단국대학교 DANKOOK UNIVERSITY

Compaction과 WAL, WAF의 상관성 분석

과 목 명: 오픈소스SW분석(빅데이터) 1분반
발 표 일: 2025년 04월 24일 (목)
팀　　원: 구선주 (32220207), 임수연 (32193772), 최예림 (32224684)

Contents

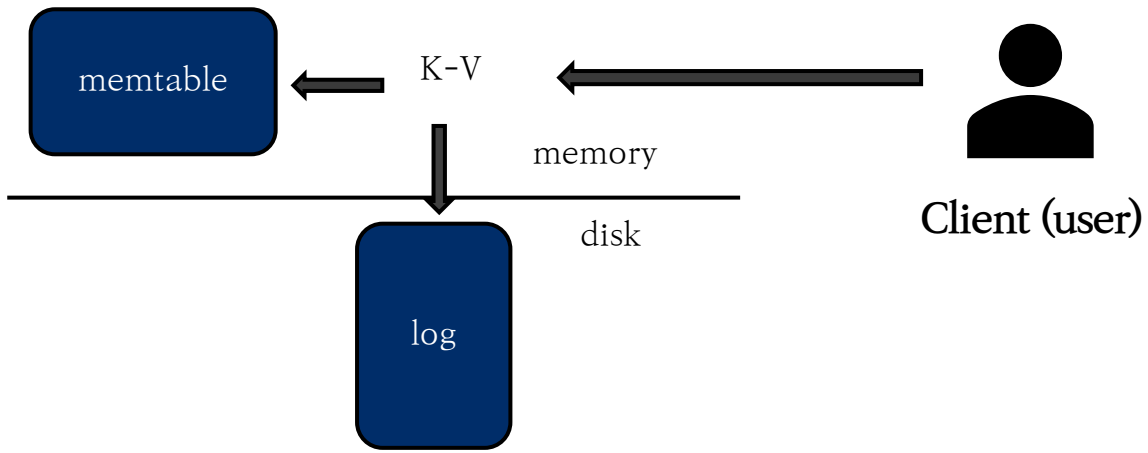
01

서론

- 배경 지식
- 연구 배경

배경 지식

WAL (Write-Ahead Log)



RocksDB는 데이터 일관성을 유지하기 위해 WAL을 이용

쓰기 요청이 들어온 순간

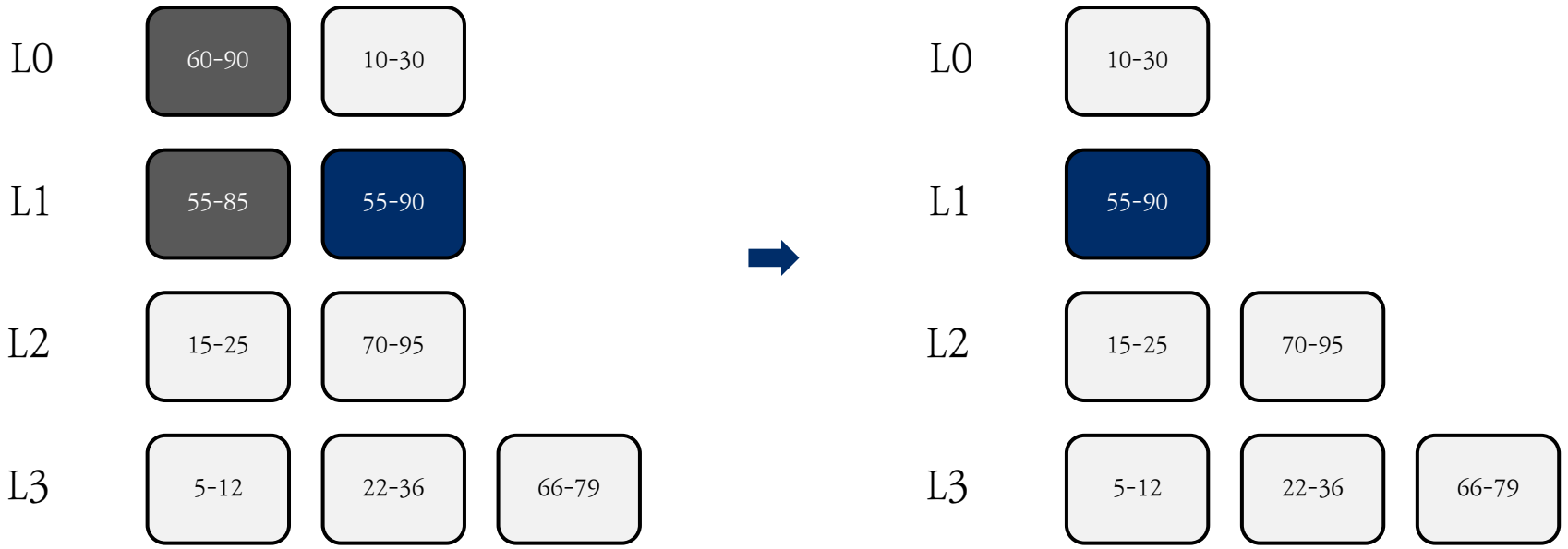
로그 파일로 디스크에 저장

Memtable이 디스크에 안전하게

저장된 후 WAL 파일 삭제

배경 지식

Leveled Compaction



Ln과 Ln+1에 겹치는 SSTable을 Compaction하여 Ln+1에 새로 생성

배경 지식

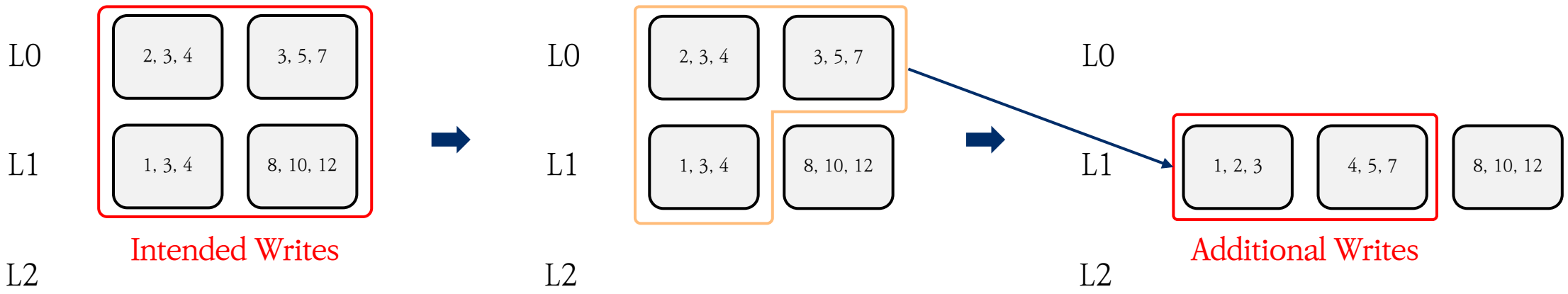
Universal Compaction

Update나 Delete가 많은 워크로드에 적합

배경 지식

WAF (Write Amplification Factor)

$$\text{Write Amplification Factor} = \frac{\text{Bytes written to Storage}}{\text{Bytes written to Database}}$$



연구 배경

가설 1

WAL 설정과 Compaction 전략 간의 상호작용은 WAF에 유의미한 영향을 미칠 수 있다.

가설 2

Update 및 Delete가 빈번한 워크로드에서 Leveled Compaction의 compaction 빈도를 최적화할 경우, Universal Compaction과 비슷한 수준의 WAF 성능을 달성할 수 있다.

02

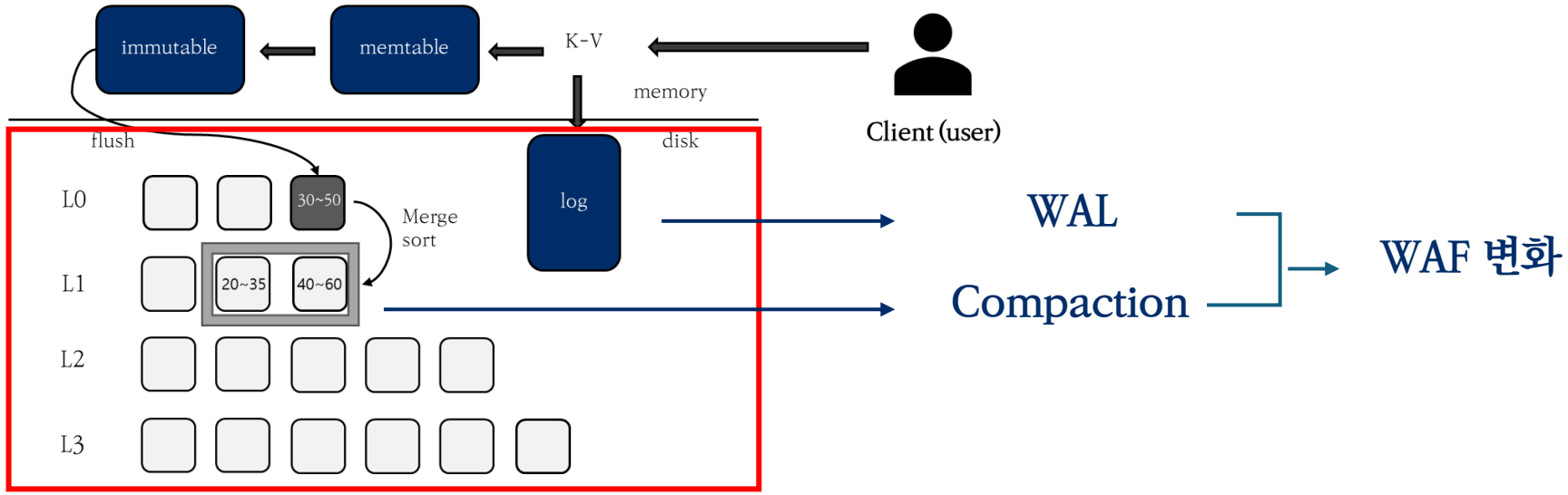
본론 1

- 가설 설정
- 실험 설계
- 실험 결과

가설 설정

Compaction과 WAL

WAL 설정과 Compaction 전략 간의 상호작용은 WAF에 유의미한 영향을 미칠 수 있다.



실험 설계

실험 계획

Configs Option	내용
perf	disable_wal=true sync=false use_direct_io_for_flush_and_compaction=true
stable	disable_wal=false sync=false use_direct_io_for_flush_and_compaction=false

WAL의 사용 여부에 따라 실험을 진행

실험 설계

실험 계획

Case	Compaction	workload	WAL
Leveled	Leveled Compaction	Fillrandom, overwrite	perf, stable
Universal	Universal Compaction	Fillrandom, overwrite	perf, stable

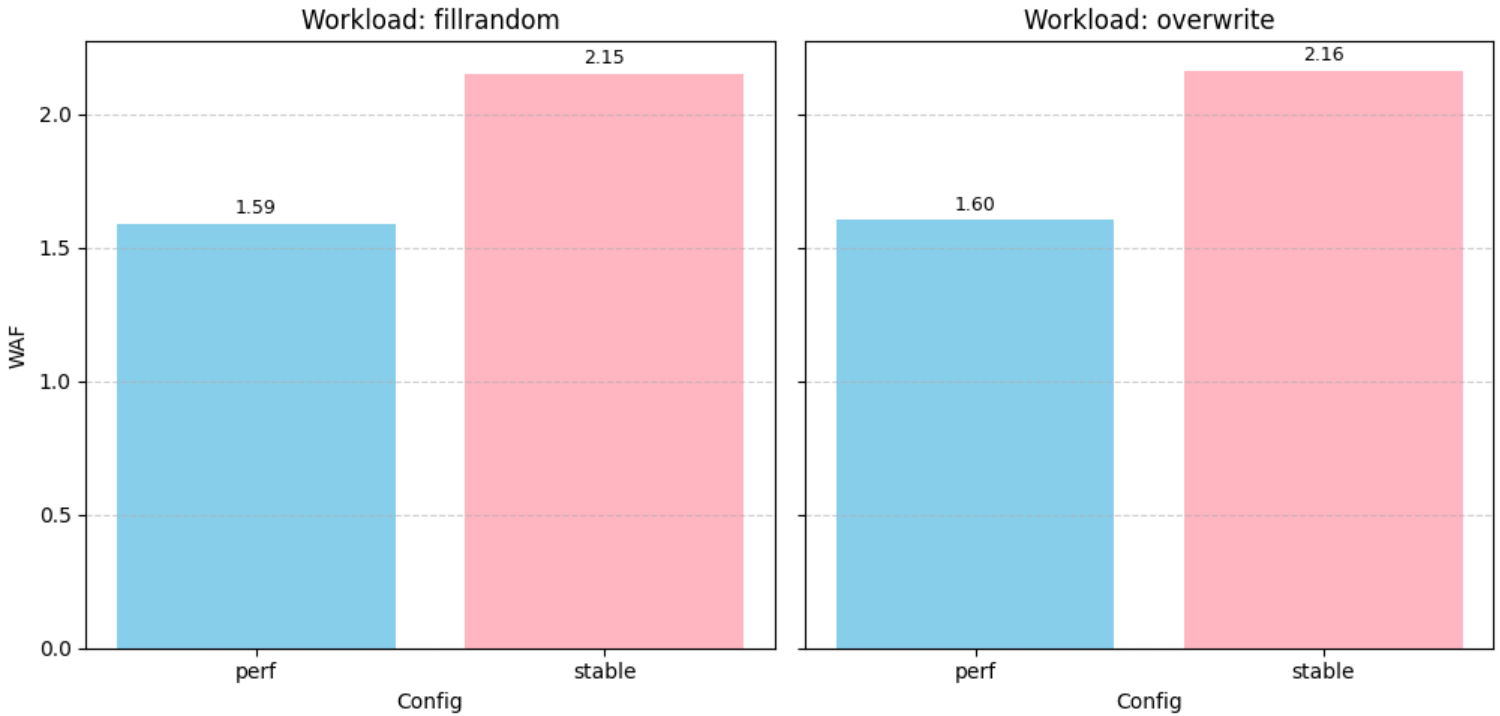
실험 결과

실험 환경

OS	UBuntu 20.04.6 LTS (64bit)
CPU	16 * Intel(R) Core(TM) i7-10700F CPU @ 2.90GHz
RAM	31.2GiB
SSD	1.0TB
RocksDB	Version 10.2.0

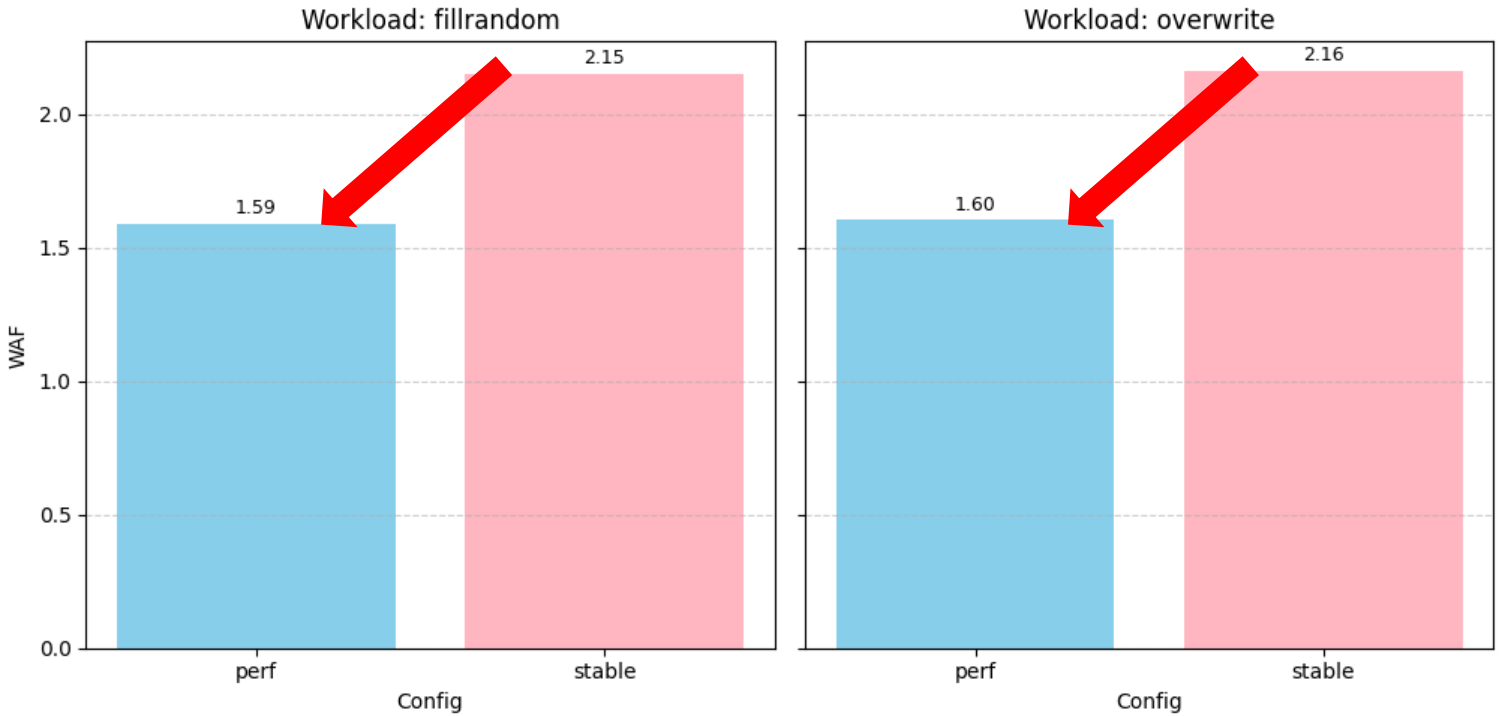
실험 결과

Leveled Compaction



실험 결과

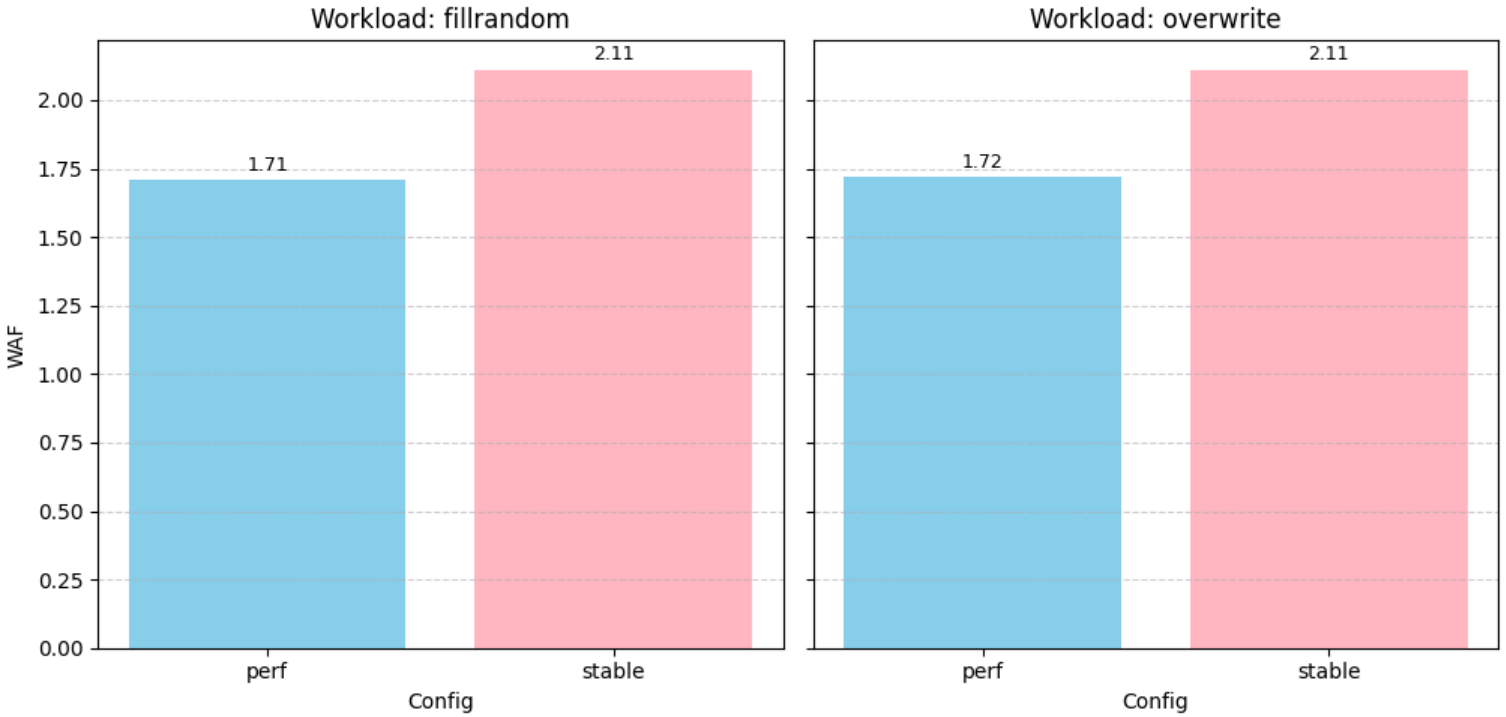
Leveled Compaction



Fillrandom, Overwrite의 perf가 stable보다 WAF 약 35% 감소

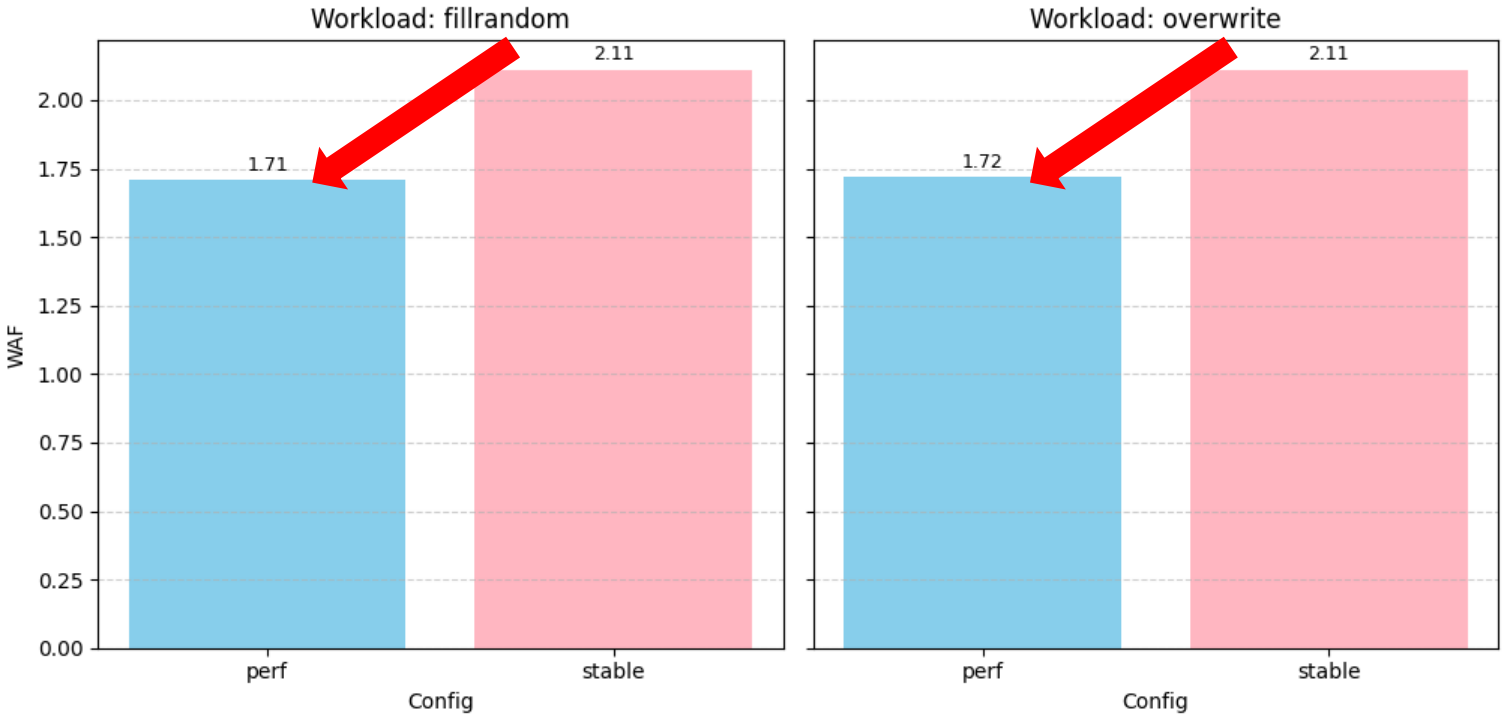
실험 결과

Universal Compaction



실험 결과

Universal Compaction



Fillrandom, Overwrite의 perf가 stable보다 WAF 약 23% 감소

03

본론 2

- 가설 설정
- 실험 설계
- 실험 결과

가설 설정

Leveled와 Universal Compaction

Update 및 Delete가 빈번한 워크로드에서 Leveled Compaction의 compaction 빈도를 최적화할 경우, Universal Compaction과 비슷한 수준의 WAF 성능을 달성할 수 있다.

Universal Compaction	Leveled Compaction
Update 및 Delete가 빈번한 workload WAF가 낮다	RocksDB에서 default로 사용 WAF가 높다

➡ Leveled Compaction의 base size와 fanout을 조정하여 compaction 빈도 변경 가능

실험 설계

실험 계획

Leveled Compaction Option	내용
--max_bytes_for_level_base	레벨 0의 최대 데이터 크기
--max_bytes_for_level_multiplier	다음 레벨의 용량 증가 비율

레벨 L의 최대 바이트: (max_bytes_for_level_base) * (max_bytes_for_level_multiplier^{(L-1)})

실험 설계

실험 계획

Case	Compaction	Value_size	workload	max_bytes_for_level_base (1) / max_bytes_for_level_multiplier (2)
Universal	Universal Compaction	1KB, 4KB, 16KB	Fillrandom, overwrite	-
Leveled	Leveled Compaction	1KB, 4KB, 16KB	Fillrandom, overwrite	-
Base	Leveled Compaction	1KB, 4KB, 16KB	Fillrandom, overwrite	64MB, 256MB, 512MB (1)
Multiplier	Leveled Compaction	1KB, 4KB, 16KB	Fillrandom, overwrite	4, 10, 20 (2)

실험 결과

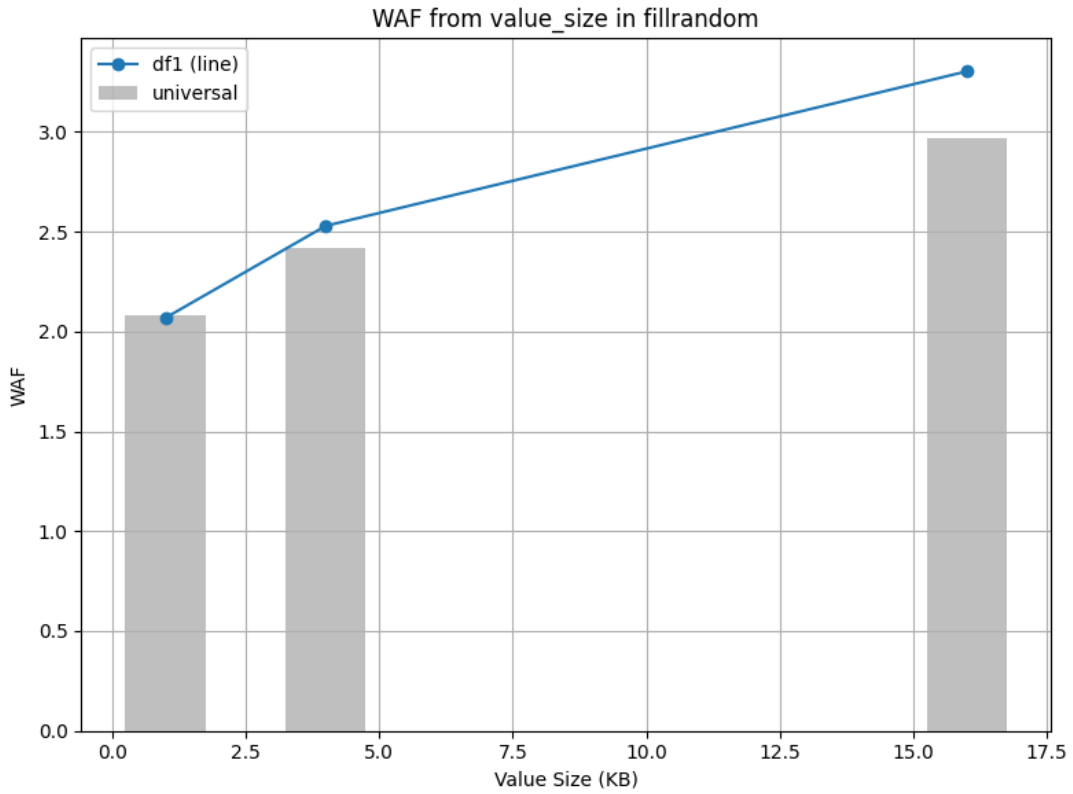
실험 환경

OS	UBuntu 20.04.6 LTS (64bit)
CPU	16 * Intel(R) Core(TM) i7-10700F CPU @ 2.90GHz
RAM	31.2GiB
SSD	1.0TB
RocksDB	Version 10.2.0

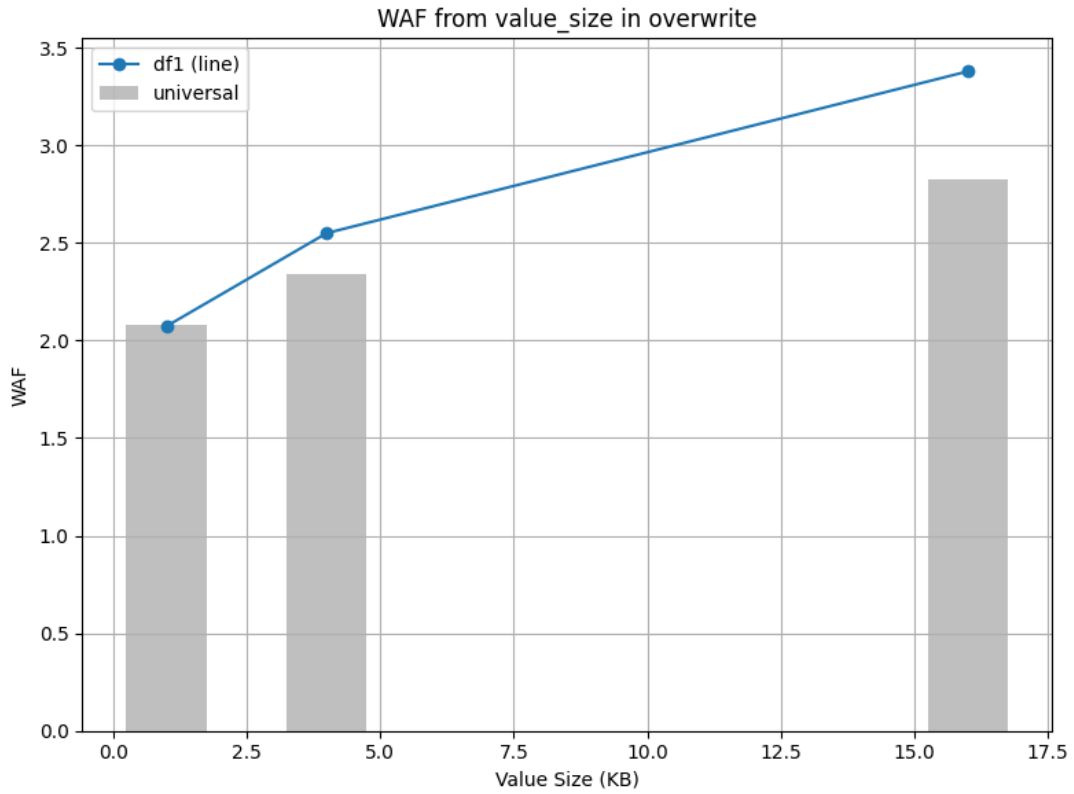
실험 결과

Leveled vs Universal

fillrandom



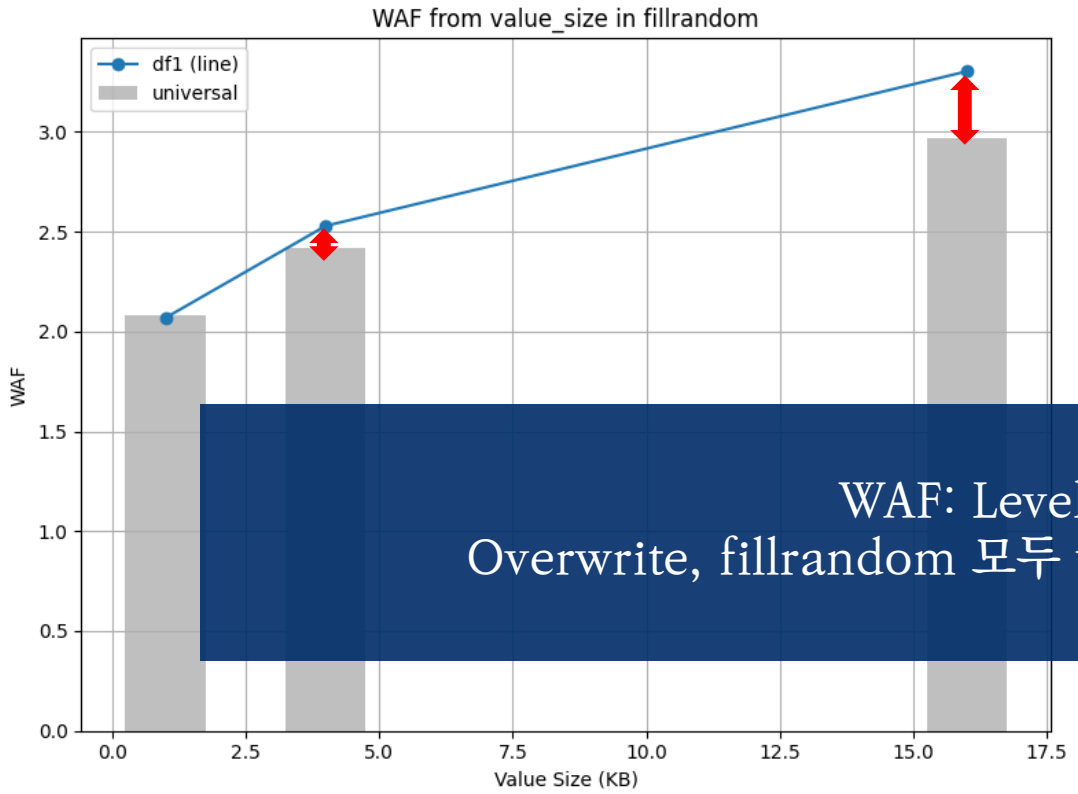
overwrite



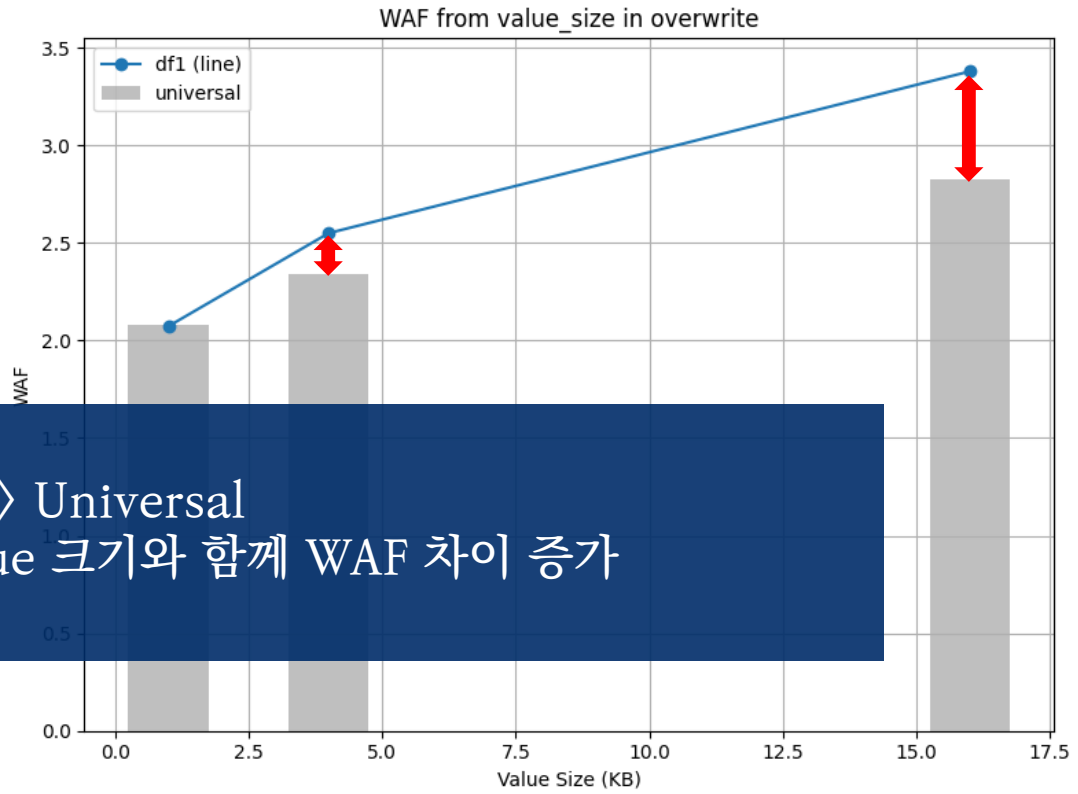
실험 결과

Leveled vs Universal

fillrandom



overwrite

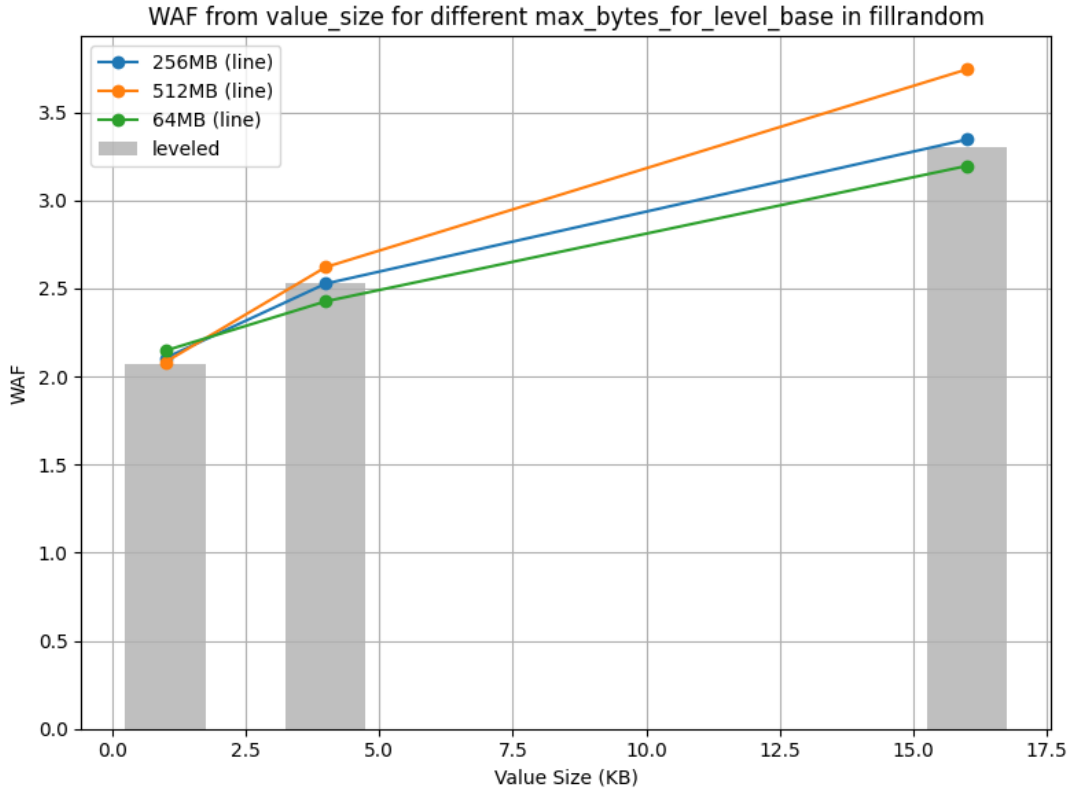


WAF: Leveled 〉 Universal
Overwrite, fillrandom 모두 value 크기와 함께 WAF 차이 증가

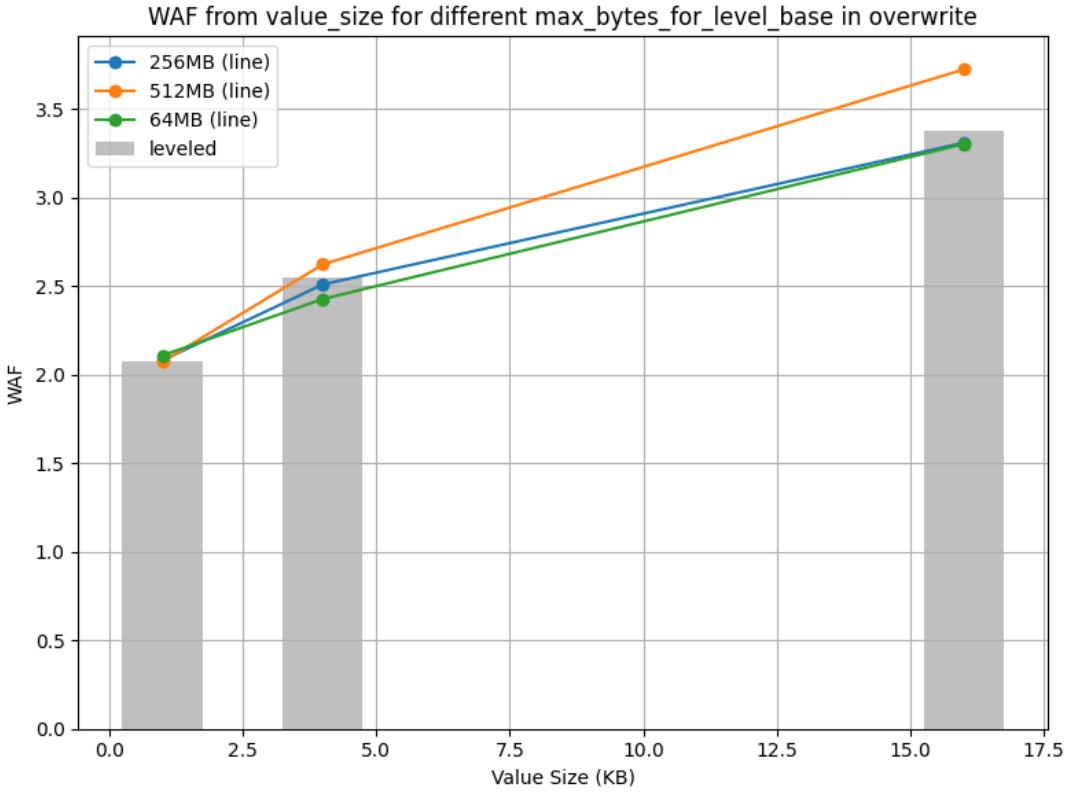
실험 결과

max_bytes_for_level_base

fillrandom



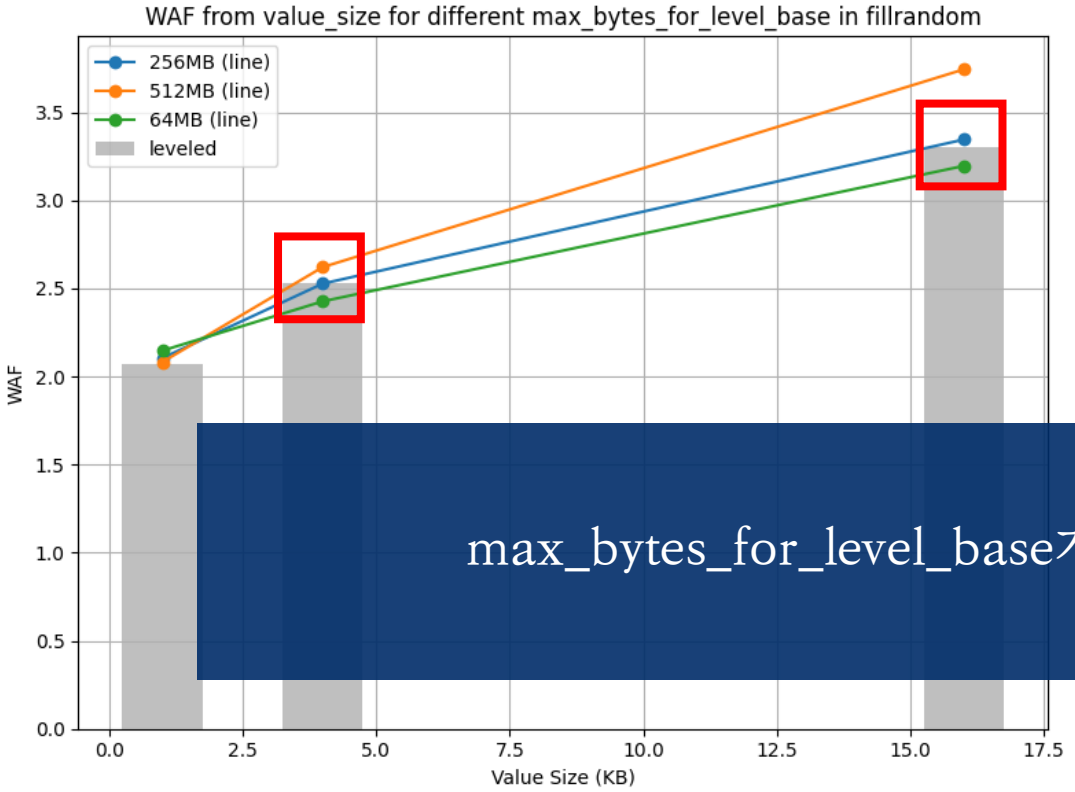
overwrite



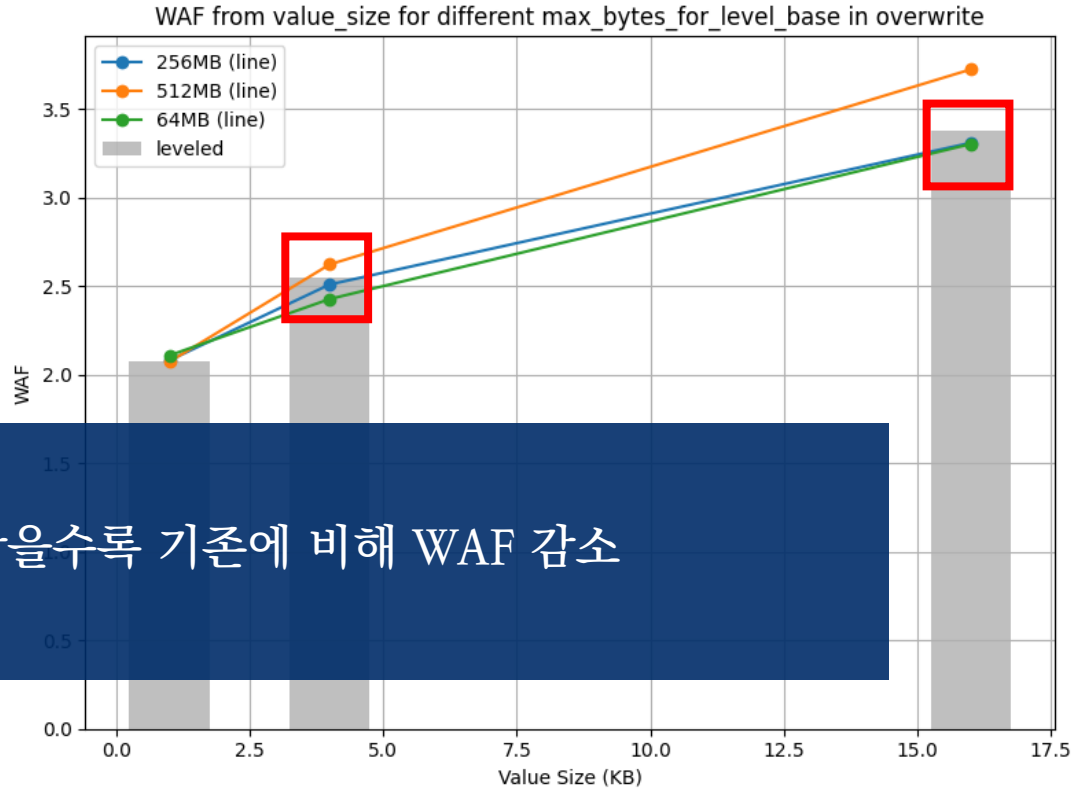
실험 결과

max_bytes_for_level_base

fillrandom



overwrite

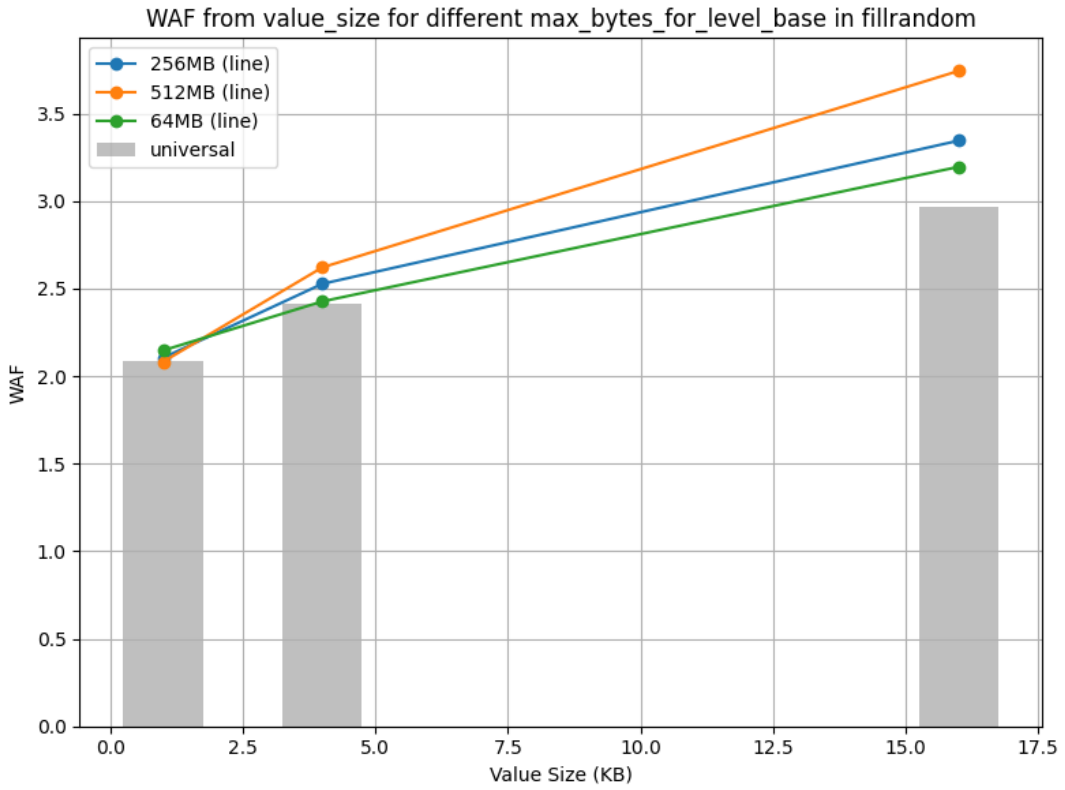


max_bytes_for_level_base가 작을수록 기존에 비해 WAF 감소

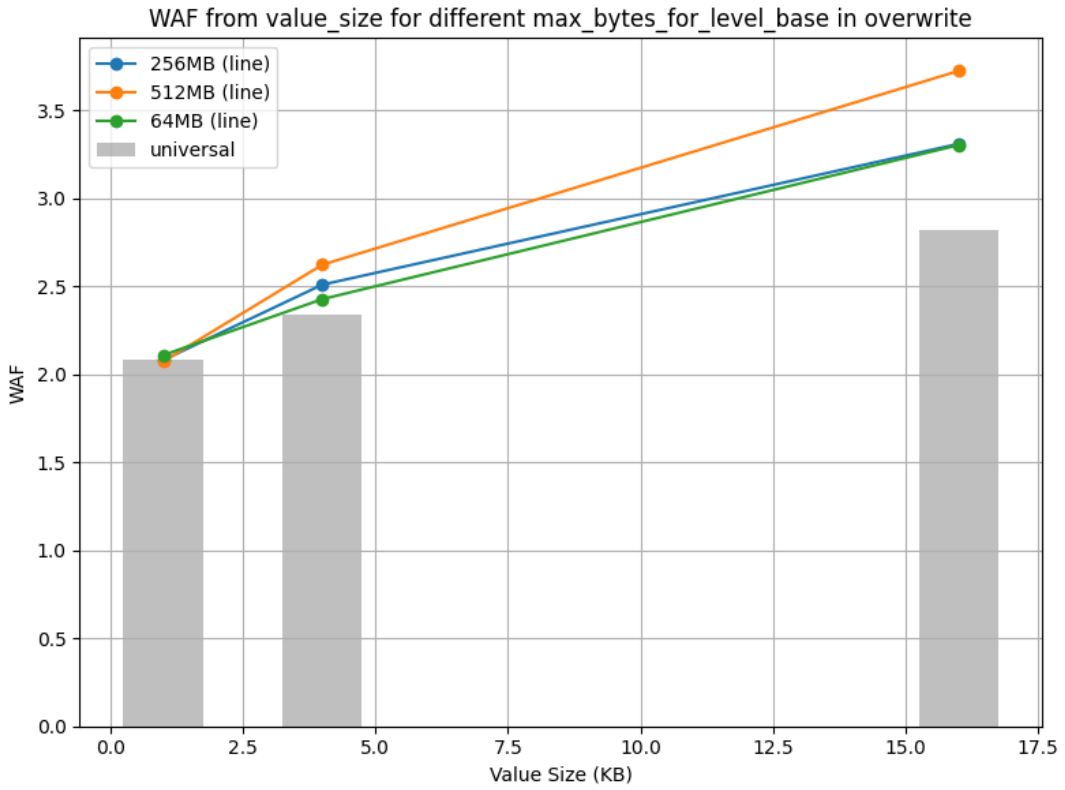
실험 결과

max_bytes_for_level_base

fillrandom



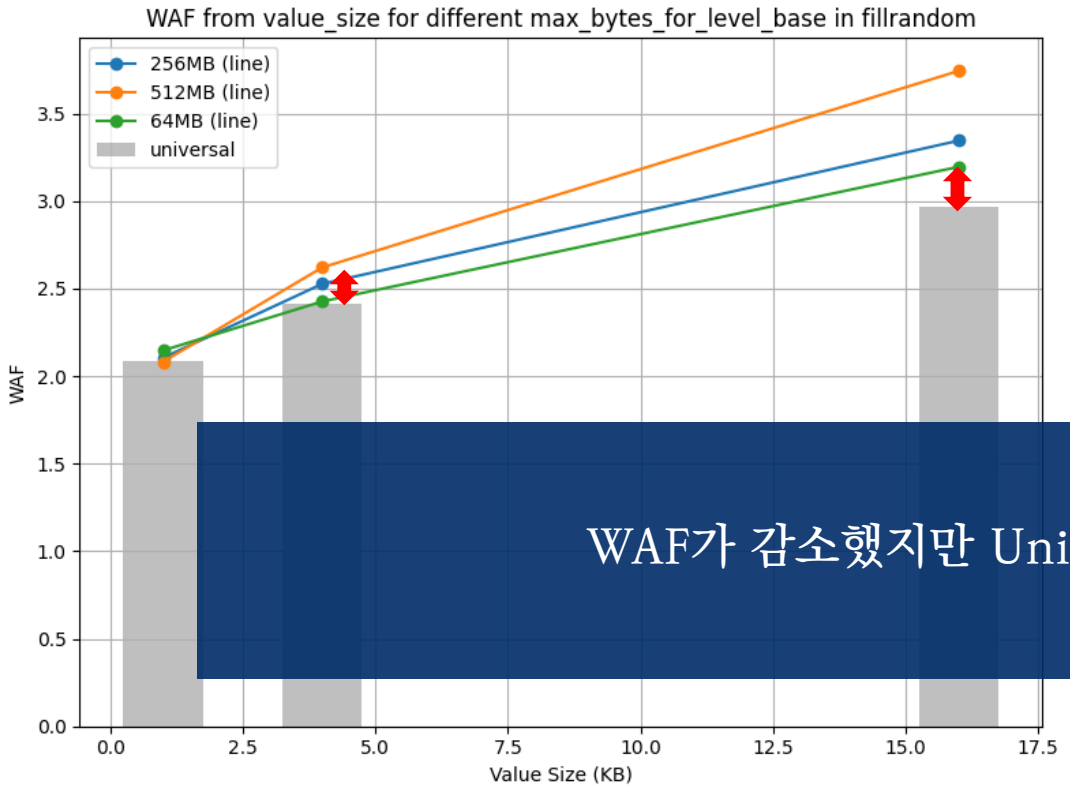
overwrite



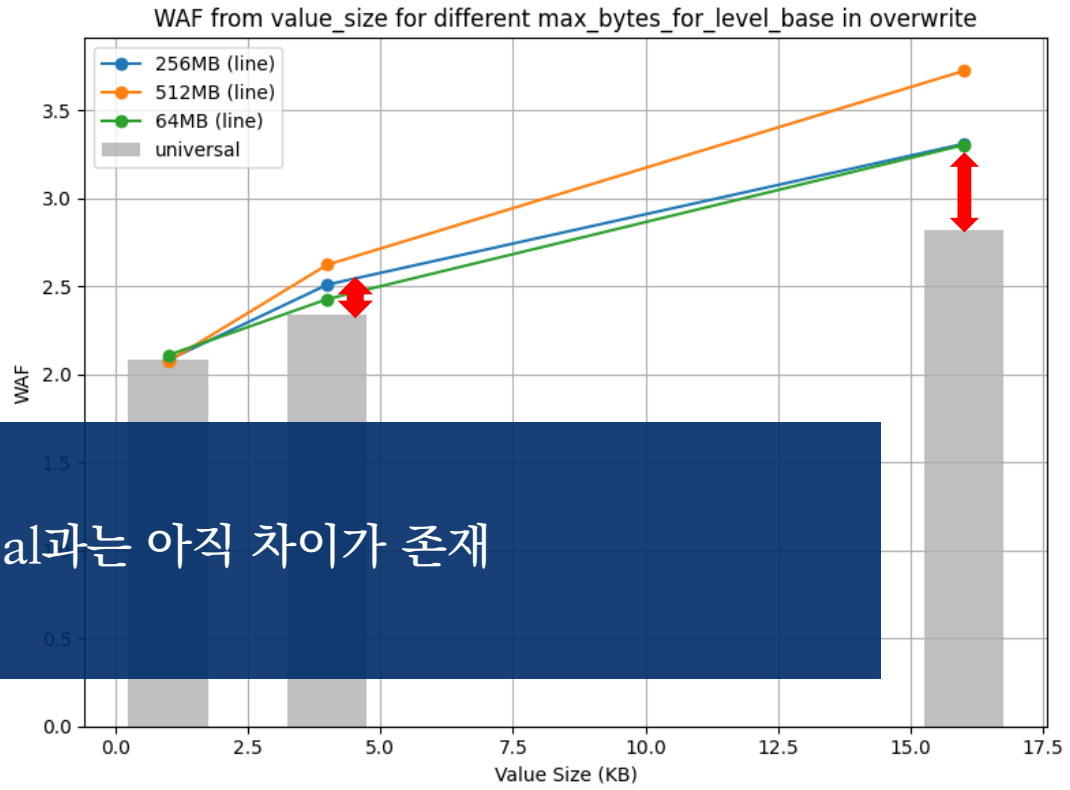
실험 결과

max_bytes_for_level_base

fillrandom



overwrite

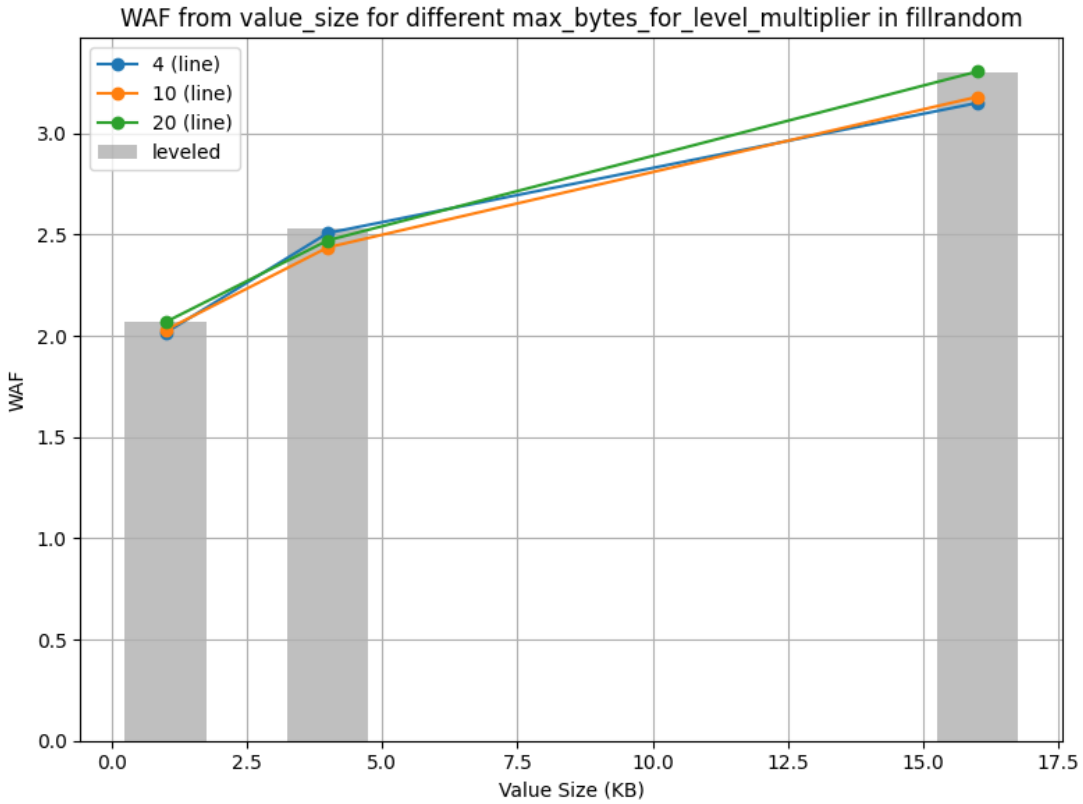


WAF가 감소했지만 Universal과는 아직 차이가 존재

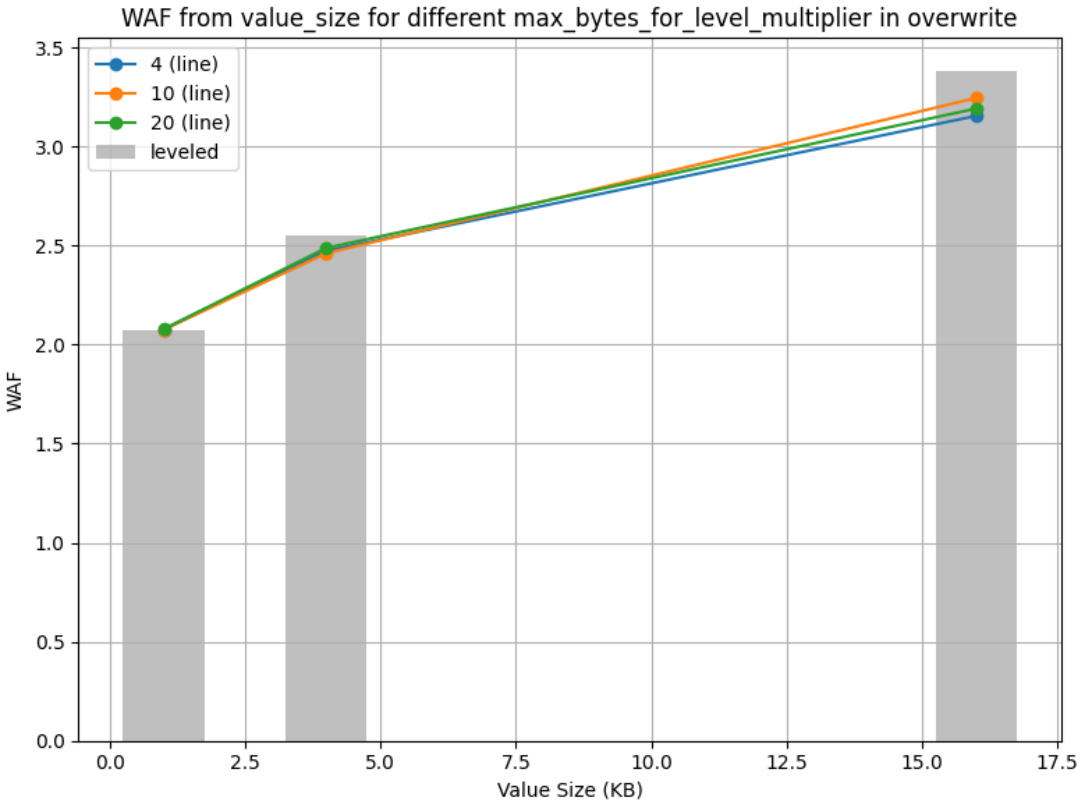
실험 결과

max_bytes_for_level_multiplier

fillrandom



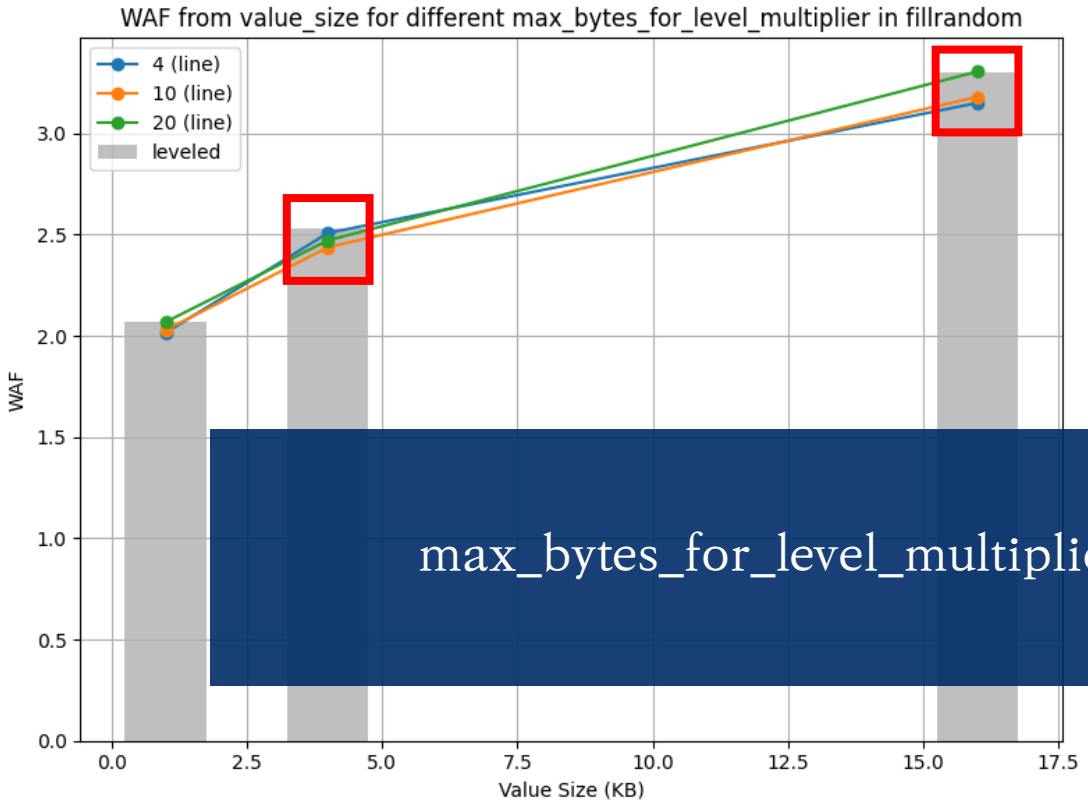
overwrite



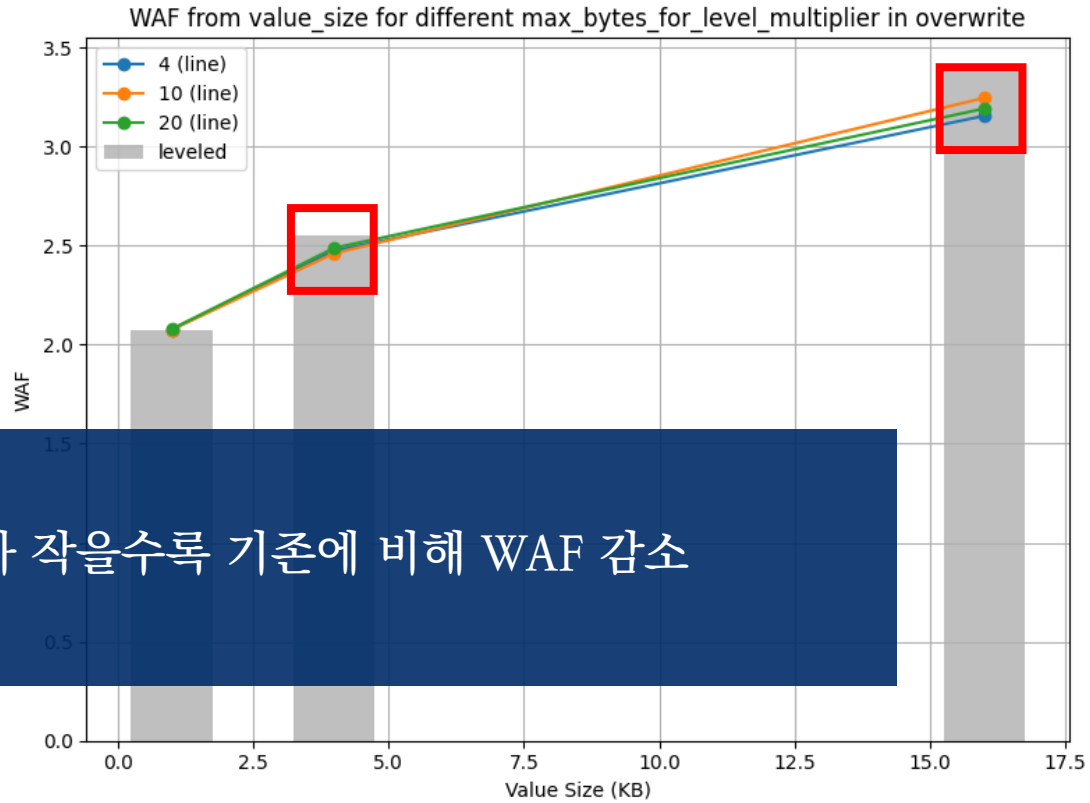
실험 결과

max_bytes_for_level_multiplier

fillrandom



overwrite

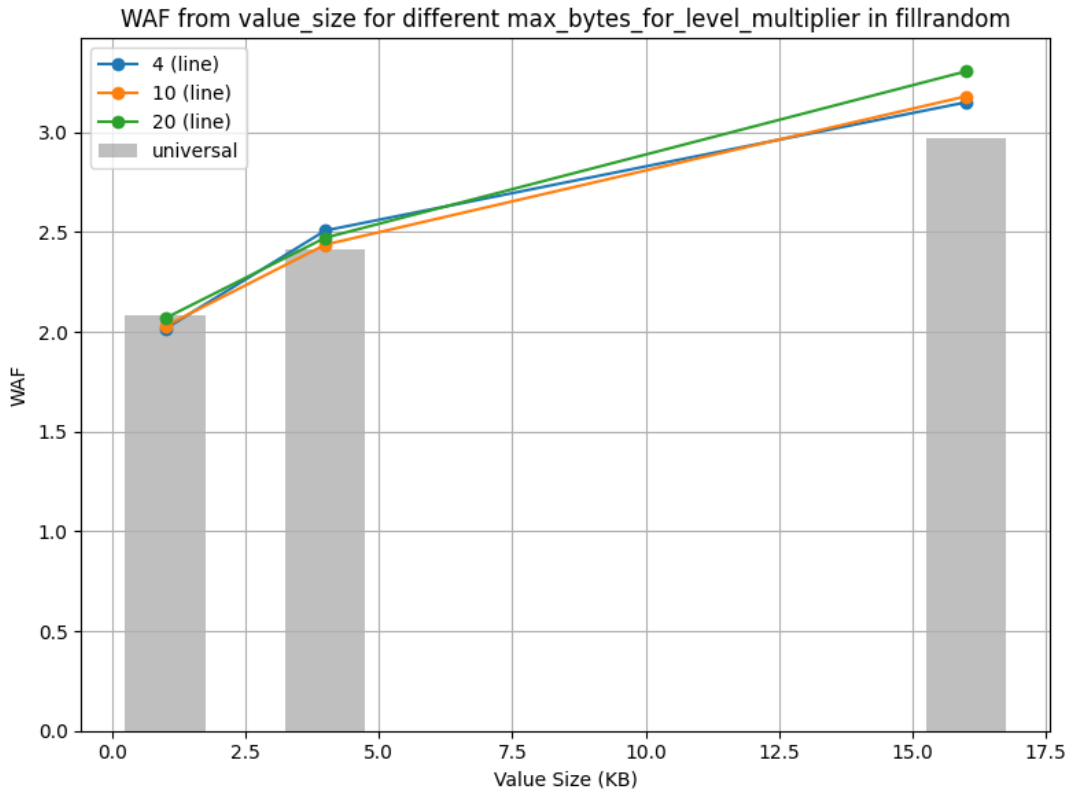


max_bytes_for_level_multiplier가 작을수록 기존에 비해 WAF 감소

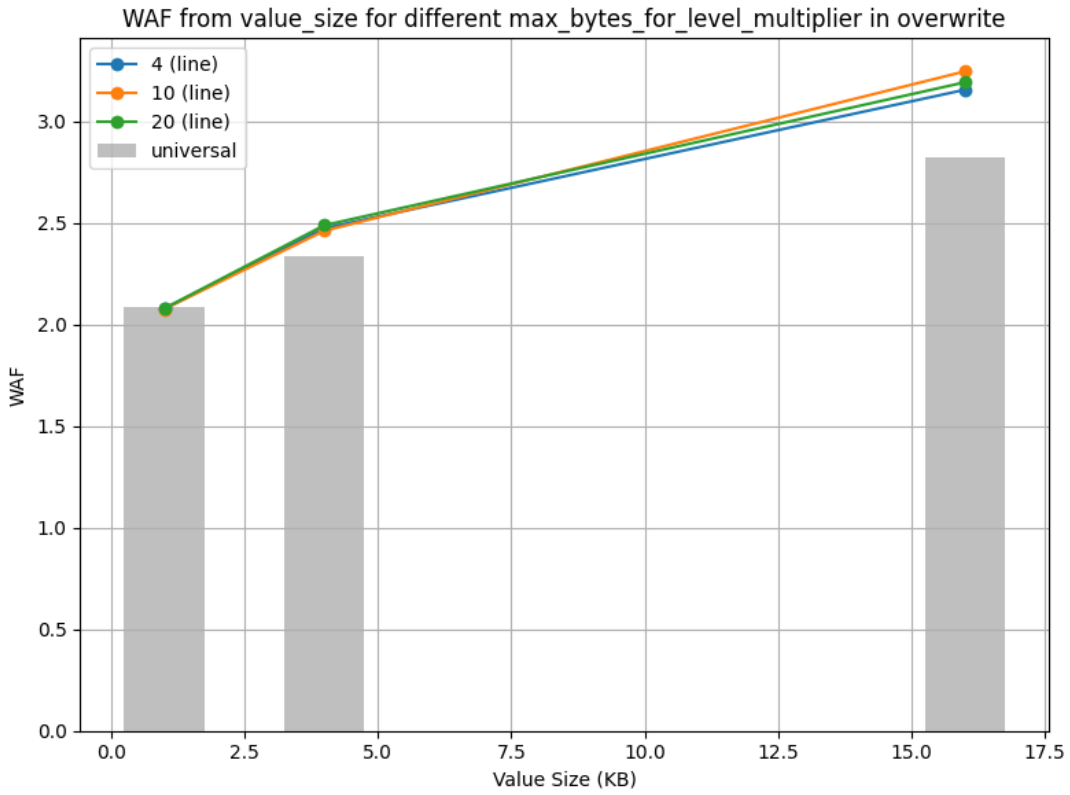
실험 결과

max_bytes_for_level_multiplier

fillrandom



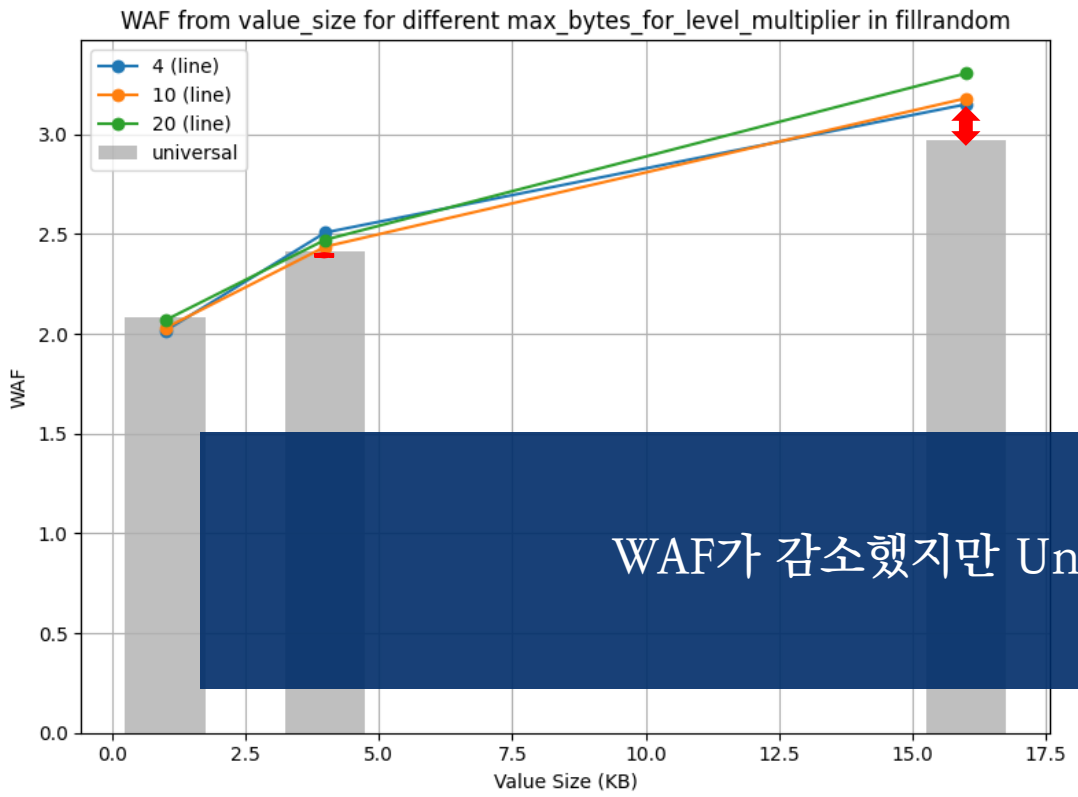
overwrite



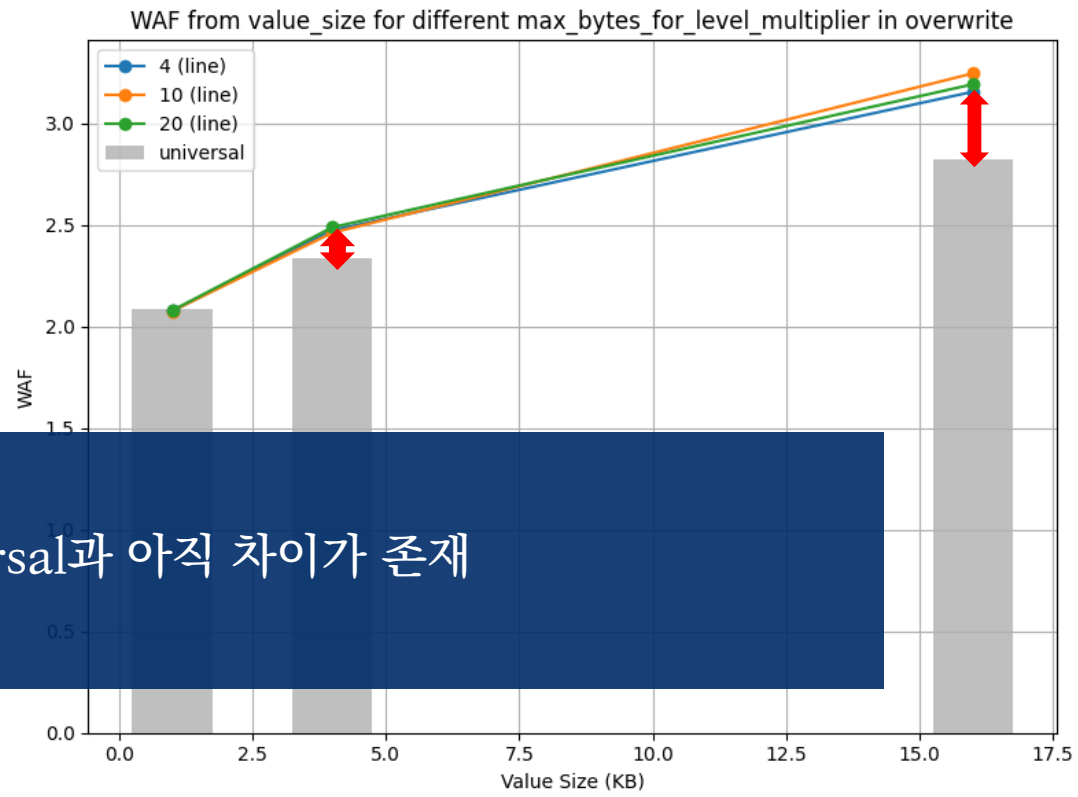
실험 결과

max_bytes_for_level_multiplier

fillrandom



overwrite



WAF가 감소했지만 Universal과 아직 차이가 존재

04

결론

- 실험 결과 요약
- 고찰 및 향후 연구

실험 결과 요약

가설 1

가설 채택

WAL 설정과 Compaction 전략 간의 상호작용은 WAF에 유의미한 영향을 미칠 수 있다.

➡ perf의 경우 stable보다 WAF 감소

WAL 사용

- 안전하게 데이터 복구
- Flush 자주 사용
- 작고 중복 많은 SST 파일
- WAF 증가

WAL 미사용

- 데이터 손실 감수
- Flush를 늦게 함
- 적고 큰 SST 파일
- WAF 낮음

실험 결과 요약

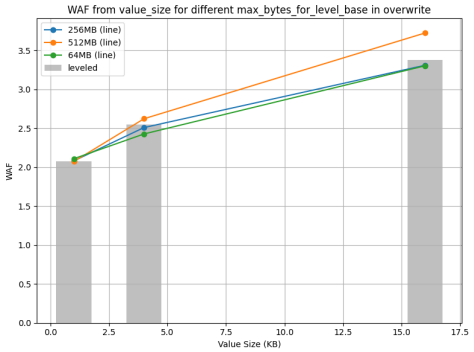
가설 2

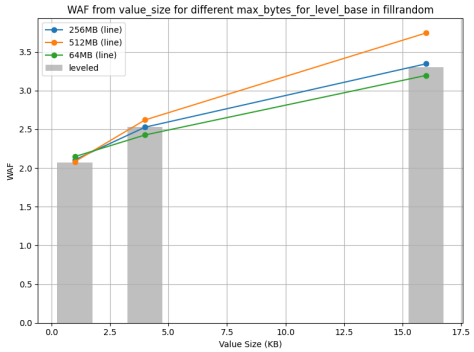
추가 실험 및 분석 필요

Update 및 Delete가 빈번한 워크로드에서 Leveled Compaction의 compaction 빈도를 최적화할 경우,

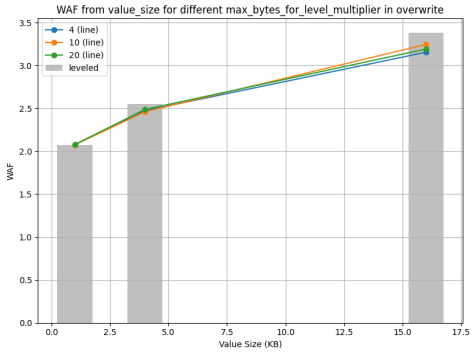
Universal Compaction과 비슷한 수준의 WAF 성능을 달성할 수 있다.

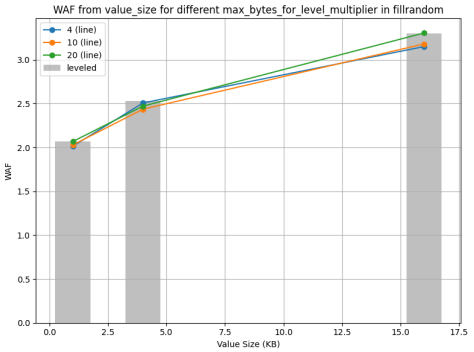
max_bytes_for_level_base





max_bytes_for_multiplier





→ max_bytes_for_level_base와 max_bytes_for_multiplier가 낮으면 WAF가 낮다

고찰 및 향후 연구

고찰

분석 과정
• 시간 부족으로 인해 설계 및 준비가 미흡 • 실험을 여러 번 돌려 충분히 검증하지 못함

다음 실험에 보완할 부분
• RocksDB 구조 추가 조사로 실험 결과 원인 찾기 • 반복 실험으로 평균값 도출

고찰 및 향후 연구

향후 연구

Column Family
• 가설 1번에서 착안한 가설 • Write 집중 워크로드에서 hot-cold 데이터 구분 저장 • Column family 별로 Compaction 정책 다르게 설정

Hot Data	Cold Data
Leveled Compaction	Universal Compaction

05

참고 문헌

참고문헌

[1] 2.4.SSTable . (2022). https://sslab.dankook.ac.kr/2-4-sstable/.

[2] Leveled Compaction. (2023). https://github.com/facebook/rocksdb/wiki/Leveled-Compaction.

[3] Write Ahead Log. (2023). https://github.com/facebook/rocksdb/wiki/Write-Ahead-Log-(WAL).

[4] 신호진, 최종무. (2021). RocksDB에서 집약과 WAL의 성능 특성 분석. 2021년 한국 컴퓨터종합학술대회 논문집, pp.1470-1472.

[5] 조민수, 최원기, 박상현. (2017). RocksDB에서 SST 파일에 따른 WAF 감소에 관한 연구. 2017년 추계학술발표대회 논문집, 24(2), pp. 709-712.

Thank you

Thank you for listening to my presentation