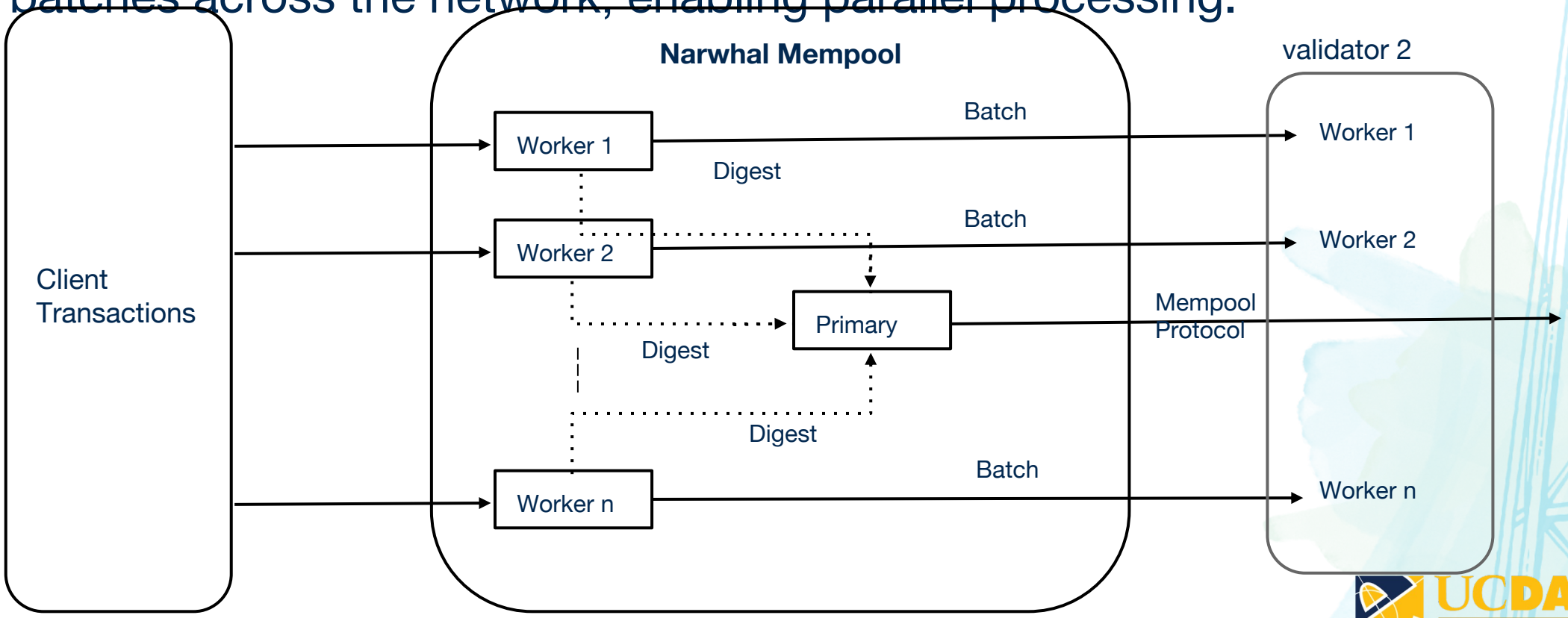
Narwhal

Narwhal introduces a novel approach to transaction processing in blockchain, utilizing a DAG-based Mempool system



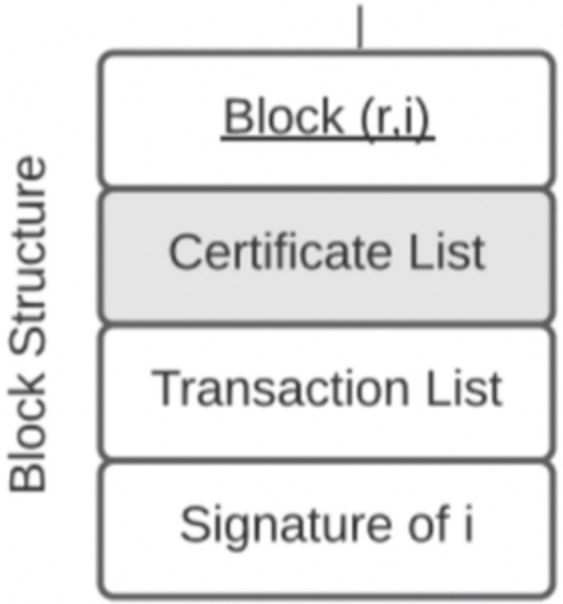
Narwhal

Workers within Narwhal batch transactions and broadcast these batches across the network, enabling parallel processing.



Narwhal

Block Structure



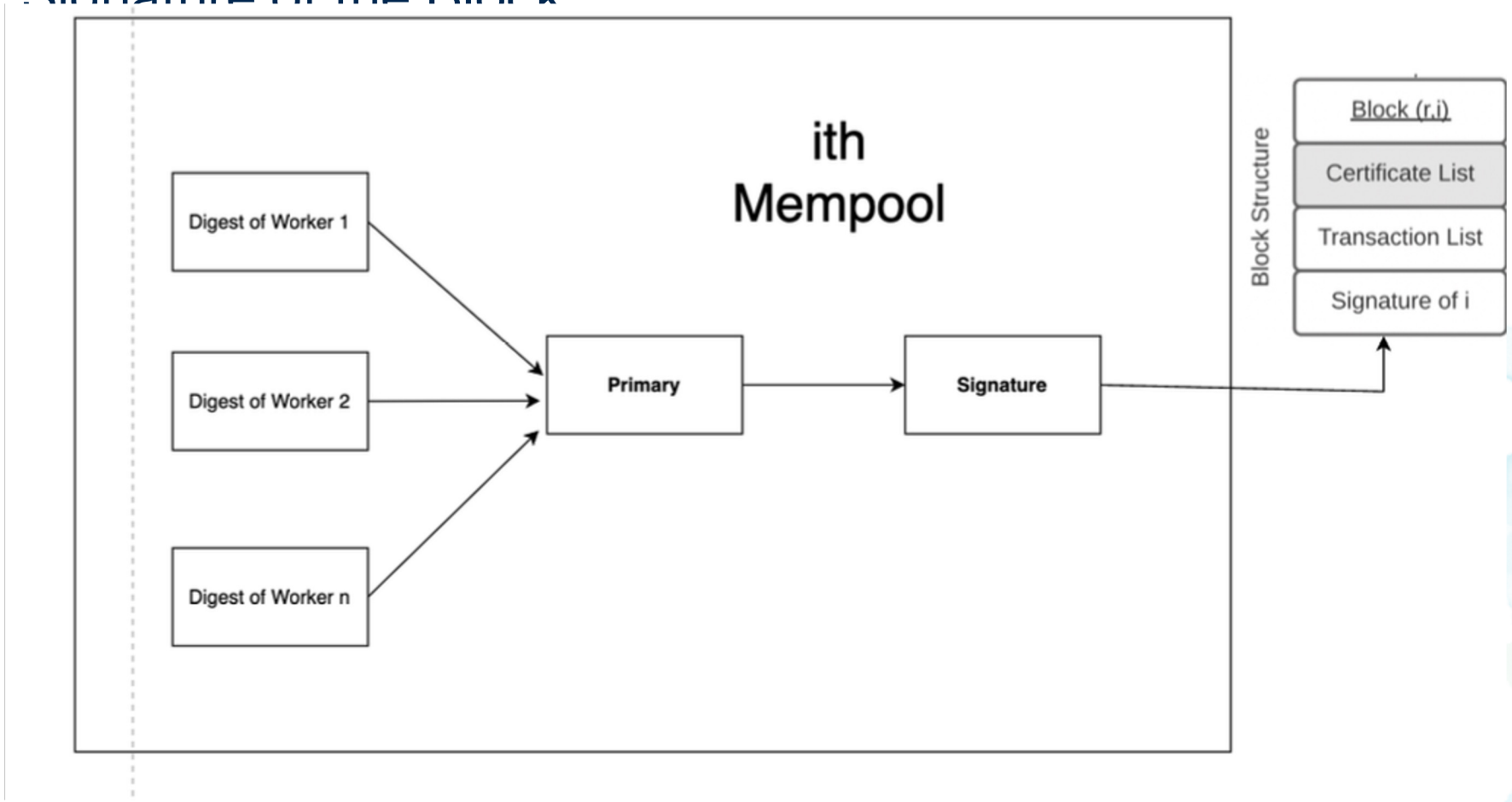
Where is this coming from?

We know where this is from. The Client

Where is this coming from?

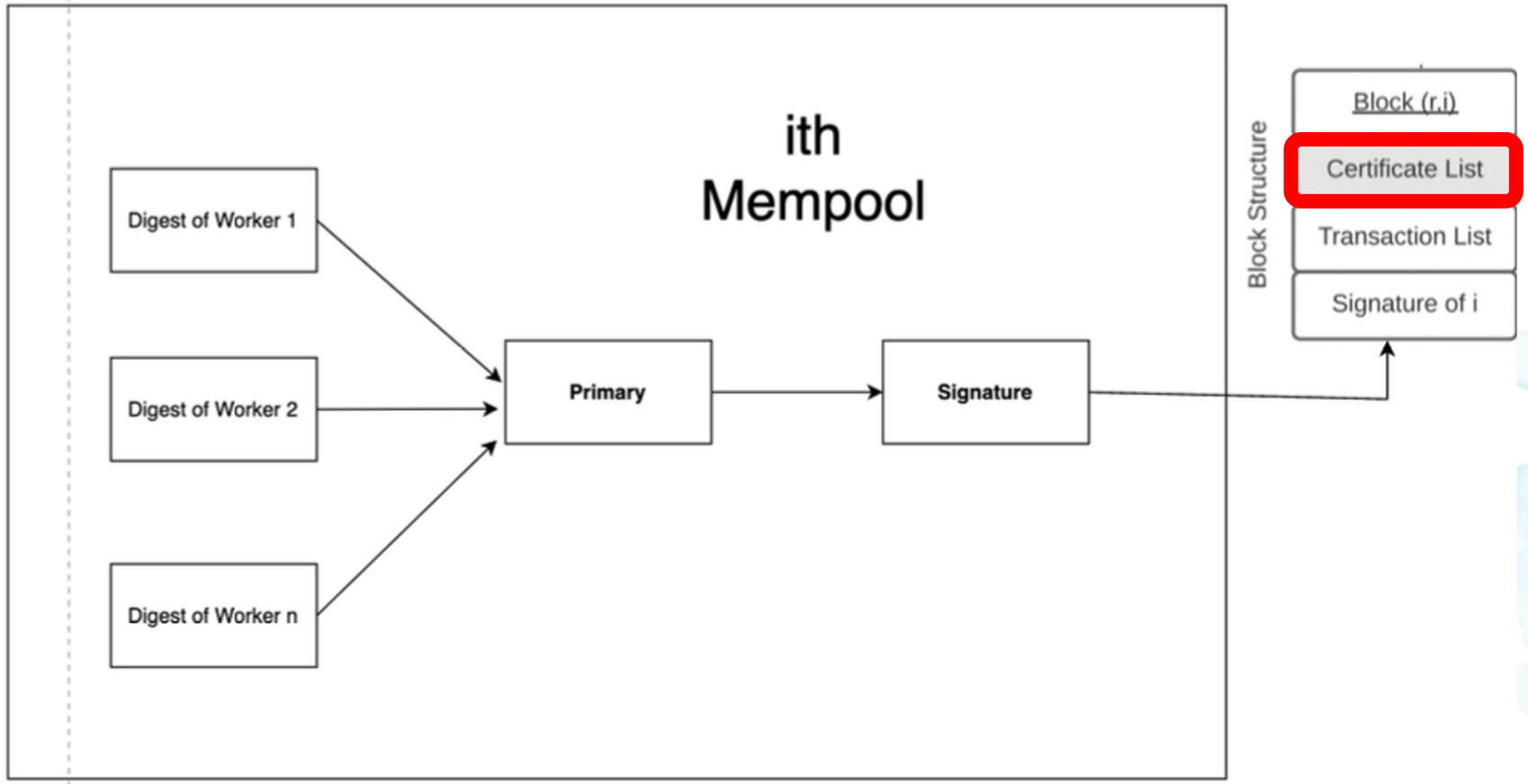
Narwhal

Signature of the Block



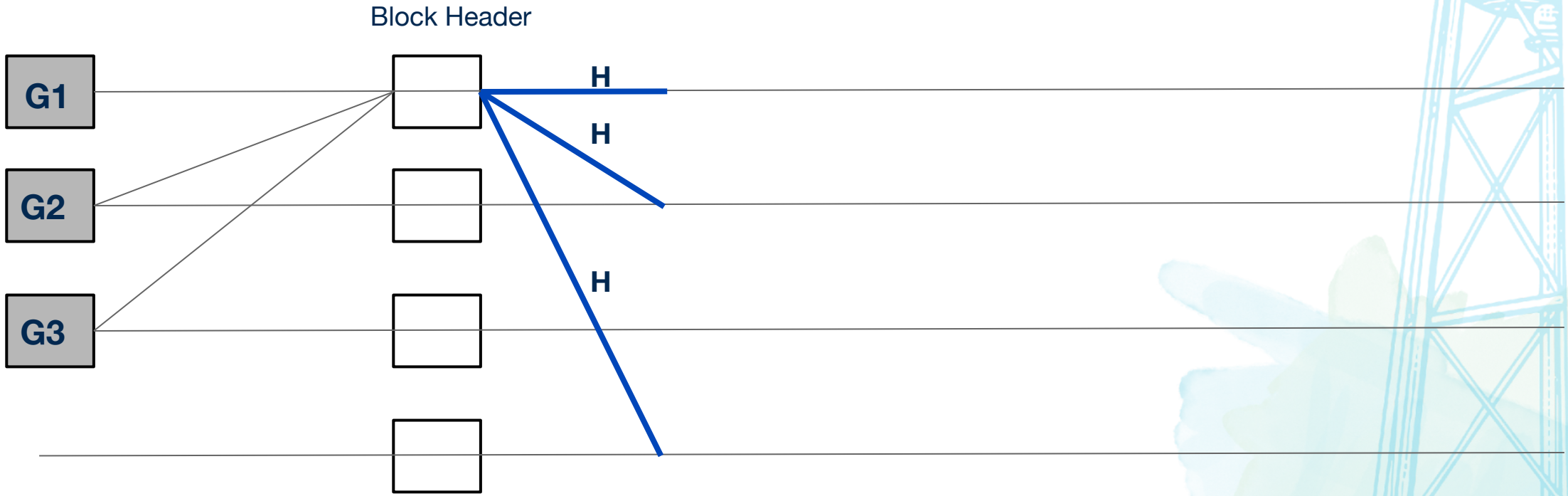
Narwhal

Signature of the Block

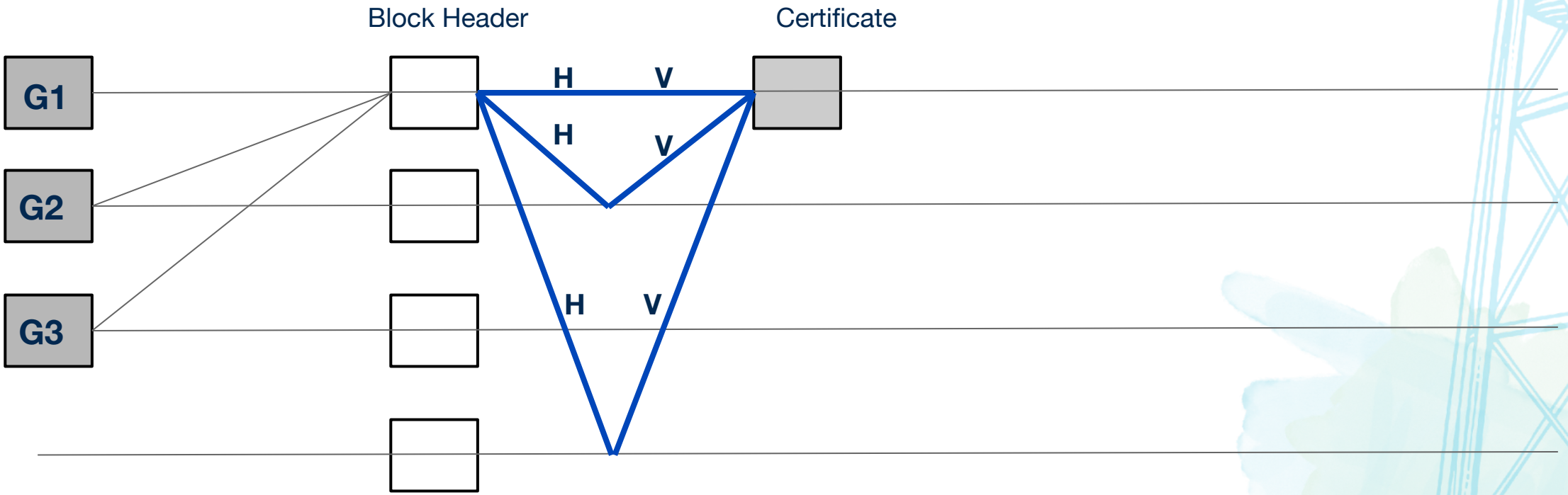


???

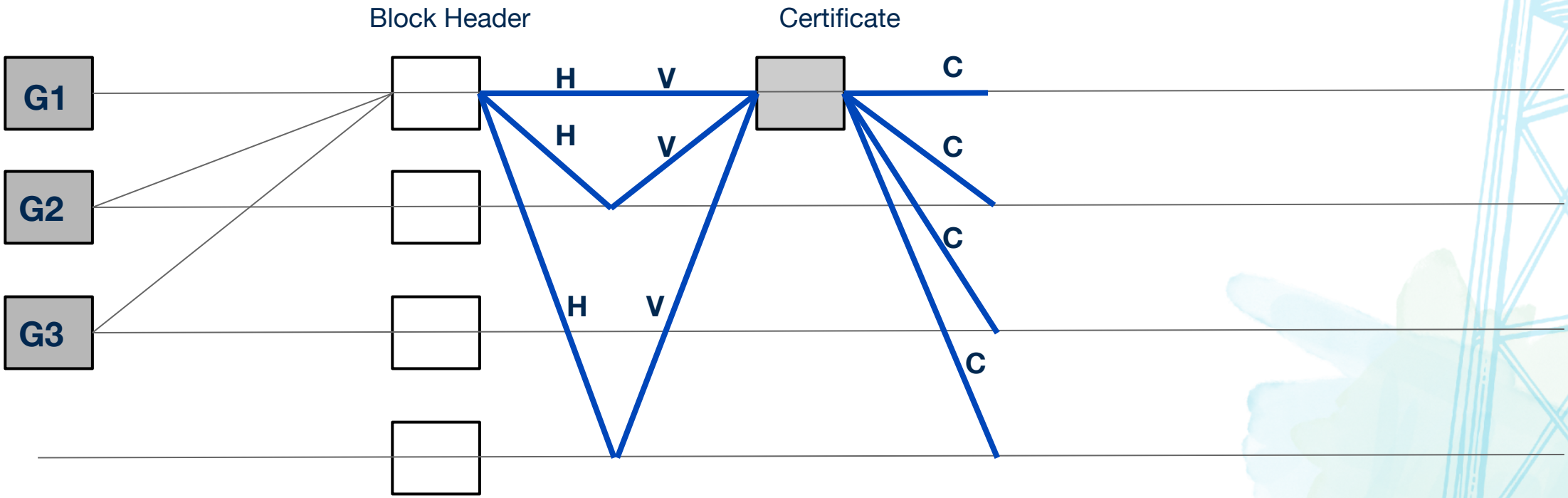
Narwhal



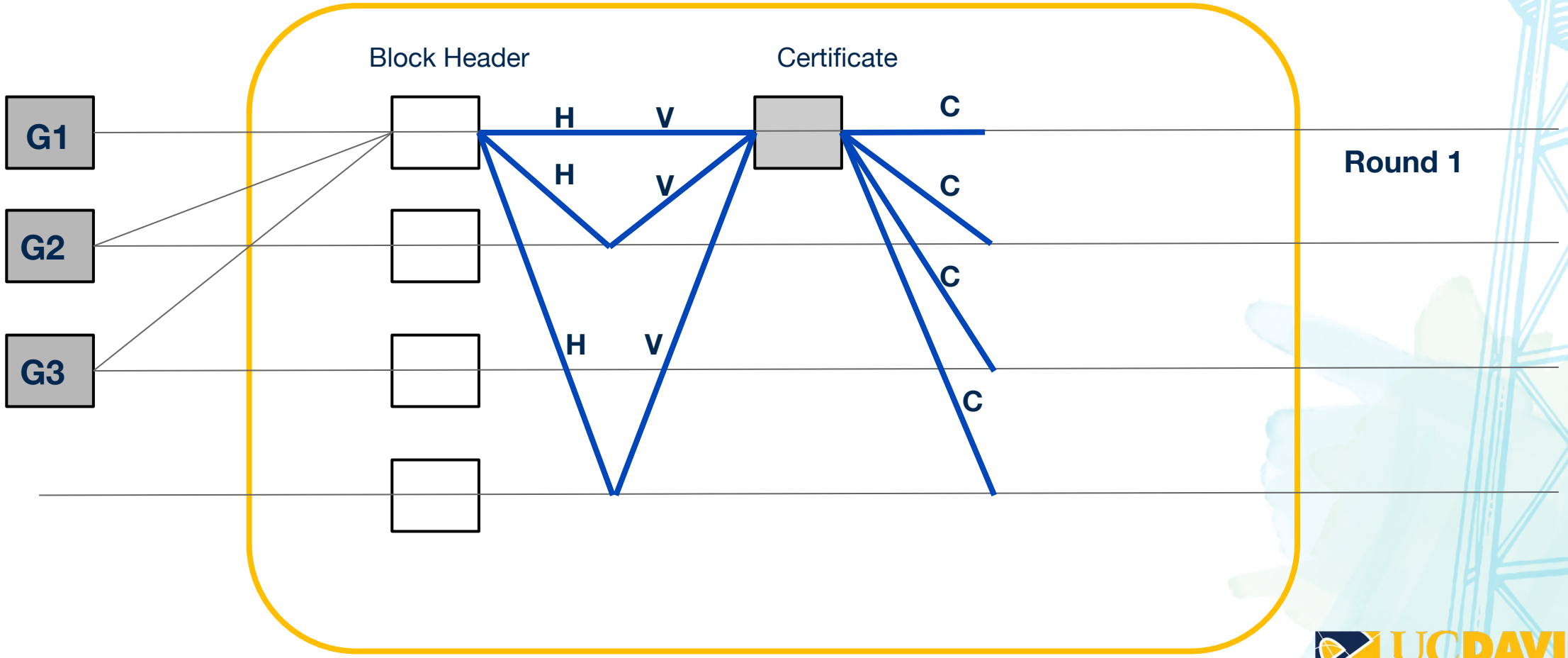
Narwhal



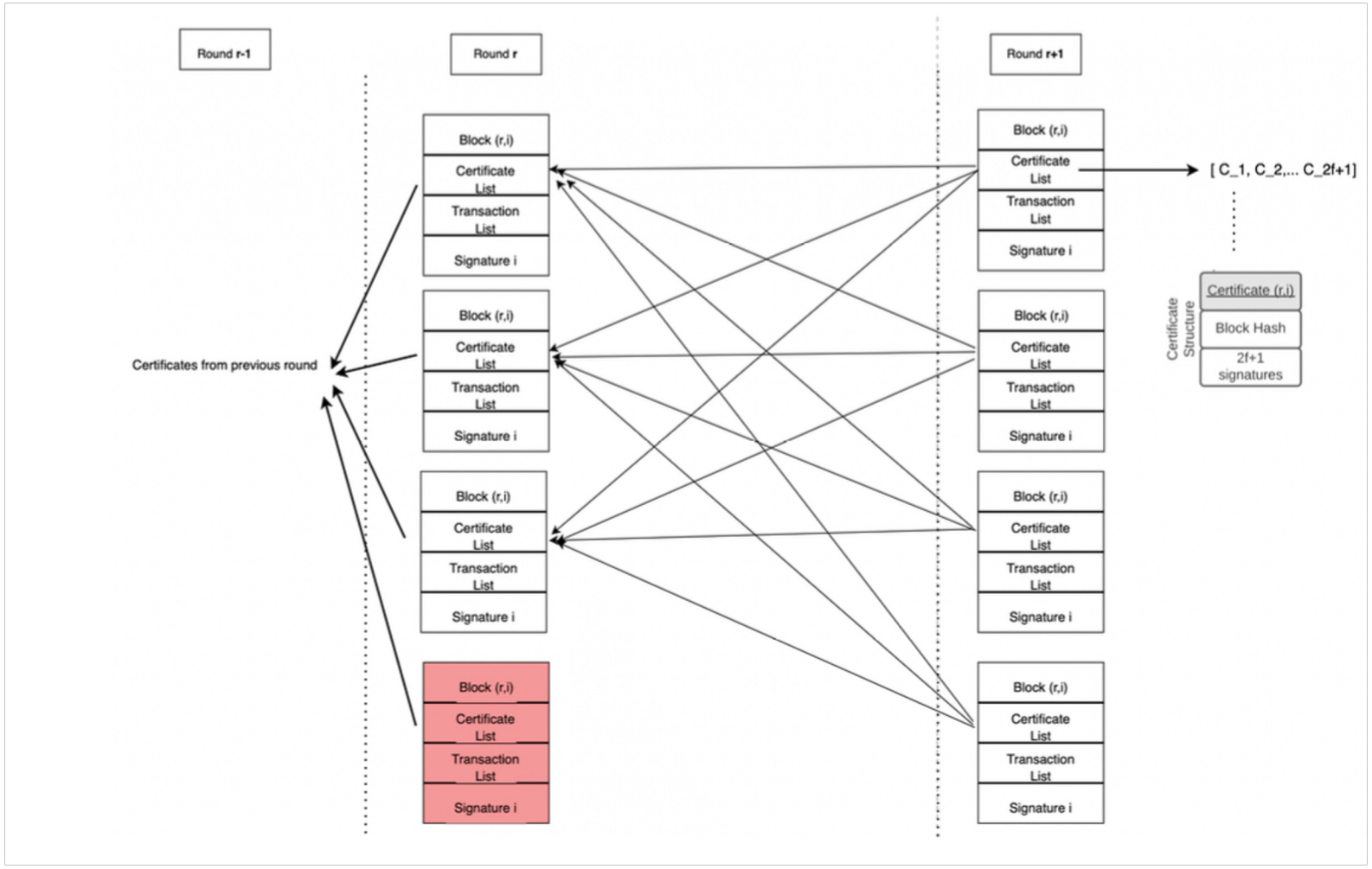
Narwhal



Narwhal

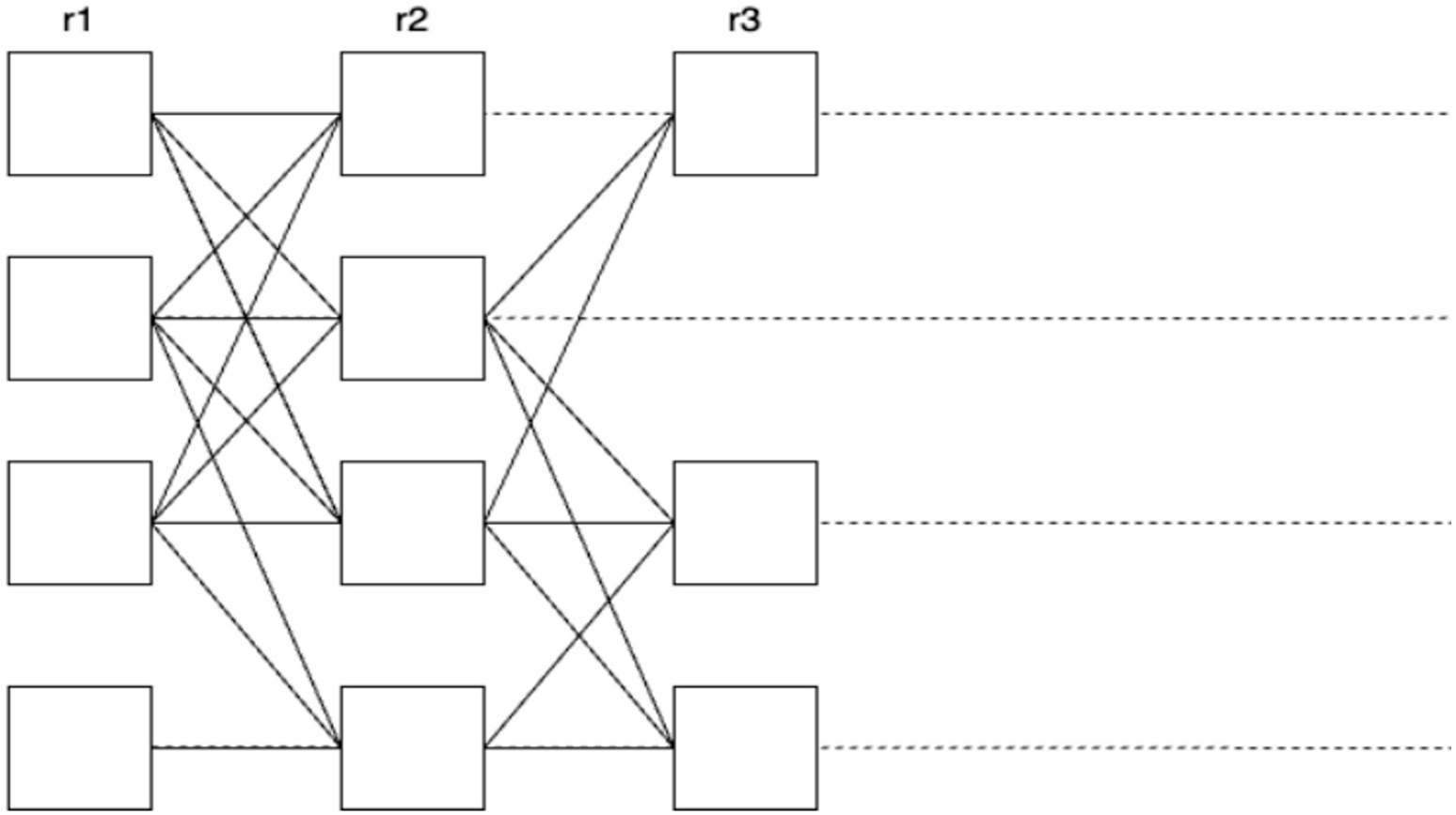


Narwhal



This is the DAG created in Narwhal

Narwhal



This is the DAG created in Narwhal

Tusk - asynchronous consensus

- A **zero-message overhead** asynchronous consensus protocol
- Tusk's theoretical starting point is DAGRider, from which it inherent its safety guarantees.
- Tusk modifies DAG-Rider into an implementable system and improves its latency.
- To remain live under asynchronous or DDoS attacks, **Tusk** was proposed in the paper.

Tusk - asynchronous consensus

•DAG-Rider is the first asynchronous Byzantine Atomic Broadcast protocol with optimal resilience, optimal communication complexity, and optimal time complexity.

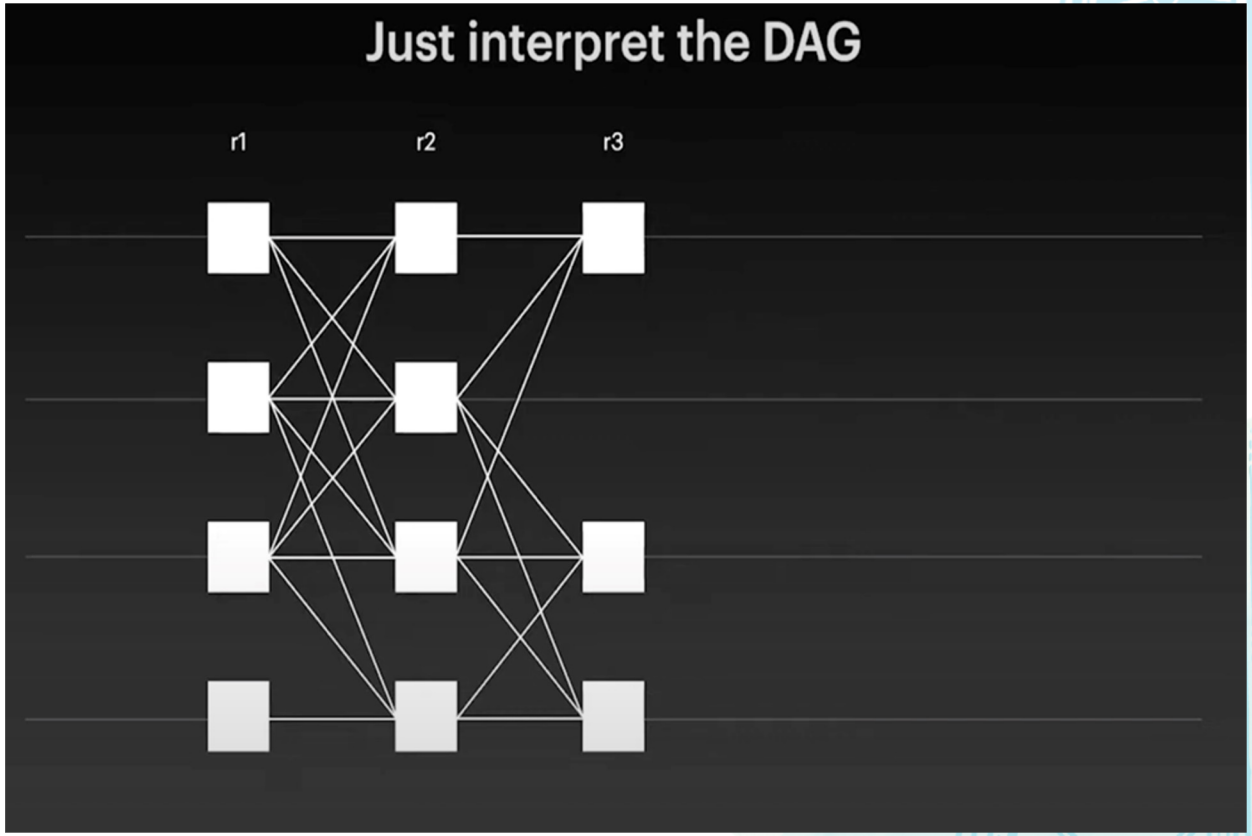
•It's ensures all correct processes' messages are decided. Its design is notable for its efficiency, modularity, and concise logic.

•The protocol consists of two layers:

- ❑ a communication layer for broadcasting proposals and forming a Directed Acyclic Graph (DAG), and
- ❑ an ordering layer for local observation and ordering of proposals with no extra communication.

Tusk - asynchronous consensus

- Tusk validators operate a Narwhal mempool, but also include in each of their blocks information to generate a distributed perfect random coin.



Tusk - asynchronous consensus

•Every validator interprets the DAG based on its view. To interpret the DAG, validators divide it into waves, each of which consists of 3 consecutive rounds.

•What is a wave ? - Three consecutive rounds form a wave

- ❏ First round: Each validator proposes its block and **causal** history (PROPOSE)
- ❏ Second round: Each validator votes on the proposal by including them in their block (VOTE)
- ❏ Third round: Validators produce randomness to elect a random leader's block.

•The order may be different since it is asynchronous.

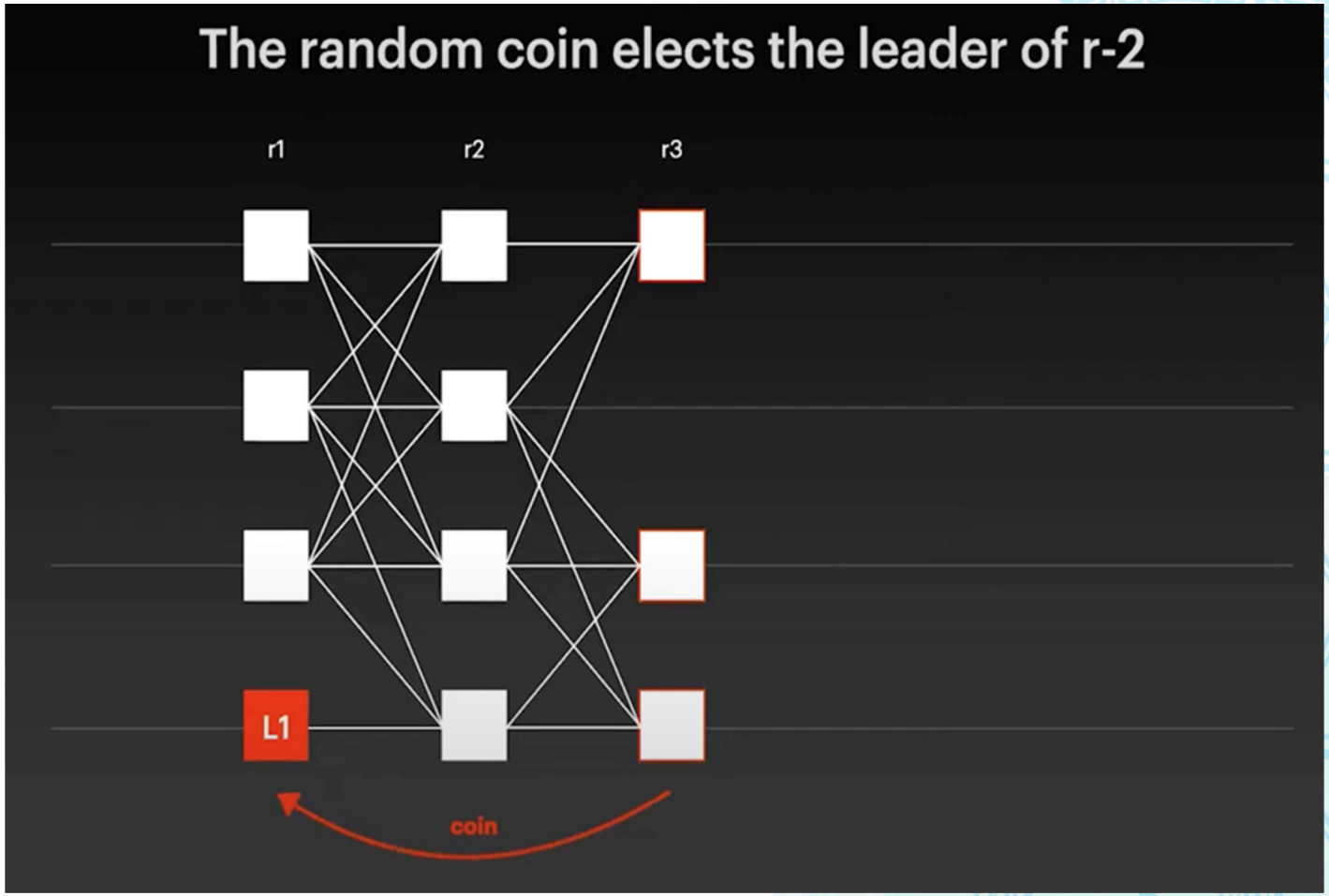
•Leader is elected and all its Sub-DAGs are committed

Tusk - asynchronous consensus

- The generation of distributed perfect random coin only happens at odd rounds such as round 3, round 5 and so on.
- The elected validators are called as Leaders

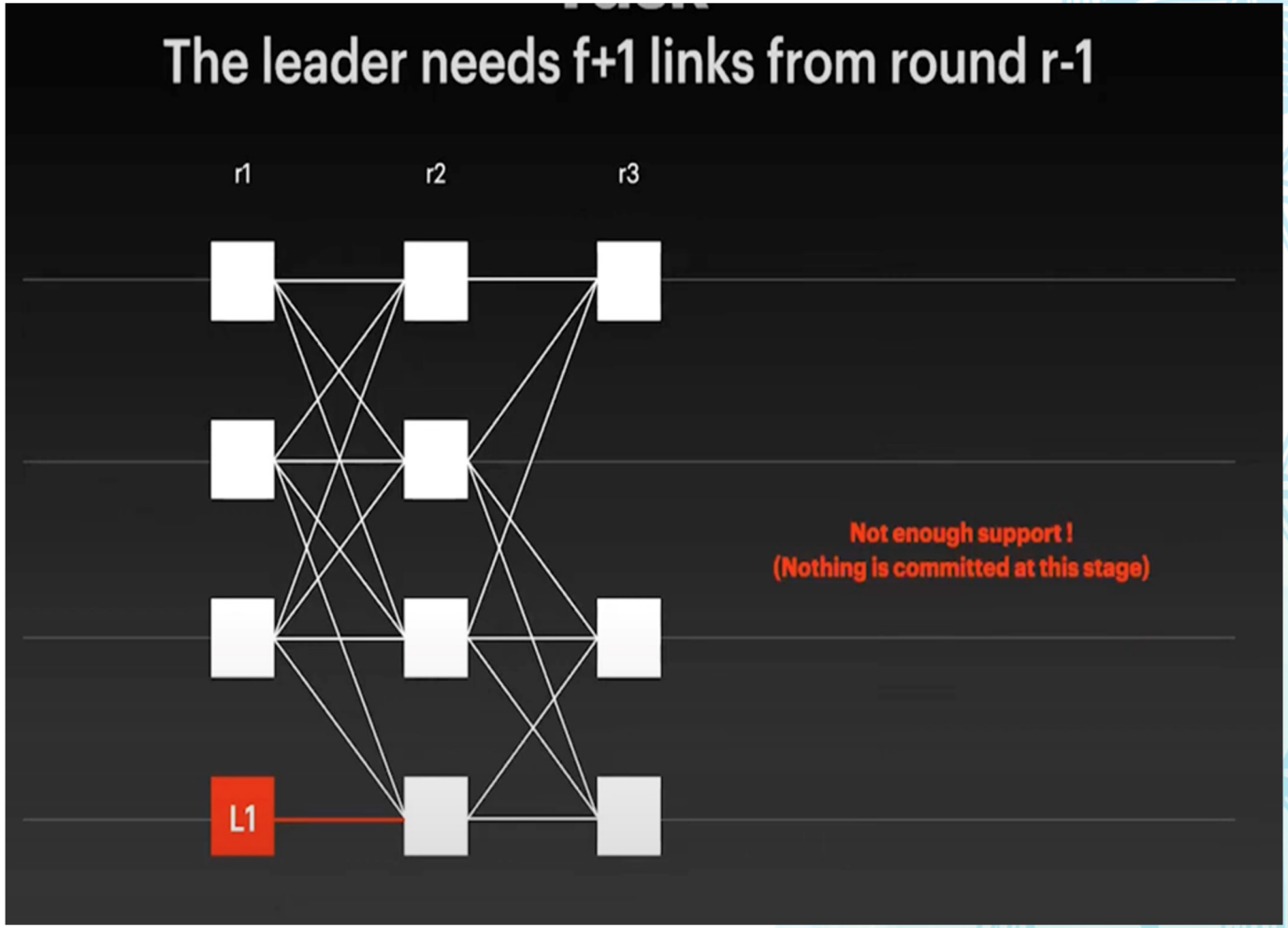
Tusk - asynchronous consensus

Here you can see when we are at round 3 which is an odd round, there is random coin generation which takes place for round r-2 i.e the round 1. Here we elect a leader and name it as L1

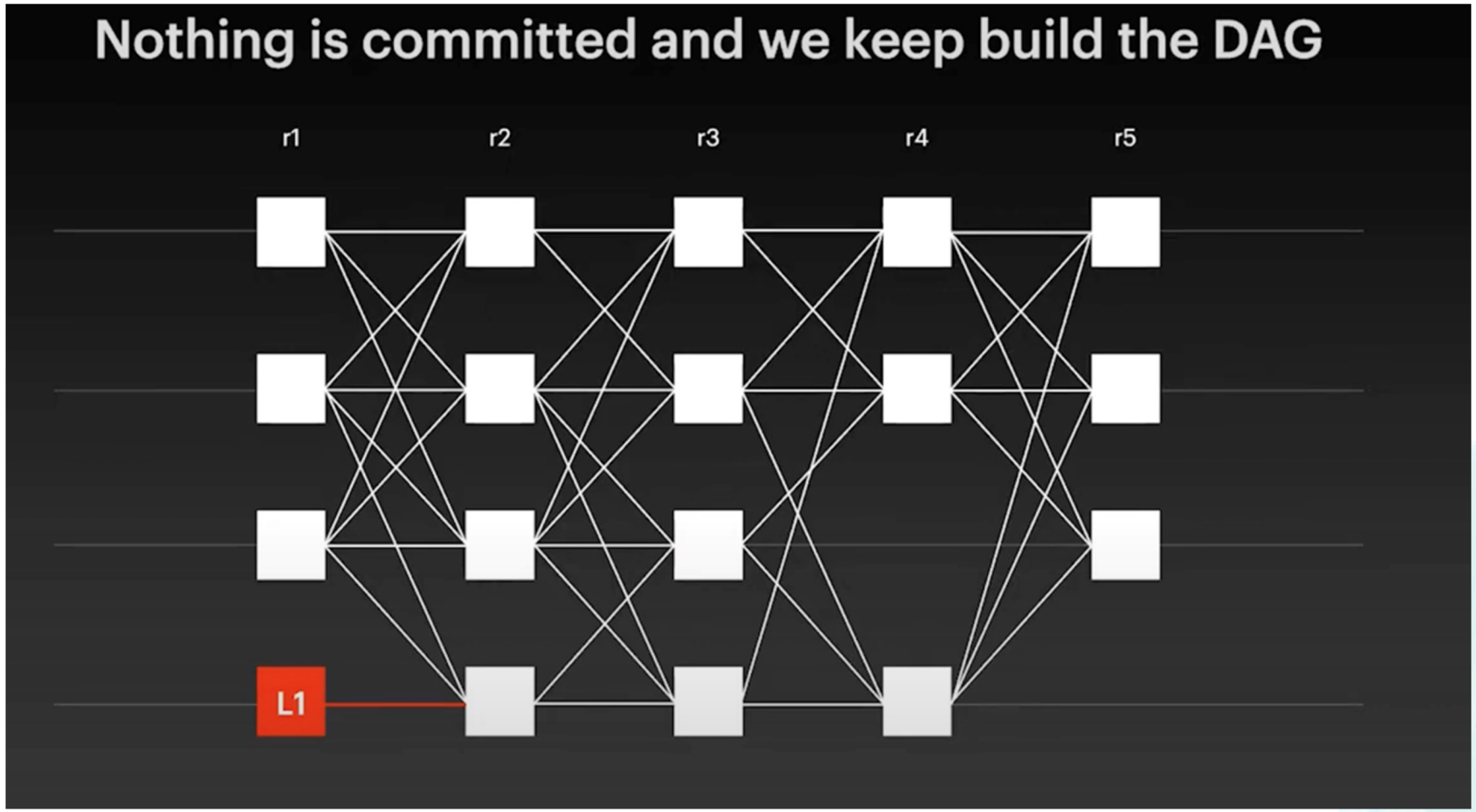


Tusk - asynchronous consensus

There are an insufficient number of blocks in round 2 (less than f + 1) that refer to/vote for $L1$ and thus $L1$ is not committed when round 3 is interpreted.

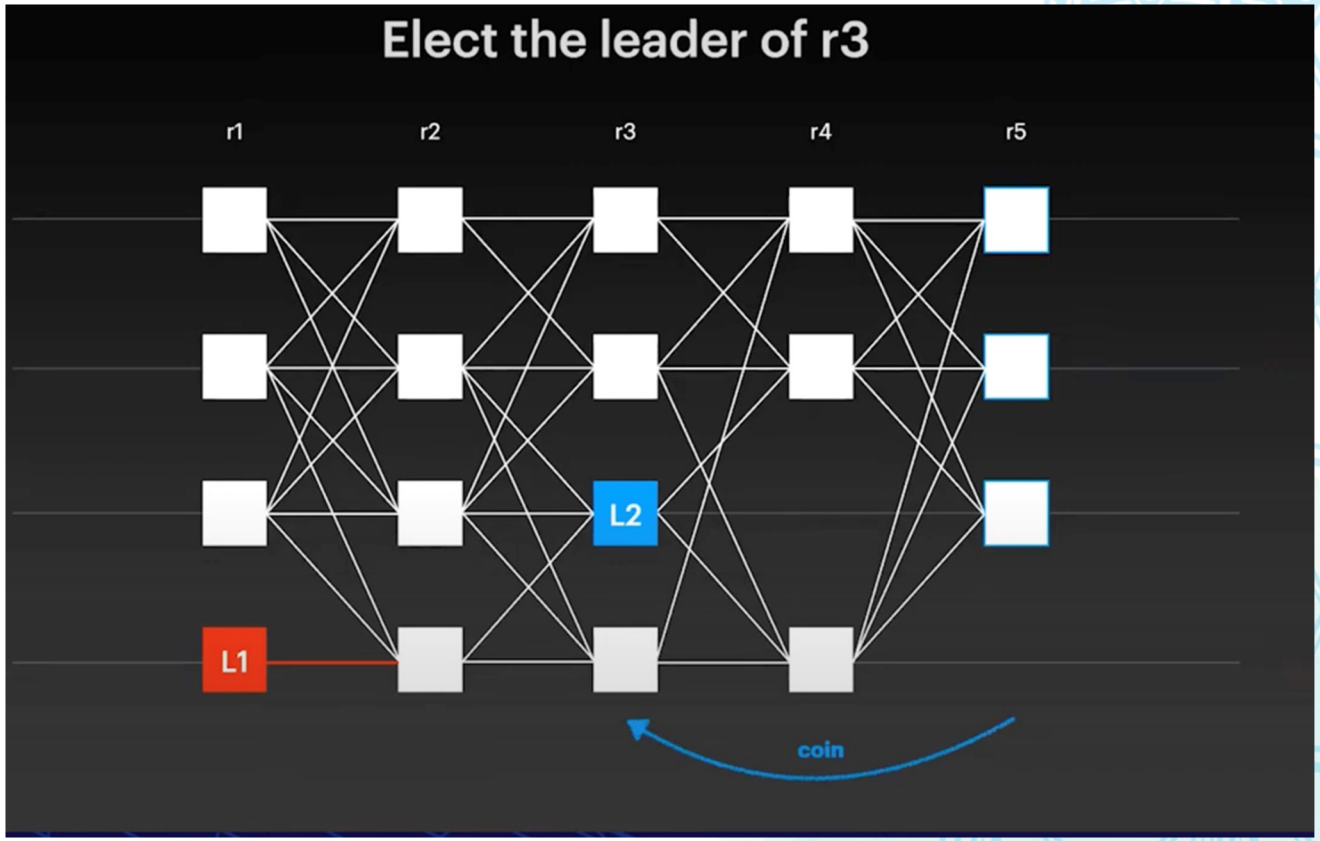


Tusk - asynchronous consensus



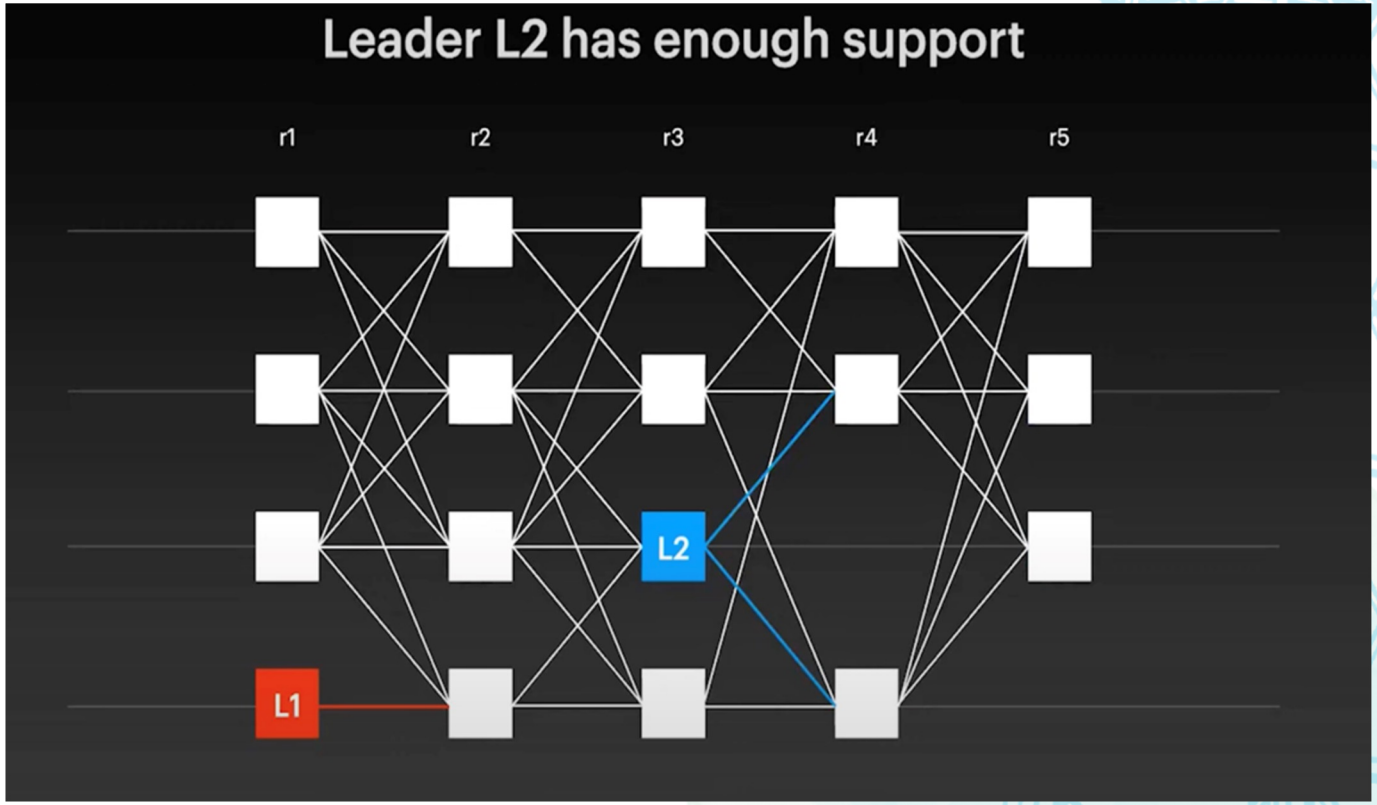
Tusk - asynchronous consensus

The elected leaders of waves 1 and 2 are determined at rounds 3 and 5, and we denote them $L1$ and $L2$, respectively



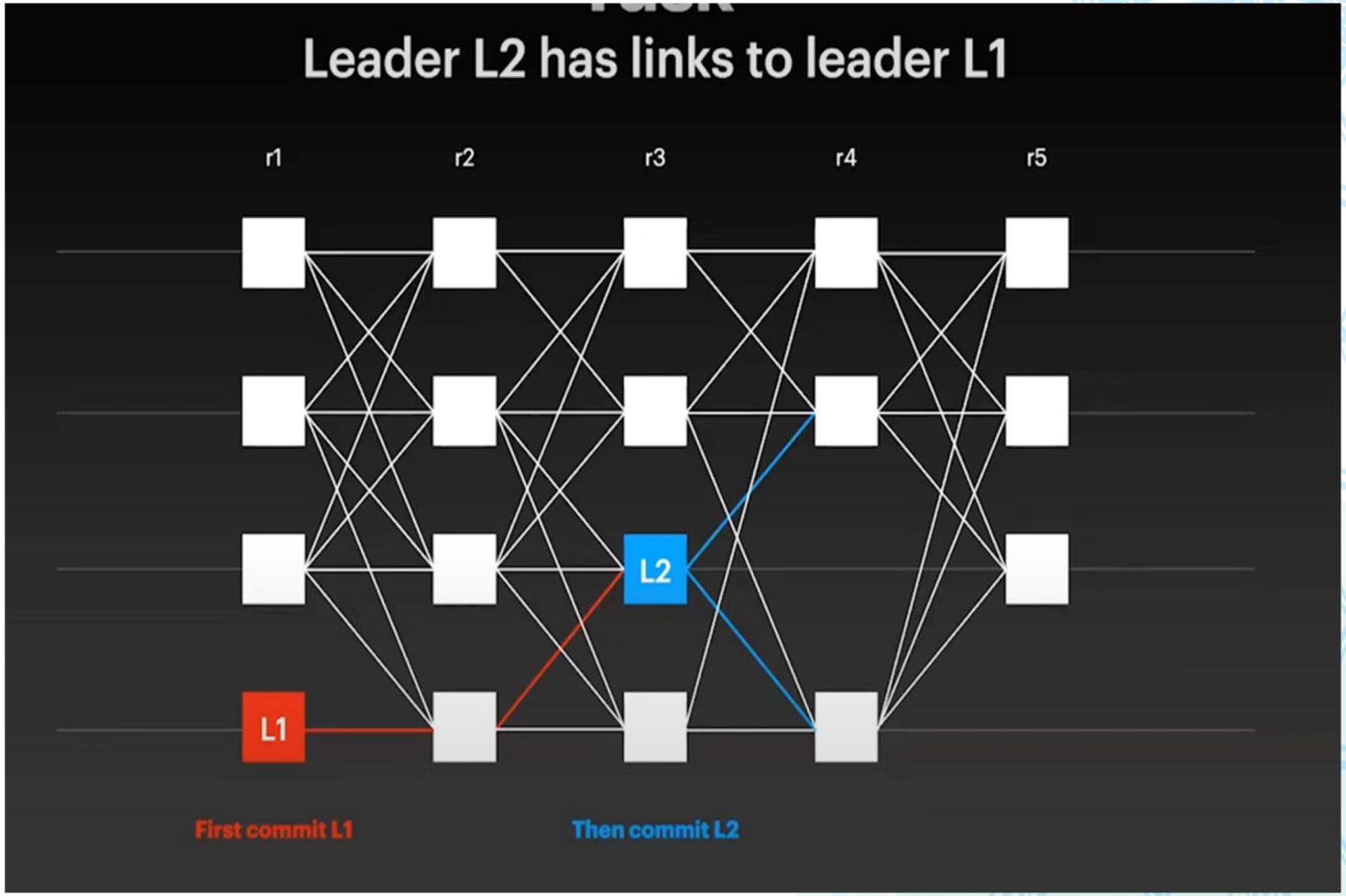
Tusk - asynchronous consensus

Since there are $f + 1 = 2$ blocks in round 4 that refer to $L2$, and as a result $L2$ is eventually committed.



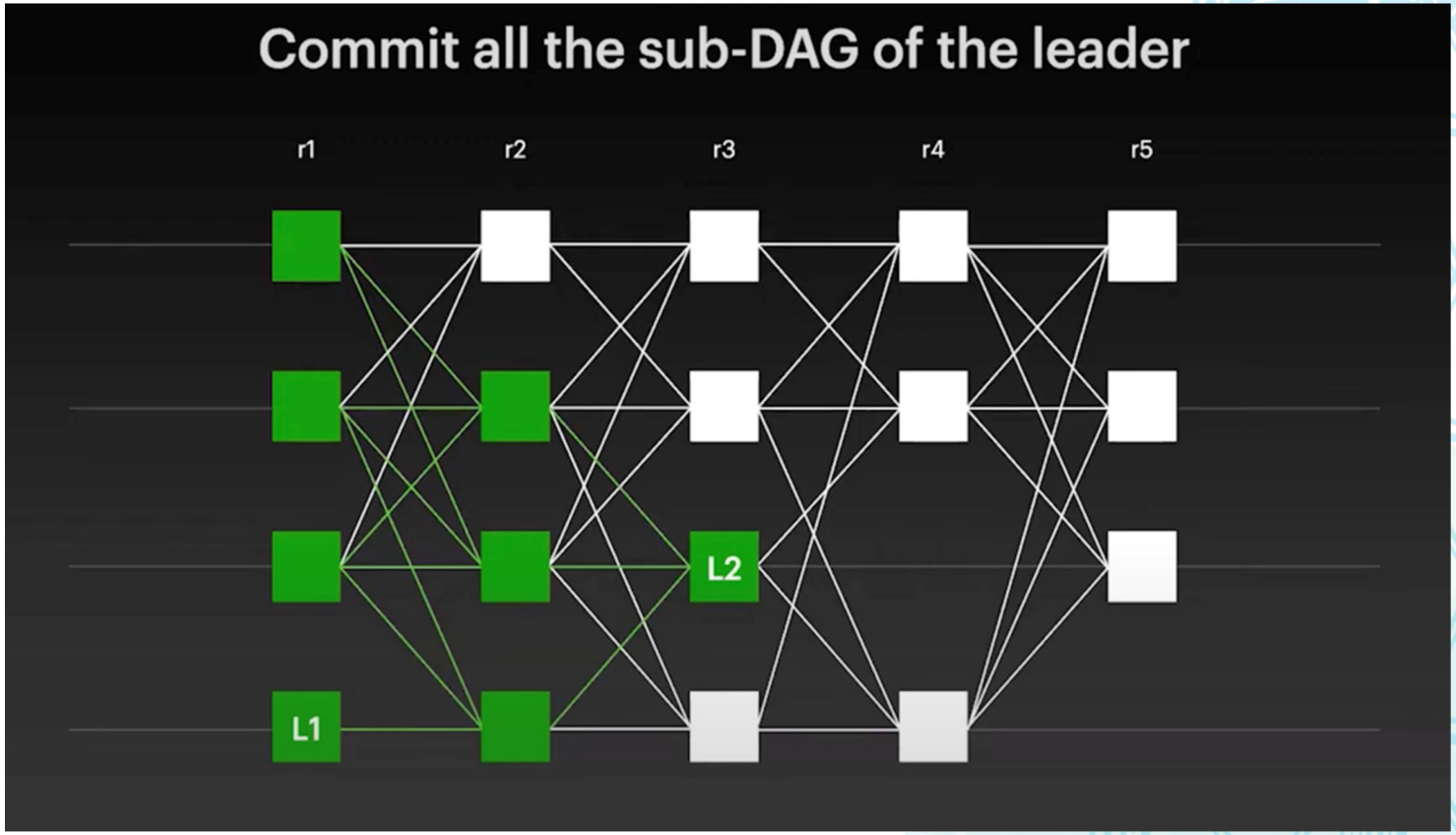
Tusk - asynchronous consensus

Since there is a path between $L2$ and $L1$, $L1$ is ordered before $L2$. Meaning that the sub-DAG causally dependent on $L1$ is ordered first (by some deterministic rule), and then the same rule is applied to the sub-DAG causally dependent on $L2$.



Tusk - asynchronous consensus

So whenever we have enough support and Commit can take place, then a lot of blocks are committed altogether(Sub-DAGs) and not just a single block, this way we are able to commit a gigantic number of transactions all at once.

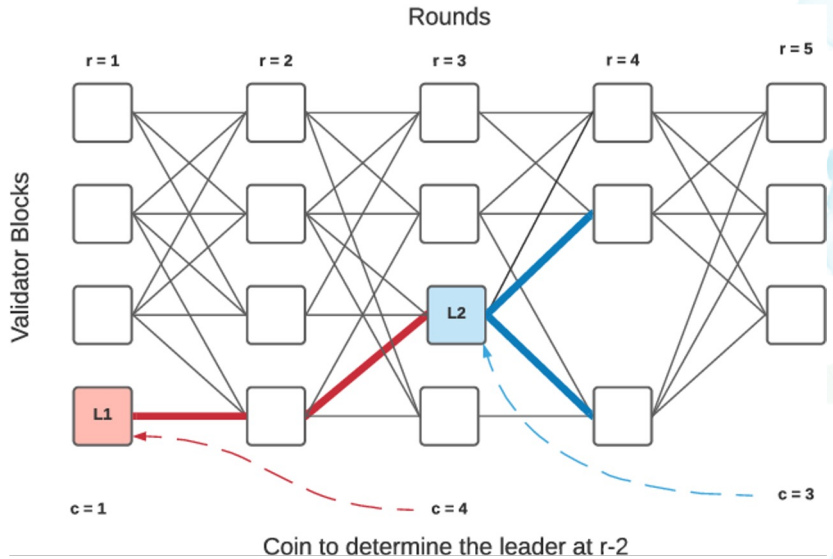


Safety and Liveness

Safety intuition

Each round in the DAG contains of at least $2f + 1$ blocks, and since any quorum of $2f + 1$ blocks intersect with any quorum of $f + 1$ blocks we get:

> Lemma 1. If an honest validator commits a leader block b in a wave i, then any leader block b' committed by any honest validator v in a future wave have a path to b in v's local DAG.



Safety

Given the above lemma, the recursive mechanism that is described above to order block leaders guarantees the following:

> Lemma 2. Any two honest validators commit the same sequence of block leaders.

Liveness

As for liveness, they use a combinatorial argument to prove

> Lemma 3. For every wave w there are at least $f + 1$ blocks in the first round of w that satisfy the commit rule.

How about the adversary which could attack the leader?

Randomness

Three rounds

The attacker doesn't know the leader after the first two round.

Thus, the $f + 1$ blocks that satisfy the commit rule are determined before the adversary learns the block leader.

Evaluation

Evaluation

- The efficacy of Narwhal is “evaluated through experiments on AWS” . Let us delve into what the authors’ setup is. As mentioned earlier, they are using AWS. More specifically, they deployed “8 m5 large instances across 5 different regions” Each of these instances “provide 10Gbps of bandwidth, 3 virtual CPUS, 2.5GHz clock for cpus, and the oberating system is Ubuntu 20.04.
- When the authors switched to their version of Hotstuff which also included Narwhal, “Batched Hotstuff” they noticed a 20x improvement over the original Hotstuff. Tusk only performed slightly worse than Narwhal in asynchronous settings.

Sources

https://en.m.wikipedia.org/wiki/File:Bitcoin.png

)https://twitter.com/work_matters/status/566706831808925696

https://images.app.goo.gl/PAD46MPGrh8YVHKeA

https://www.google.com/url?sa=i&url=https%3A%2F%2Fdepositphotos.com%2Fvectors%2Fmoses-red-sea.html&psig=AOvVaw3CqqYvOfRxSwDhdCZ7O6TL&ust=1699916618364000&source=images&cd=vfe&opi=89978449&ved=0CBIQjRxqFwoTCJj5xIXJv4IDFQAAAAAdAAAAABAD

https://st.depositphotos.com/1007168/3030/i/950/depositphotos_30306183-stock-illustration-smiling-scientist-or-professor-with.jpg

https://images.cartoonstock.com/lowres/education-teaching-pass-toll-highway_toll-e_z_pass-hard-NA200249_low.jpg

https://www.google.com/url?sa=i&url=https%3A%2F%2Ftw.123rf.com%2Fphoto_58707549_%25E8%2589%25AF%25E5%25A5%25BD%25E7%259A%2584%25E7%25B6%2593%25E7%2587%259F%25E6%25A5%25AD%25E7%25B8%25BE%25E3%2580%2582%25E5%258D%25A1%25E9%2580%259A%25E5%2595%2586%25E4%25BA%25BA%25E5%2591%2588%25E7%258F%25BE%25E5%25A2%259E%25E9%2595%25B7%25E5%259C%2596%25E3%2580%2582%25E5%25BE%25A9%25E5%258F%25A4%25E9%25A2%25A8%25E6%25A0%25BC%25E7%259A%2584%25E6%258F%2592%25E5%259C%2596%25E3%2580%2582.html&psig=AOvVaw2IBh9N8erR23TNglZa8Kk2&ust=1699917838932000&source=images&cd=vfe&opi=89978449&ved=0CBIQjRxqFwoTCPjJ_cnNv4IDFQAAAAAdAAAAABAD

https://expolab.org/ecs265-fall-2023/slices/HotStuff%20Presentation.pdf

DAG Based Consensus Protocols: An Introduction https://www.youtube.com/watch?v=v7h2rXNtrV0

Narwhal and Tusk - Alberto Sonnino https://www.youtube.com/watch?v=K5ph4-7vvHk&t=7s

Narwhal and Bullshark: DAG-based Mempool and Efficient BFT Consensus https://www.youtube.com/watch?v=xKDDuPrYUag

Direct Acyclic graph https://ashourics.medium.com/directed-acyclic-graph-dag-vs-blockchain-b16a85a95c30

Directed Acyclic Graph (DAG) vs Blockchain https://ashourics.medium.com/directed-acyclic-graph-dag-vs-blockchain-b16a85a95c30