SILT: A MEMORY-EFFICIENT, HIGH-PERFORMANCE KEY-VALUE STORE

Hyeontaek Lim, Bin Fan, David G. Andersen (CMU), Michael Kaminsky (Intel Labs)

INDEXING DATA IN HIGH-PERFORMANCE KEY-VALUE STORES

In-memory index: **Index structure entirely in DRAM**

- **Common design in high-performance key-value stores**
- **E.g., BufferHash, Haystack, FlashStore, FAWN-DS, ...**

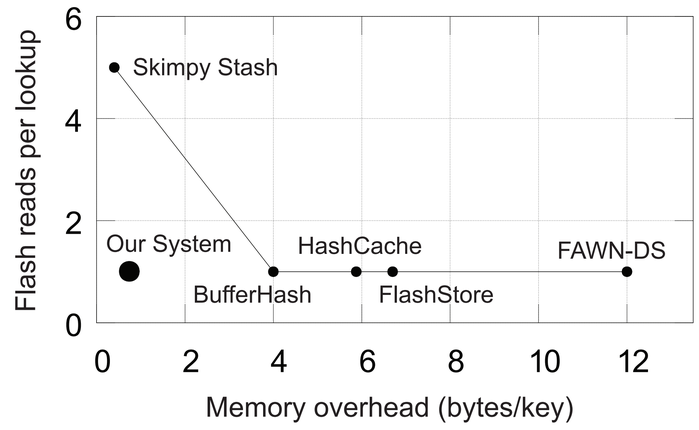
Main challenge: **Memory overheads of in-memory indexes**

- **In-memory indexes are taking much memory**
  - **4 billions of 32-byte items (128 GB) may require 16 GB DRAM**
- **Per-entry DRAM space is being more scarce**
  - **Flash capacity/$ is growing faster than DRAM capacity/$**
- **High performance is still important in key-value stores**

→Memory-efficient, high-performance key-value store?

Our key-value store system requires only

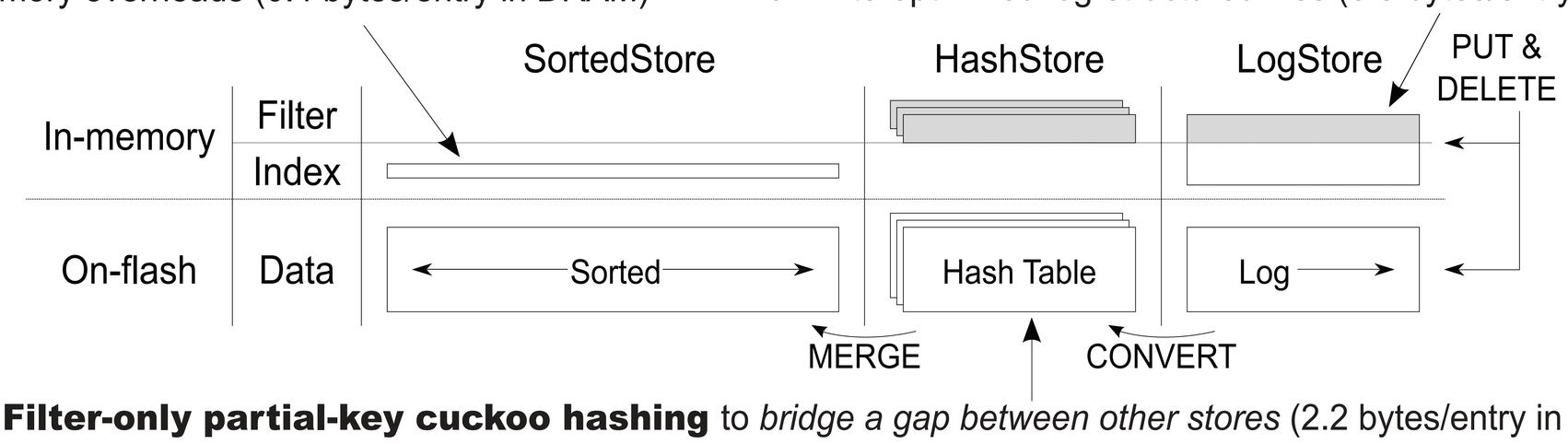
0.7 bytes/entry in DRAM & **1.01 flash reads/lookup**



DESIGN: THREE DIFFERENT BASIC STORES WITH NEW INDEXING DATA STRUCTURES

Entropy-coded trie index
for low memory overheads (0.4 bytes/entry in DRAM)

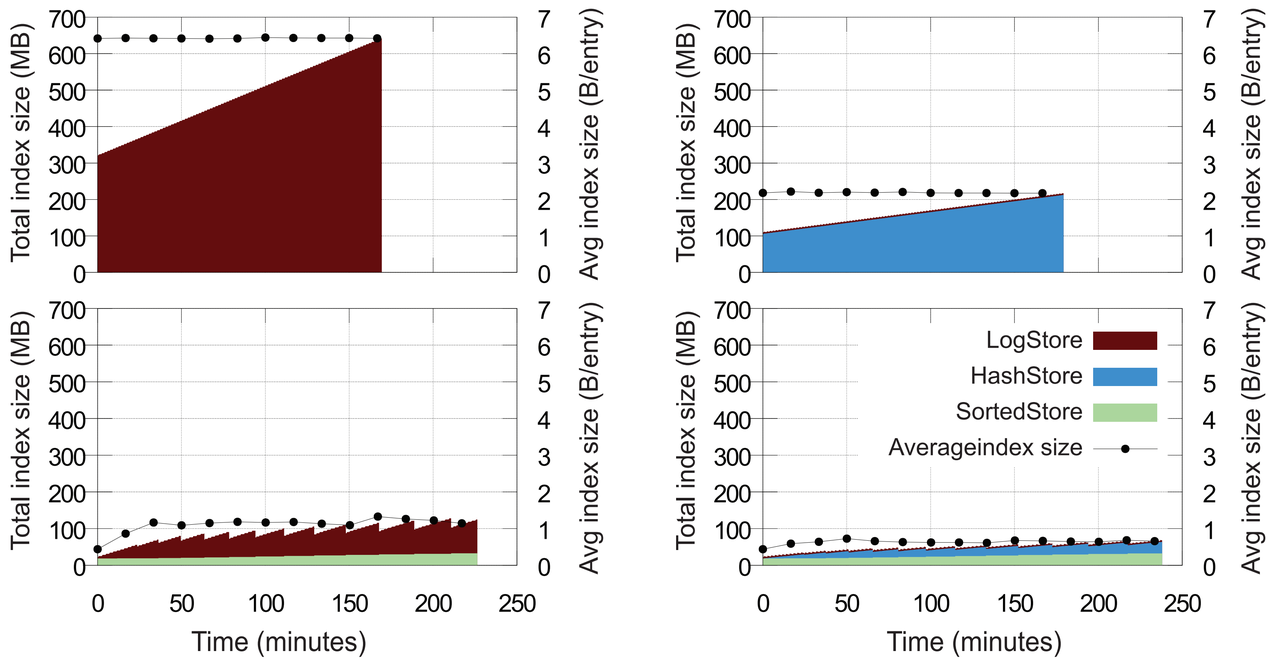
Partial-key cuckoo hash index
for write-optimized log-structured files (6.5 bytes/entry in DRAM)



Filter-only partial-key cuckoo hashing to *bridge a gap between other stores* (2.2 bytes/entry in DRAM)

INDEX SIZE OF STORE COMBINATIONS

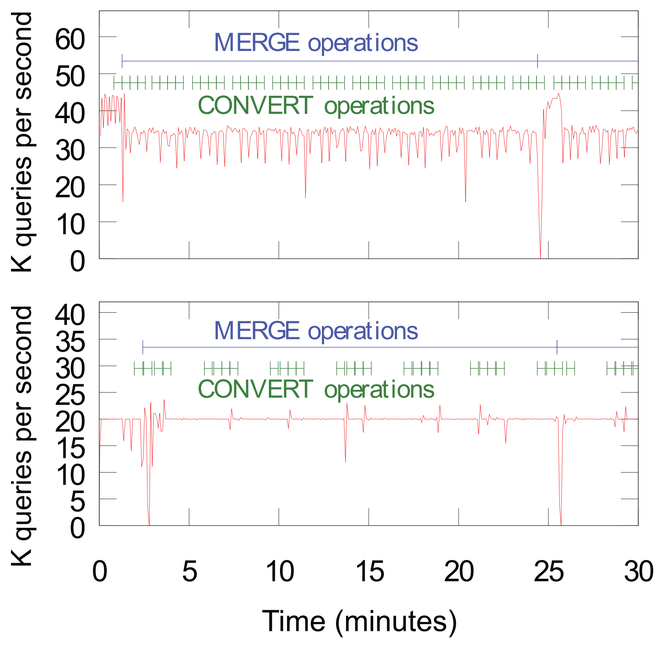
Inserting 50 million new key-value pairs



→ Combining **three** stores yields the **best memory efficiency**

QUERY THROUGHPUT

Under high (upper) and low (lower) loads



→ **High query performance** is maintained in both cases

Intel Science & Technology Center for Cloud Computing
Carnegie Mellon University
Georgia Tech
intel
PRINCETON UNIVERSITY
UC Berkeley