COSI 167A - Fall 2024 - Research Project

Title: *Time-Traveling LSM-Engines*

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, updates in LSM-trees are realized *logically*, i.e., via logical invalidation of target entries without physically deleting the older version of the entries from the database. Lookups on updated keys never return older versions of a key, which are eventually physically deleted from the database at a later time instant during compaction. LSM-trees are, thus, unable to store multiple versions of a key by design. This implies state-of-the-art LSM-engines are unable to support queries such as: "What was the value of key X at a past time instant t1?" or "Which keys are updated between timestamps t1 and t2?" [3].

Objective: In this work, design an LSM-tree (or a family of LSM-trees that can support point and range queries along the time axis.

(a) Step 1: Formalize the different types of time-traveling queries.
(b) Step 2: Propose new designs based on the LSM-paradigm that can distinguish among multiple versions of a given key.
(c) Step 3: Compare the performance of the different proposed designs to determine if there is a dominant design.
(d) Step 4: Run experiments with a wide array of workloads and report the results.

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] Muth et al., "The LHAM log-structured history data access method", The VLDB Journal, 2000.