Multi-Temperature LSM Tree-Based Database Storage

Hobin Yoon  
hobinyoon@gatech.edu  
Georgia Institute of Technology

Ymir Vigfusson  
ymir@mathcs.emory.edu  
Emory University & Reykjavik University

Ada Gavrilovska  
ada@cc.gatech.edu  
Georgia Institute of Technology

LSM (Log-Structured Merge) tree is a data structure that transforms random writes to sequential writes, thus used by many write-favoring databases, including:

- Cloud Bigtable
- cassandra
- LEVELDB
- RocksDB
- MongoDB

Problem: When you serve a LSM tree-based database using homogeneous storage devices, you get either high performance or low cost, but not both.

Storage cost and read latency of Cassandra using various storage devices with YCSB “read latest” workload.

- EBS gp2
- EBS st1
- EBS sc1
- Local SSD

We propose Mutant, a multi-temperature LSM tree-based database storage system. Mutant stores more frequently-accessed data to fast, expensive storage devices, and less frequently-accessed data to slow, less-expensive storage devices, thus achieving better cost-performance trade-offs.

Why does it work? Many data access patterns exhibit localities. In a user-generated content system, objects lose their popularity quickly as they age.

Access patterns of Facebook photos (left) [Kumar '15] and YouTube videos (right) [Brodersen '12]

The record access frequency disparity translates to the access frequency disparity of SSTables.

SSTable temperature progression with the YCSB “read latest” workload. Temperature = (number of accesses) / sec / byte

The “warm” or “cold” data increases as time goes by.

Mutant monitors SSTable temperature, classifies them into different temperature levels, and stores them in different storage device types.

Preliminary result: Up to 78.98% cost reduction without any sacrifice in read latency!

Related work

Hierarchical storage systems
- AutoRAID (block storage)
- Facebook Haystack and f4 (BLOB storage)
- Siberia and Anti-caching (In-memory database)

Caching systems: tons of them

They complement each other. Caching systems absorb “hot” data requests (head part of the Zipfian curve), while hierarchical storage systems focus on cost-efficient storage of “warm” or “cold” data (tail part).