A Low-latency Consensus Algorithm for Geographically Replicated Sites

Master of Science Thesis Defense

Balaji Arun

Committee:
Dr. Binoy Ravindran (chair)
Dr. Haibo Zeng
Dr. Robert Broadwater

Agenda

- Introduction
- Motivation
- Thesis Contribution: CAESAR
- Evaluation
- Conclusion

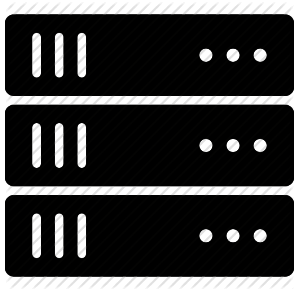
Building online services today…

Desired Properties

- Availability – Fault Tolerance
- Low-latency
- High-throughput
- Strong Consistency

Desired Properties

- Availability under faults ⟶ **Replication**
- Low-latency
- High-throughput
- Strong Consistency

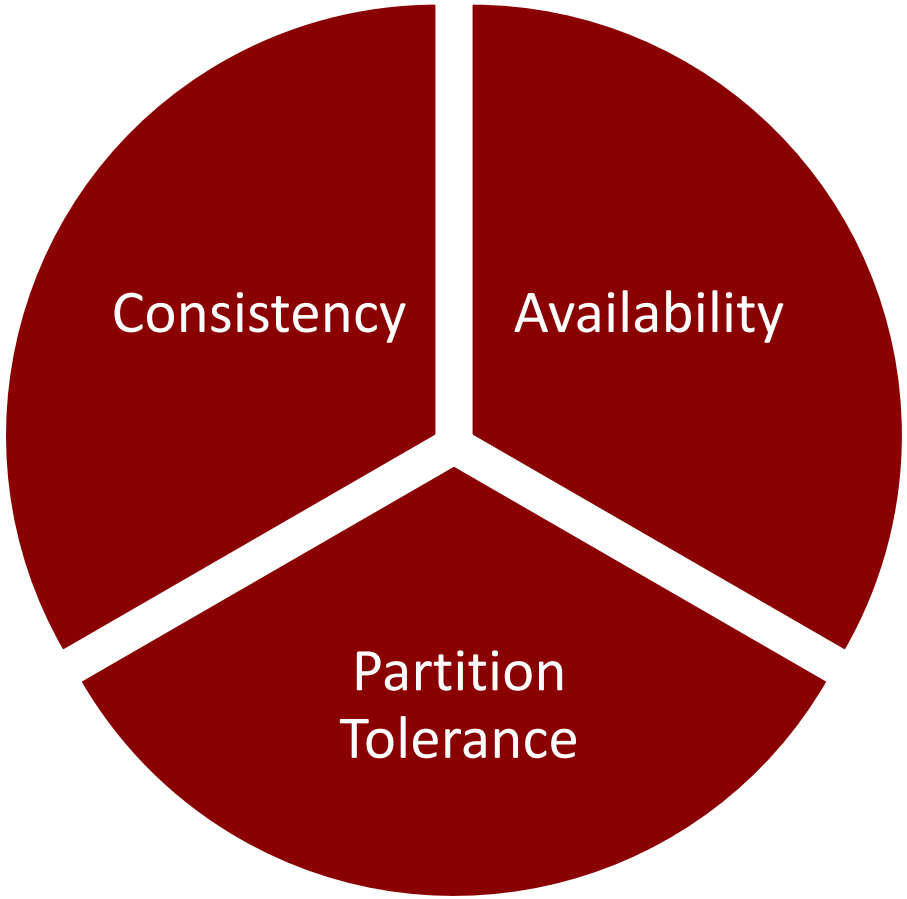


Distributed System

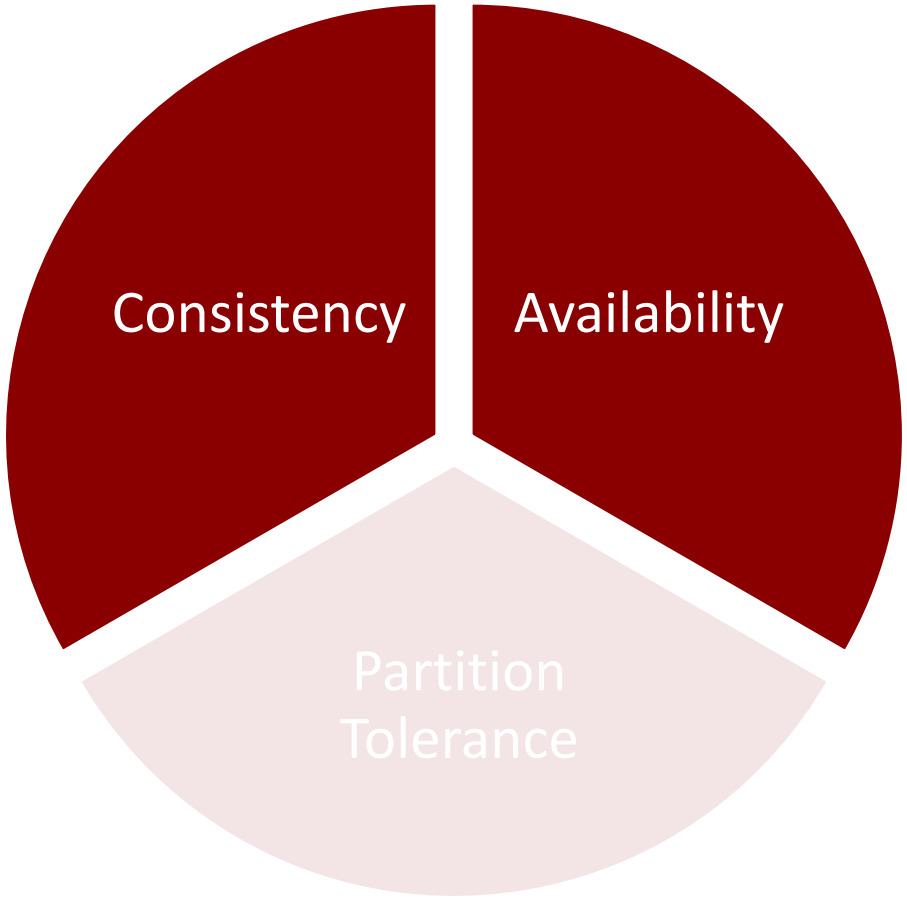
Replicating Stateful Systems

- Stateful systems
  - e.g. databases, in-memory caches
- Requires heavy coordination among nodes
  - to maintain consistency
- Expensive in wide area
  - due to high latency links

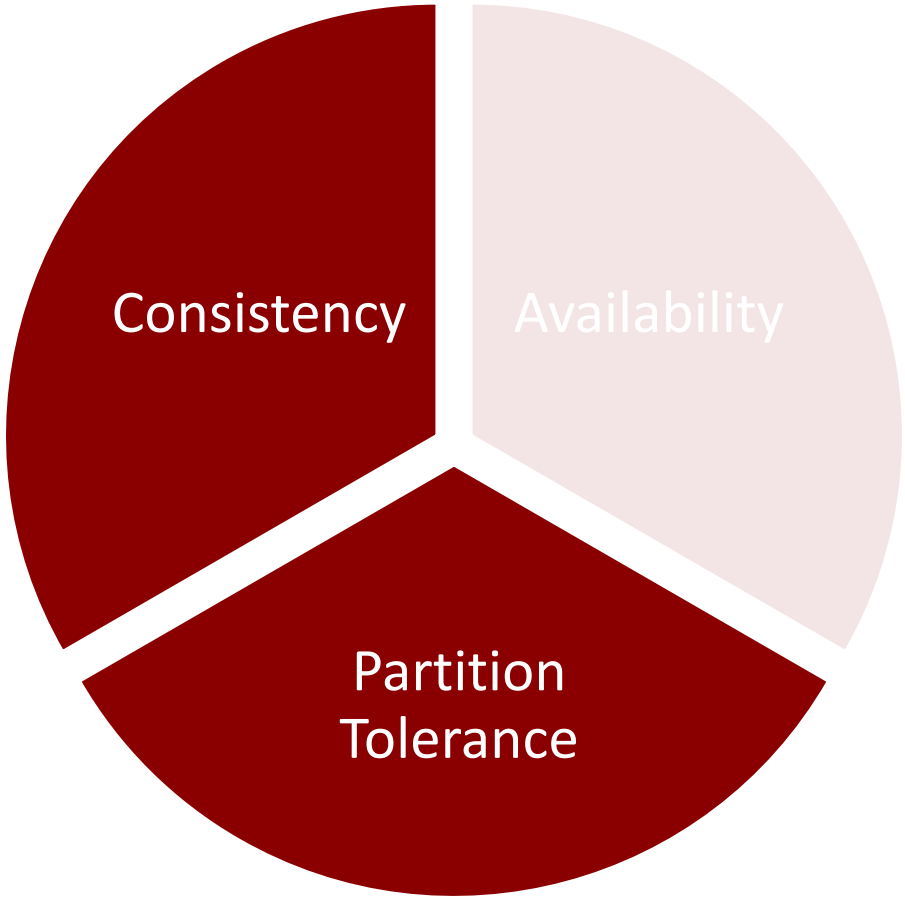
CAP Theorem [Brewer 1997]



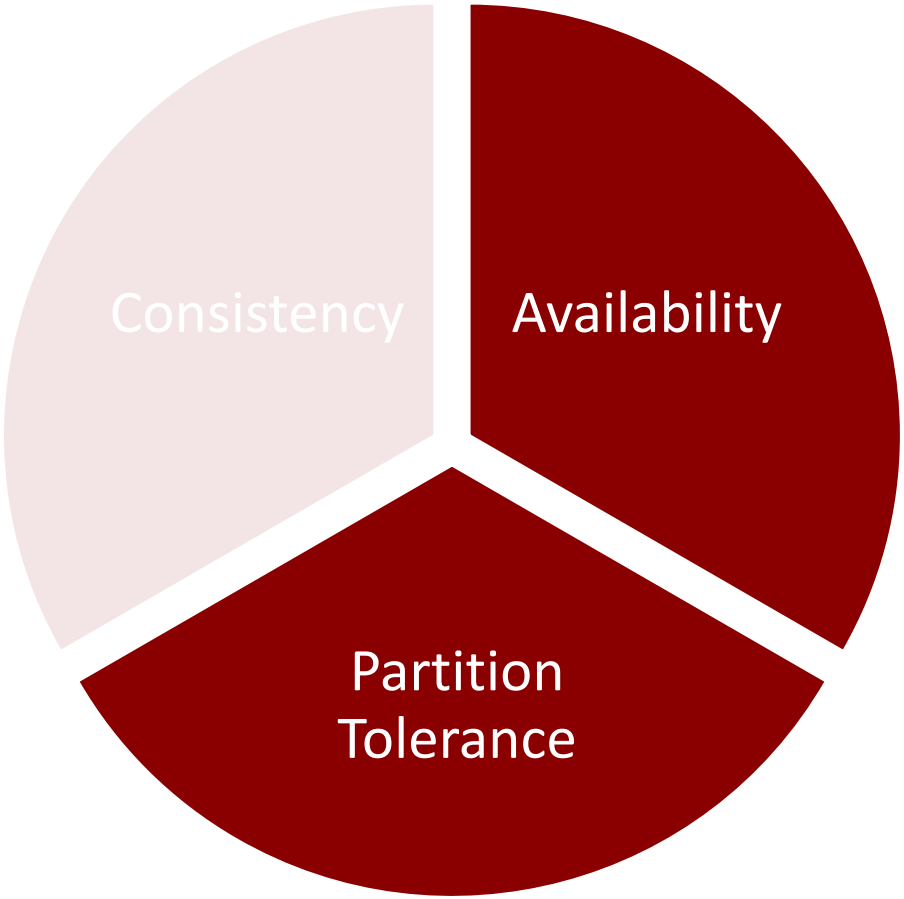
CAP Theorem [Brewer 1997]



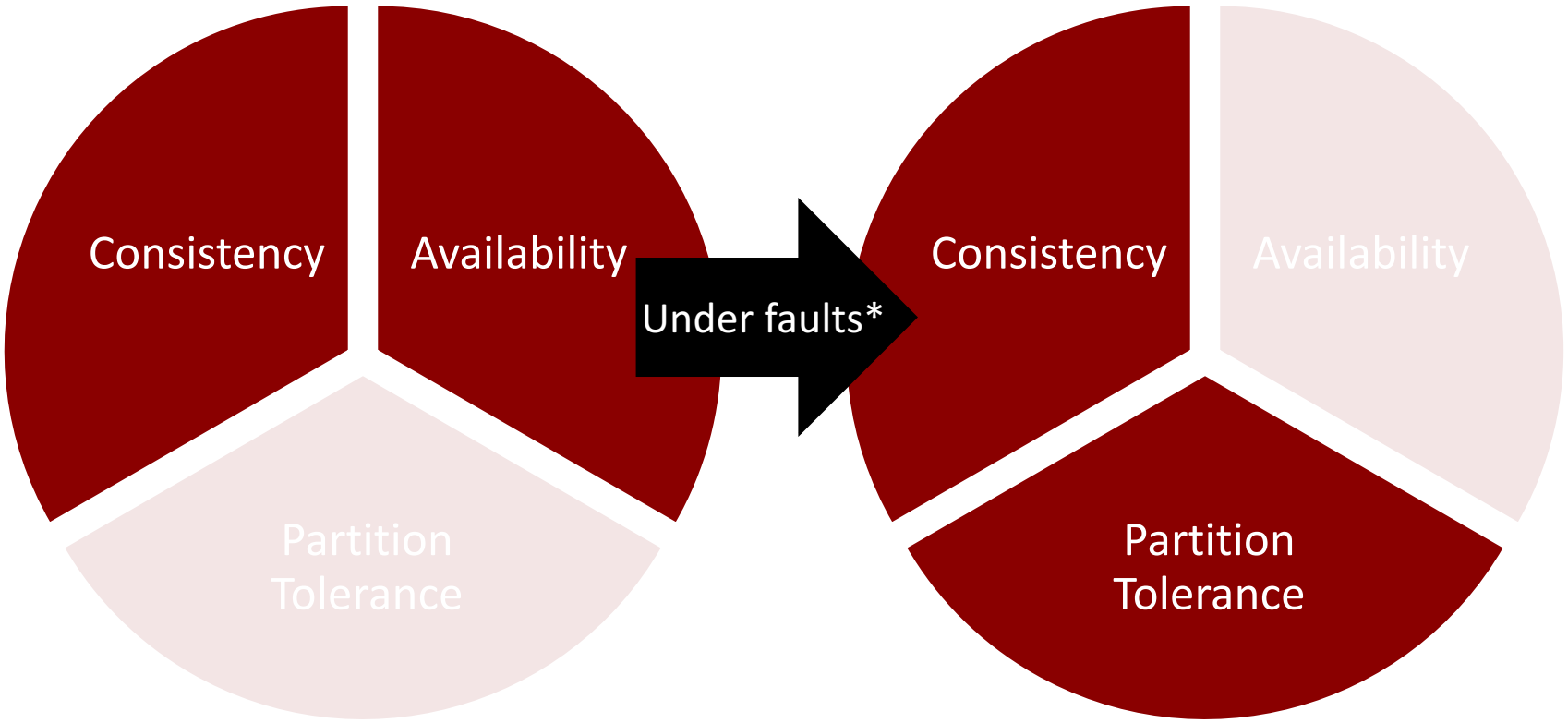
CAP Theorem [Brewer 1997]



CAP Theorem [Brewer 1997]



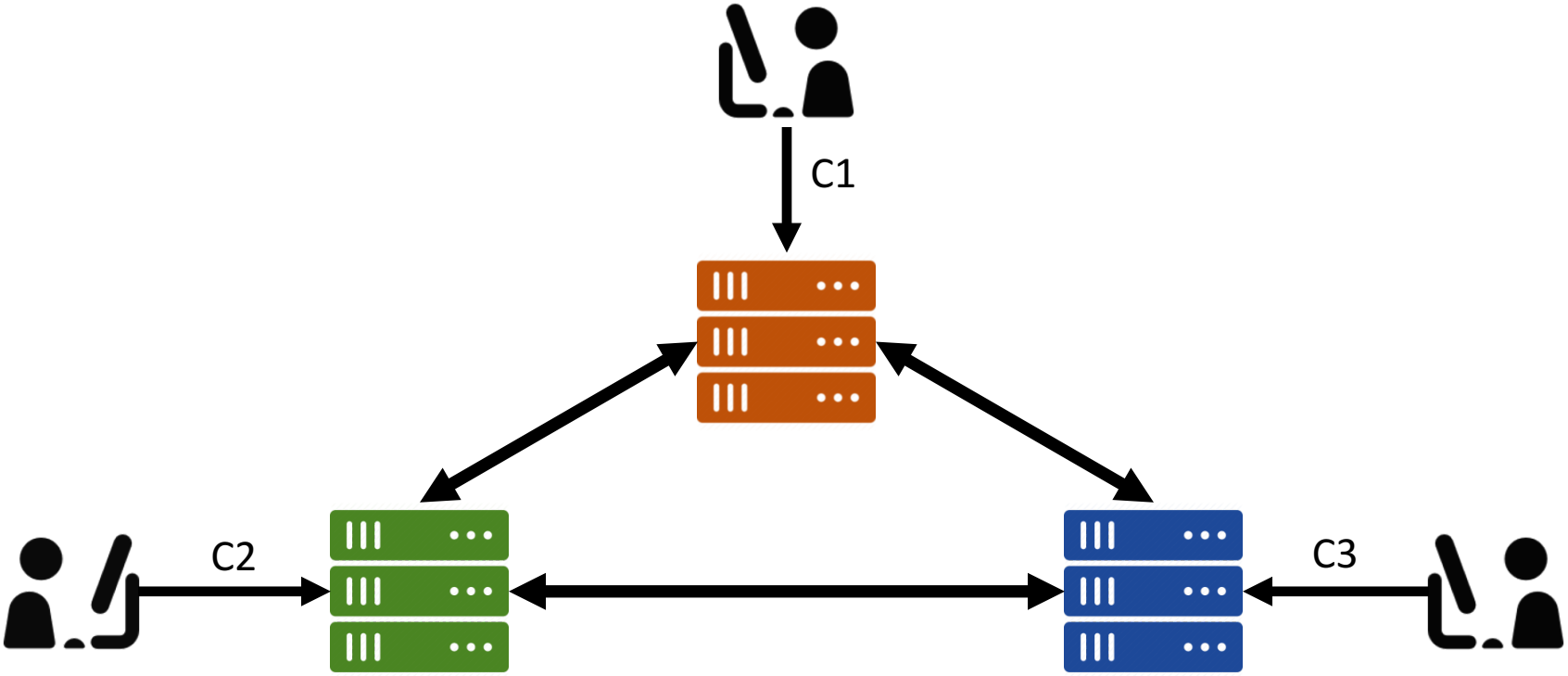
Thesis Contribution



*only when majority of nodes in the system fail

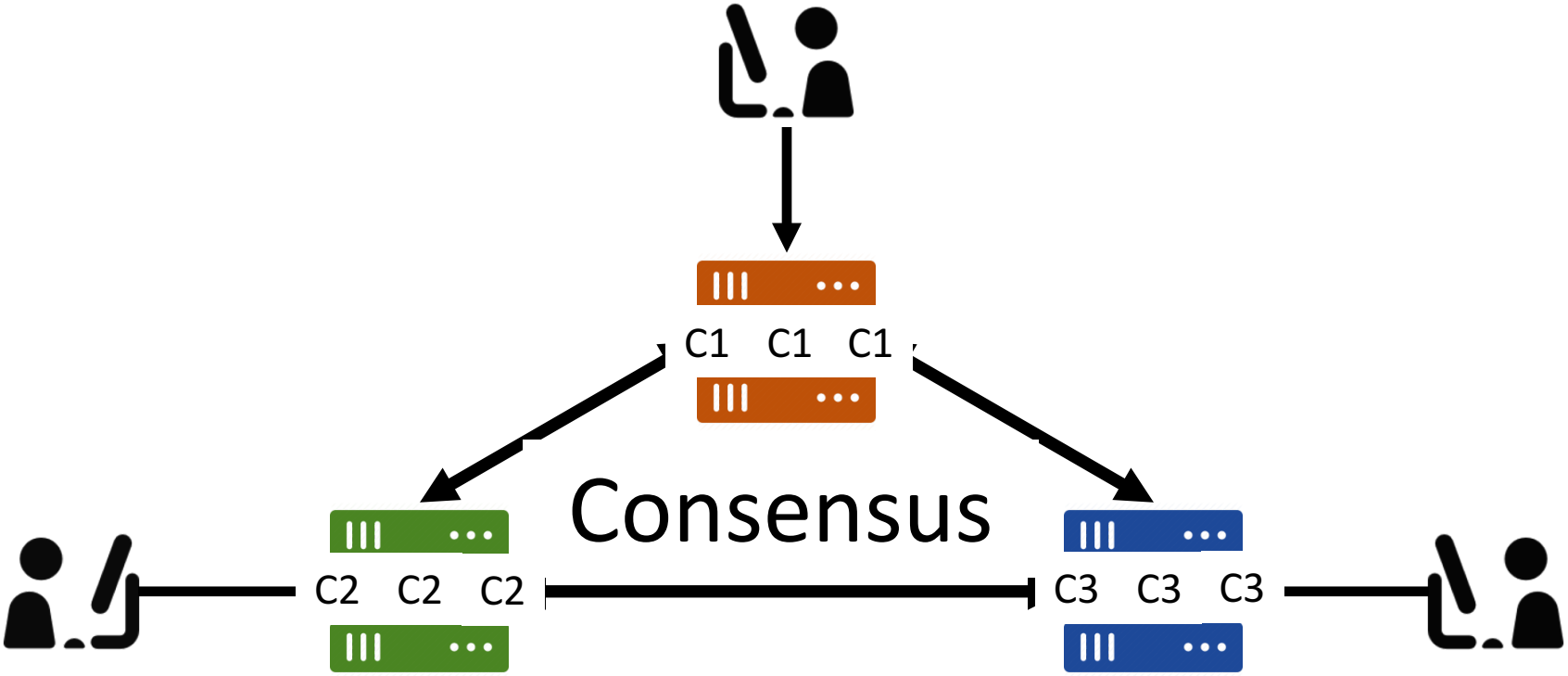
State Machine Replication [F.B. Schneider 1990]

- Execute commands in the same order in all replicas.



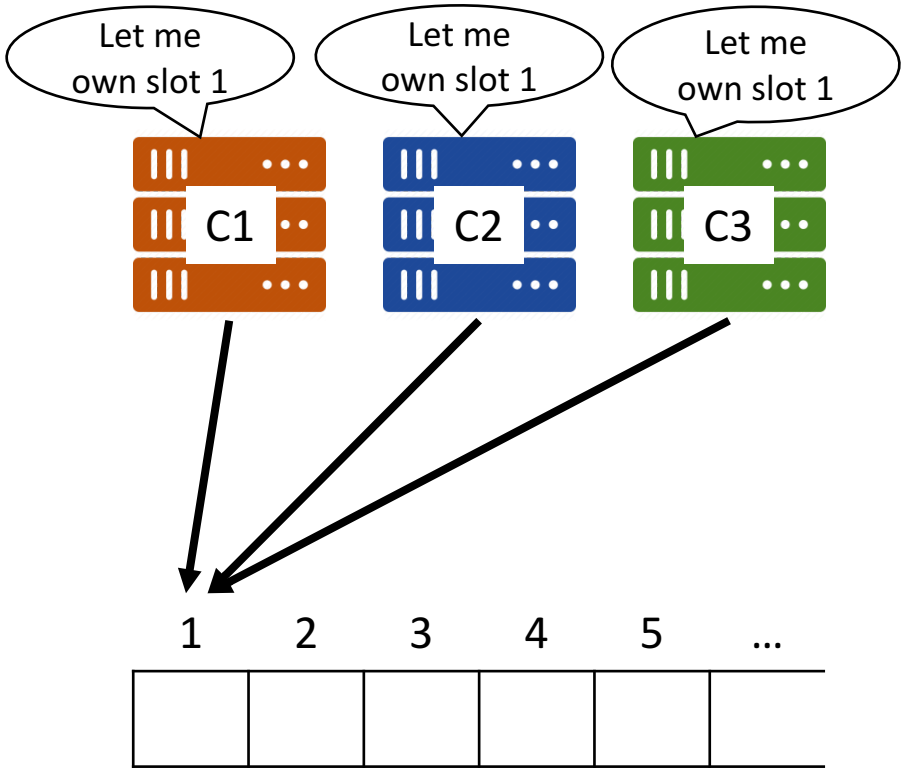
State Machine Replication

- Execute commands in the same order in all replicas



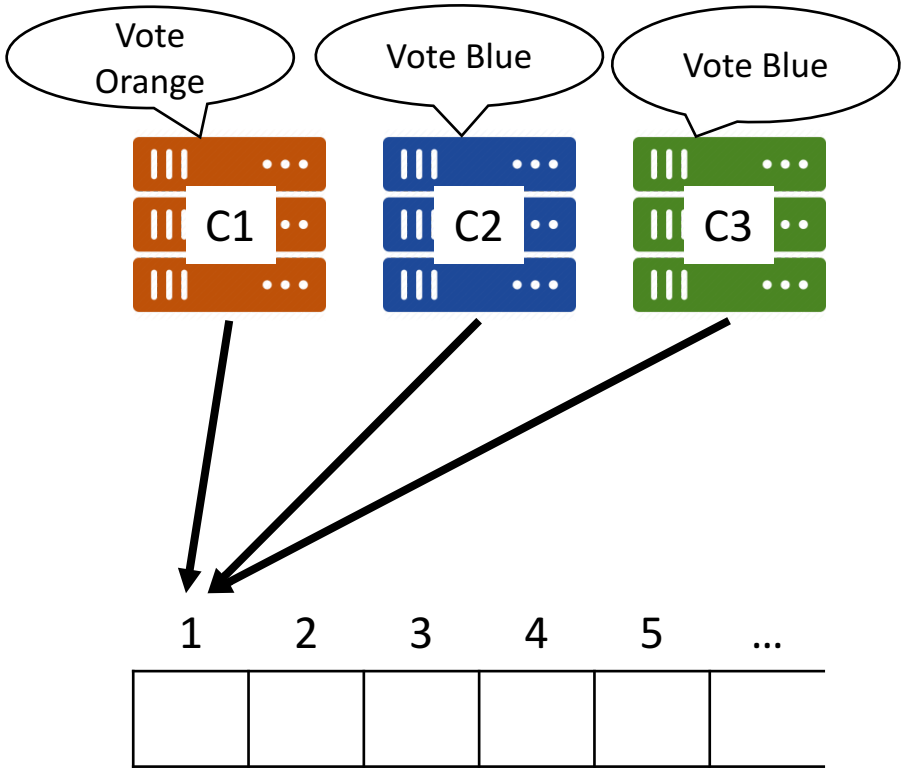
Paxos [Lamport 2001]

- Agreement protocol
- Choose command per slot
- 2 RTTs
  - One for slot ownership
  - One for deciding command
- Survives F crashes in 2F+1 nodes



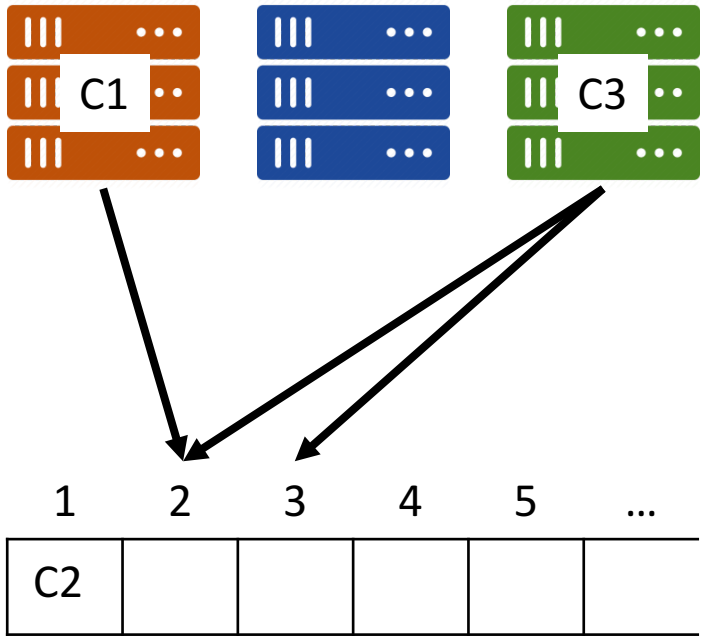
Paxos

- Agreement protocol
- Choose command per slot
- 2 RTTs
  - One for slot ownership
  - One for deciding command
- Survives F crashes in 2F+1 nodes



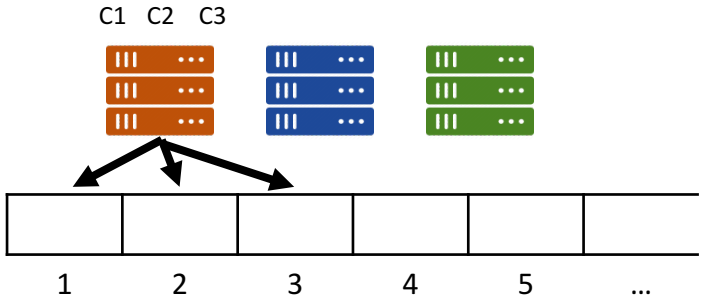
Paxos

- Agreement protocol
- Choose command per slot
- 2 RTTs
  - One for slot ownership
  - One for deciding command
- Survives F crashes in 2F+1 nodes

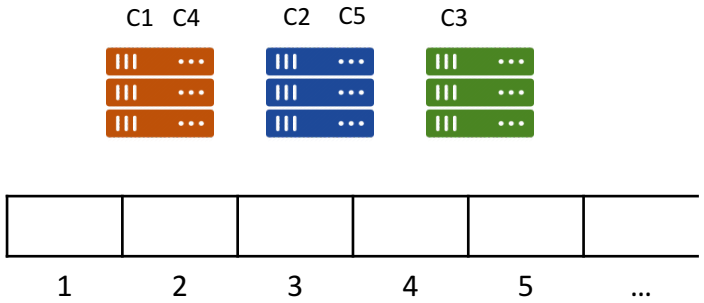


Paxos variants

- One replica decides:
  - Multi-Paxos, Fast Paxos



- Round robin approach:
  - Mencius [Mao '2008]



Generalized Consensus [Lamport 2005]

- Order only non-commutative commands
  - e.g. same key in Key-Value store.
- EPaxos [Moraru '13], M^2Paxos [Peluso '16]
  - Best performance under no conflicts (Fast Decision, 1RTT)
  - Performance degrades under conflicts (Slow Decision, 1+RTT)
- Thesis contribution: CAESAR

✓Introduction

✓Motivation

CAESAR

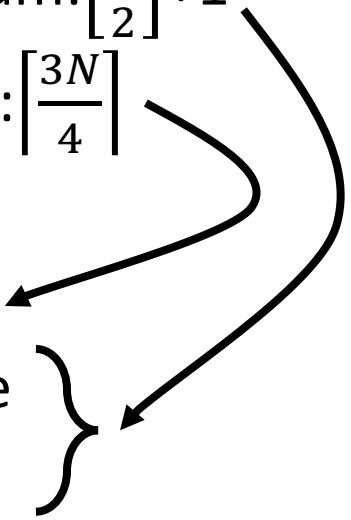
- Evaluation
- Conclusion

Overview

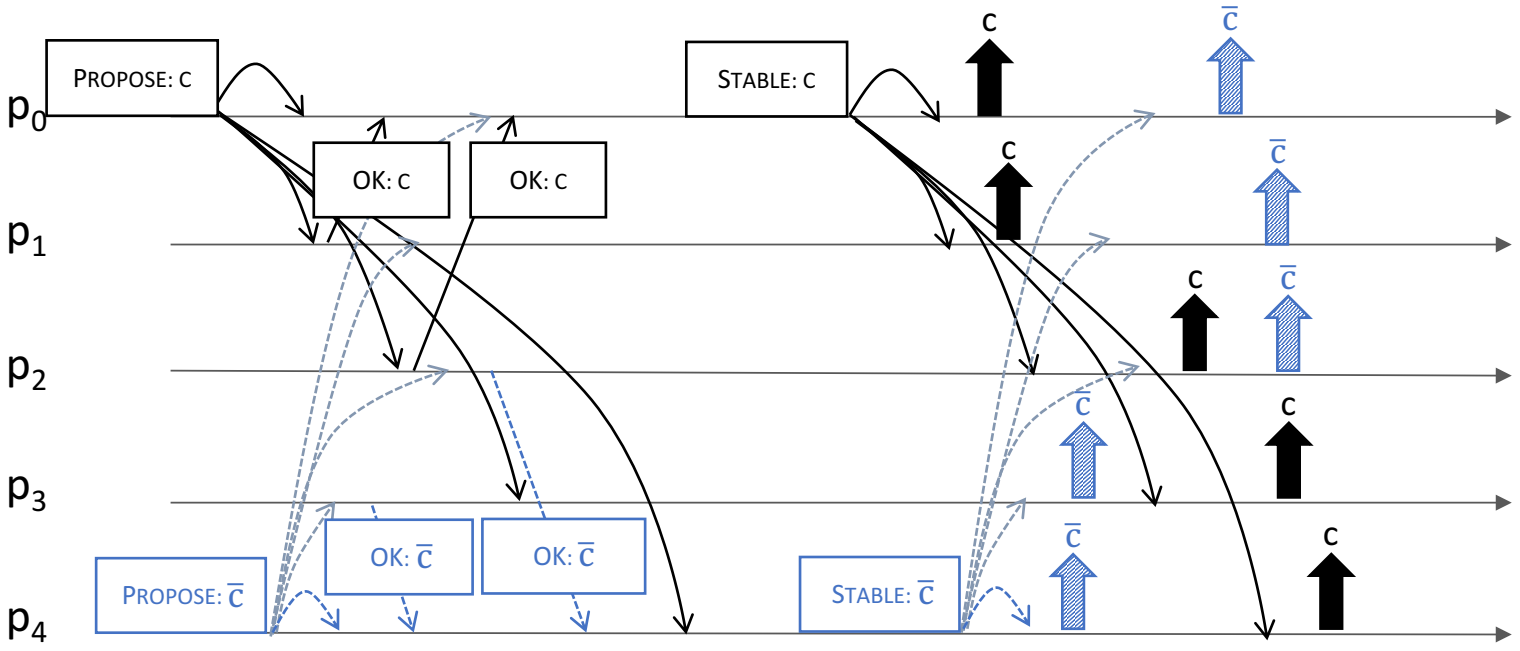
- Implements Generalized Consensus
- Uses logical timestamping to order commands (like Mencius)
- Exploits quorums by gathering dependencies for commands (like EPaxos)
- Deploys a novel *wait* condition to boost fast decisions
  - under conflicts

System model

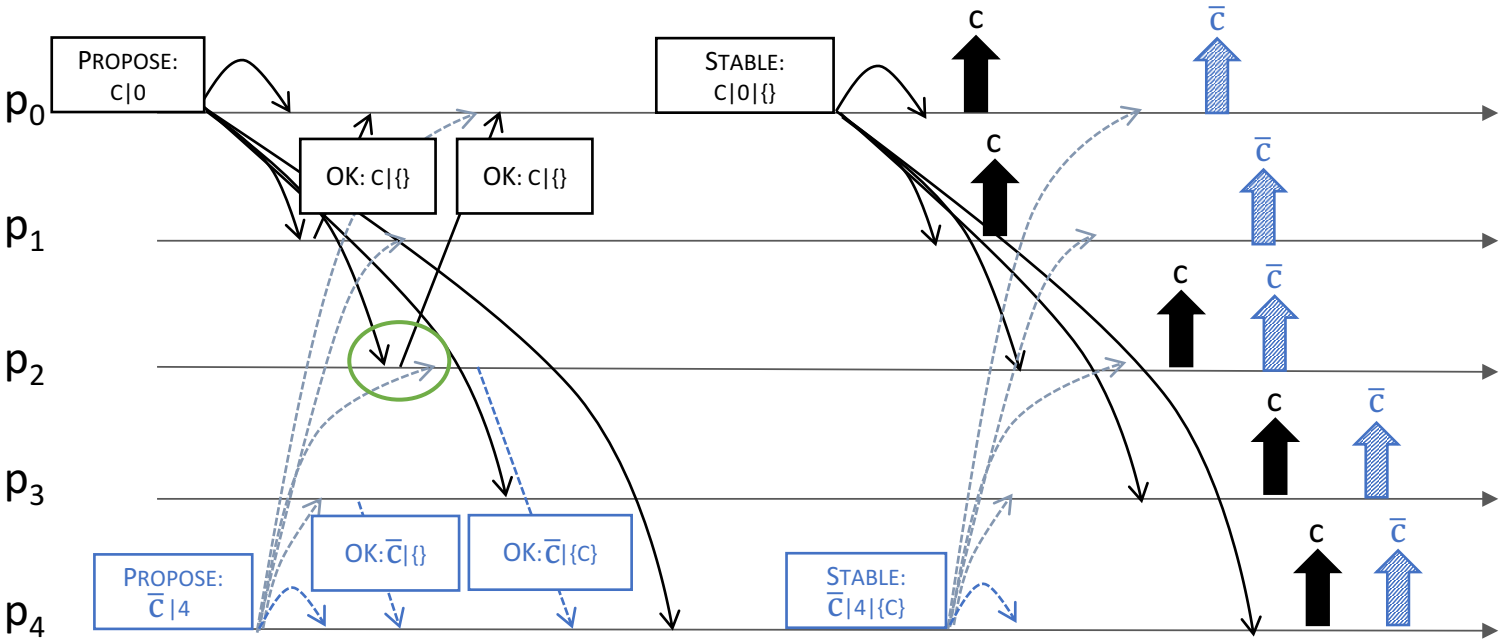
- Nodes communicate through message passing
- Uses two quorum types
  - Classic Quorum: $\left\lfloor \frac{N}{2} \right\rfloor + 1$
  - Fast Quorum: $\left\lceil \frac{3N}{4} \right\rceil$
- Four phases
  - Fast Propose
  - Slow Propose
  - Retry
  - Stable



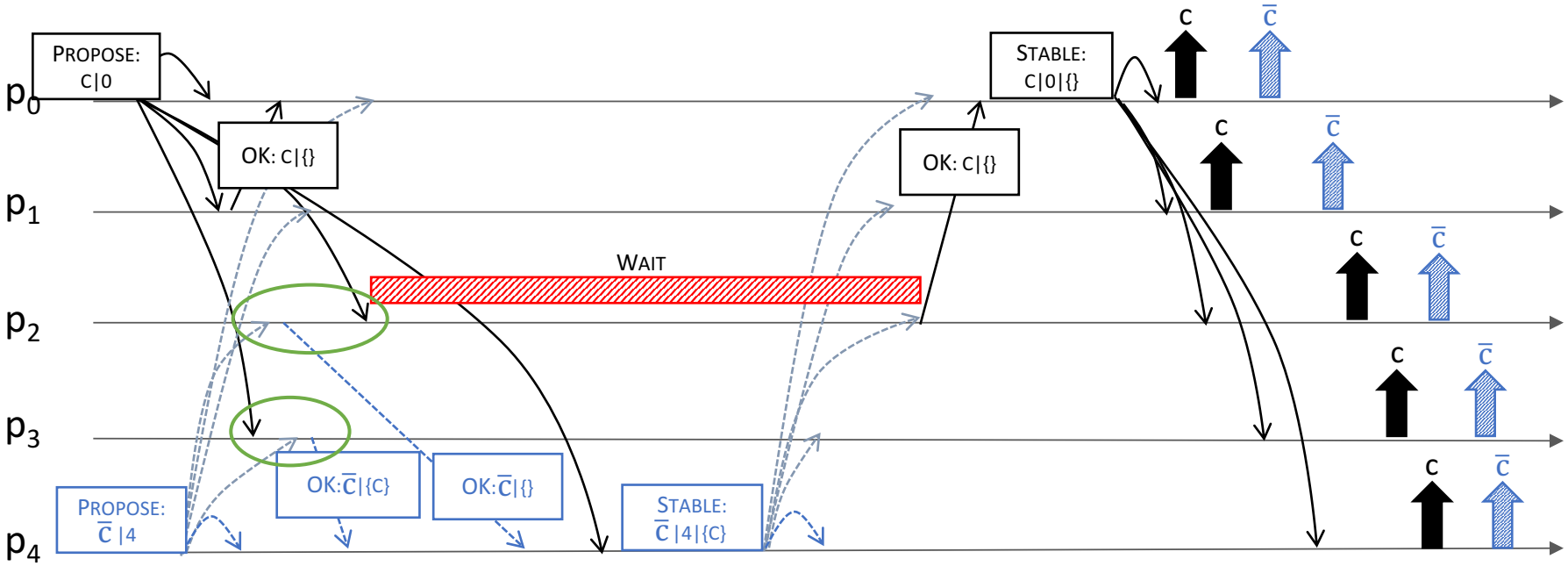
Uniform Reliable Broadcast



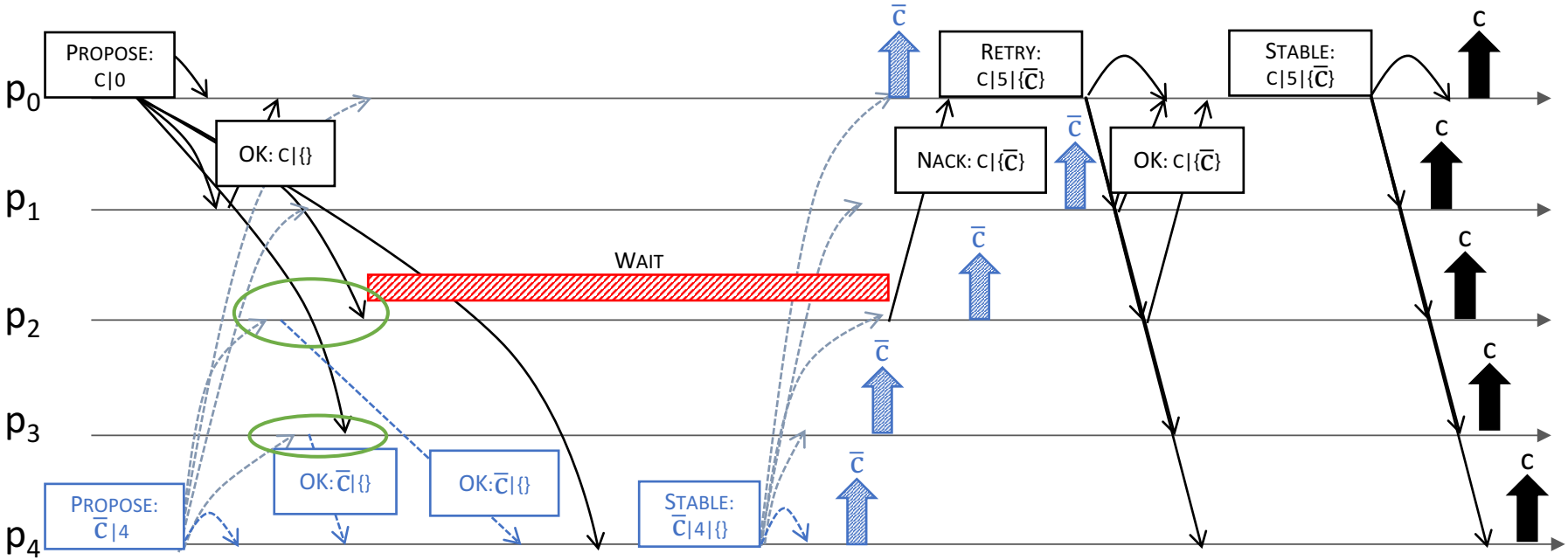
CAESAR: Basic Protocol



CAESAR: Wait Condition (*fast path*)



CAESAR: Wait Condition (*slow path*)



✓Introduction

✓Motivation

✓The Contribution: Caesar

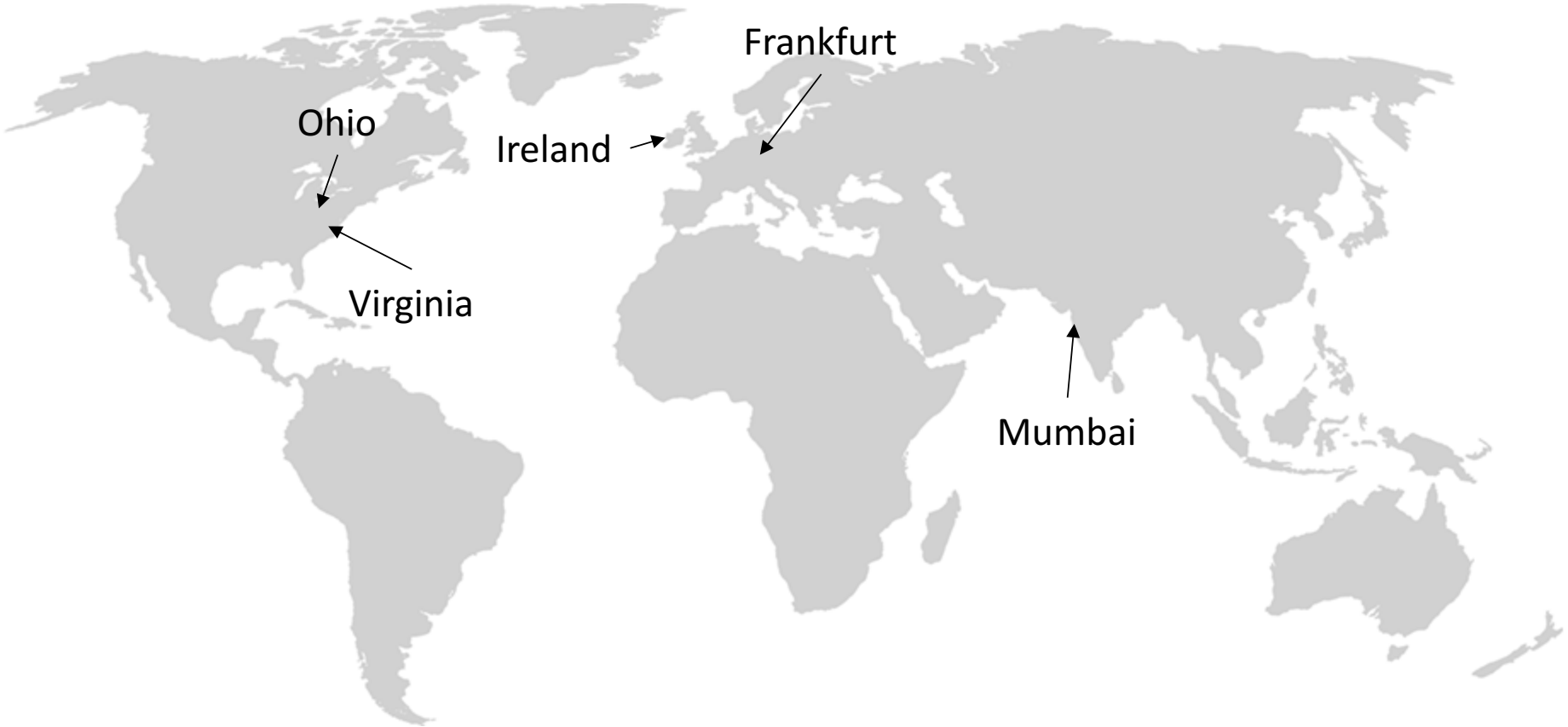
Evaluation

- Conclusion

Implementation

- CAESAR
  - Implemented in Java 8
  - Adopted and modified JPaxos
    - used network, messaging and state machine abstractions
- Competitors
  - EPaxos, M^2Paxos, Mencius, Multi-Paxos publicly available
- Benchmark
  - Fully replicated Key-Value store benchmark

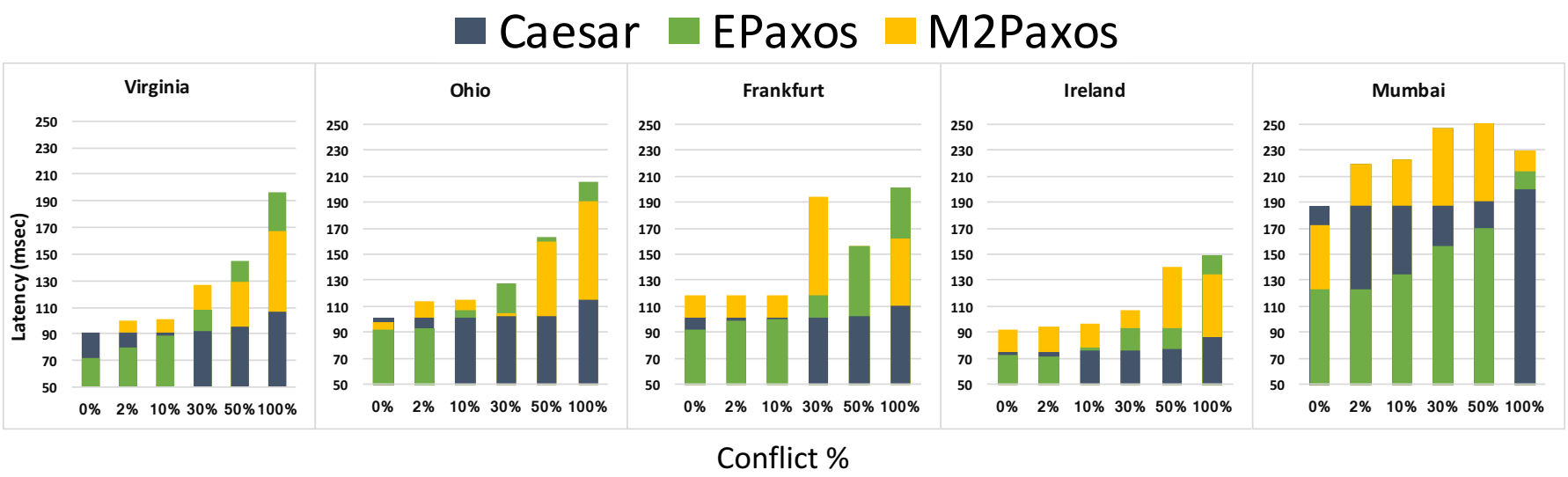
Deployment



- Amazon EC2
  - m4.2xlarge instances (8 vCPUs, 32GB Memory)

Client-perceived performance

Latency



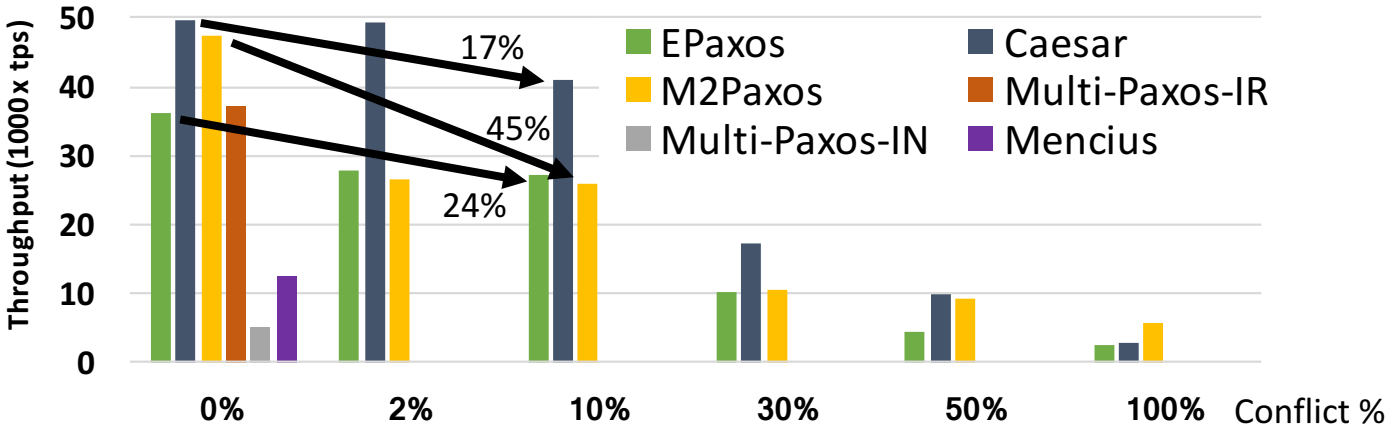
- Closed-loop request injection
- 10 clients per node

System performance

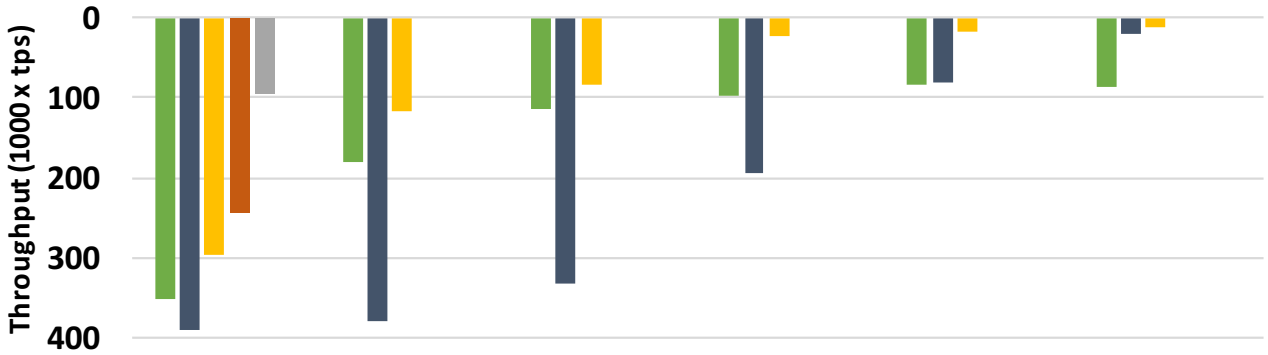
Throughput

- Open loop request injection

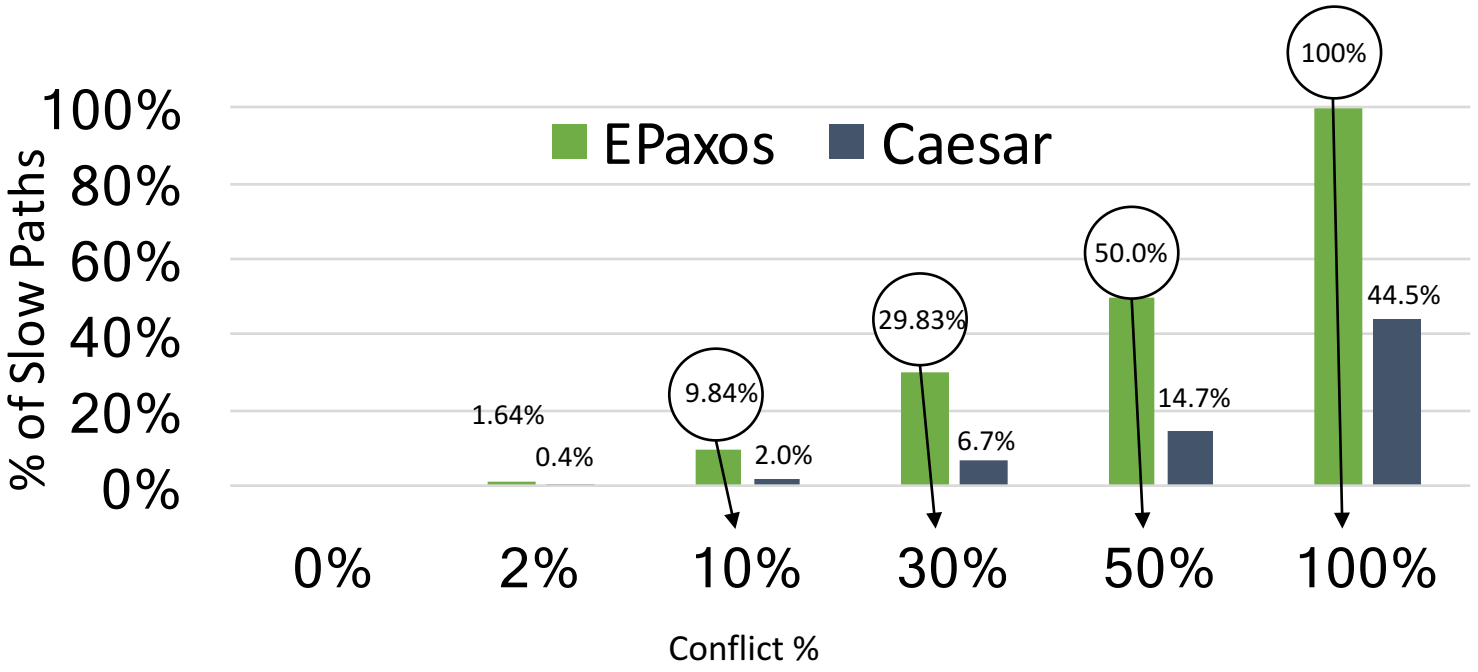
Network Batching Disabled



Network Batching Enabled



Slow Paths vs EPaxos



✓Introduction

✓Motivation

✓The Contribution: Caesar

✓Evaluation

Conclusion

Conclusion

- CAESAR provides all the desired properties for building today's services.

High Availability under faults Strong consistency	→	State Machine Replication and Consensus
Low-latency	→	Fast Paths
High-throughput	→	Generalized consensus and simple execution phase

Future Work

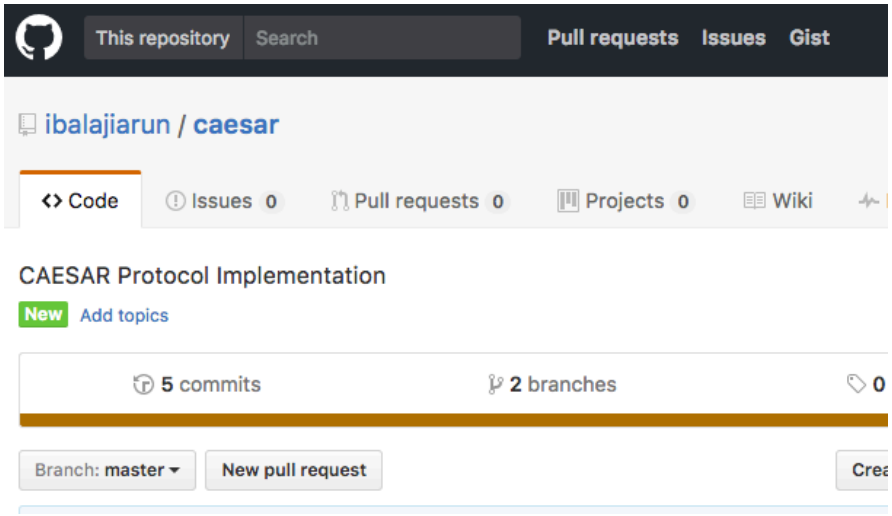
- CAESAR can exhibit a latency distribution with a large standard deviation
  - due to multiple phases and wait condition
- This is a challenge for meeting Service Level Agreement (SLAs)
- Future work should minimize large tail latencies

Submission

Submitted to DSN 2017

Source code

- Open source @ https://www.github.com/ibalajiarun/caesar



References

- Lamport, Leslie, “Paxos made simple,” ACM Sigact News, 2001.
- I. Moraru, D. G. Andersen, and M. Kaminsky, “There is More Consensus in Egalitarian Parliaments,” in Proceedings of the Twenty-Fourth ACM Symposium on Operating Systems Principles, ser. SOSP ’13. ACM, 2013, pp. 358–372.
- Y. Mao, F. P. Junqueira, and K. Marzullo, “Mencius: Building Efficient Replicated State Machines for WANs,” in Proceedings of the 8th USENIX Conference on Operating Systems Design and Implementation, ser. OSDI’08. USENIX Association, 2008, pp. 369–384.
- L. Lamport, “Generalized Consensus and Paxos,” Microsoft Research, Tech. Rep. MSR- TR-2005-33, March 2005.
- S. Peluso, A. Turcu, R. Palmieri, G. Losa, and B. Ravindran, “Making fast consensus generally faster,” in DSN, 2016.
- F. B. Schneider, “Implementing fault-tolerant services using the state machine approach: A tutorial,” ACM Comput. Surv., vol. 22, no. 4, pp. 299–319, Dec. 1990. [Online]. Available: http://doi.acm.org/10.1145/98163.98167

Thanks to:

My advisor, Dr. Binoy Ravindran,

Roberto, Sebastiano

Dr. Broadwater, Dr. Zeng

and

All my fellow SSRGians (present and past)