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 2023 - Research Project

Title: *Exploring the Optimal Compaction Strategy for A Given Workload*

Background: Choosing the file with the minimum overlapping ratio (MinOverlappingRatio) to compact is the default compaction priority in RocksDB, since it effectively and robustly reduces write amplification (WA). In fact, there are five different compaction algorithms in RocksDB and some of them may have a smaller WA than MinOverlappingRatio for certain workloads.

Problem: Imagine that there exists a global objective function to accumulate the total WA, we are pursuing the best file that should be chosen to minimize the global cost. In fact, for each compaction, MinOverlappingRatio behaves as a local minimizer, that tries to minimize the WA, triggered by the current compaction. Since it is hard to formalize the global WA cost, it is even harder to investigate whether the objective function is convex or not. That is to say, MinOverlappingRatio does not necessarily lead us to the global minimum WA. In fact, as we have seen that other strategies under certain scenarios have lower WA, we can ensure that the MinOverlappingRatio cannot always have the optimal solution.

Objective: The objective of the project is to find the best compaction strategy to compact every time a level reaches its capacity, for a given small workload (with fixing #ingest, #updates, #deletes, etc).

(a) Read the code under "compaction/" directory in RocksDB and understand when compaction picking logic is being executed.
(b) Expose the picking logic to application level so that we can customize which files should be selected to compact.
(c) Enumerate all possible file selections and find the best files (the options that lead to global minimum WA) are picked every time a compaction is triggered for a small workload (at least 3-level). Note that enumeration has exponential complexity, so it is infeasible to find the best answer for large workloads. In addition, in case there are pending compactions and flushes, you have to ensure the LSM tree is in a stable state before you measure the final WA. (You can consult the mentor for more details about how to confirm a stable state).
(d) Run the same workload with MinOverlappingRatio and RoundRobin, and compare the WA from these two policies against the minimum value, generated by our brute-force algorithm.
(e) Vary the workload composition (#updates, #deletes, distribution) using our workload generator and repeat the above comparison.

Responsible Mentor: *Zichen Zhu (zczhu@bu.edu)*
RocksDB Repo: https://github.com/facebook/rocksdb