Archie: A Speculative Replicated Transactional System

Sachin Hirve, Roberto Palmieri, Binoy Ravindran
Systems Software Research Group
Virginia Tech

What do we want for our transactional application?

High availability

Strong consistency

High performance

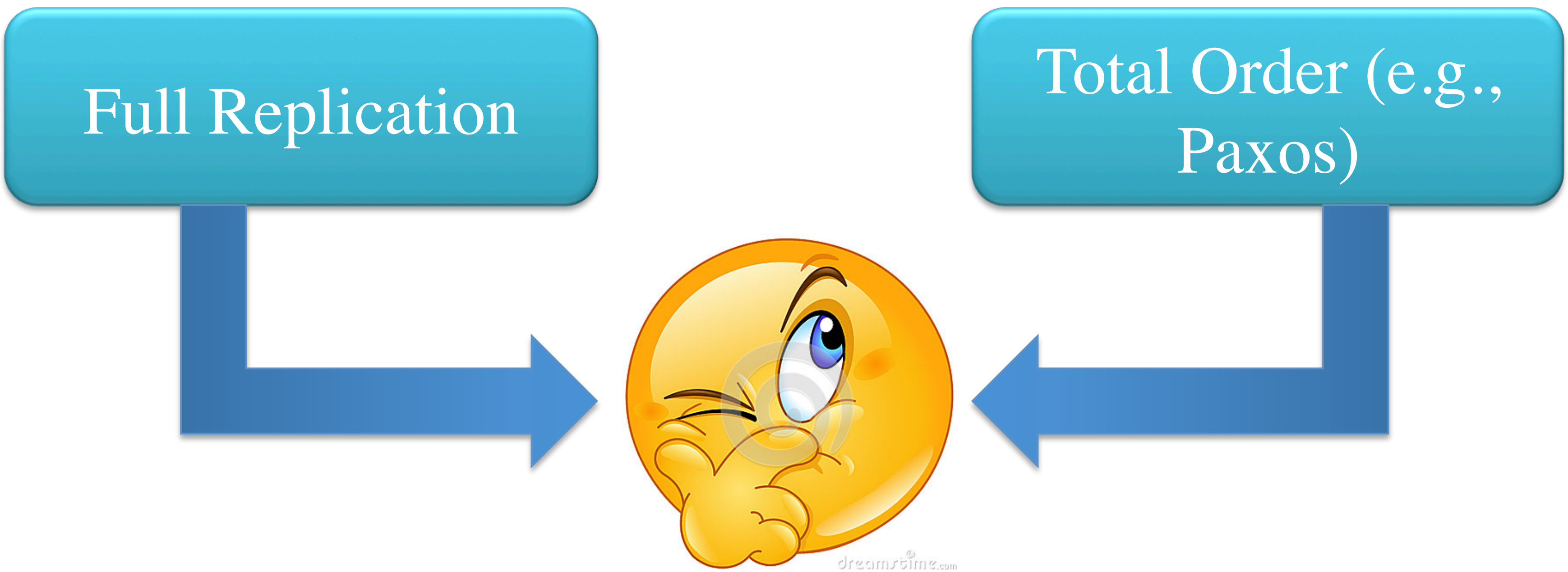
Programmability

Great scalability in small/med size

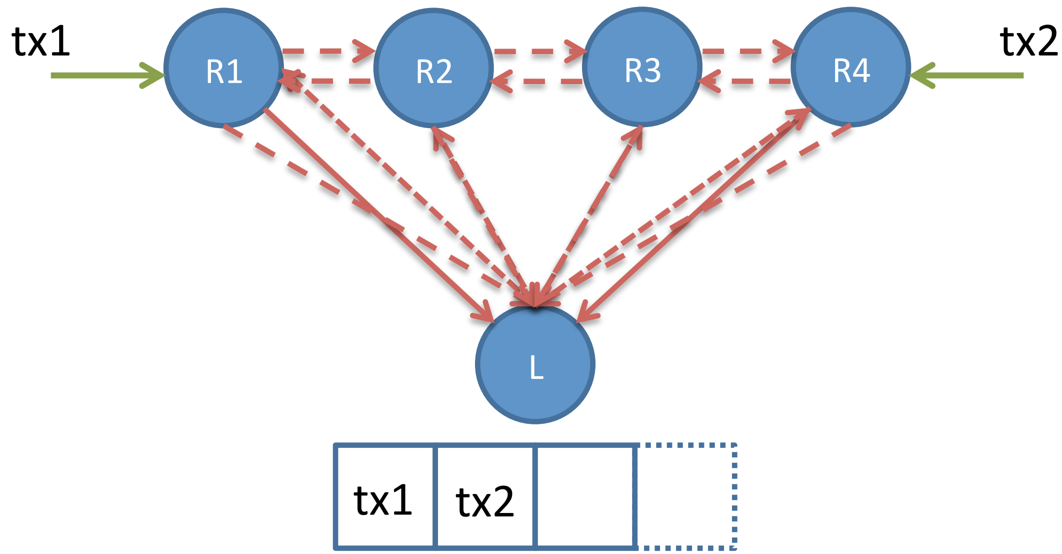
Fault tolerance

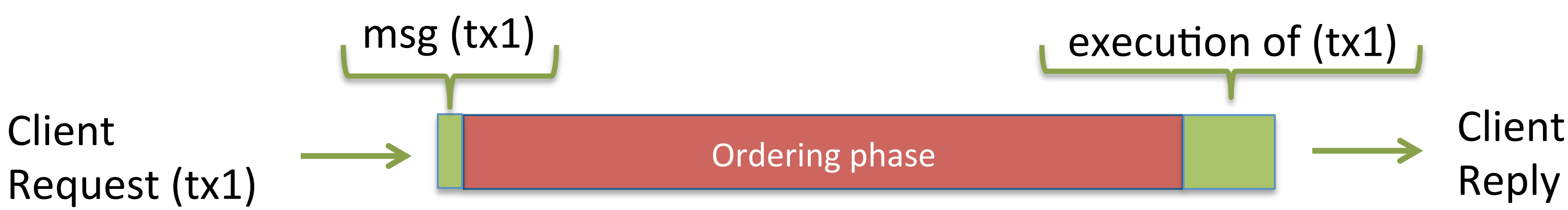
Low latency

Commonly used building blocks

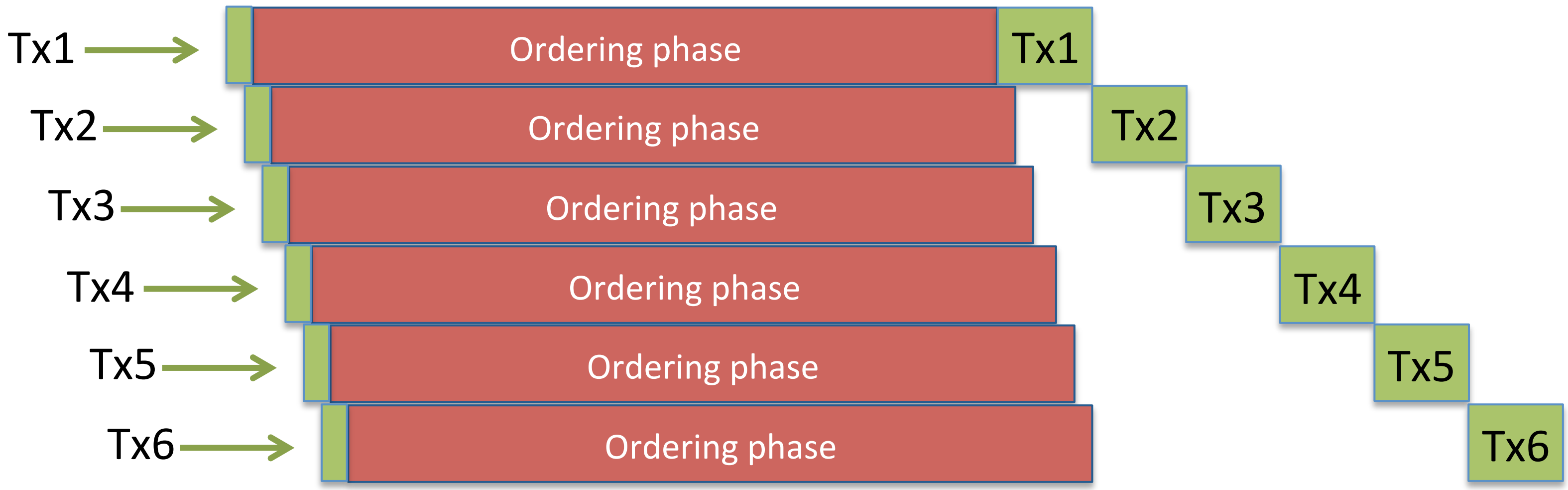


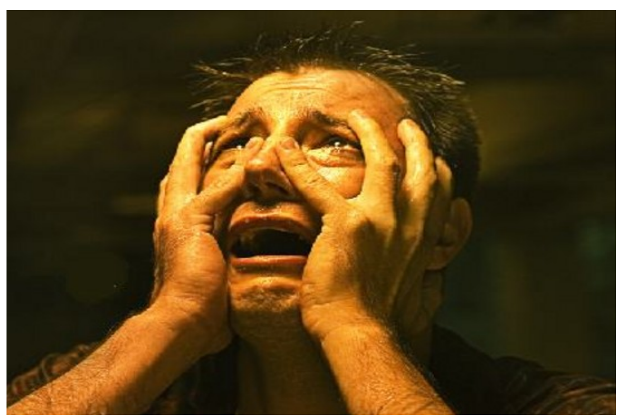
How does it work when combined?





…and when the load increases?



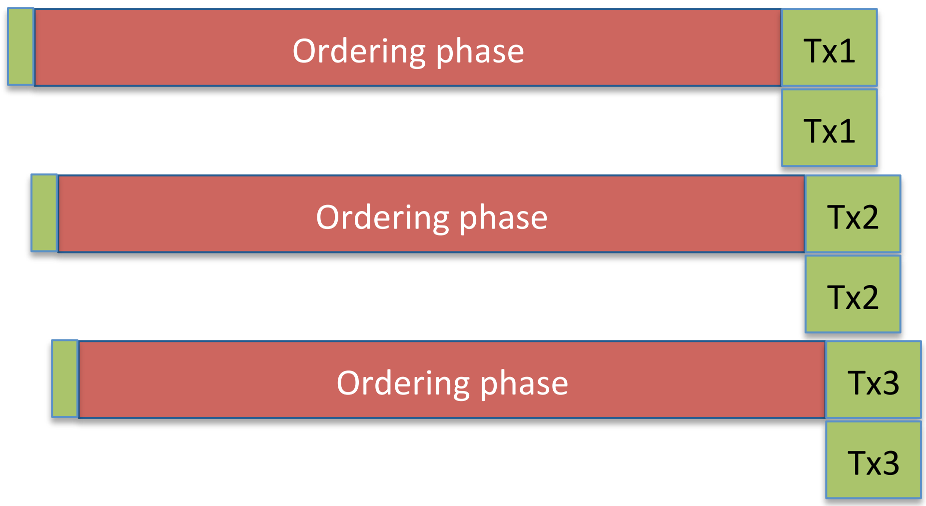


Goal:
Boost performance in the common case

Challenge:

Transaction response time = total order latency

Anticipate the work

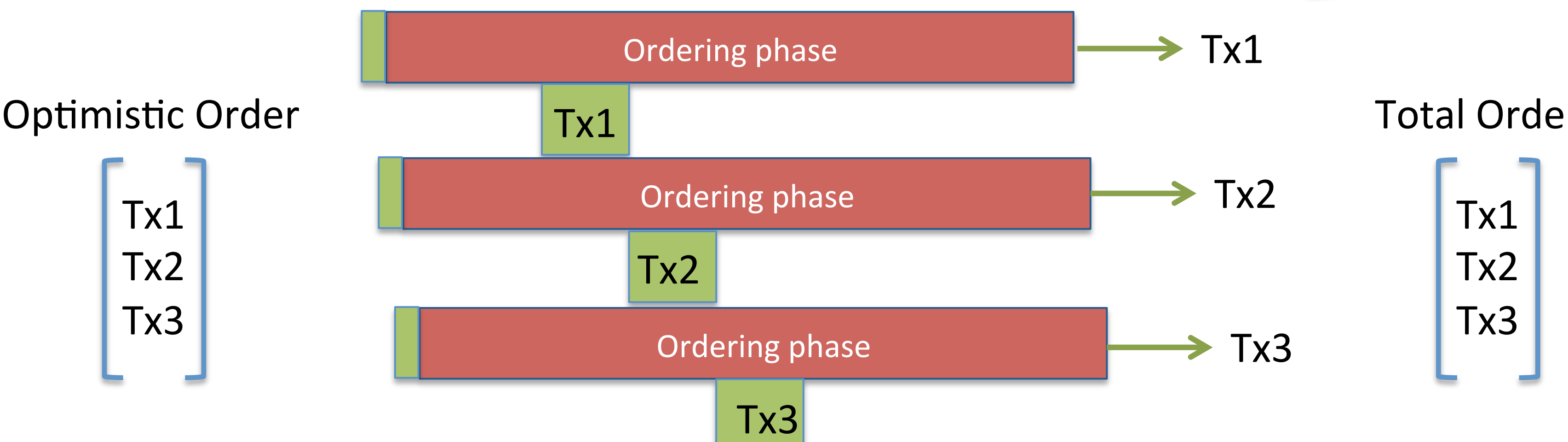


- Exploitation of an early notification triggered during the ordering phase (called *optimistic-delivery*)

Reliable Optimistic Delivery

- Avoid any mismatch between the optimistic delivery order and total order
- Maximize the time between the optimistic delivery and the establishment of the total order

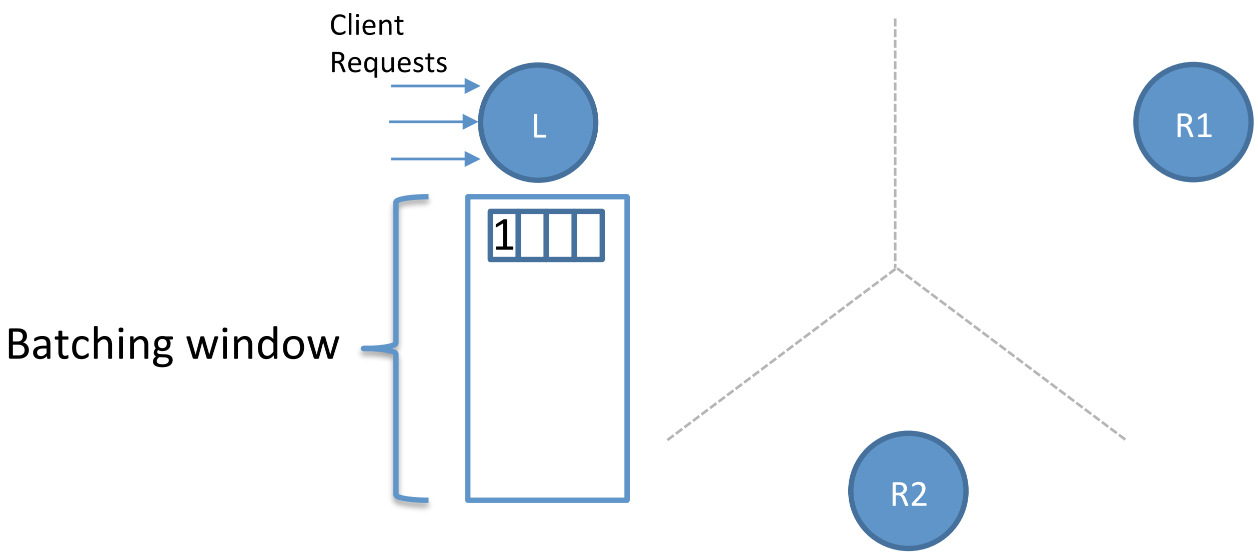
Stable leader



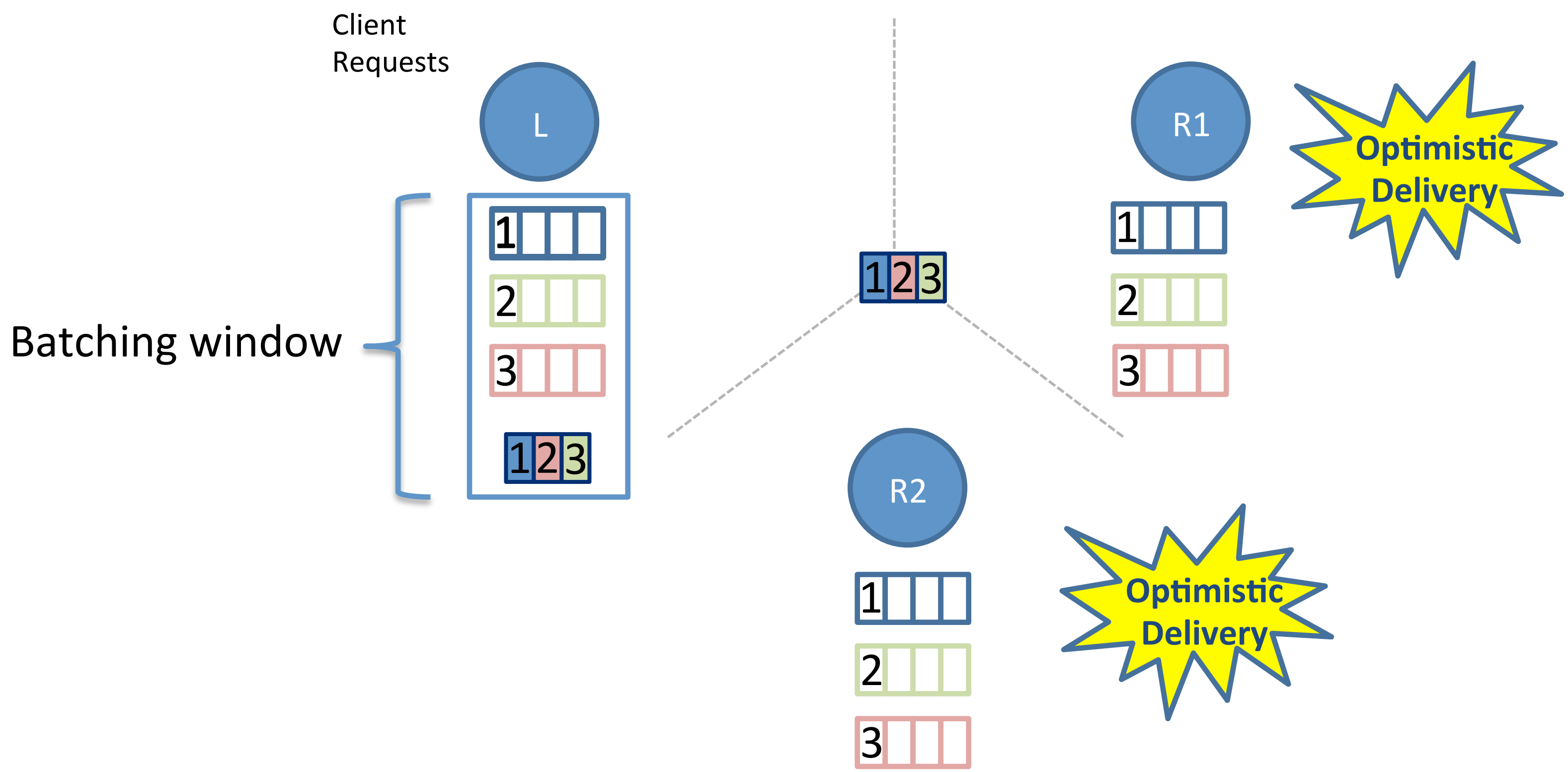
MIMOX

- Multi Paxos + Reliable Optimistic Delivery
  - Exploit the leader's batching time for maximizing the overlapping time
  - Exploit the leader's knowledge for making the optimistic delivery reliable
    - Tag messages with leader's expected order

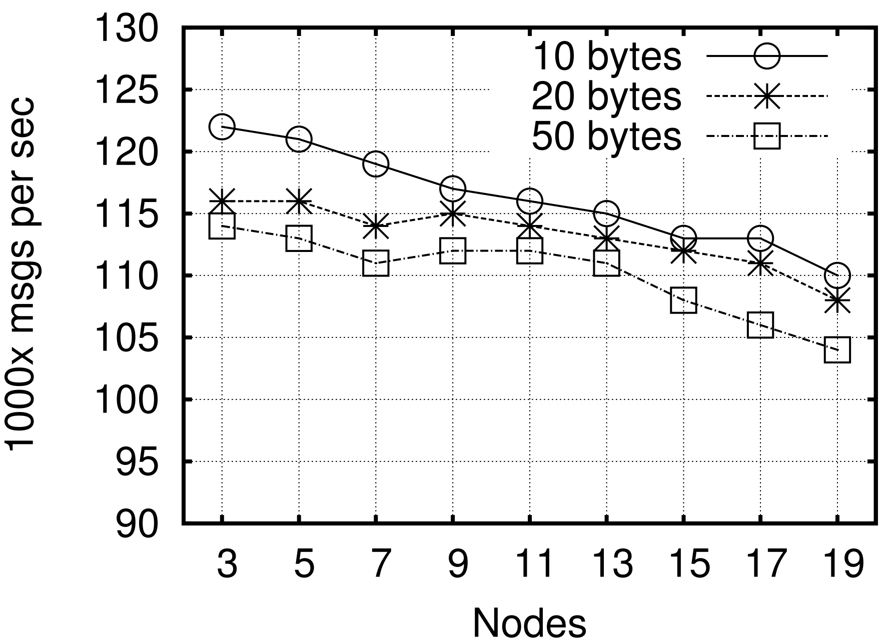
MIMOX: Batch creation

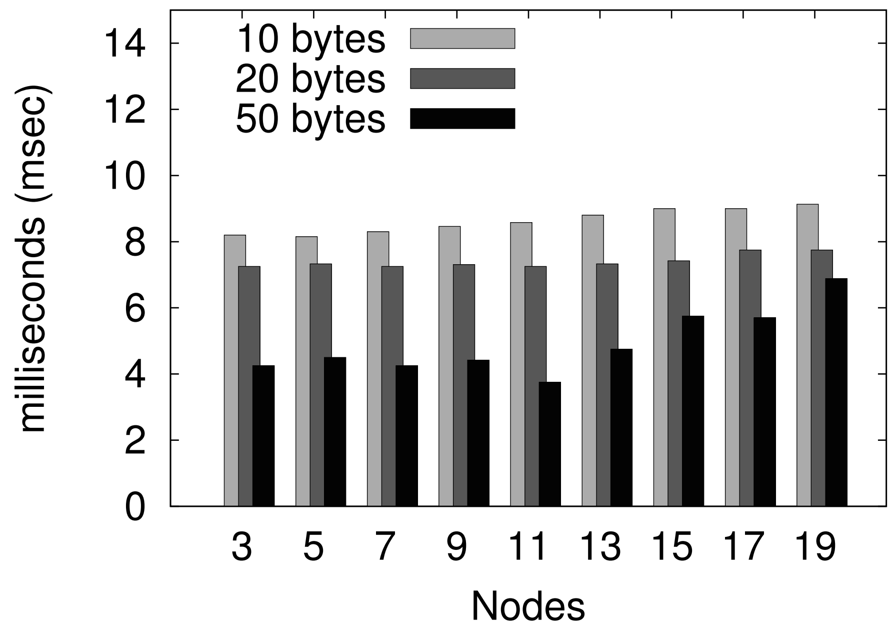


MIMOX: Batch broadcast

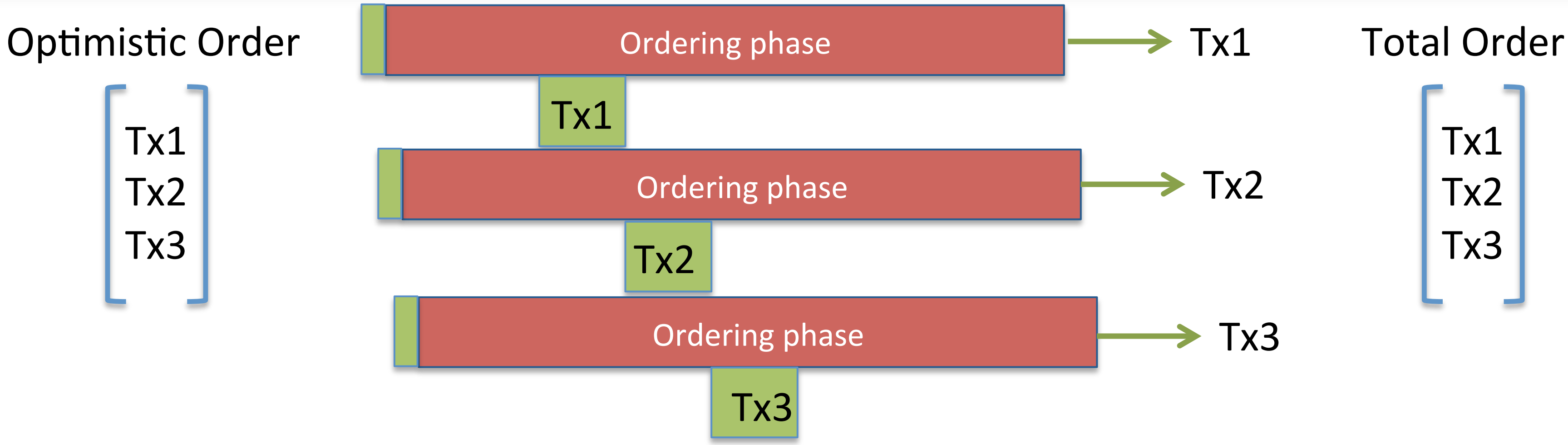


MIMOX: Performance





Transaction Processing



- Exploit parallelism
- Commit in order
- Minimal overhead for conflict detection and resolution

Key points of ParSpec

- Reduce problem's complexity by activating MaxSpec transactions at-a-time
  - meta-data's size is fixed
  - set of possible conflicting transactions is limited
- Parallel execution but in-order speculative commit (*x-commit*)

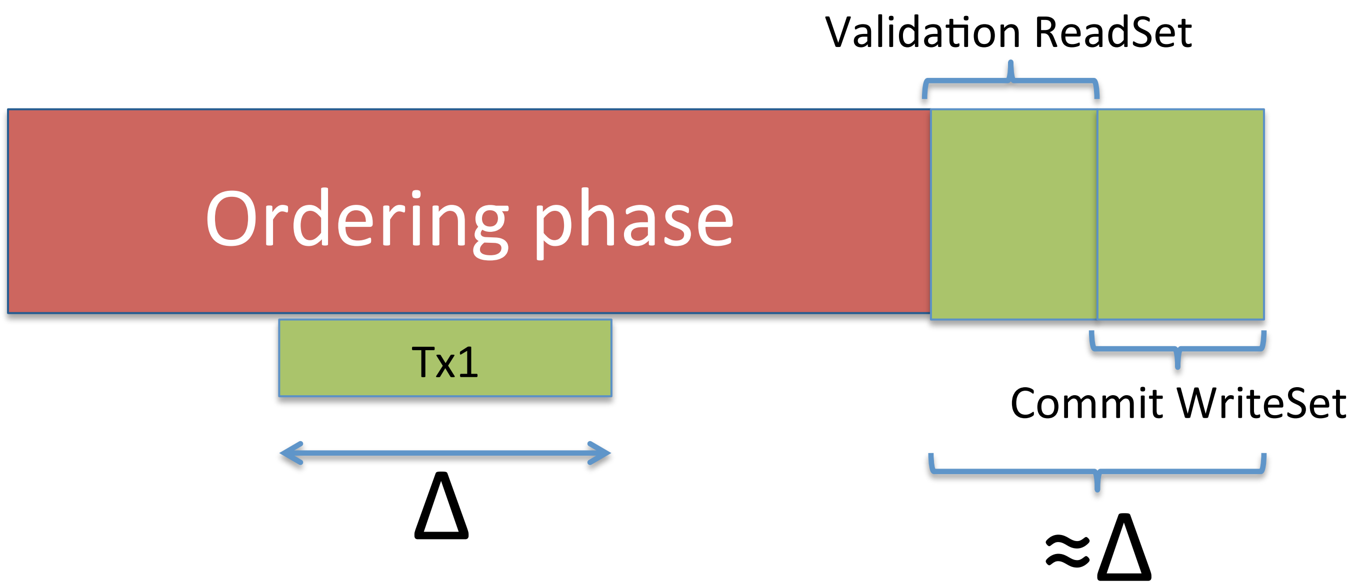
ParSpec: details

Optimistic Order: | T1 | T2 | T3 | T4 | T5 | T6 | T7 | T8 | MaxSpec = 4

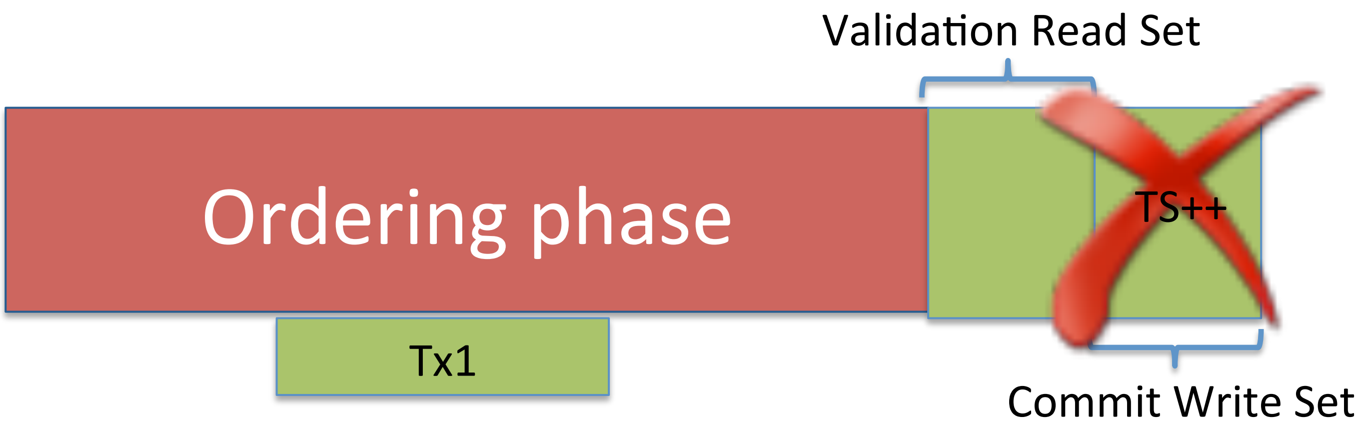
T2 (X): Begin	T3 (X): Begin	X: Read bit-array	Abort bit-array
		0 0 0 0	0 0 0 0
		0 0 0 0	0 0 0 0
	Read(X)	0 0 1 0	0 0 0 0
	Wait	0 0 1 0	0 0 0 0
Read(X)		0 1 1 0	0 0 0 0
Write(X)		0 1 1 0	0 0 0 0
X-commit			0 0 1 0
	Abort; Re-execute	OR bitwise	

Is this enough for achieving the goal?

- Unfortunately not…

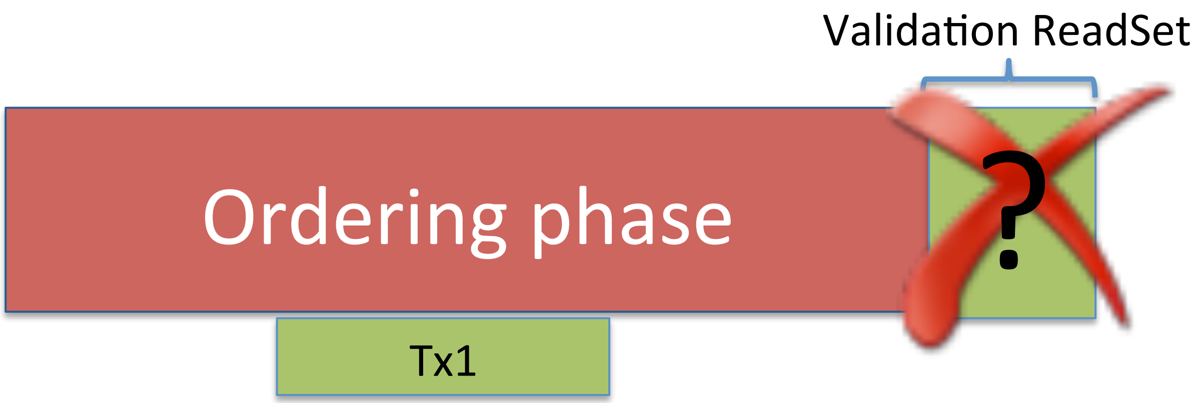


Solution for Write Set



- Speculative versions are associated with an tentative commit timestamp, which is the expected commit timestamp in case of no mismatch between optimistic and total order

Solution for Read Set



- If the leader is not either crashed or suspected (it is stable), then there is no chance of mismatch between optimistic and total order
- ParSpec commits according to optimistic order and make them visible during the total order

We made it!

Ordering phase

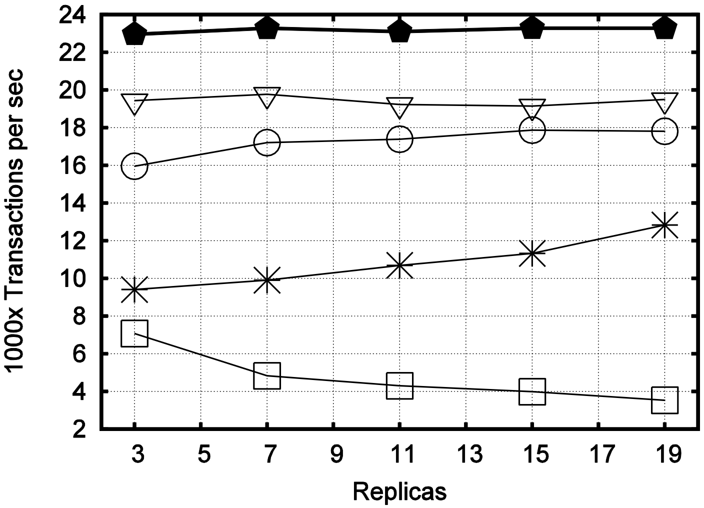
TS++

Tx1

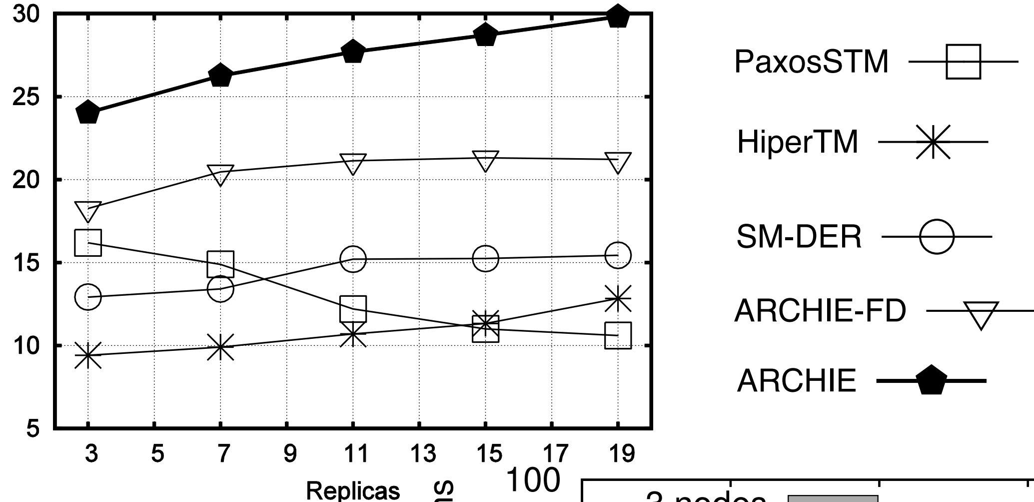
Evaluation

- Testbed – PRObE cluster (19 nodes)
  - AMD Opteron 6272, 64-core, 2.1 GHz CPU
  - 128 GB RAM and 40 Gbps Ethernet
- Benchmarks
  - Bank, TPC-C and Vacation
- Competitors
  - PaxosSTM: DUR-based approach; it suffers from remote aborts
  - SM-DER: Post final delivery single-thread execution
  - HiperTM: SM-DER based single thread execution
  - Archie-FD: Archie but post final delivery parallel processing

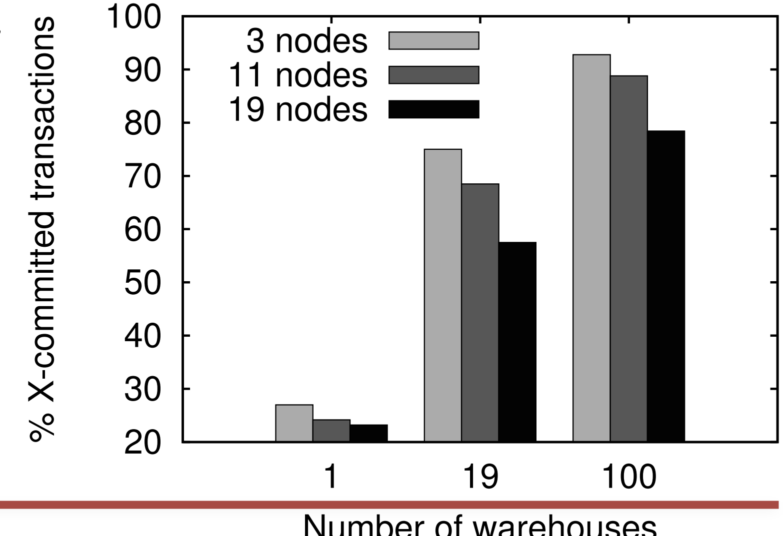
Evaluation: TPC-C



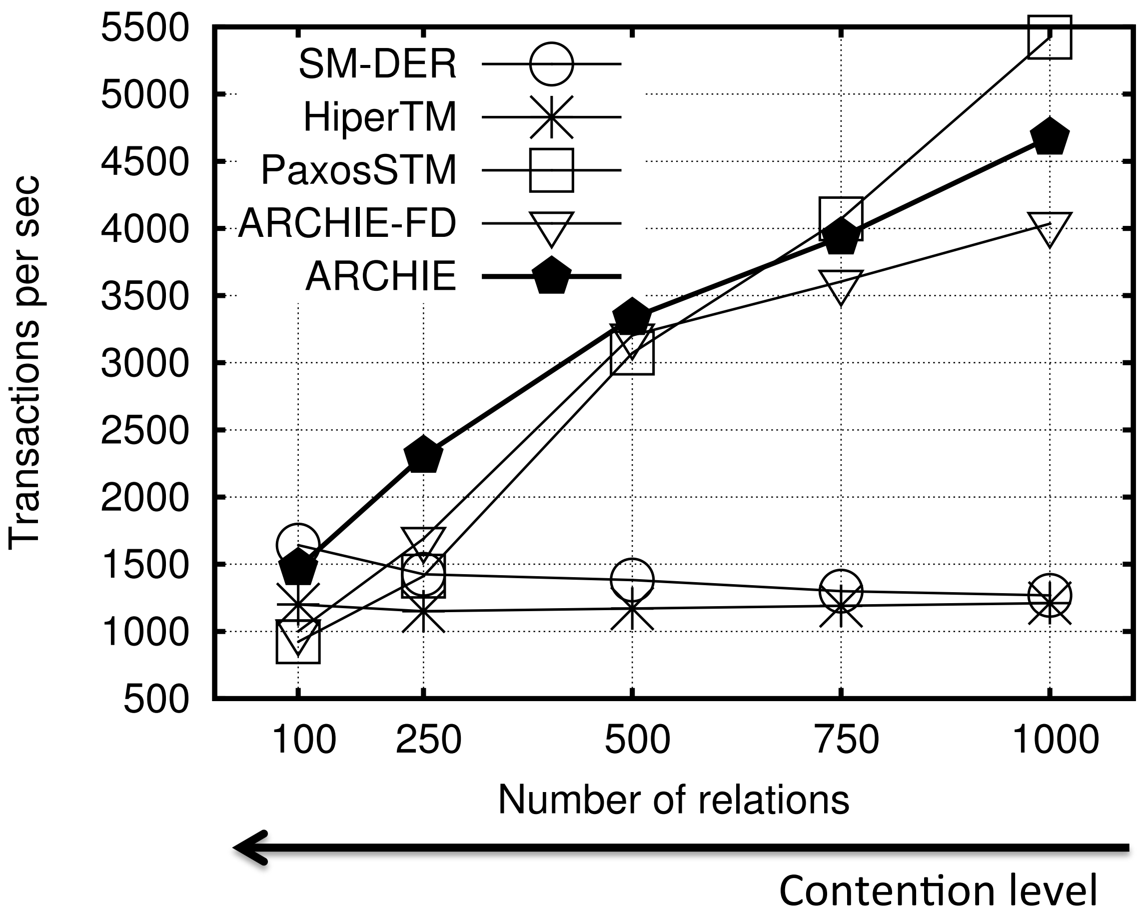
Throughput
19 warehouses



Throughput
100 warehouses



Evaluation: Distributed Vacation



Thanks!

Questions?

Hyflow

Systems Software Research Group

Research project's web-site: www.hyflow.org