Overview

ALEX: An Updatable Adaptive Learned Index

Baseline: B+ Tree

Setup

Machines used:

Mimi cluster (teach-node-07, 08)
Intel(R) Xeon(R) Gold 6234 CPU @ 3.30GHz
16GB RAM

Differences:

Single-threaded experiments on an Ubuntu Linux machine
Intel Core i9-9900K **3.6GHz CPU**
64GB RAM.

B+ tree (baseline)

- **Traverses tree using comparisons**
- **Supports OLTP-style mixed workloads:**
  - **reads: Point lookups, range queries**
  - **writes: Inserts, updates, deletes**

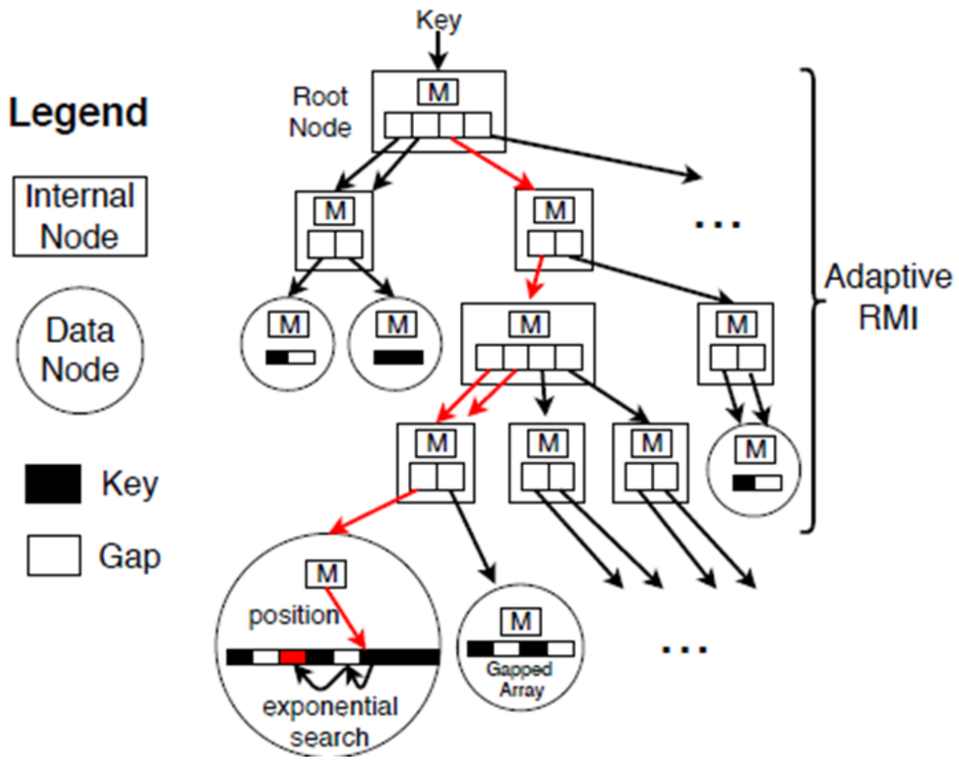
ALEX: **supports read and writes.**

Learned Index

- **Traversing tree using models.**
- **Supports point lookups and range queries**
- **3X faster reads, 10X smaller size**
  - **Limitation: read-only**

ALEX

- Dynamic data structure
- Efficient support for point lookups, short range queries, inserts, updates, deletes, and bulk loading.
- Storage layout optimized for models: Gapped Array **(faster inserts)**
- Exponential search strategy **(faster reads)**



Configs

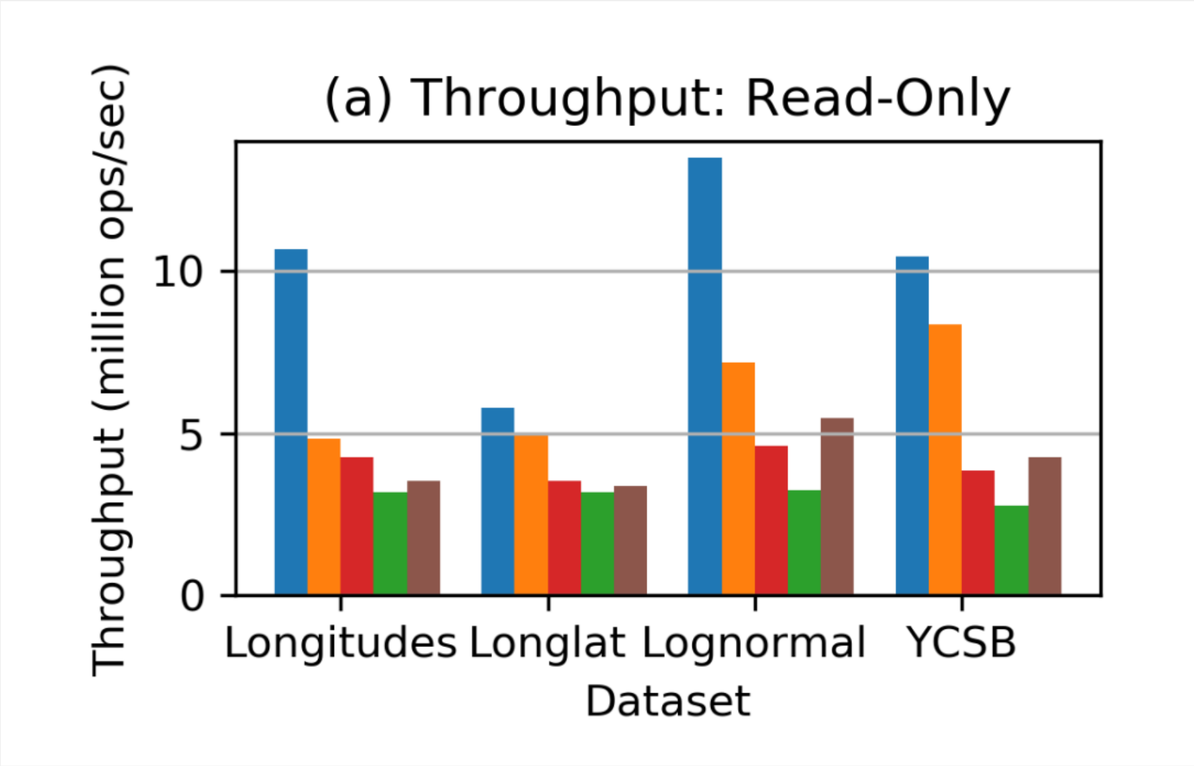
- Longlat 200M total keys , 10M init_num_keys, 1M batch size.
- Longitudes 200M total keys, 10M init_num_keys, 1M batch size.
- Lognormal 190M total keys, 10M init_num_keys, 1M batch size.

- The experiment for YCSB here was with 200M, 10M, 1k.

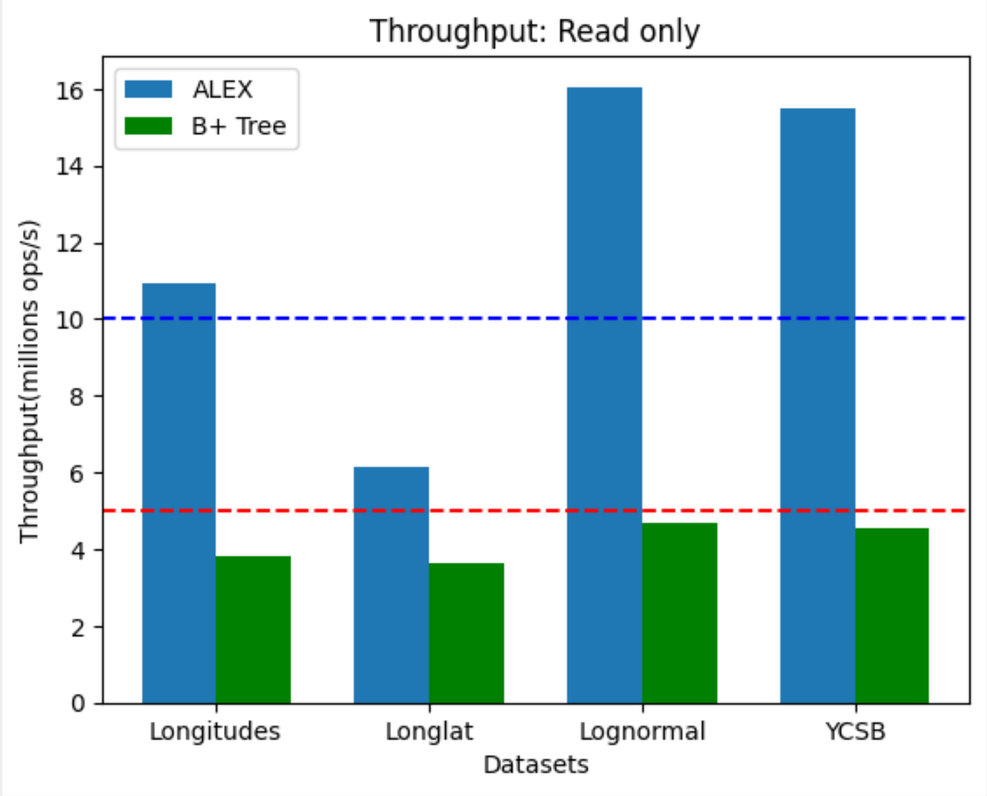
- Read-only
- Read-heavy 5% inserts
- Write-heavy 50% inserts

reads consist of a lookup of a single key

Workloads



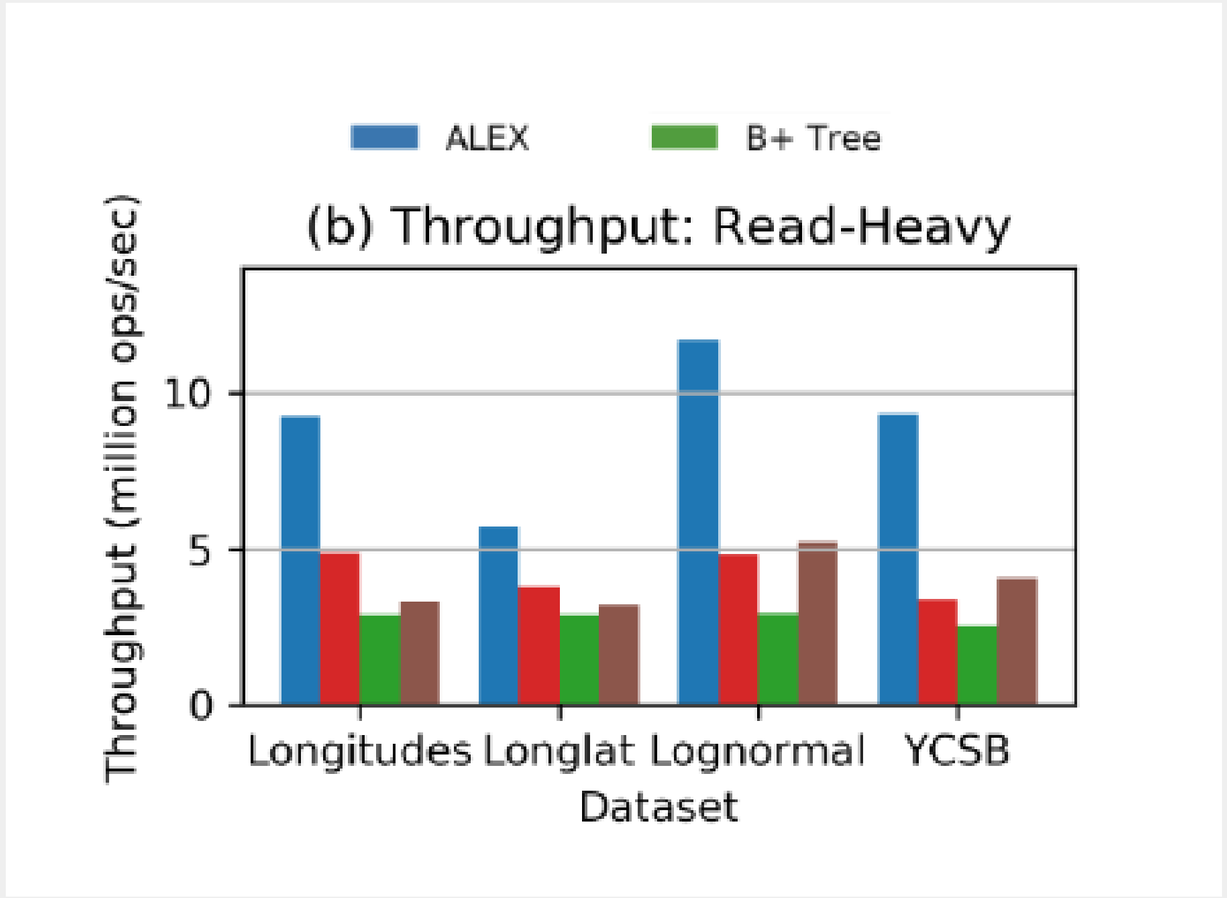
Paper

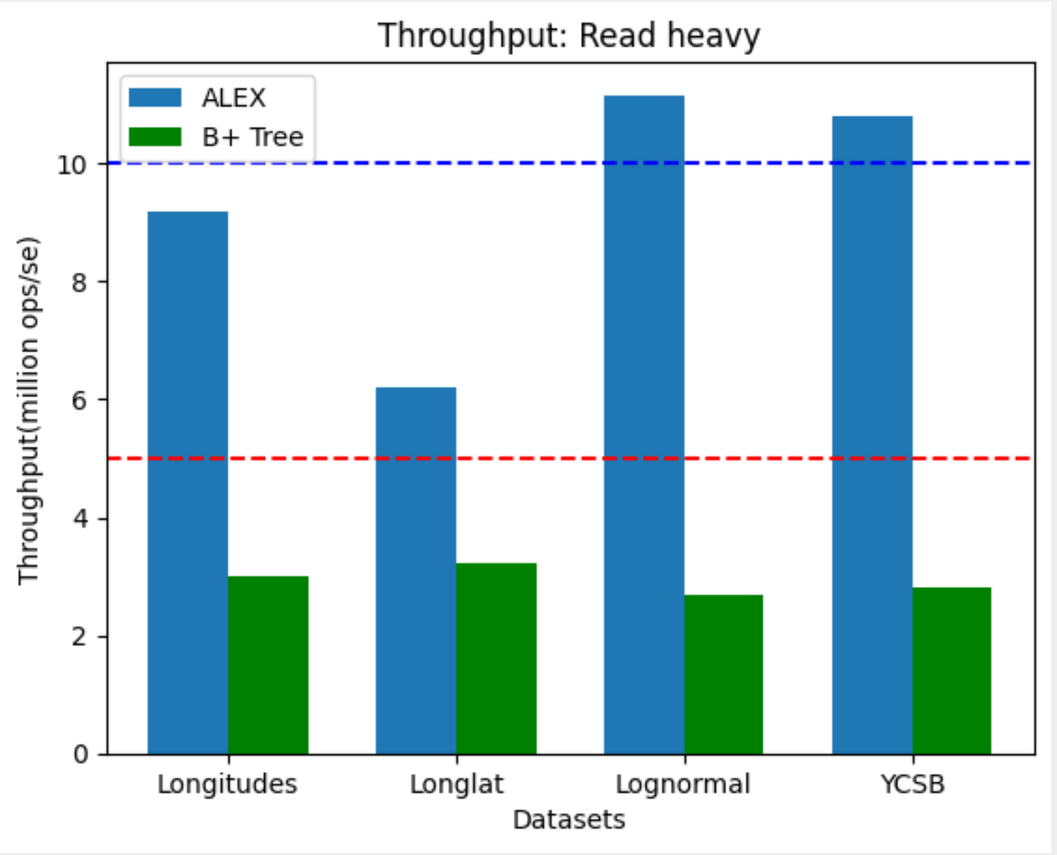


Reproduced

Comments:

ycsb slightly different



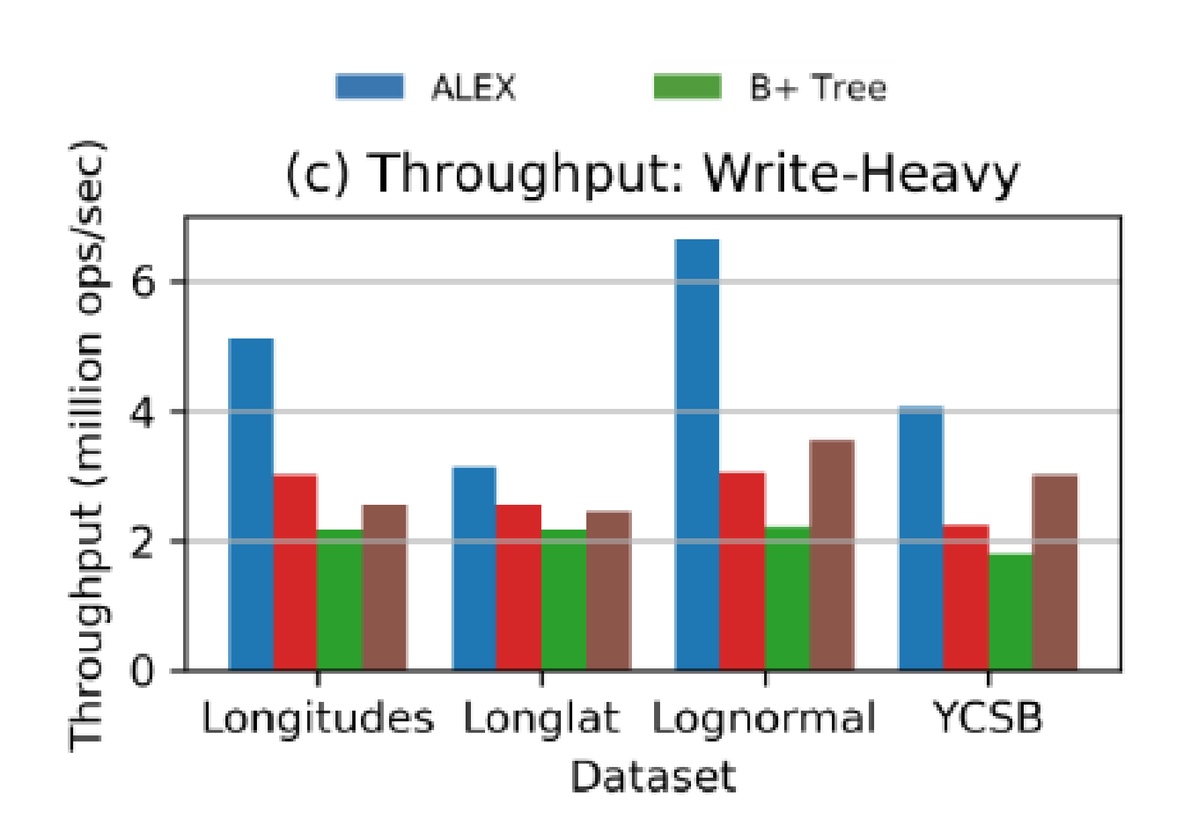


Paper

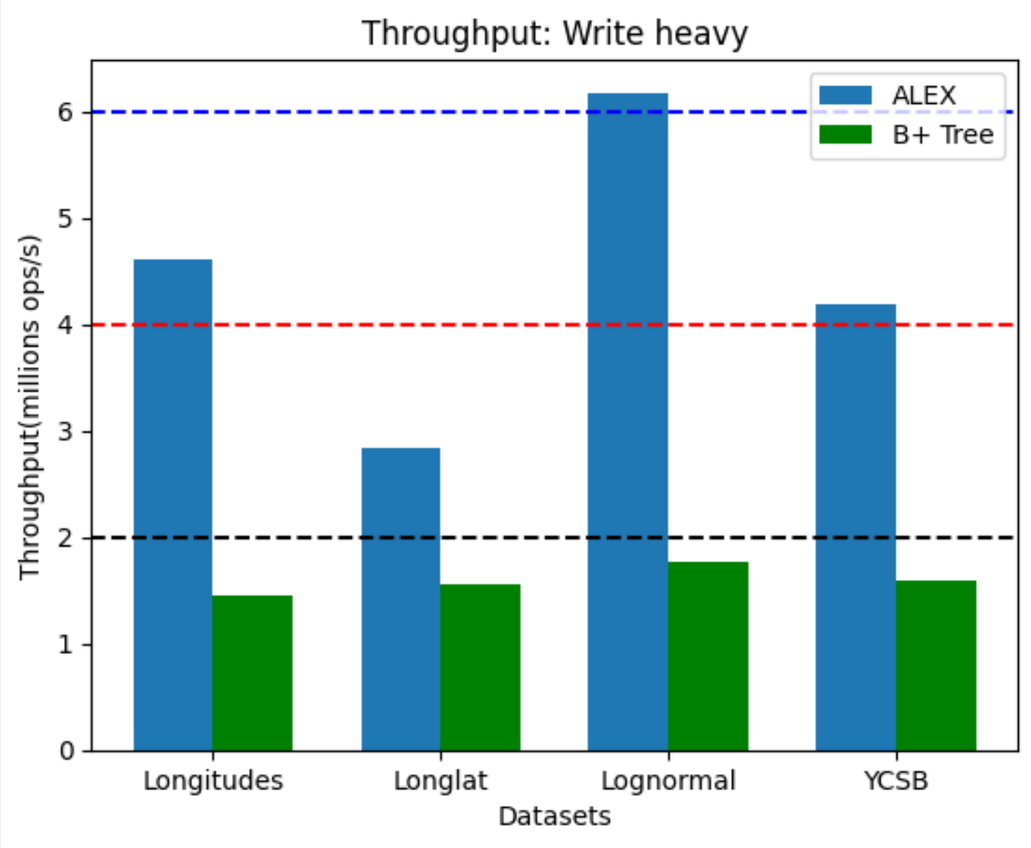
Reproduced

Comments:

ycsb slightly different



Paper



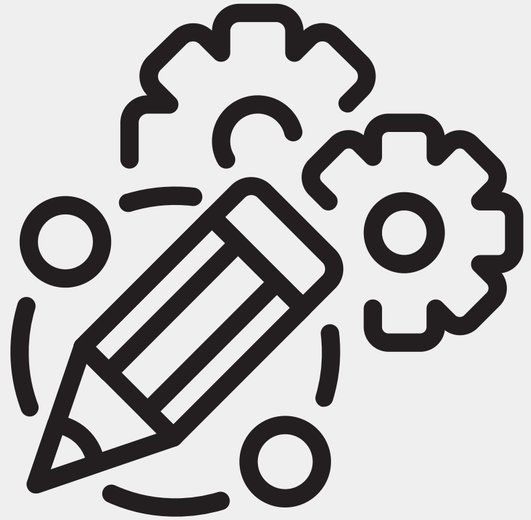
Reproduced

Comments:

Baslelines values slightly lower.

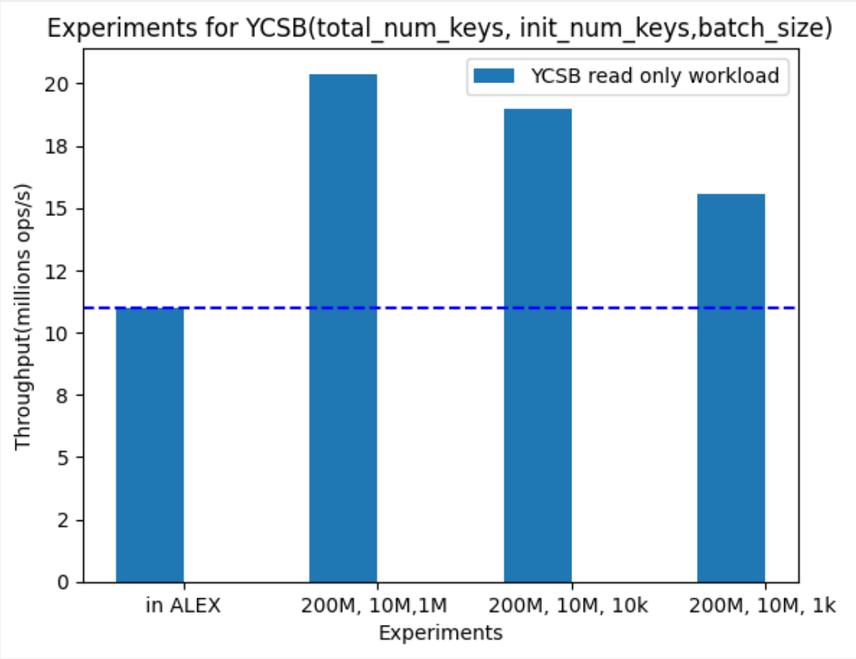
Grid search tuning

Why we see these differences?



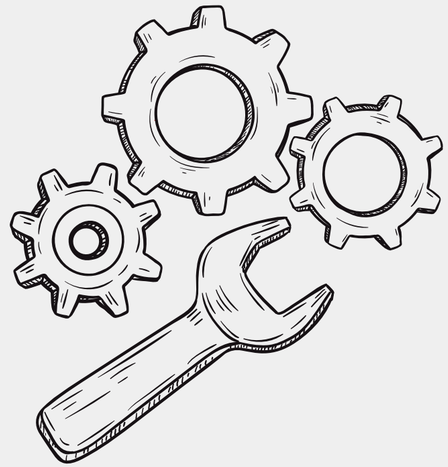
ycsb

- Parameters used in our experiments were less intensive than real workloads.
- The closer we are to real parameters, the closer the measurements become.



- 200m total, 100m init, 1k batch
  - B+ tree ~3million ops/s closer to the paper
  - We were able to get it as confirmation.

Why we see these differences?



Ycsb

- payload: 80 Bytes
- Size of workload - 8Bytes (Instead of 80 bytes)

Data files size discrepancy (paper vs github)

Text files ~ 17GB, binary files ~ 1.5GB

Memory constraint

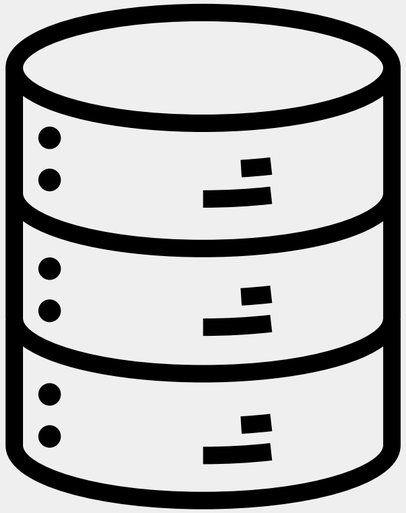
- We calculated: (8 Bytes + 8 Bytes) for 200 mill records -> 3.2GB bare minimum memory (works)
    - 8 + 80 Bytes --> 17.6GB may not work :(

Shared machine

Interference: from other users logged

Network Storage

We were storing our data and scripts on CS accounts (not locally on mimi)
Home directory on network server which gets mounted to mimi.



Project Reflections

Key Challenges

Bulk loading 100M was problematic for ALEX.

Replicating the exact configs the paper used was challenging.
ex: batch size, node size, payload

Surprisingly, the exact implementations used in paper were not directly linked.

End

Thank you

Do you have any questions?