Sydney

Red Belly Snake

Red Belly Blockchain

Vincent Gramoli

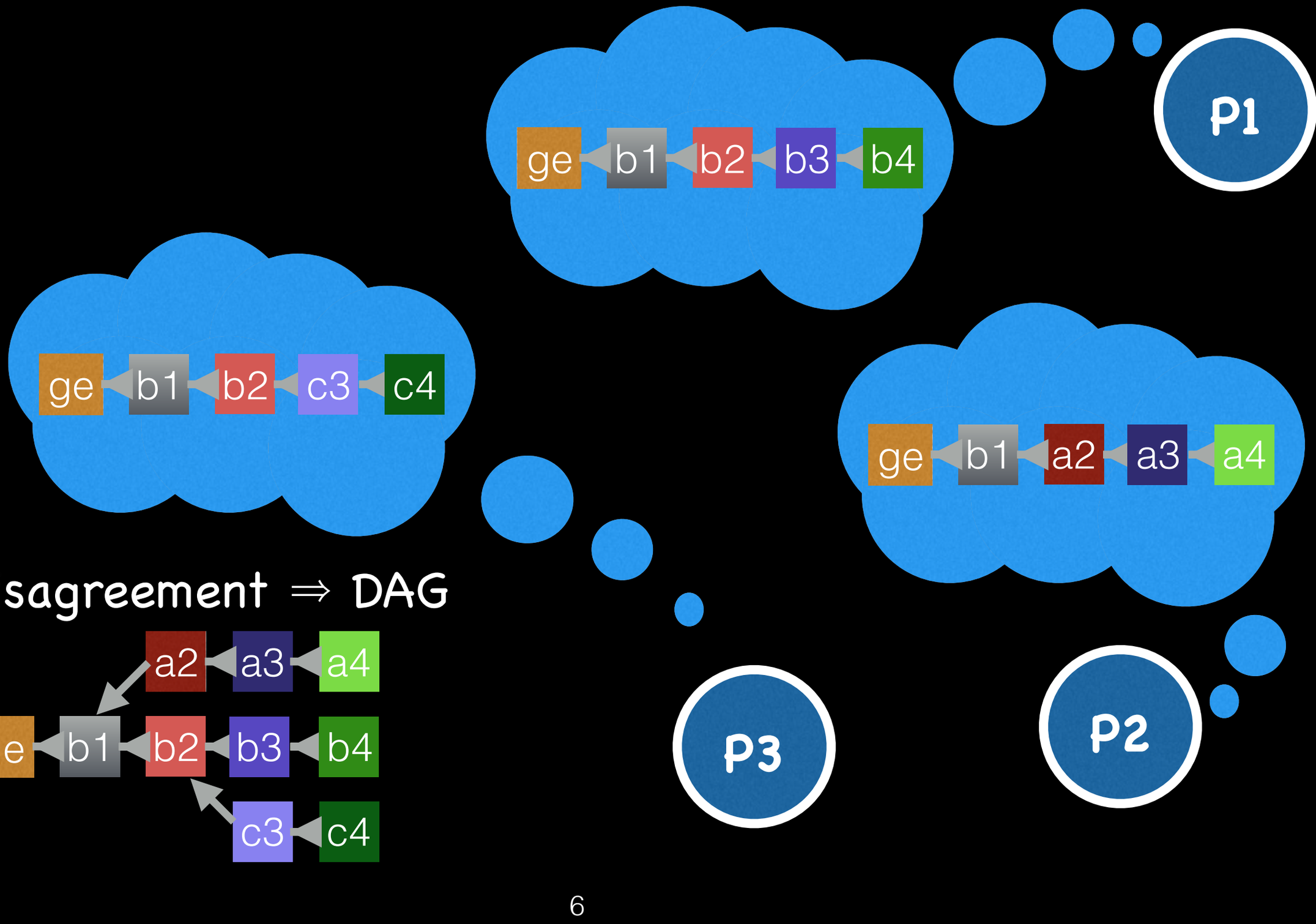
(Special thanks to Tyler Crain, Mikel Larrea, Chris Natoli, Michel Raynal, Guillaume Vizier)

Roadmap

- Context: Blockchain
- Problem: Man-in-the-Middle Attack
- Solution: Blockchain consensus
- Illustration: Red Belly Blockchain
- Experimental Results

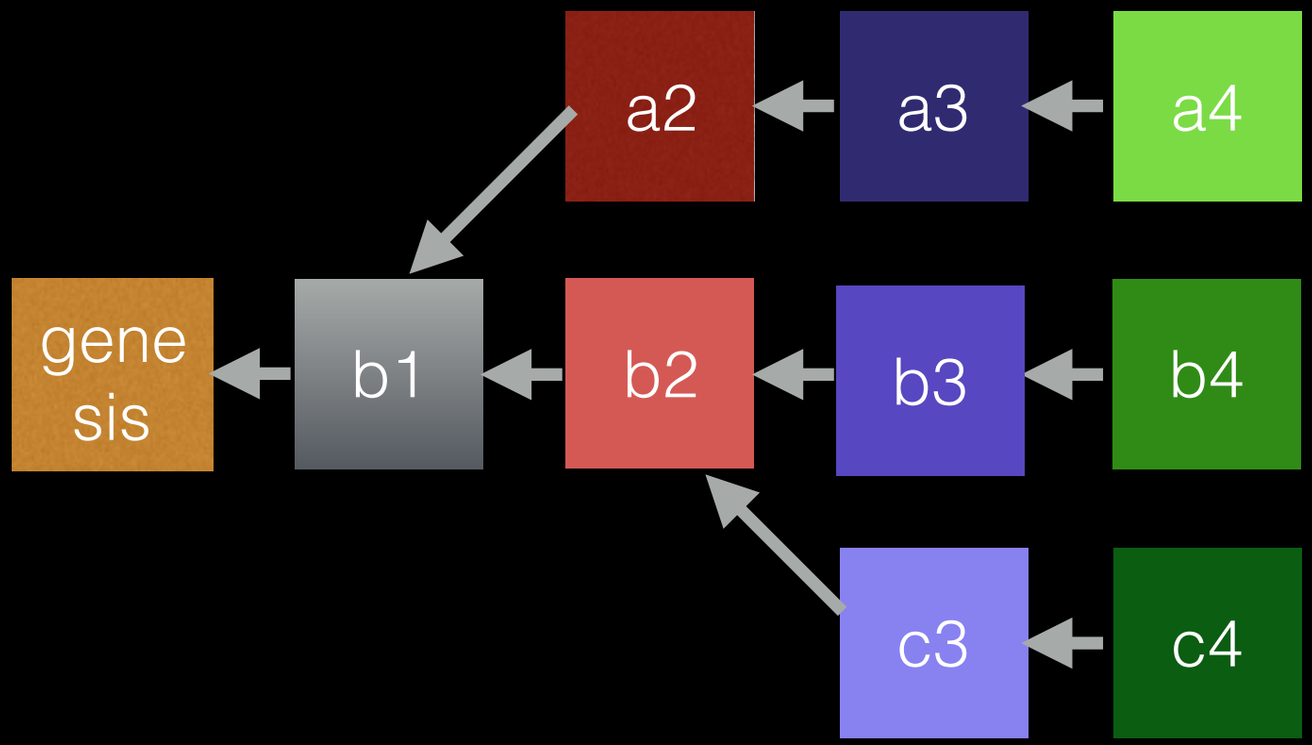
What is a blockchain?

Blockchain



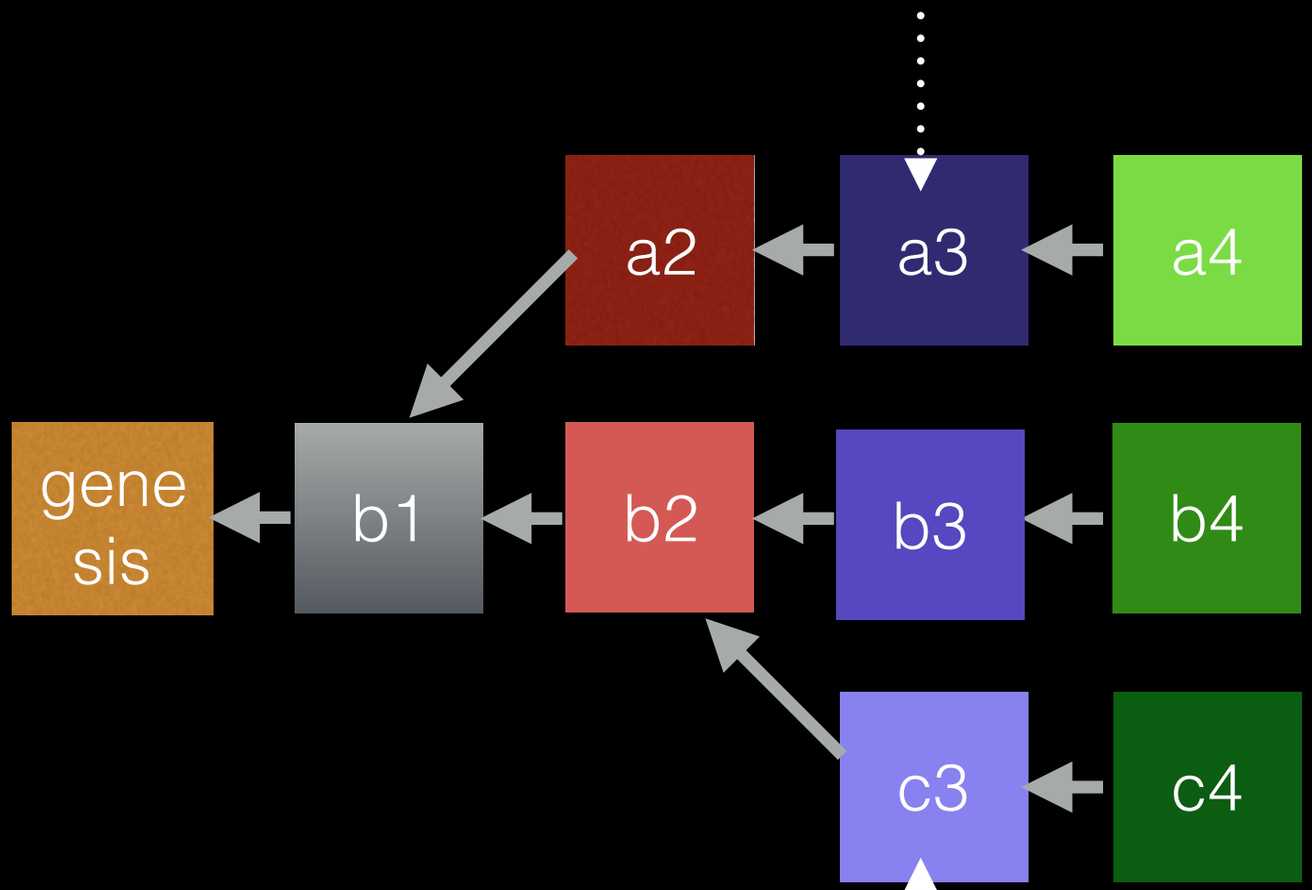
Disagreement ⇒ DAG

Blockchain (con't)



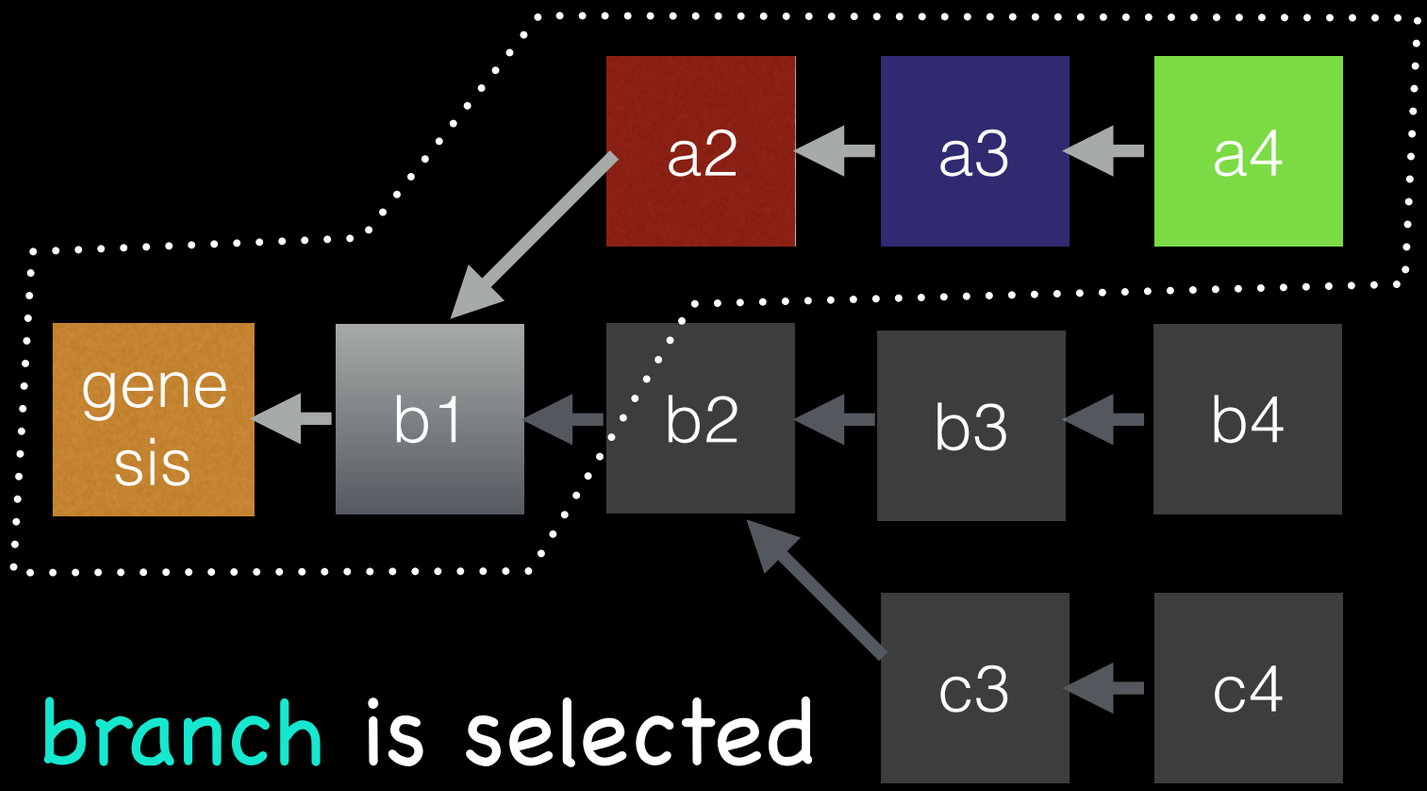
Blockchain (con't)

$\sigma_A(t_A)$: Alice gives all her coins to Bob



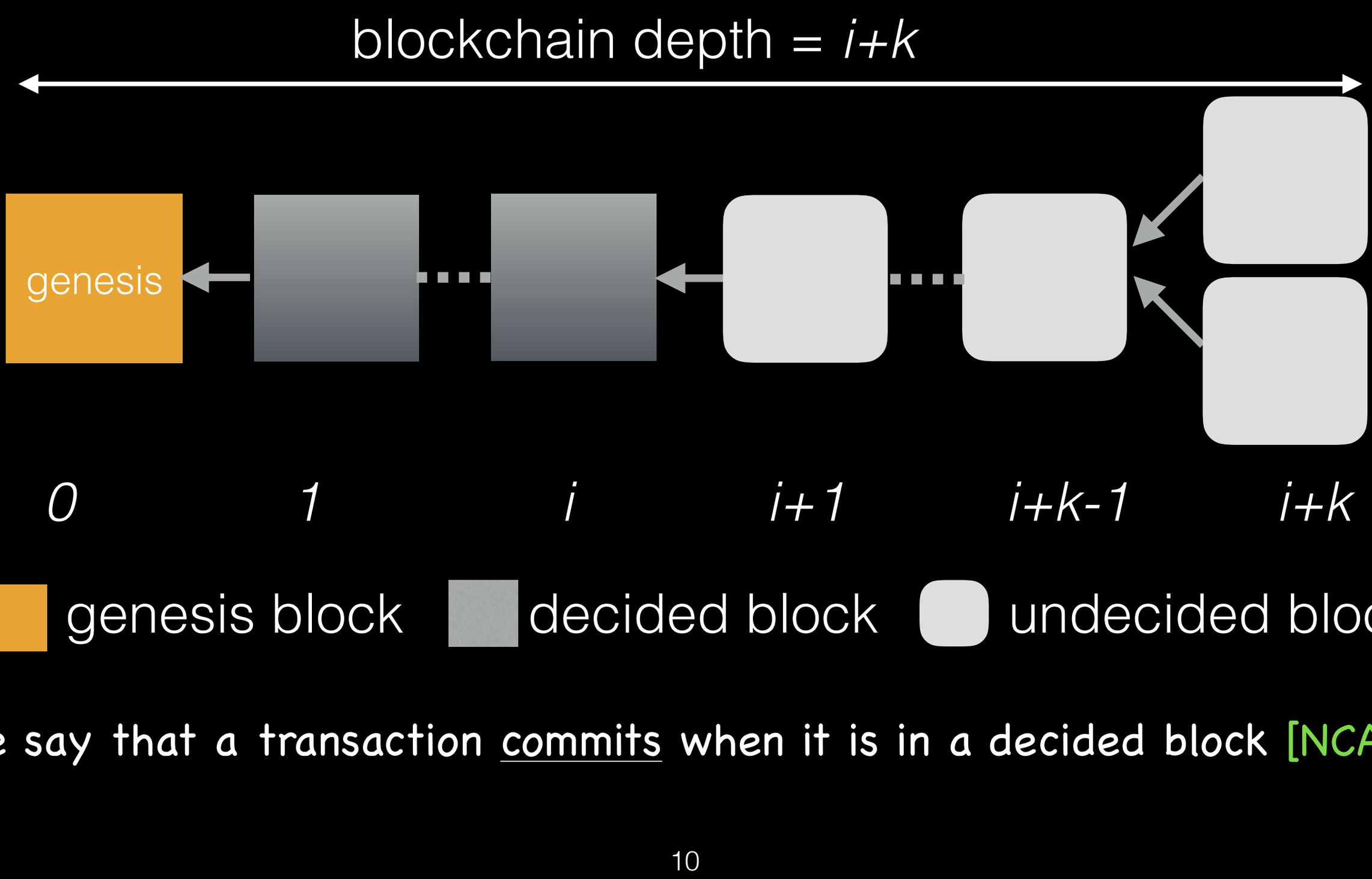
$\sigma_A(t_A')$: Alice gives all her coins to Carole

Blockchain (con't)



One branch is selected based on its length, the weight of its subtrees, its content...

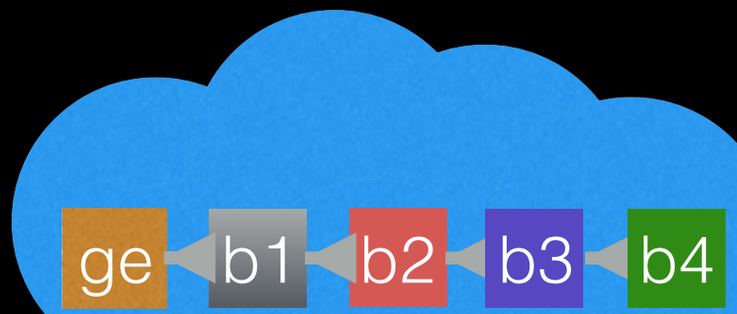
Blockchain (con't)



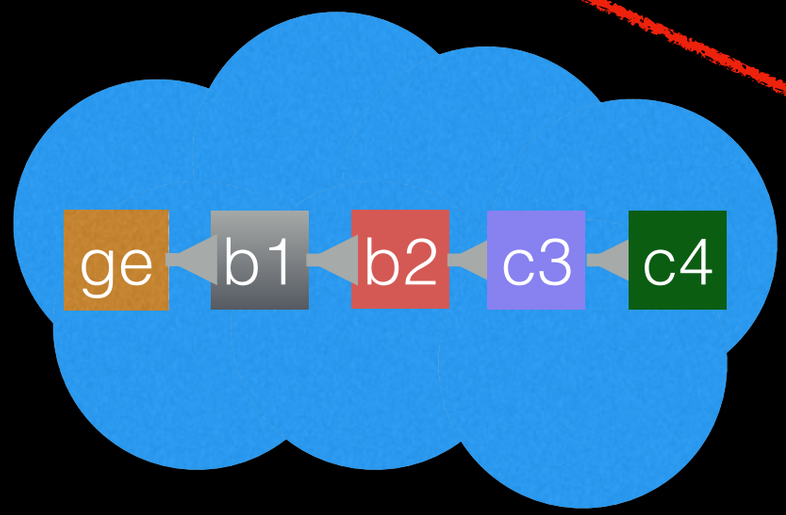
We say that a transaction commits when it is in a decided block [NCA'16]

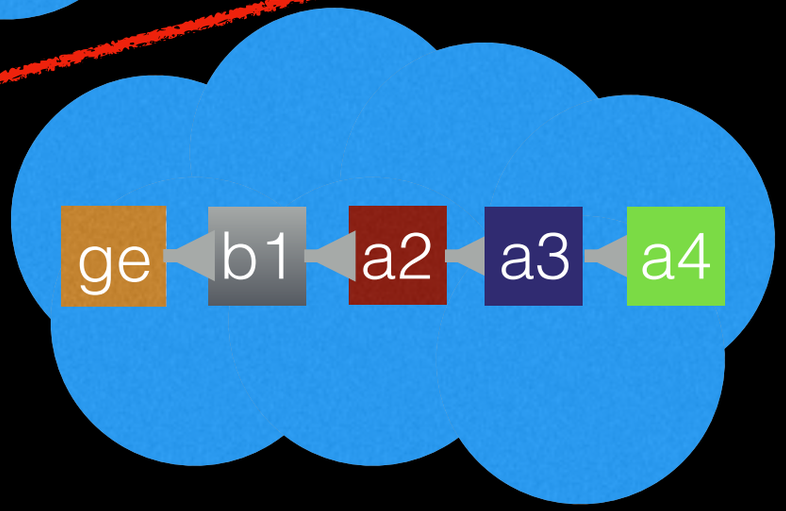
Attacking Ethereum [SRDS'18]

1. MitM Attack

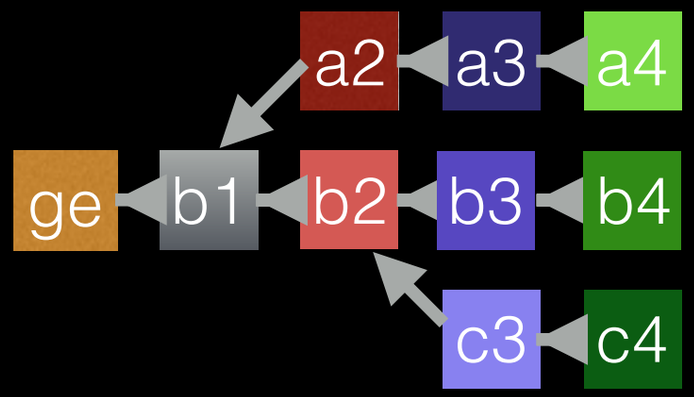


P1





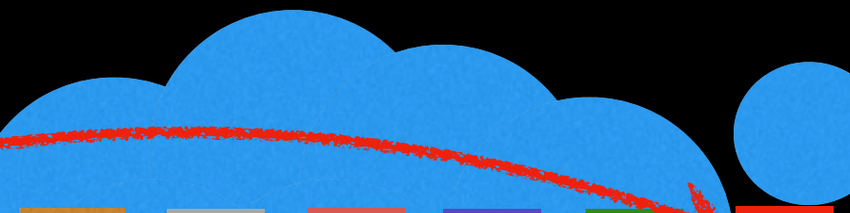
Disagreement ⇒ DAG



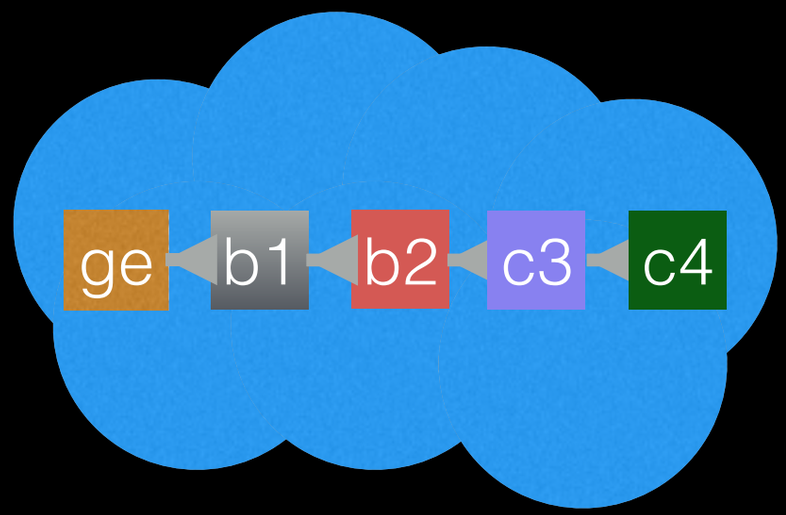
P3

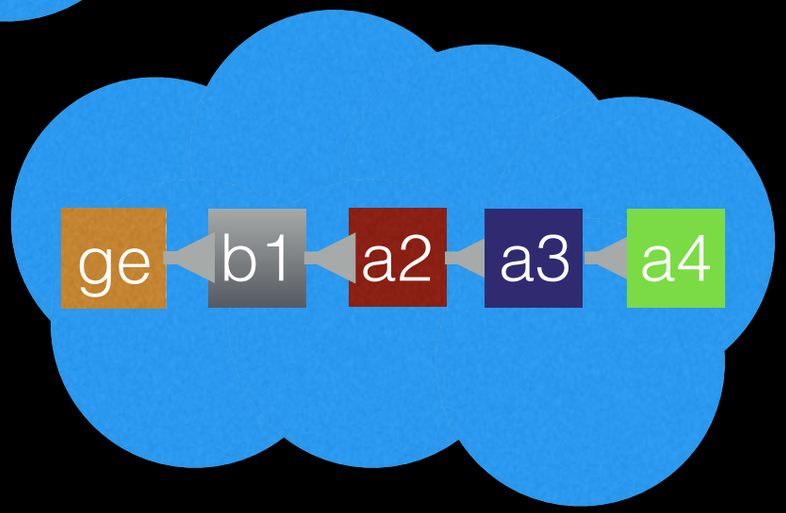
P2

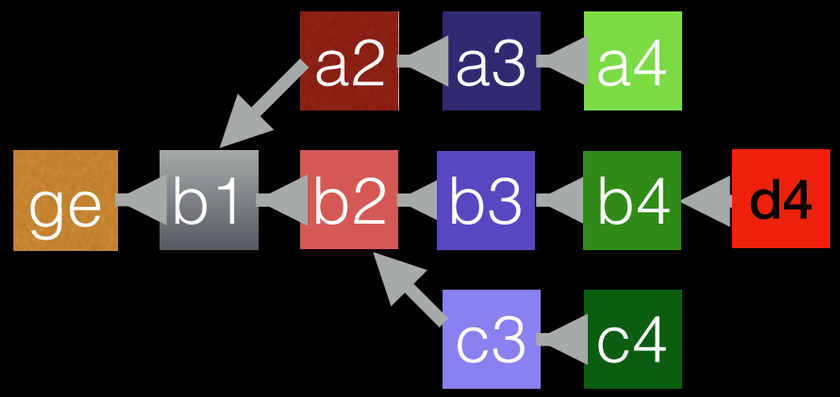
2. Balance Attack [DSN'17]



P1





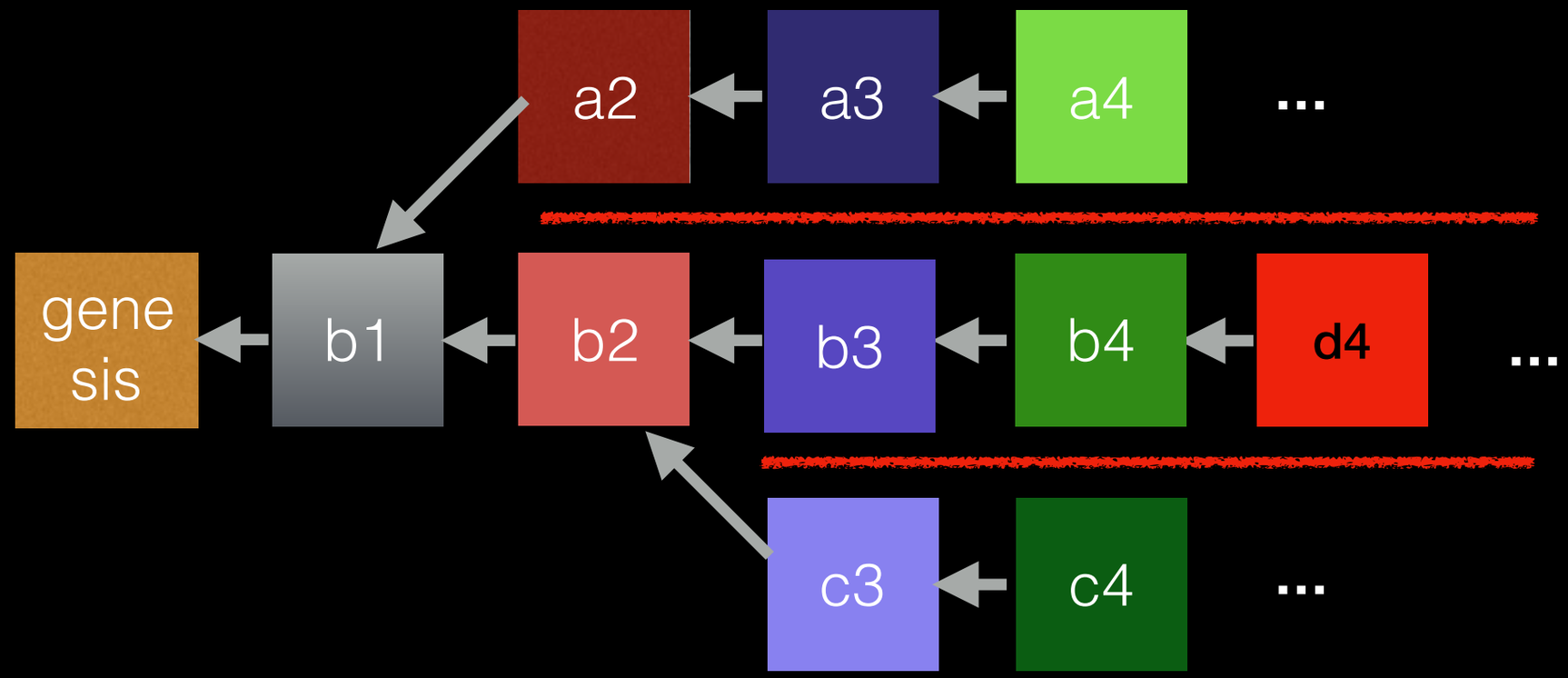


P3

P2

3. Multiple Spending

~~$\sigma_A(t_A)$~~

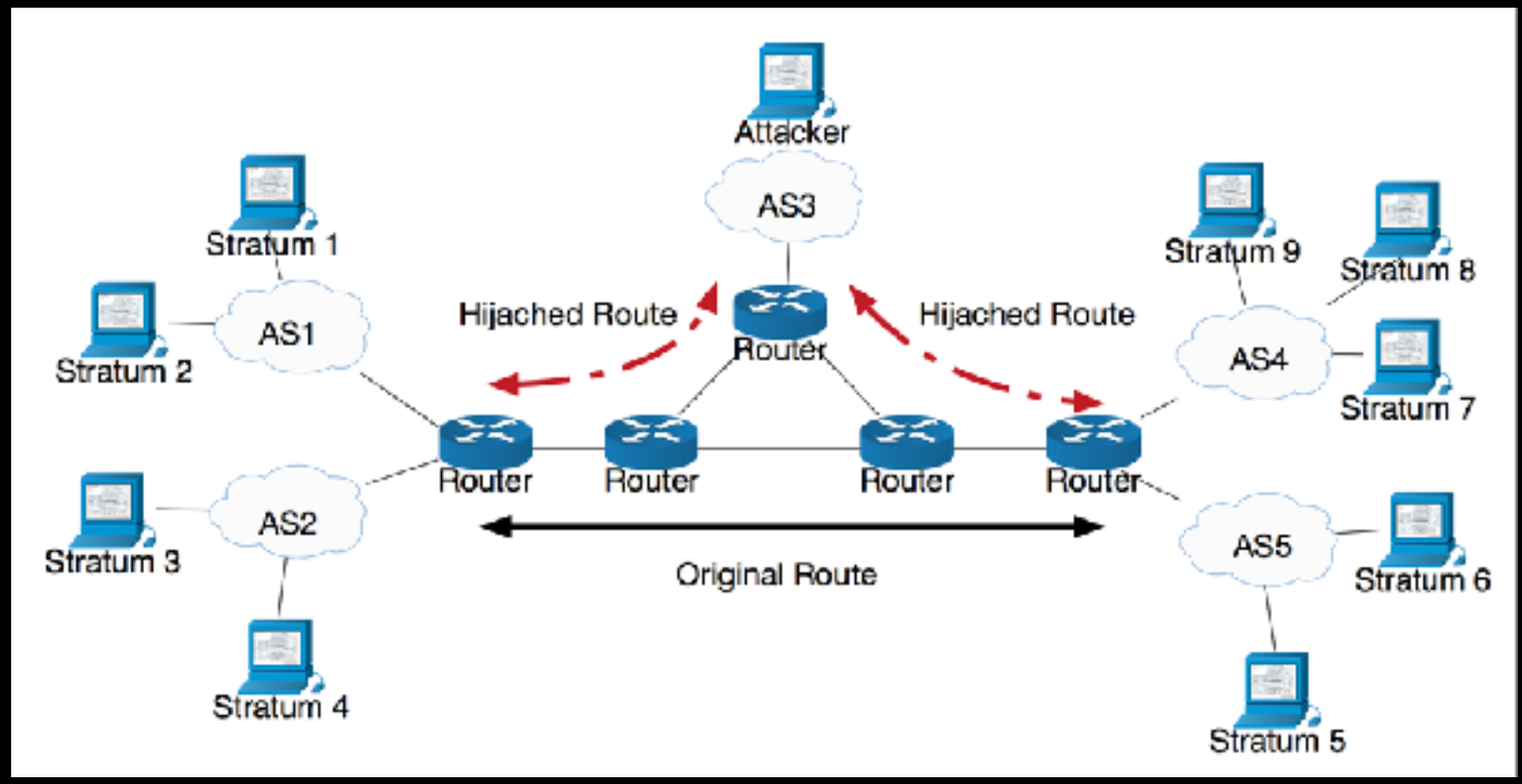


~~$\sigma_A(t_{A'})$~~

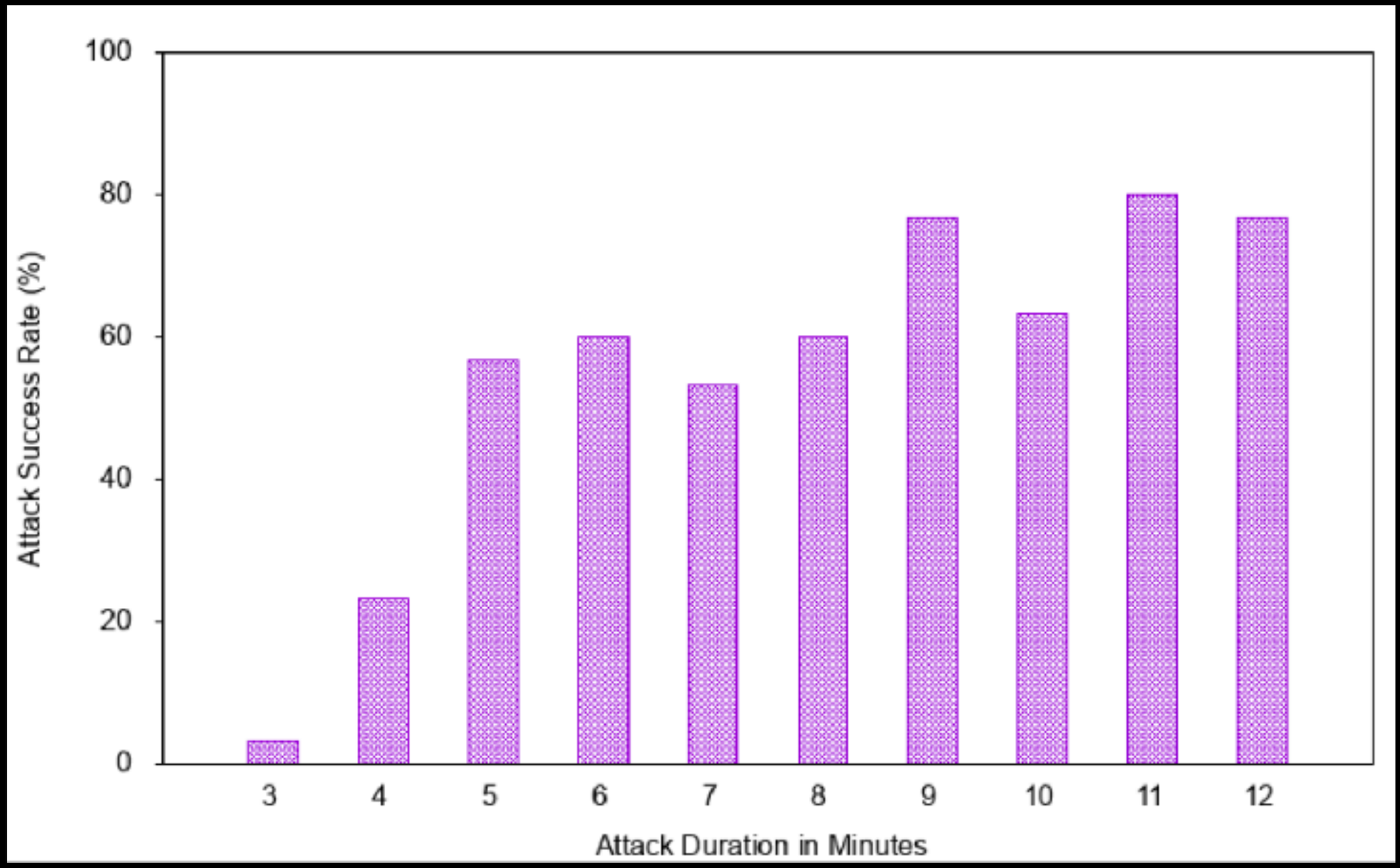
Alice

Attacking Ethereum (con't)

- Ethereum v1.5 [Woo'15] (k=11 blocks for commit)
- 10 largest mining pools
- Set CPU power using cgroup to adjust quantum
- BGP-hijacking attack of various durations (VMs with OpenStack)

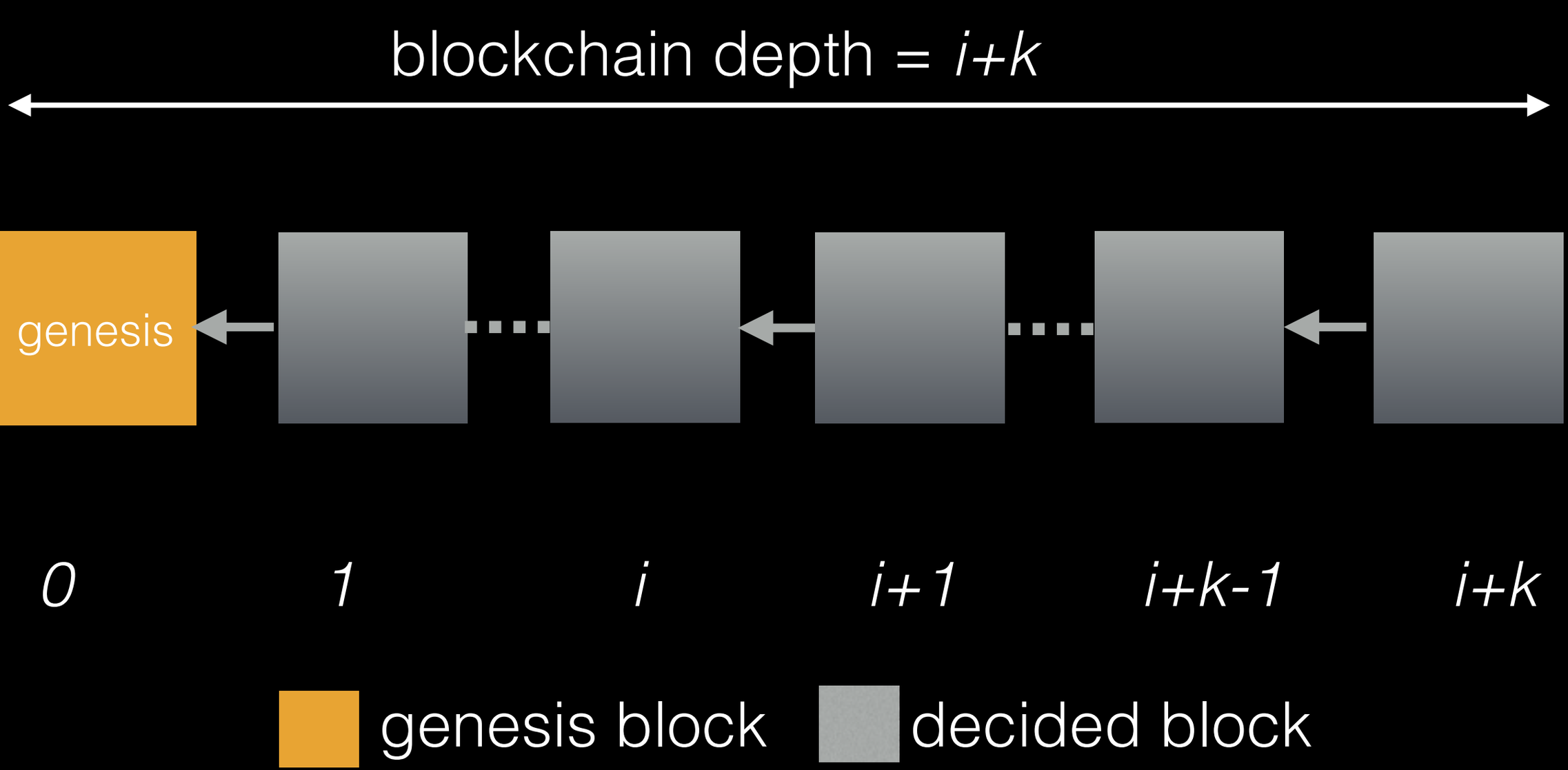


Attacking Ethereum (con't)



Blockchain Consensus

Unforkable blockchain



We say that a transaction commits when it is in a decided block [NCA'16]

Model

- Distributed system: n processes ⇐ but additional processes can issue transactions and read the blockchain
- Partially synchronous: the upper-bounds on the delay of messages and computation is not known ⇐ Internet can be congested
- Byzantine failures: there can only be $t < n/3$ arbitrary failures, all other processes are correct ⇐ Attackers have incentives to try stealing

Byzantine Consensus?

Each correct process invokes propose(v) with its value v and decides the returned value such that:

1. Agreement: no two correct processes decide differently
2. Termination: every correct process decides
3. Validity: the decided value is proposed by a correct process

Byzantine Consensus?

Each correct process invokes propose(v) with its value v and decides the returned value such that:

1. Agreement: no two correct processes decide differently
2. Termination: every correct process decides
3. Validity: the decided value is proposed by a correct process

Blockchain Consensus [AlgoTel'17]

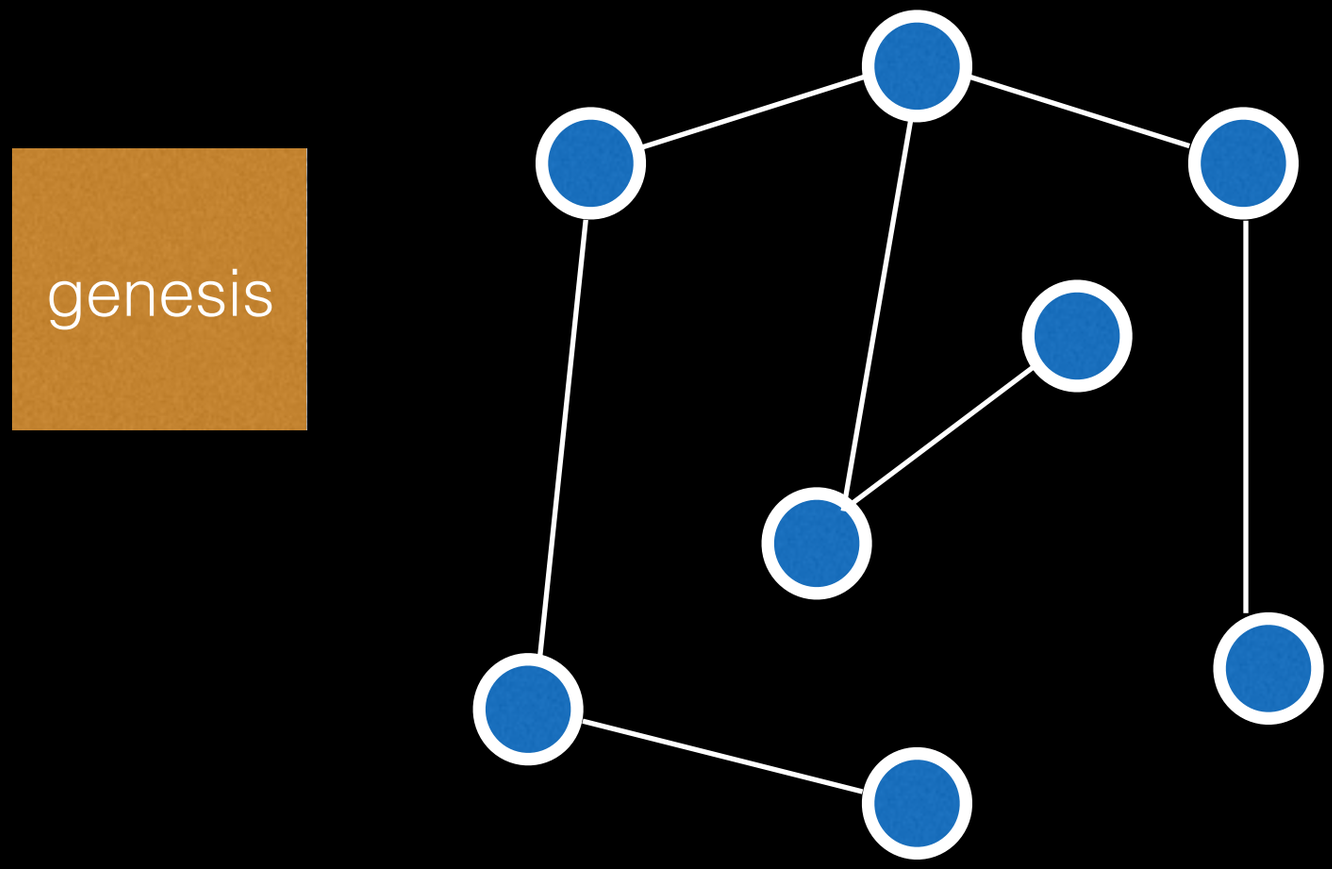
Provided an application-specific valid() predicate, each correct process invokes propose(v) and decides the returned value such that:

1. Agreement: no two correct processes decide differently
2. Termination: every correct process decides
3. Validity: the decided value satisfies the predicate valid() and if all correct propose a valid v, they decide v.

Red Belly Blockchain

The Red Belly Blockchain

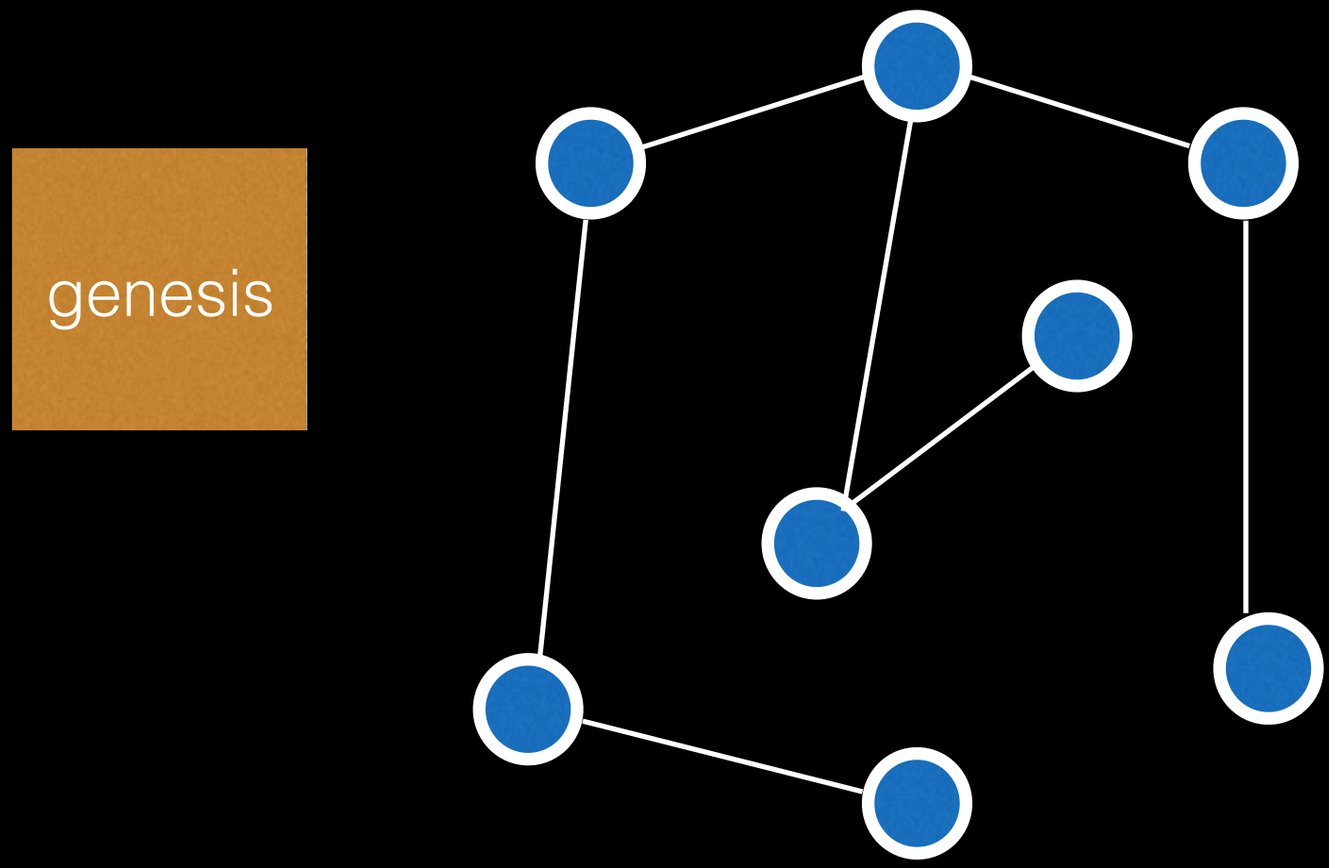
All nodes communicate through TCP + SSL



Certificates are given in blocks

The Red Belly Blockchain

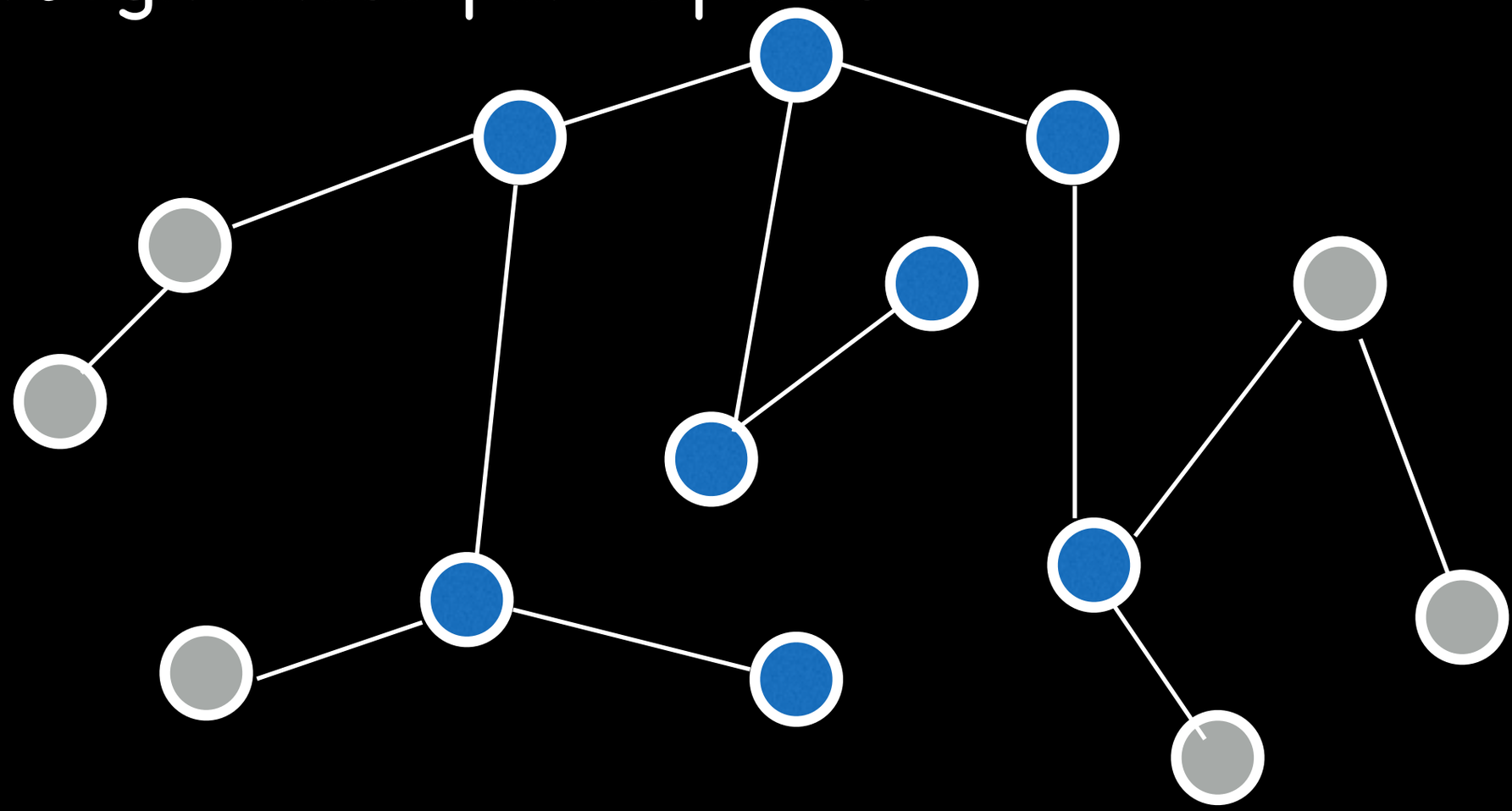
The genesis block also contains a list of n participants



…that run a leaderless blockchain consensus alg.[CGLR'17]

The Red Belly Blockchain

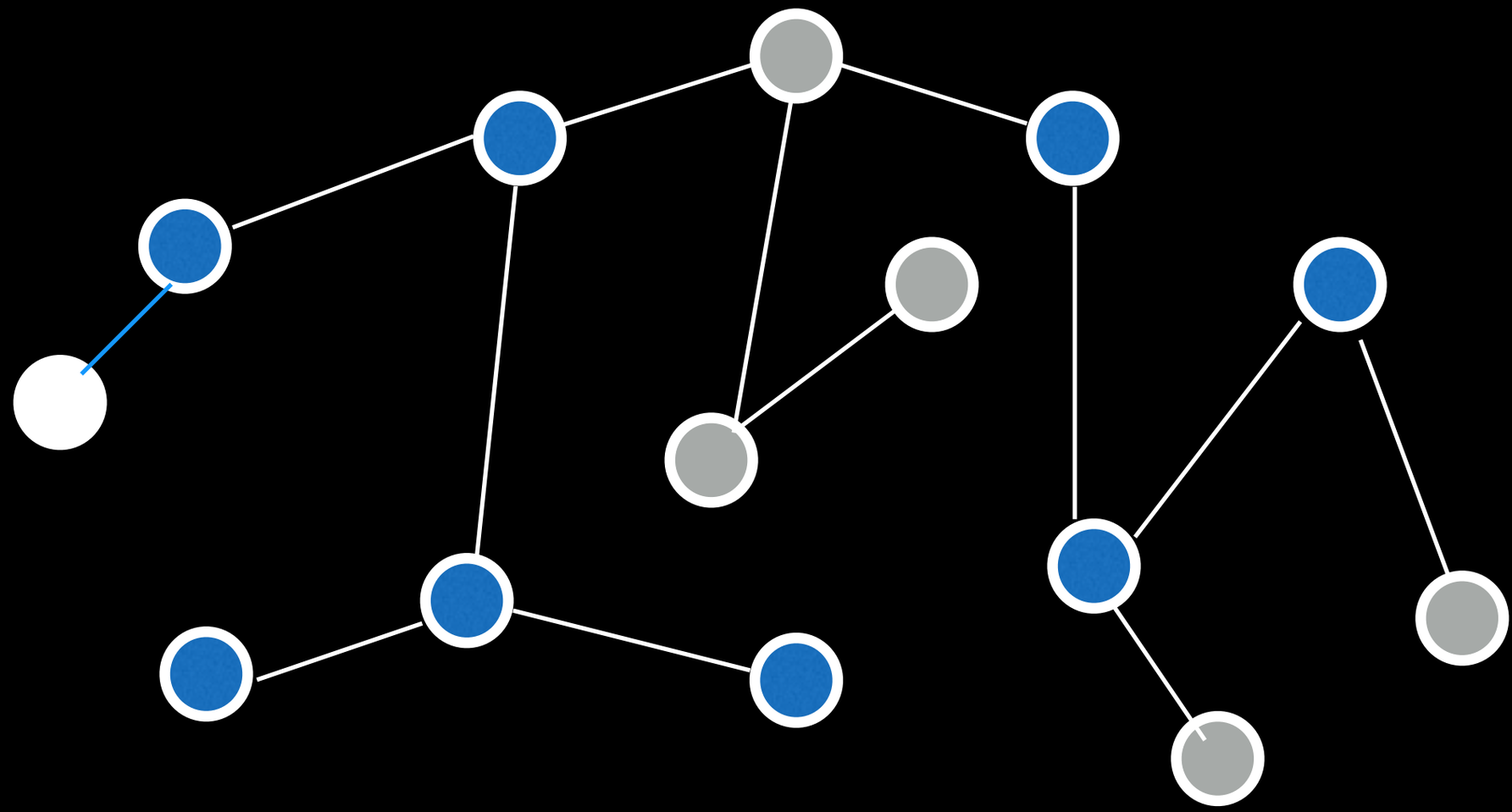
External nodes (clients) access the blockchain through these participants



A tx is committed if t+1 participants say so.

The Red Belly Blockchain

This is a community blockchain [Blockchain'18]

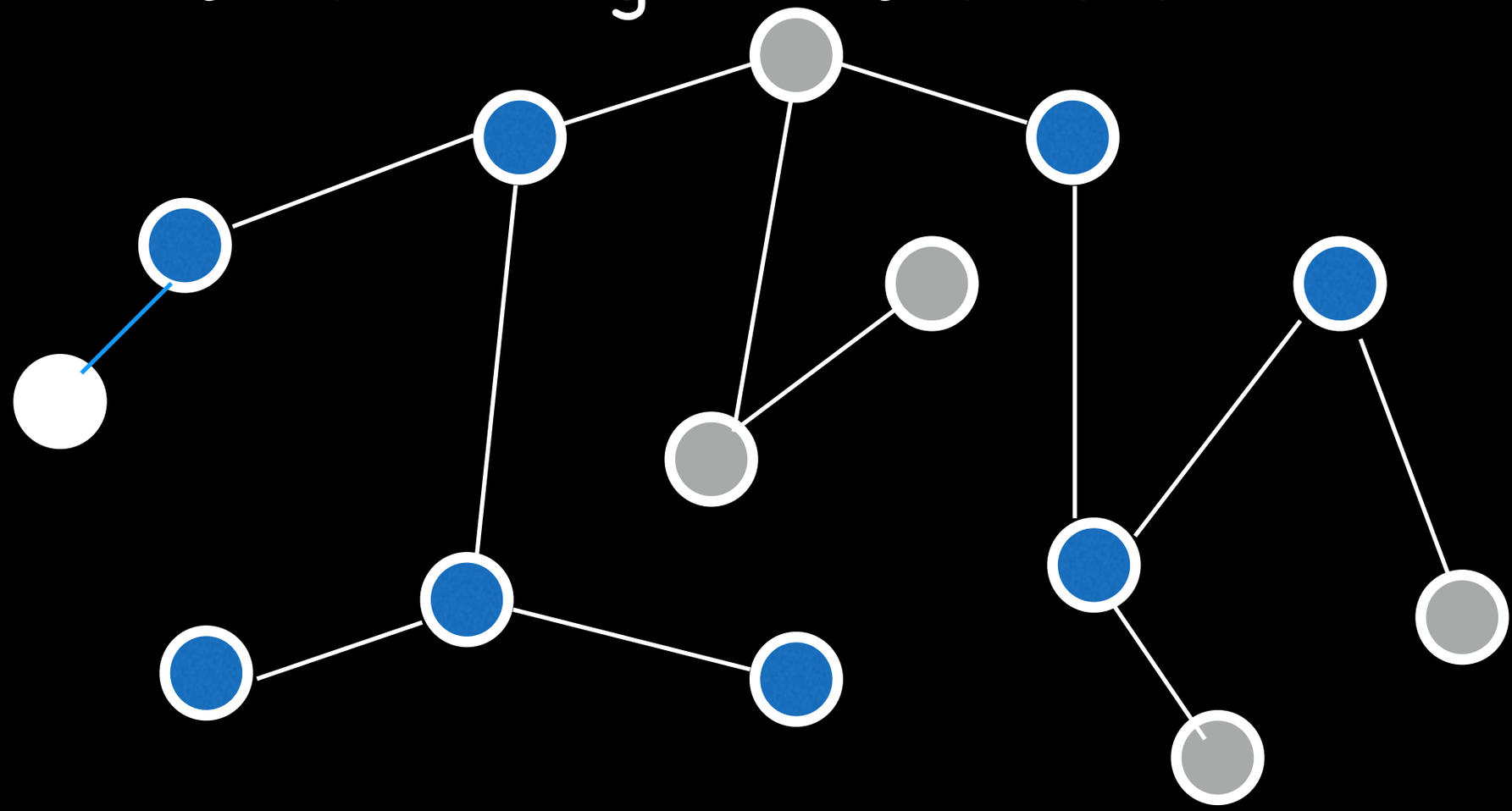


Not every node decides the block at **every** index, but every node decides upon a block at **some** index

The Red Belly Blockchain

This is a community blockchain [Blockchain'18]

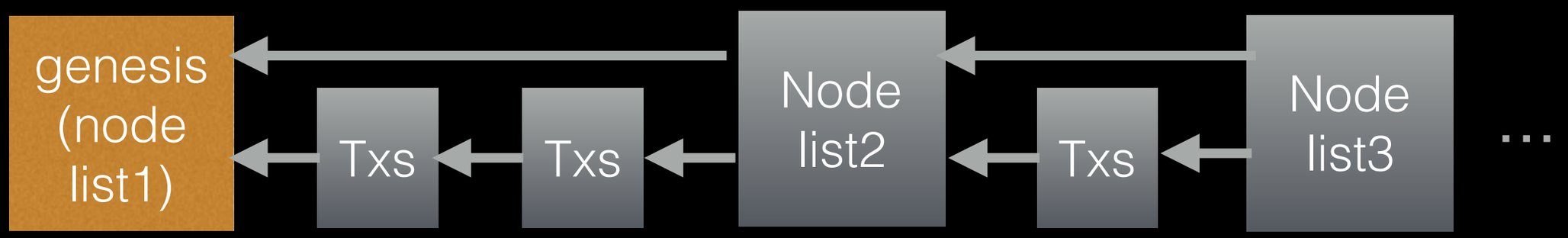
The n nodes running the consensus...



are regularly changed: n, n', n"... but t'<n'/3, t"<n"/3...

The Red Belly Blockchain

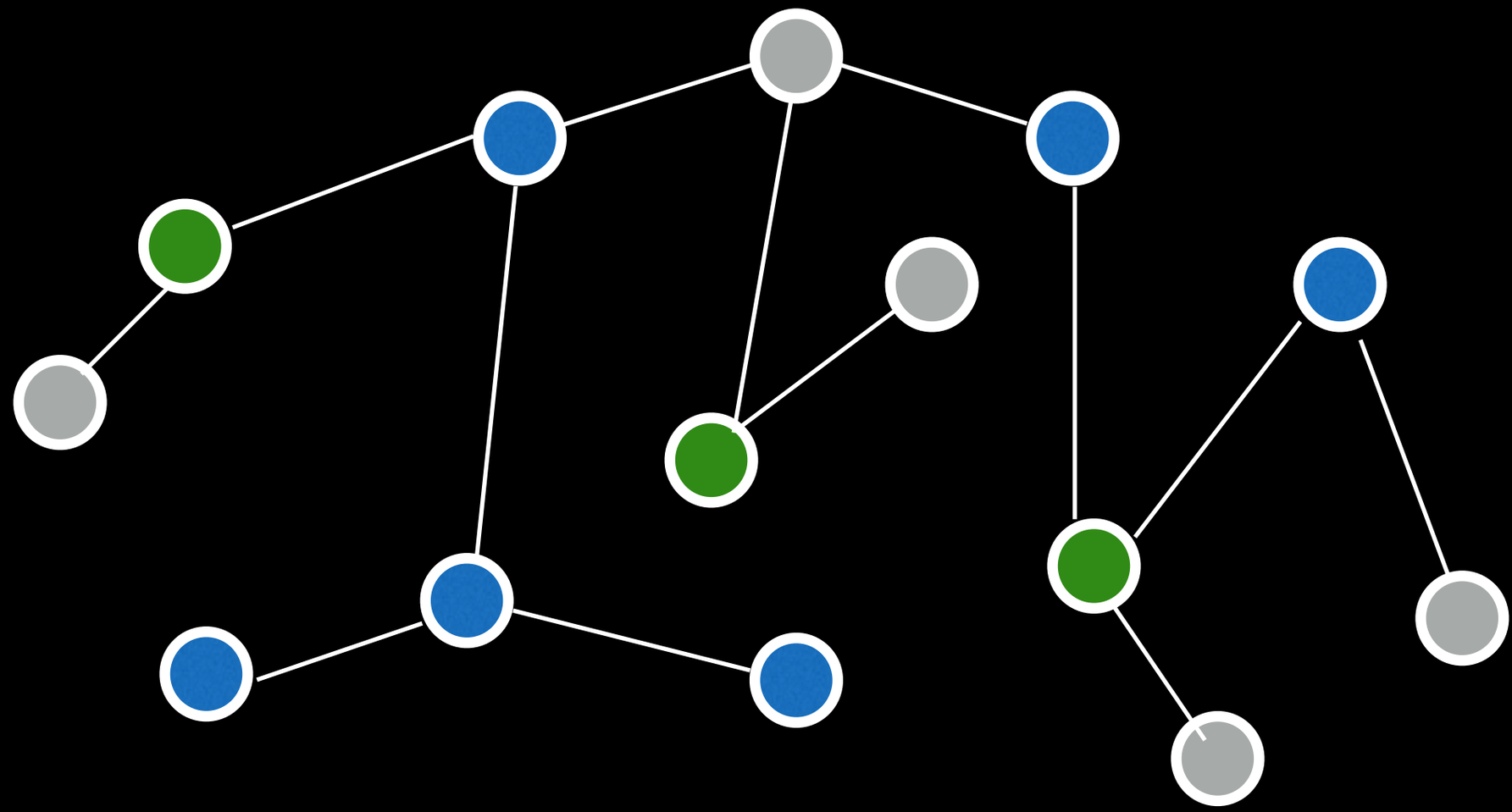
This is a community blockchain [Blockchain'18]
These nodes form a community...



...listed in special blocks, and deciding upon next transaction blocks

The Red Belly Blockchain

Signature verification uses ECDSA and is sharded



...each transaction is verified by $t+1 \leq k \leq 2t+1$ nodes

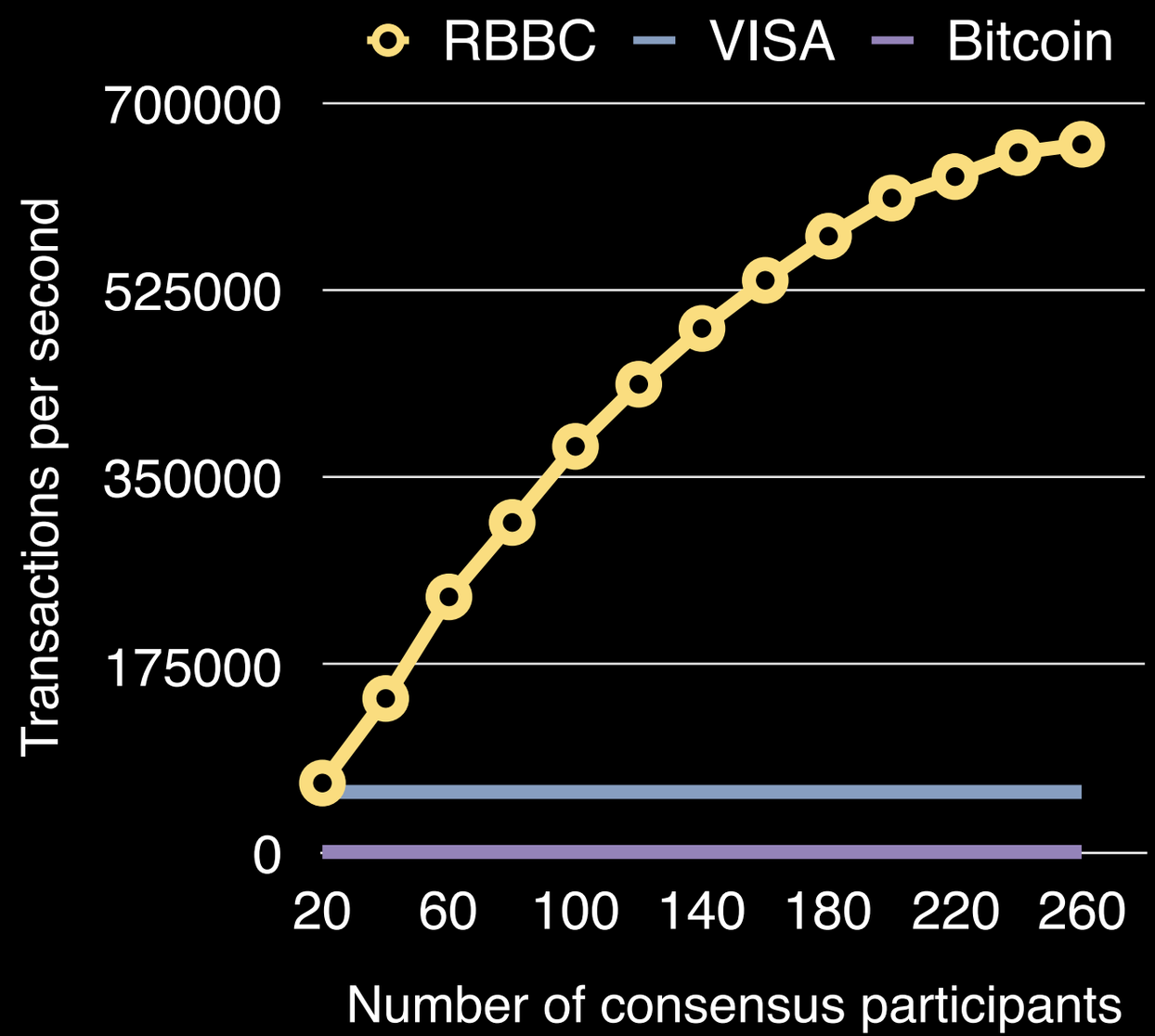
Results

Benchmark

- Initiator sends a message to n nodes to start (with same genesis block)
- Each node connects to each other through SSL/TLS
- Average over multiple instances of consensus in which:
  - Each of the n nodes proposes a block of 10K txs
  - Each node spawns n instances of RBbcast and BBC
  - Each tx is a 350-byte UTXO transaction
  - Each transaction gets validated by $t+1 \leq k \leq 2t+1$ nodes
  - Each node stores the blockchain locally

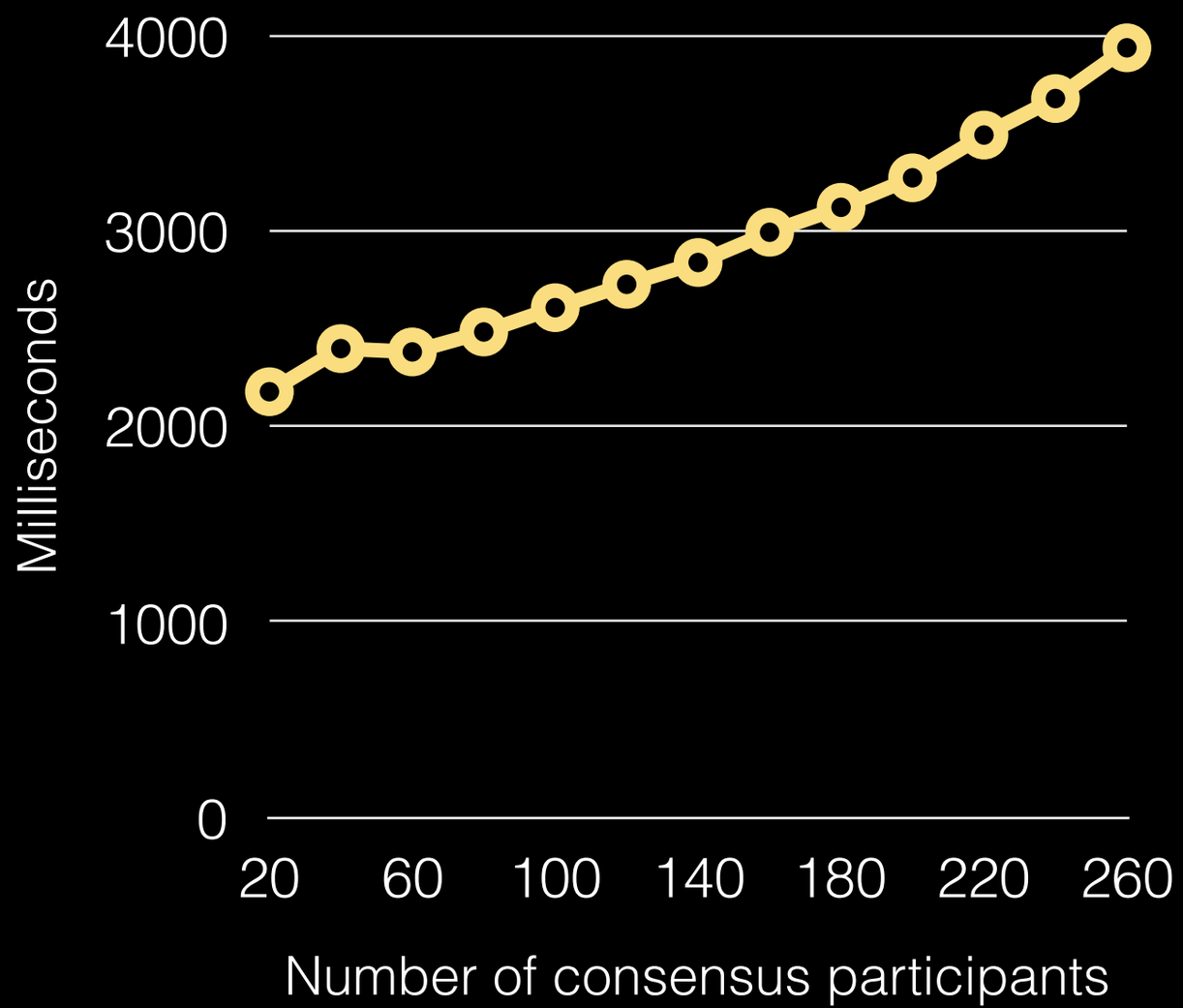
Scalability

Amazon EC2
c4 instances,
18 HT cores,
60 GiB mem,
2 Gbps,
t=6

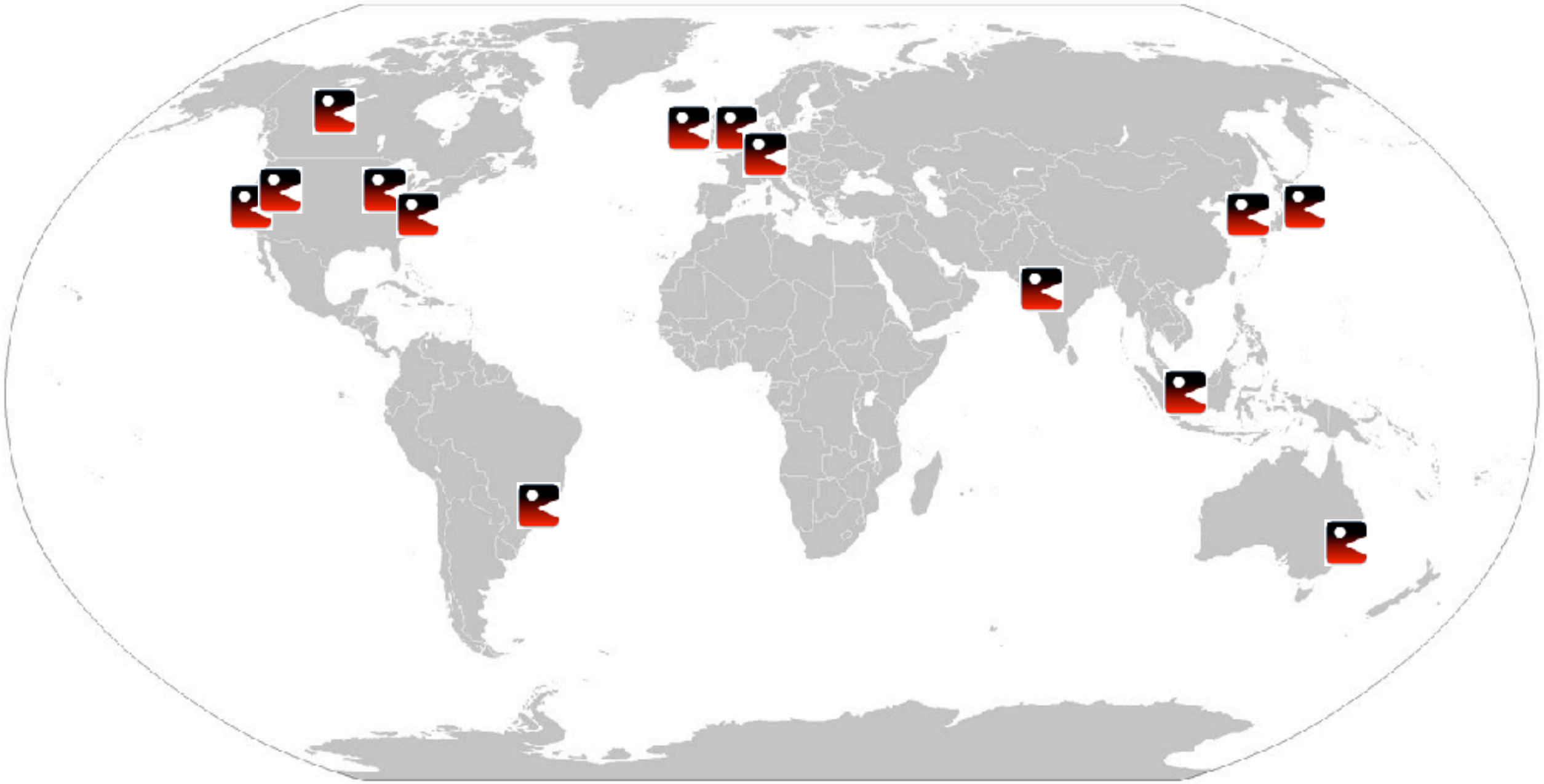


Latency

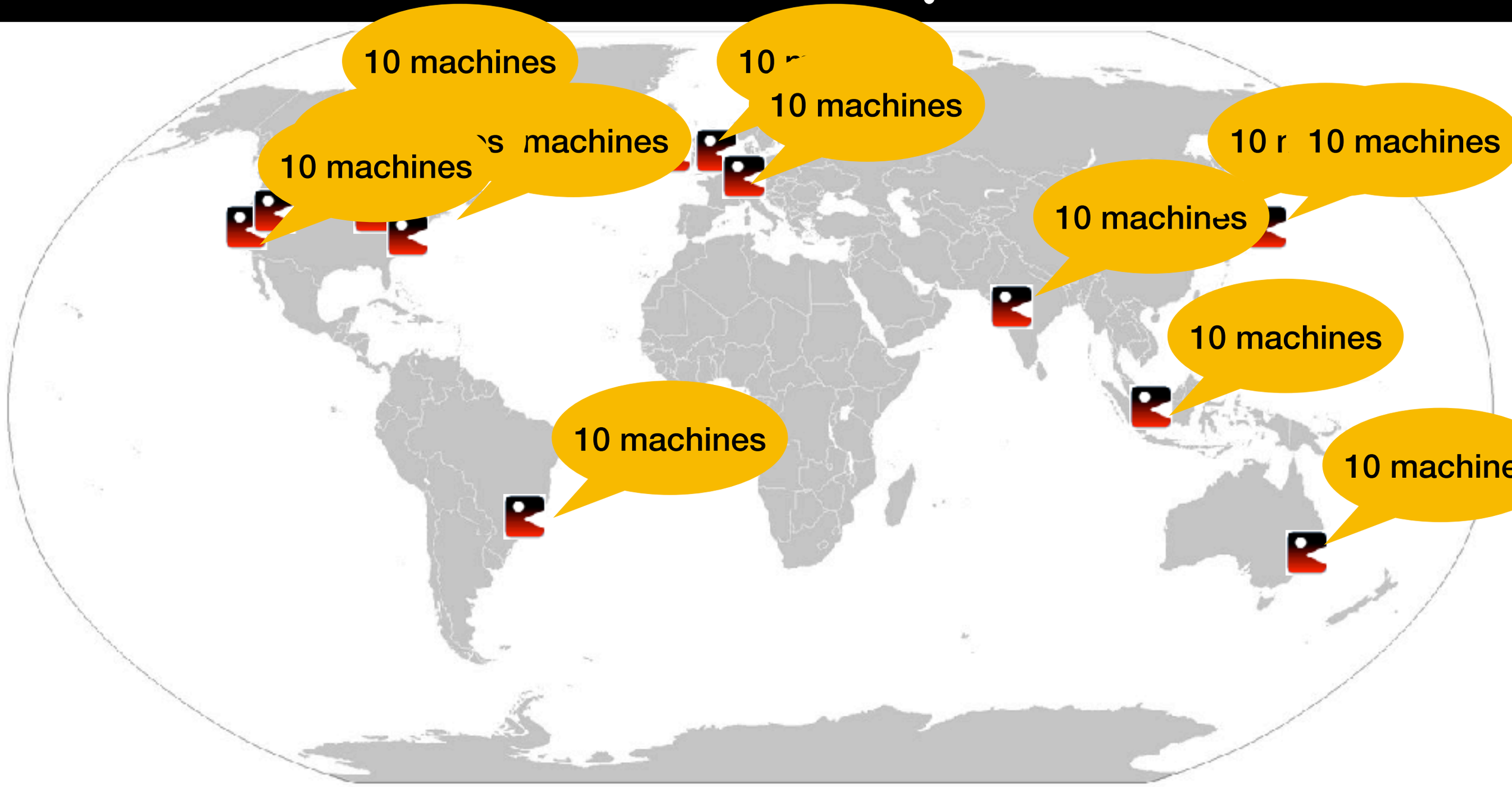
Amazon EC2
c4 instances,
18 HT cores,
60 GiB mem,
2 Gbps,
t=6



World-Wide Deployment



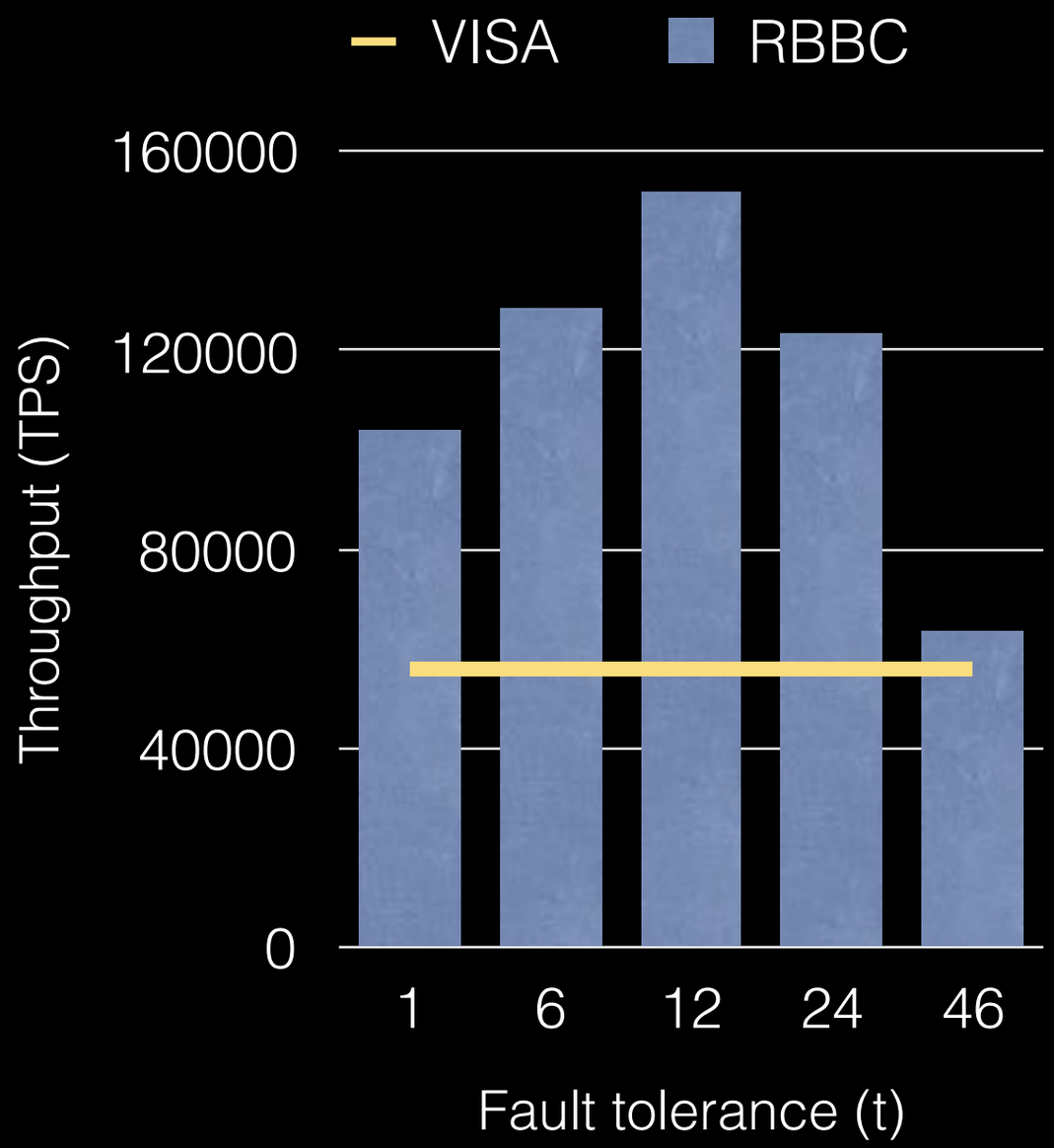
World-Wide Deployment



World-Wide Deployment

World-Wide Deployment

c4 instances, 4 vCPU, 7.5 GiB, 750 Mbps, n=140



Consensus Comparison

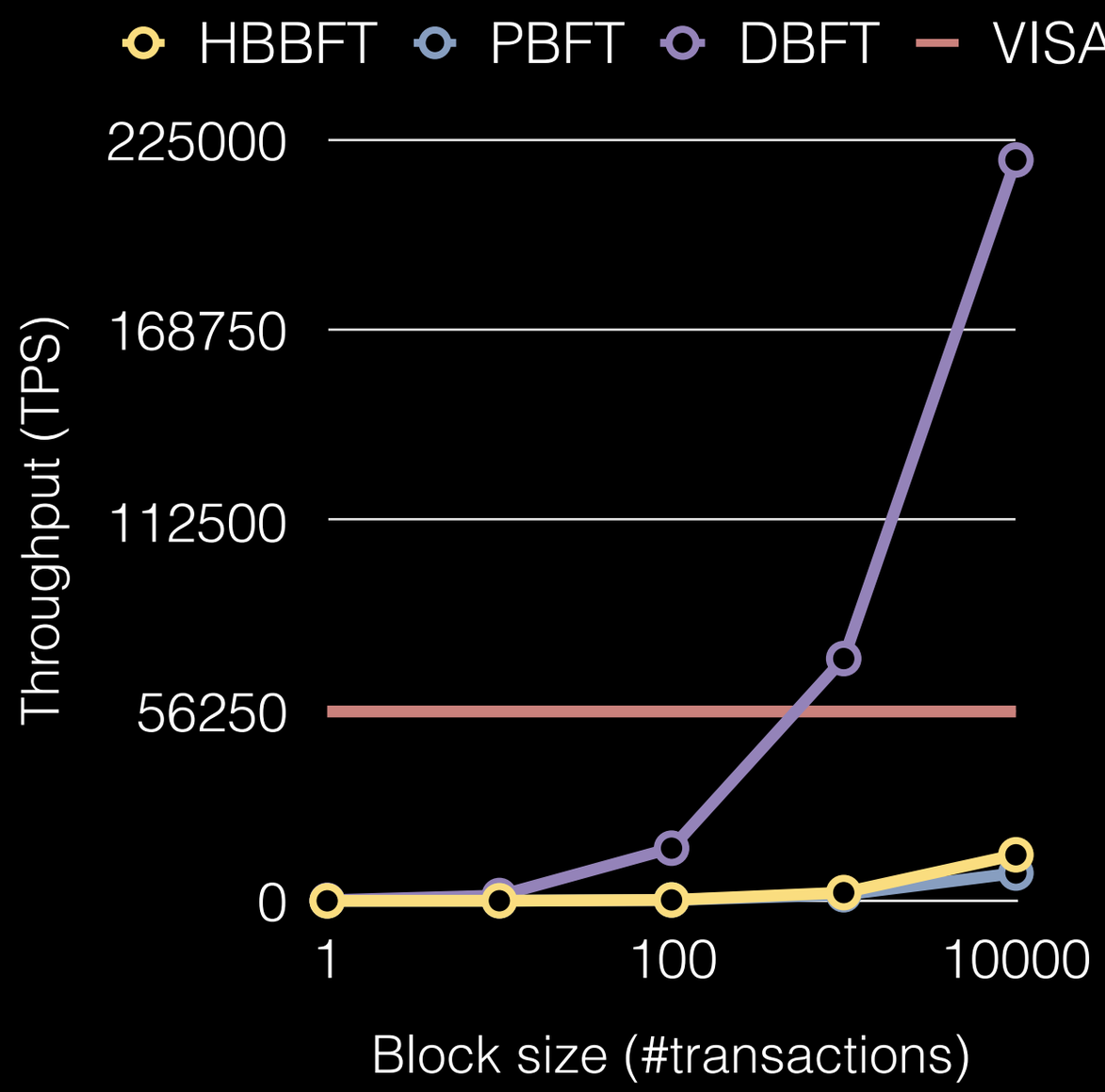
- PBFT: State-of-the-art Byzantine consensus implementation [OSDI'02]. It relies on a leader and decides on one of the proposed value.

- HBBFT: The Honey Badger BFT [CCS'16] is based on the binary randomized consensus algorithm [PODC'14], a consensus reduction [PODC'94] and uses erasure codes.

- DBFT: The Democratic BFT [CGLR17] we introduced for RBBC. It is leader-less, does not exchange erasure codes but block hashes.

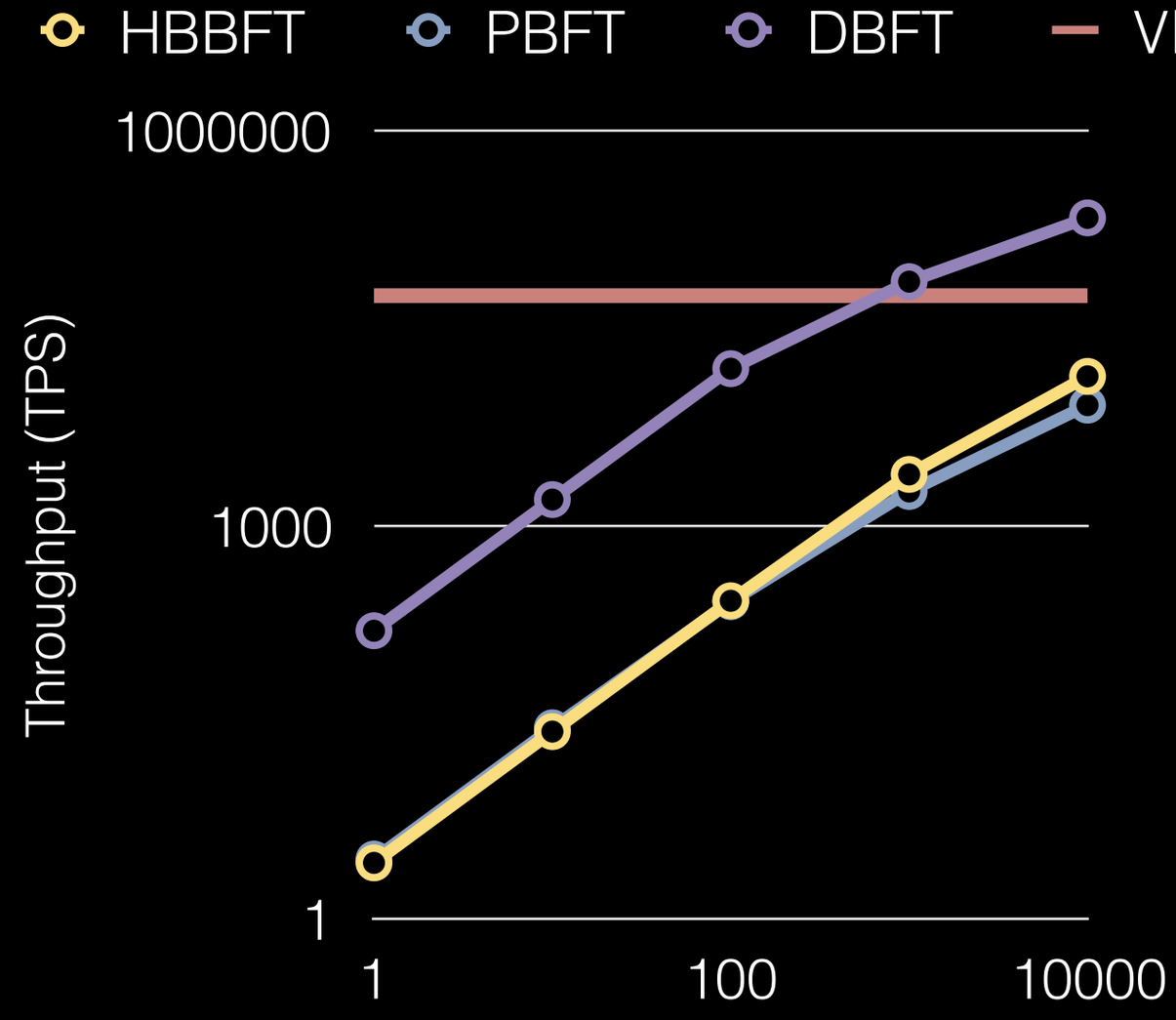
Consensus Comparison

c4 instances, 4 vCPU, 7.5 GiB, 750 Mbps, n=140, t=46



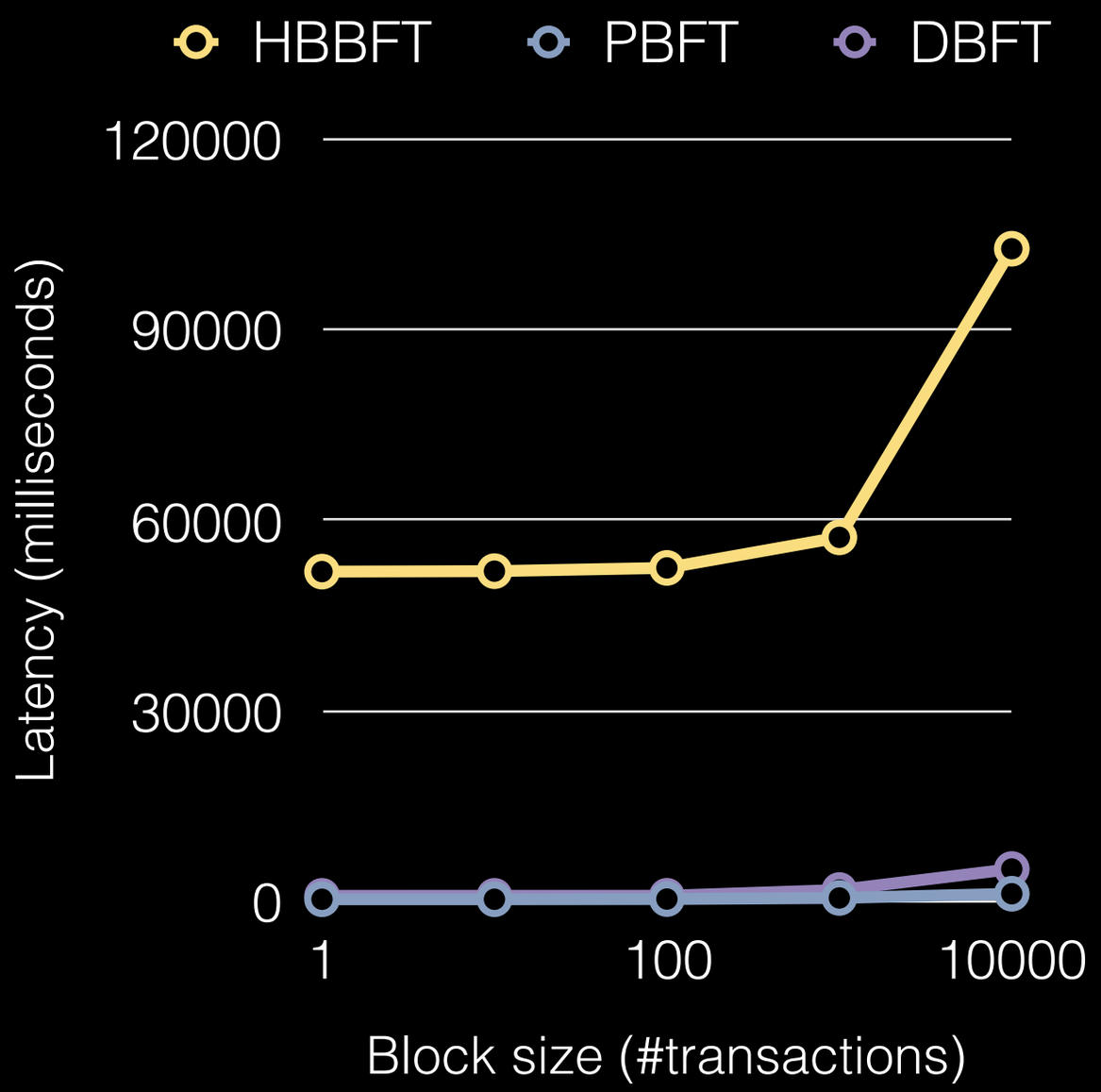
Consensus Comparison

c4 instances, 4 vCPU, 7.5 GiB, 750 Mbps, n=140, t=46

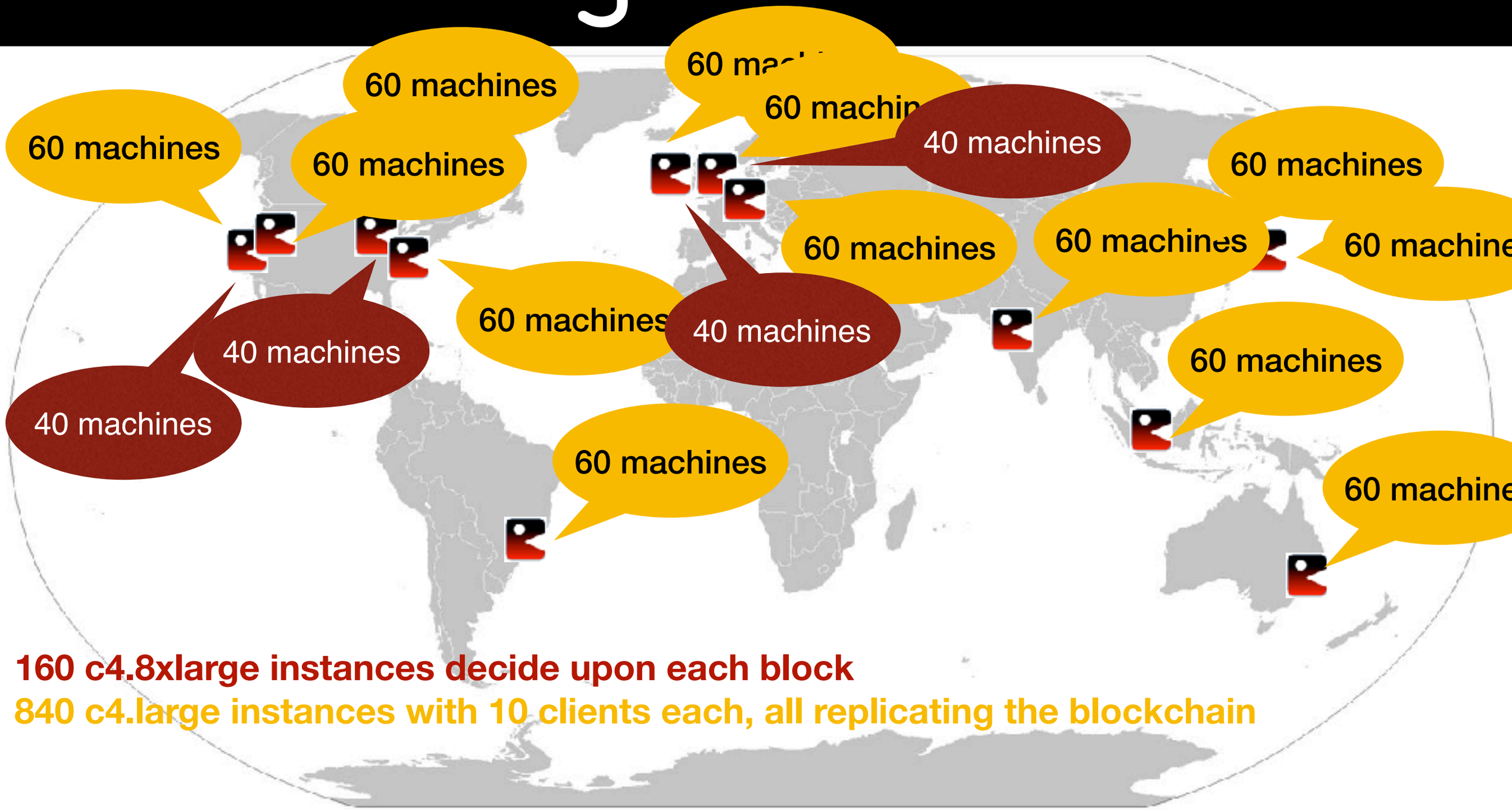


Consensus Comparison

c4 instances, 4 vCPU, 7.5 GiB, 750 Mbps, n=140, t=46



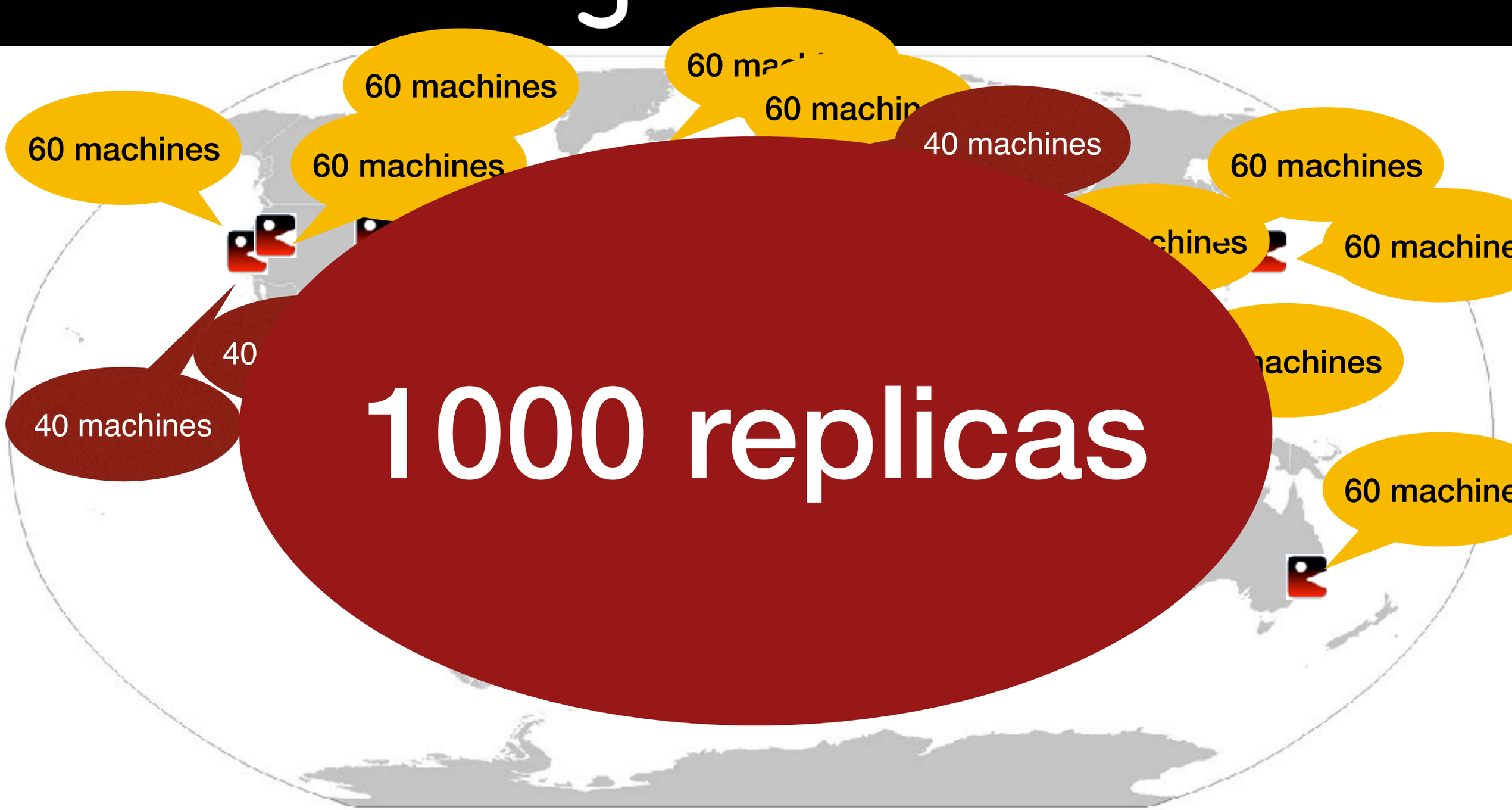
Larger Scale



160 c4.8xlarge instances decide upon each block

840 c4.large instances with 10 clients each, all replicating the blockchain

Larger Scale



Larger Scale

#replicas	#requesters	Valid tx/sec	Async write Latency	Latency	Valid tx/ block	Invalid tx/block
1000	8400	30684	238ms	3103ms	95407	378

Conclusion

- We propose the Red Belly Blockchain
  - Secure: does not fork
  - Efficient: commits up to 660,000 TPS
  - Scale to 1000 geodistributed replicas with a 3 second latency
  - Dynamic: A community blockchain that avoids wastes

Future Work

1. Deploy community nodes under the control of distinct jurisdictions and representative of different parts of the population

2. We are implementing incentives (identify and punish misbehaviors) for a more realistic model (rational instead of correct/ Byzantine)

References

- [OSDI'02] M. Castro, B. Liskov Practical Byzantine fault tolerance. Proc. of the 3rd Symposium on OS Design, OSDI, 2002.
- [Nak'08] S. Nakamoto, "Bitcoin: a peer-to-peer electronic cash system," 2008. http://www.bitcoin.org
- [Woo'15] G. Wood. "Ethereum: A secure decentralised generalised transaction". Yellow paper. 2015Process. 2014. https://github.com/ethereum/wiki/wiki/White-Paper
- [NCA'16] C. Natoli, V. Gramoli. The Blockchain Anomaly. Proc. of the 15th IEEE International Symposium on Network Computing and Applications 2016.
- [CCS'16] A. Miller, Y. Xa, K. Croman, E. Shi, D. Song. The Honey Badger of BFT Protocols. Proc. of the ACM SIGSAC Conference on Computer and Communications Security, 2016.
- [AlgoTel'17] T. Crain, V. Gramoli, M. Larrea, M. Raynal. Blockchain Consensus. May 2017.
- [CGLR17] T. Crain, V. Gramoli, M. Larrea, M. Raynal (Leader/Randomization/Signature)-free Byzantine Consensus for Consortium Blockchains. arXiv:1702.03068
- [DSN'17] C. Natoli, V. Gramoli. The Balance Attack or Why Forkable Blockchains Are Ill-Suited for Consortium. DSN, 2017
- [Blockchain'18] G. Vizier, V. Gramoli. ComChain: Bridging the Gap Between Public and Consortium Blockchains. IEEE Blockchain 2018.
- [SRDS'18] P. Ekparinya, V. Gramoli, G. Jourjon. Double-Spending Risk Quantification in Private, Consortium and Public Ethereum Blockchains. IEEE Blockchain 2018.

See you in Sydney

More information

https://redbellyblockchain.io