COMP 513 Project

Rolis

Presented by
Olivier Michaud, Akshay Gopalakrishnan
McGill University

December 5th 2023

Overview

Project Description

Rolis: A software approach to efficiently replicating multi-core transactions

- Proposes a new consensus algorithm to improve throughput.
- Uses multiple threads per leader/follower to process transactions.
- Performs well upon failure recovery using *watermarks* to ensure synchronization when necessary.

Choice of Experiments

- Throughput
  - vs Silo - Algorithm is built by modifying Silo.
  - vs Calvin - Existing state-of-the-art.
- Latency
  - On different batch sizes.
  - Measured for $10^{th}, 50^{th}, 95^{th}$ percentiles.

Chosen Test Environment

AWS EC2 shared instances.
Differs from original system

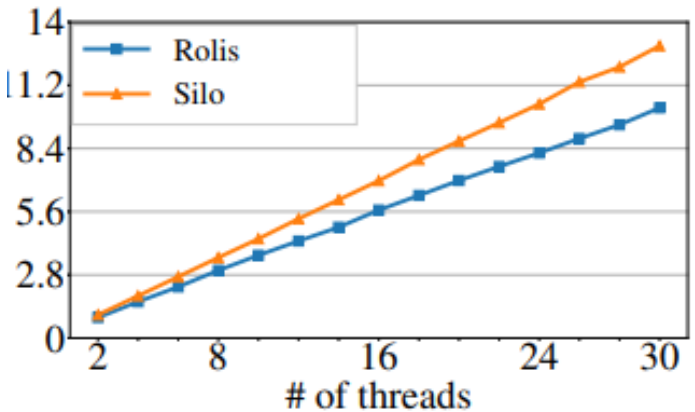
- 32vCPUs (Intel Xeon Platinum 8272CL)
- 128GB RAM
- 16,000Mbps Network
- Ubuntu 18.04
- Hypervisor
- Single Socket (?)

- 32vCPUs (Intel Xeon Platinum 8259CL)
- 128GB RAM
- 10,000Mbps Network
- Ubuntu 18.04
- Hypervisor
- Shared Instance

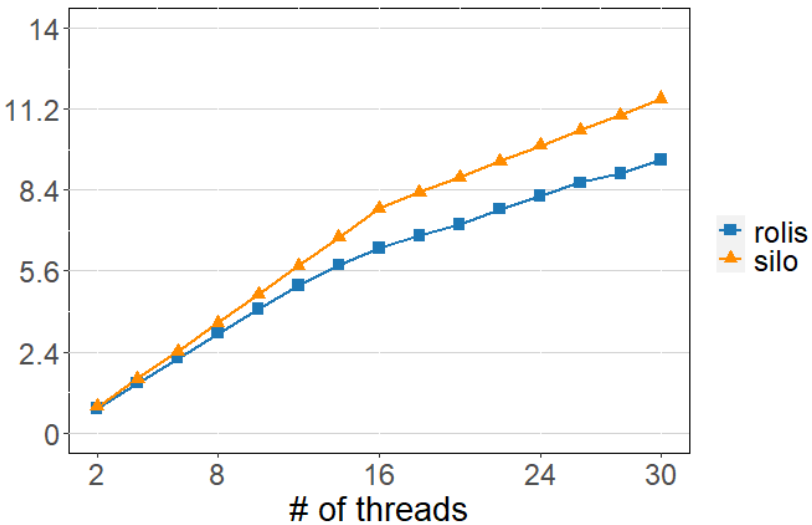
How we ran it

- Virtual Private Cloud
- Security Groups
- Start EC2 instances.
- Setup SSH connections.
- Setup IP addresses (guide given by the paper).
- Run one-click.sh.

Throughput Rolis vs Silo: YCSB++

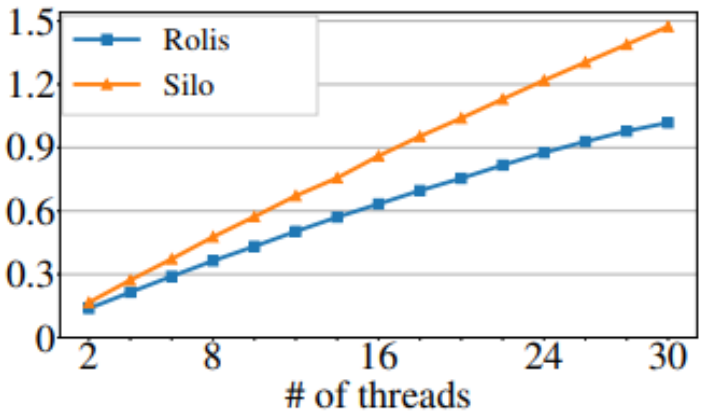


(a) Original

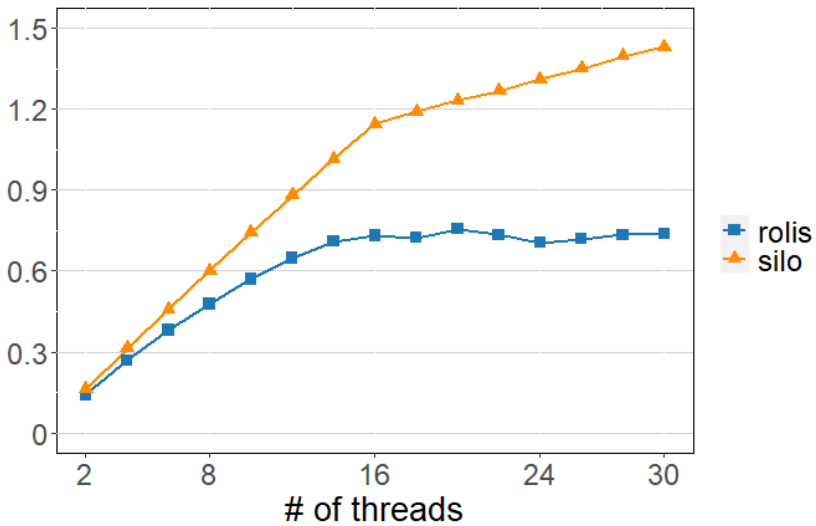


(b) Observed

Throughput: Rolis vs Silo: TPCC

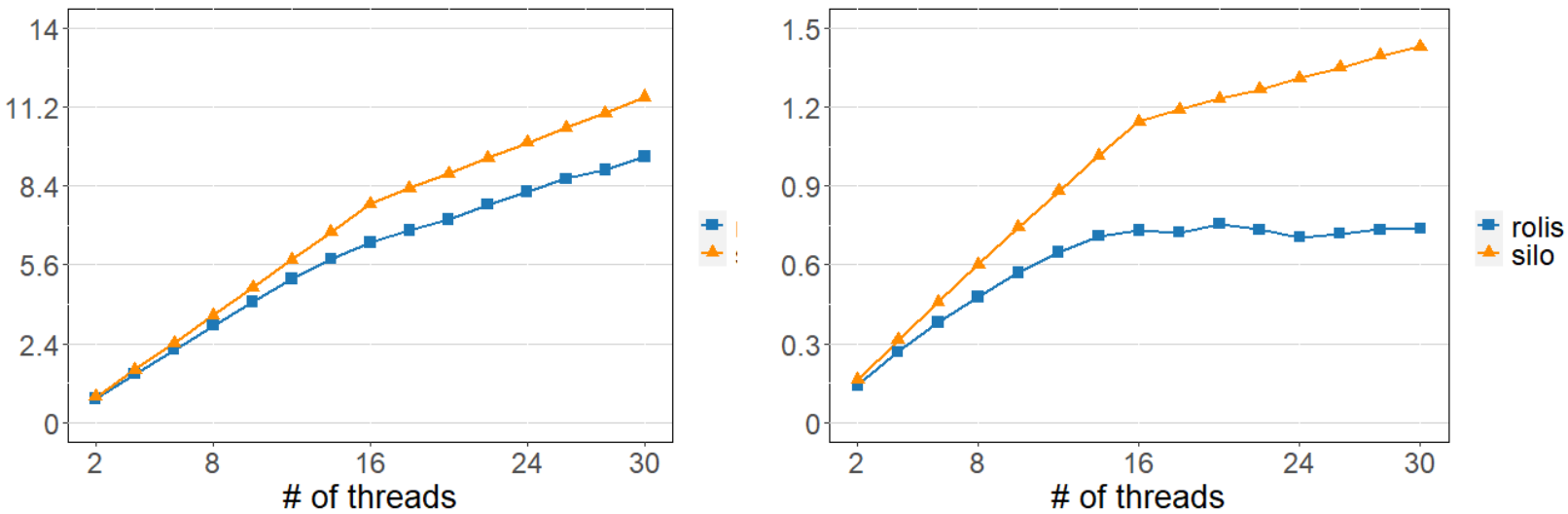


(a) Original



(b) Observed

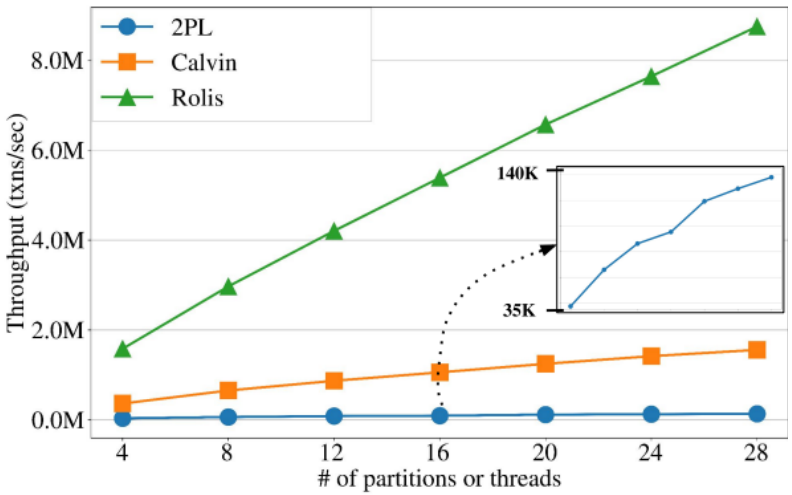
Discuss Observation



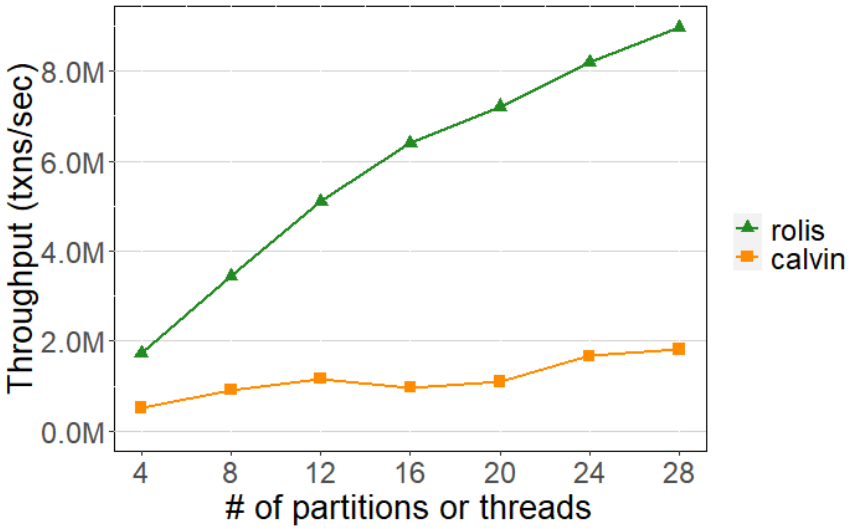
(a) YCSB++ (b) TPC-C

- VM Resource Overcommitment vs Bare Metal Instance
- CPU sockets

Throughput: Rolis vs Calvin



(a) Original



(b) Observed

Discuss

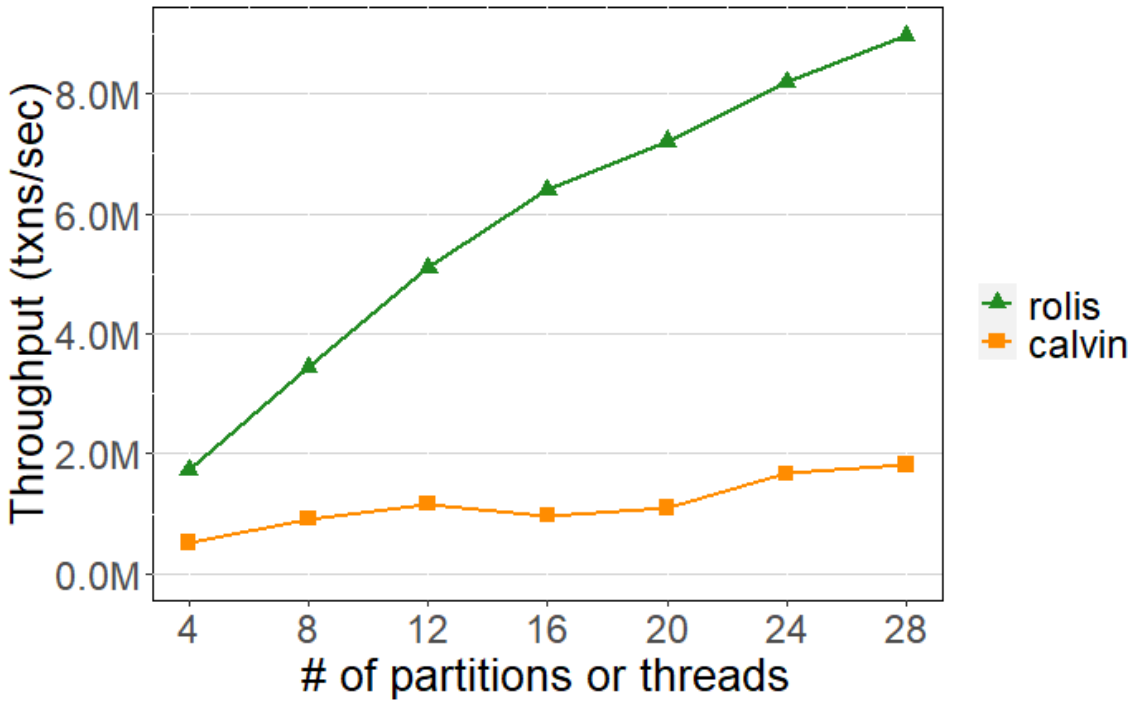
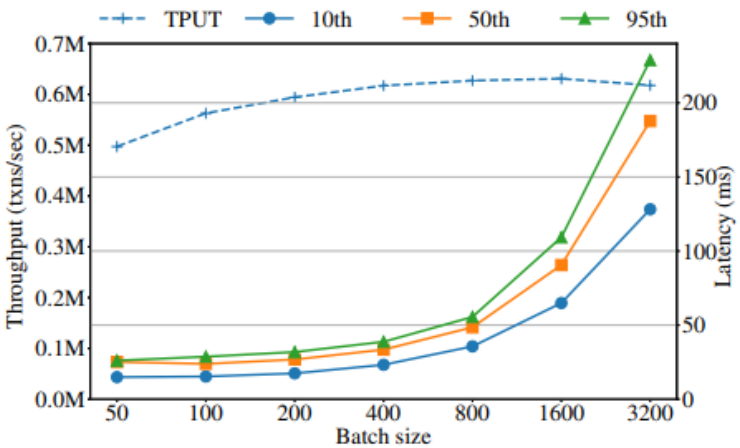


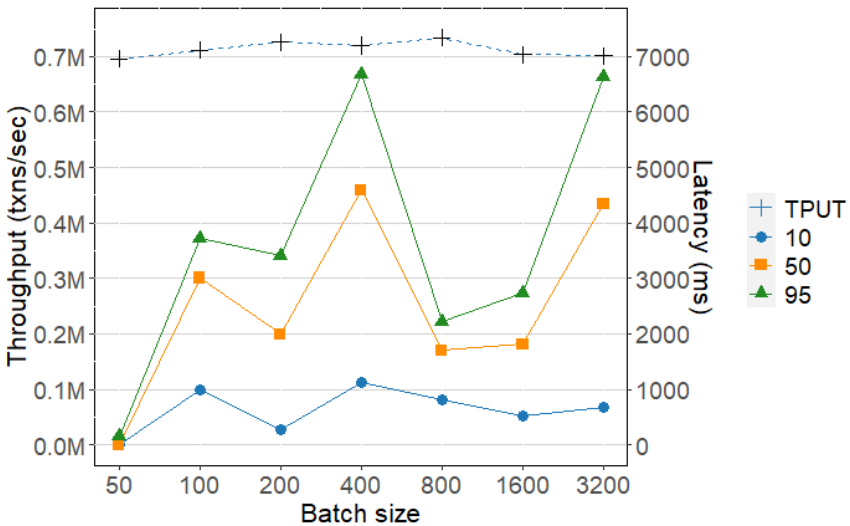
Figure: Observed Throughput of Rolis vs Calvin

- Calvin's thread-implementation vs Rolis.
- CPU Sockets (Calvin experiment needs just one Machine).

Latency: Batch-Size Take 1

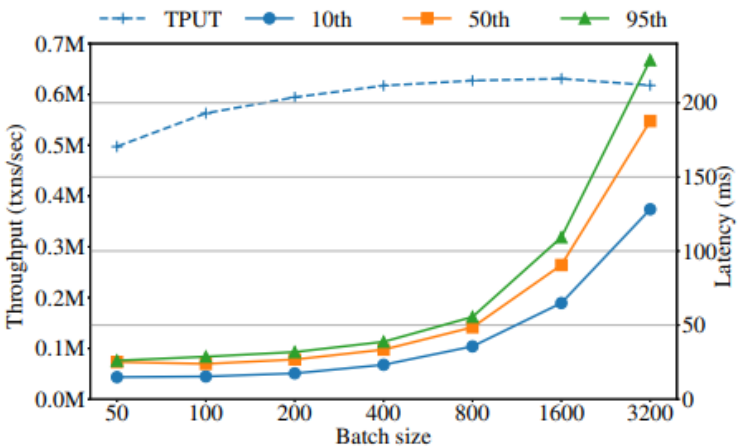


(a) Original (16 threads)

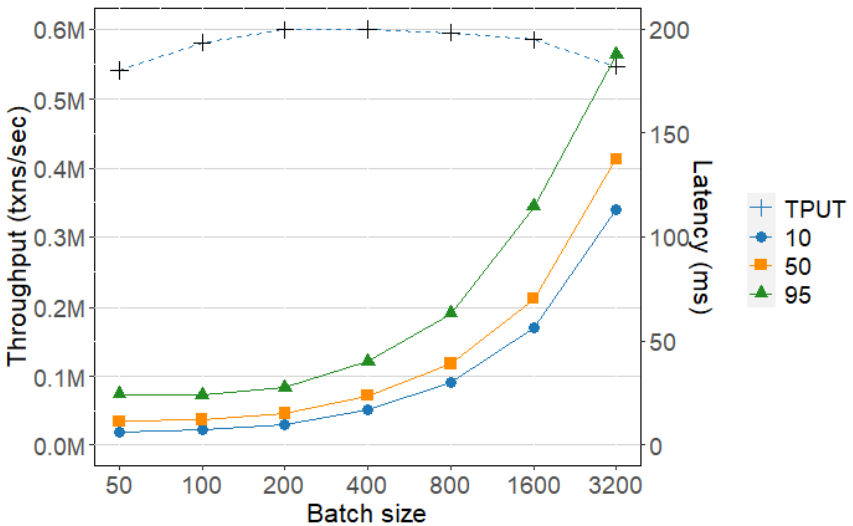


(b) Observed (16 threads)

Latency: Batch-size Take 2

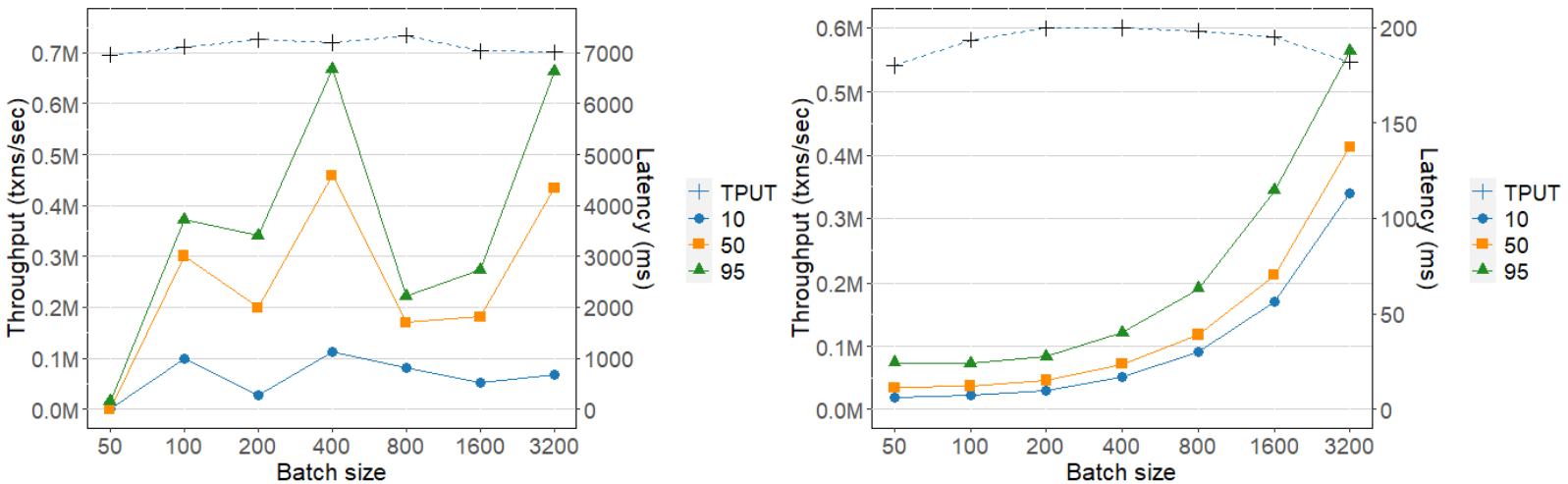


(a) Original (16 threads)



(b) Observed (12 threads)

Discuss



(a) Original (16 threads)

(b) Observed (12 threads)

- Shared Instances - Network Bandwith
- Congestion control.

Thank you

- dkglkhfg
- kfghkfg

Questions?