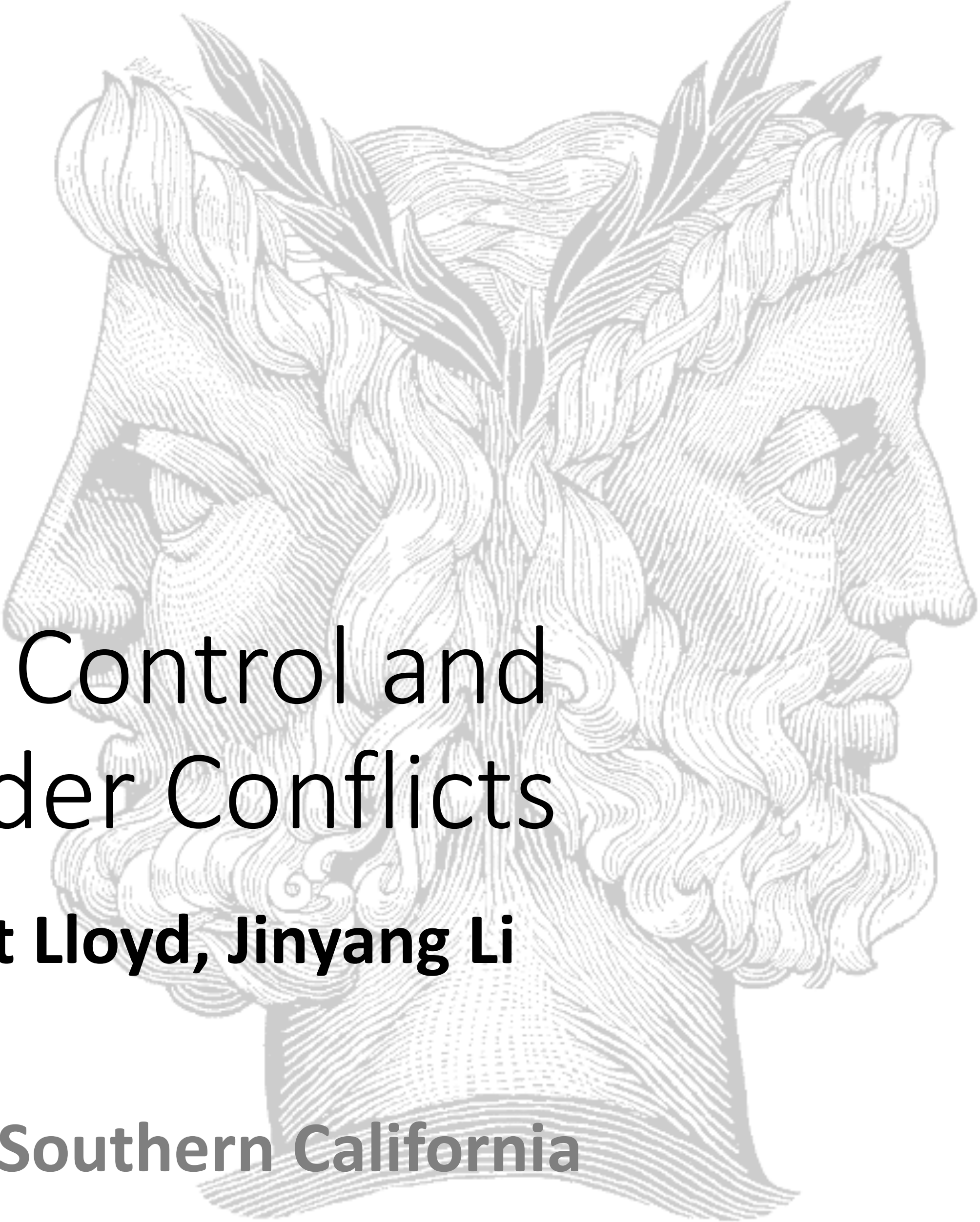
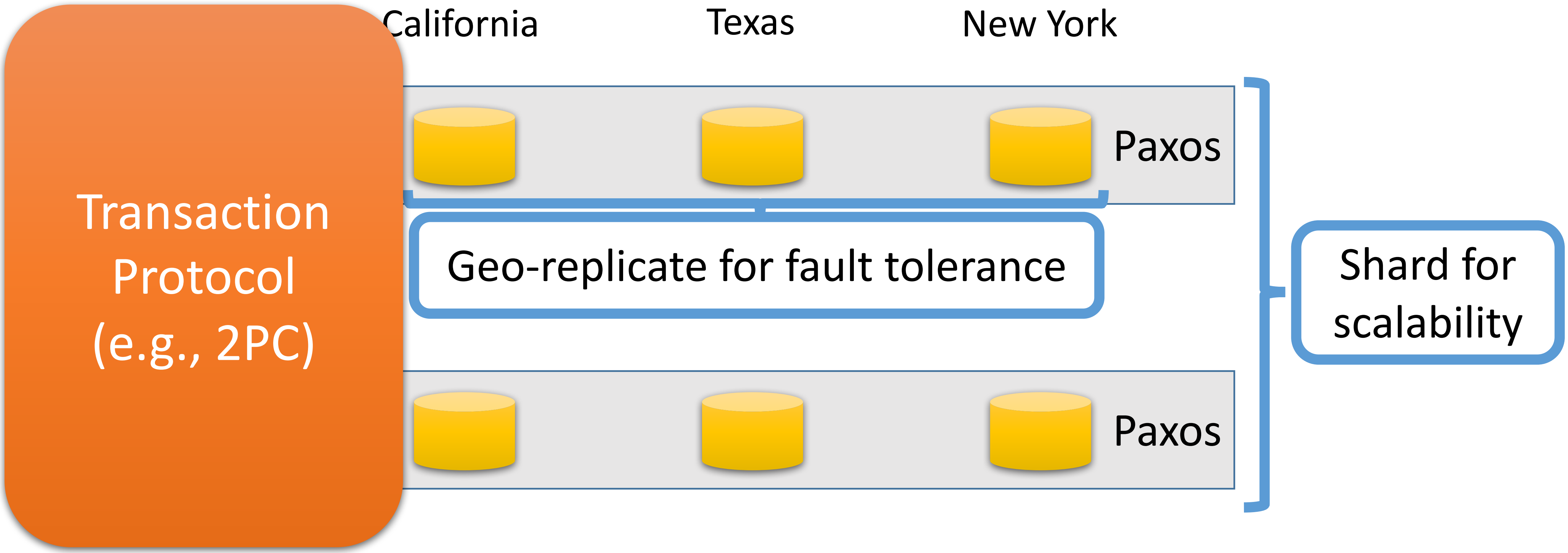
Janus
Consolidating Concurrency Control and Consensus for Commits under Conflicts



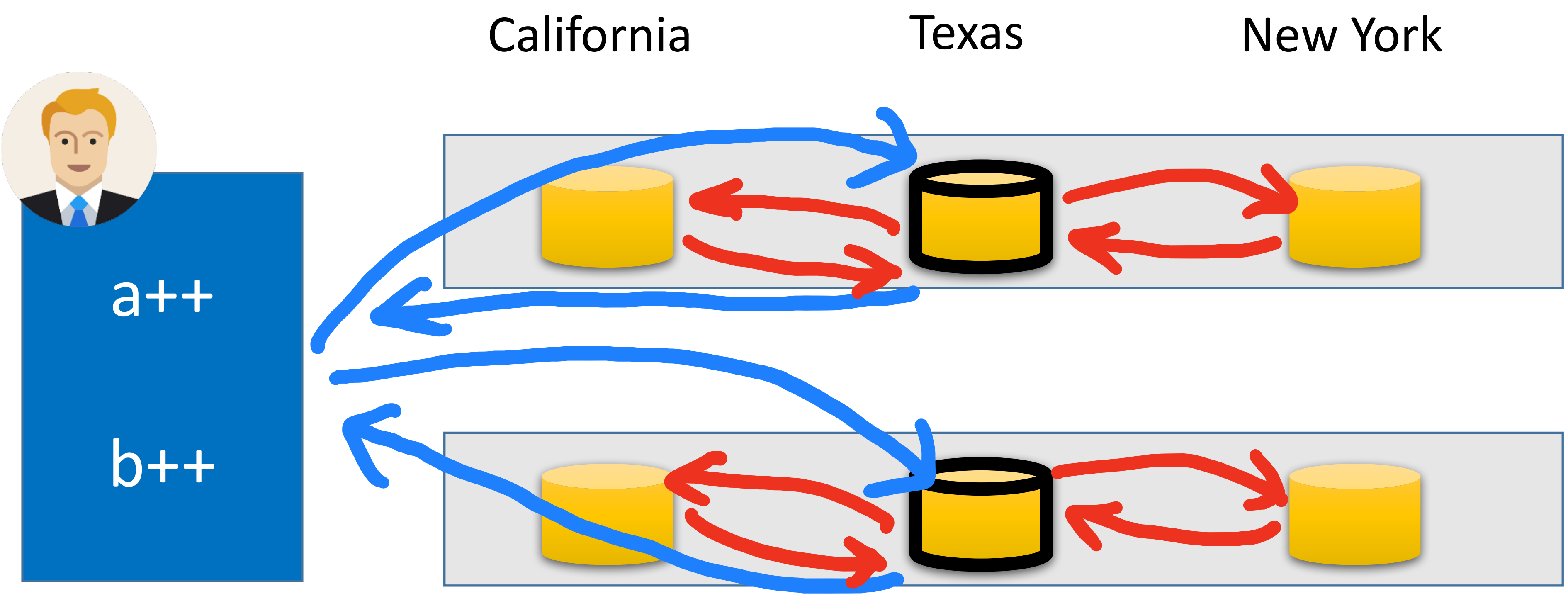
Shuai Mu, Lamont Nelson, Wyatt Lloyd, Jinyang Li

New York University, University of Southern California

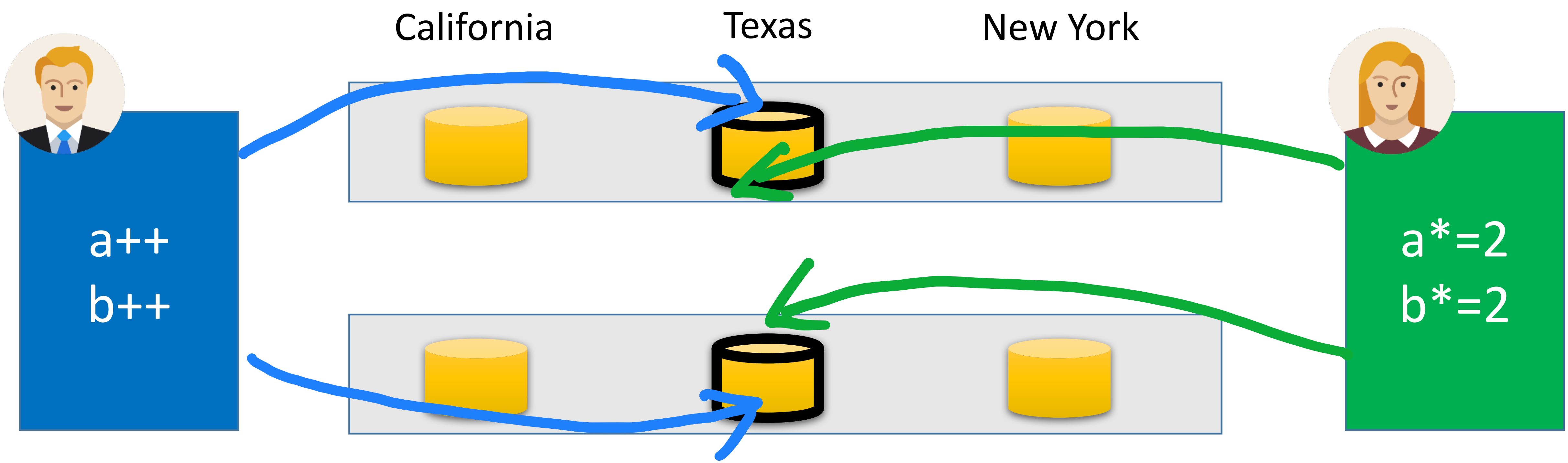
State of the Art for Distributed Transactions
Layer Concurrency Control on top of Consensus



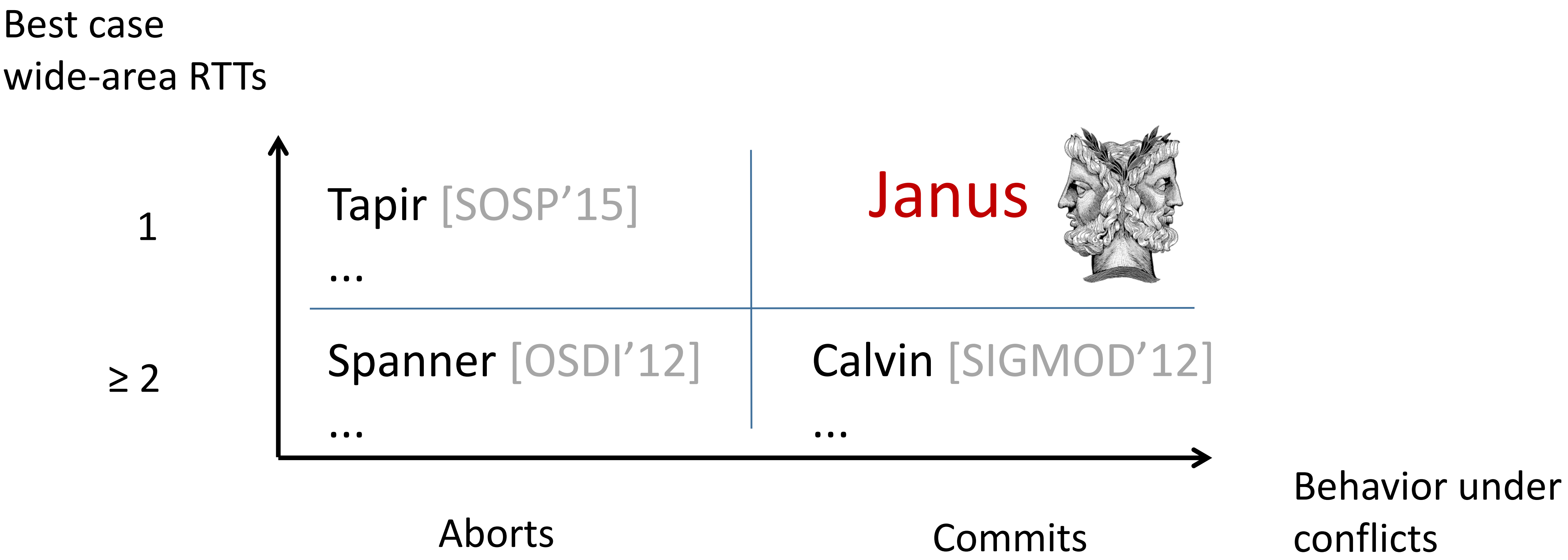
Latency Limitation:
Multiple Wide-Area Round Trips from Layering



Throughput Limitation: Conflicts Cause Aborts

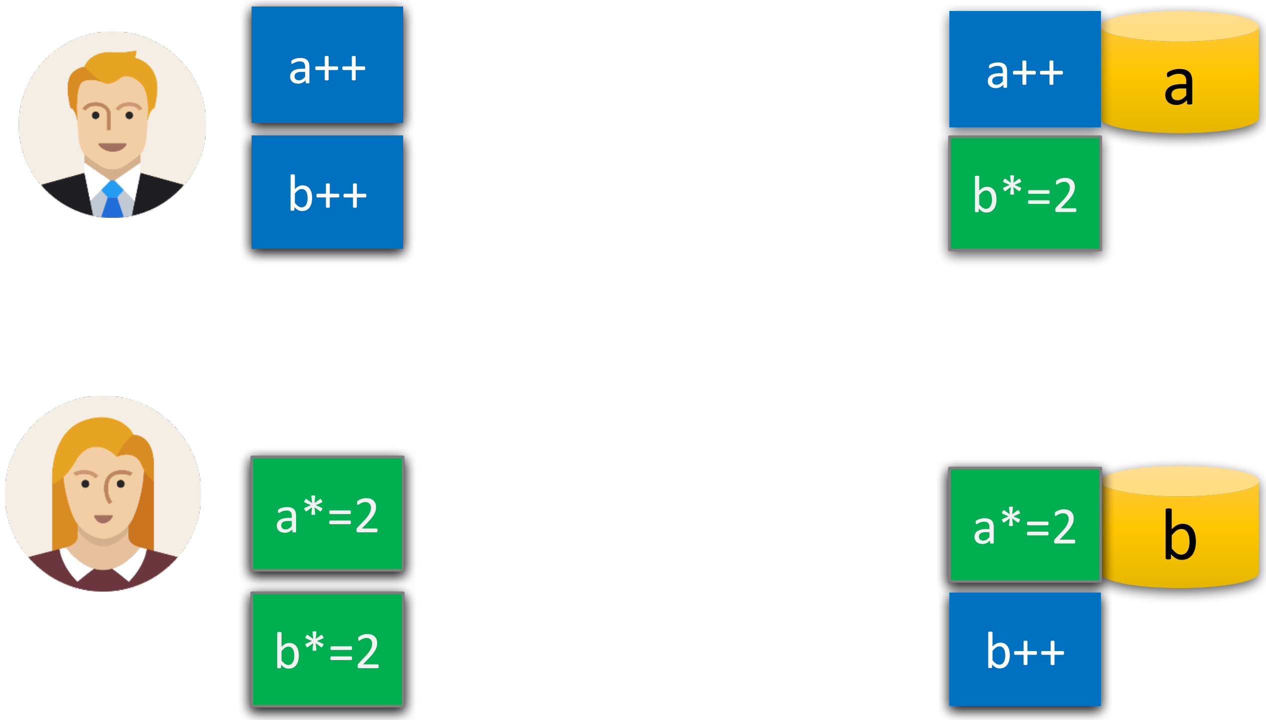


Goals: Fewer Wide-Area Round Trips and Commits Under Conflicts



Establish Order Before Execution to Avoid Aborts

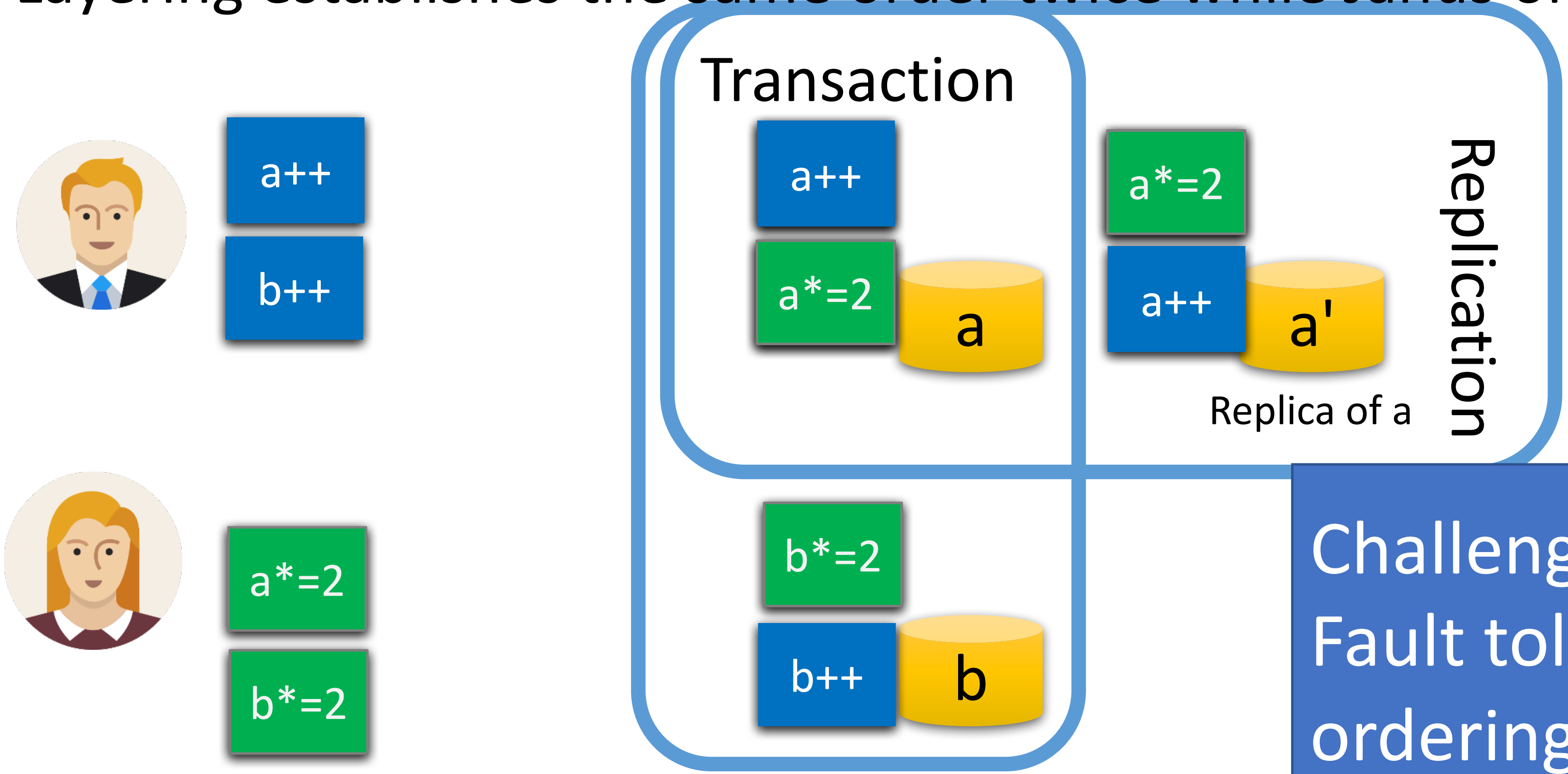
- Designed for transactions with static read & write-sets
- Structure a transaction as a set of stored procedure pieces
- Servers establishes consistent ordering for pieces before execution



Challenge:
Distributed ordering to avoid bottleneck

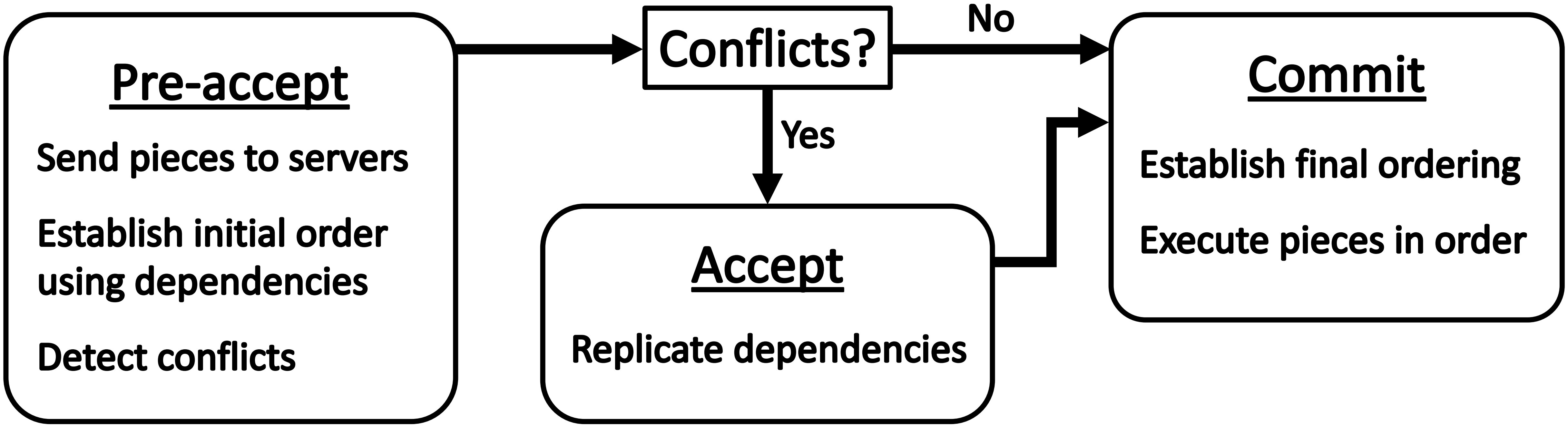
Establish Order for Transactions and Replication Together to Commit in 1 Wide-area Roundtrip

- Consistent ordering for transaction and replication is the same!
- Layering establishes the same order twice while Janus orders once

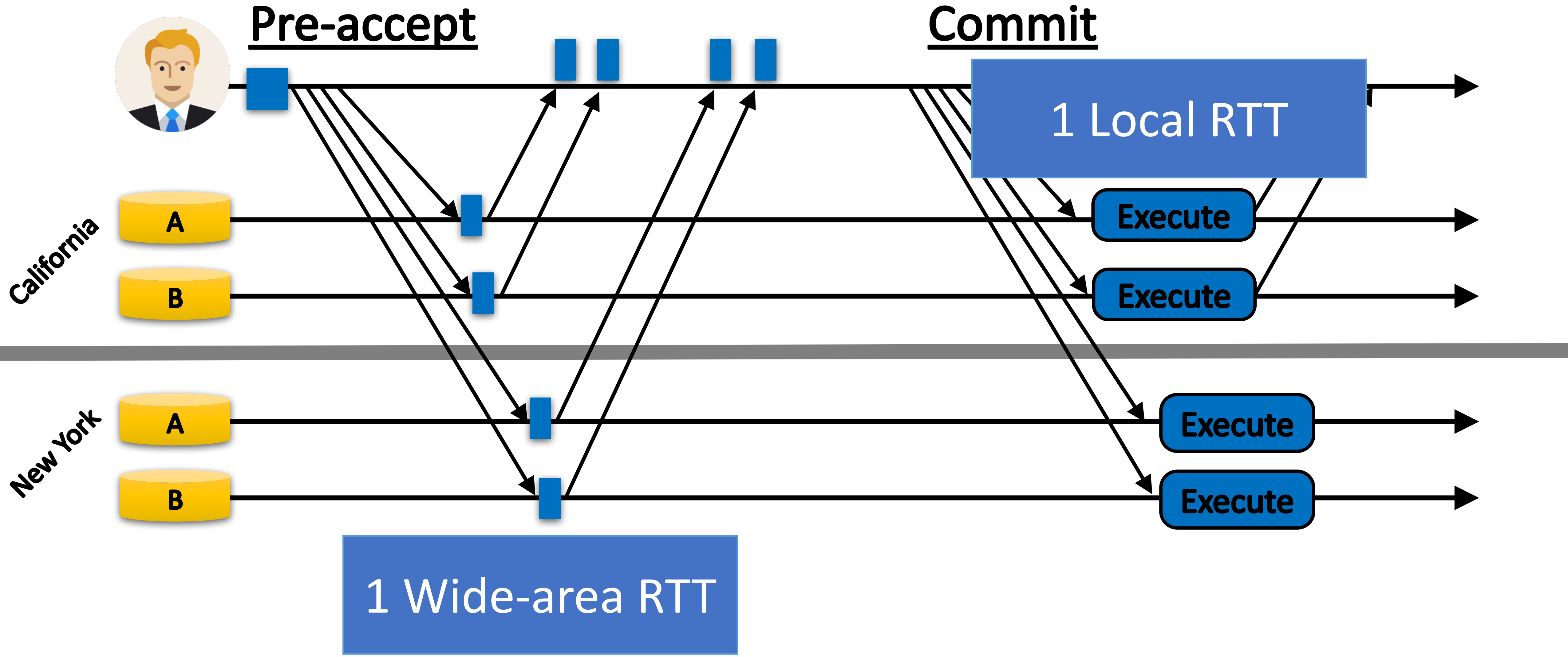


Challenge:
Fault tolerance for ordering

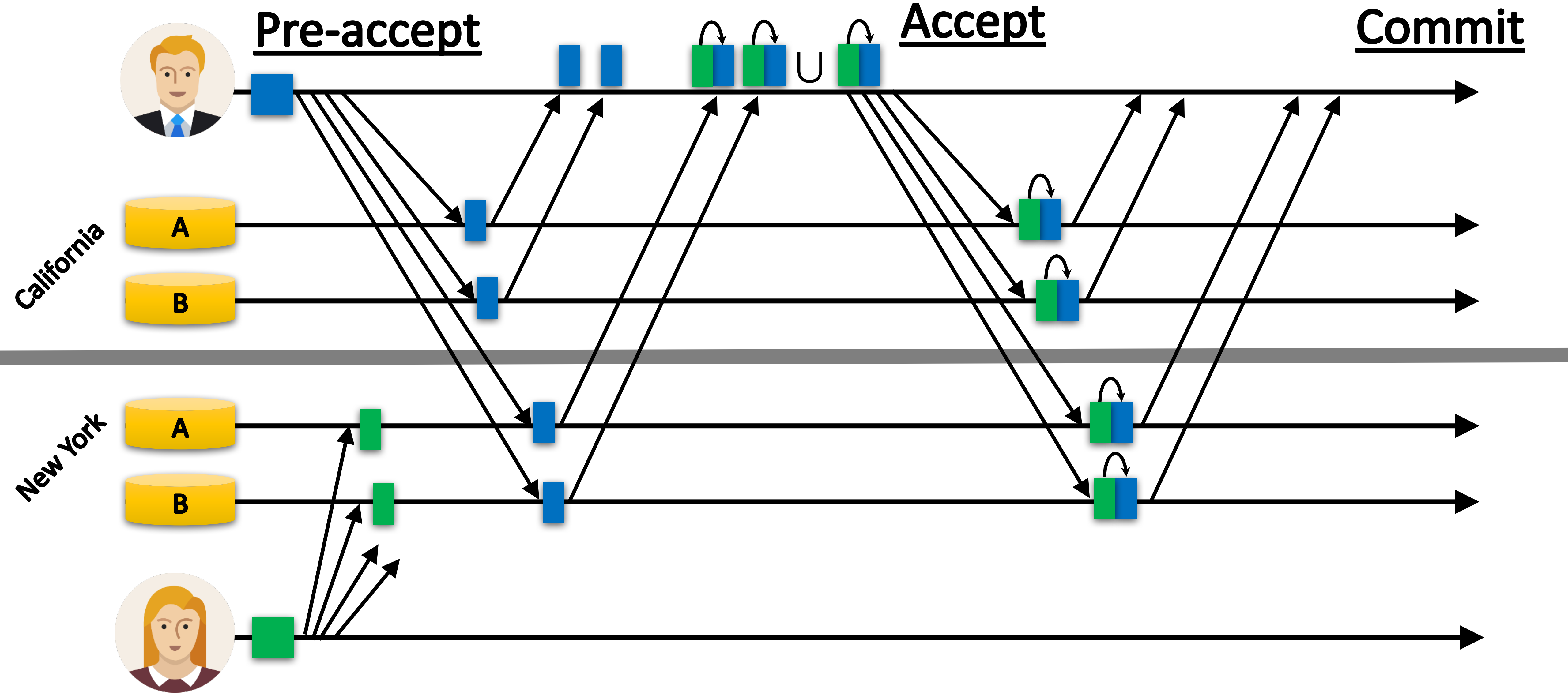
Overview of the Janus Protocol



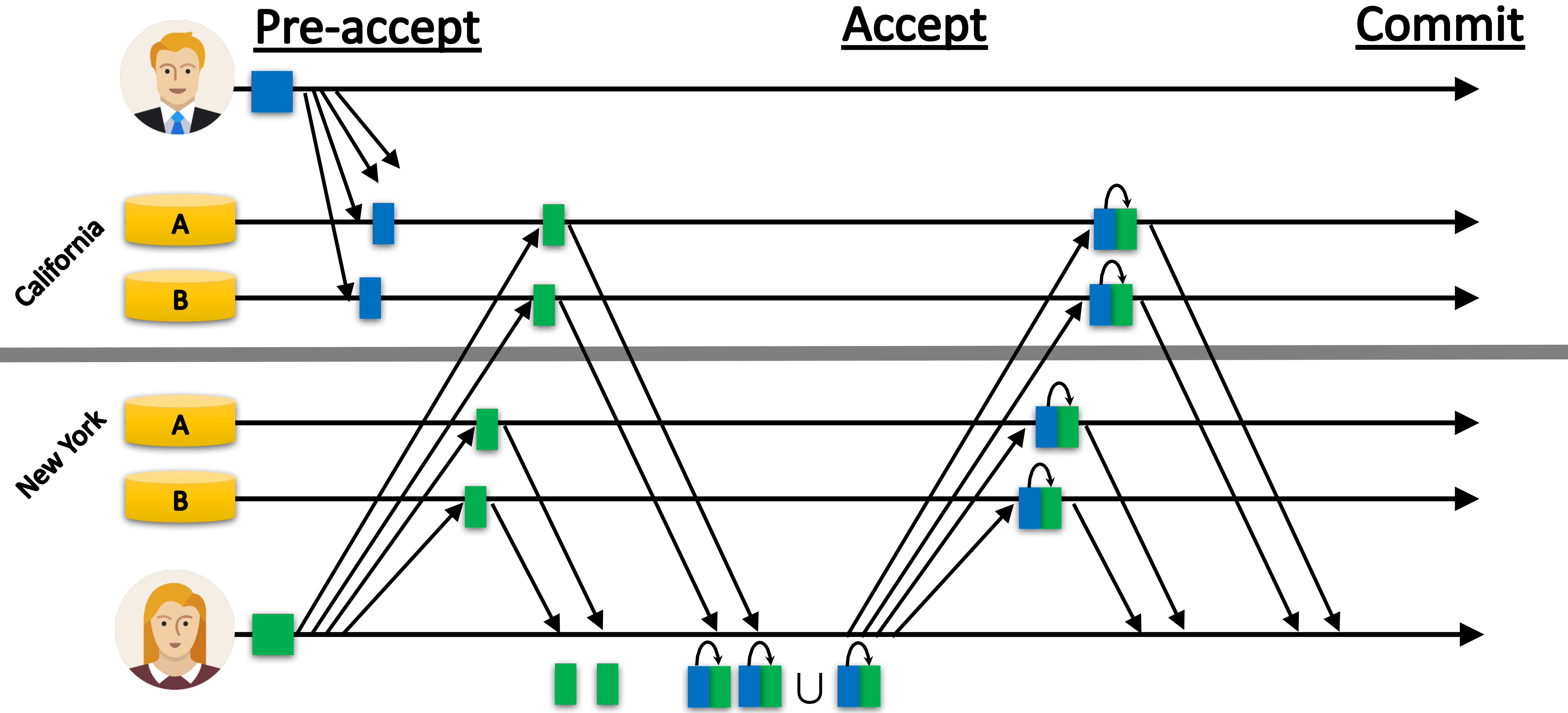
No Conflicts: Commit in 1 Wide-Area Round Trip



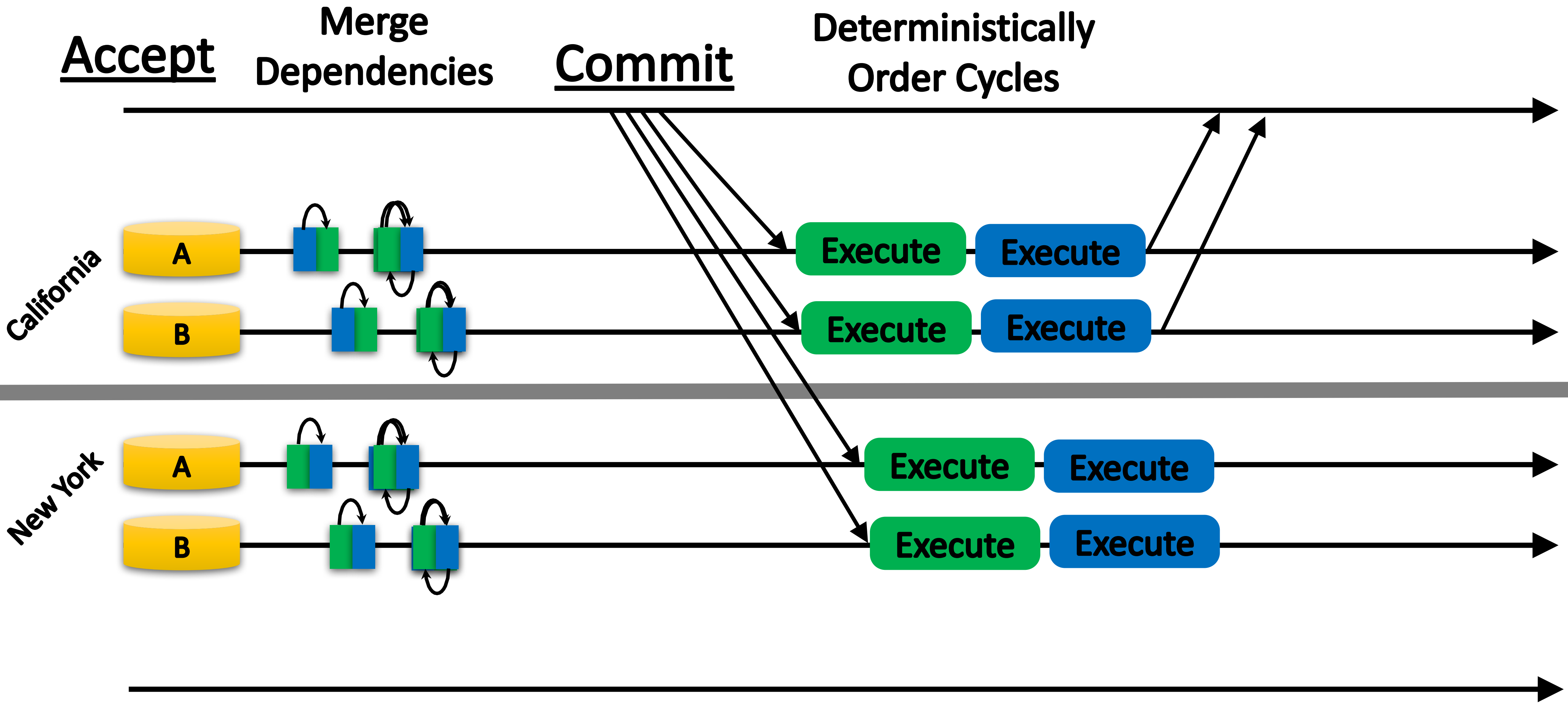
Conflicts: Commit in 2 Wide-Area RTT



Conflicts: Commit in 2 Wide-Area Round Trips



Conflicts: Commit in 2 Wide-Area Round Trip



Janus Achieves Fewer Wide-Area Round Trips and Commits Under Conflicts

- No conflicts: commit in 1 wide-area round trip
  - Pre-accept sufficient to ensure same order under failures

- Conflicts: commit in 2 wide-area round trips
  - Accept phase replicates dependencies to ensure same order under failures

Janus Paper Includes Many More Details

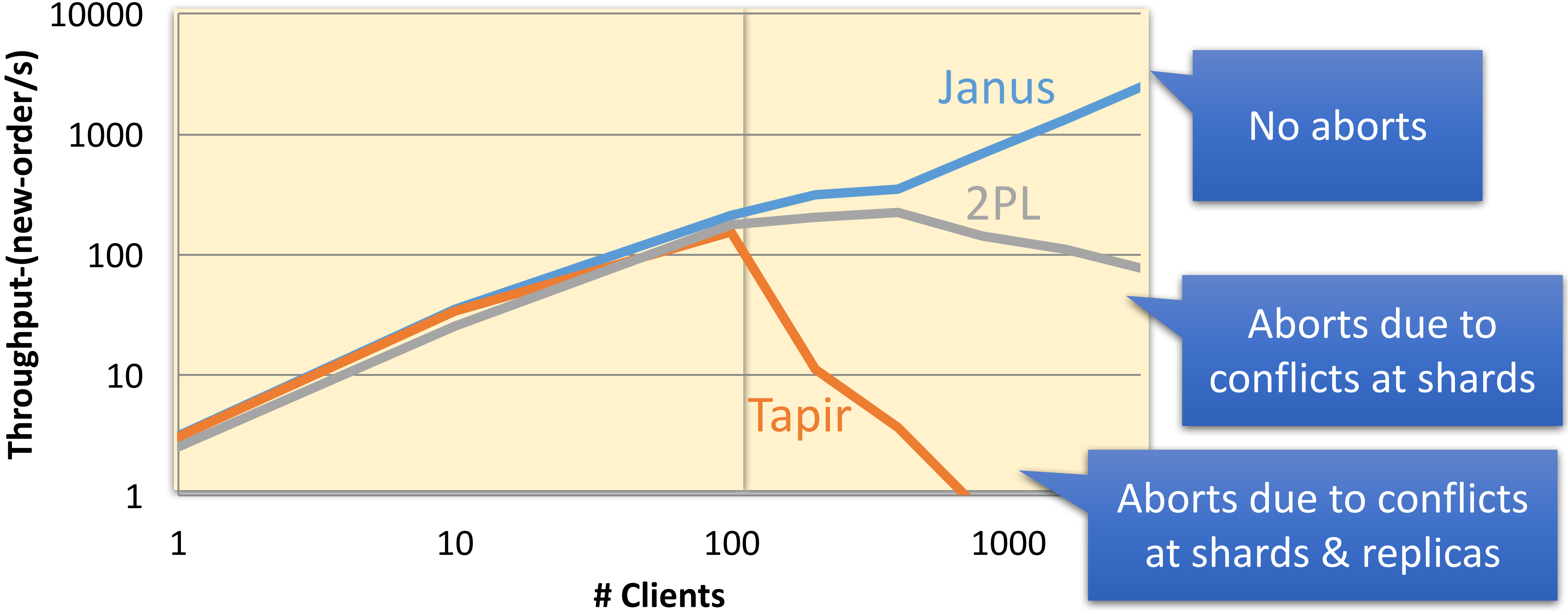
- Full details of execution
- Quorum sizes
- Behavior under server failure
- Behavior under coordinator (client) failure
- Design extensions to handle dynamic read & write sets

Evaluation

https://github.com/NYU-NEWS/janus

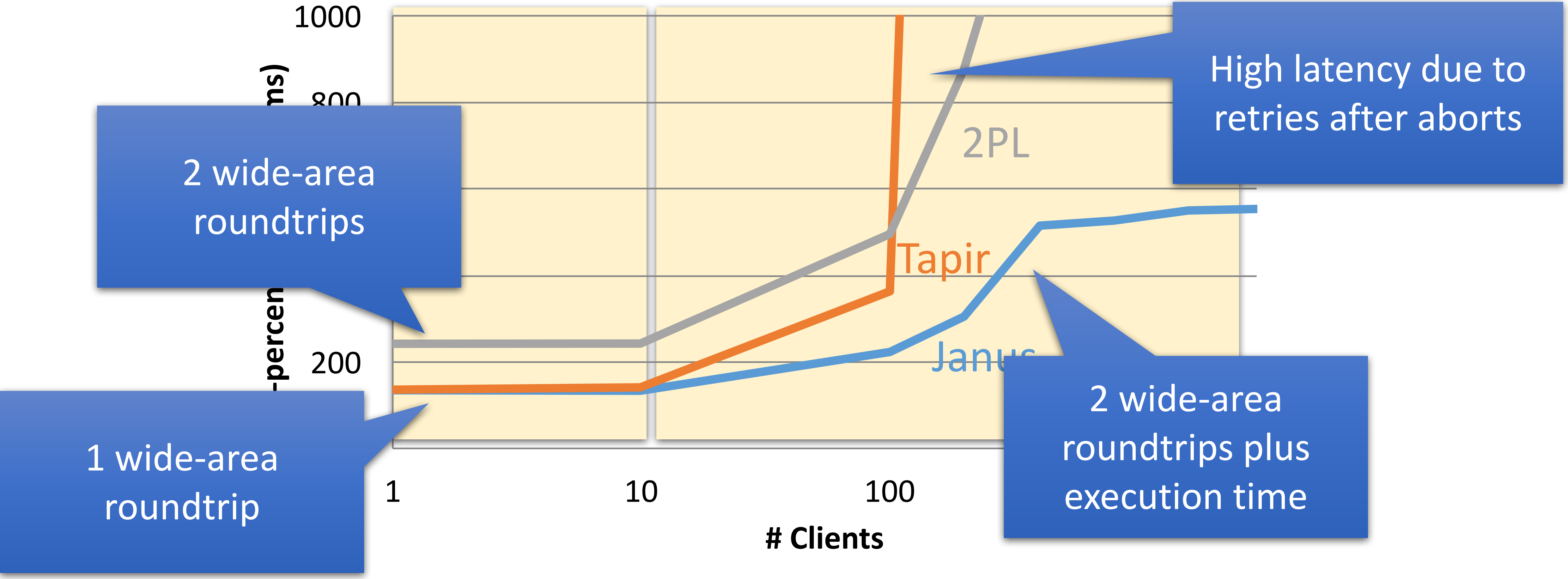
- Throughput under conflicts
- Latency under conflicts
- Overhead when there are no conflicts?
- Baselines
  - 2PL (2PC) layered on top of MultiPaxos
  - TAPIR [SOSP'15]
- Testbed: EC2 (Oregon, Ireland, Seoul)

Janus Commits under Conflicts for High Throughput



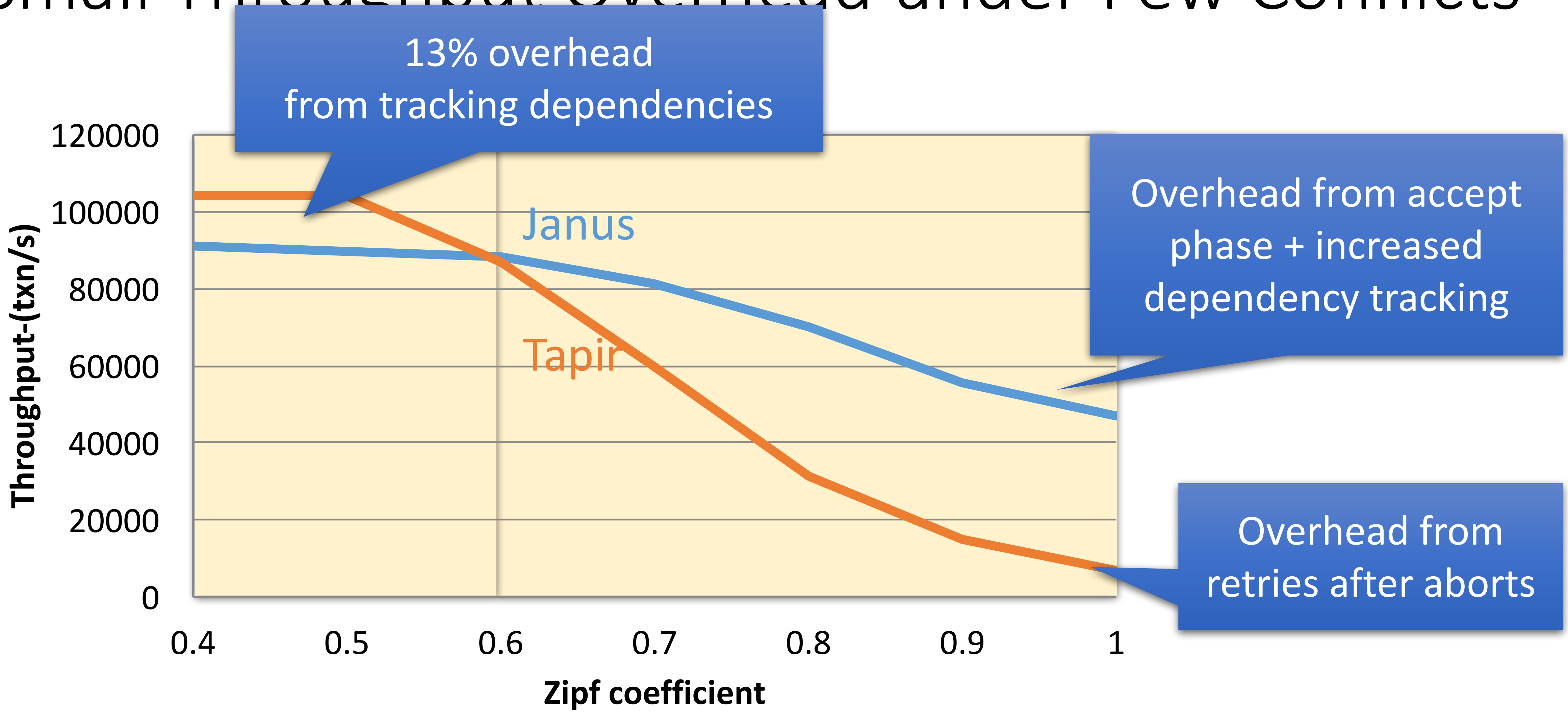
TPC-C with 6 shards, 3-way geo-replicated (9 total servers), 1 warehouse per shard.

Janus Commits under Conflicts for Low Latency



TPC-C with 6 shards, 3-way geo-replicated (9 total servers), 1 warehouse per shard.

Small Throughput Overhead under Few Conflicts



Microbenchmark with 3 shards, 3-way replicated in a single data center (9 total servers).

Related Work

	Isolation Level	1 RTT	Commit under Conflicts
Janus [OSDI’16]	Strict-Serial	✔	✔
Tapir [SOSP’15]	Strict-Serial	✔	✖
Rep.Commit [VLDB’13]	Strict-Serial	✔	✖
Calvin [SIGMOD’12]			✔
Spanner [OSDI’12]			✖
MDCC [EuroSys’13]			✖
COPS [SOSP’11]	Causal+	✔	✔
Eiger [NSDI’13]	Causal+	✔	✔

EPaxos [SOSP’13]
Rococo [OSDI’14]

Conclusion

- Two limitations for layered transaction protocols
  - Multiple wide-area round trips in the best case
  - Conflicts cause aborts
- Janus consolidates concurrency control and consensus
  - Ordering requirements are similar and can be combined!
  - Establishing a single ordering with dependency tracking enables:
    - Committing in 1 wide-area round trip in the best case
    - Committing in 2 wide-area round trips under conflicts
- Evaluation
  - Small throughput overhead when there are no conflicts
  - Low latency and good throughput even with many conflicts

Conclusion

- Two limitations for layered transaction protocols
  - Multiple wide-area round trips in the best case
  - Conflicts cause aborts

- Janus consolidates concurrency control and consensus
  - Ordering requirements are similar and can be combined!
  - Establishing a single ordering with dependency tracking enables:
    - Committing in 1 wide-area round trip in the best case
    - Committing in 2 wide-area round trips under conflicts

- Evaluation
  - Small throughput overhead when there are no conflicts
  - Low latency and good throughput even with many conflicts