Xiao ZHANG, Mengyu LI, Michael NGULUBE, Yonghao CHEN, Yiping ZHAO. MyWAL: performance optimization by removing redundant input/output stack in key-value store. *Frontiers of Information Technology & Electronic Engineering*, 24(7):980-993. https://doi.org/10.1631/FITEE.2200496

MyWAL: performance optimization by removing redundant input/output stack in key-value store

Key words: Key-value (KV) store; Log-structured merge (LSM) tree; Non-volatile memory (NVM); Non-volatile memory express soild-state drive (NVMe SSD); Write-ahead log (WAL)

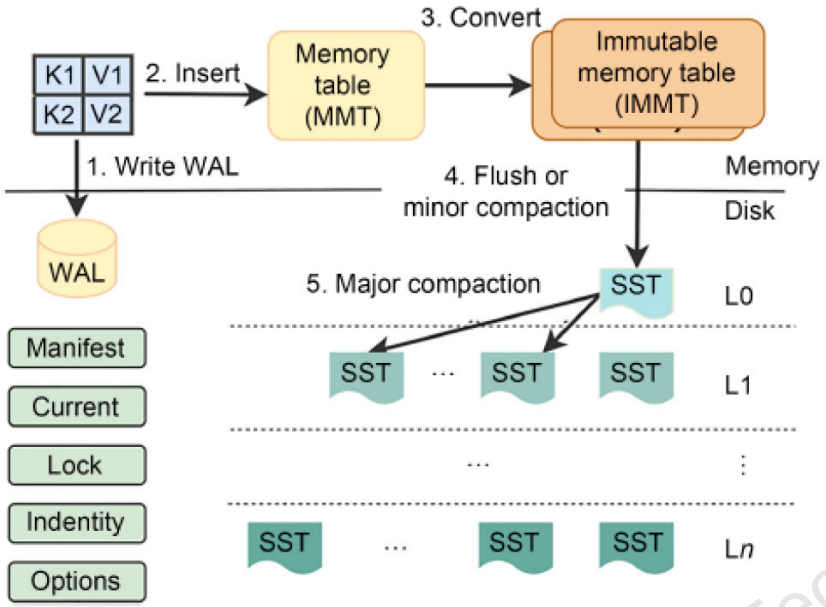
Corresponding author: Xiao ZHANG
E-mail: zhangxiao@nwpu.edu.cn

ORCID: http://orcid.org/0000-0001-7680-1179

Motivation

- RocksDB saves new data in the memory, achieving much higher performance than the disk-based database. However, the data in the memory can be lost if there is a power failure or operating system (OS) crash without write-ahead log (WAL).
- The WAL is an essential mechanism to ensure data availability. Unfortunately, writing WAL reduces the RocksDB writing performance.
- In this paper, we propose a new WAL mechanism named MyWAL, which manages and saves WAL data to the raw device. It removes unnecessary input/output (IO) operations caused by the file system and improves the writing performance of key-value (KV) data on different storage media.

Background



Architecture of the LSM tree used in RocksDB

- System architecture
  - New and updated KV data are saved in the memory table (MMT), which is a readable and append-write file.
  - RocksDB saves KV data in a WAL file before it updates in the MMT.
  - WAL is important to keep data consistency under abnormal crash.

- WAL cost
  - New and updated data are saved in the memory except WAL.
  - Writing WAL is the bottleneck of updating process.

- File system cost
  - When RocksDB updates WAL, the lower file system also updates metadata of the WAL.
  - The writing position of metadata and WAL are different.
  - Many metadata are meanless for RocksDB such as update time.

Performance loss caused by WAL

- The thoughtput of no-WAL is 7%–200% higher than that of async-WAL and 11 to 210 times higher than that of sync-WAL.
- The operations per second (OPS) of no-WAL is is 4%–212% times higher than that of async-WAL and 11.36 to 246.92 times higher than that of sync-WAL.

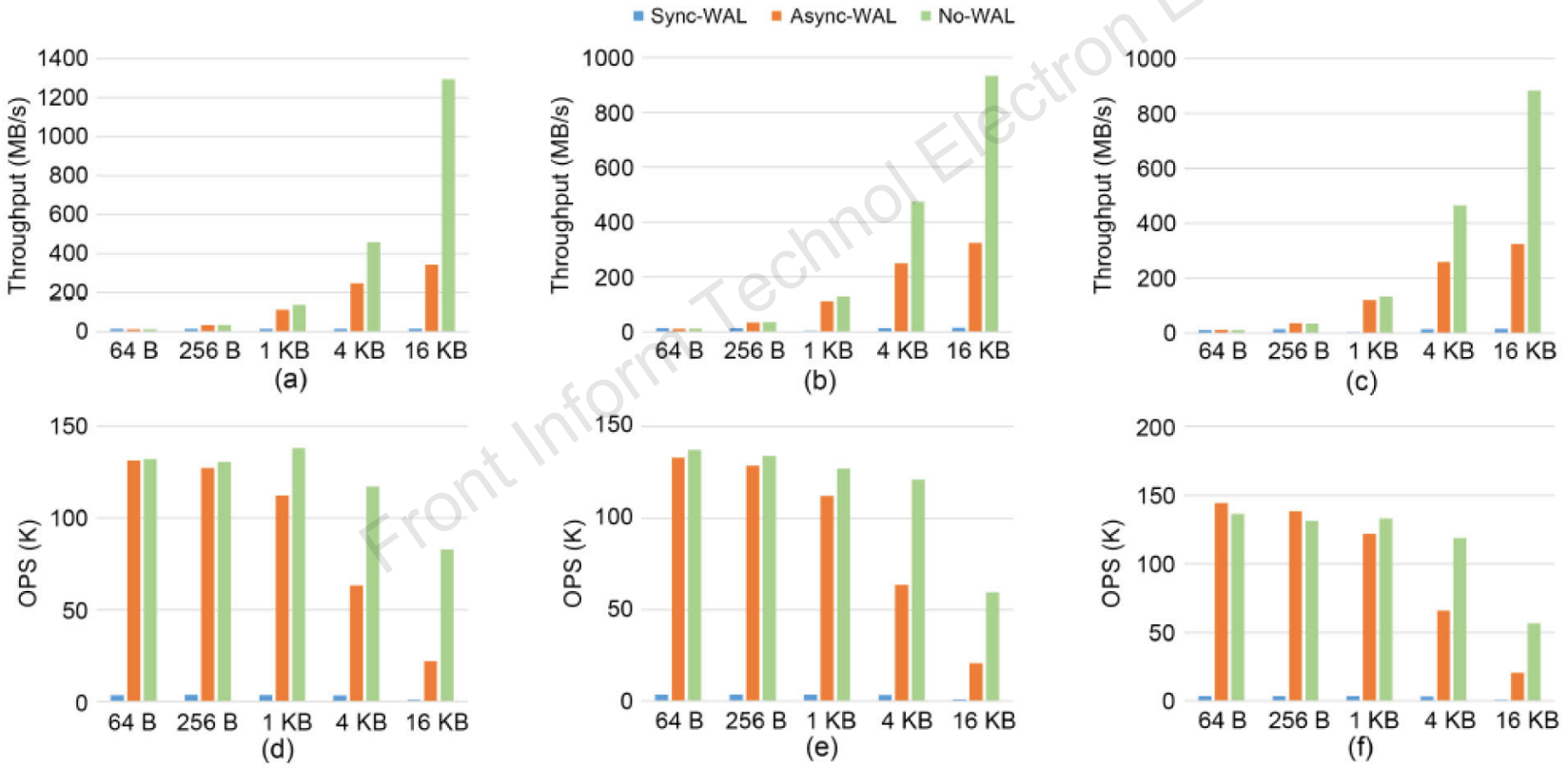


Fig. 3 Performance loss caused by WAL: (a) fillseq throughput; (b) fillrandom throughput; (c) overwrite throughput; (d) fillseq OPS; (e) fillrandom OPS; (f) overwrite OPS (WAL: write-ahead log; OPS: operations per second. The horizontal axis represents the size of each data written by db_bench)

Performance between raw device and Ext4

- We compare the sequential and random writing bandwidth between raw device and Ext4.
- For SSD, the bandwidth of Ext4 is 5% less than that of the raw device.
- For NVMe SSD, the bandwidth of Ext4 is 6%–16% less than that of the raw device.
- For NVM, the bandwidth of Ext4 is 50%–70% less than the that of raw device.

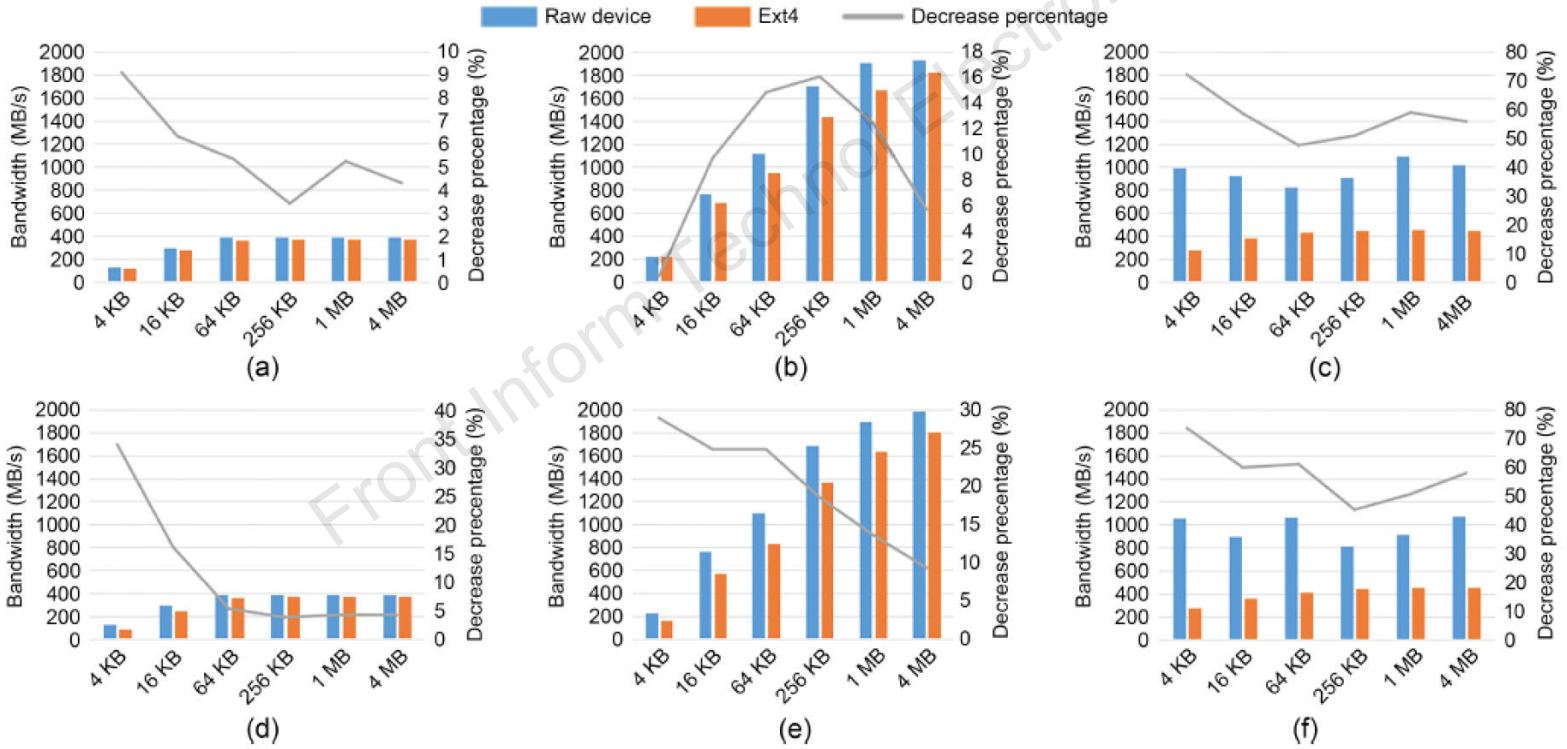


Fig. 4 Comparison of writing performance between raw devices and Ext4: (a) sequential write (SSD); (b) sequential write (NVMe SSD); (c) sequential write (NVM); (d) random write (SSD); (e) random write (NVMe SSD); (f) random write (NVM) (The horizontal axis represents the size of each data written by FIO)

Design and implementation

- WAL is an append-only file, and there is only one active WAL file when the RocksDB is running.
- The key feature of MyWAL:
  - Save WAL files to the raw device directly without file system.
  - Reconstruct the format of metadata in the disk.
  - Add a linked list during WAL updating. It is unnecessary to update the size and end position of the WAL file.
  - Design a method to recover WAL file under abnormal crash.

Management of space and WAL files

- MyWAL saves the metadata and data directly to the raw device without file system.
- The metadata include WAL number, flag, time stamp, previous file, and start position. Note that MyWAL does not save file length in the metadata.
- When MyWAL creates a new WAL file, it appends the new WAL after the previous file. The head positon of the new file is the end position of the previous file.

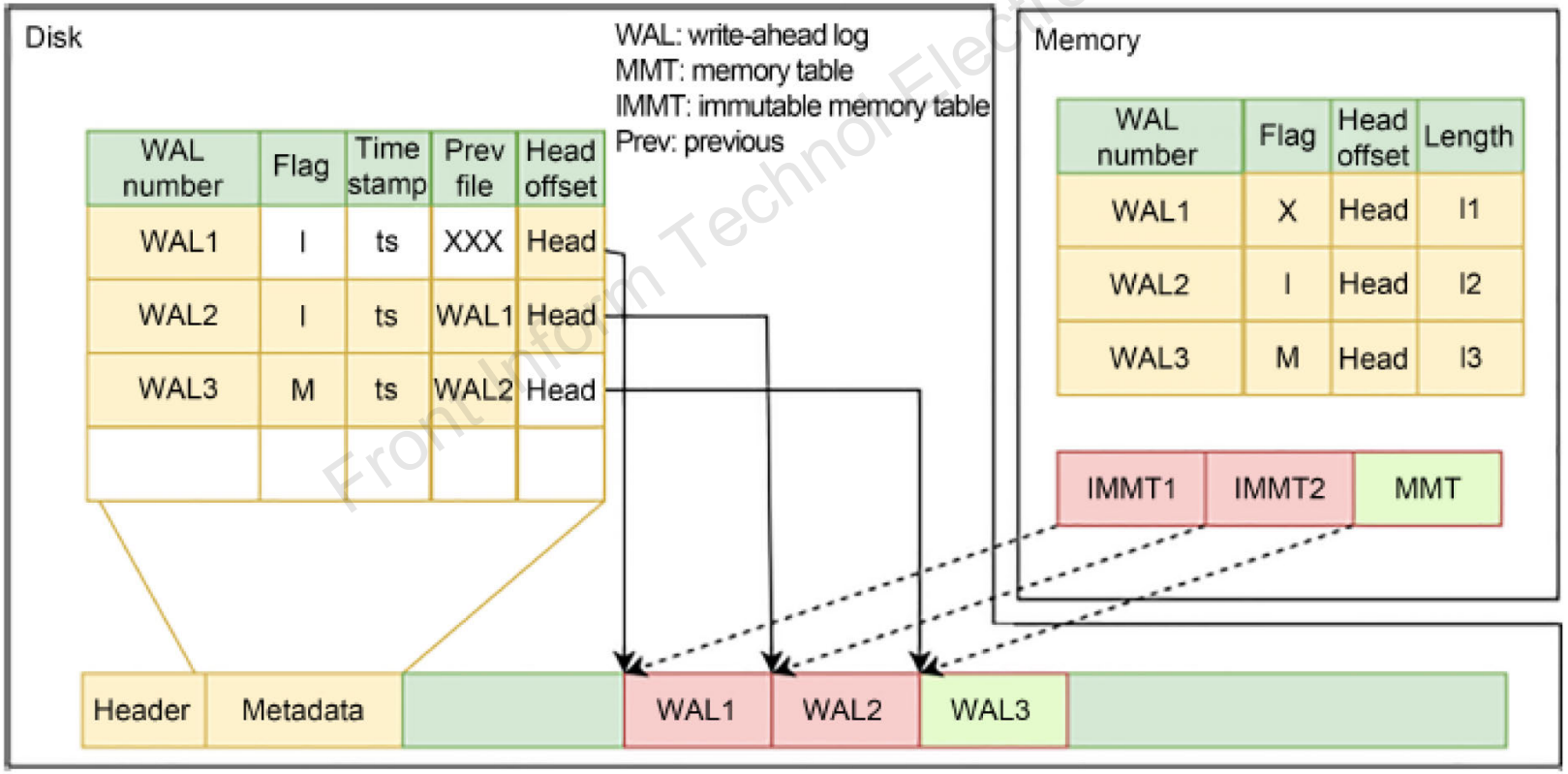


Fig. 6 Disk format and memory tables of MyWAL

Reconstruct of WAL records

- For the newest WAL file, we append records without update length information of the file.
- We add a four-byte flag in each record. We put the system time in the first record in a logical block. For each subsequent record, it saves the cyclic redundancy check (CRC) of the previous record.
- Time stamp in the first block is same with the time stamp saved in the metadata of the WAL file.
- The previous CRC data construct a linked list, which can be used to find the end positon of the file.

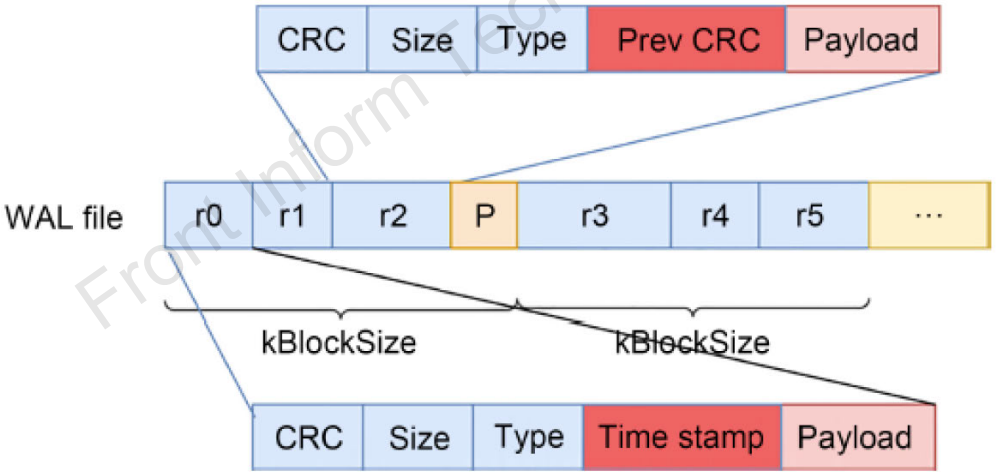


Fig. 7 Data format of a WAL file in MyWAL (WAL: write-ahead log; CRC: cyclic redundancy check)

Recover process

- First MyWAL rebuilds the WAL list and analyzes which WAL files should be rebuilt to the memory table.
- We do not update the file length while writing data for the newest WAL file. A record is the basic writing unit for WAL files.
- Fig. 9 lists all eight cases in the writing process. We use a solid line to represent the completed writing operations and a dashed line to represent uncompleted writing operations.
- By doing the CRC of a record, MyWAL can find out if the record is unabridged or broken.
- For cases 1, 3, and 7, a broken record indicates that the end position is the previous record. In case 5, the data in P is not all 0, indicating that P is the end of the WAL file.
- For uncompleted writing, the data is not written on the disk, the previous CRC is different from previous record.

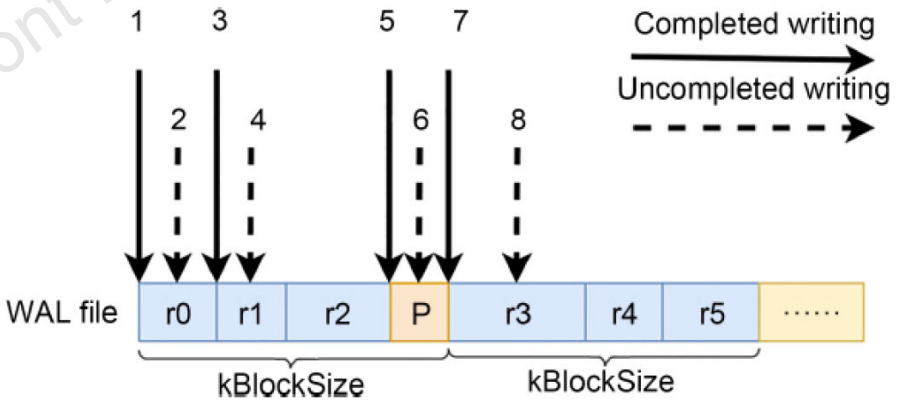


Fig. 9 Rebuilding of the newest WAL file

Evaluation—Environment

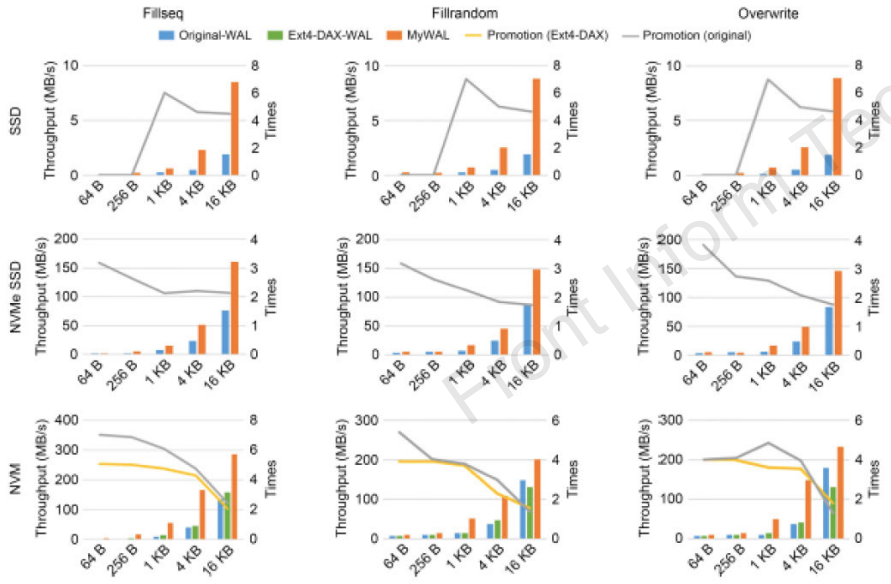
- We compare the performance of MyWAL with the original RocksDB using db_bench on three kinds of media.
- We compare the performance of MyWAL with state-of-the-art KV store SpandDB using YCSB.

Table 1 Experimental configuration information

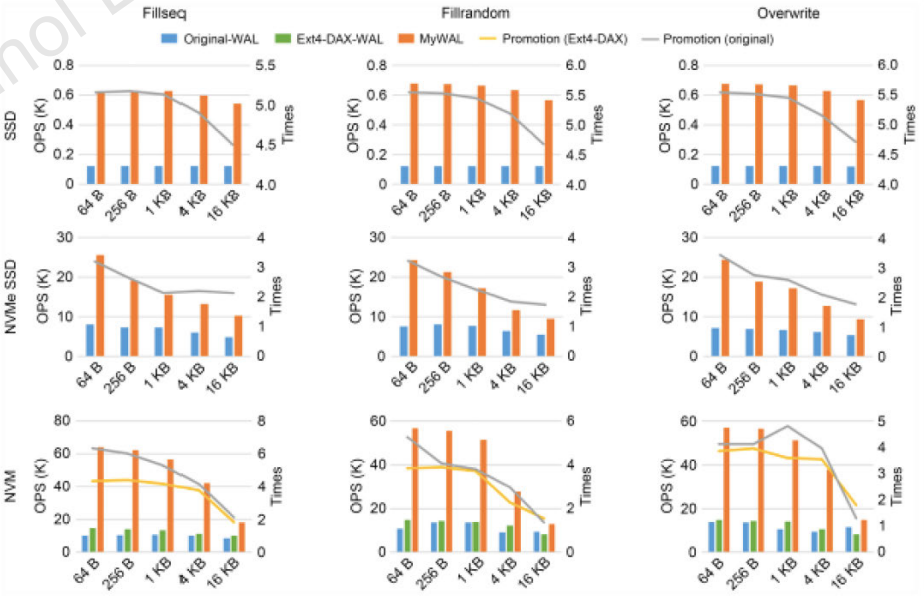
Item	Description
CPU	Intel® Xeon® Gold 5218 CPU @ 2.30 GHz
Memory	Samsung 2666 MHz DDR4 32 GB
SSD	Samsung SSD 860 EVO 250 GB
NVMe SSD	Intel® Optane™ SSD 900P 480 GB
NVM	Two Intel® Optane™ DC persistent memory 128 GB
RocksDB	RocksDB-6.28.2
Benchmark	db_bench of RocksDB-6.28.2
	FIO-3.28
	YCSB-0.17.0

Evaluation—Single thread

- The columns represent the results of three workloads: fillseq, fillrandom, and overwrite. The rows represent different media.
- The throughput of MyWAL is approximately 4 to 7 times larger than the original WAL in all three workloads for SSD.
- The throughput of MyWAL has about 1.72 to 3.8 times improvement on NVMe SSD and 1.29 to 6.13 times on the NVM.



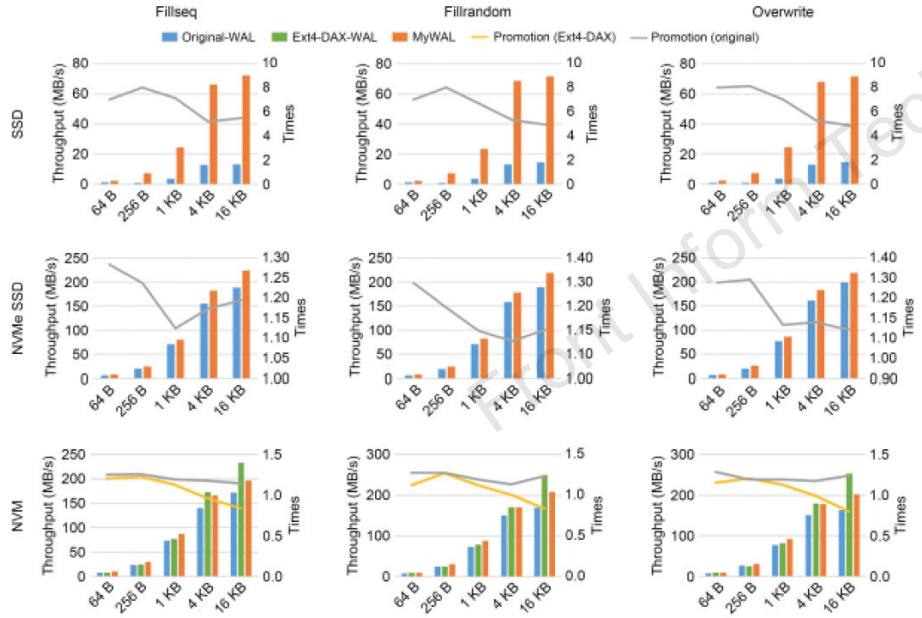
Throughput comparison



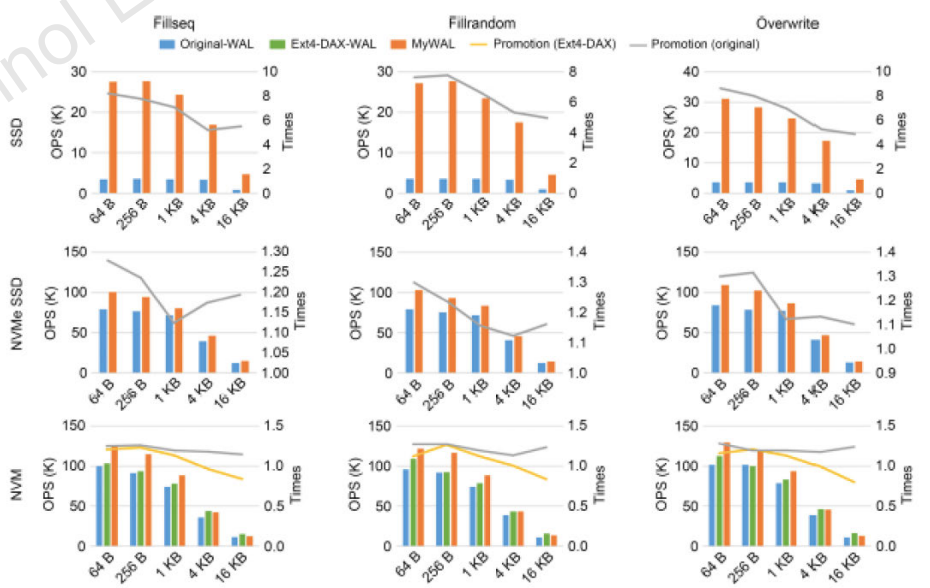
OPS comparison

Evaluation—Multiple thread

- For SSD, the throughput of MyWAL is approximately 5 to 8 times larger than that of the original WAL in all three workloads.
- The throughput of MyWAL has about 12%–30% improvement on NVMe SSD and 13%–29% on the NVM.



Throughput comparison



OPS comparison

Evaluation—Macro benchmark

- We use YCSB (Yahoo! Cloud Serving Benchmark) compare the performance of MyWAL, original RocksDB, and SpanDB.
- The writing latency of MyWAL is better than that of SpanDB. The average latency decreases by 50% and 70% compared to those of SpanDB and RocksDB in 100% write case. The P99 latency is also improved by 47% and 64%.

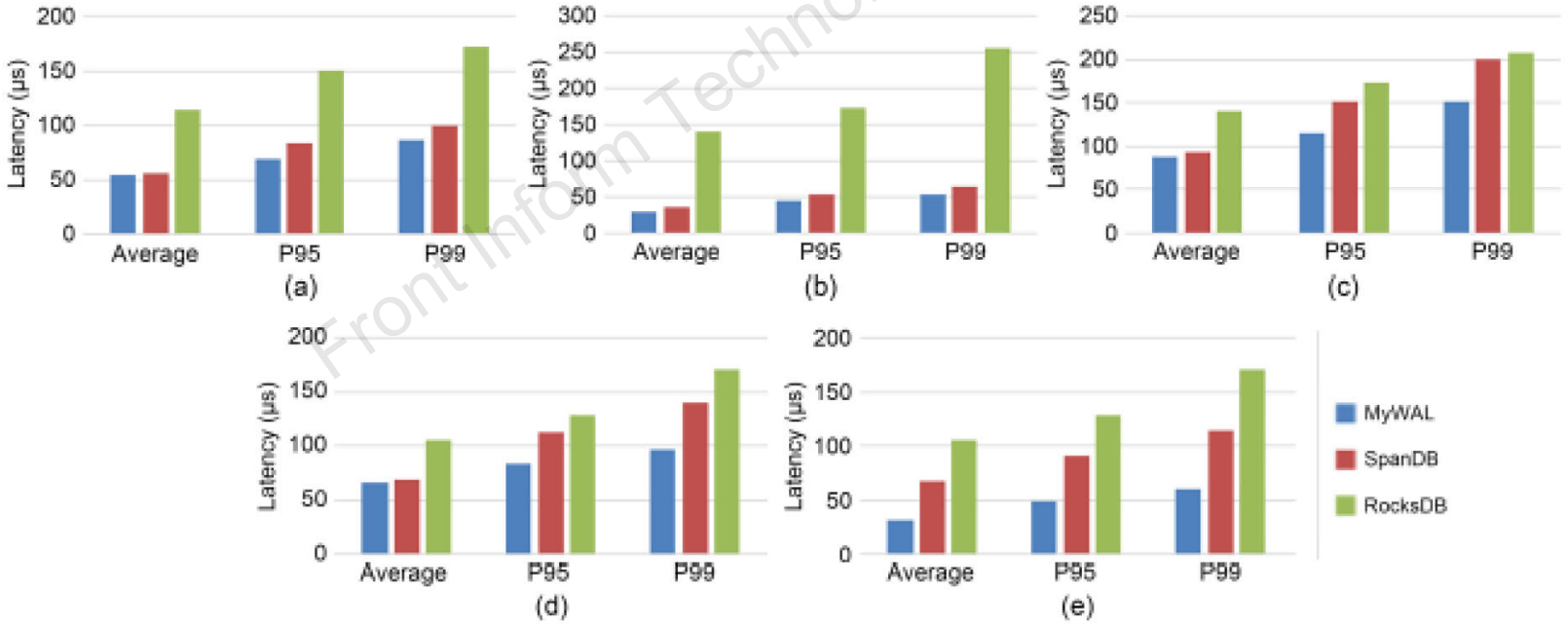


Fig. 14 Comparison of latency of MyWAL, SpanDB, and RocksDB: (a) YCSB-A; (b) YCSB-B; (c) YCSB-E; (d) YCSB-F; (e) 100% write

Conclusions

- We present a new WAL mechanism named MyWAL . It reconstructs the format of WAL files and manages space directly on the raw devices.
- By removing the useless metadata and unnecessary update operations, MyWAL reduces the latency of WAL writing.
- MyWAL has excellent performance on SSD, and the results show that it is eight times faster than the original RocksDB. This optimization can be applied on different block devices, such as NVMe SSD and NVM.

Xiao ZHANG is an associate professor in the School of Computer Science at Northwestern Polytechnical University. He received his doctoral degree in 2006. His research interests include storage systems, computer networks, and distributed file systems.

Mengyu LI graduated from Northwestern Polytechnical University in 2023 with a MS degree in electronic information. Her main research direction is the performance improvement of distributed storage systems and backend storage engines.