EuroSys 2025

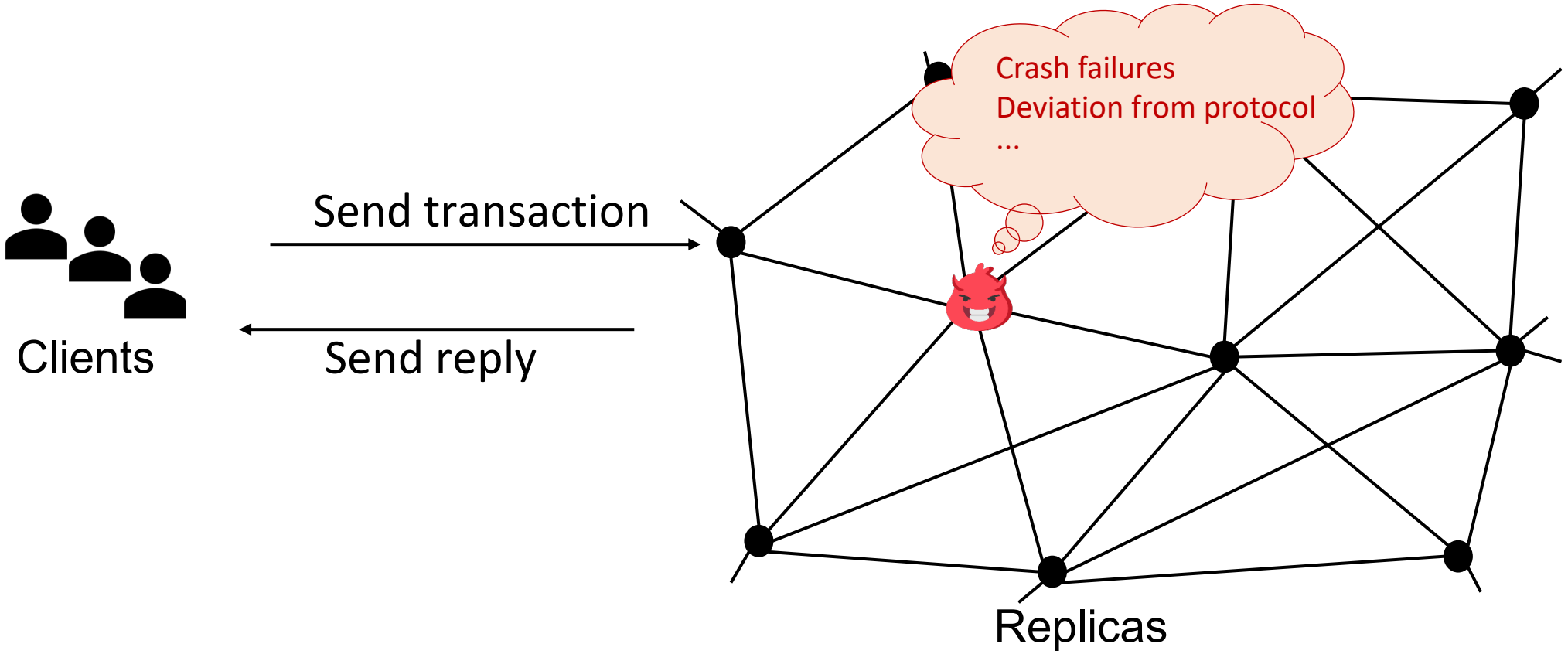
Ladon: High-Performance Multi-BFT Consensus via Dynamic Global Ordering

Hanzheng Lyu[1], Shaokang Xie[2], Jianyu Niu[2], Chen Feng[1],
Yinqian Zhang[2], and Ivan Beschastnikh[1]
[1]University of British Columbia, Canada
[2]Southern University of Science and Technology, China

BFT Consensus

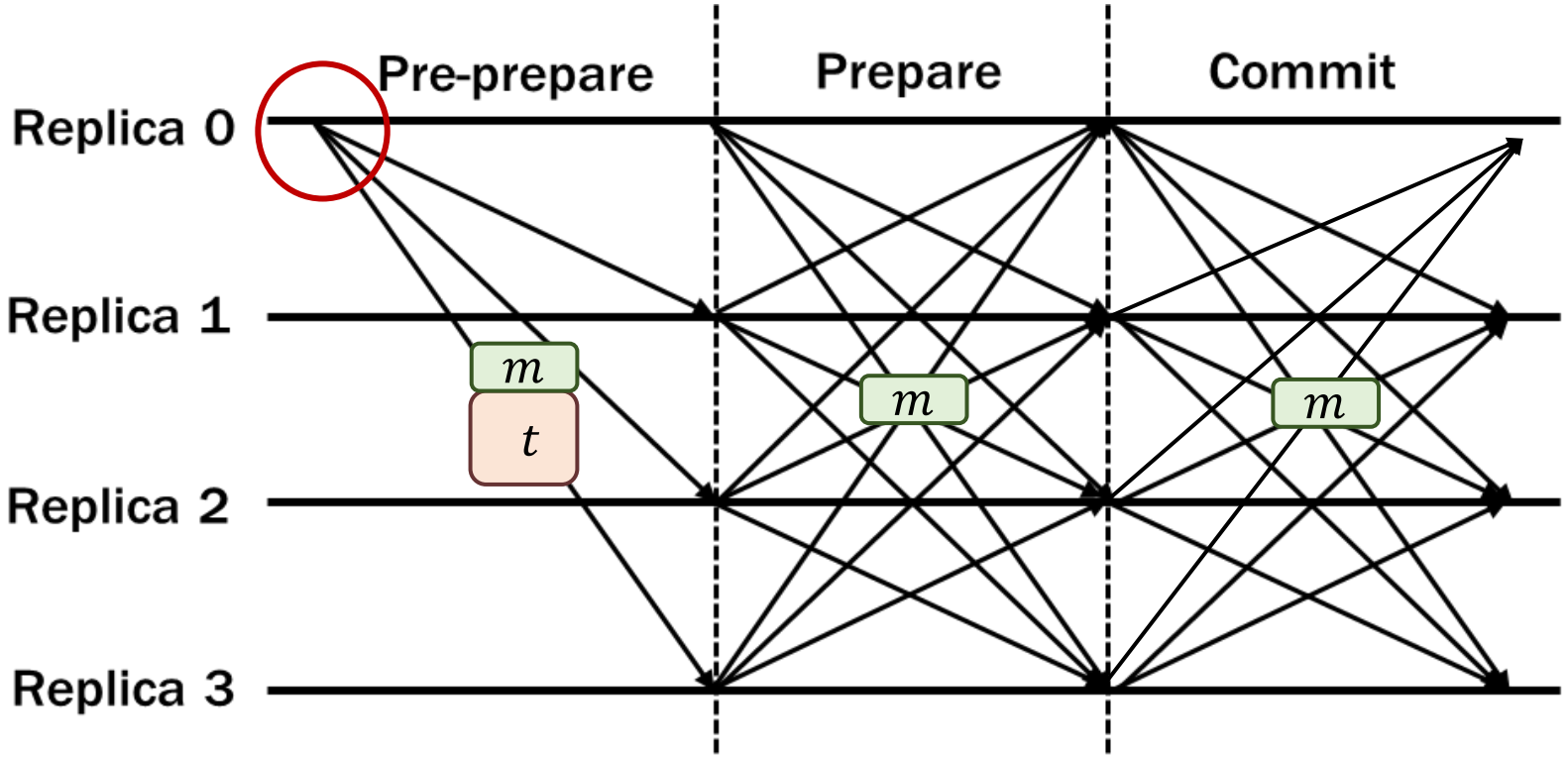
Byzantine Fault Tolerant (BFT) Consensus

- allows a decentralized system to reach agreement among replicas
- despite Byzantine failures



Leader-based BFT[1]

The leader limits the overall performance of the BFT protocol.



Practical Byzantine Fault Tolerance (PBFT) Protocol

$$size(t) >> size(m)$$

$$T_{\max} \approx \frac{B}{(n-1)size(t)}$$

Maximum throughput

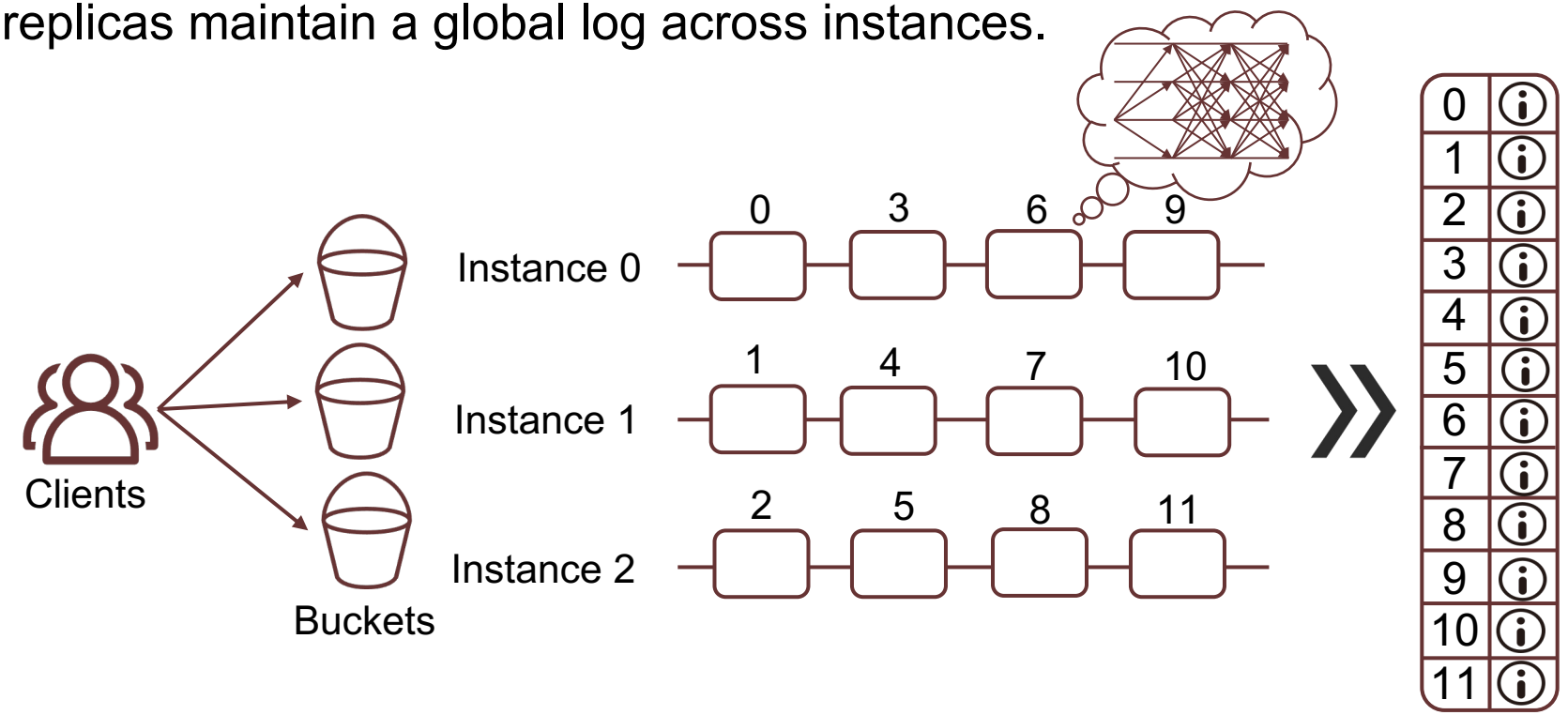
Leader's bandwidth

number of replicas

[1] Miguel Castro and Barbara Liskov. Practical Byzantine Fault Tolerance. In OSDI, 1999.

Multi-BFT Consensus[1-3]

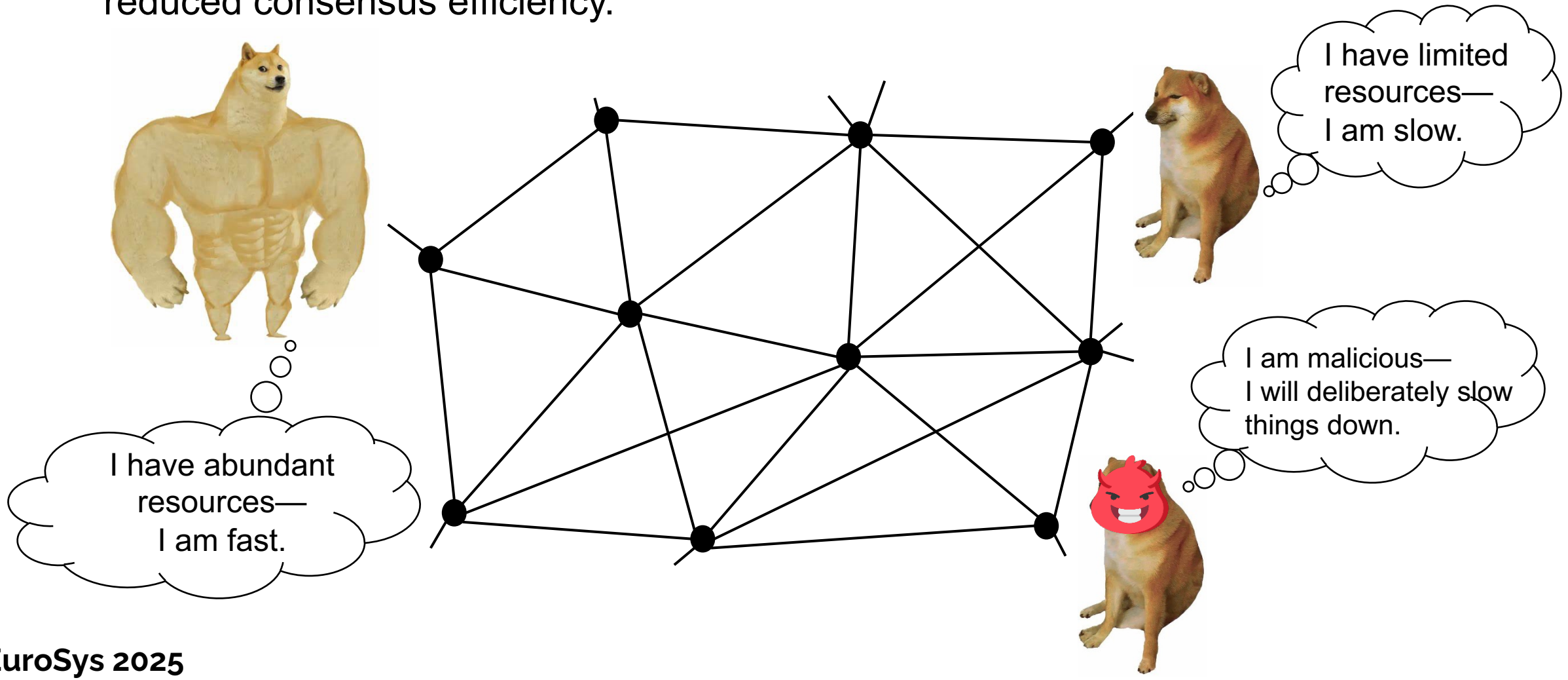
- Each replica act as the leader of one instance and backups of other instances.
- Each instance independently outputs a sequence of committed blocks.
- All replicas maintain a global log across instances.



[1] Chrysoula Stathakopoulou, Matej Pavlovic, and Marko Vukolić. State Machine Replication Scalability Made Simple, in ACM EuroSys'22.
[2] Chrysoula Stathakopoulou, Tudor David, and Marko Vukolic. Mir BFT: Scalable and Robust BFT for Decentralized Networks, in Jsys'22.
[3] Suyash Gupta, Jelle Hellings, and Mohammad Sadoghi. RCC: Resilient Concurrent Consensus for High-Throughput Secure Transaction Processing, in IEEE ICDE'21.

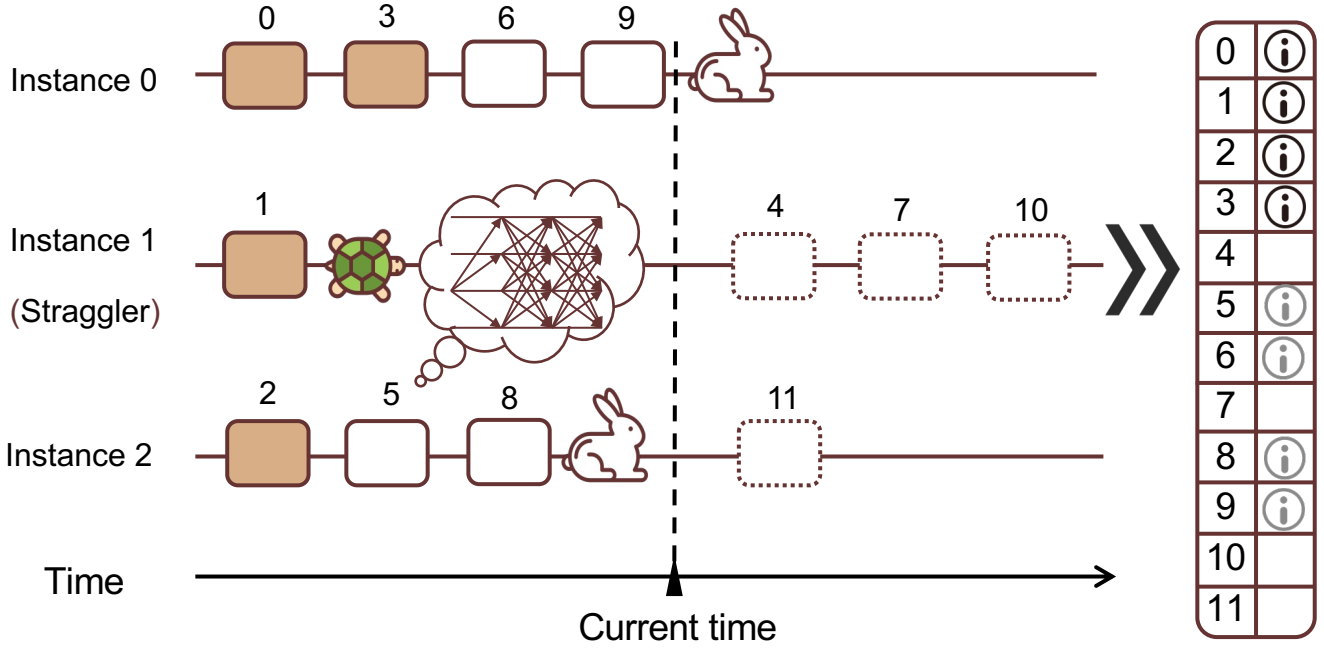
Stragglers in Network

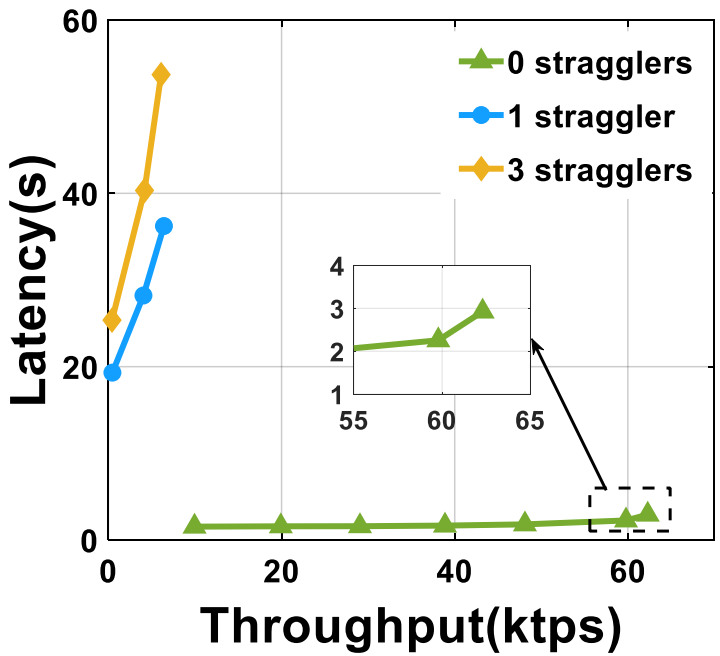
In distributed systems, disparities in CPU, bandwidth, and other resources, along with potential malicious behavior, can lead to some replicas being slower, causing reduced consensus efficiency.



The Impact of Stragglers

Performance degradation of pre-determined global ordering with stragglers

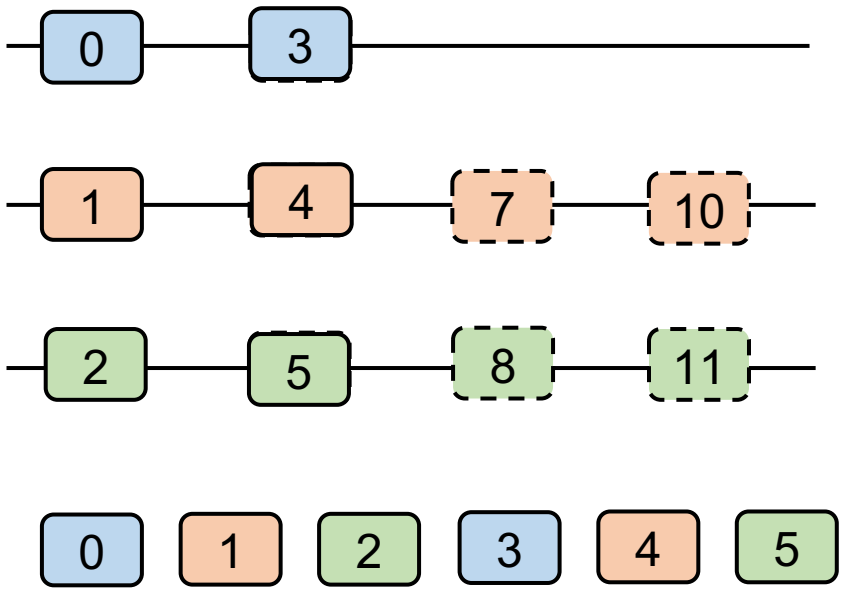




Key Observation

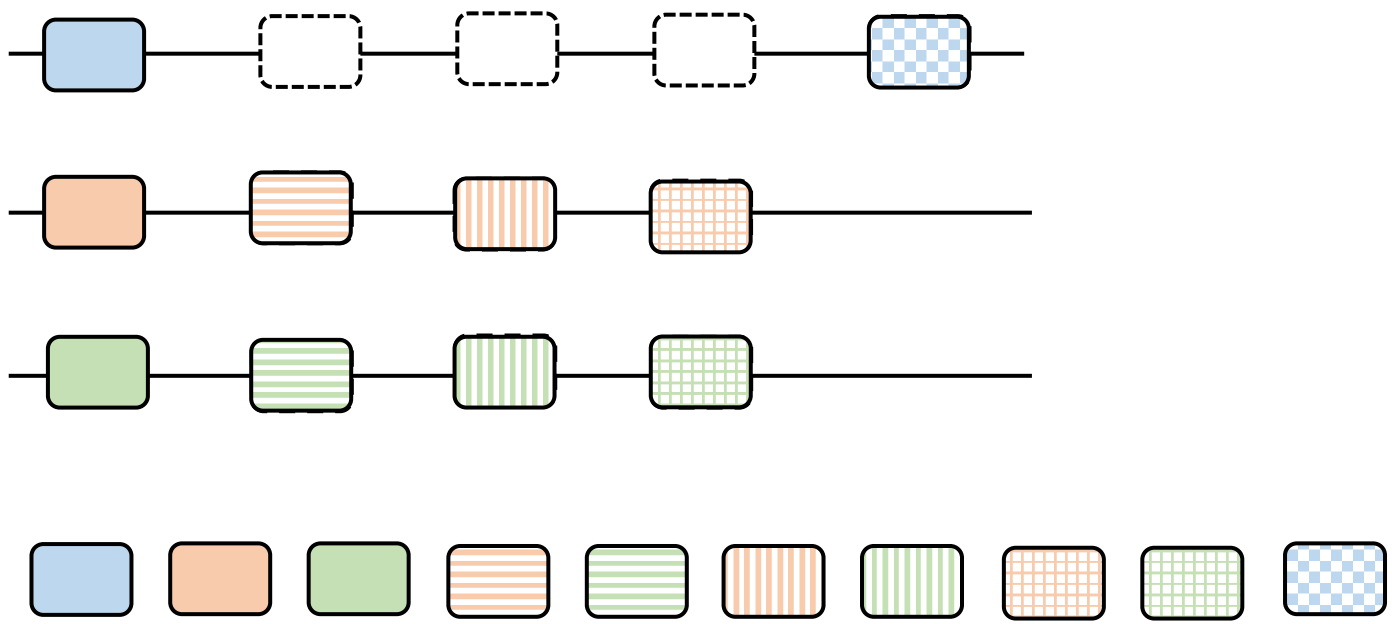
Limitations of Pre-Determined Ordering

- A slow instance prevents other instances from confirming their blocks.



Key Observation

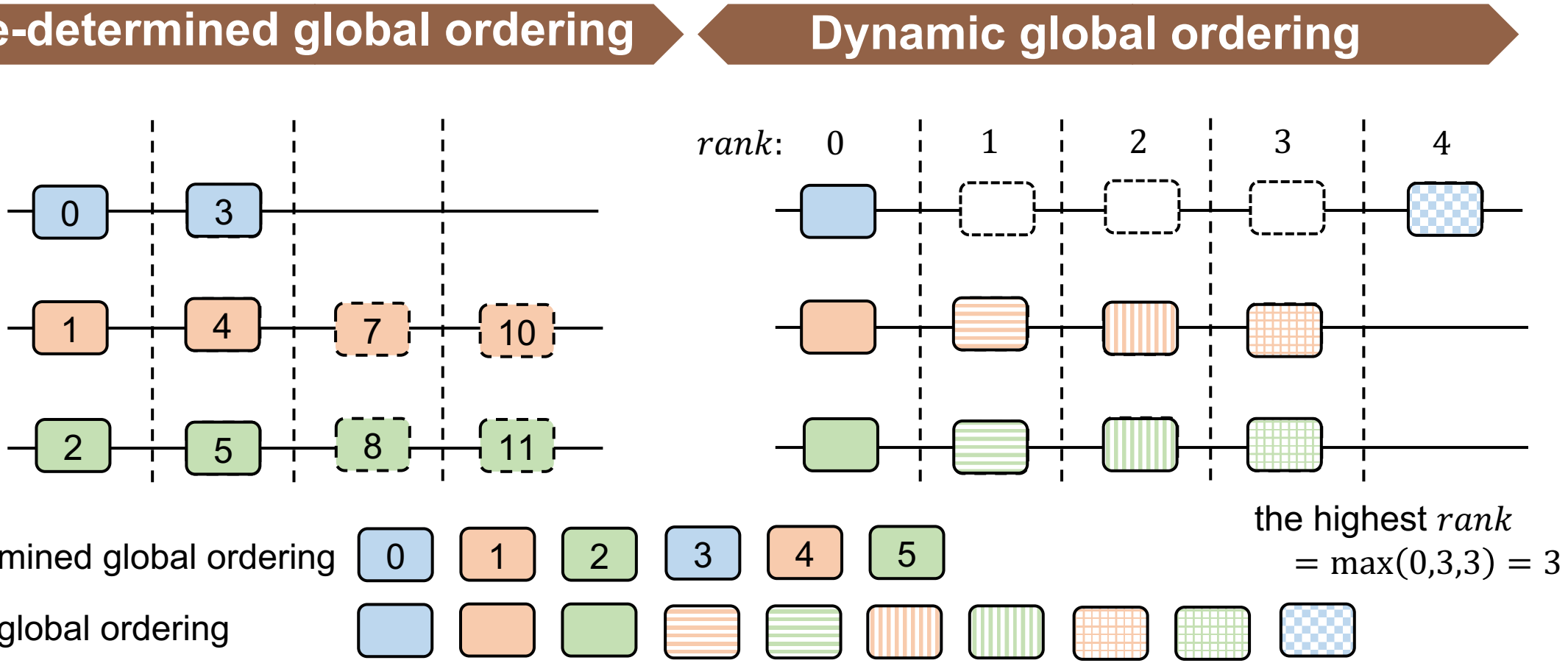
If a new block could jump directly to the front, it would no longer obstruct global ordering.



Dynamic Global Ordering

We use a dynamically generated parameter $rank$ to mark the position of a new block.

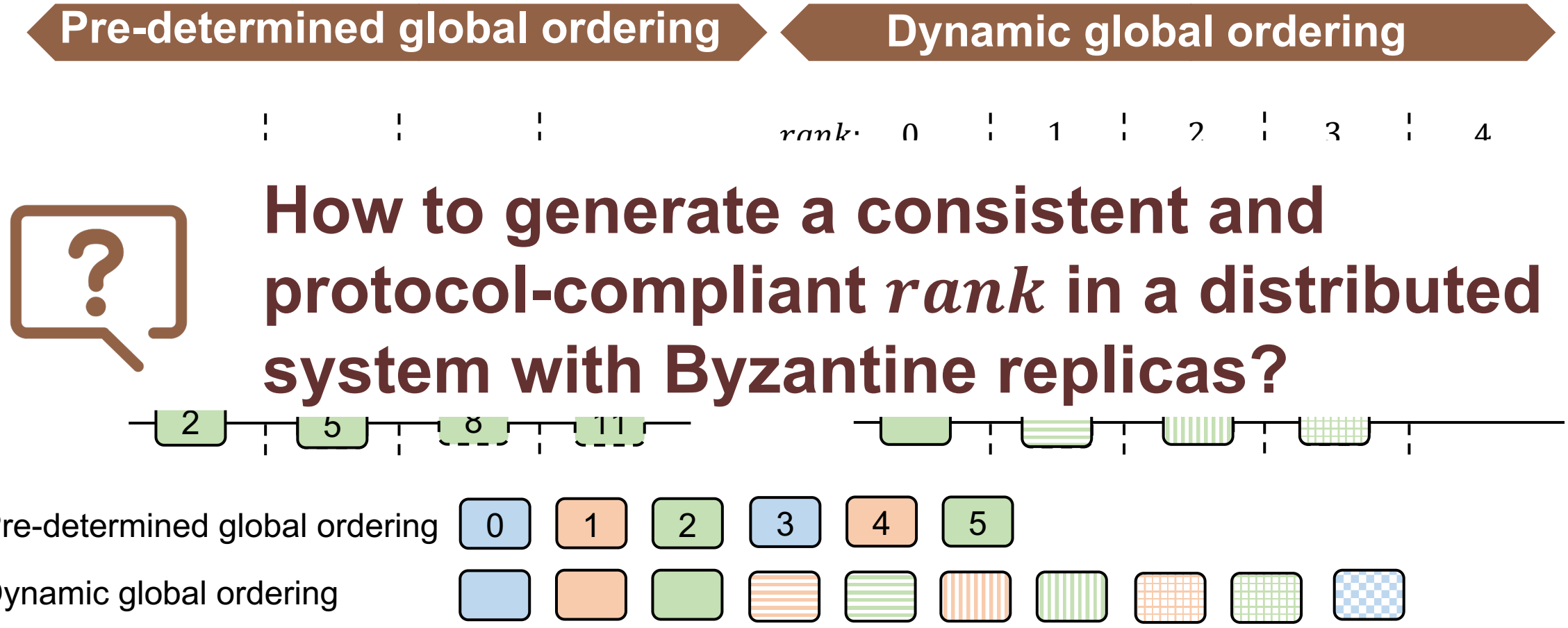
- The $rank$ of a new block = the highest $rank$ $+1$



Dynamic Global Ordering

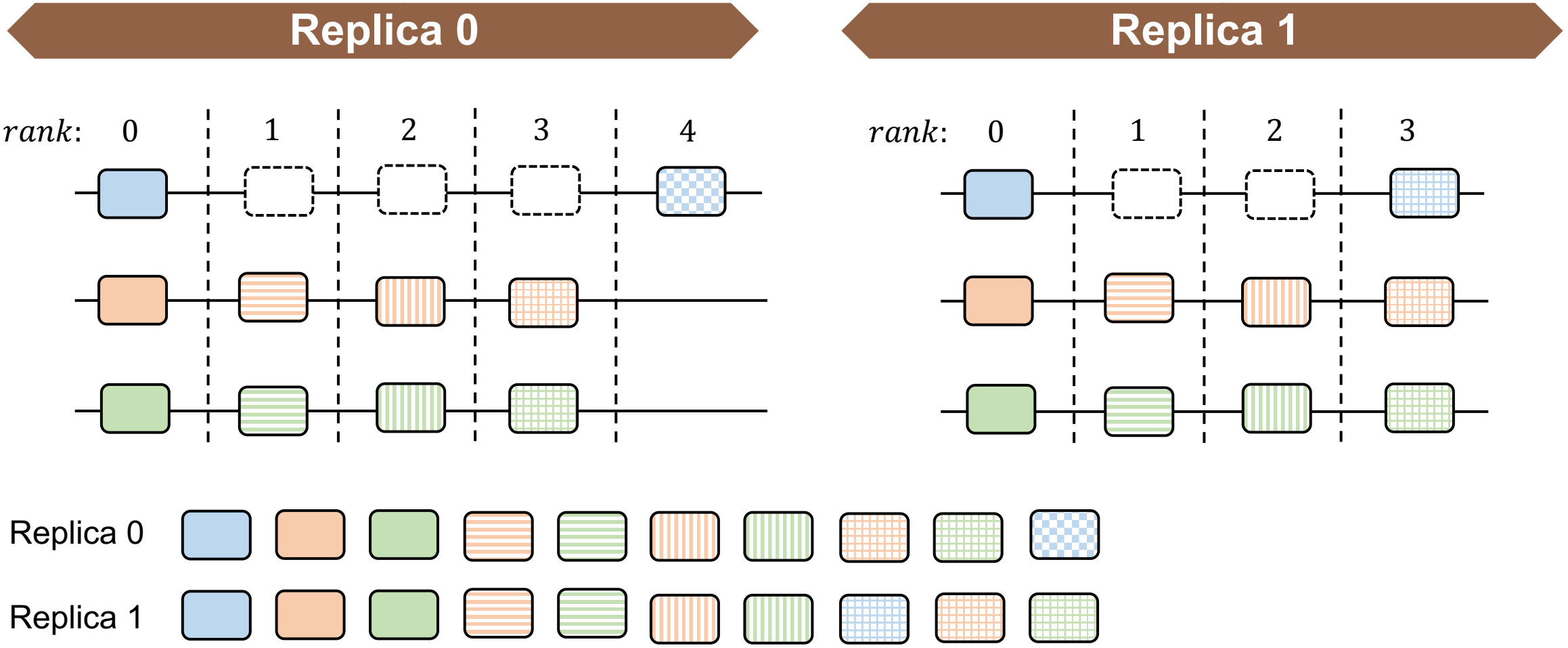
We use a dynamically generated parameter $rank$ to mark the position of a new block.

- The $rank$ of a new block = the highest $rank$ $+1$



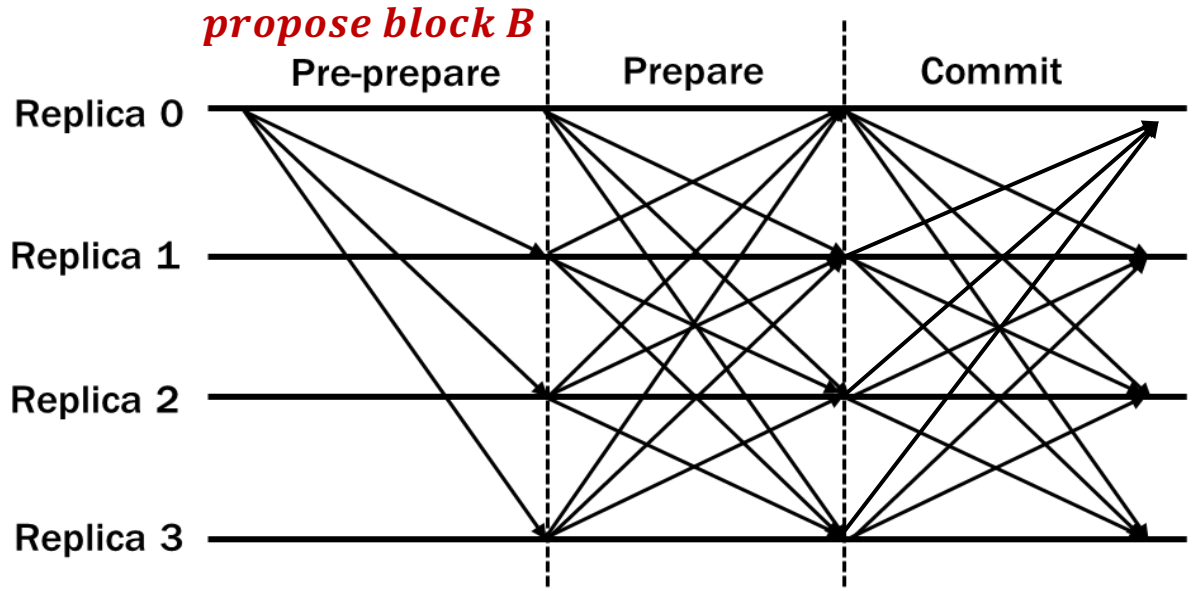
Dynamic Global Ordering

Challenge 1: Different replicas may have different views on system state



Dynamic Global Ordering

Solution: Make rank as a parameter of the block when the leader proposes



B =(txs,$index$,$round$,$rank$)

txs: a batch of clients' transactions

$index$: the index of the consensus instance

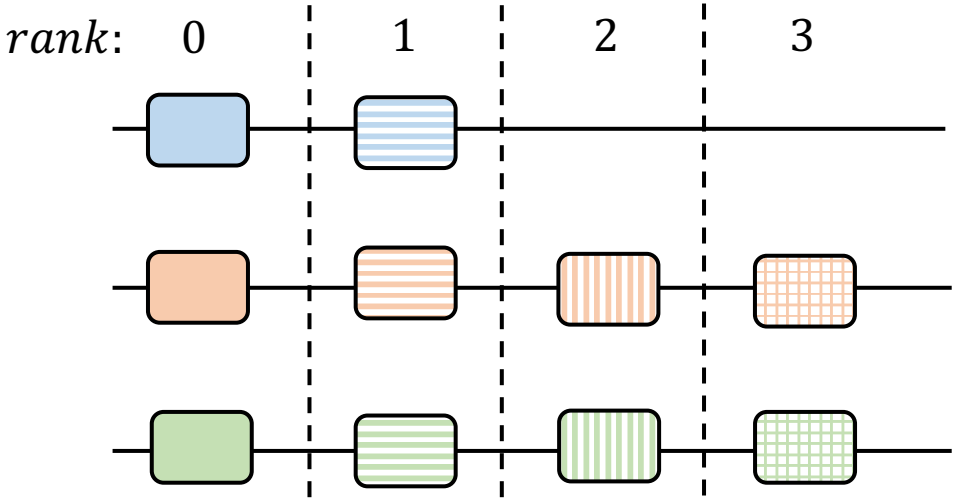
$round$: the consensus round

Agreement: All honest replicas have the same rank for a partially committed block.

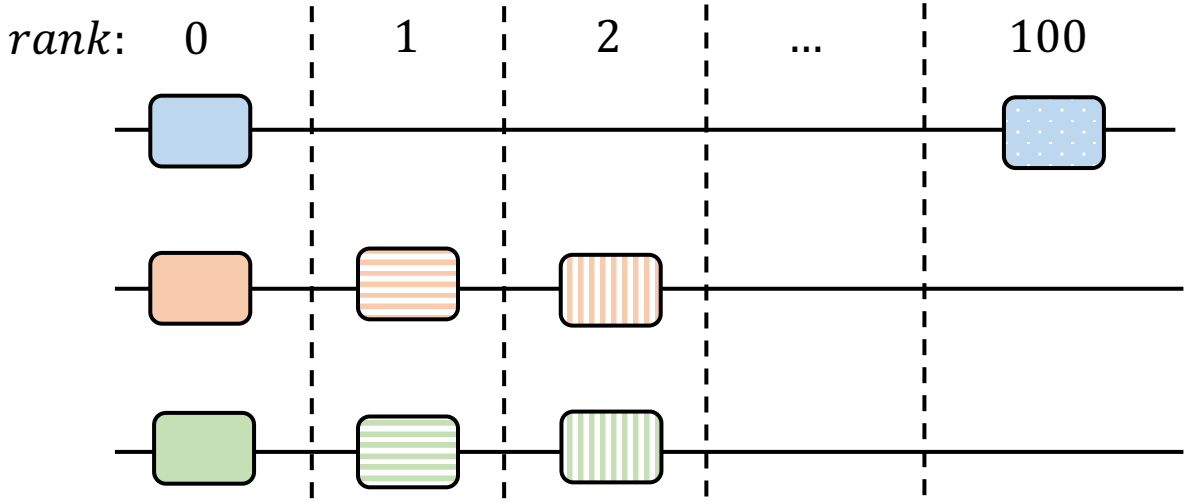
Dynamic Global Ordering

Challenge 2: A malicious leader might generate a fake rank

Case 1

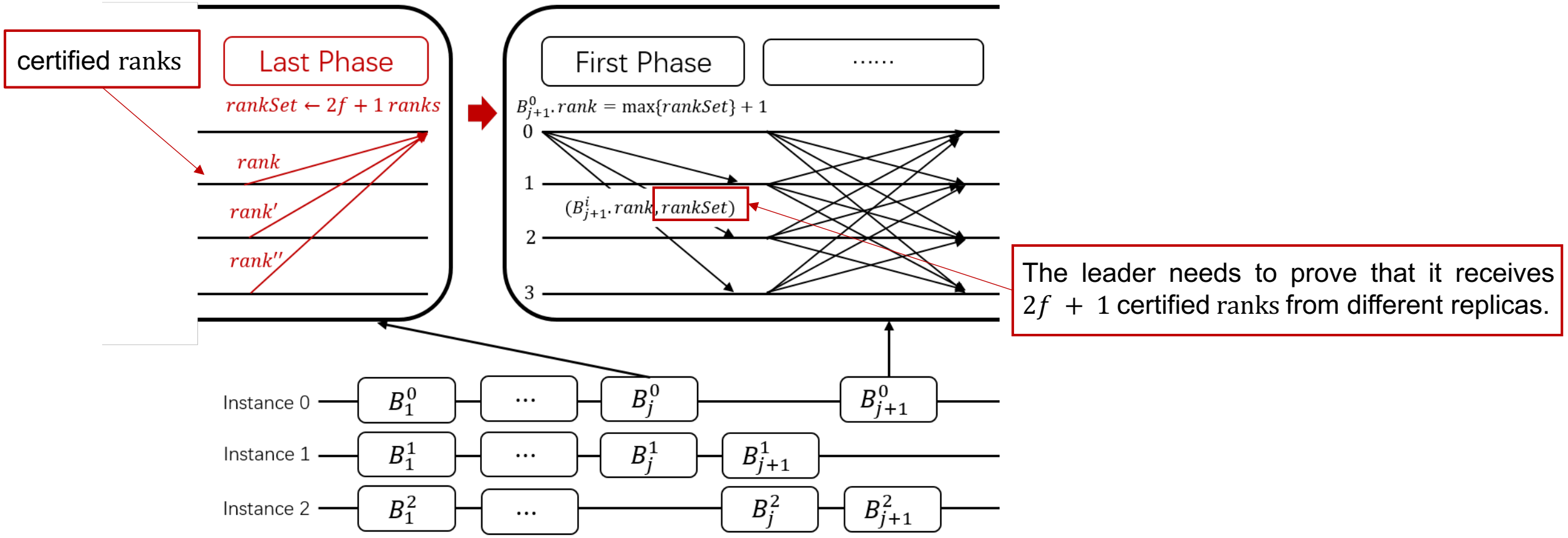


Case 2



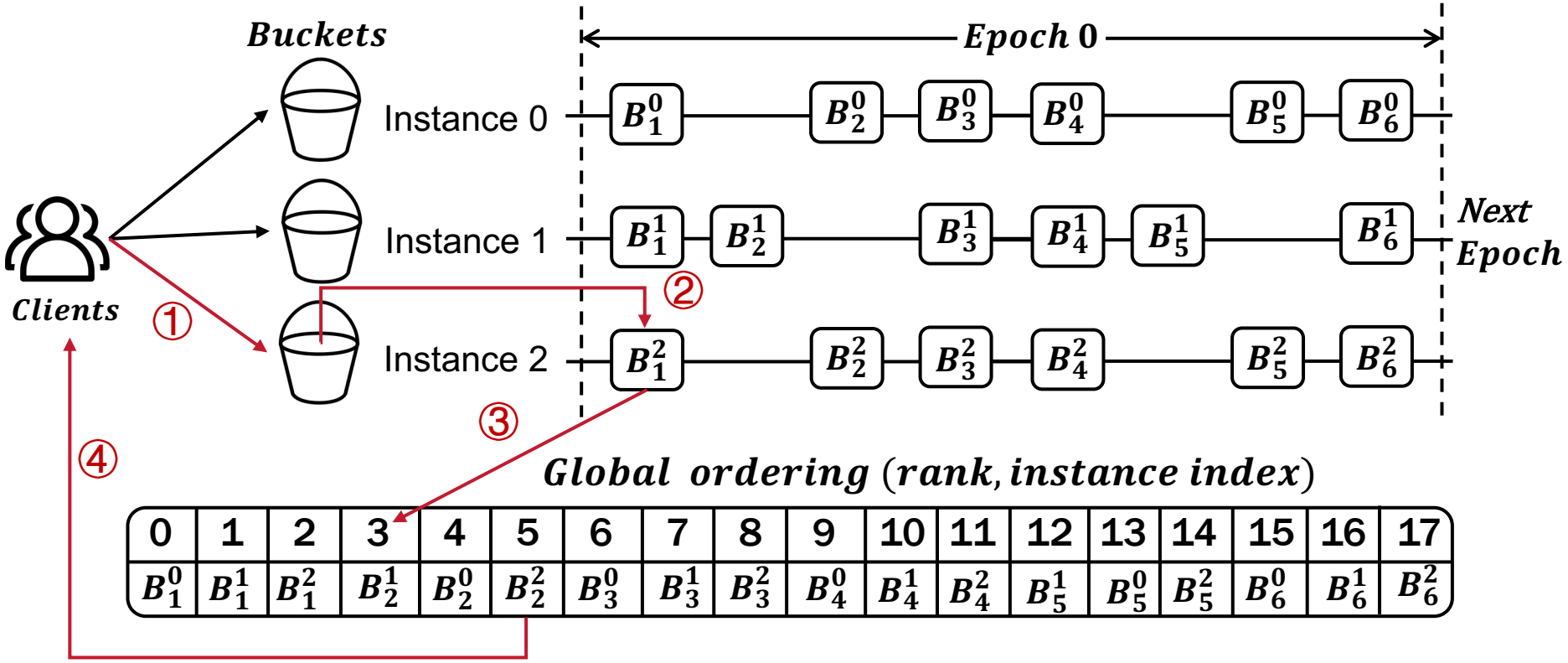
Dynamic Global Ordering

Solution: A leader collects the highest ranks from at least $2f + 1$ replicas.



Monotonicity: If a block B' is generated after an intra-instance (or a partially committed inter-instance) block B, then the rank of B' is larger than the rank of B.

Ladon Overview



Evaluation Settings

Testbeds

- AWS EC2 c5a.2xlarge machines
- 8vCPUs and 16GB RAM Ubuntu Linux 22.04
- Deployed both in LAN and WAN
- 1Gbps bandwidths

Baseline protocols :

- ISS[1]: PBFT/HotStuff
- Mir-BFT[2]: PBFT
- RCC[3]: PBFT
- DQBFT[4]: PBFT

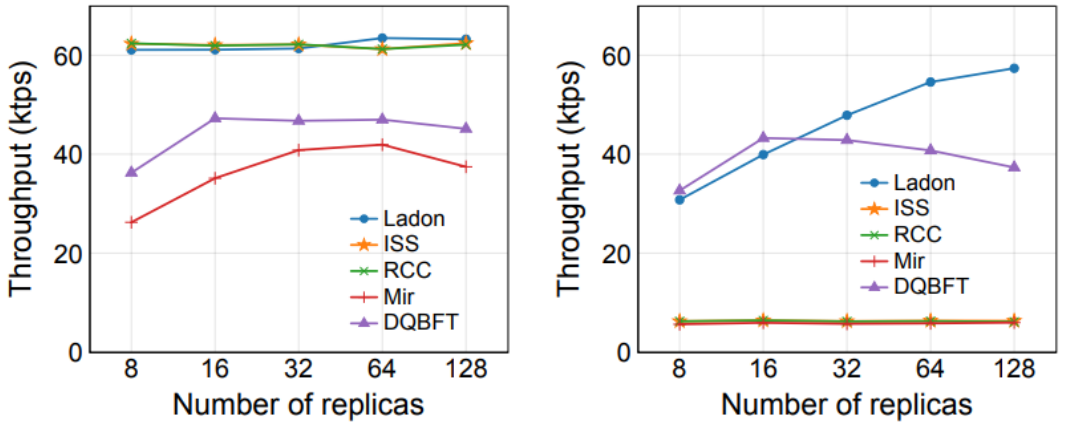
Research questions:

- How does Ladon perform with varying replicas as compared to the Baseline protocols?
- How does Ladon perform under crash and Byzantine faults?

[1] Chrysoula Stathakopoulou, Matej Pavlovic, and Marko Vukolić. State Machine Replication Scalability Made Simple, in ACM EuroSys'22.
[2] Chrysoula Stathakopoulou, Tudor David, and Marko Vukolic. Mir BFT: Scalable and Robust BFT for Decentralized Networks, in Jsys'22.
[3] Suyash Gupta, Jelle Hellings, and Mohammad Sadoghi. RCC: Resilient Concurrent Consensus for High-Throughput Secure Transaction Processing, in IEEE ICDE'21.
[4] Balaji Arun and Binoy Ravindran. Scalable Byzantine Fault Tolerance via Partial Decentralization, in PVLDB'22.

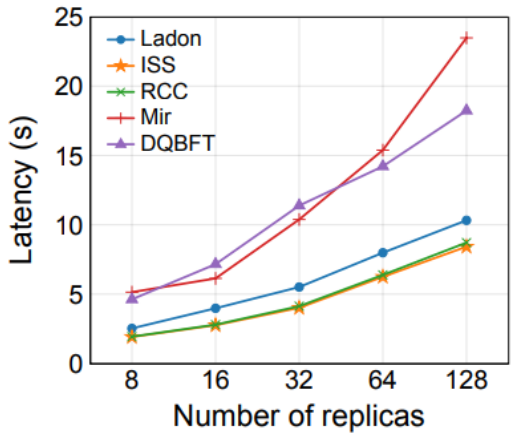
Evaluation

System scalability with varying number of replicas in WAN

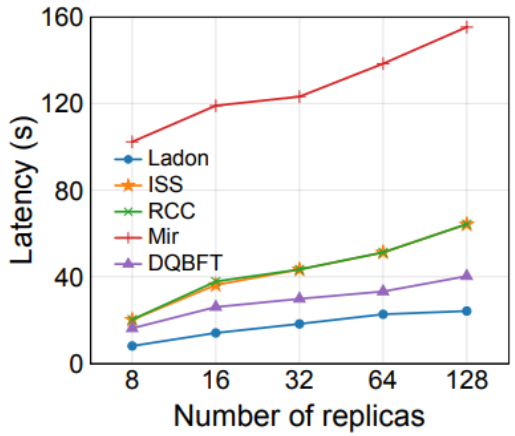


(a) #Straggler=0, WAN

(b) #Straggler=1, WAN



(c) #Straggler=0, WAN

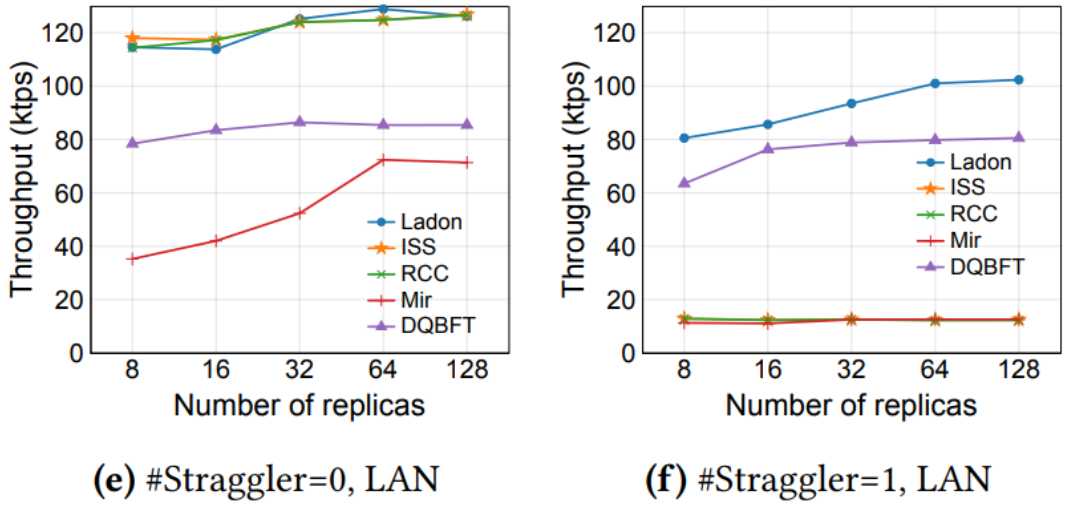


(d) #Straggler=1, WAN

- 8/16/32/64/128 total replicas
- 0/1 stragglers
- WAN
- Without stragglers
  - Comparable throughput/latency
- With one straggler
  - Superior throughput (9X)
  - Lower latency

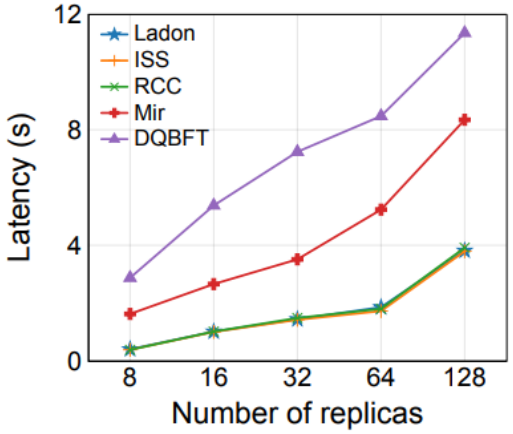
Evaluation

System scalability with varying number of replicas in LAN

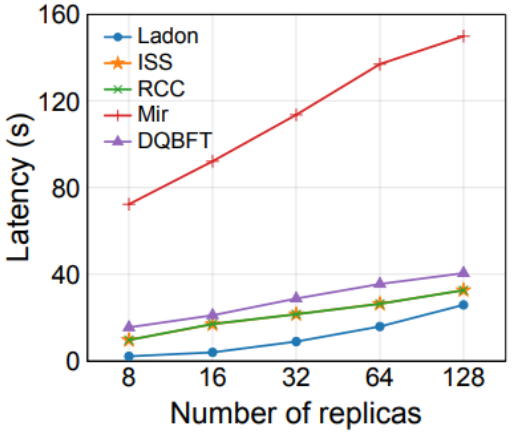


(e) #Straggler=0, LAN

(f) #Straggler=1, LAN



(g) #Straggler=0, LAN

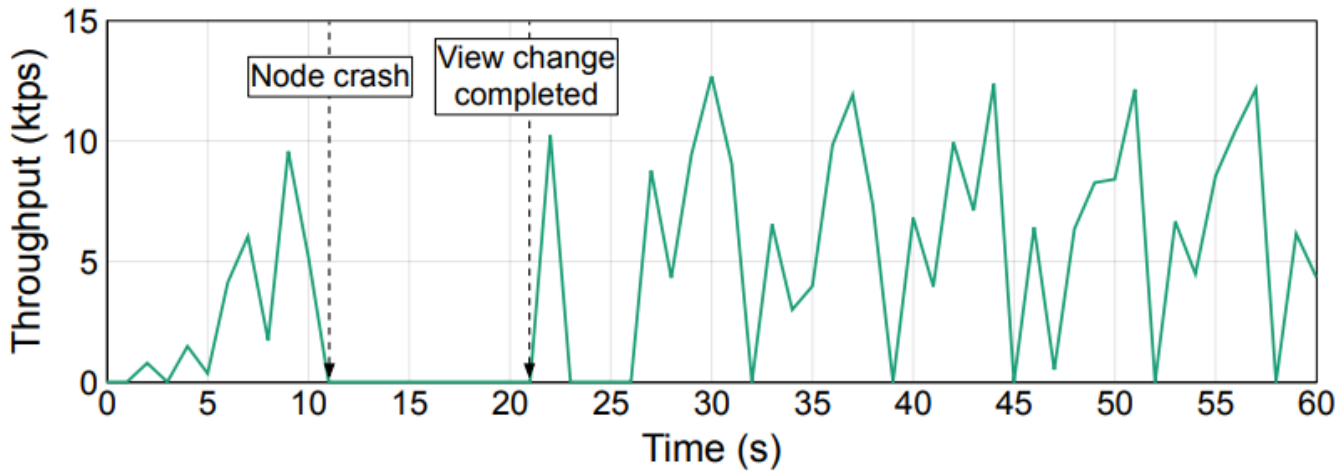


(h) #Straggler=1, LAN

- 8/16/32/64/128 total replicas
- 0/1 stragglers
- LAN
- Similar performance trends to WAN

Evaluation

Impact of crash faults

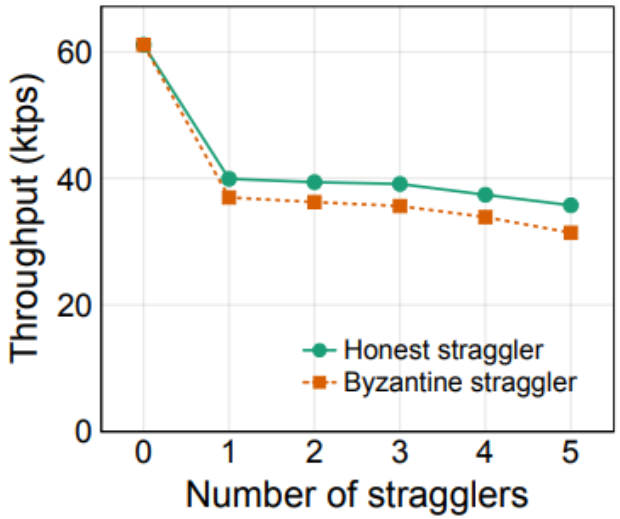


- 16 total replicas with 1 Crash leader
- View change timeout sizes: 10s
- Temporary stall when view change
- Recovers after view change

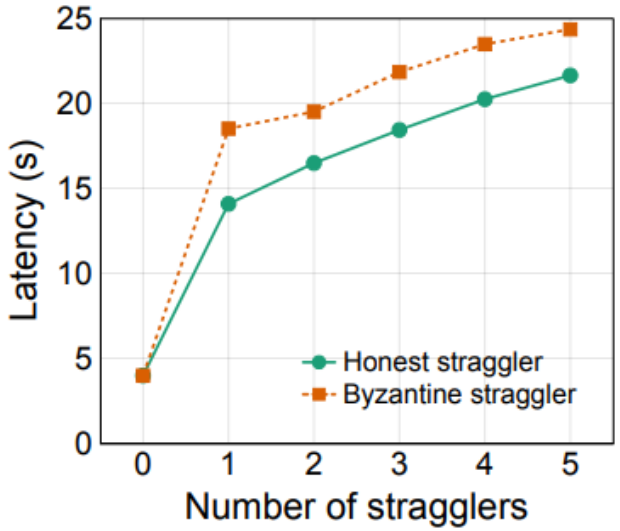
Ladon's throughput average (over 1s intervals) over time with one crash fault.

Evaluation

Impact of Byzantine faults



(a) Throughput



(b) Latency

- 16 total replicas with 1-5 stragglers
- Byzantine/honest stragglers
- Slight difference
- Performance degradation is gradual

Summary

Ladon: A high-performance Multi-BFT protocol that mitigates the impact of stragglers

- Code: **https://github.com/eurosys2024ladon/ladon**
- Adopting dynamic global ordering to assign a global ordering index to blocks according to the real-time states of all instances.
- Using monotonic ranks and pipeline their distribution and collection with the consensus process.

THE UNIVERSITY OF BRITISH COLUMBIA

SUSTech Southern University of Science and Technology

Thanks for Listening!

Hanzheng Lyu