RocksDB Storage Engine

Igor Canadi | Facebook

DATA PERFORMANCE CONFERENCE – MySQL – NoSQL – DATA IN THE CLOUD

PERCONA
LIVE EUROPE
AMSTERDAM

Overview

- Story of RocksDB
- Architecture
- Performance tuning
- Next steps

Story of RocksDB

Pre-2011

- FB infrastructure – many custom-built key-value stores
- LevelDB released

Experimentation (2011 – 2013)

- First use-cases
- Not designed for server – many bottlenecks, stalls
- Optimization
- New features

Explosion (2013 – 2015)

- Open sourced RocksDB
- Big success within Facebook
- External traction – Linkedin, Yahoo, CockroachDB, …

New Challenges (2015 -)

- Bring RocksDB to databases

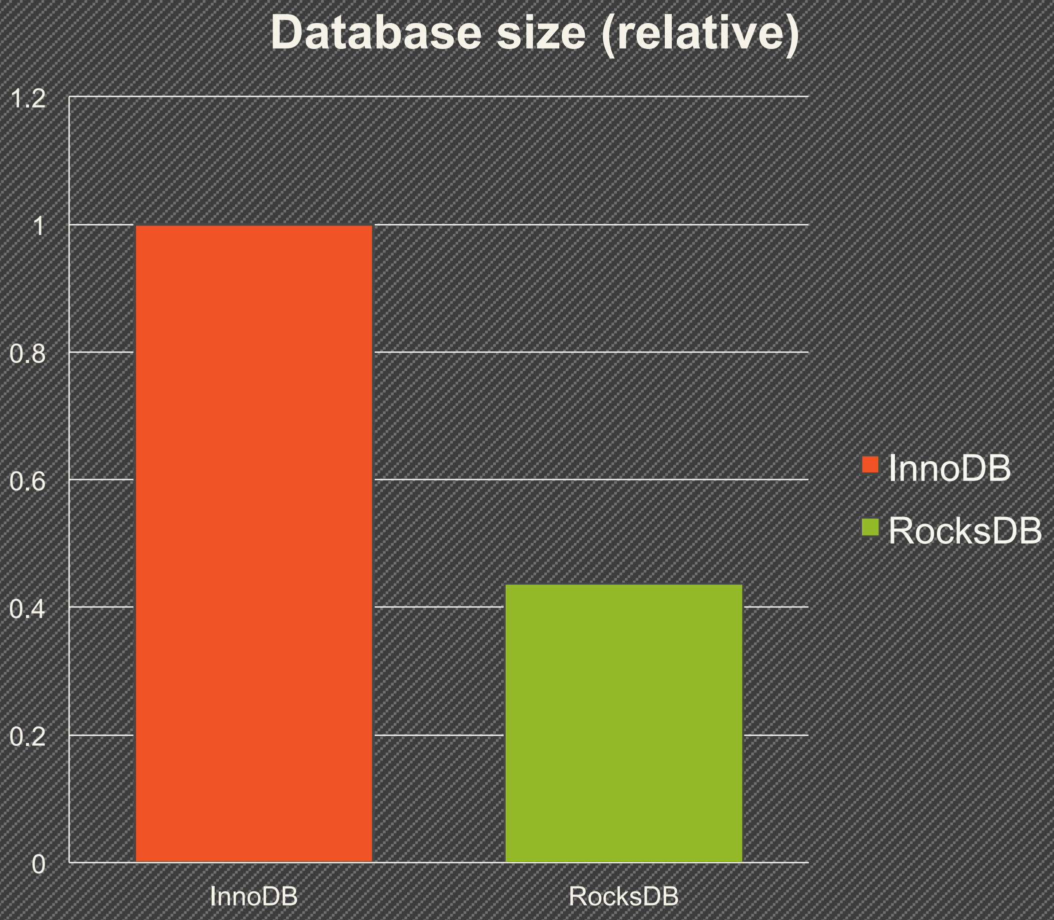
MySQL®

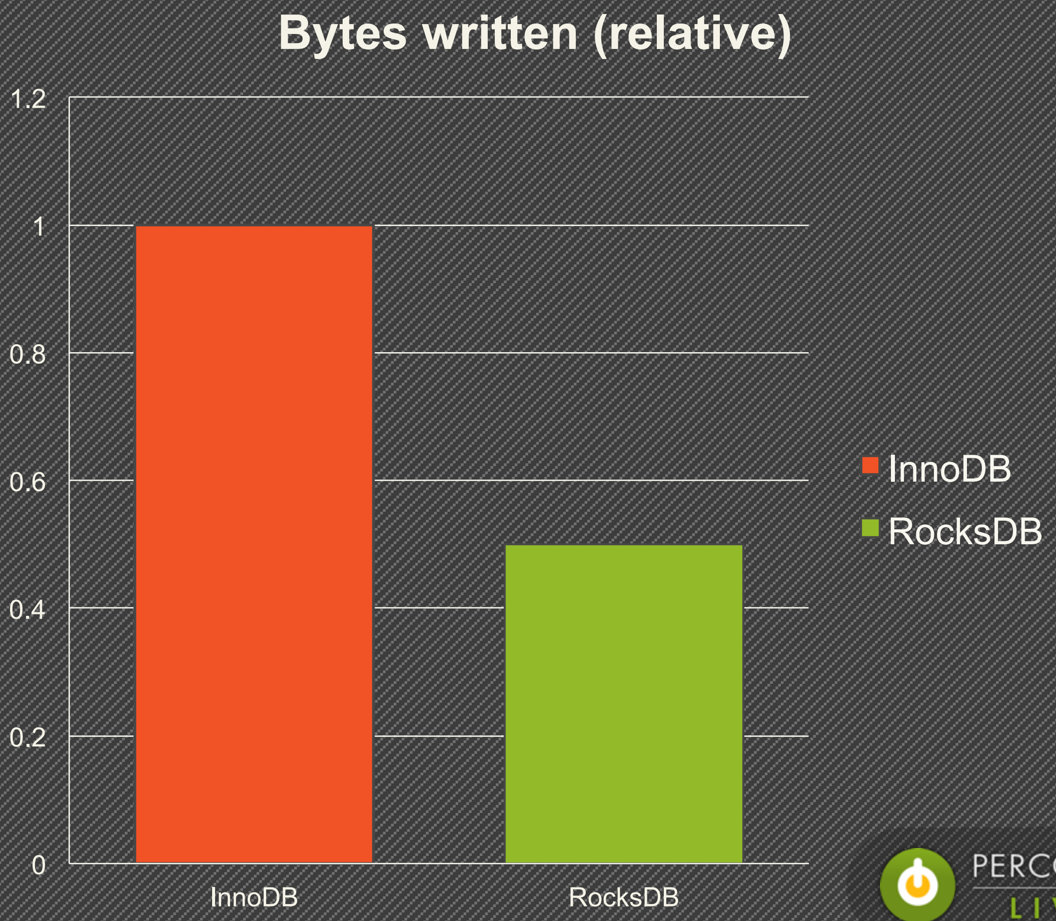
mongoDB

MongoRocks

- Running in production at Parse for 6 months
- Huge storage savings (5TB → 285GB)
- Document-level locking

MyRocks

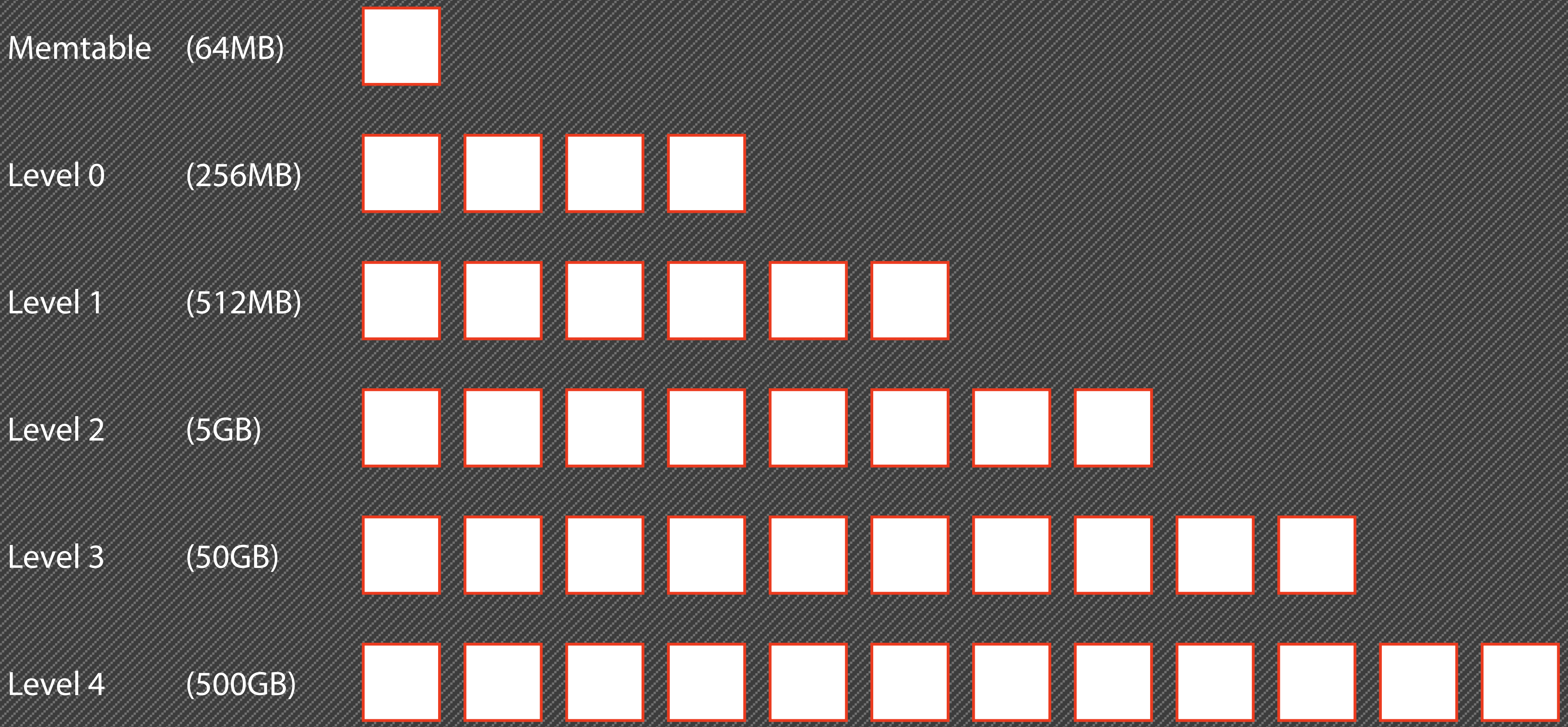

Database size (relative)
1.2
1
0.8
0.6
0.4
0.2
0
InnoDB
RocksDB
🟥 InnoDB
🟩 RocksDB


Bytes written (relative)
1.2
1
0.8
0.6
0.4
0.2
0
InnoDB
RocksDB
🟥 InnoDB
🟩 RocksDB

Architecture

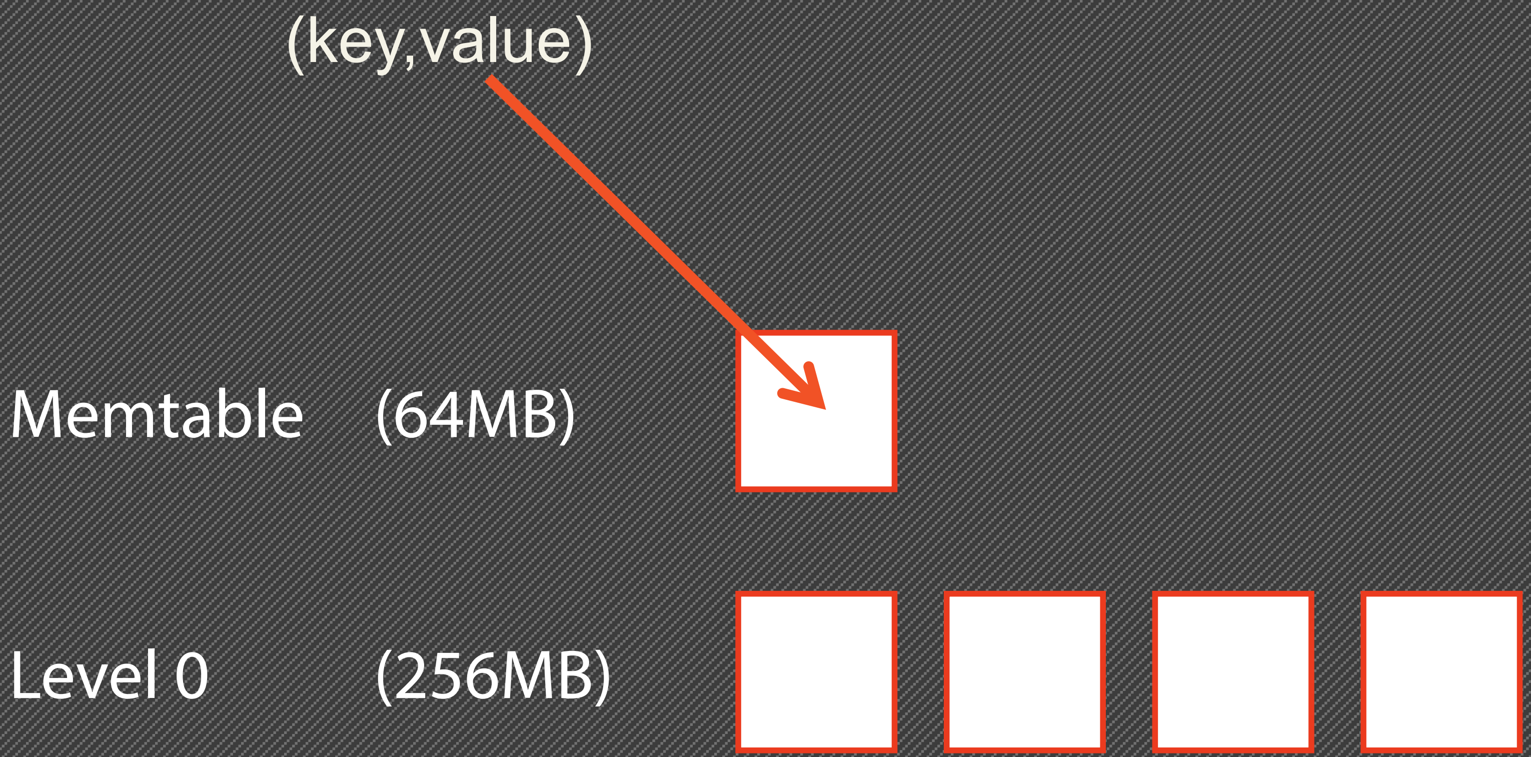
Log Structured Merge Trees

Log Structured Merge Trees



Memtable	(64MB)
Level 0	(256MB)
Level 1	(512MB)
Level 2	(5GB)
Level 3	(50GB)
Level 4	(500GB)

Log Structured Merge Trees – write



(key,value)

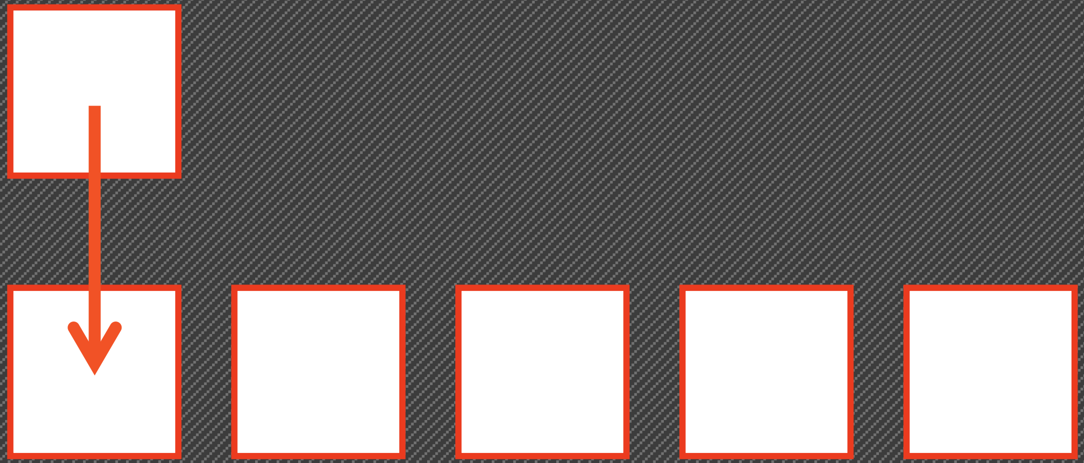
Memtable (64MB)

Level 0 (256MB)

Log Structured Merge Trees – flush

Memtable (64MB)

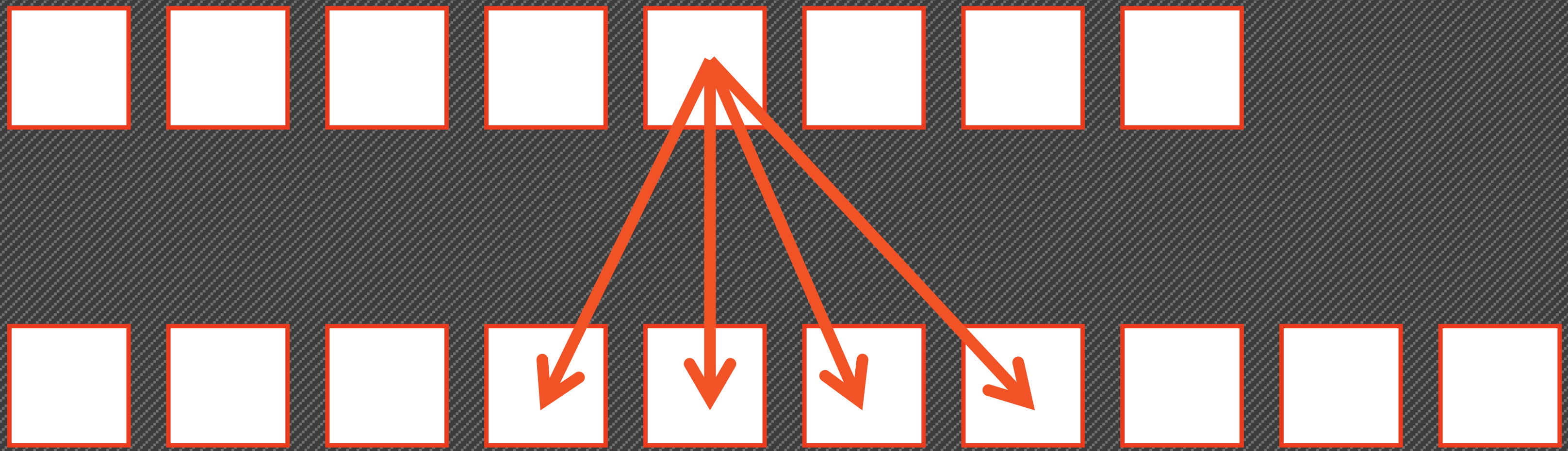
Level 0 (256MB)



Log Structured Merge Trees – compaction

Level 2 (5GB)

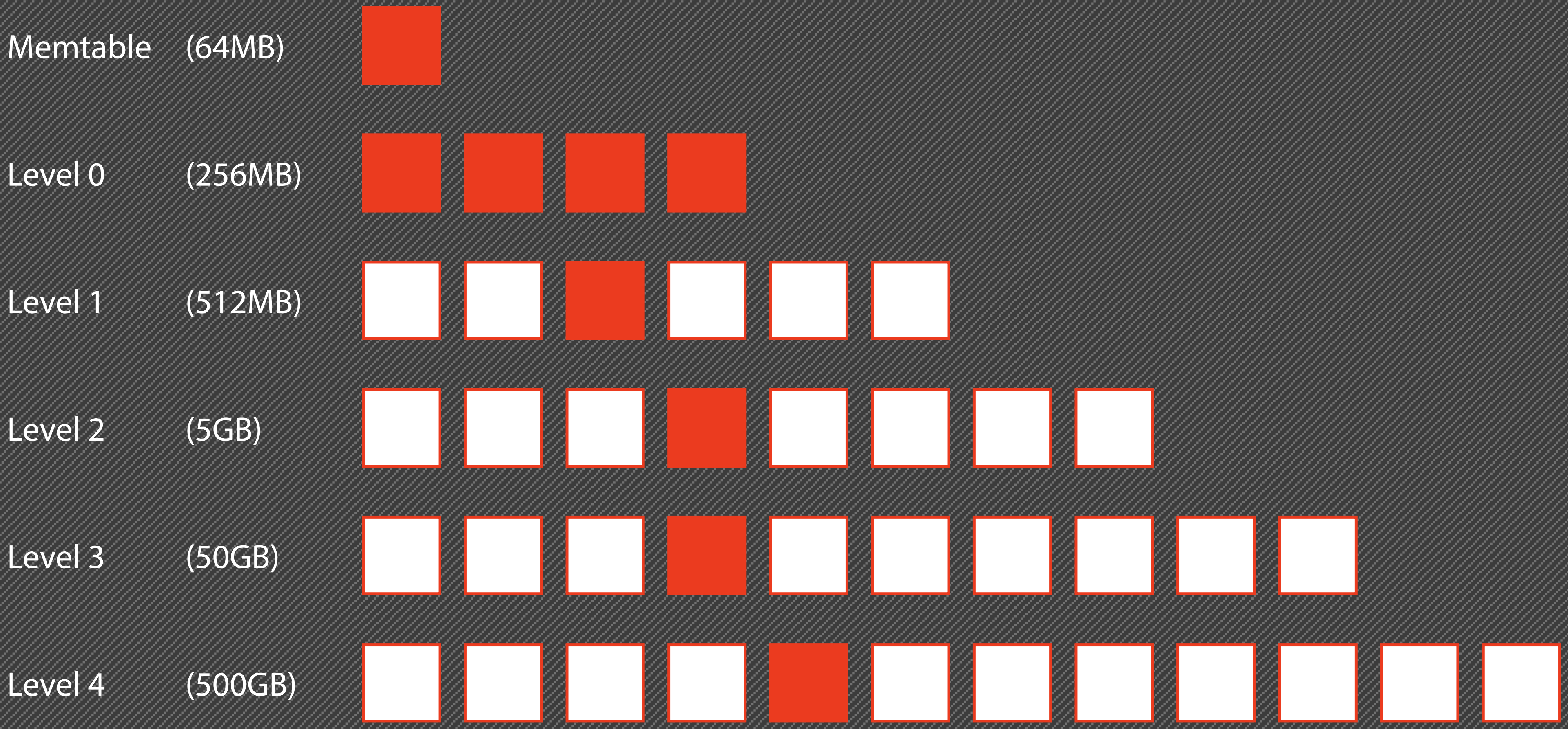
Level 3 (50GB)



Writes

- Foreground:
  - Writes go to memtable (skiplist) + write-ahead log
- Background:
  - When memtable is full, we flush to Level 0
  - When a level is full, we run compaction

Reads



Memtable (64MB)

Level 0 (256MB)

Level 1 (512MB)

Level 2 (5GB)

Level 3 (50GB)

Level 4 (500GB)

Reads

- Point queries
  - Bloom filters reduce reads from storage
  - Usually only 1 read IO
- Range scans
  - Bloom filters don’t help
  - Depends on amount of memory, 1-2 IO

RocksDB Files

```
rocksdb/> ls
MANIFEST-000032
000024.log
000031.log
000025.sst
000028.sst
000029.sst
000033.sst
000034.sst
LOG
LOG.old.1441234029851978
...
```

RocksDB Files – MANIFEST

- Atomical updates to database metadata

(initial state) Add file 1 Add file 2 Add file 3 Add file 4 …	(flush) Add file 9 Mark log 6 persisted	(compaction) Add file 10 Add file 11 Remove file 9 Remove file 8	Add new column family “system”

RocksDB Files – Write-ahead log

- Persisted memtable state

Write (A, B)	Write (C, D)
	Write (E, F)
		Delete(A)	Write(X, Y)
			Delete(C)

RocksDB Files – Table files

(Data block) • compressed • prefix encoded	(Data block) <key, value>	(Data block)	(Data block)
(Data block)	(Data block)	(Data block)	(Data block)
(Index block) <key, block>	(Filter block)	(Statistics)	(Meta index block) Pointers to blocks

RocksDB Files – LOG files

- Debugging output
- Tuning options
- Information about flushes and compactions
- Performance statistics

Backups

- Table files are immutable
- Other files are append-only
- Easy and fast incremental backups
- Open sourced Rocks-Strata

Performance tuning

Tombstones

- Deletions are deferred
- May cause higher P99 latencies
- Be careful with pathological workloads, e.g. queues

Caching

Block cache	Page cache
• Managed by RocksDB • Uncompressed data • Defaults to 1/3 of RAM	• Managed by kernel • Compressed data

Memory usage

- Block cache
- Index and filter blocks (0.5 – 2% of the database)
- Memtables
- Blocks pinned by iterators

Reduce memory usage

- Reduce block cache size – will increase CPU
- Increase block size – decrease index size
- Turn off bloom filters on bottom level

Reduce CPU

- Profile the CPU usage
- Increase block cache size – will increase memory usage
- Turn off compression
- It might be tombstones

Reduce write amplification

- Write amplification = 5 * num_levels
- Increase memtable and level 1 size
- Stronger (zlib, zstd) compression for bottom levels
- Try universal compaction

Next steps

Next steps

- Increase performance & stability
- Deploy MyRocks at Facebook
- External adoption of MyRocks and MongoRocks
- Build an ecosystem

Thank you

DATA PERFORMANCE CONFERENCE – MySQL – NoSQL – DATA IN THE CLOUD

PERCONA
LIVE EUROPE
AMSTERDAM