K2vTune: A Workload-aware Configuration Tuning for RocksDB

연세대학교 컴퓨터과학과 이지은

2023년 10월

과제명: IoT 환경을 위한 고성능 플래시 메모리 스토리지 기반 인메모리 분산 DBMS 연구개발

과제번호: 2017-0-00477

과학기술정보통신부
Ministry of Science and ICT

연세대학교
YONSEI UNIVERSITY

정보통신기술진흥센터
Institute for Information & communications Technology Promotion

Outline

01

Introduction

- Motivation
- Introduction

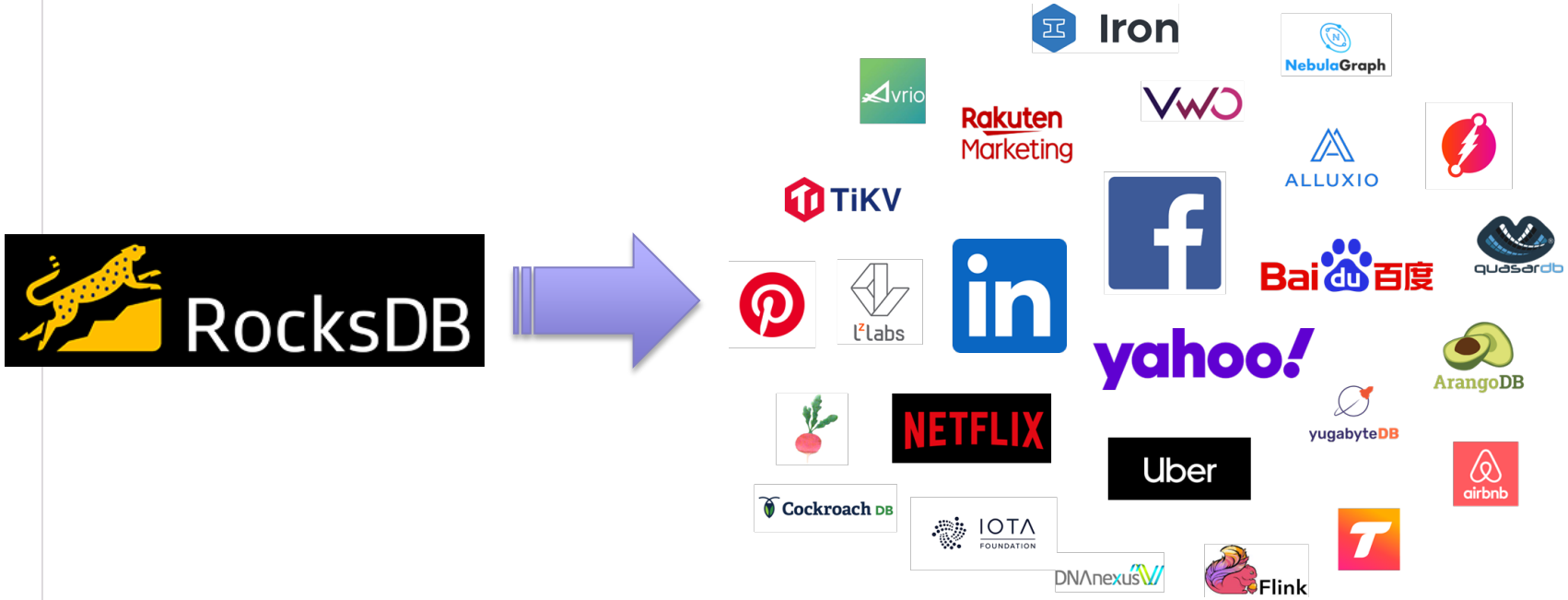
Motivation

- 데이터베이스는 데이터의 체계적인 관리를 위해 많은 기업에서 활용되고 있으며, 빅데이터의 사용이 증가함에 따라 다양한 종류의 데이터베이스 시스템(DBMS)가 개발됨

- 비정형 데이터를 효율적이게 처리할 수 있는 NoSQL DBMS가 여러 기업에서 storage engine으로 이용 (ex. RocksDB)

- RocksDB는 디스크 기반의 키-값 임베디드 데이터베이스이며, 로그 병합 트리(LSM-Tree) 구조를 사용하기 때문에 빠른 데이터 쓰기 성능을 보이는 것이 특징임

Usecases of RocksDB



https://github.com/facebook/rocksdb/wiki/RocksDB-Users-and-Use-Cases

Motivation

- 하지만 로그 병합 트리 구조로 인해 쓰기 증폭(Write Amplification)과 공간 증폭(Space Amplification)의 문제를 고려해야함
- 효율적인 사용을 위해서는 앞의 증폭뿐만 아니라 데이터 처리량(Throughput) 등의 RocksDB 성능을 최적화하는 과정이 필요
- **쓰기 증폭:** 불필요한 추가적인 쓰기 작업 비율을 나타내며, 높게 측정될 수록 불필요한 쓰기 작업으로 인해 플래시 메모리 수명을 저하시키며 RocksDB 성능 또한 저하시킴
- **공간 증폭:** 불필요한 데이터 적재 공간 비율을 나타내며, 높게 측정될 수록 쓸모 없는 데이터로 인해 공간 낭비가 되고 있는 것을 뜻함

Introduction

- RocksDB 에는 RocksDB의 동작 방식 및 구조를 조정할 수 있는 약 200개의 노브(Knob)가 제공됨
- 사용자는 노브들을 조정하여 요청하고자 하는 워크로드에 따라서 적절히 수정이 가능하여 원하는 성능을 도출할 수 있음
- 하지만 워크로드에 따라서 최적의 성능을 도출할 수 있는 노브 수치들이 상이하기 때문에 매번 DBA가 튜닝을 해야하는 어려움이 있음
- 본 연구는 이를 해결하기 위해 딥러닝을 활용하여 워크로드에 따른 RocksDB 튜닝 기법을 제안함

Introduction

- 기존 연구에서는 다음과 같은 흐름을 따르며, 사용자가 원하는 성능을 도출할 때까지 1-3의 과정을 반복함

  1. 임의의 configuration 생성
  2. 생성된 configuration으로 RocksDB의 성능을 측정
     - 측정하는 방법은 벤치마킹 툴을 사용하거나 머신러닝 혹은 딥러닝 기반의 예측 모델 사용
  3. 측정된 성능을 비교하여 최적의 성능을 도출하는 configuration을 추천함

- 위의 과정에서 2번의 과정이 중요한 역할을 함

- 위의 흐름을 최소 100번 등의 반복을 진행하기 때문에 1번 측정하는 데에도 시간이 많이 소요되는 벤치마킹 툴을 사용하는 것은 튜닝 연구에서 비효율적이기 때문에 예측 모델을 사용하는 연구가 많이 진행되고 있음

- 또한, 예측모델을 사용한다면 벤치마킹 툴을 충분히 대체할 수 있는 정확도를 가져야 함

[용어 설명]

- **워크로드(Workload):** 데이터를 처리하기 위해 요청되는 작업 환경임. 데이터의 읽기 쓰기 비율에 따라서 워크로드가 달라지며, 워크로드가 읽기 쓰기를 처리할 때에 사용하는 데이터 양 및 데이터베이스의 크기가 달라져도 워크로드가 다른 것.
- **노브(Knob):** 데이터베이스에서 제공하는 파라미터이며, 해당 값을 조절함으로써, 데이터베이스 구조를 변경할 수 있음. 예시로 데이터베이스에서 사용하는 cpu core 개수, 최대 사용 가능한 memory 용량, 데이터베이스 구조에서 사용하는 node의 최대 크기 및 개수 등 다양한 요소를 정의해 줌. 성능에 직접적으로 영향을 주는 knob 들이 많음.
- **Configuration:** Knob의 집합
- **내부 메트릭스(Internal metrics):** 데이터베이스가 요청을 처리하면서 측정된 내부 통계치 (rocksdb.block.cache.hit count, rocksdb.memtable.miss COUNT 등)
- **외부 메트릭스(External metrics):** 데이터베이스 성능을 나타내는 지표 (요청을 처리하는 데 걸리는 시간, throughput, 시간 증폭, 공간 증폭)

[참고]

- 데이터베이스 성능을 측정하는 것을 벤치마킹한다고 표현하며, 이 때에 워크로드를 정의하여 성능을 측정함
  ➔ 예를 들면 읽기를 많이 요청하는 상황으로 워크로드를 정의하여 RocksDB의 성능을 측정함

- 치마킹 툴을 사용하여 임의의 configuration으로 RocksDB를 정의하였을 때의 성능을 측정하며, 측정된 결과 값들은 외부 내부 메트릭스로 구분하여 저장함. 그리고 이러한 성능을 측정하는 과정은 시간이 많이 소요됨.

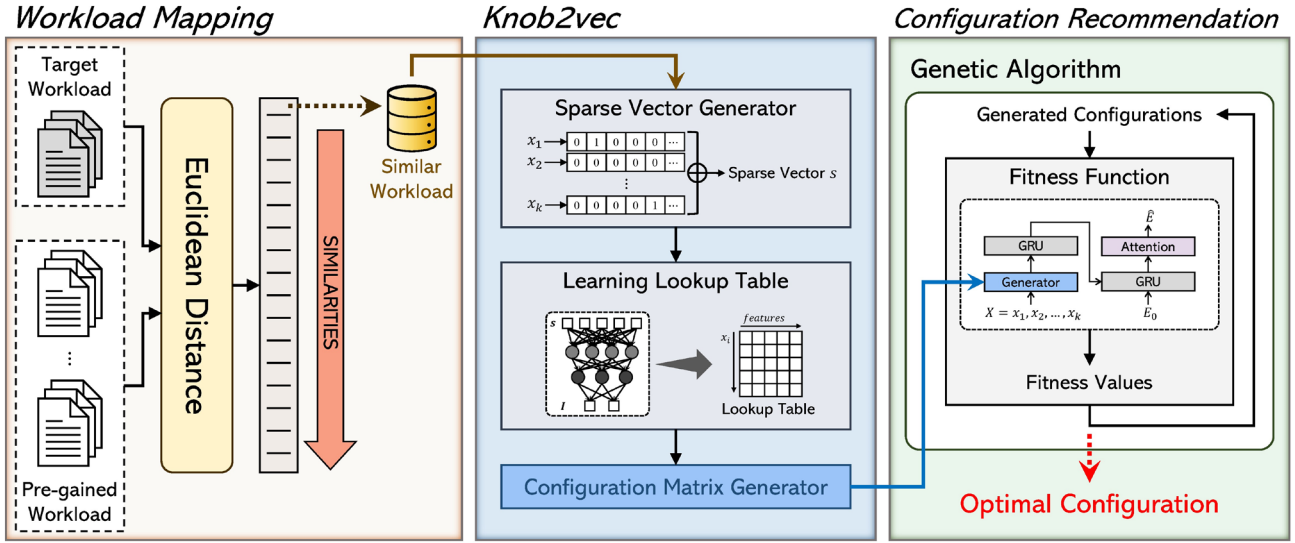
02

Method

- Method
- Workload Mapping
- Knob2vec
- Configuration Recommendation

Method

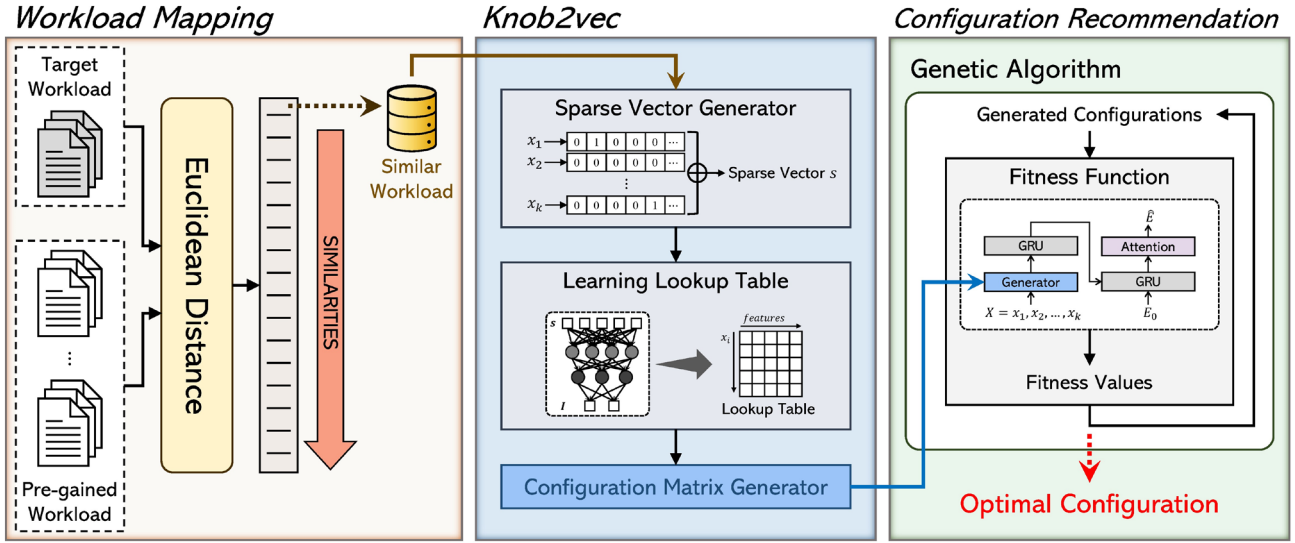
- 본 연구는 사용자가 요구하는 워크로드에 따라서 RocksDB의 노브를 자동으로 튜닝하여 성능을 최적화시키는 것이 목표임. 워크로드의 특성에 따라서 최적의 파라미터 수치가 변화하기 때문에 워크로드에 따라서 매번 튜닝을 새로 해야 하기 때문에 자동 튜닝 기법을 제안함



Method

- 모델의 3가지 단계

1. Workload Mapping
2. Knob2vec
3. Configuration Recommendation

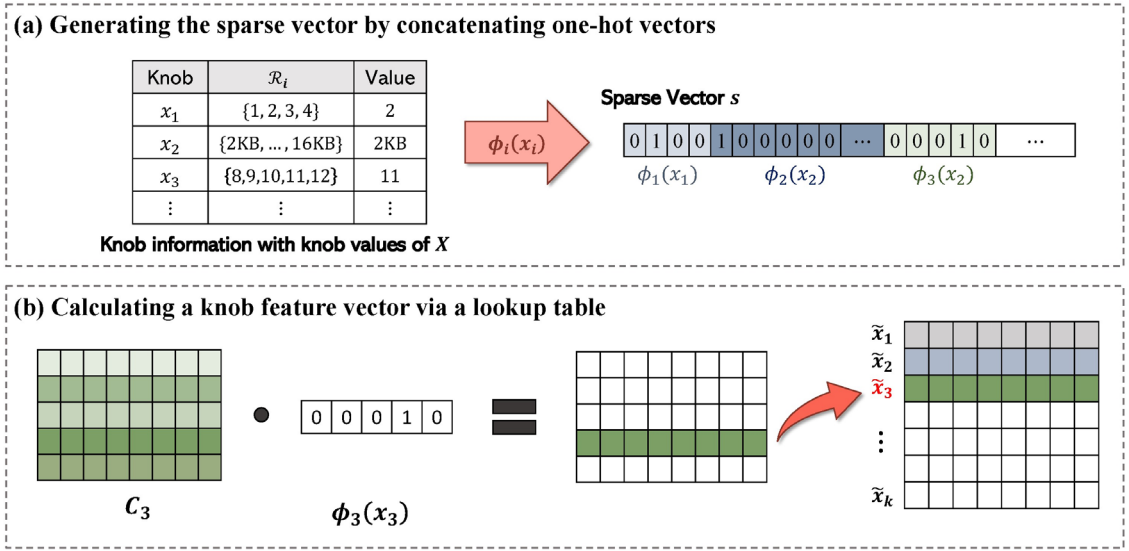


1. Workload Mapping

- 사용자가 성능을 최적화하고자 하는 워크로드(target workload)와 비슷한 성격을 가지는 워크로드를 찾는 과정
- 비슷하다고 판명나는 워크로드의 샘플들을 타겟 워크로드 샘플로 가정하여 뒤의 단계에서 딥러닝 모델을 학습하는 데이터셋으로 활용됨
- 데이터는 [configuration, internal metrics, external metrics] 로 구성됨
- internal metrics를 Euclidean distance 계산을 통해 유사도를 측정하여 target 워크로드와 가장 유사한 워크로드를 찾음

2. Knob2vec

- 임의의 Configuration으로 RocksDB를 벤치마킹했을 때 측정되는 과정이 시간이 많이 소요되기 때문에 이를 대체할 수 있는 정확도가 높은 예측 모델이 필요함. 정확도를 향상시키기 위해 노브를 벡터로 표현할 수 있는 방법론을 제안함



2. Knob2vec

- 입력되는 값은 Configuration, 즉 Knob의 값임. 그러나 Knob의 종류는 다양하며 가질 수 있는 값의 종류가 다름
- Boolean, Categorical, Numerical 등 다양한 형태의 값을 가지기 때문에 지정하는 값들에 대해 RocksDB는 다르게 인식됨
- 예를 들면, Knob1은 Boolean 값을 가지고, Knob2는 Categorical 값을 가지고 Knob3은 Numerical 값을 가진다고 가정하였을 때, 같은 값에 대해서 데이터베이스가 인식하는 의미는 다음과 같음

Knob1: cache_index_and_filter_blocks ➔ true

Knob2: compaction_pri ➔ kByCompensatedSize

Knob3: write_buffer_size ➔ 1 Byte

- 이렇듯 노브에 따라서 지정해주는 수치를 고유하게 표현하고, 또한 추가적인 정보를 제공하기 위해서 노브에서 내부 메트릭스를 예측하는 과정에서 학습한 가중치를 활용하는 노브 벡터 표현형 방법론(Knob2vec)을 제안함
- 내부 메트릭스는 벤치마킹 실행 중에 측정된 통계치들이기 때문에 워크로드의 특성을 포함한다고 볼 수 있음
- 그러므로 해당 데이터로 학습하여 생성한 노브 벡터는 워크로드 특성 및 정보를 포함한다고 볼 수 있음

3. Configuration Recommendation

- 최적의 configuration 을 추천하기 위해서 유전알고리즘(Genetic Algorithm)을 사용함.
  유전알고리즘은 다음과 같은 절차를 가짐.

1. 유전알고리즘을 통해 후보 configuration(Generated configurations)을 생성

2. 생성된 configuration 을 Fitness function에서 평가함
   - 딥러닝 기반의 예측 모델을 사용하여 configuration 으로 측정한 RocksDB 성능(외부 메트릭스)을 예측함
   - 예측된 결과는 fitness value 로 지정됨

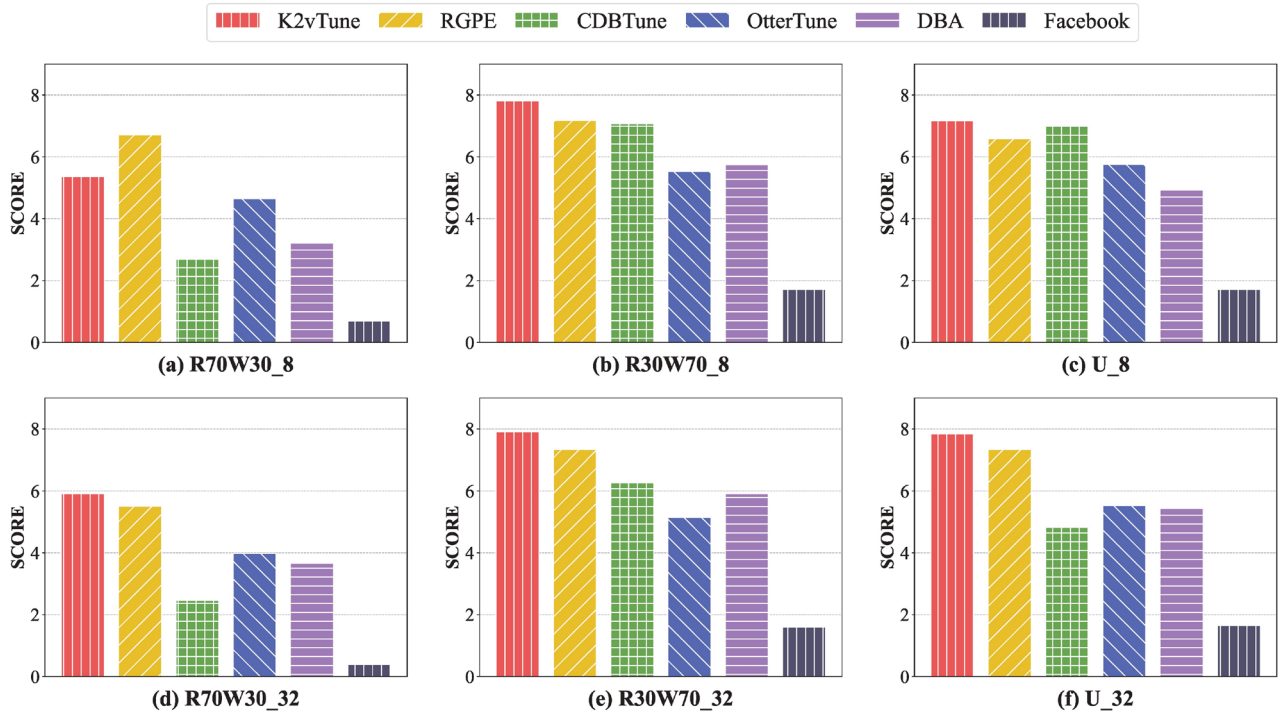
3. 후보 configuration 들의 fitness value 를 비교하여 가장 좋은 성능을 도출하는 configuration 을 최종적으로 추천함

03

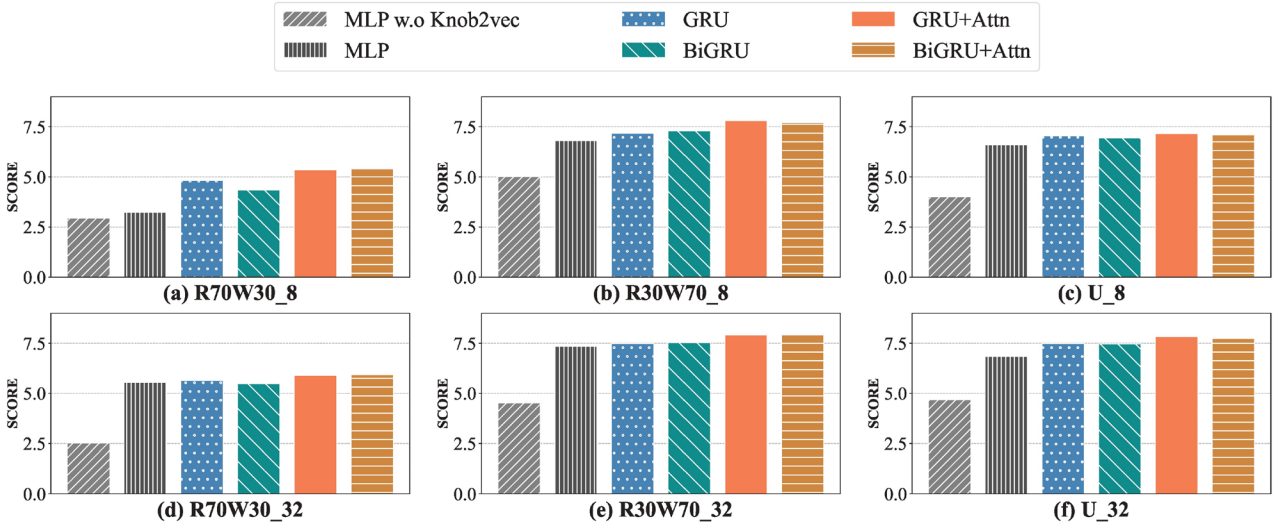
Experimental Results

- Motivation
- Introduction

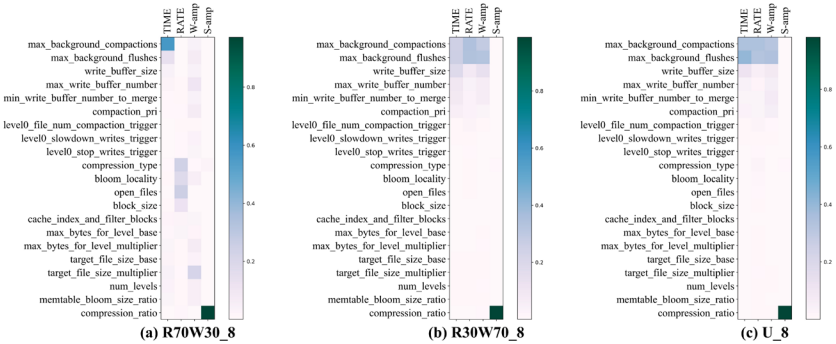
Experimental Results



Experimental Results

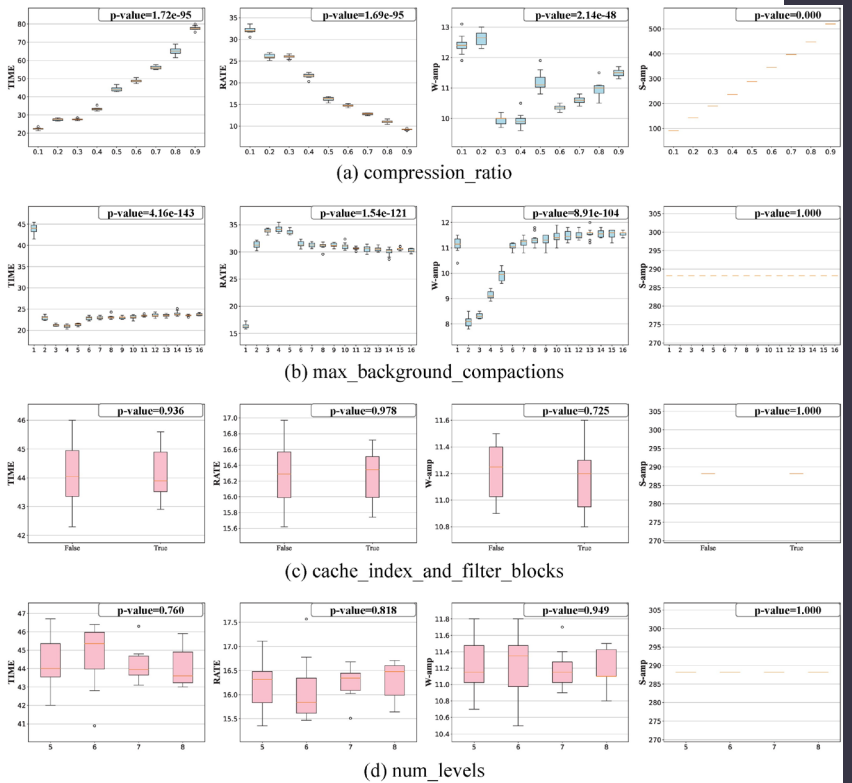


Experimental Results



(a) R70W30_8 (b) R30W70_8 (c) U_8

(d) R70W30_32 (e) R30W70_32 (f) U_32



Thank you!

Published on
https://www.sciencedirect.com/science/article/pii/S0306457323003047