

RocksDB Festival

Supported by IITP, StarLab.

July 26, 2021

송인호, 한예진

inhoinno@dankook.ac.kr , hbb97225@naver.com

TeamName : 멘탈모델을 만들고 싶어요

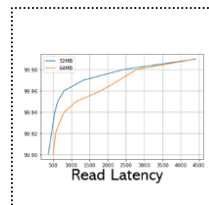
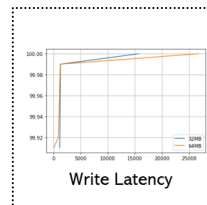
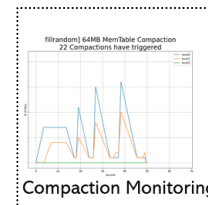
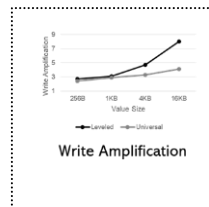
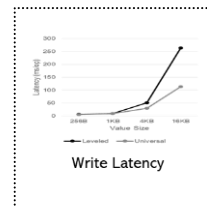
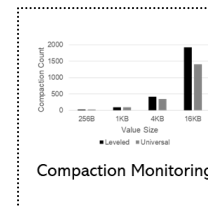
Contents

■ Last Week

- ✓ 컴팩션에 영향을 미치는 녀석들 #1 KV Size
- ✓ 컴팩션에 영향을 미치는 녀석들 #2 SST Size

■ This Week

- ✓ Mental Model
 - Alleviating *[PROBLEMS]* of Level Compaction by *[Method]* in Universal Compaction
- ✓ Leveled vs Universal Compaction Comparison
 - Throughput
 - Write Amplification
 - Latency distribution + # of Compactions
 - ~~Read/Space Amplification~~

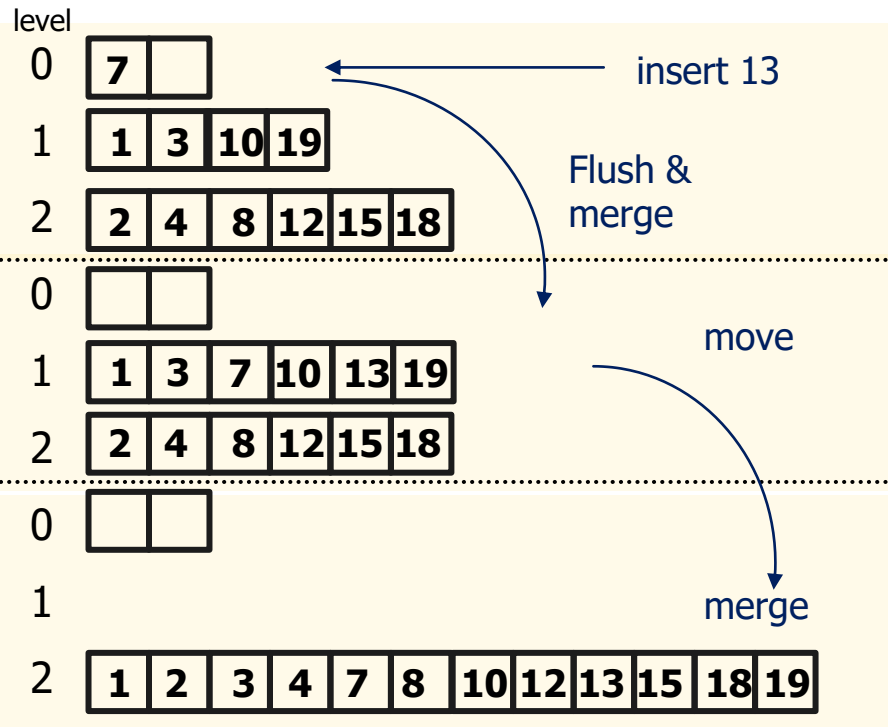


Compaction Style

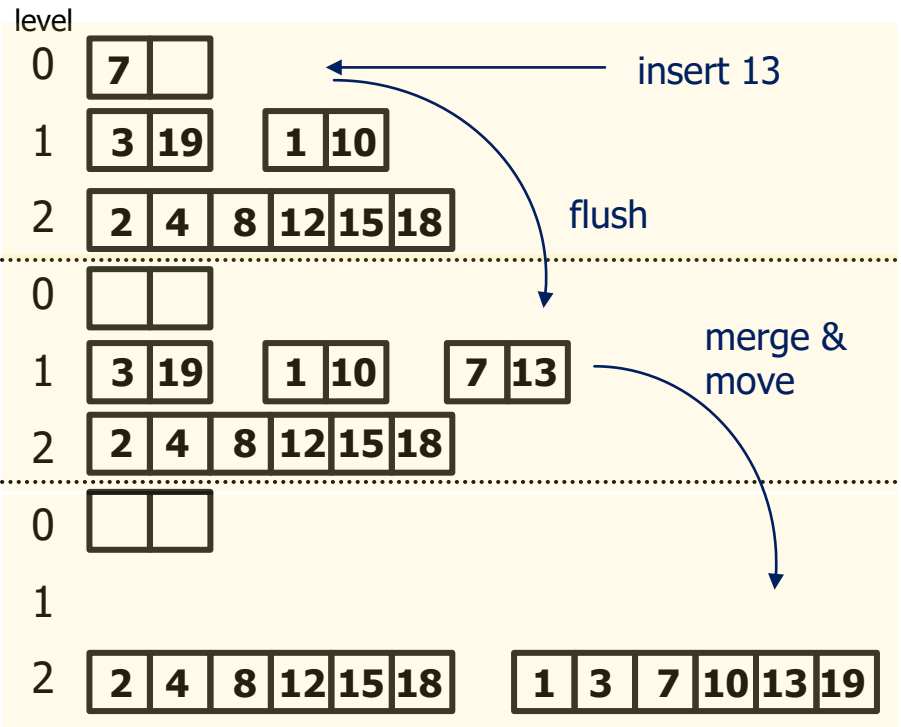
■ Leveled Compaction, Universal Compaction

✓ Sorted Level vs Sorted Run

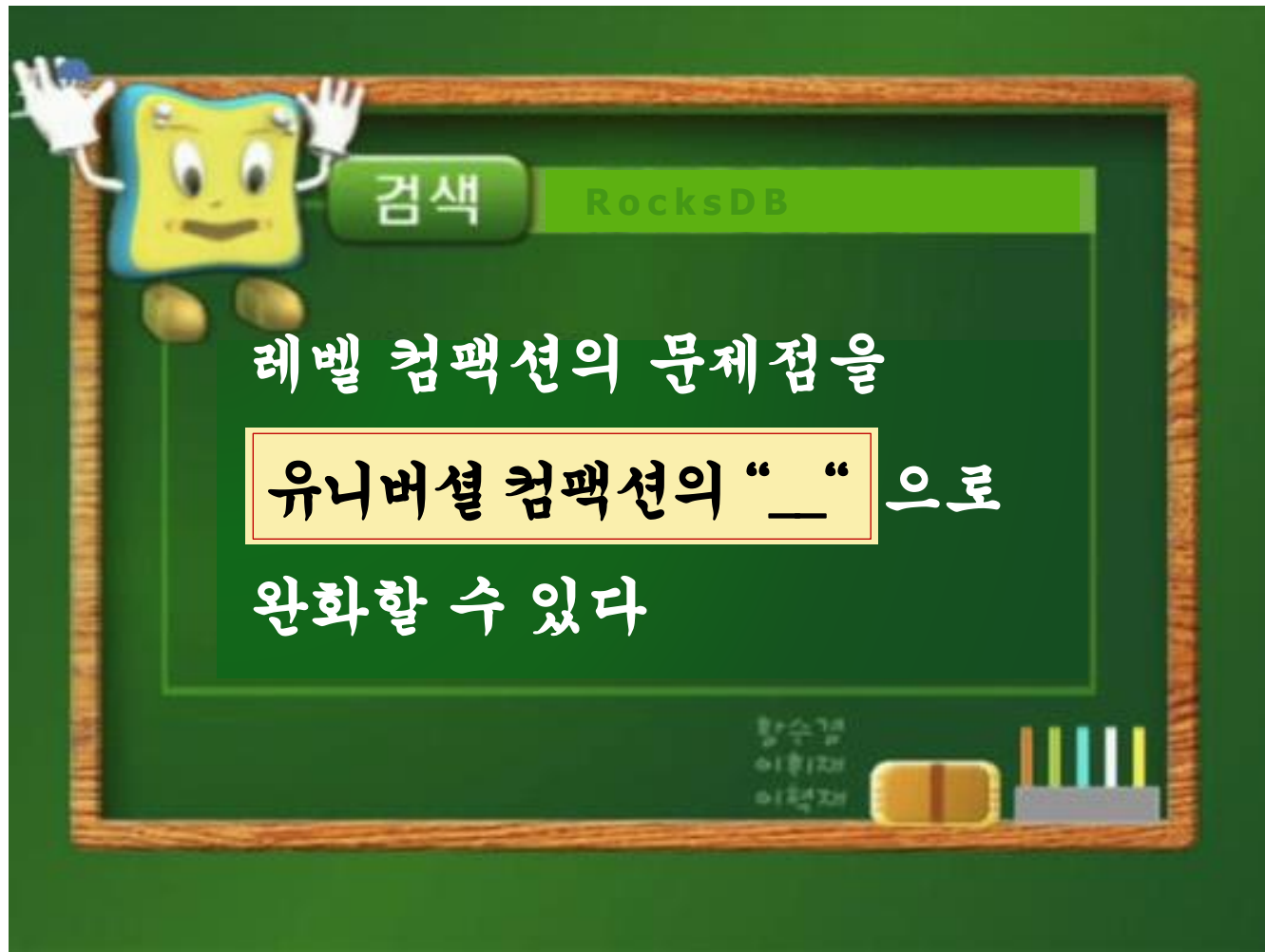
Example of LeveledCompaction



Example of Universal Compaction



Mental Model



LVL vs Univ Throughput Comparison

Fillrandom

Key [16, 32, 64, 128, 256, 1024]

Value [64, 128, 256, 512, 1024, 4096]

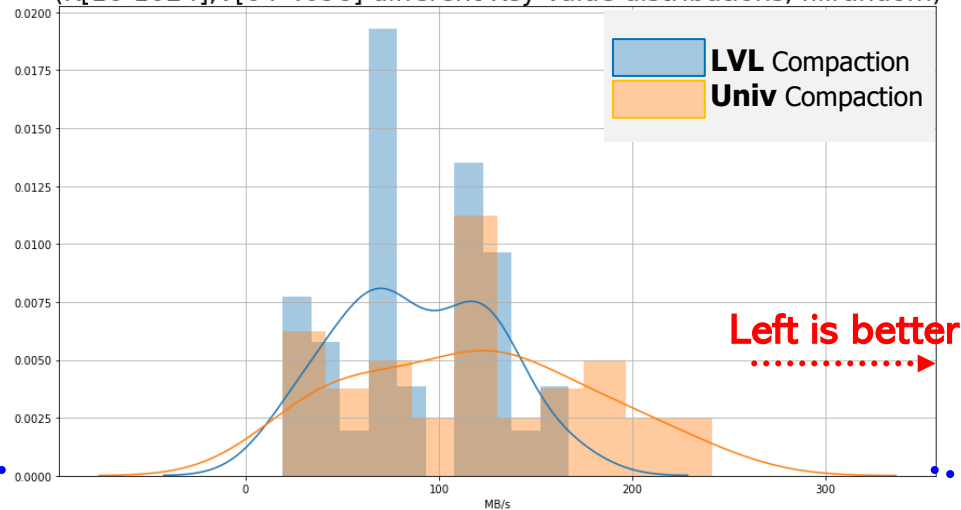
Entries 500 0000

Storage Samsung 1TB 860 Pro

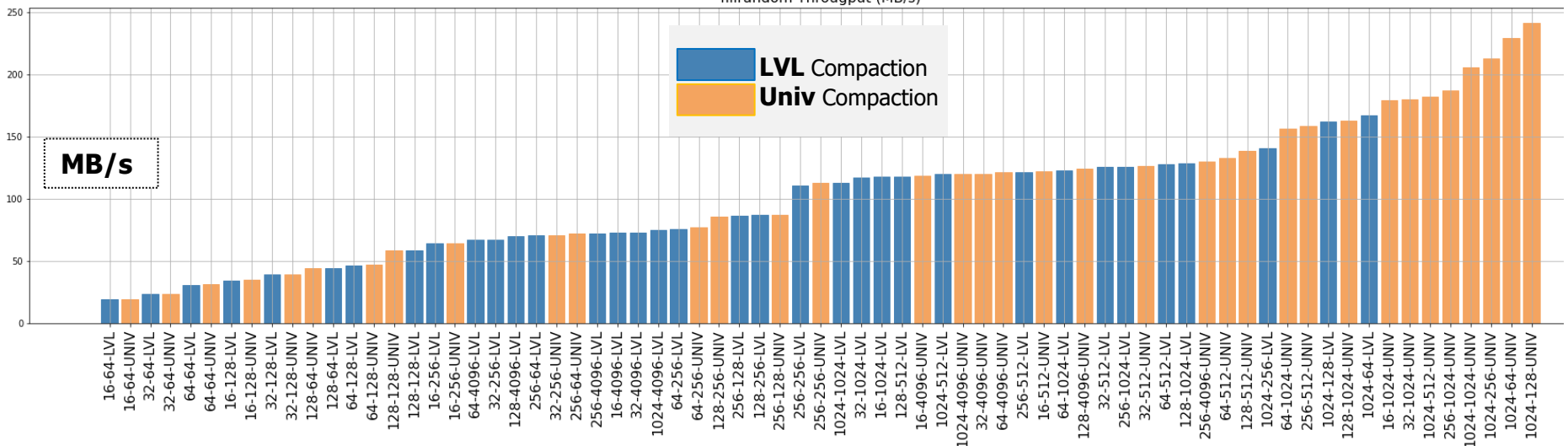
File System Ext4

CPU Intel(R) Core(TM) i7-10700K CPU @ 3.80GHz

Throughput comparison
between different compaction style
(K[16-1024],V[64-4096] different Key-Value distributions, fillrandom)



fillrandom Throughput (MB/s)



MB/s

LVL Compaction
Univ Compaction

LVL vs Univ Throughput Comparison

Throughput comparison
between different compaction style

K[16-1024], V[64-4096] different Key-Value distributions(readrandom)

Readrandom

--use_existing_db

Key [16, 32, 64, 128, 256, 1024]

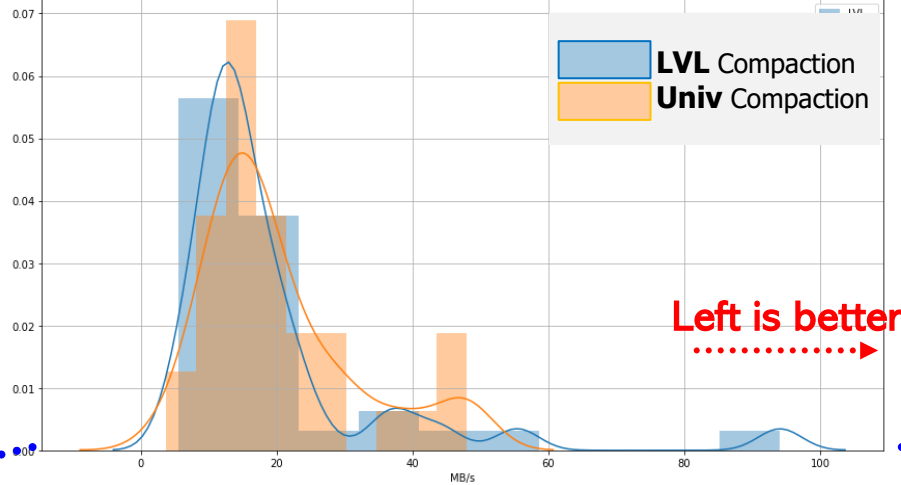
Value [64, 128, 256, 512, 1024, 4096]

Entries 500 0000

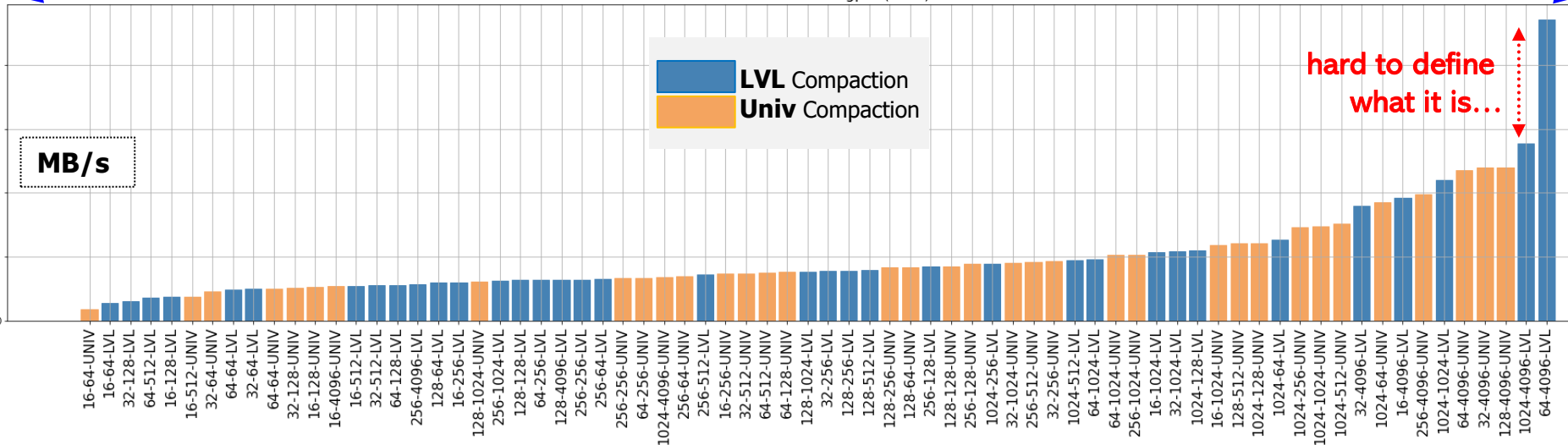
Storage Samsung 1TB 860 Pro

File System Ext4

CPU Intel(R) Core(TM) i7-10700K CPU @ 3.80GHz



Readrandom Throughput (MB/s)



MB/s

16-64-UNIV, 16-64-LVL, 32-128-LVL, 64-512-LVL, 16-128-LVL, 16-512-UNIV, 32-64-LVL, 32-64-UNIV, 64-64-LVL, 32-128-UNIV, 16-128-UNIV, 16-4096-UNIV, 32-512-LVL, 64-128-LVL, 256-4096-LVL, 128-64-LVL, 16-256-LVL, 256-1024-LVL, 128-128-LVL, 64-256-LVL, 128-4096-LVL, 256-256-LVL, 256-64-LVL, 64-256-UNIV, 1024-4096-UNIV, 256-512-LVL, 16-256-UNIV, 32-512-UNIV, 64-512-UNIV, 64-128-UNIV, 128-1024-LVL, 32-256-LVL, 128-256-LVL, 128-512-LVL, 128-256-UNIV, 128-64-UNIV, 256-128-LVL, 128-128-UNIV, 256-128-UNIV, 1024-256-LVL, 32-1024-UNIV, 256-512-UNIV, 32-256-UNIV, 1024-512-LVL, 64-1024-LVL, 256-1024-UNIV, 16-1024-LVL, 32-1024-LVL, 1024-128-LVL, 16-1024-UNIV, 128-512-LVL, 1024-128-UNIV, 1024-64-LVL, 1024-256-UNIV, 1024-1024-UNIV, 32-4096-LVL, 1024-64-UNIV, 16-4096-LVL, 256-4096-UNIV, 1024-1024-LVL, 64-4096-UNIV, 32-4096-UNIV, 128-4096-UNIV, 1024-4096-LVL, 64-4096-LVL

LVL vs Univ Throughput Comparison

Readrandom

--use_existing_db

Key [16, 32, 64, 128, 256, 1024]

Value [64, 128, 256, 512, 1024, 4096]

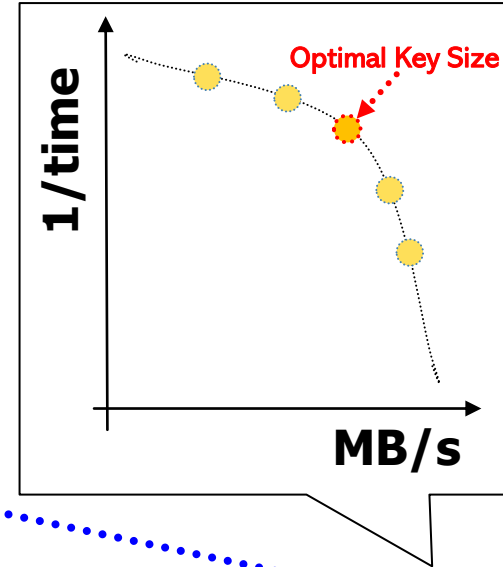
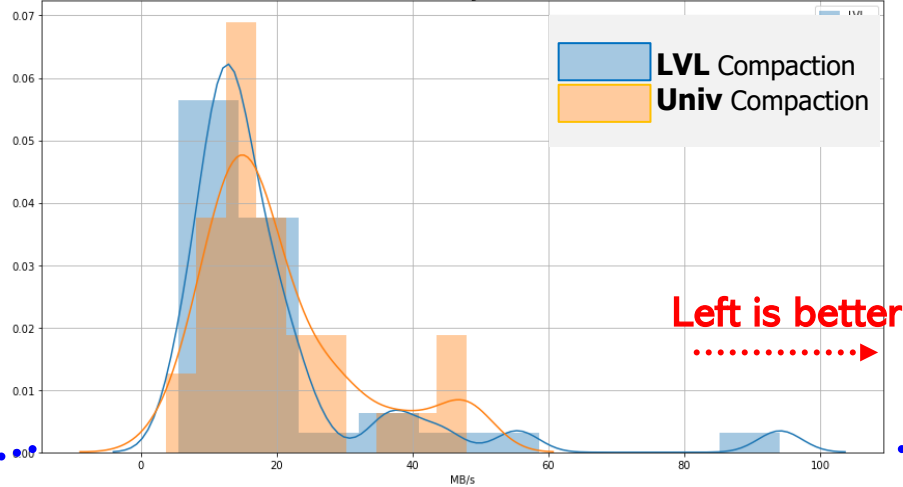
Entries 500 0000

Storage Samsung 1TB 860 Pro

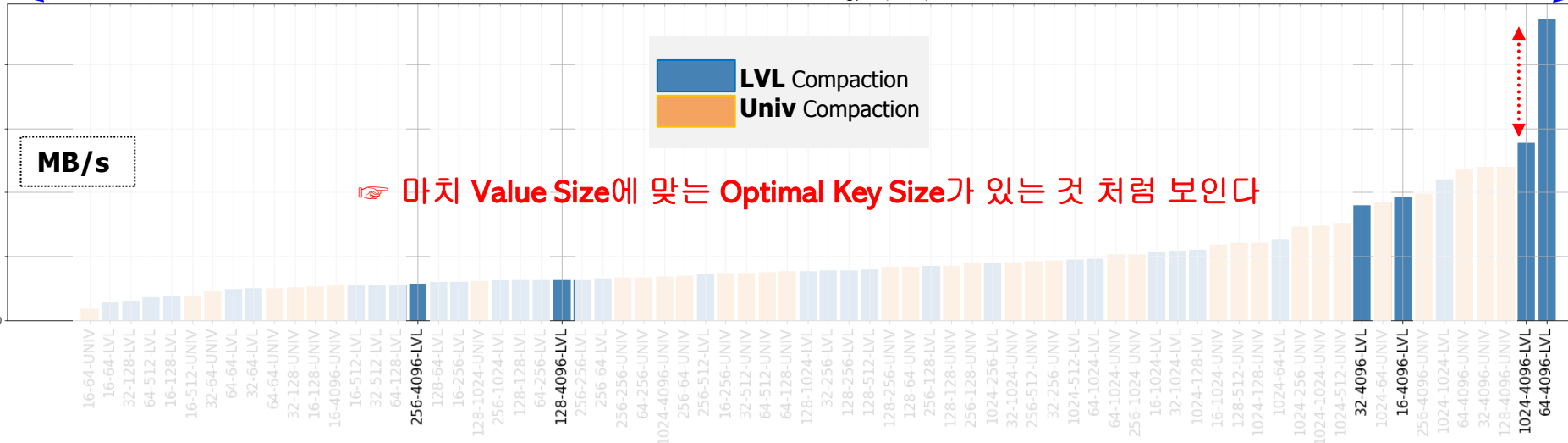
File System Ext4

CPU Intel(R) Core(TM) i7-10700K CPU @ 3.80GHz

Throughput comparison
between different compaction style
K[16-1024], V[64-4096] different Key-Value distributions(readrandom)

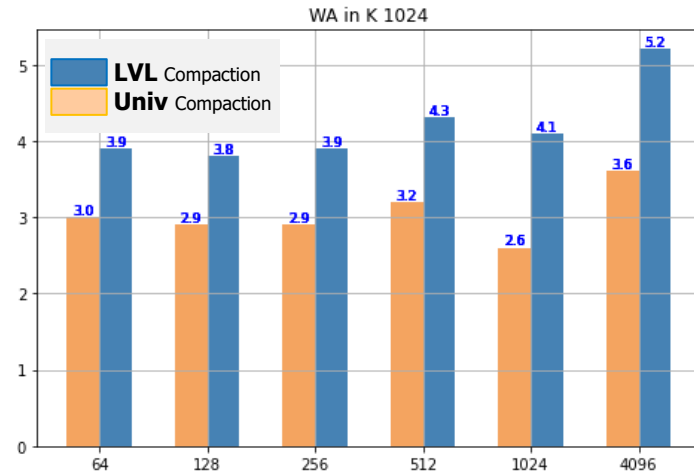
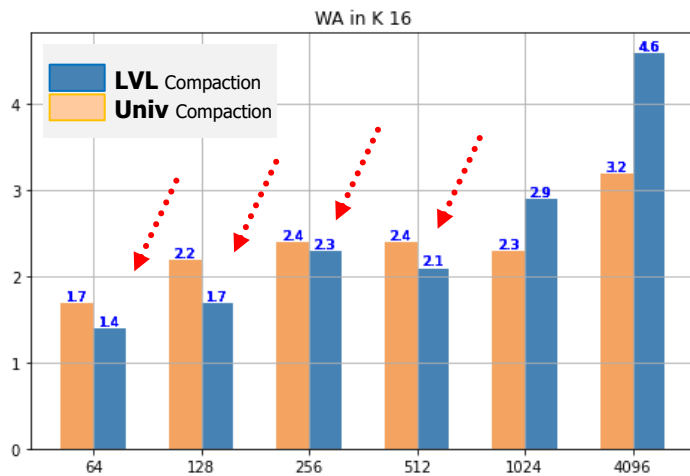
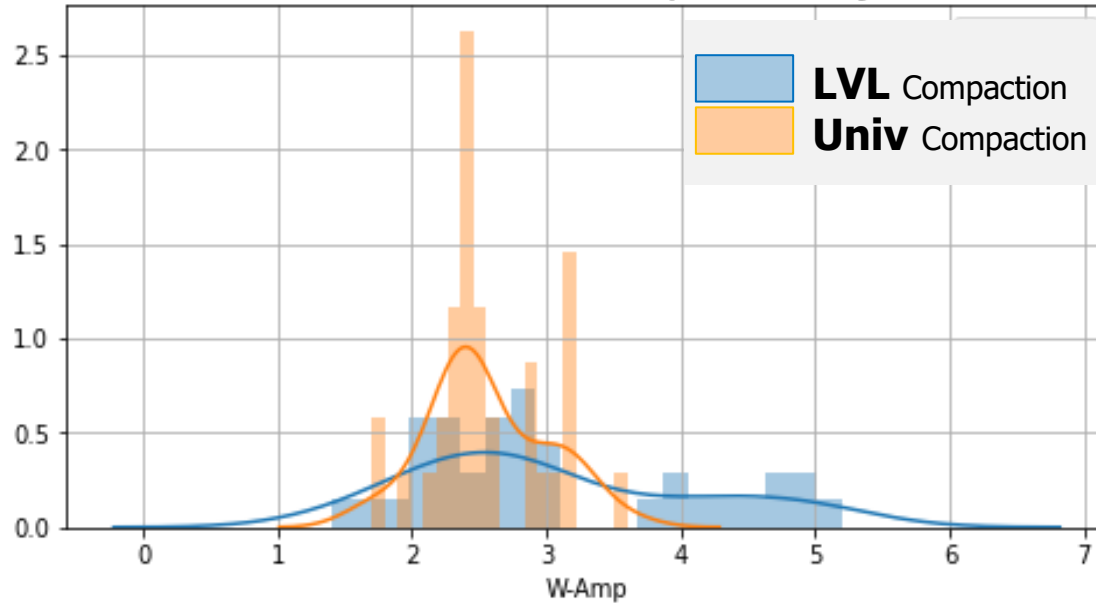


Readrandom Throughput (MB/s)



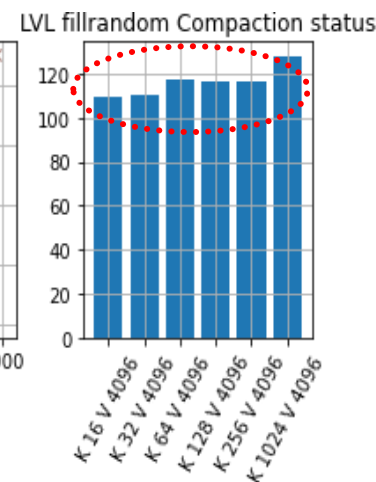
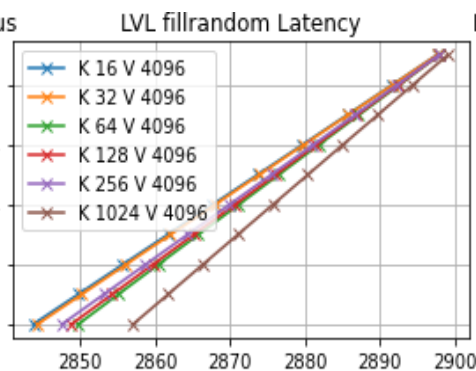
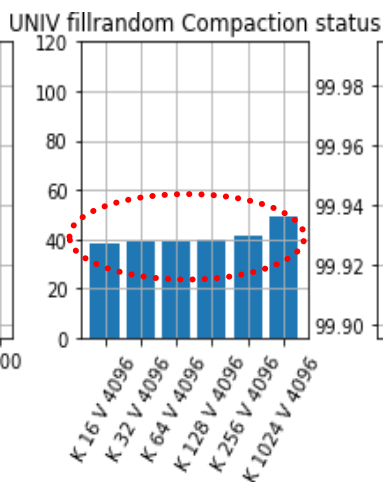
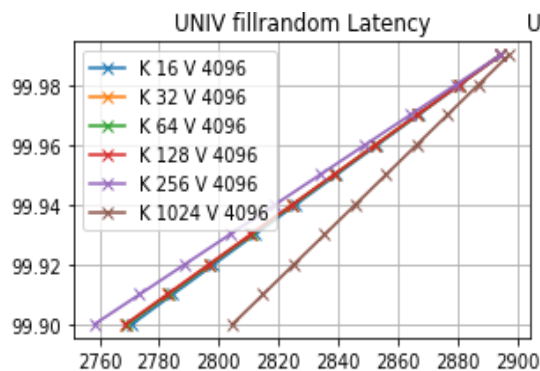
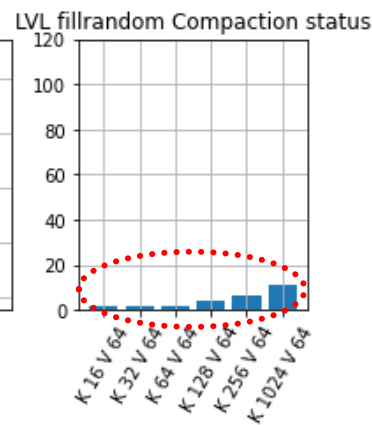
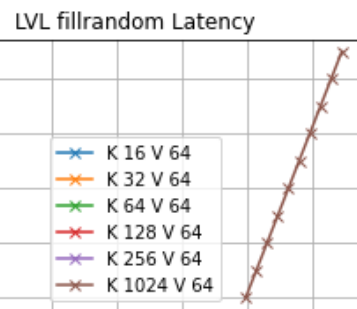
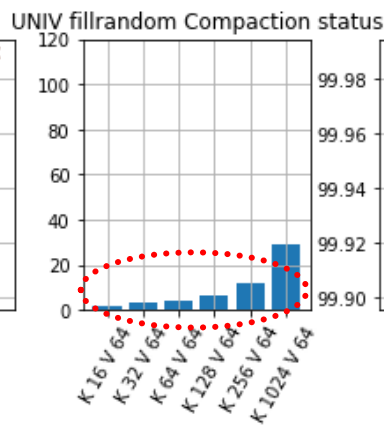
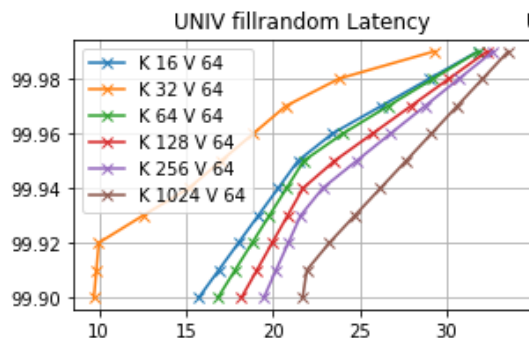
LVL vs Univ WAF Comparison

Write Amplification comparison
between different compaction Style



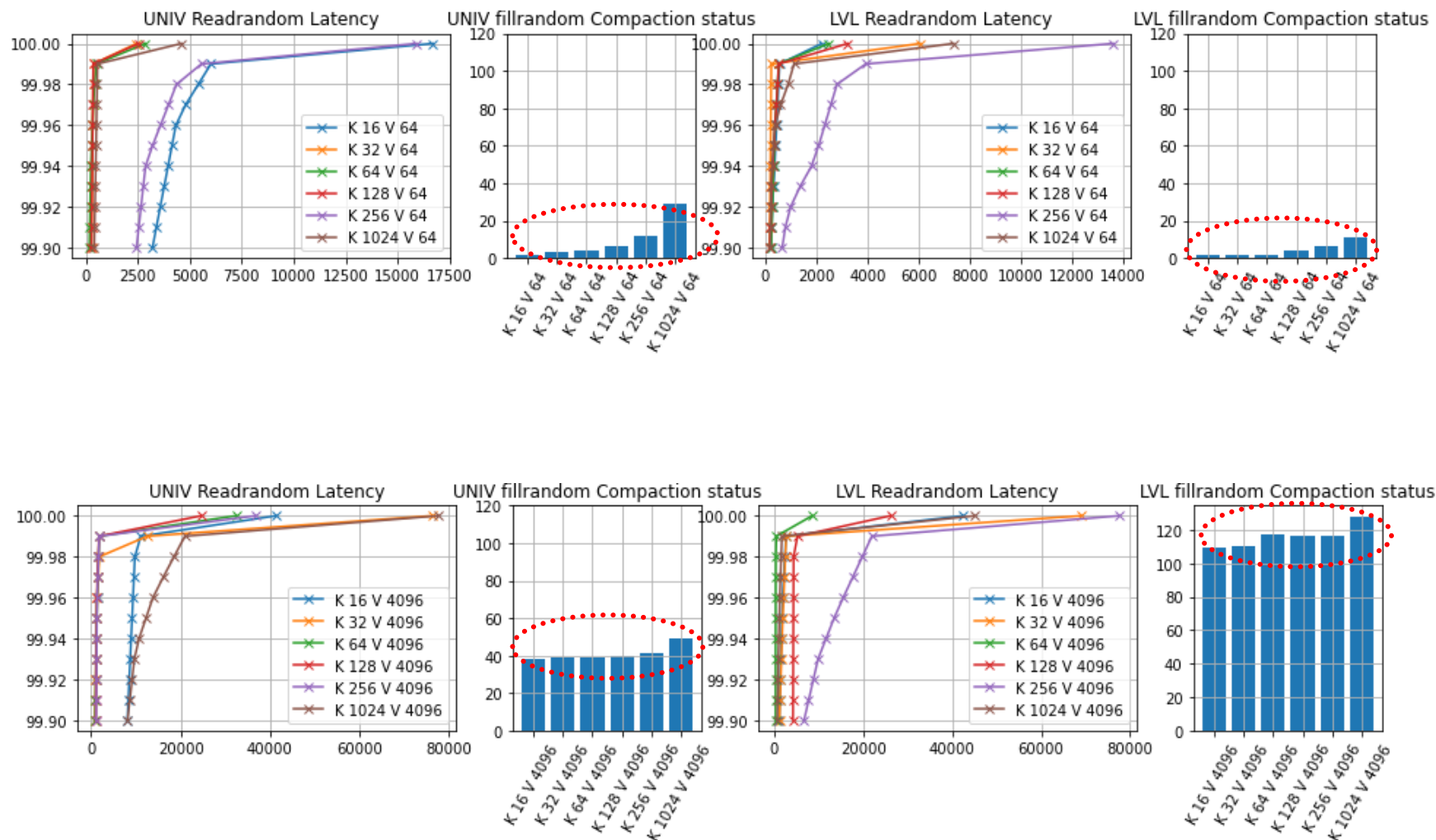
LVL vs Univ # of Compactions , latency Comparison

■ Fillrandom

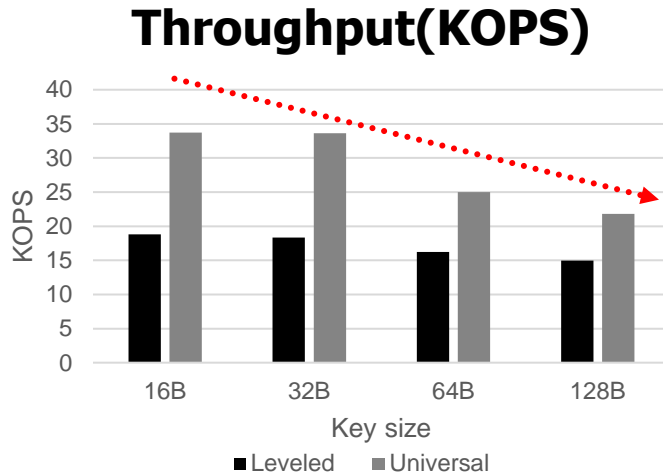


LVL vs Univ # of Compactions / latency Comparison

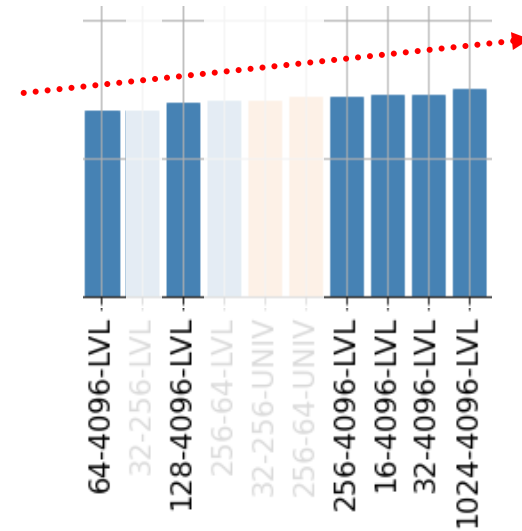
Readrandom



■ Issue on Last week



Throughput(MB/s)



☞ 한 쪽은 **OPS**, 한 쪽은 **MB/s** 임.

즉, **KV Size**가 늘어날 수록 **MB/s**는 늘어나고, **OPS**는 줄어들게 됨

■ Next Week

✓ Mental Model

- Leveled vs Universal Compaction 관련 논문 조사
- Hybrid Compaction 고민

✓ Leveled vs Universal Compaction Comparison

- Read 성능에 대한 fine grain 측정 – Read 부분별 시간 측정
- Space Amplification 측정
- Read Amplification 측정

Discussion

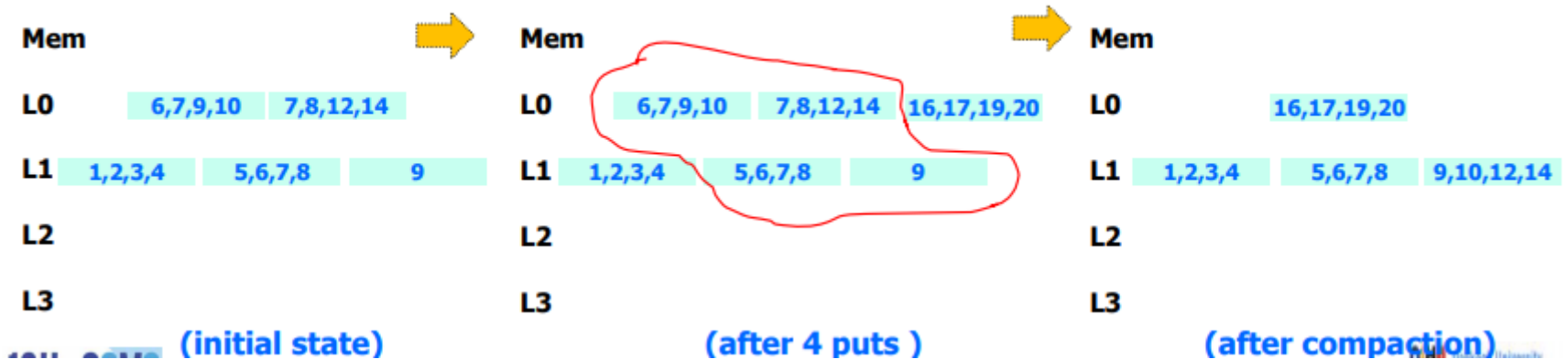


Appendix

■ How to estimate Space Amplification Factor?

$$SAF = \frac{\text{Actual used space}}{\text{Required space}} = \frac{\text{DB에 실제로 저장된 공간}}{\text{최소로 필요한 공간}} \quad \text{Not so simple}$$

· WAF = 13/5, SAF = 16/16 (note, 21/16 in the middle, lazy deletion)



Jongmoo Choi, Key-Value Store: Database for Unstructured Bigdata (Focusing on RocksDB), 12th CSMS 융합·스마트미디어 시스템 워크샵

중첩된 KV 를 고려하지 않아야함

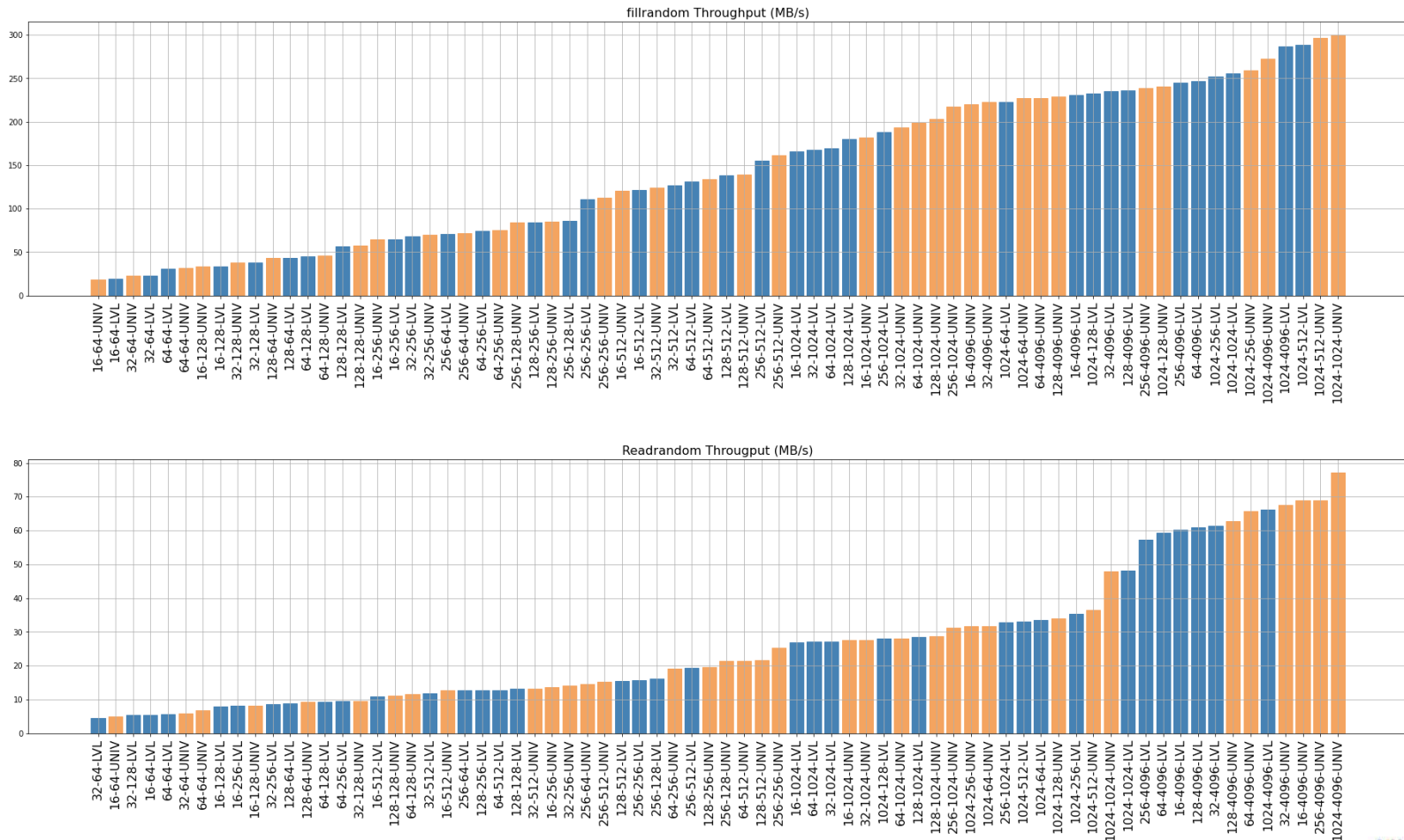
따라서, (바라건대,) Sequential pattern 으로 쓰인 Size를 '최소로 필요한 공간' 으로 두고 구할 예정.

Comment Plz

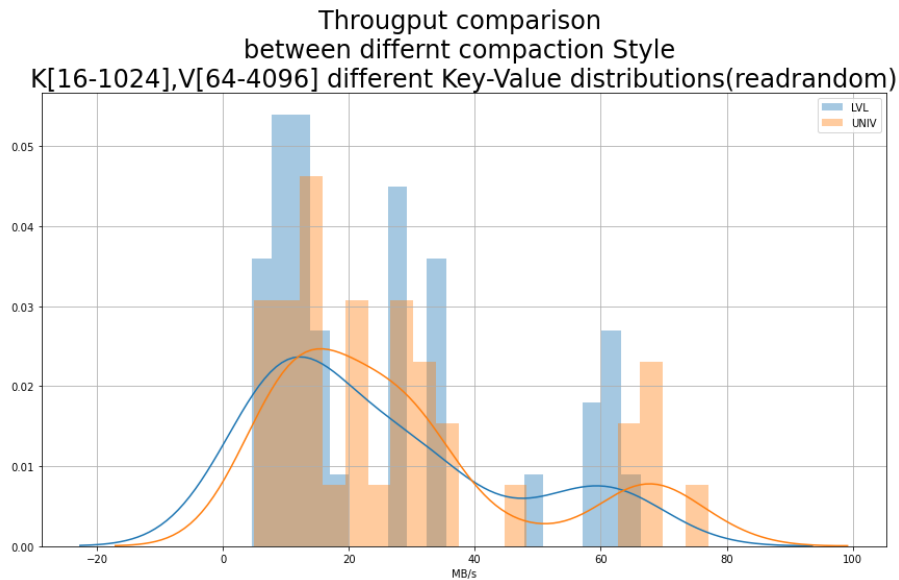
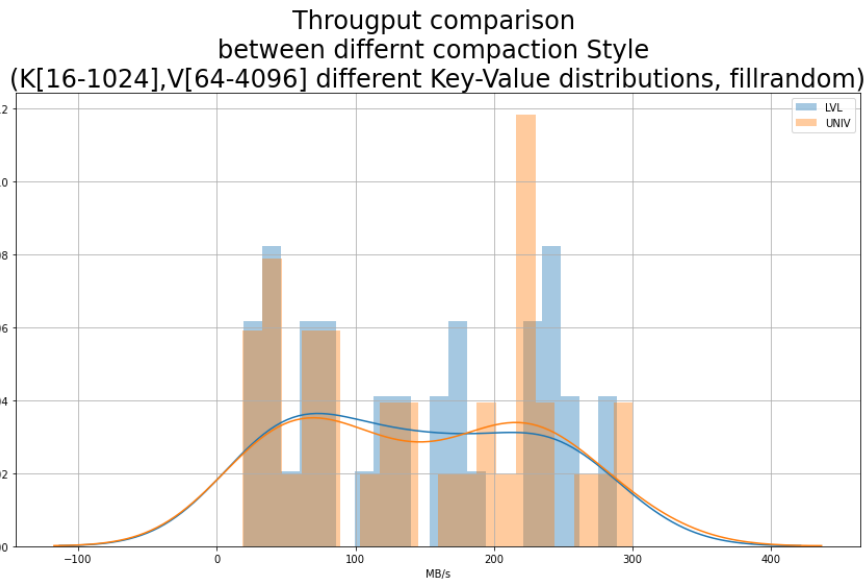
■ When we fix the DB size (Request size \approx 2.4GB)

- 현재 Request size = 800GB로 수정하고 DB bench 실험중..

Throughput Comparison of differnt Compaction Style

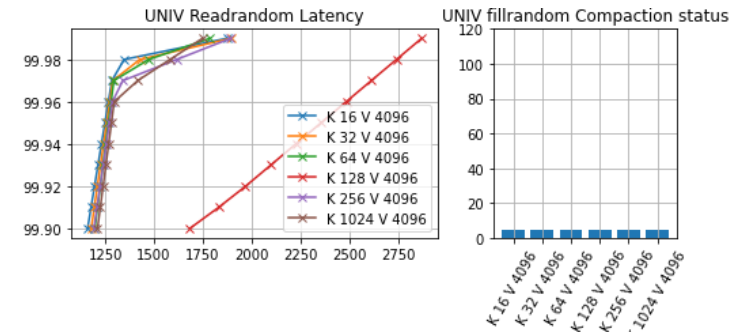
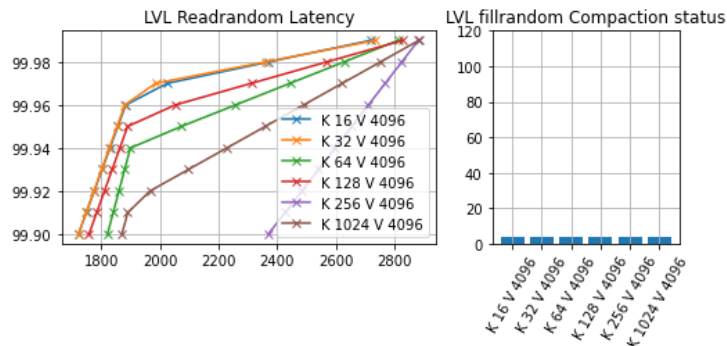
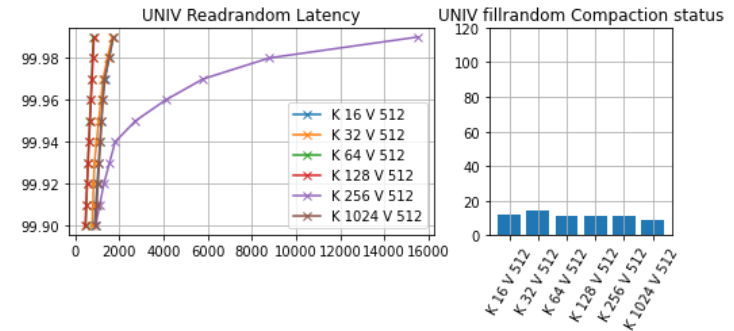
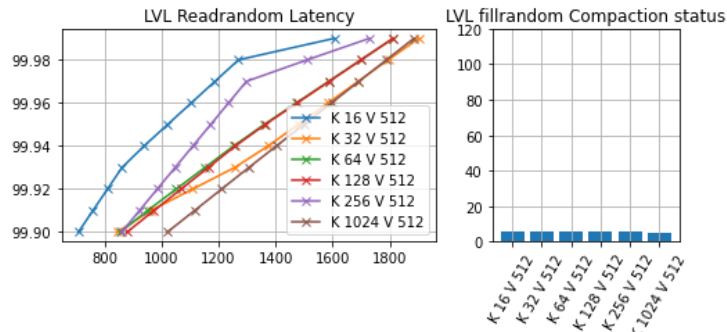
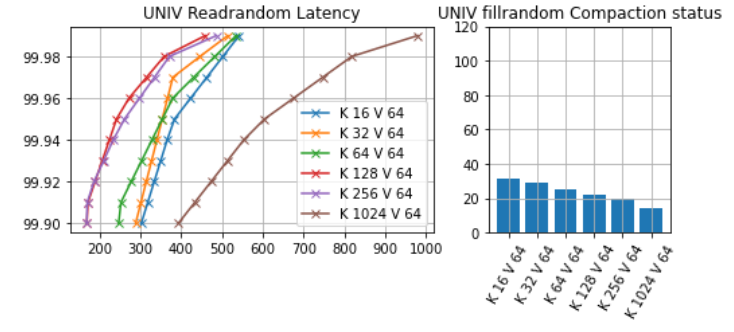
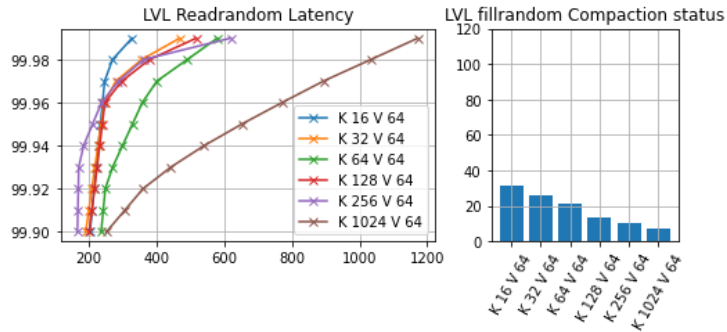


- When we fix the DB size (Request size \approx 2.4GB)
- 현재 Request size = 800GB로 수정하고 DB bench 실험중..



■ When we fix the DB size (Request size $\approx 2.4\text{GB}$)

LVL compaction latency 99.99% UNIV compaction latency 99.99%



Last Week

■ Compaction에 영향을 미치는 녀석들

✓ #1 KV-Size

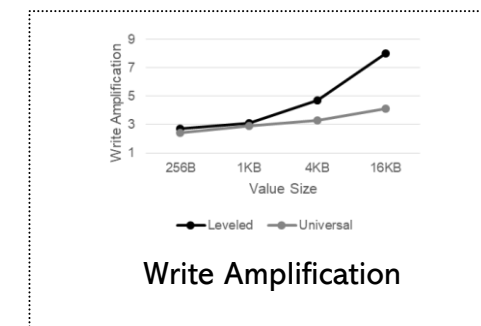
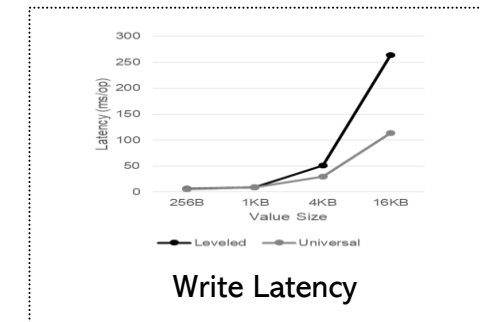
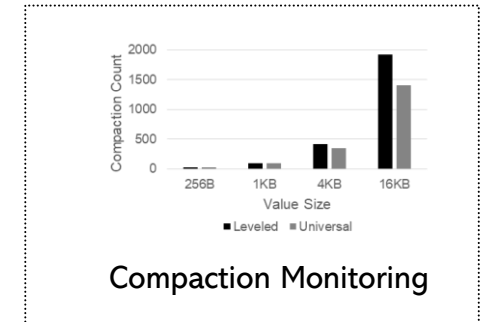
■ Various Key Size

- Key: 16B, 32B, 64B, 128B
- Value: 8K
- fillrandom, readrandom, range query, 5000000
- Leveled Compaction vs. Universal Compaction
- Write Amplification

■ Various Value Size

- Key: 16B
- Value: 256B, 1KB, 4KB, 16KB
- fillrandom, readrandom, range query, 5000000
- Leveled Compaction vs. Universal Compaction
- Write Amplification

+YCSB Workload, compare Read/Space Amplification

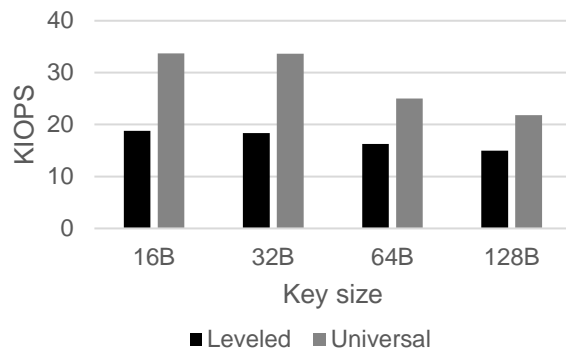


RocksDB Festival

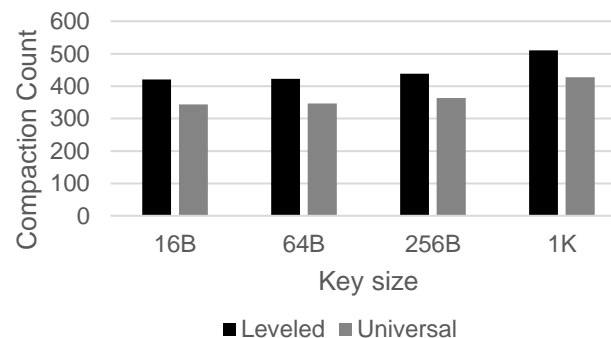
■ RocksDB::Compaction

✓ Trial#3 Compaction on various Key size (random write)

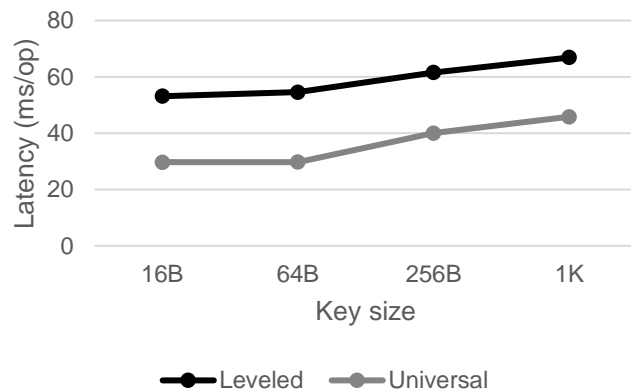
Throughput



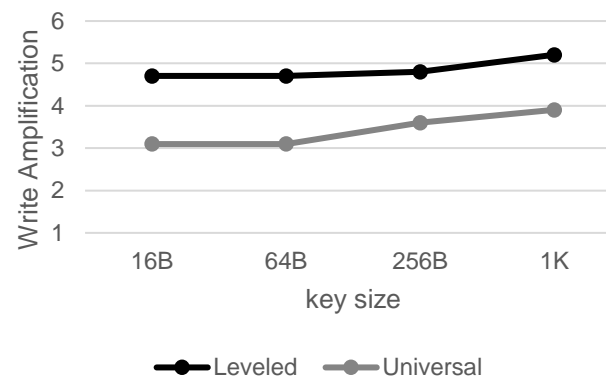
Compaction



Latency



WAF

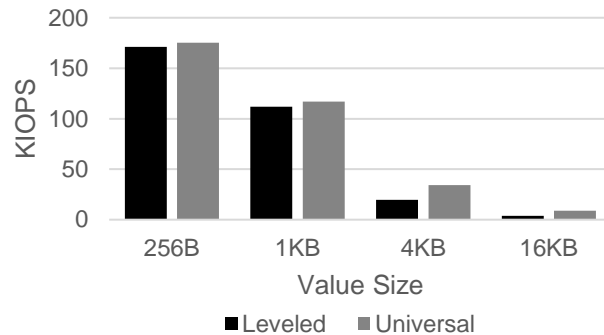


RocksDB Festival

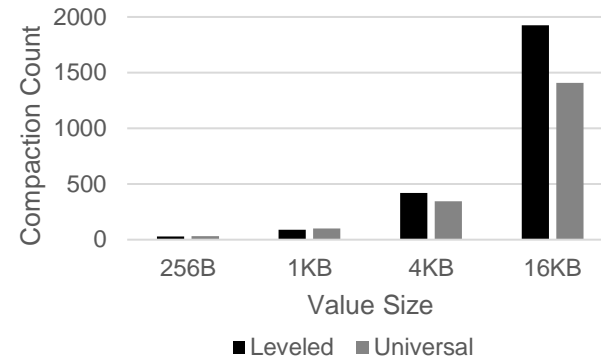
■ RocksDB::Compaction

- ✓ Trial#4 Compaction on various Value size (random write)

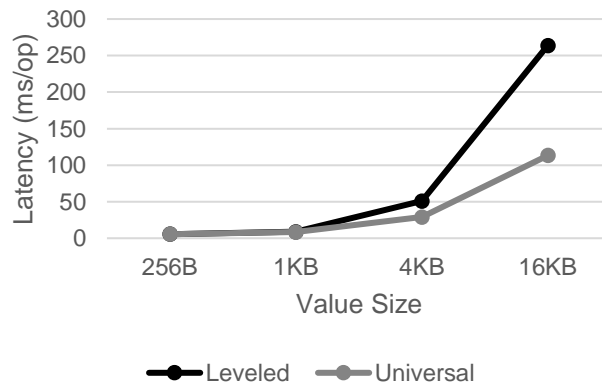
Throughput



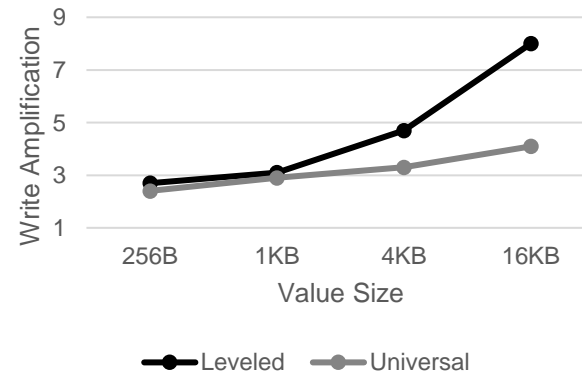
Compaction



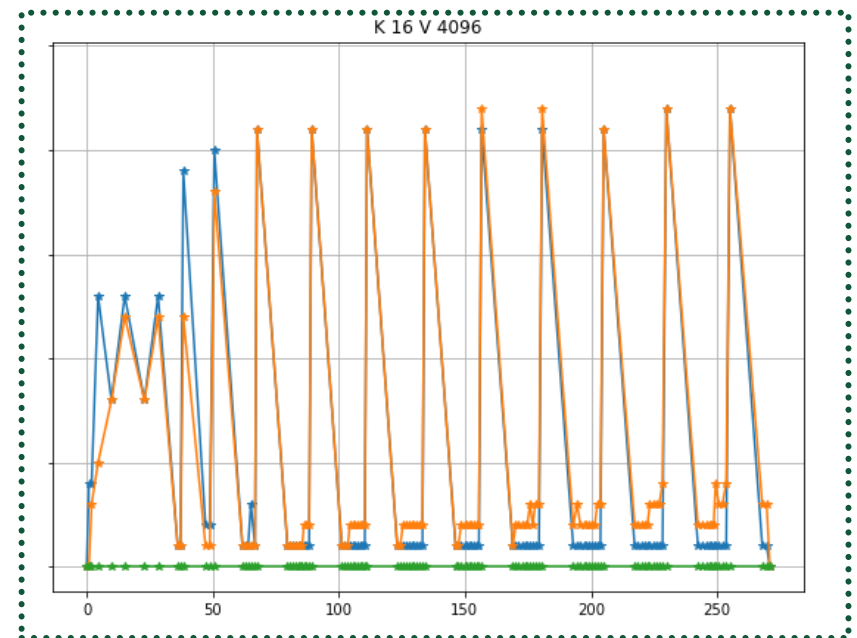
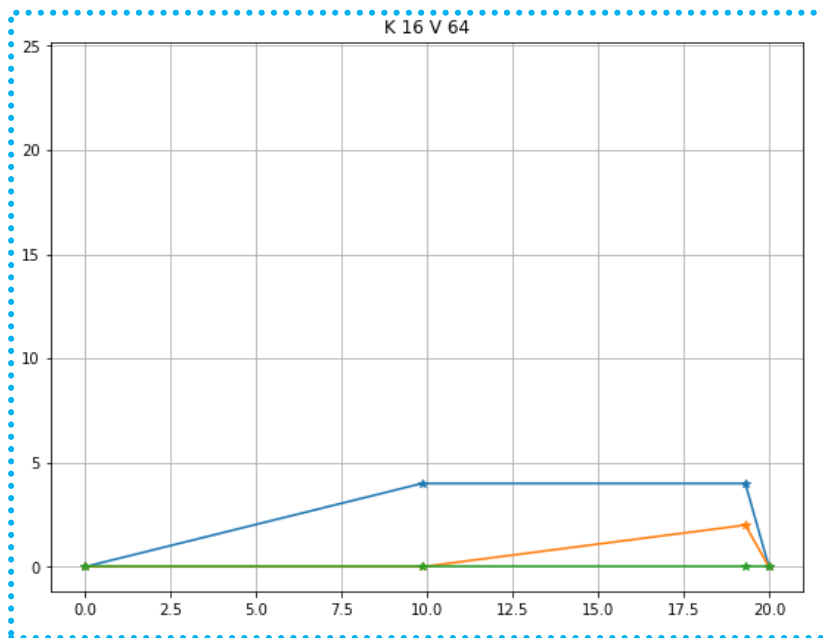
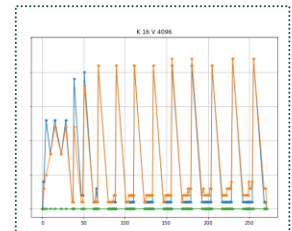
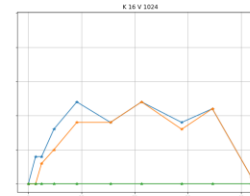
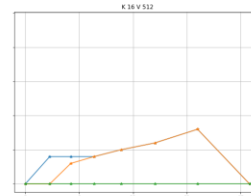
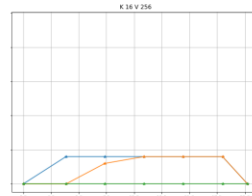
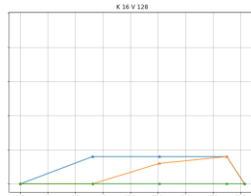
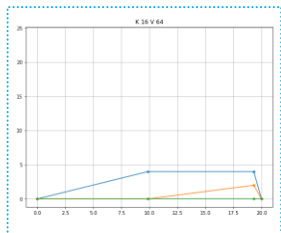
Latency



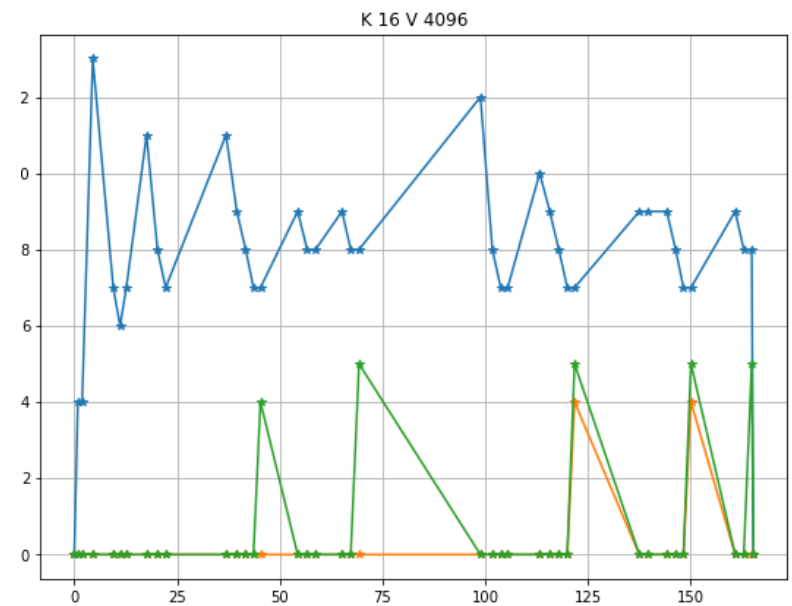
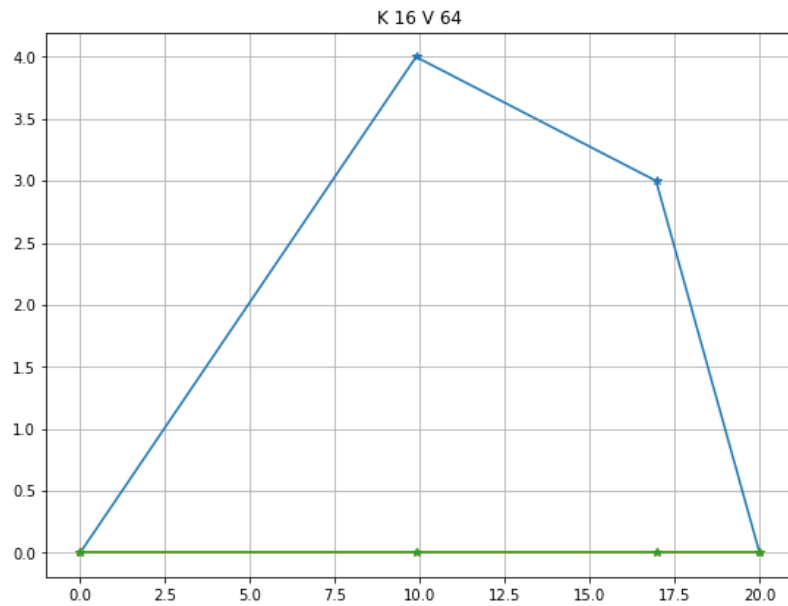
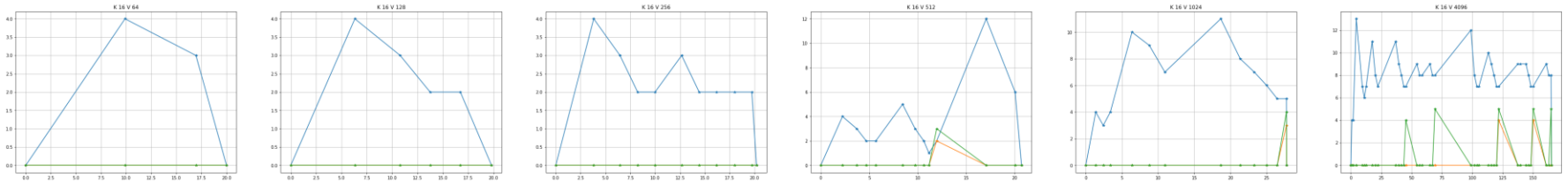
WAF



Level Compactions Trace



Universal Compactions Trace

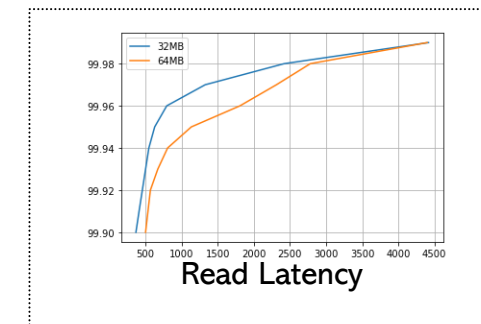
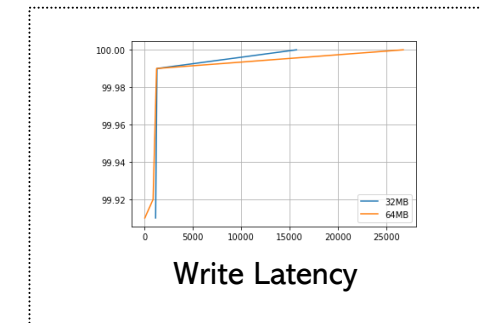
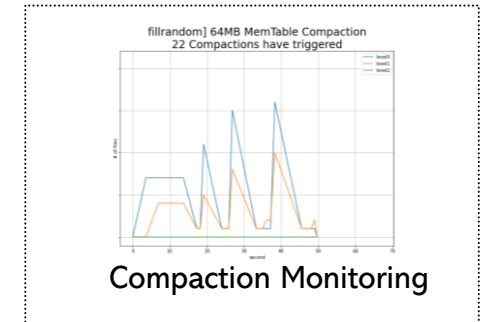


K 16 V 4096

■ Compaction에 영향을 미치는 녀석들

✓ #2 MemTable, SSTable

- Various MemTable + Various SST
 - 64MB, 32MB
 - fillrandom, readrandom, 16-512, 10000000
- Various MemTable + 64MB SST
 - 64MB, 32MB, ~~16MB, 8MB, 4MB, 2MB~~
 - fillrandom, readrandom, 16-512, 10000000

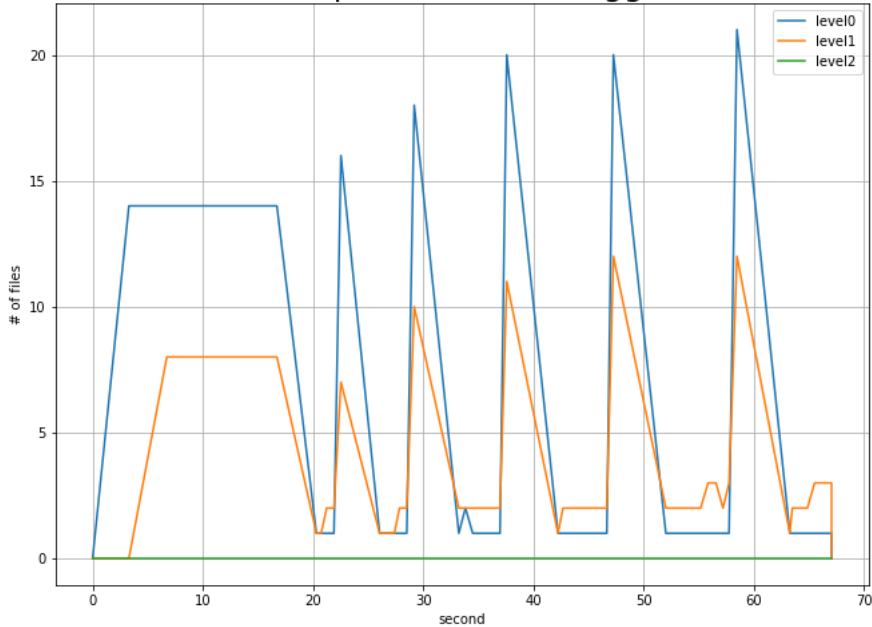


RocksDB Festival

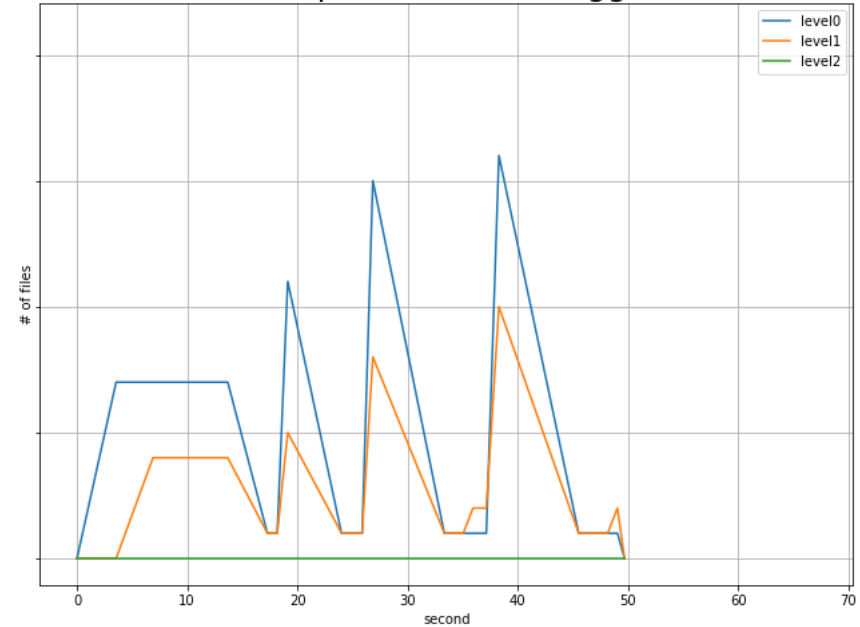
■ Compaction::SSTable

- ✓ Trial#1 Compaction on MemTable size&Target File Size (32MB vs 64MB)

fillrandom] 32MB MemTable Compaction
52 Compactions have triggered

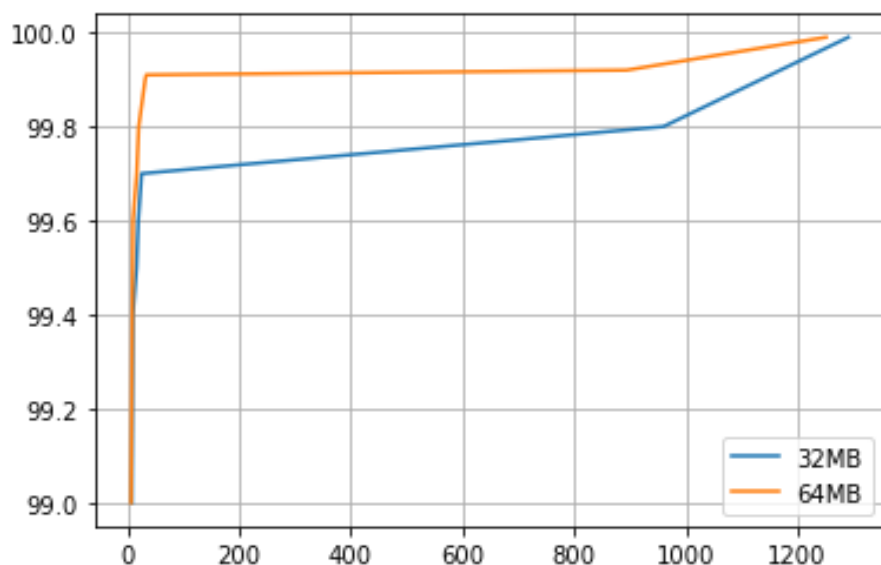


fillrandom] 64MB MemTable Compaction
22 Compactions have triggered

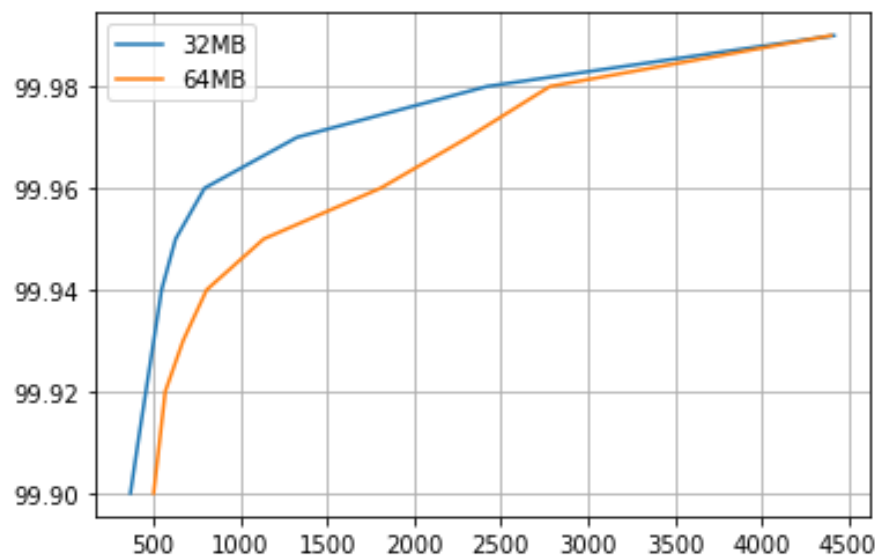


■ Compaction::SSTable

- ✓ Trial#1 Compaction on MemTable size&Target File Size (32MB vs 64MB)



fillrandom] Write Latency (99%)



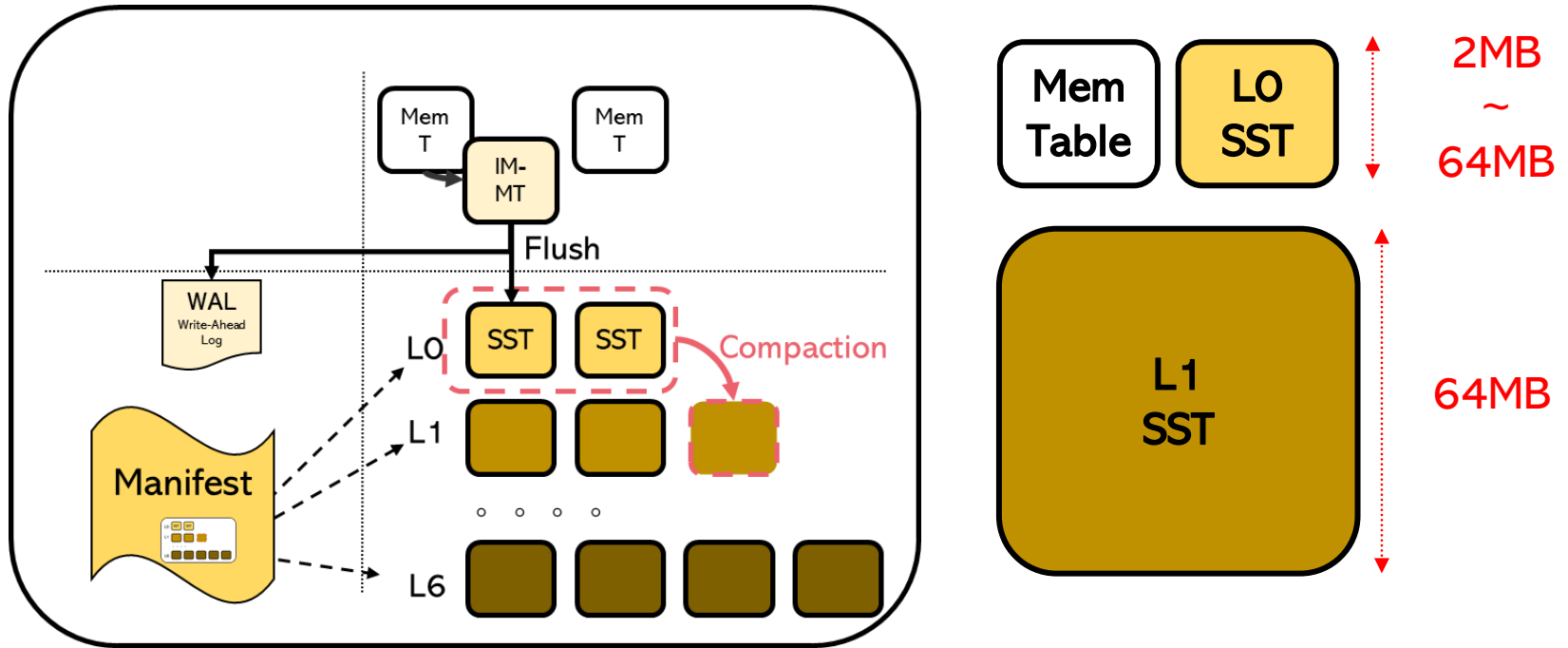
readrandom] Read Latency (99%)

👉 Read/Write latency Trade-off on MemTable Size

RocksDB Festival

■ Compaction::SSTable

- ✓ Trial#2 Compaction on MemTable size, but Target File Size 64MB
(MemT=[2,4,8,16,32,64]MB, SST_Level1 = 64MB)



LSM Tree based KV Store

( levelDB ,  RocksDB)

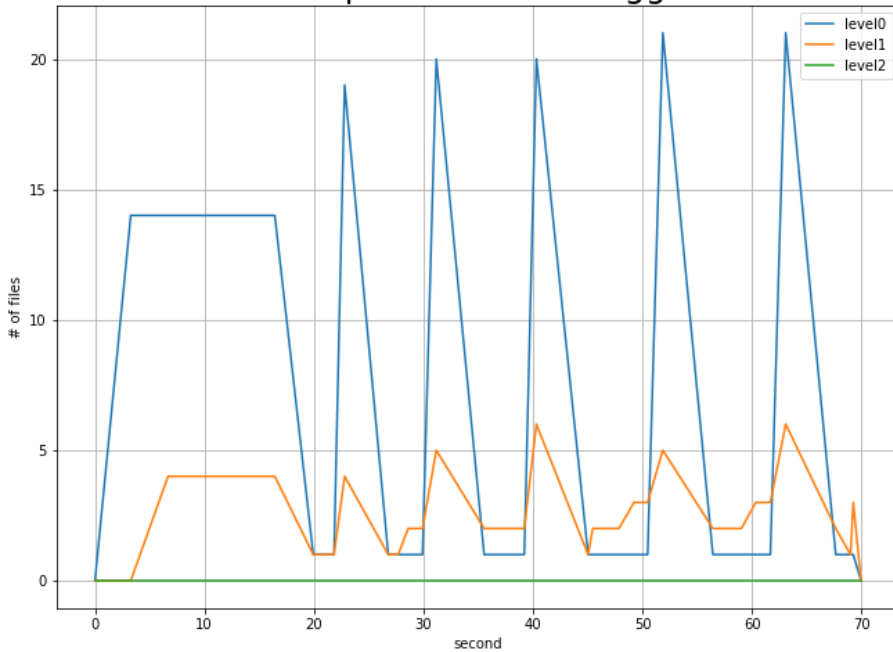


RocksDB Festival

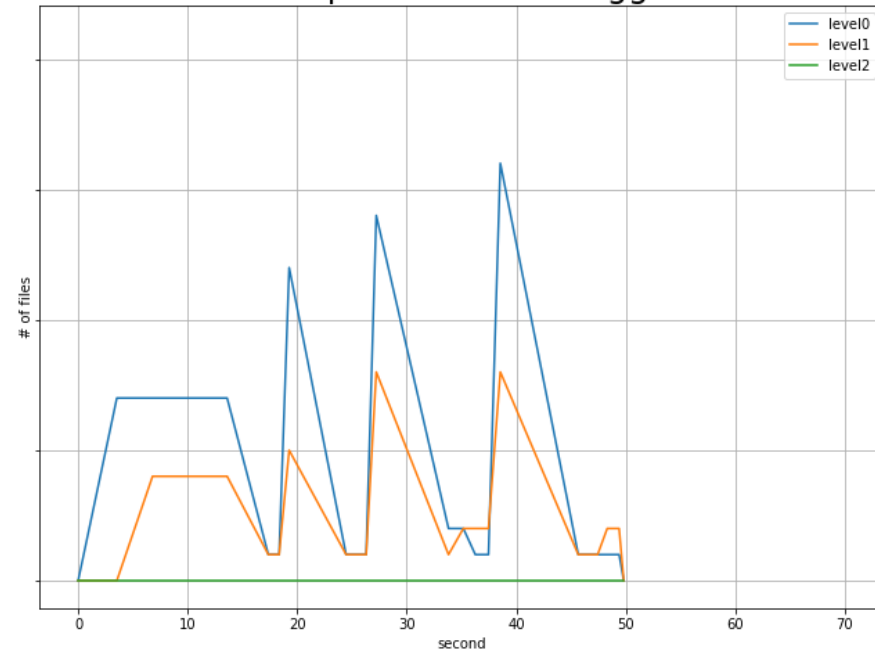
■ Compaction::SSTable

- ✓ Trial#2 Compaction on MemTable size, but Target File Size 64MB (MemT=[2,4,8,16,32,64]MB, SST_Level1 = 64MB)

fillrandom] M:32MB SST_lv1:64MB Compaction
36 Compactions have triggered



fillrandom] M:64MB SST_lv1:64MB Compaction
23 Compactions have triggered

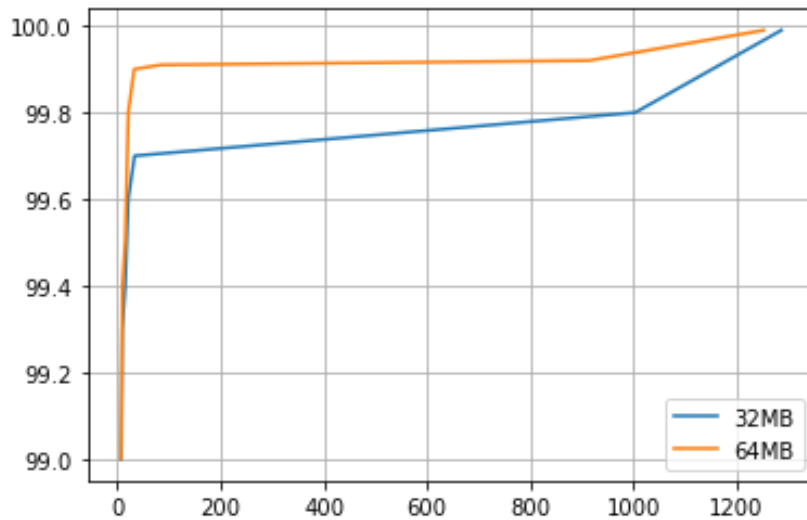


👉 No difference between previous experiment

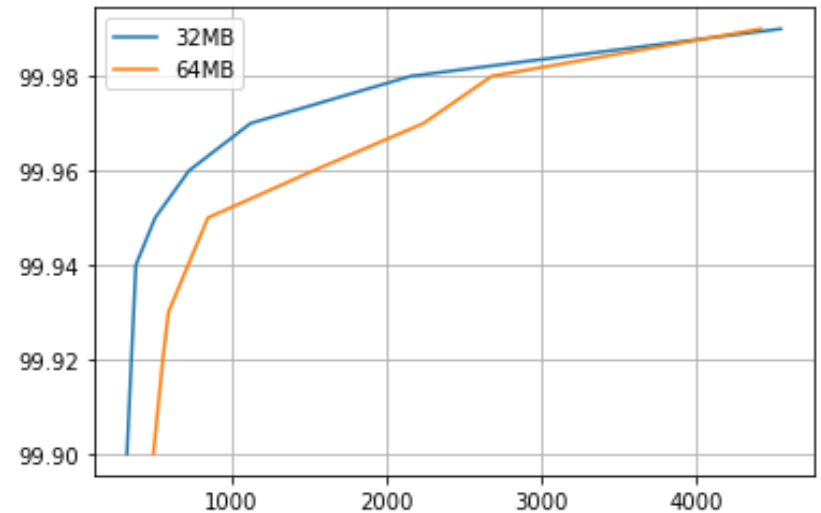
RocksDB Festival

■ Compaction::SSTable

- ✓ Trial#2 Compaction on MemTable size&Target File Size (32MB vs 64MB)



fillrandom] Write Latency (99%)



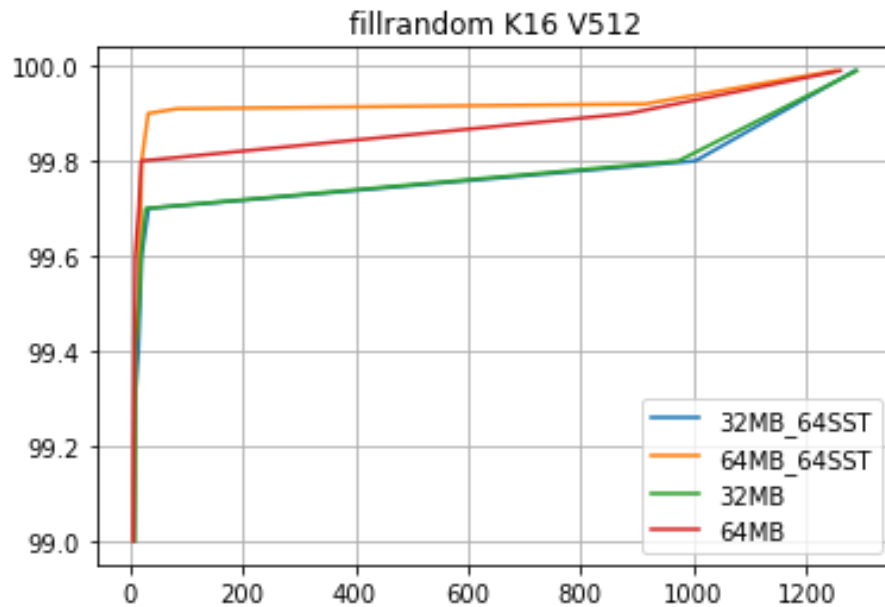
readrandom] Read Latency (99%)

👉 No difference between previous experiment

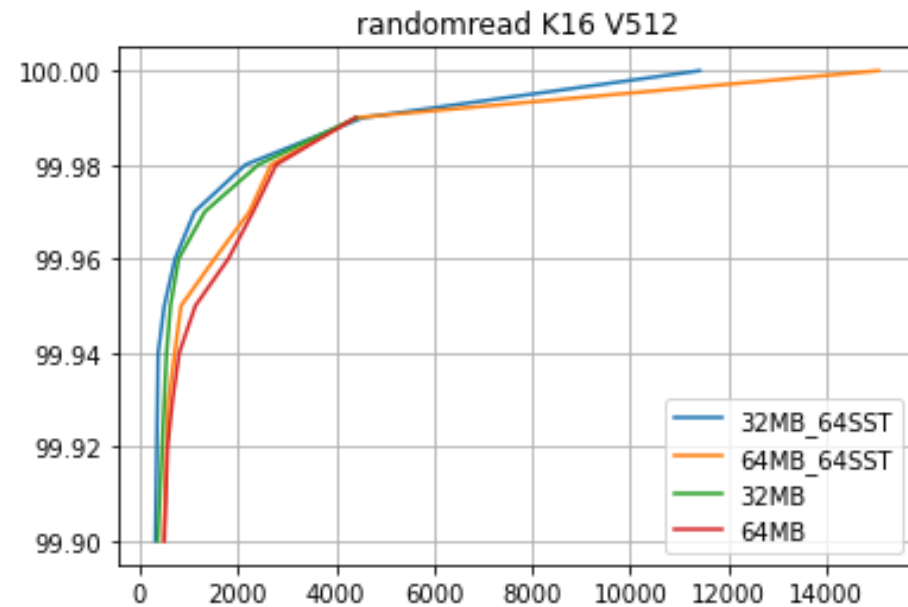
RocksDB Festival

■ Compaction::SSTable

✓ Trial#1 vs Trial#2



fillrandom] Write Latency (99%)



readrandom] Read Latency (99%)

