LSM-Tree Structured Storage

by Yinxuan Feng and Xiaotong Niu

Interfaces

Put/update

Get

Scan (not implemented yet)

Delete

Important techniques

- Level structure
- B+ tree indexing
  - Split implementation
  - Persistence policy (save and restore)
- Merge
- Write to component from memory

Level structure

- Constant number of levels
- Meta information file per level: all the components
- Each component has
  - A .bpt file for indexing
  - A .data file for actual data

B+ Tree

- Self-balanced tree
- Split and recursive insert
- Serialization and deserialization (recursive)
- One key per tree
    - Update delete only changes the value
- Only leaves have data (indexes)

Merge

- for all the components of the level
  - While there is something to read
    - Get the first kv pair
    - Append the latest pair with the smallest key to temp file
    - Also set up B+ tree
- If leveling and too large:
  - copy temp file to the next level as component
- If tiering:
  - copy temp file to the next level as component

Write to component from memory

- Only write to level 1 as component
- Sort kv map inside memory
  - (no more than 1 value per key)
  - Insert in order to component
  - Set up B+ tree

Benchmark

- Large (enough) number of operations on disk
- Check multiple files on disk every time when read?
- Write data to Log
  - Query and write latency
- Generate data larger than memory size