Paper review rocks db

Since there is a tradeoff between space, write and read amplication, Facebook decided to focus on efficient usage of space and write using LSM tree. They claim that this makes them unique as other database focuses more on the performance instead of the hardware.
RocksDB is a persistent key-value forked from Googles LevelDB.
They have a system called MyRocks which uses RocksDB for MySQL storage engine instead of the default InnoDB. This move is inspired by the fact that Facebook uses MySQL and has 10s of petabytes and using MyRocks uses half the storage as compared to InnoDB with better throughput but affecting read latency.
For the Facebook data-load, InnoDB is not able to fully utilize the SSD, since the query per node is low. Using RocksDB helps in terms of space, the number of nodes are reduce(less sharding) hence increasing performance.
The space efficiency is achieved due to the property of the LSM tree and different compression techniques. Since the typical implementation of InnoDB uses the B+ tree and the Facebook data load seems to only utilize around 1/2 to 2/3 of the tree, a lot of space gets wasted. RocksDB uses a dynamic implementation of the LSM tree that contains a tunable level parameter with tradeoff between (space, read) and write amplification. A constant parameter at each level would be great for optimizing write amplification but the paper poses an open question whether the same would work for the space amplification as well.

It also exploits properties of LSM tree for compression. They use different compression techniques at each level as discussed in the tradeoff section. They also use bloom filter for membership identification except at the last level. They also made a variation of bloom filter called prefix bloom filter for range queries.

Due to the usage of Dynamic LSM tree, variety of tiered-compression techniques , using bloom and prefix bloom filter, tunable level parameter and shared dictionary compression and the nature of the data load inside Facebook, RocksDB is able to reduce the space amplication by 50% as compared to InnoDB and work well on OLAP work load and also competitive for OLTP workloads.

Since rocksdb claims to reduce space by 50% over INNODB,
How much is affected by the choice of using a lsm tree vs just using the same compression techniques on a B+ tree.
It mentions that the read latency is increased only marginally but provides no statistical data to justify what it considers “marginal”.
Since the paper mentions that having same level parameter works well for optimizing write amplification and space is still in research, I am curious to find out if a similar research is done for optimizing the read amplification using tunable parameter of the dynamic LSM tree.
The paper majorly focuses on the Facebook data load. I would have liked some

benchmarks on some of the open source data loads that can be recreated. They do show LinkBench(again made by Facebook mimicking their data load) and the standard TPC-C benchmarks but would have appreciated some other benchmarks as well.