
# Redesigned Compaction LevelDB

Lighting Talk

@bcoverston

Datastax

DataStax

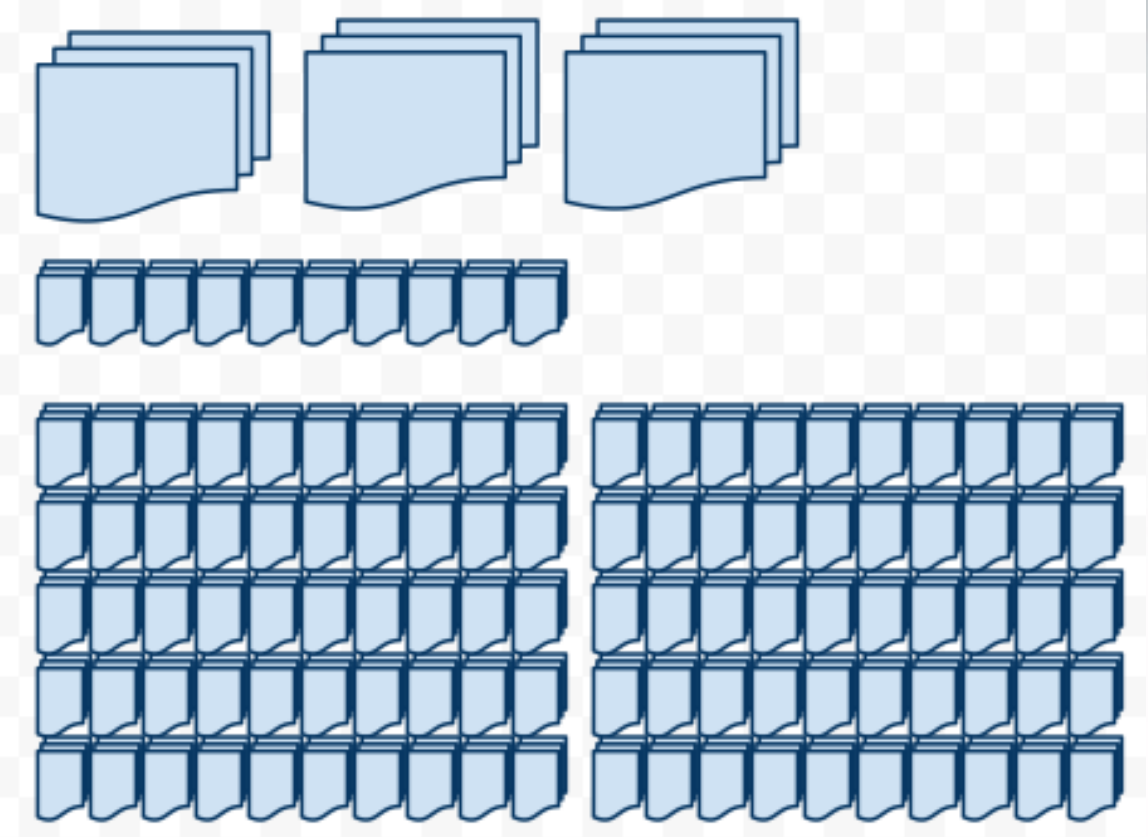
# Motivation

- Compaction is “bursty”
- Worst case scenario for compaction requires 50% excess capacity
- Worst case row merging scenario could require seek on each SSTable
- CASSANDRA-1608

# LevelDB

- Inspiration from the Chromium Project

# LevelDB Structure



# Implementation Challenges

- L0 accumulation
- Many SSTables
  - Range checking on 1500-10k SSTables is still slow
  - BF check for tombstone eviction
- Remedies:
  - Add interval tree to make range and BF checks O (log(n)

# Pros and Cons

- Cons
  - Increased total IO?
- Pros
  - Increased IO represents higher level of de-duplication for repeated writes, row merges are “amortized” during compaction.
  - No more 50% excess capacity requirement
  - Compaction applied to the structure until it is ‘done’

# Questions?