

Ambulance

Saving BFT through Racing

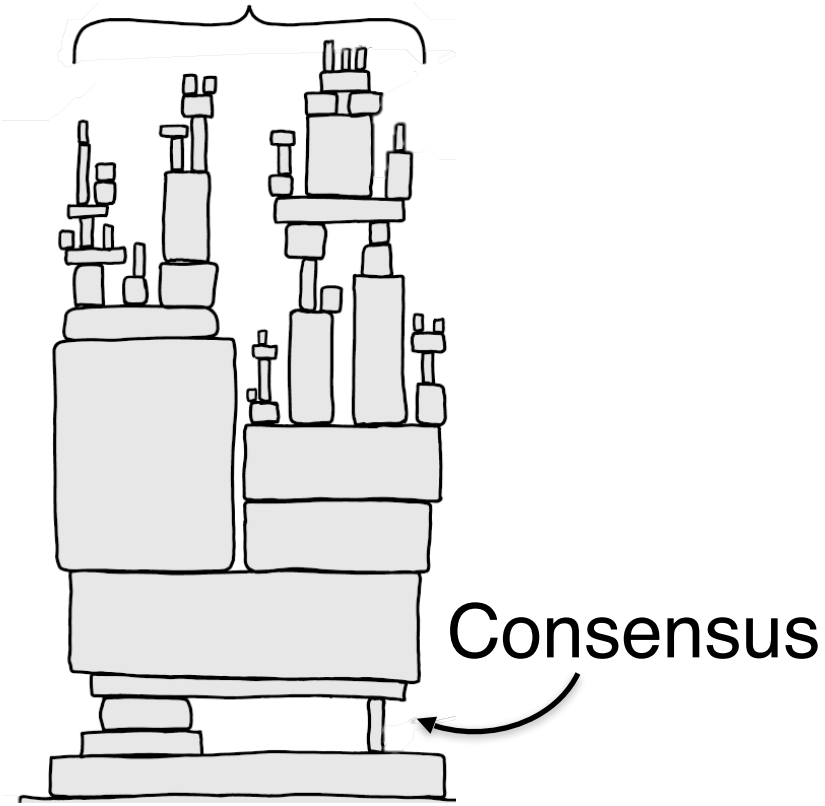
Neil Giridharan

Shubham Mishra, Lorenzo Alvisi, Natacha Crooks, Ben Marsh

Hein Meling, Kartik Nayak, Grzegorz Prusak

Consensus and Distributed Systems

Distributed Systems

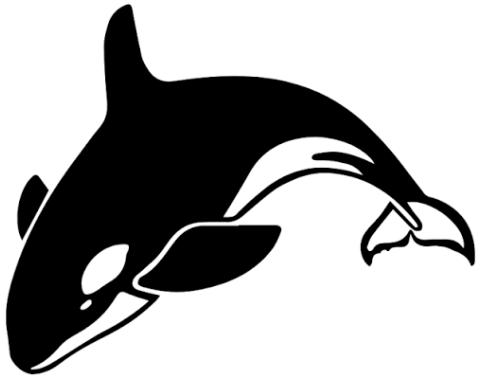


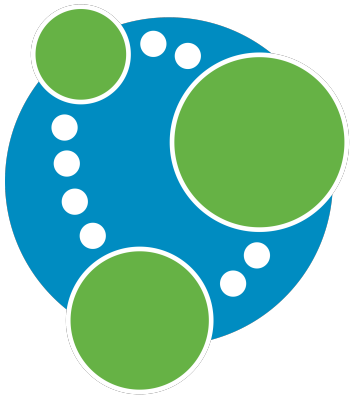
The state of
distributed systems







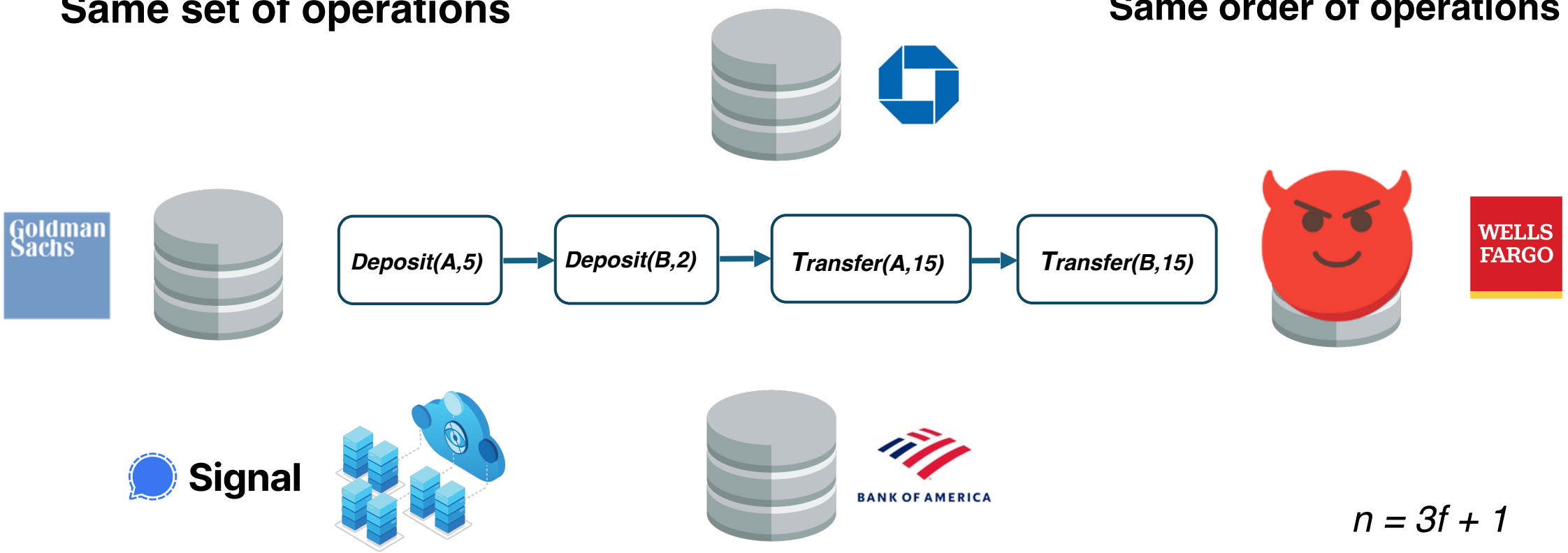




Consensus Abstraction

Same set of operations

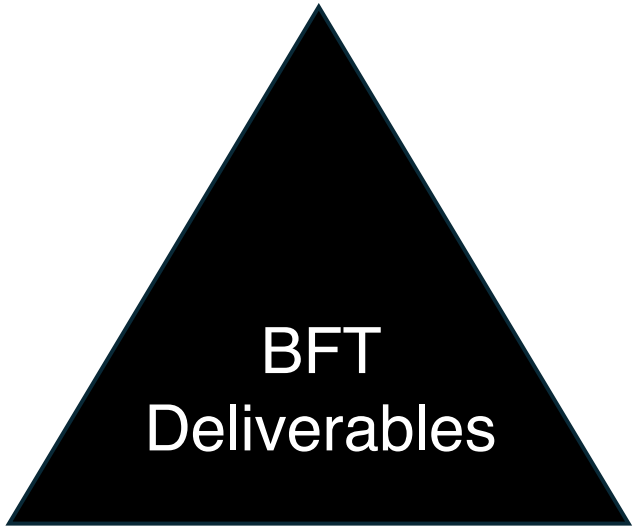
Same order of operations



$n = 3f + 1$

Challenging Demands

High Throughput



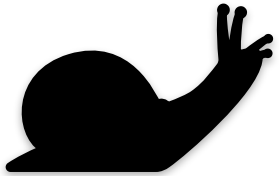


Low Latency

Robustness

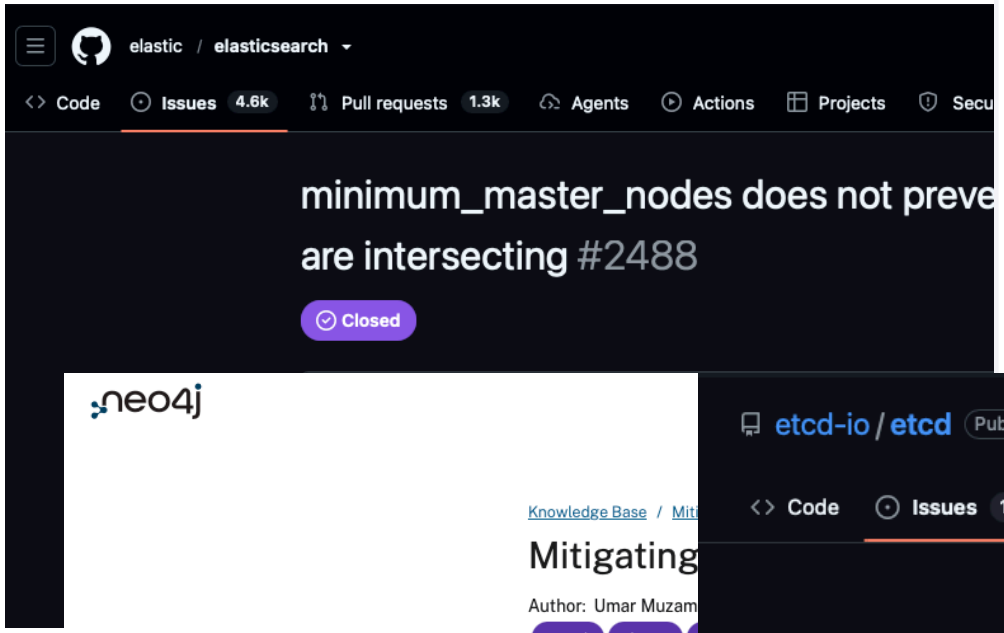
Slowdowns

Slowdown = Event causing latency degradation

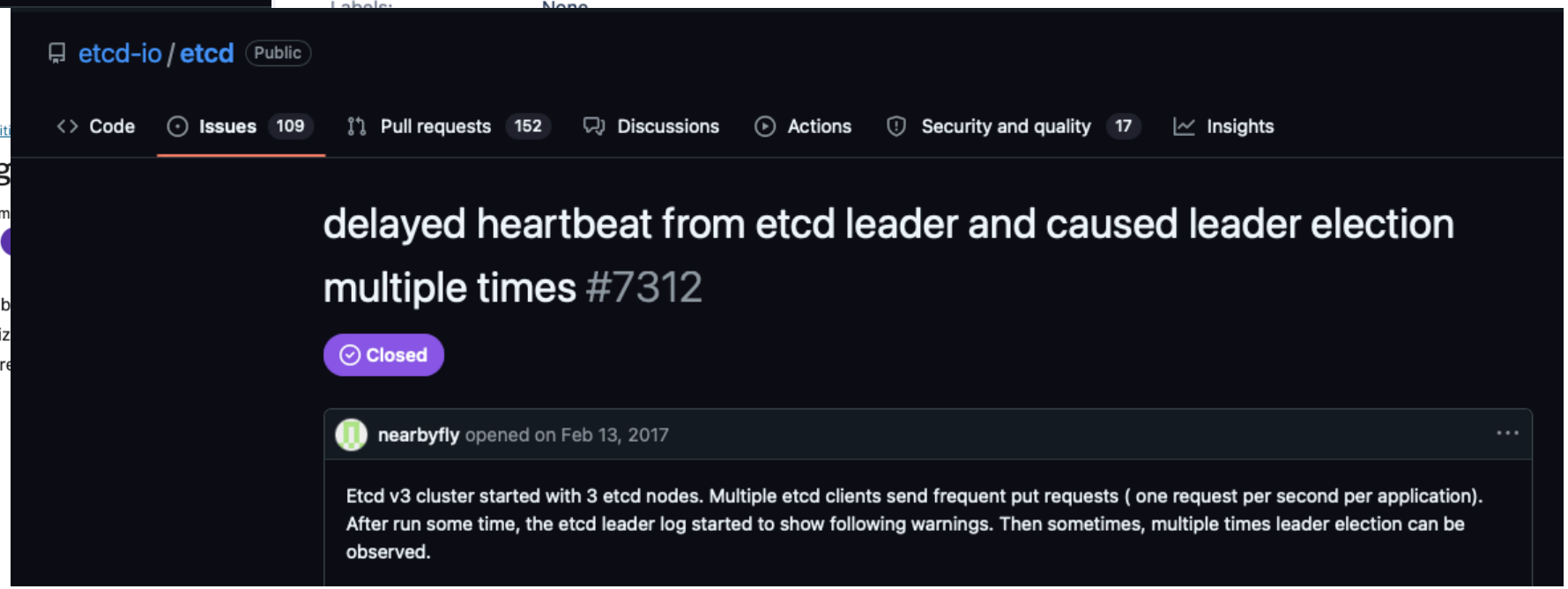


- Replica crashes
- Garbage collection
- Latency spikes
- Packet Loss

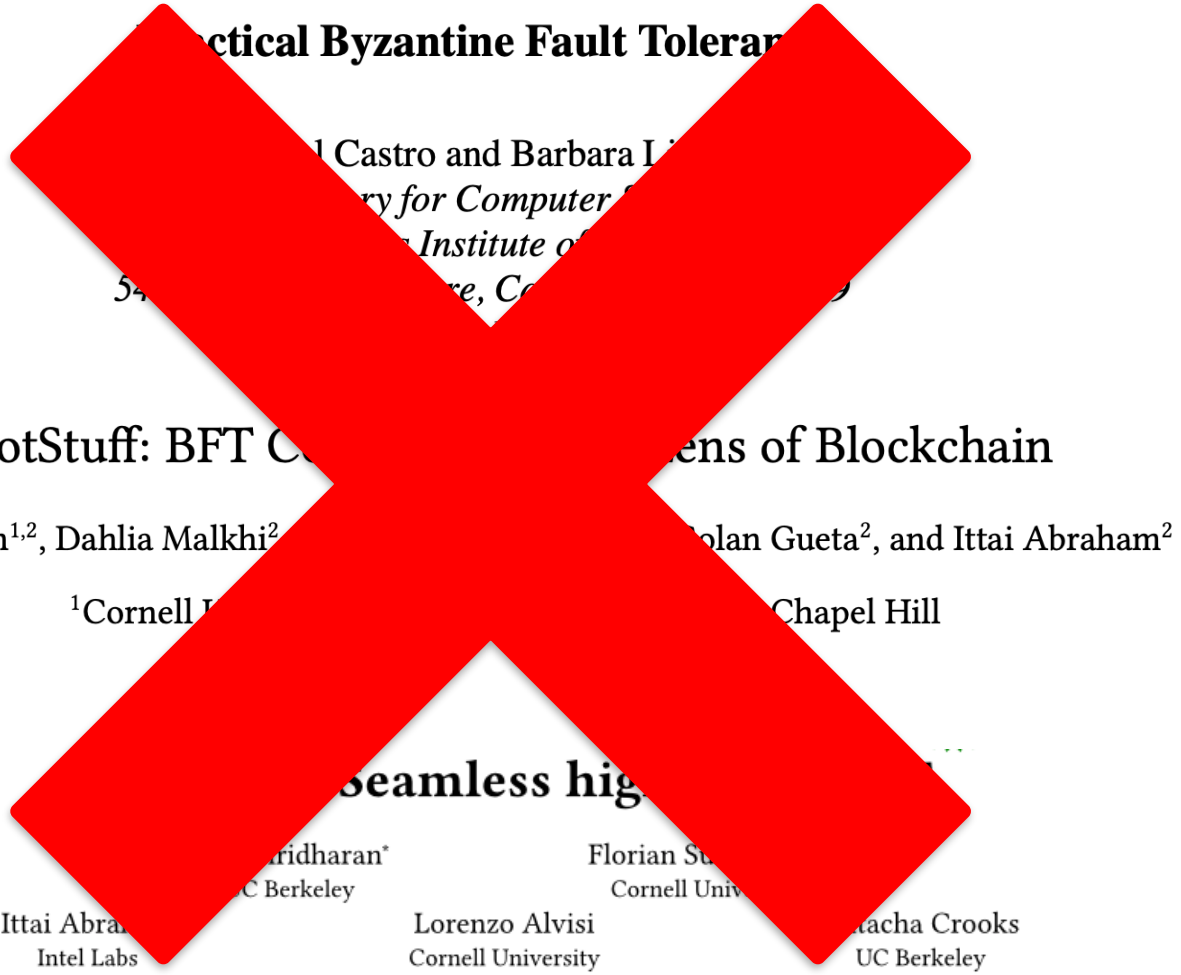
Slowdowns in the wild



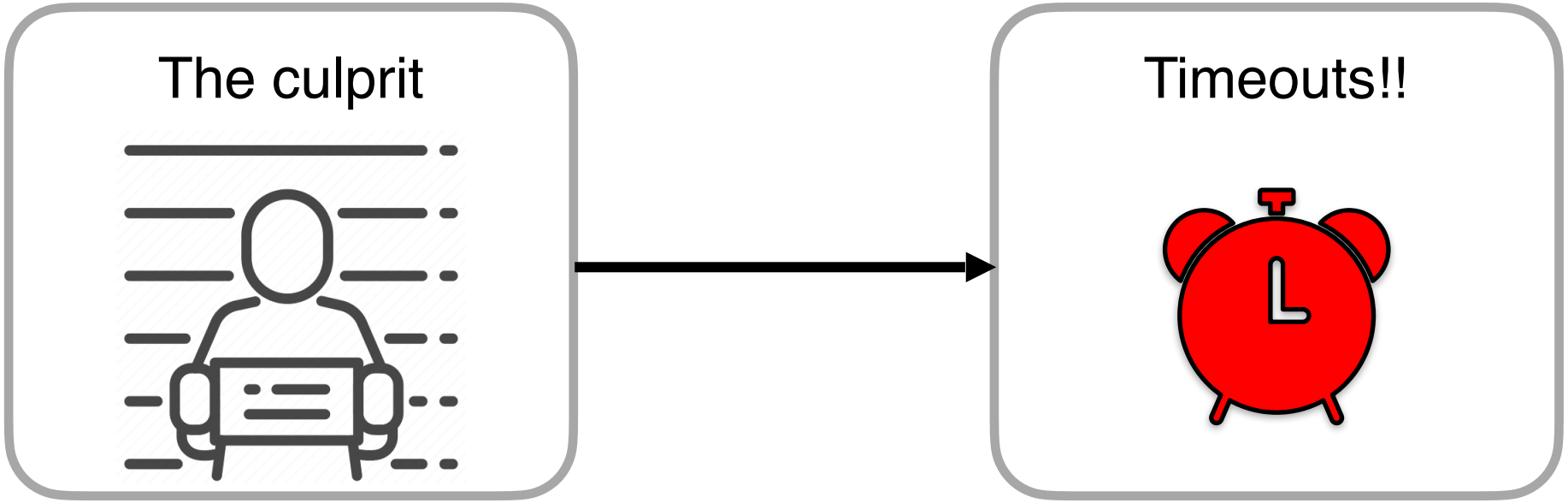




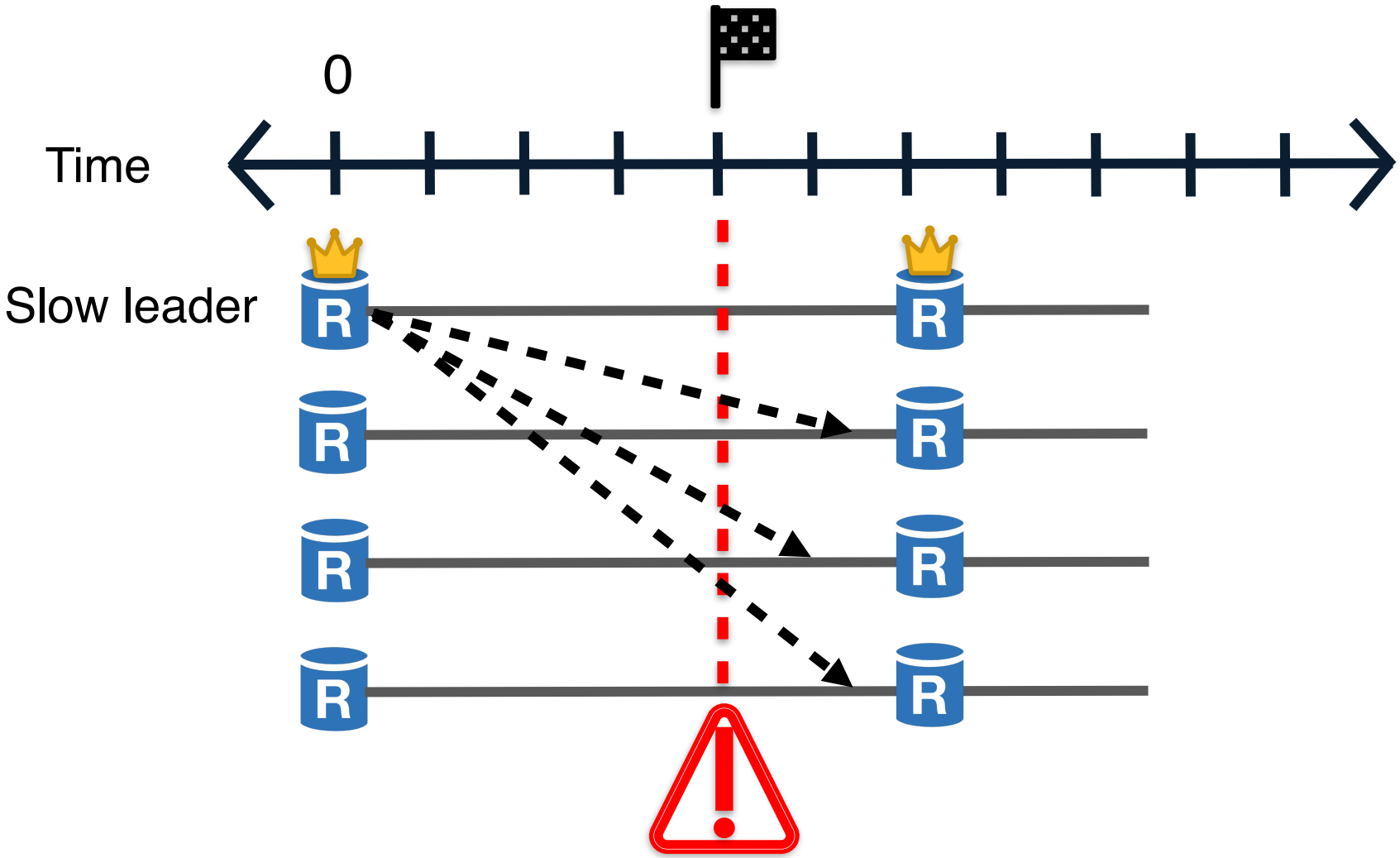
Impact of Slowdowns



Why are they not robust?



Timeouts



Timeouts are evil

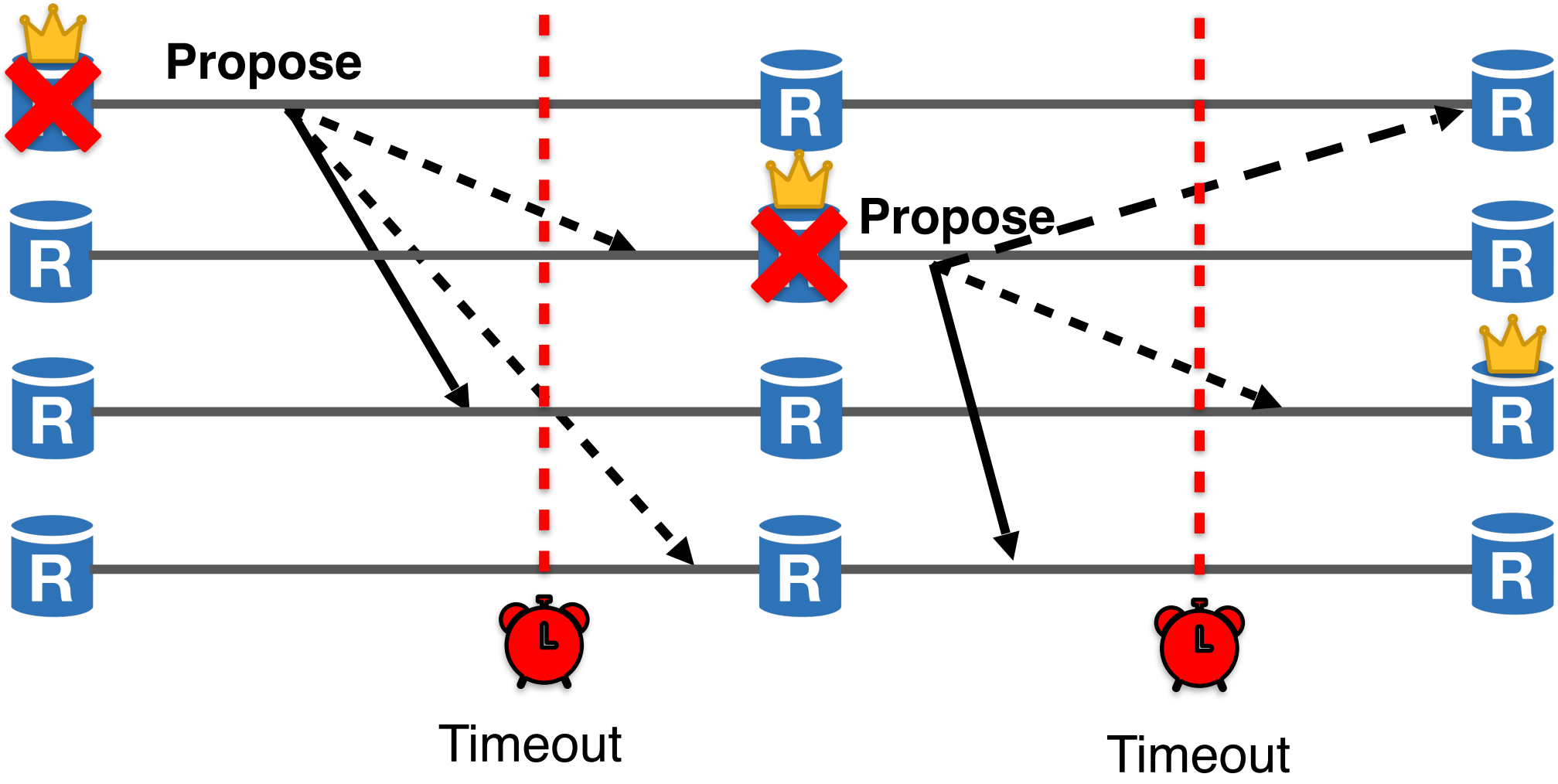
1) Destructive

Aggressive timeouts cause false alarms

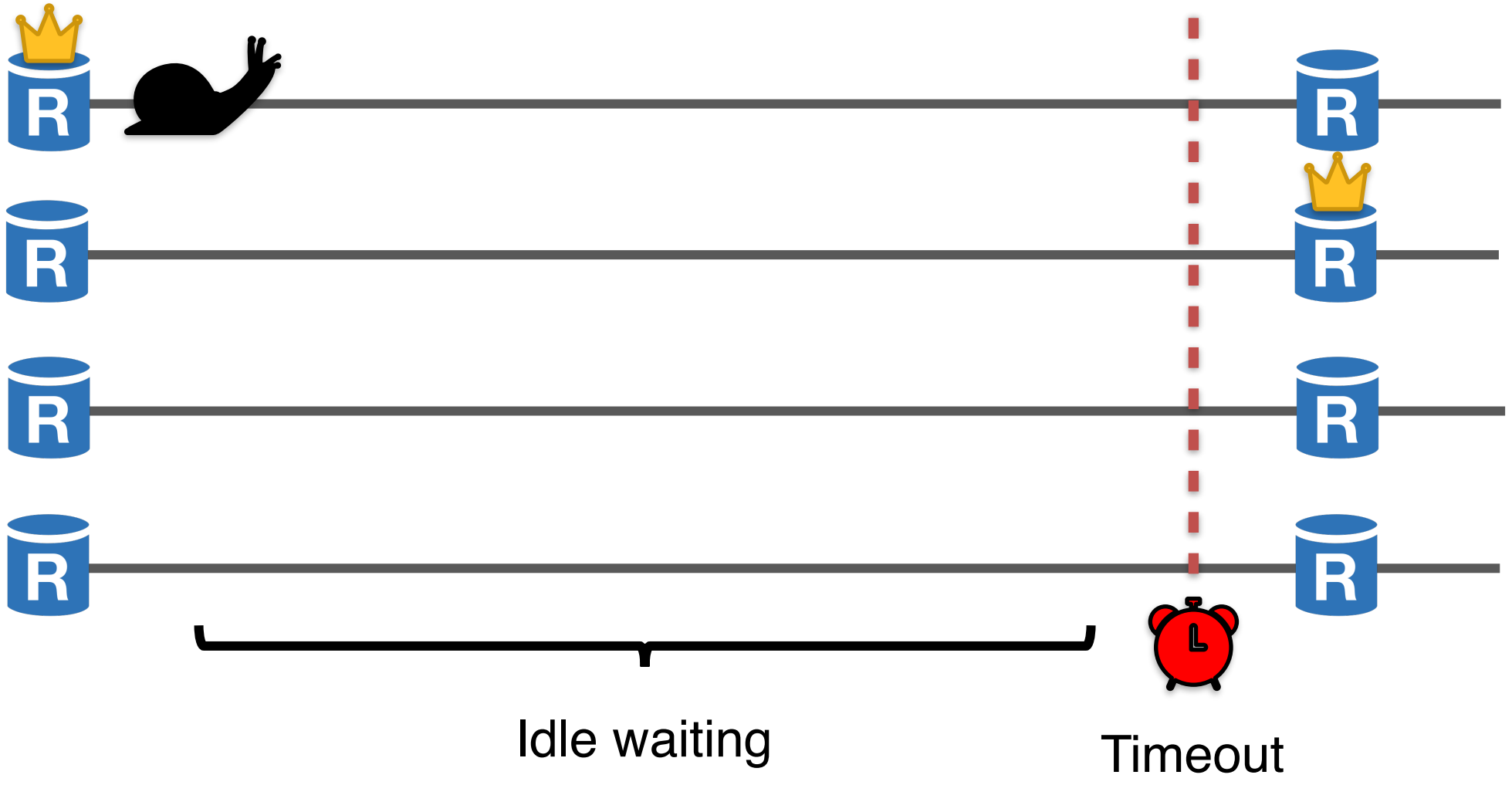
2) Blocking

Conservative timeouts delay detection

Timeouts are Destructive



Timeouts are Blocking



Slowdown Detector Requirements

Cooperative

Concurrent proposers don't block each other

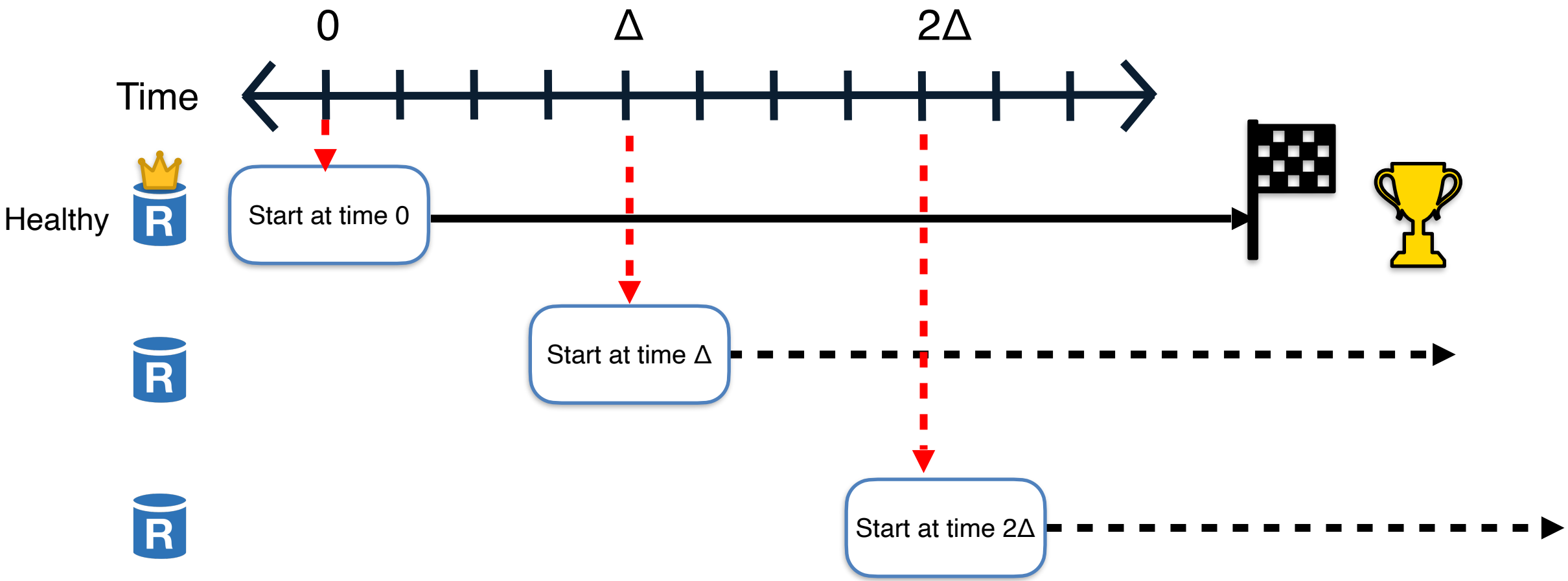
Productive

Useful work is done during slowdown detection that accelerates recovery

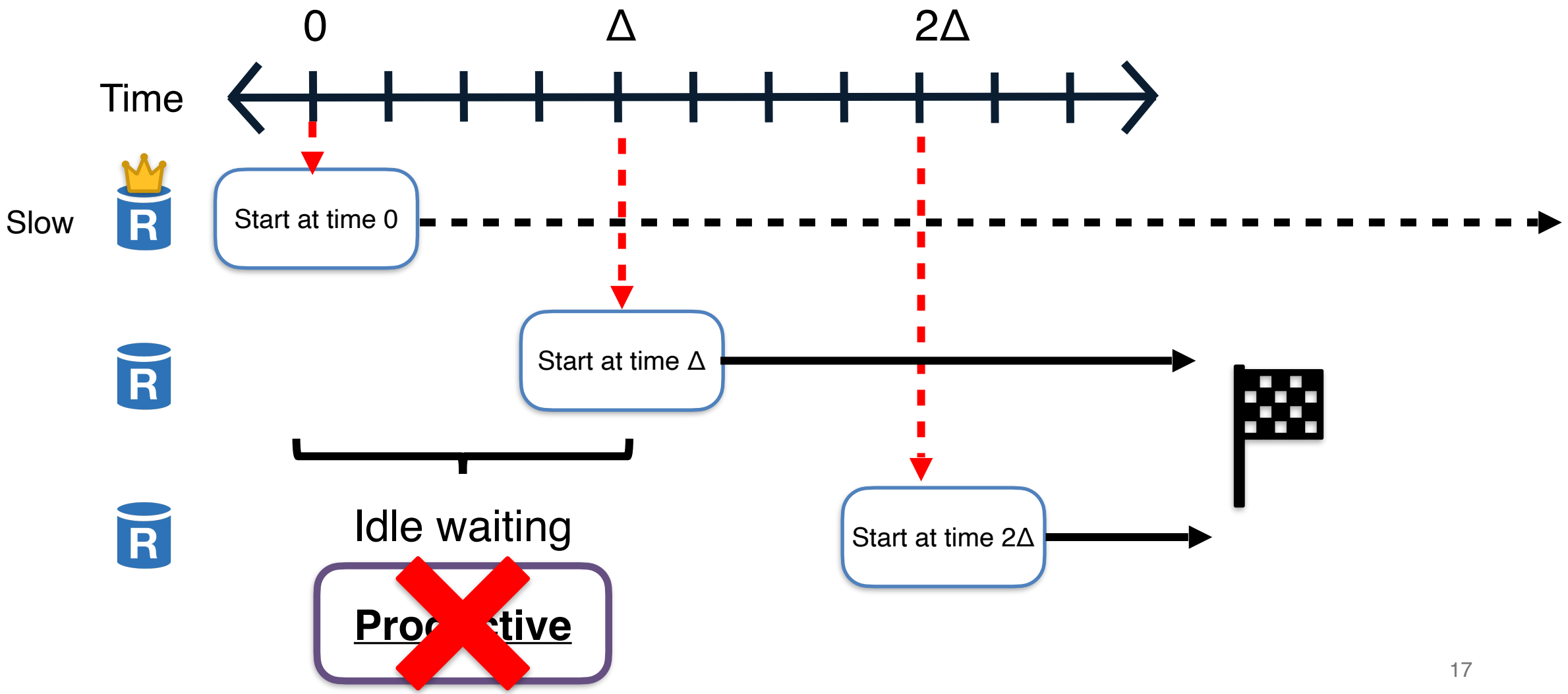
Slowdown Detection Mechanisms

	Cooperative	Productive
Timeouts	✗	✗

Hedging

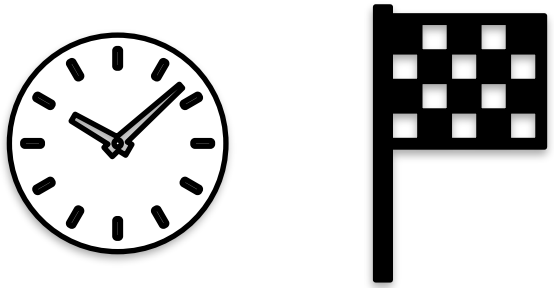

0
Δ
2Δ
Time
Healthy
R
R
R
Start at time 0
Start at time Δ
Start at time 2Δ

Hedging


0
Δ
2Δ
Time
Slow
Start at time 0
Start at time Δ
Start at time 2Δ
Idle waiting
Proactive

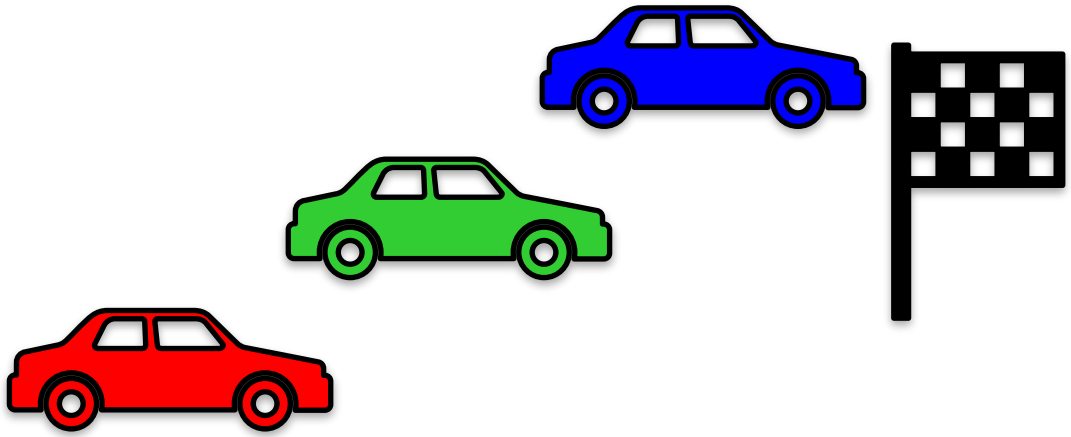
Timeouts vs Hedging

Timeouts



Race against the clock

Hedging



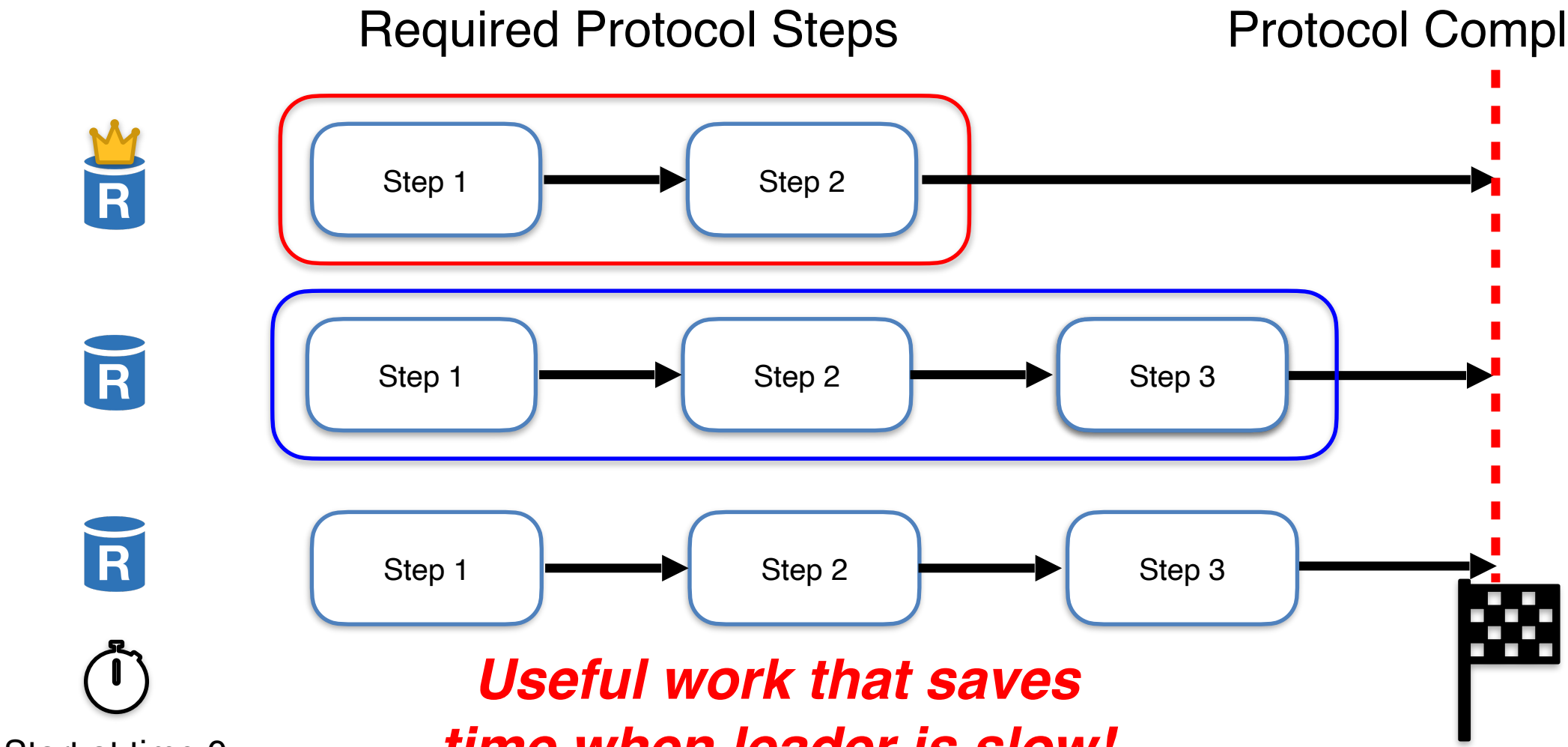
Race against each other

But leader gets a time-based head start!

Slowdown Detection Mechanisms

	Cooperative	Productive
Timeouts	✗	✗
Hedging	✓	✗

Protocol-Rigged Racing



Useful work that saves time when leader is slow!

Slowdown Detection Mechanisms

	Cooperative	Productive
Timeouts	✗	✗
Hedging	✓	✗
Protocol-Rigged Racing	✓	✓

What Ambulance Promises

Match performance of timeout-based BFT in common case

Provide stronger robustness to slowdowns

Ambulance Overview

Race

Replicas race against each other to determine whether leader is slow

Recovery Path

If leader is slow, recover proposal from race and commit

Race

BFT

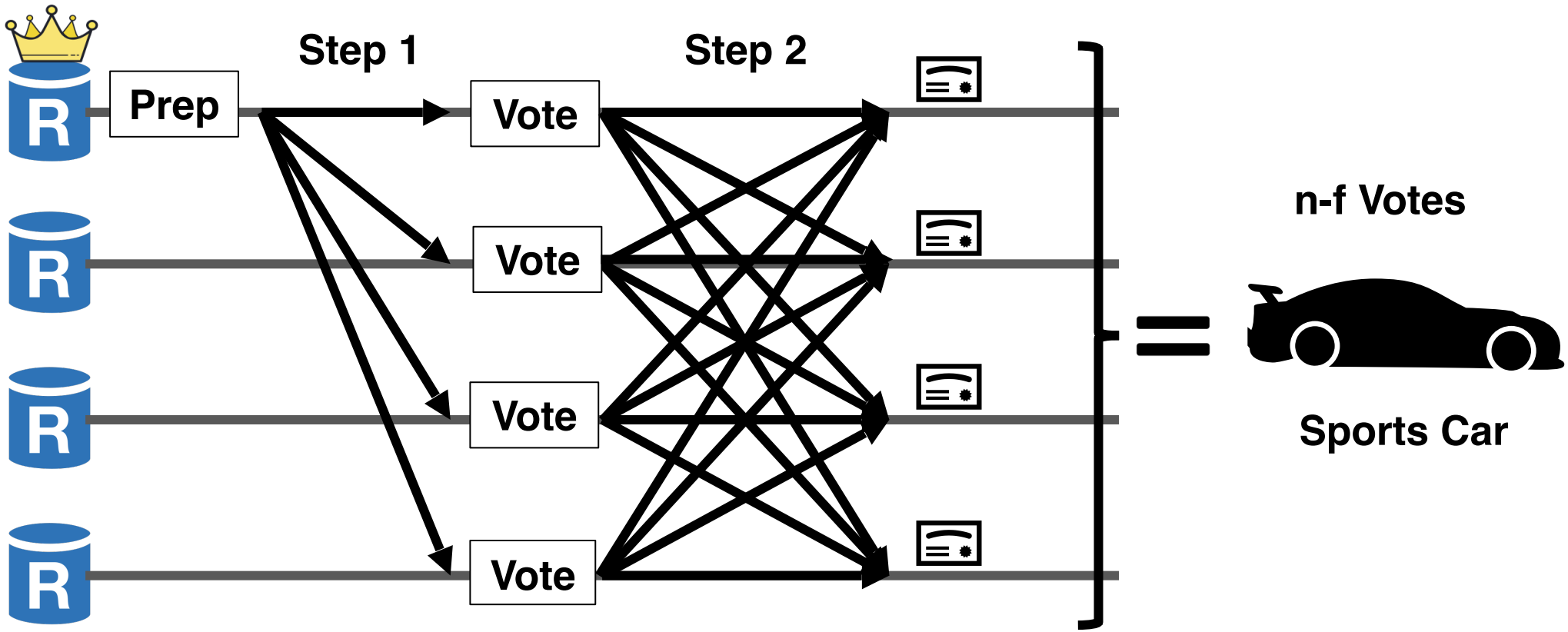
Non-equivocation phase + Persistence phase



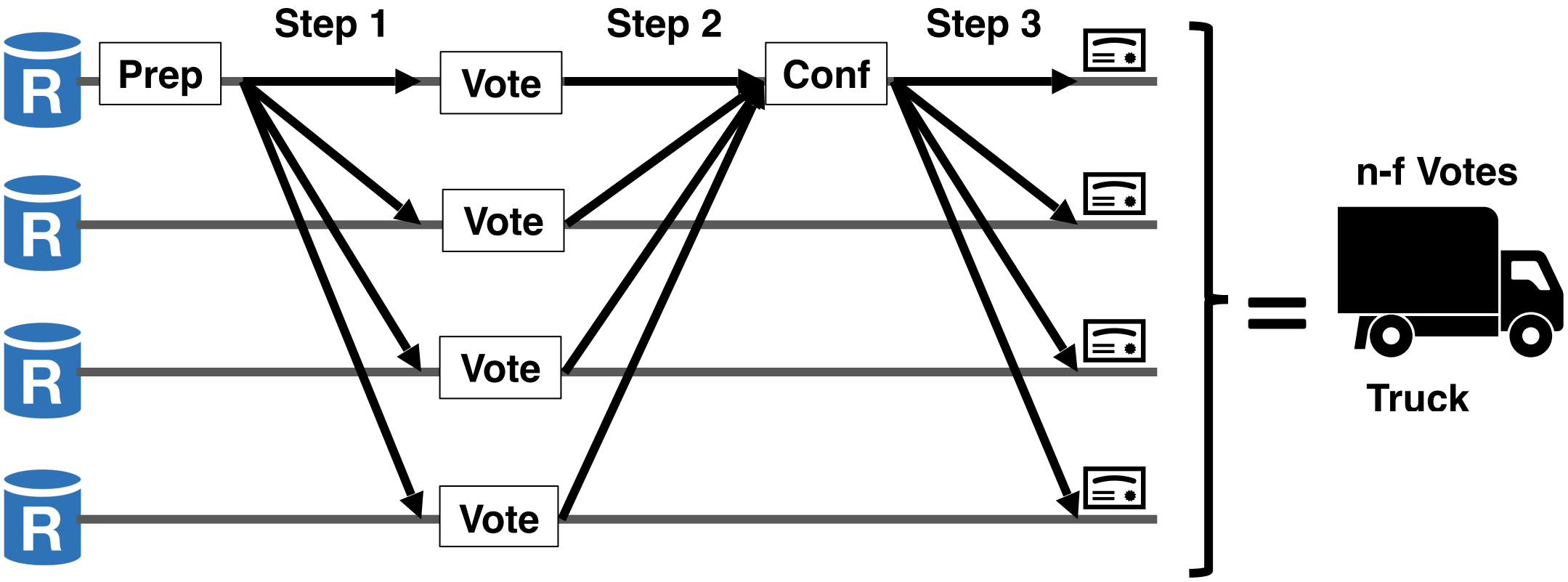
Race work

Ensures productivity!

Race: Leader protocol

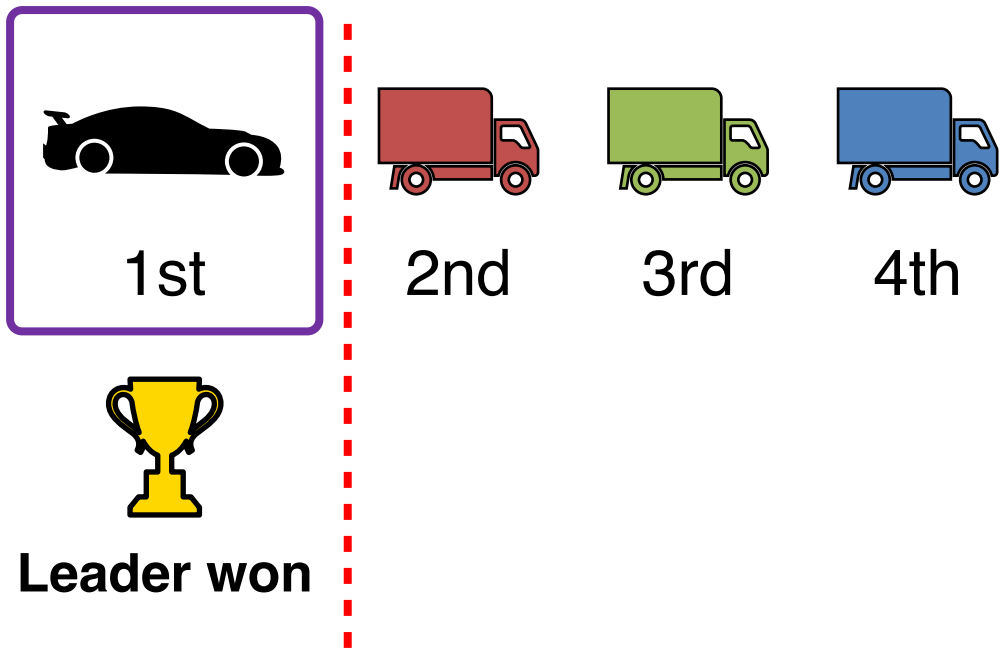


Race: Replica protocol

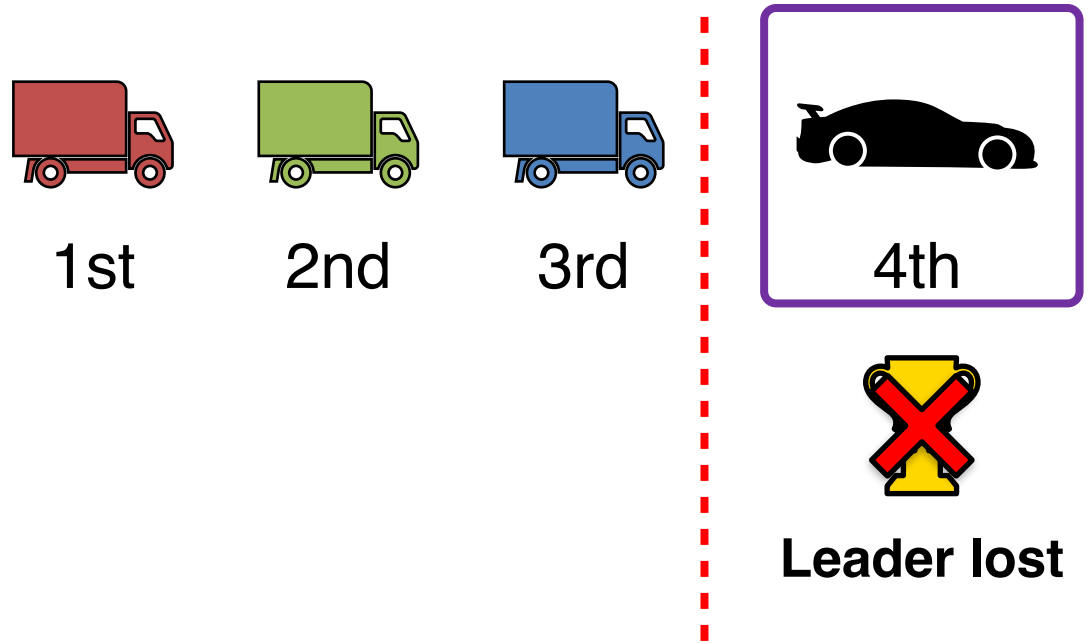


Race outcome

Replicas may disagree on race outcome!

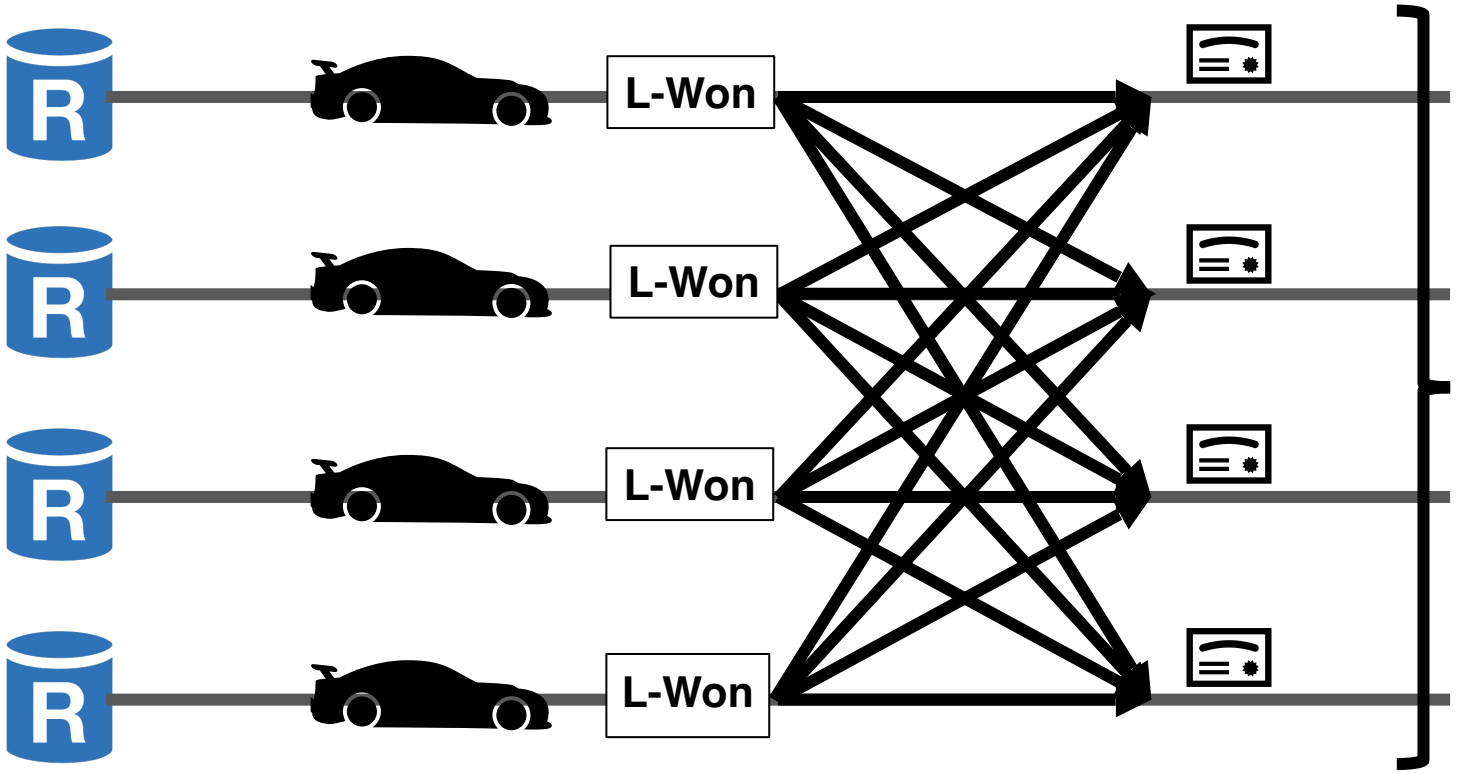


Local view of Replica 1



Local view of Replica 2

Happy Path

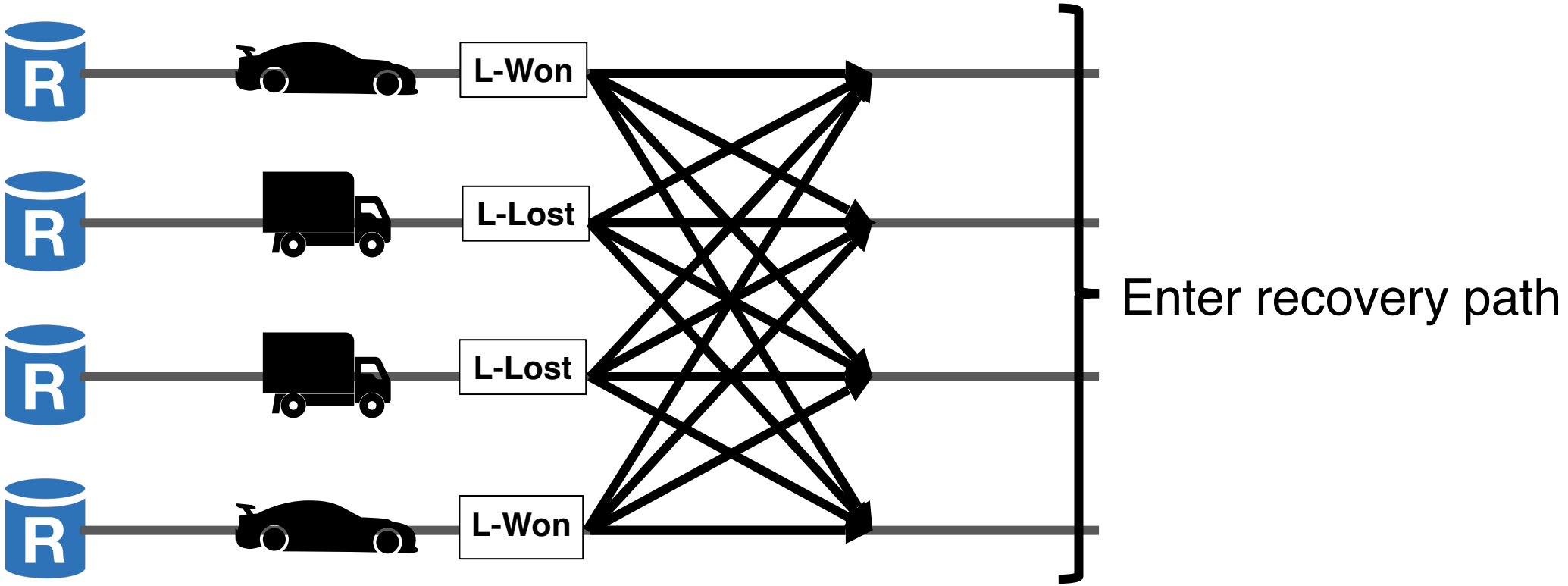


Commit certificate

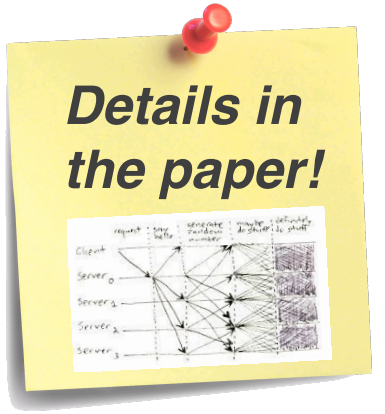
Commits as fast as traditional BFT!

Persistence phase

Unhappy Path



Recovery path



Must safely pick a single proposal from many proposers

Reuses work done during the race to ensure productivity

Evaluation: Setup

Amazon Web Services (AWS)
Elastic Cloud Compute (EC2)

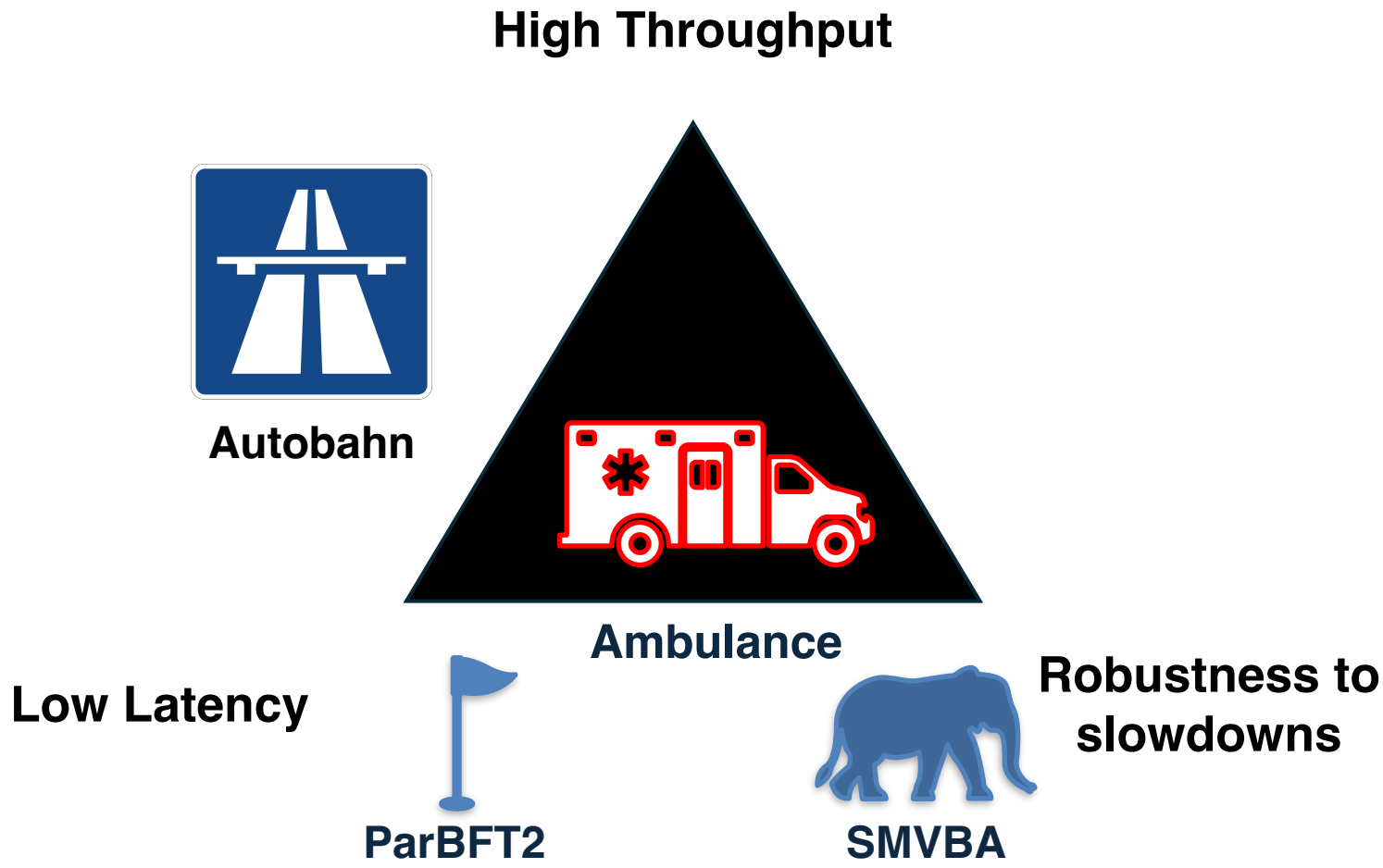
Regions:

us-west-1
us-west-2
us-east-1
us-east-2

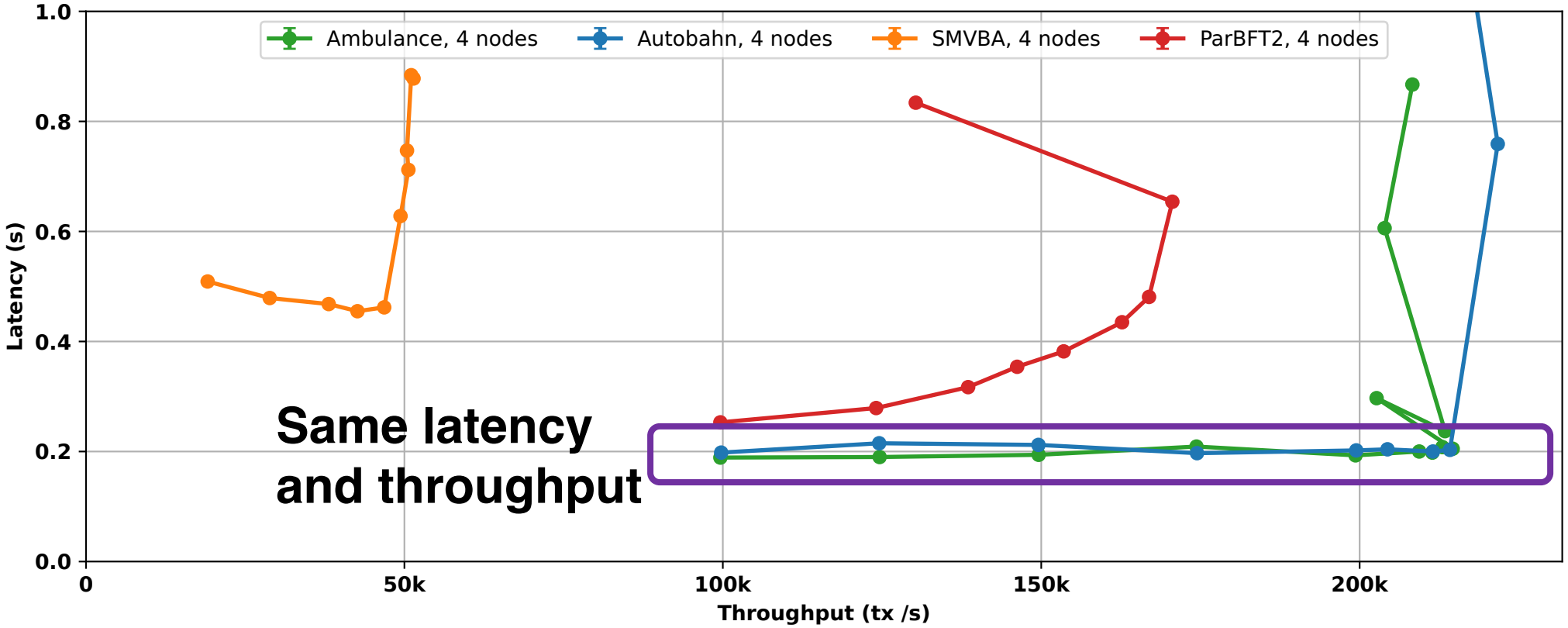
VM specs:

16 vCPU
64 GB RAM
30 GB SSD

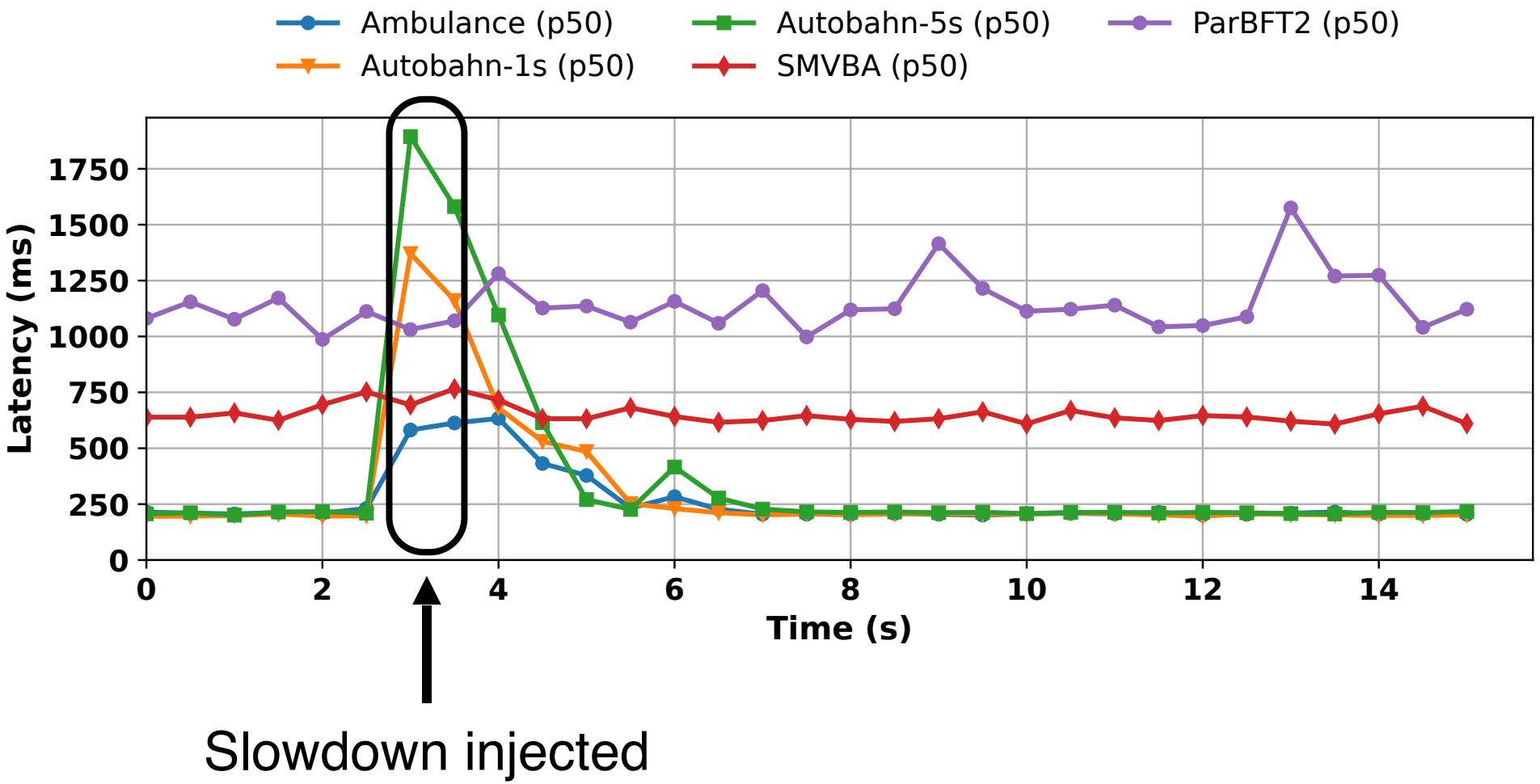
Evaluation: What to expect



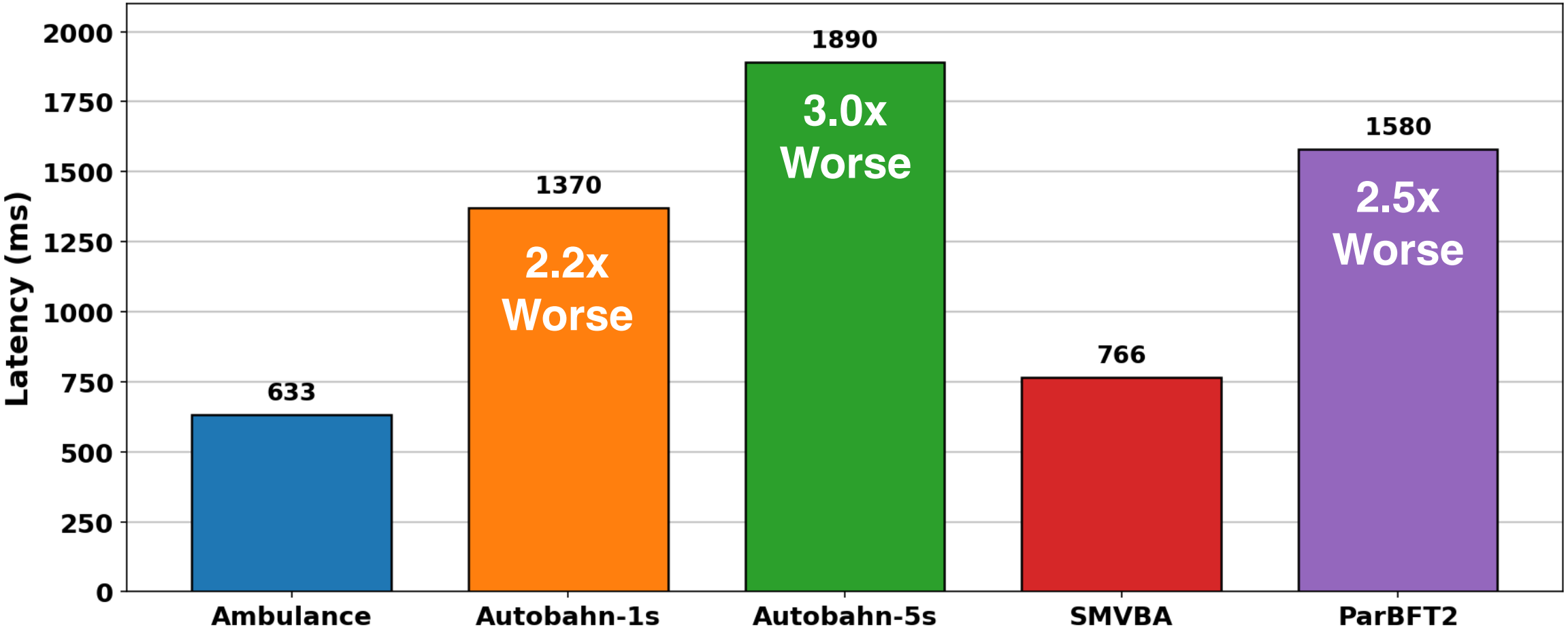
Evaluation: Ideal Case



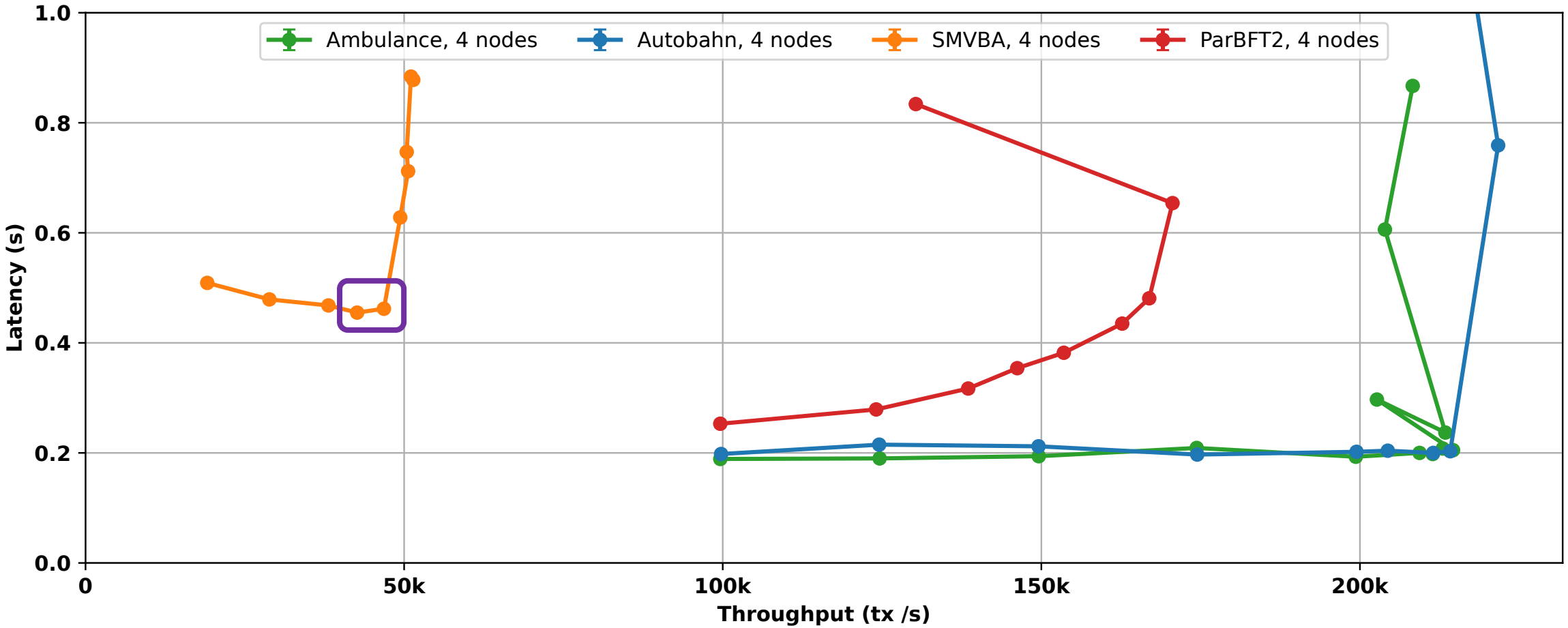
Evaluation: Slowdown Case



Evaluation: Slowdown Case



Evaluation: Slowdown Case



Evaluation: Production Results

	p50	p99
Autobahn	0.244 s	0.660 s
Ambulance	0.242 s	1.28 s

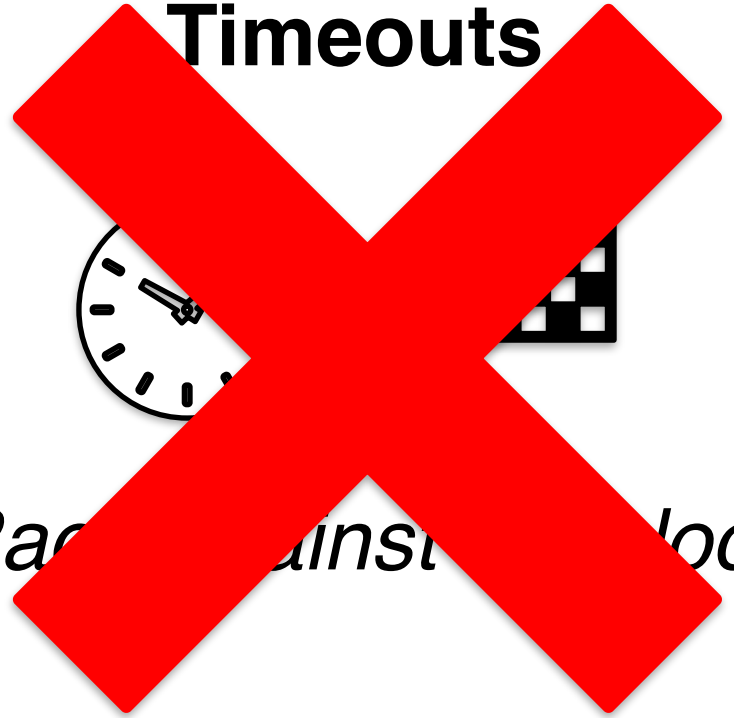
1.94x lower



Conclusion

Stop using time for slowdown detection!

Timeouts



Rac[illegible]ainst [illegible]lock

Hedging



R[illegible]gainst [illegible]other

But leader gets a time-based head start!

Conclusion

Rig the race using the protocol structure itself!



Questions?

Q&A or e-mail at

giridhn@berkeley.edu



