Strong consistency is not hard to get: Two-Phase Locking and Two-Phase Commit on Thousands of Cores

Conference Paper

Author(s):
Barthels, Claude; Müller, Ingo; Taranov, Konstantin; Alonso, Gustavo; Hoefler, Torsten

Publication date:
2019-09

Permanent link:
https://doi.org/10.3929/ethz-b-000394714

Originally published in:
Proceedings of the VLDB Endowment 12(13), https://doi.org/10.14778/3358701.3358702

Systems Group
Department of Computer Science
ETH Zürich

Strong consistency is not hard to get: Two-Phase Locking and Two-Phase Commit on Thousands of Cores

Claude Barthels, **Ingo Müller**, Konstantin Taranov, Gustavo Alonso, Torsten Hoefler.
In: PVLDB 12(13): 2019. DOI: 10.14778/3358701.3358702
Presented at VLDB Tokyo, September 1st and 4th, 2020.

Recent advances in transaction processing have renewed the interest in the field, but also highlighted some challenges

Encountered challenges

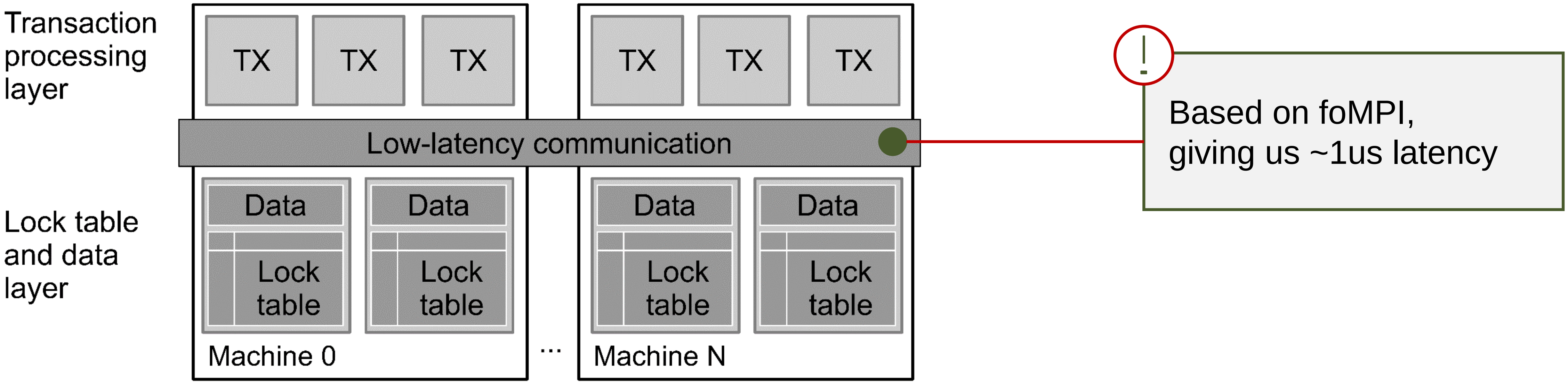
- Snapshot isolation instead of serializability (e.g., NAM-DB)
- Restrictions on long-running transactions (e.g., FaRM, H-Store, Calvin, R/W transactions: Silo, HyPer)
- Only support partitioned workloads (e.g., H-Store)
- Require read/write set to be known ahead of time/discovered (e.g., H-Store, Calvin)

Common assumption

> *Two-phase locking and two-phase commit are slow*

Is this assumption still true on modern high-performance networks?

We use low-latency communication primitives to implement distributed concurrency control mechanisms on modern networks



- Simple, traditional, yet powerful design of a lock table
- Supports concurrent readers, hierarchical locking, and arbitrary transactions
- Achieves strict serializability

The experiments were conducted on a high-end supercomputer with a state-of-the-art network

The cluster

- Cray XC40 super-computer with up to 4096 cores
- Aries routing and communi-cations ASIC, and Dragonfly network topology

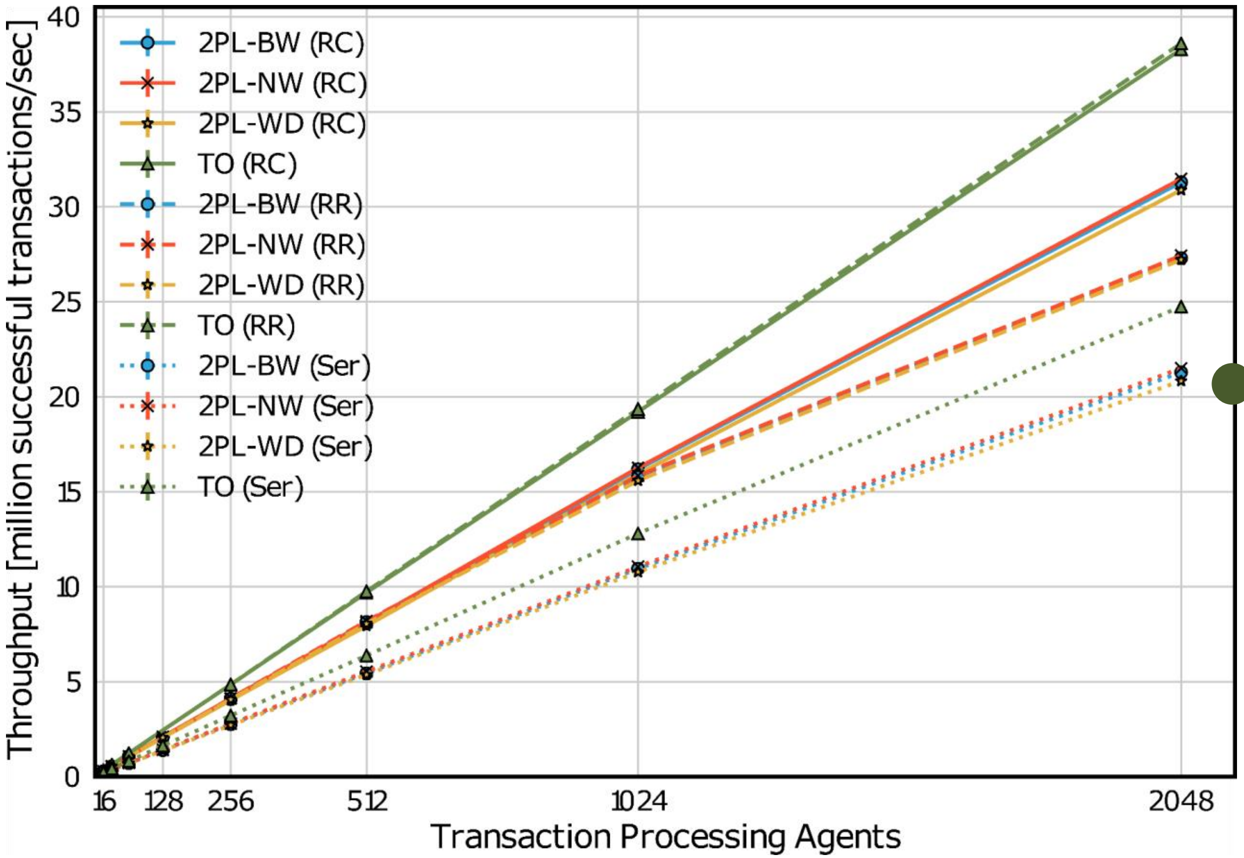
Implementation variants

- 2PL+2PC with different dead-lock detection and avoidance mechanisms:
  - No Wait (NW)
  - Wait-Die (WD)
  - Bounded Wait (BW)
- Timestamp ordering (TO)

Workload

- Lock trace produced by MySQL running TPC-C
- Isolation levels:
  - Serializable (Ser)
  - Repeatable Read (RR)
  - Read committed (RC)

Weak scaling experiments without contention (2048 warehouses) show that our implementation can scale to thousands of cores

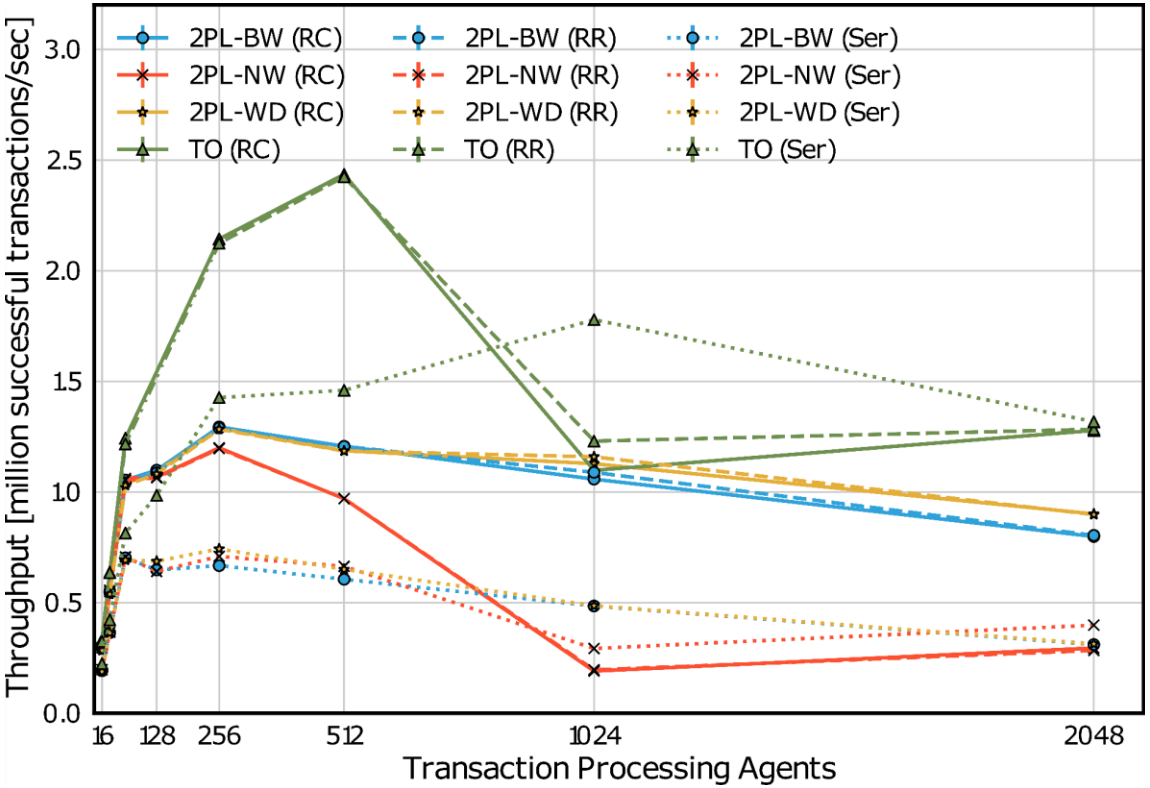


2PL+2PC:	9.5M SNOT/s
TO:	11.0M SNOT/s
FaRM:	4.5M SNOT/s
NAM-DB:	6.5M SNOT/s
Calvin:	380k SNOT/s

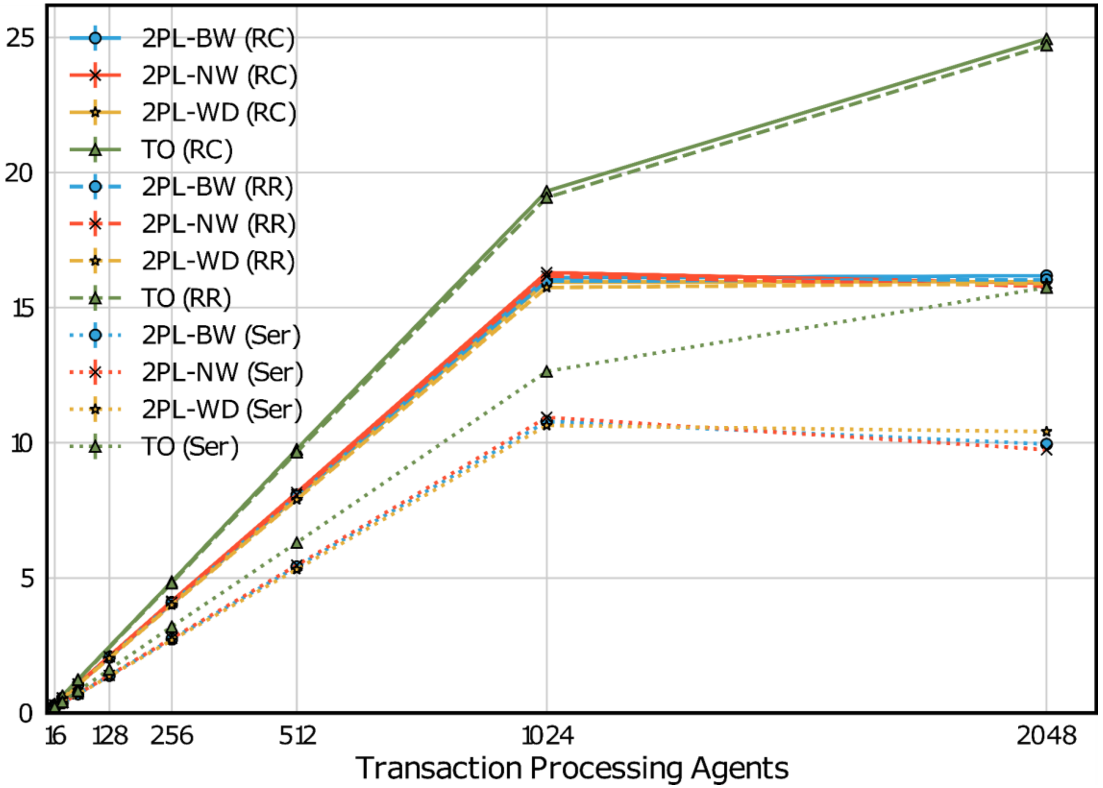
Two-phase locking and two-phase commit are actually very competitive!

Weak scaling experiments with contention show that scaling to thousand cores requires a scalable system as well as a scalable workload

Medium contention (64 warehouses)

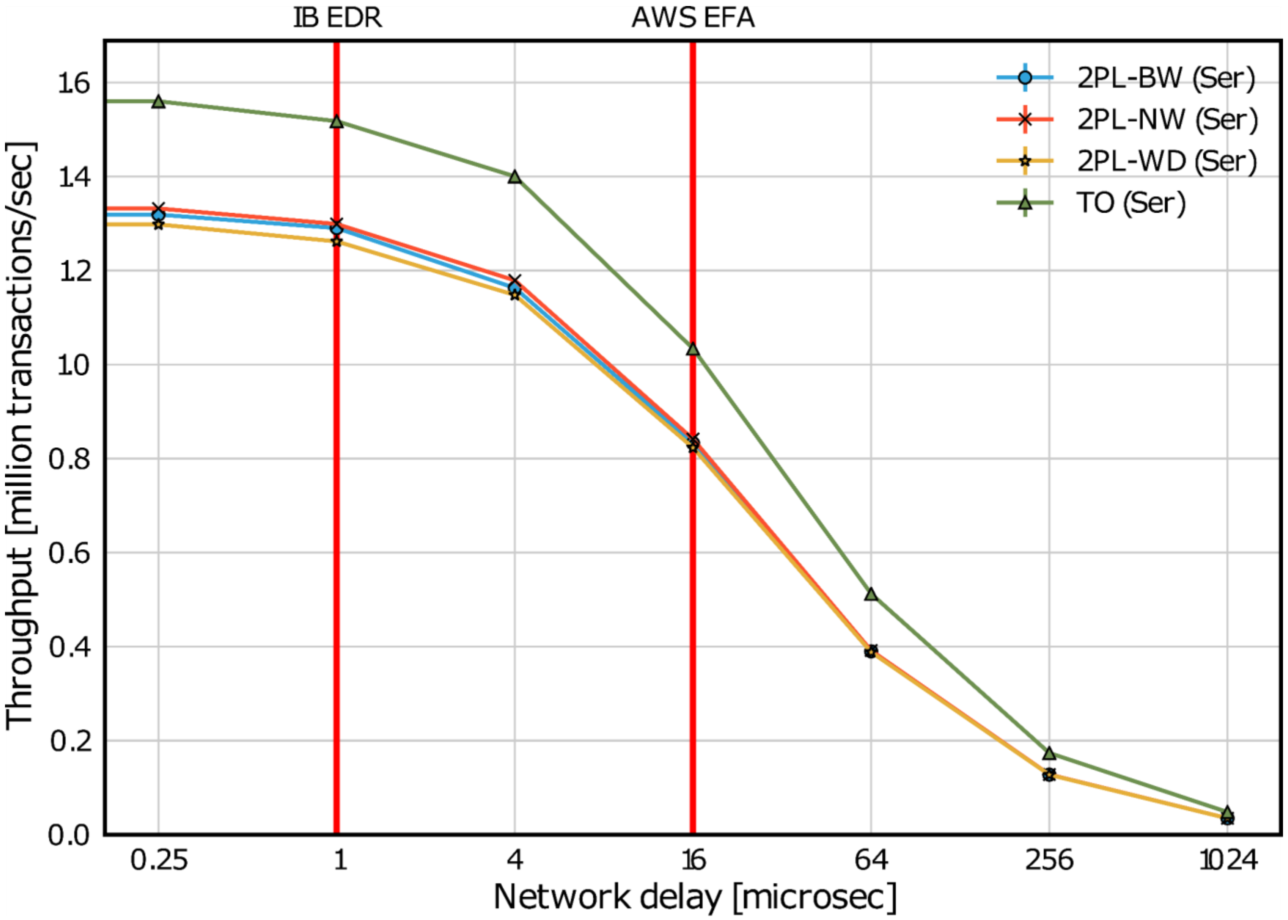


Light contention (1024 warehouses)



Workloads with high contention have limited scaling!

A higher network latency will reduce the overall throughput, but modern networks (both on premise and in the cloud) are competitve



Results are transferrable to commodity hardware and the cloud!

Summary: How does modern networking hardware affect concurrency control?

- Traditional 2PL+2PC and TO on state-of-the-art networks scale to thousands of cores and achieve competitive performance
  - Concurrency control mechanism is not the bottleneck
  - No compromise in isolation level or transaction types
- Results hold on commodity hardware and in the cloud
- Smart NICs promise even higher performance (see paper for details)

Thank you! Questions?