Old Algorithms for New Problems

Theory, Practice, & Stories about Tendermint

/ AdiSeredinschi
adi@informal.systems

M. Nweke @ artstation.com

SAPIENS

A Brief
History of
Humankind

YUVAL HARARI

M. Nweke @ artstation.com

About "Sapiens" book

C. R. Hallpike is an anthropologist who once wrote a review of a young author's new book on the history of humans. The review states:

> It would be fair to say that whenever his facts are broadly correct they are not new, and whenever he tries to strike out on his own he often gets things wrong, sometimes seriously. . . . [It is not] a contribution to knowledge.

Two things are notable here.

One is that the author, Yuval Noah Harari, has sold over twenty-eight million books, making him one of the bestselling contemporary authors in any field, and his book *Sapiens*—which Hallpike was reviewing—is the most successful anthropology book of all time.

The other is that Harari doesn't seem to disagree with Hallpike's assessment.

- indeed, Harari cut some corners

- yet he knew his goal & audience
* communicate to the general public
* inspire, educate, elevate others
* essential <-> non-essential
* simplify

Simplifications are not always bad

- simplifications as a starting point
* not an end or a scientific result

- simplifications can also be important when applying in practice a theory or an algorithm

- especially in distributed systems

Roadmap

Old Algorithms for New Problems

1. Simplify

2. Productionize

3. Commoditize

M. Nweke @ artstation.com

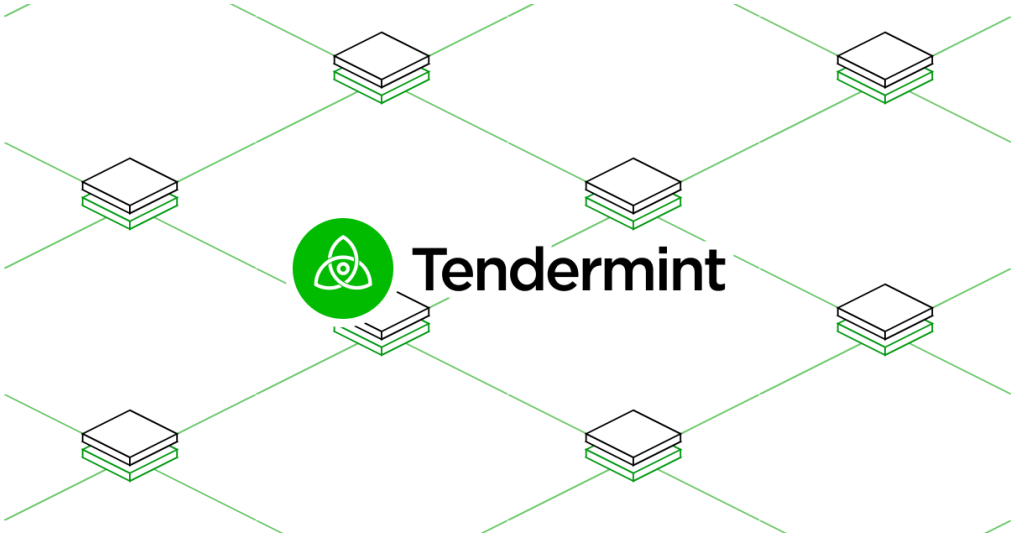
Simplifying,
and the Tendermint algorithm

- represents the meeting of theory <> practice
circa 2013

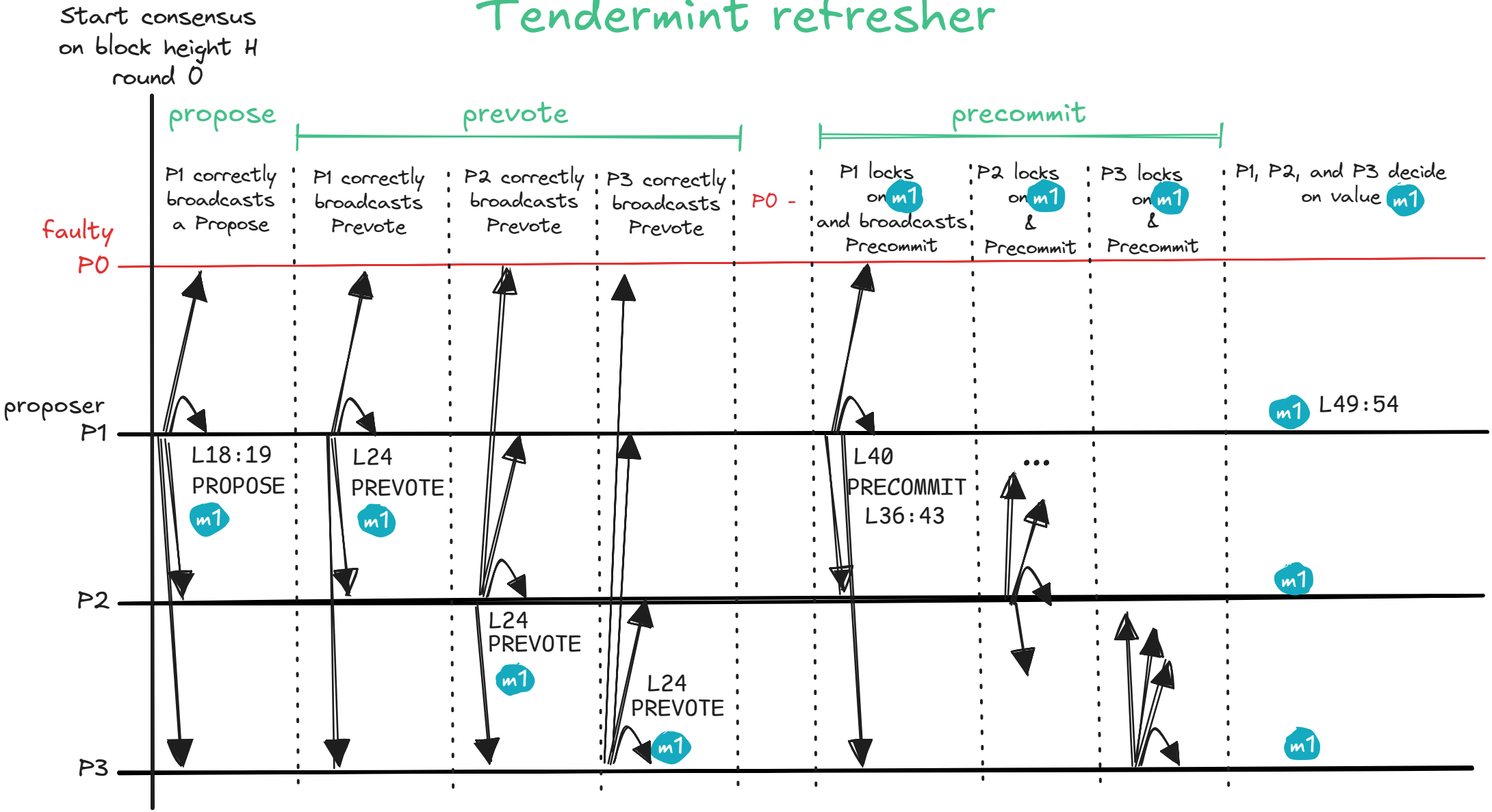
- some constraints brought from practice:
* for decentralized ledgers
* builds on a gossip network
* rotate leaders all the time
* PoS & membership

- main novelty:
* the simplified termination mechanism
* unified graceful & degraded paths

- like "Sapiens" --
it was wrong the first couple of times..



Tendermint refresher



Simplifications beyond the algorithm

* A consensus engine

* Originally called "Tendermint Core"

* Implements Tendermint algorithm

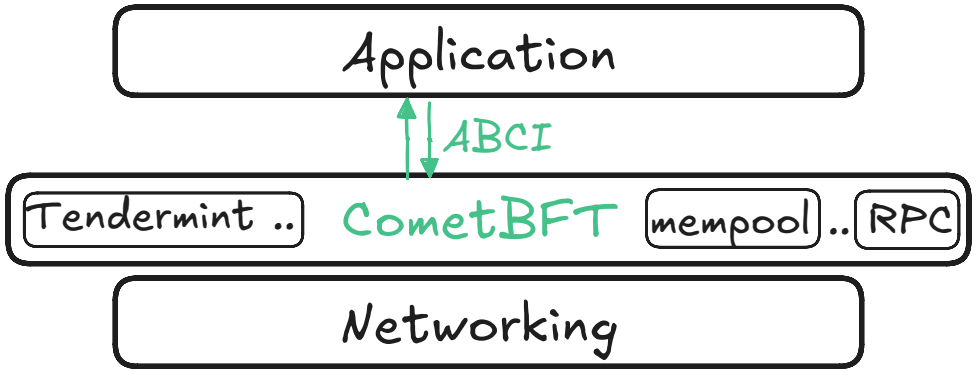
* A lot of "batteries included"

* discovery
* RPC server
* mempool
* indexer
* write-ahead log

→ Simplified adoption by users

ABCI interface

* a simple broadcast/deliver API
* the "narrow waist" of decentralized applications



Roadmap

Old Algorithms for New Problems

1. Simplify
2. Productionize
3. Commoditize

https://www.artstation.com/artwork/0n4d4G

What does it mean to productionize?

Generally

* Handling tens to hundreds of corner-cases
* Sanitizing assumptions made during during design
* Making clear promises w.r.t. performance (& other dimensions)

Specifically

* Ensuring scale of ~100 validators
* Configuration params, all the "batteries" included
* Logs, metrics & observability
* Timeout commit
* The gossip property

Regular maintenances and releases

* New features & improvements -> new problem statements
  - interoperability
  - mempool
  - abstraction

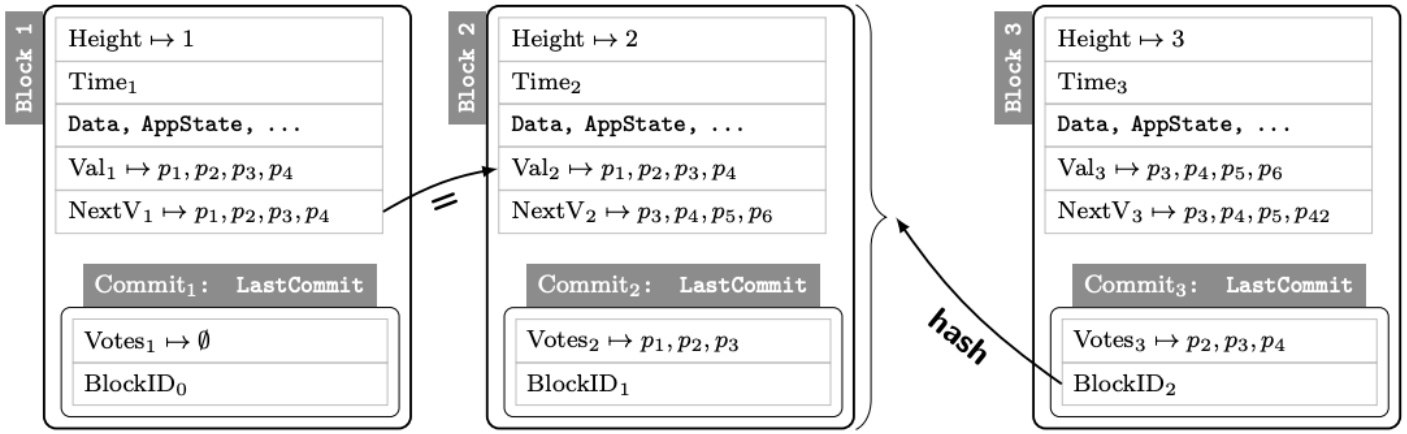
New problem: Interoperability

"Make two CometBFT applications interoperate"

IBC protocol & light clients

- Why is it challenging?
(1) SMR-to-SMR communication is O(n^2)
(2) Dynamic validator set changes

- Light client protocol allows verification of
(a) block headers that validators produced
(b) application state



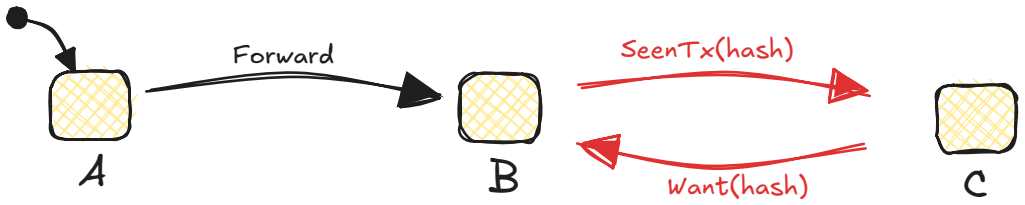
https://www.ibcprotocol.dev/

New problem: Mempool

"The CometBFT mempool is too bandwidth intensive"

CAT mempool

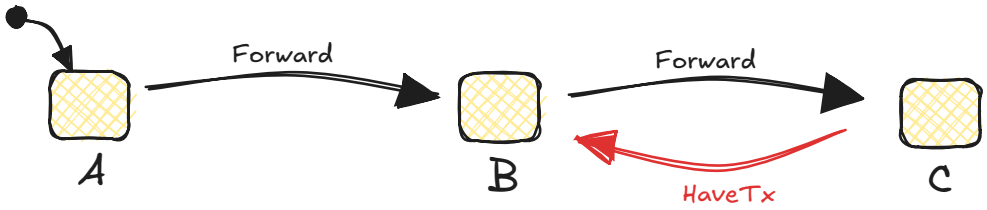
* Content-addressable transactions
* Used in Celestia blockchain
* Push-pull model



* Impacts latency
* Works better for larger transactions

DOG protocol

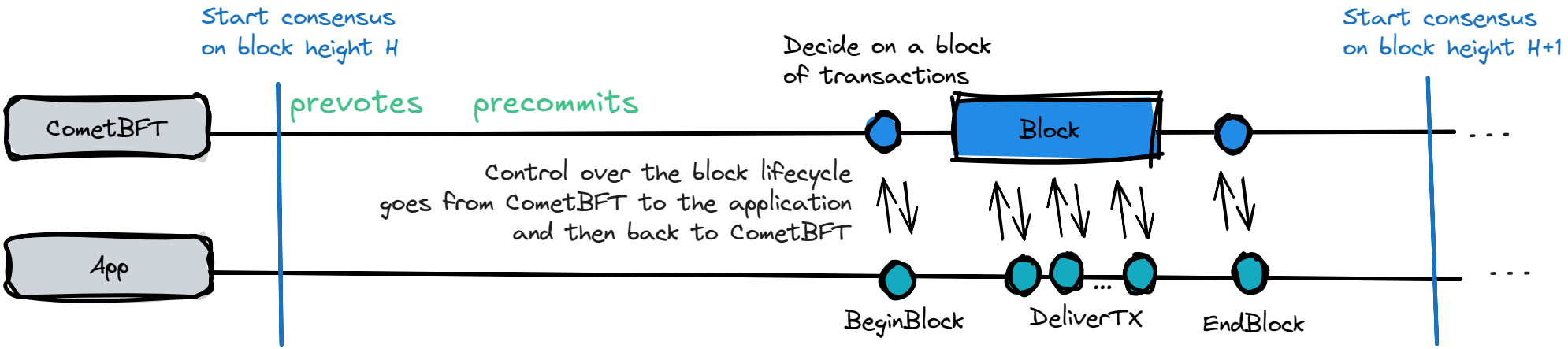
* Dynamic Optimal Graph
* State of the art in Comet mempool efficiency
* Push model w/ redundancy control



* Regularly "trims" redundant dissemination paths
* Requires tuning

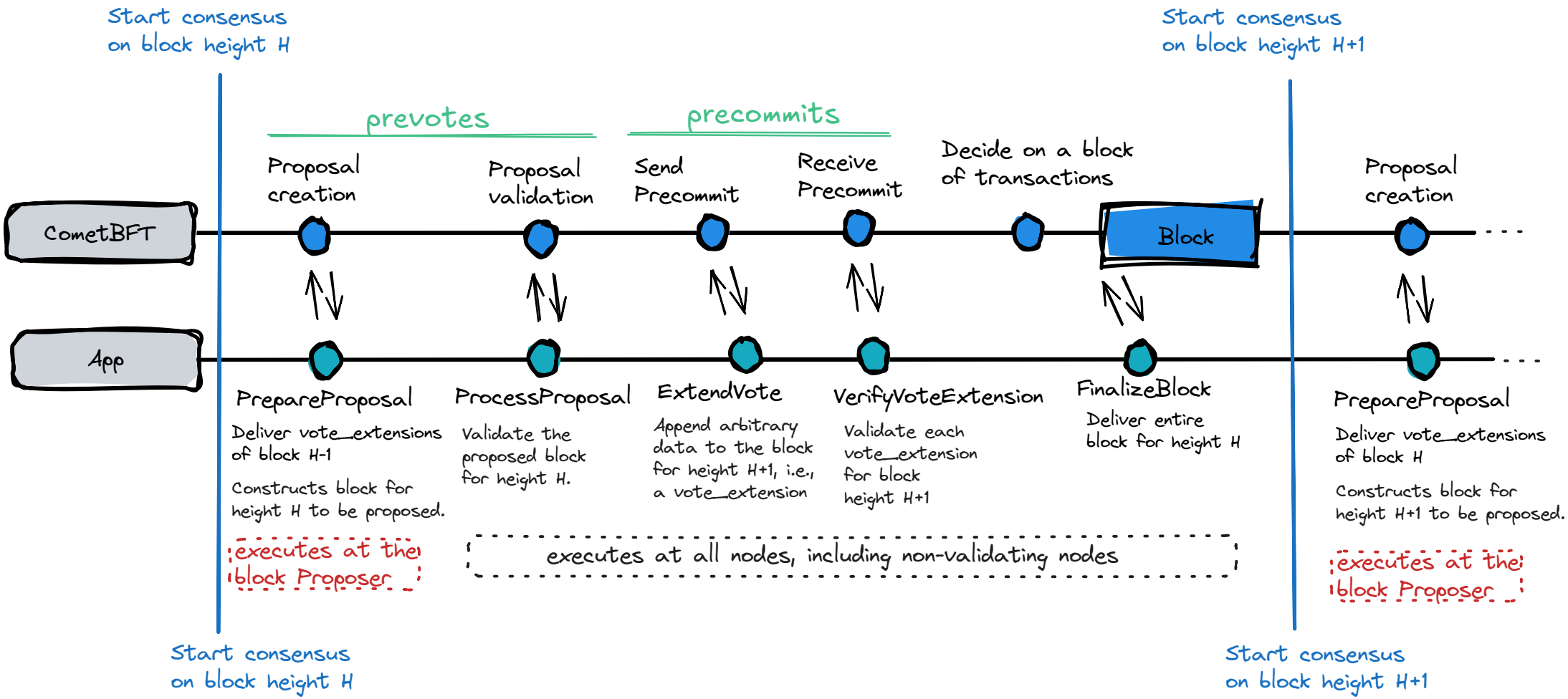
New problem: Abstractions

"ABCI is too restrictive, the application needs more control"



Control flow is simple & sequential, at the end of block lifecyle

ABCI v0



ABCI v2

Old algorithm -- with slight modifications -- for a new problem

Productionize - summary

We discussed:

* Timeout commit
* The gossip property
* IBC & interop
* Mempool 🐈 & 🐕
* ABCI extensions

We did not discuss today:

* Non-mempool efficiency
* Transport stack (e.g. QUIC, libp2p)
* Decorrelation of various sub-systems called reactors

Why is it difficult?

- Requires specific knowledge
- Extensive E2E and QA testing
- Inertia: changes versus new features

→ Commoditize

Roadmap

Old Algorithms for New Problems

1. Simplify
2. Productionize
3. Commoditize

https://www.doscher-design.com/

What does it mean to commoditize?

Malachite

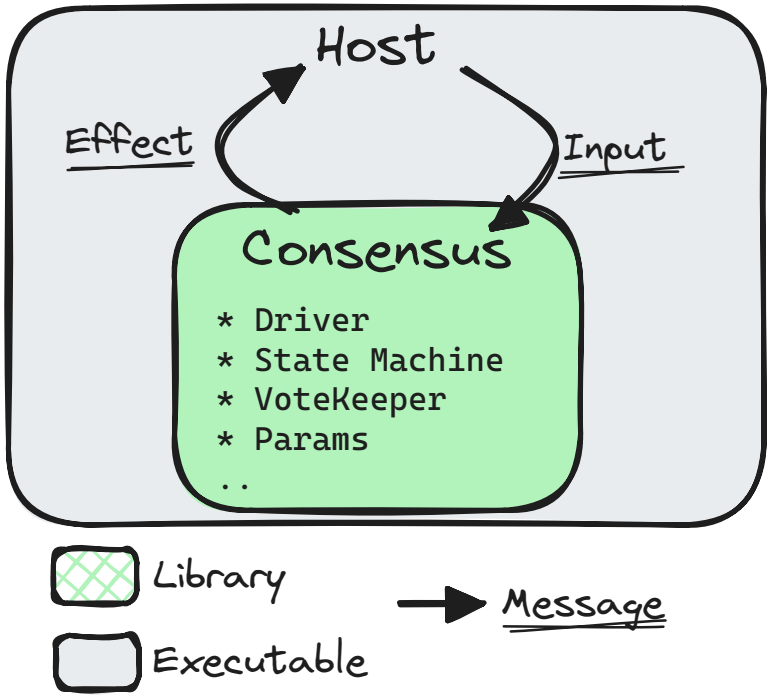
Both implement the Tendermint algorithm

OSS & Long-term maintenance

Scale: O(100s) nodes

Go	Rust
Monolithic	Modular
Many "batteries" included	Small & extensible
Build blockchains (i.e., Layer 1s)	Build whatever: sequencers, provers, .. ?

The most flexible consensus API in the world



Entrypoint

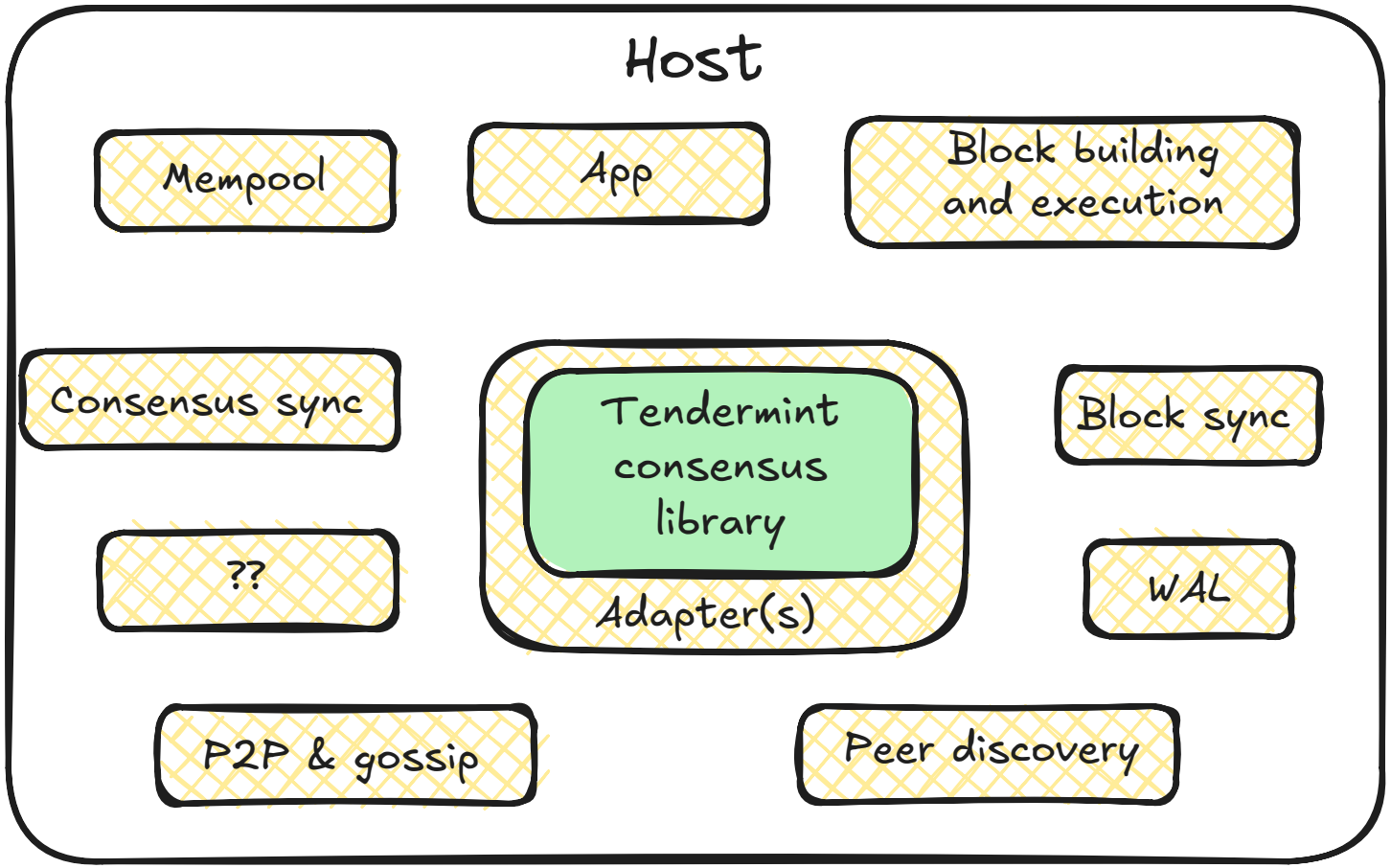
```
malachite_consensus::process!(
event: Event::StartHeight(1),              // Event to process
state: &mut state,                         // Consensus state
on: effect => handle_effect(effect)        // Effect handler
)
```

Inputs for the consensus library:

```
enum Event {
StartHeight(Height)
TimeoutElapsed(Timeout)
ProposeValue(Height, Round, Value)
ReceiveVote(SignedVote)
ReceiveProposal(SignedProposal)
ReceiveProposedValue(ProposedValue)
}
```

Malachite-based application architecture

- Lean, simple core
- Actor-based model
- Message passing
- Lower entry barrier
- We don't know what is needed next



Open areas of R&D

* Starknet L2 dec. sequencer & Quint specs
* Integration with Reth
* Upgradeability

new problems

* Direct validator links (not gossip)
* Node discovery & DOG protocol
* Loopback
* Algorithm variants
  - Vote Extensions
  - 5f + 1
  - Multi-proposer
  - Signature aggregation

It should be faster and less error-prone to demonstrate these extensions with Malachite than with other libraries

Informal Systems AG

* Lausanne, Zurich, Toronto, Berlin ..

* Core R&D, formal methods, and security audits in
the decentralization industry (Cosmos, Ethereum, Bitcoin universe)

Malachite
CONSENSUS KERNEL

QUINT

ibc-rs
The Rust implementation of the Inter-Blockchain Communication Protocol
IBC

Comet

CYCLES
The Open Clearing Protocol

Hermes
Rust relayer implementation for the Inter-Blockchain Communication Protocol
IBC

We are hiring!
https://informal.systems/careers

References

"Same as Ever," M. Housel

"Sapiens," Y. Harari

Braithwaite, Sean, et al. "A tendermint light client." arXiv preprint arXiv:2010.07031 (2020).

https://github.com/cometbft .. /docs/references/architecture/adr-119-dog-mempool-gossip.md

https://github.com/cometbft/cometbft .. docs/rfc/tendermint-core/rfc-013-abci%2B%2B.md

"The gossip property in Tendermint" https://www.adi.monster/the-gossip-property-in-tendermint/