Benchmarking Geo-Distributed Databases using the SmallBank benchmark

Author: Cirtog Filip-Andrei
Responsible professor: Dr. Asterios Katsifodimos
Supervisor: Oto Mraz

TU Delft
Delft University of Technology

Introduction

Distributed & Geo-distributed Databases power global platforms:

- E-commerce (Amazon), Finance (US Bank), Social Media (Facebook)

Challenge on geo-distributed databases:

- Complex design trade-offs.
- Unclear which systems perform best under specific workloads or at global scale.

Academy vs Industry:

- TPC-C [1] → Too simple key-value operations, with half involving access to a single row [2].
- YCSB-T [3] → Assumes a centralized warehouse model and cannot create data hotspots [2].

Project Goal:

- Compare Calvin [4], SLOG [5], Detock [6], Janus [7] using the SmallBank benchmark [8].

Research Question

How do geo-distributed databases perform under the **SmallBank becnhmark [8]** in terms of throughput, latency, abort rate and bytes transferred across different scenarios.

Background

Calvin → Master sequencer to immediately process and forward transaction batches to replicas, reducing latency.

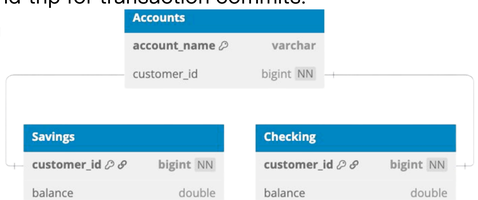
SLOG → Assigns each data item a home region. Low-latency single home transactions but uses a global ordering system for multi-home transactions.

Detock → Eliminates global ordering and uses a graph-based concurrency control protocol, reducing round-trip latencies and resolving deadlocks without transaction abortion.

Janus → Similar to Detock but slower due to synchronous data replication across regions, requiring at least one round-trip for transaction commits.

SmallBank workload replicates a bank environment:

- Balance
- DepositChecking
- TransactSaving
- Amalgamate
- WriteCheck

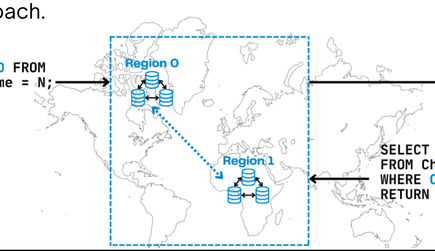


Implementation

Since Detock does not support dependent transactions, we implemented a two-phase approach.

PHASE 1

```
SELECT CustomerID FROM
Account WHERE Name = N;
```

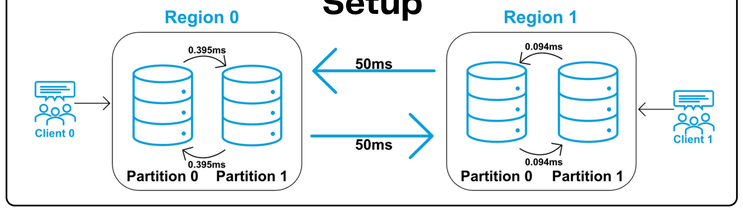


CustomerID

```
SELECT CheckingBalance, SavingsBalance
FROM Checking, Savings
WHERE CustomerID = Account. CustomerID;
RETURN CheckingBalance + SavingsBalance
```

PHASE 2

Setup



Results

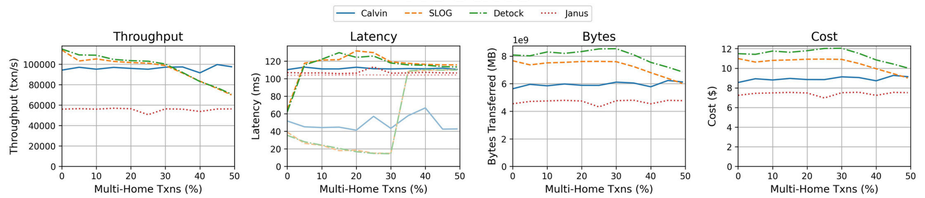


Figure 1: Comparison of results on the baseline scenario

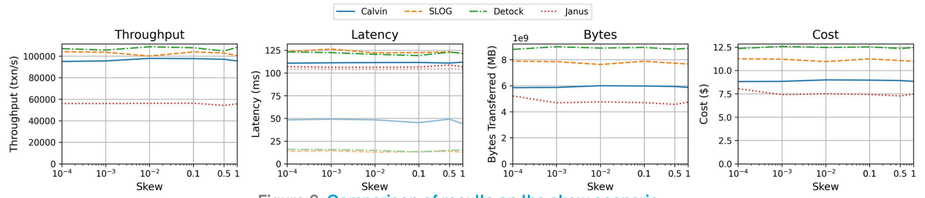


Figure 2: Comparison of results on the skew scenario

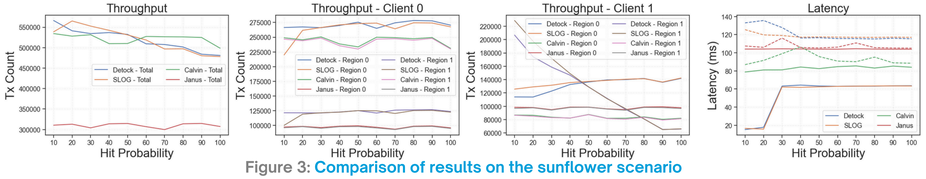


Figure 3: Comparison of results on the sunflower scenario

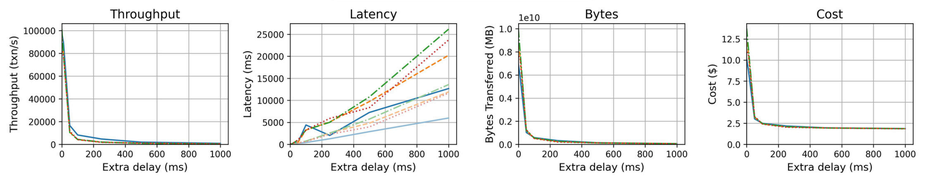


Figure 4: Comparison of results on the scalability scenario

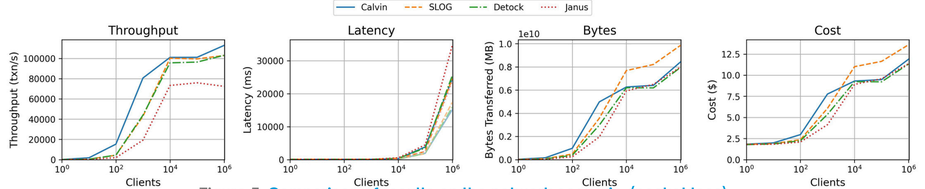


Figure 5: Comparison of results on the network scenario. (packet loss)

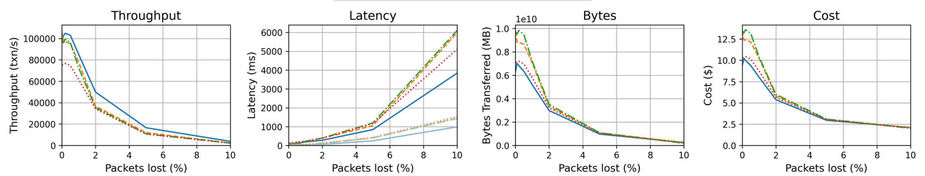


Figure 6: Comparison of results on the network scenario. (network delay)

Findings

Calvin

- Strongest performance at **high MH%**, highest throughput.
- But real-world MH% is usually low → Calvin falls behind **Detock** and **SLOG.**
- Cheaper than **Detock** and **SLOG.**

Detock

- Slightly outperforms **SLOG** in most cases.
- Graph-based concurrency control → excels under **high contention.**

Janus

- Consistently lowest throughput and highest latency.
- **Coordination round-trip** is a major bottleneck.

Other

- All systems perform similarly on **network latency, packet loss,** and **scalability tests.**

Discussion

Advantages

- Customizable Multi-Home and Multi-Partition Parameters
- Highlights Communication and Throughput Trade-offs
- Controlled Data Hotspot Generation

Limitations

- Dependent transactions
- Transaction Simplicity

Conclusion

Overall, the SmallBank benchmark shows that Detock and SLOG perform best, having high throughput and low latency. In contrast, Calvin proves more effective in environments with a high proportion of multi-home transactions and shows a lower cost of operation. In contrast, Janus shows the weakest performance across all scenarios.

References

[1] Scott T. Leutenegger and Daniel Dias. A Modeling Study of the TPC-C Benchmark. In Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data (SIGMOD '93), pp. 22–31, ACM, 1993.
[2] Luyi Qu et al. Are Current Benchmarks Adequate to Evaluate Distributed Transactional Databases? BenchCouncil Transactions on Benchmarks, Standards and Evaluations, 2(1):100031, 2022.
[3] Akon Dey et al. YCSB+T: Benchmarking Web-Scale Transactional Databases. In Proceedings of the 2014 IEEE 30th International Conference on Data Engineering Workshops (ICDEW), pp. 223–230, IEEE, 2014.
[4] Alexander Thomson et al. Calvin: Fast Distributed Transactions for Partitioned Database Systems. In Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data (SIGMOD '12), pp. 1–12, ACM, 2012.
[5] Kun Ren, Dennis Li, and Daniel J. Abadi. SLOG: Serializable, Low-Latency, Geo-Replicated Transactions. Proceedings of the VLDB Endowment (PVLDB), 12(11):1747–1761, July 2019.
[6] Cuong D. T. Nguyen, Johann K. Miller, and Daniel J. Abadi. Detock: High Performance Multi-Region Transactions at Scale. Proceedings of the ACM on Management of Data (ACM Manag. Data), 1(2), June 2023.
[7] Huai Mu et al. Consolidating Concurrency Control and Consensus for Commits Under Conflicts. In Proceedings of the 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI '16), pp. 517–532, USENIX Association, November 2016.
[8] Mohammad Alomari et al. The Cost of Serializability on Platforms That Use Snapshot Isolation. In Proceedings of the 2008 IEEE 24th International Conference on Data Engineering (ICDE 2008), pp. 576–585, IEEE, 2008.