CAS CS 561: Data Systems Architectures
Data-intensive Systems and Computing Lab
Department of Computer Science
College of Arts and Sciences, Boston University
http://bu-disc.github.io/CS561/

CS561 Spring 2022 - Research Project

Title: Range Deletes in LSM-Trees

Background: The log-structured merge (LSM) tree is a disk-based data structure that is highly write-optimized, and thus, widely used in modern NoSQL-based key-value stores [1,2]. However, deletes in LSM-trees are realized *logically*, i.e., via logical invalidation of target entries without physically deleting them from the database. Logical deletes affect the performance of LSM-trees adversely [3]. Particularly, deleting entries under a given range of keys is challenging, and it come at a cost of high read amplification and increased CPU utilization [4]. A state-of-the-art implementation of range deletes in LSM-engines requires an in-memory timestamp-driven map for the keys logically invalidated at different time instants. Maintaining and updating this range deletion map incurs CPU overheads. More importantly, in presence of range deletes, the read performance of LSM-trees suffers significantly. This is because the result returned by a point or range query must be verified against the range deletion map to ensure consistency. This slows down queries and may increase the I/O cost for reads.

Objective: In this work we will take RocksDB as the LSM-engine of our interest and study its implementation of range deletion.

(a) Step 1: Understand the implementation of range deletes in RocksDB.
(b) Step 2: Measure the "cost of ensuring consistency" in presence of range deletes in terms of point and range read throughput.
(c) Step 3: Measure the I/Os spent to read data that is logically invalidated by a range deletes.
(d) Step 4: Measure the memory footprint of the range delete map and the CPU cycles spent to ensure data consistency,

Responsible mentor*: Subhadeep Sarkar*

References

[1] O'Neil et al., "The Log-Structured Merge-Tree (LSM-Tree)", Acta Informatica, 1996.
[2] Luo and Carey, "LSM-based Storage Techniques: A Survey", The VLDB Journal, 2020.
[3] Sarkar et al., "Lethe: A Tunable Delete-Aware LSM Engine", SIGMOD, 2020.
[4] Madan and Kryczka, "DeleteRange: A New Native RocksDB Operation", https://rocksdb.org/blog/2018/11/21/delete-range.html, 2018.