Path-Sensitive Atomic Commit: Local Coordination Avoidance for Distributed Transactions

Tim Soethout, Tijs van der Storm, Jurgen J. Vinju

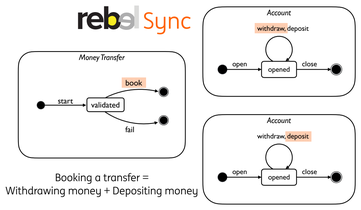
CWI Centrum Wiskunde & Informatica

ING

Goal: Domain Knowledge to Faster Transactions

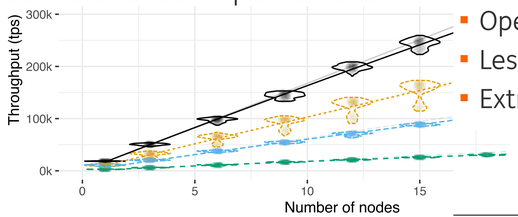
- Large distributed (enterprise) software systems are complex
- Consistency / Isolation ⟺ Scalability / Performance
- DSLs and models capture domain knowledge without implementation details
- Scope: Distributed concurrent objects with async messaging

From:
Domain Models



Booking a transfer =
Withdrawing money + Depositing money

To:
Scalable/correct implementation



Problem: Bottleneck on high-contention objects

Tax office bank account:

- Strong consistency requirements
- Strict time bounds
- Many tax and benefits money transfers
- Potential bottleneck for high contention

Implementing Sync with 2PL/2PC

Two-Phase Locking (2PL): Concurrency Control: Single object, No concurrent access to object

Two-Phase Commit (2PC): Atomic Commitment: Multiple objects, Well-understood and Often used

Combined 2PL/2PC: Serializable Isolation guarantees

A lot of waiting, but enough balance for both, right? ⟹

Approach: Reduce coordination w/ Domain Knowledge

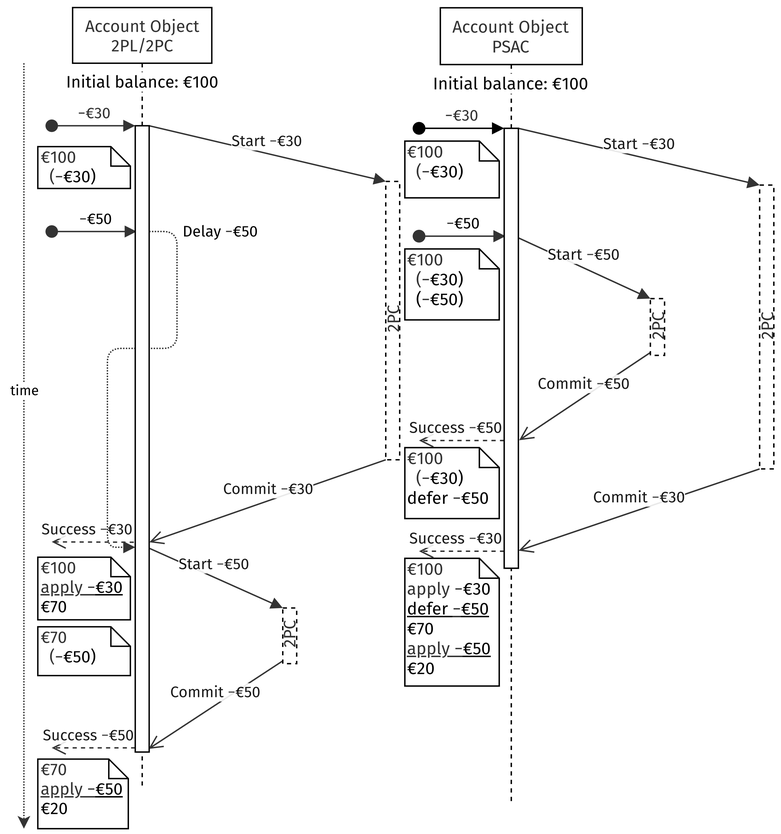
Insight: Enough balance for both withdrawals and the commit or abort of first operation does not influence second

Increase parallelism where it is safe

Enter **Path-Sensitive Atomic Commit (PSAC)**:

- Operations in parallel, when safe
- Less waiting/locking of objects
- Extra computing time vs. waiting on message IO

2PL/2PC vs. PSAC



Evaluation: Performance with varying contention

Message passing actors implementation of 2PL/2PC and PSAC.

Experiment data/results available @ doi:10.5281/zenodo.3405371

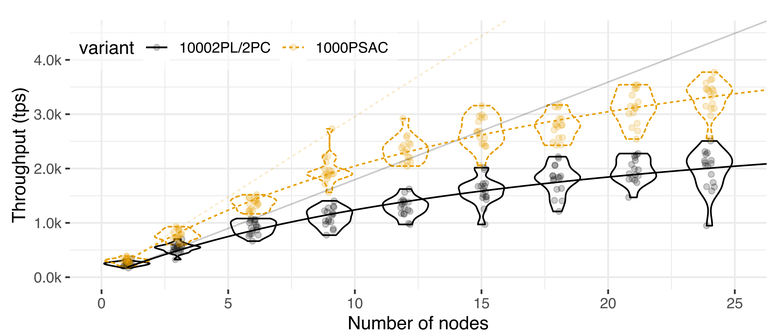
2PL/2PC is special case of PSAC with parallelism disabled

Experiments with varying contention:

- NOSYNC – Operations without synchronization
- SYNC – Uniform money transfers over 100.000 accounts
- SYNC1000 – Uniform money transfers over 1000 accounts

Results

Similar throughput for NOSYNC & SYNC, not enough contention



Under high-contention SYNC1000: Up to 1.8 times higher median throughout

Conclusion

- High contention bottleneck with 2PL/2PC
- Safe parallelism with PSAC; currently looking into isolation guarantees
- Promising for creating high-performant implementations from models

tim.soethout@ing.com