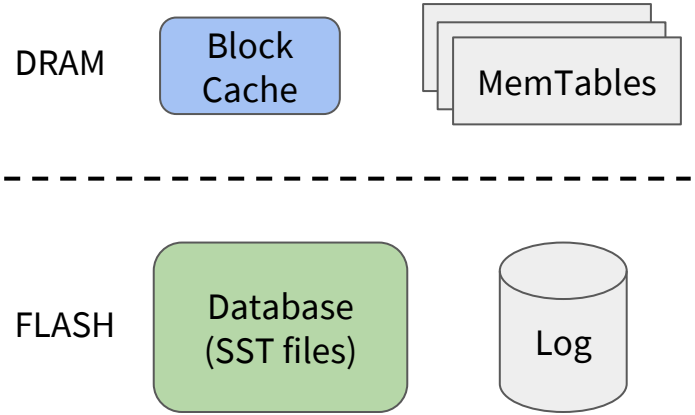
Reducing DRAM Footprint with NVM in Facebook

Presented by: Zhiqiang He

Outline

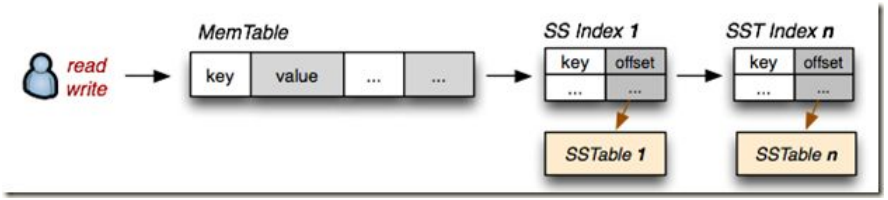
- Background
- Motivation and Goal
- Challenges
- Design and Implementation
- Evaluation
- Conclusions

Background



The Architecture of RocksDB

The Format of a Sorted String Table (SST)



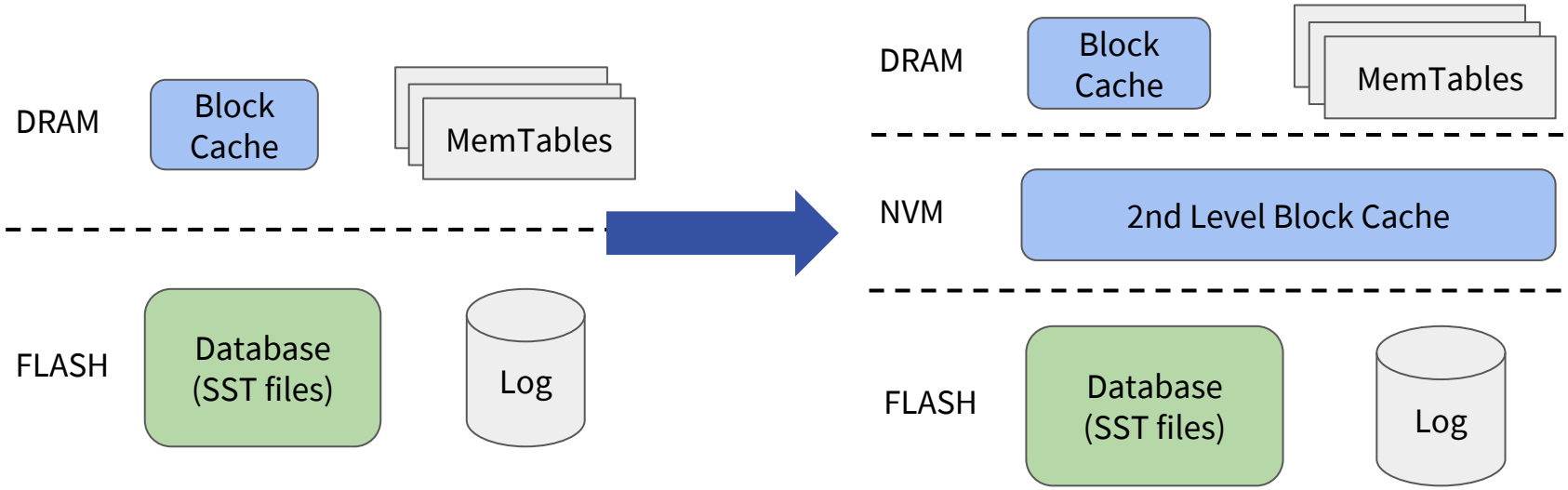
Simplized RW process in LSM KV Store

Motivation

- Modern KV stores consume much DRAM as cache
- TCO of data center increases
  - Since mismatch between DRAM's bit supply and demand
- An NVM block device
  - offers **10x faster** reads than flash
  - has **5x better** write durability than flash
  - has **no write amplification** and **no RW interference** than flash
  - **4x cheaper** than DRAM

Goal: Use NVM to reduce DRAM footprint of MyRocks, while maintaining comparable performance.

Architecture of MyNVM



Primary RocksDB

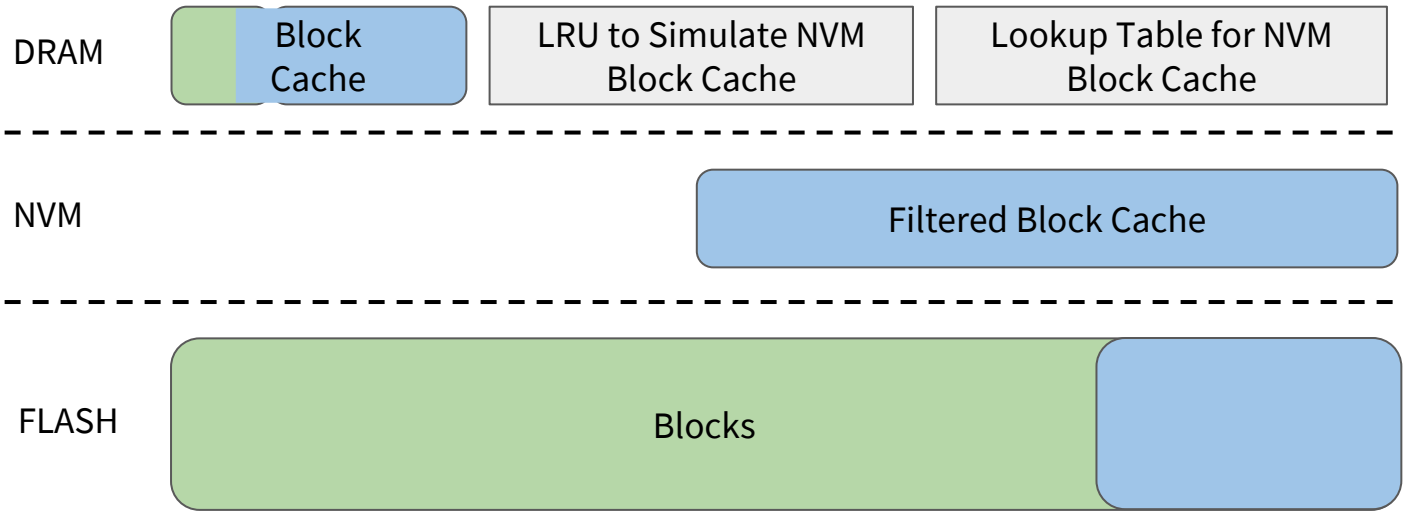
Improved RocksDB

Challenges using NVM as Block Cache

- Limited **write durability**
- **RW latency**: more than 100x slower than DRAM
- **RW bandwidth**: 34x slower than DRAM
- Smaller data blocks reduce compression ratio
- OS interrupt overhead matters

NVM's Durability Constraint

- Admission control to NVM cache
  - **Lower write traffic to NVM**
  - **Higher hit rate** of NVM block cache



Satisfying NVM's Read Bandwidth

- **Smaller data block**: from 16 KB to 4 KB
  - Leads to 4x larger index size
  - Lower hit rate of block cache
  - Inefficient compression ratio of data block
- **Partitioning the database index**
  - Overall read bandwidth **< 2.2 GBps**
  - Comparable hit rate of block cache

Figure 12: Non-partitioned and paritioned index structures. The partitioned scheme uses a top-level index, which points to lower-level indices.

Satisfying NVM's Read Bandwidth

- **Aligning data blocks** with device pages
  - Overall bandwidth **~1.25 GBps**
  - Lower P99 latency

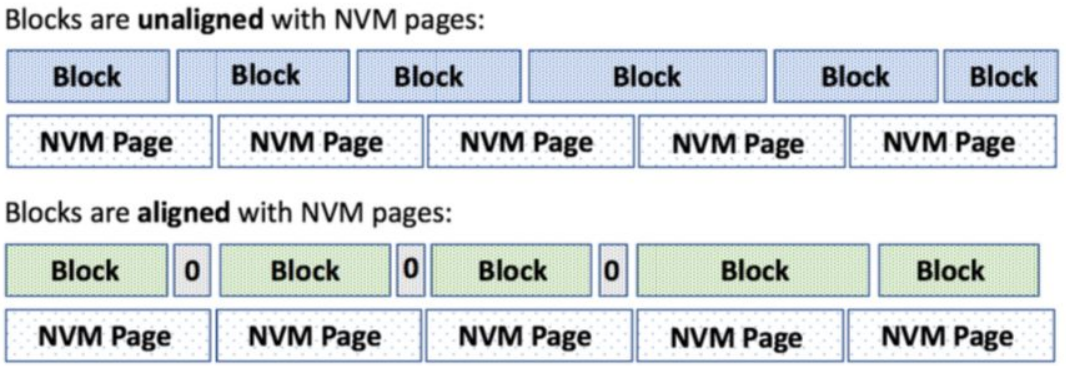


Figure 14: An illustration of blocks that are unaligned with the NVM pages, and of blocks that are aligned with the NVM pages.

Problem on Database Size @RocksDB

- Smaller data blocks have inefficient compression ratio => **larger DB size**
  - Data blocks are compressed by default
  - Do compress block by block
- **Sol: Use uniformly sampled dictionary** SST by SST

Interrupt Overhead Matters

- Polling instead of interrupt
- Hybrid polling
- Dynamic polling
- => **Less CPU I/O Wait**

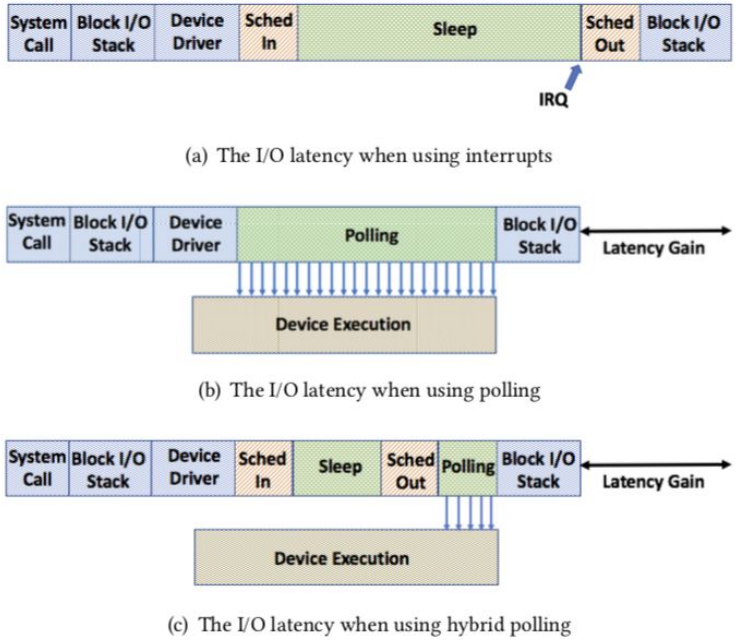


Figure 20: Diagrams of the I/O latencies as perceived by the application, using interrupts, polling, and hybrid polling.

Evaluation

- Configuration
  - MyRocks (96 GB DRAM Cache)
  - MyRocks (16 GB DRAM Cache)
  - MyNVM (16 GB DRAM Cache + 140 GB NVM Cache)
  - Single instance each
- Workload:
  - Traces from DB serving the “social graph” data at Facebook
- Metrics:
  - Overall Mean & P99 Latency
  - Queries-per-Second or QPS
  - CPU Consumption & I/O Wait

Mean & P99 Latency

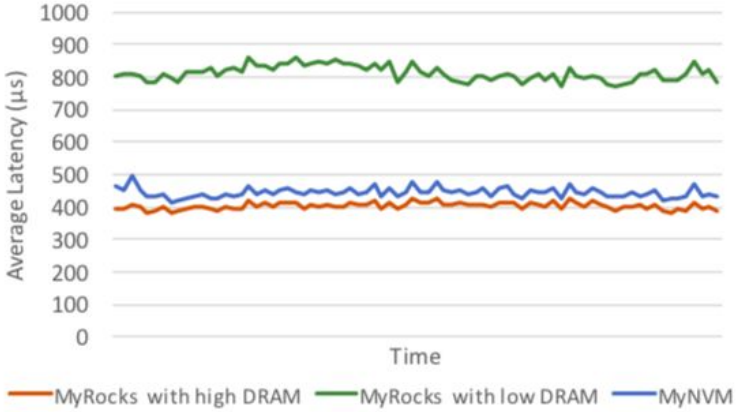


Figure 22: Mean latencies of MyRocks with 96 GB of DRAM cache, compared to MyRocks with 16 GB of DRAM cache, and MyNVM with 16 GB of DRAM cache and 140GB of NVM, over a time period of 24 hours.

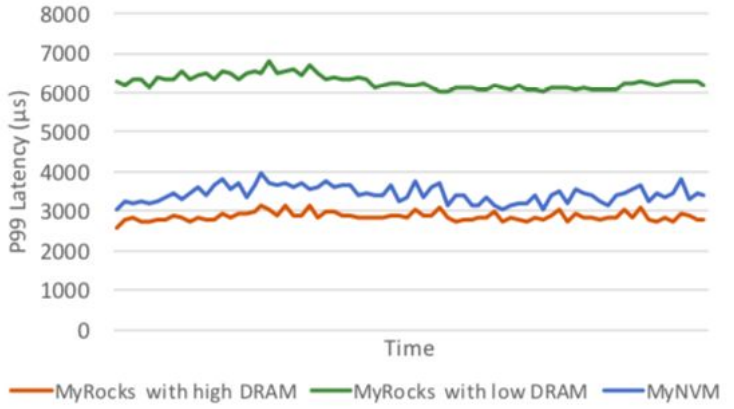


Figure 23: P99 latencies of MyRocks with 96 GB of DRAM cache, compared to MyRocks with 16 GB of DRAM cache, and MyNVM with 16 GB of DRAM cache and 140GB of NVM, over a time period of 24 hours.

Queries-per-Second or QPS

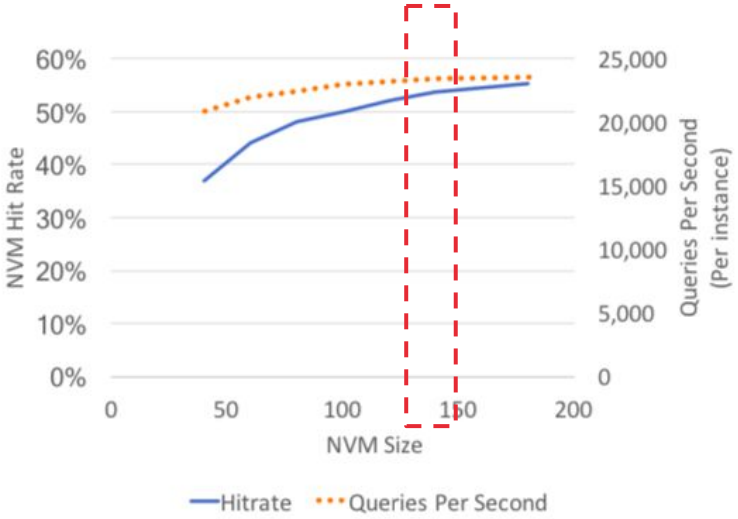


Figure 24: NVM hit rate and QPS in MyNVM as a function of NVM size.

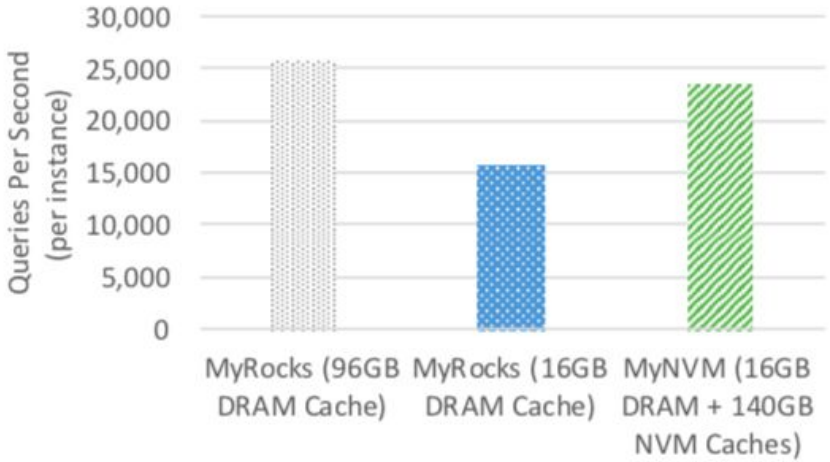


Figure 25: Queries per second (QPS) for different cache sizes in MyRocks, compared with MyNVM.

CPU Consumption & I/O Wait

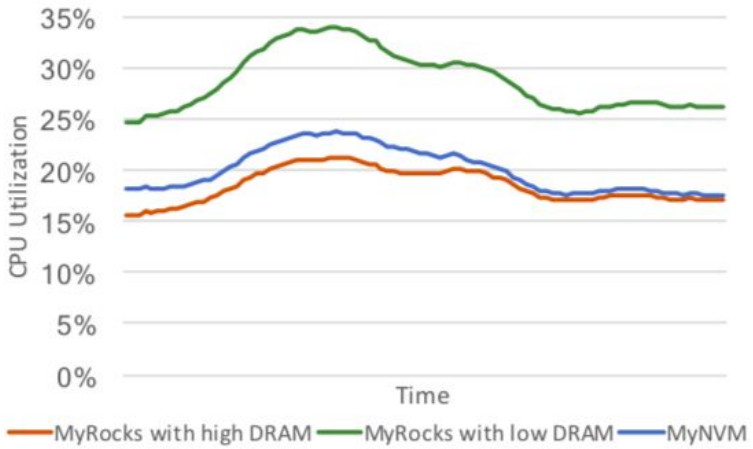


Figure 26: CPU consumption over 24 hours, of MyRocks with 96 GB of DRAM cache, compared to MyRocks with 16 GB of DRAM cache, and MyNVM with 16 GB of DRAM cache and 140 GB of NVM, over a time period of 24 hours.

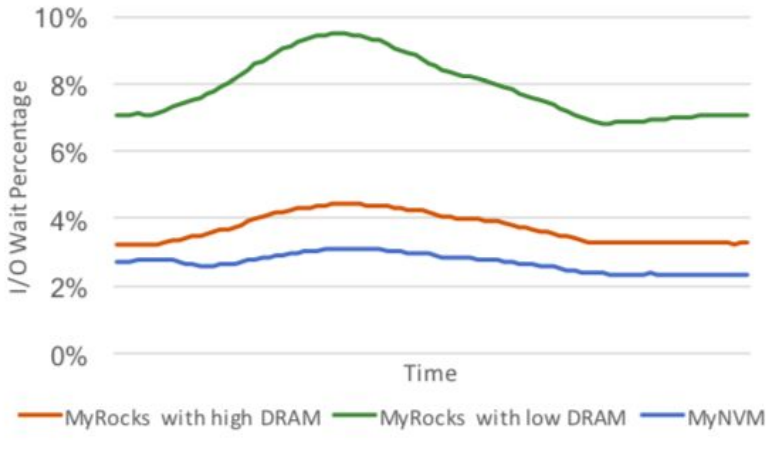


Figure 27: I/O-Wait percentage over 24 hours, of MyRocks with 96 GB of DRAM cache, compared to MyRocks with 16 GB of DRAM cache, and MyNVM with 16 GB of DRAM cache and 140 GB of NVM, over a time period of 24 hours.

Conclusions

- Challenges using NVM as DB cache
  - Limited RW bandwidth / latency
  - Smaller data blocks reduce compression ratio
  - Limited write durability
  - Interrupt overhead matters
- Solutions
  - Partitioning index
  - Aligning blocks with physical NVM pages
  - Utilizing dictionary compression
  - Admission control to NVM
  - Hybrid polling
- Evaluation
  - Maintaining comparable performance with 6x less DRAM by leveraging NVM

Thanks