ElasticBF: Elastic Bloom Filter with Hotness Awareness for Boosting Read Performance in Large Key-Value Stores

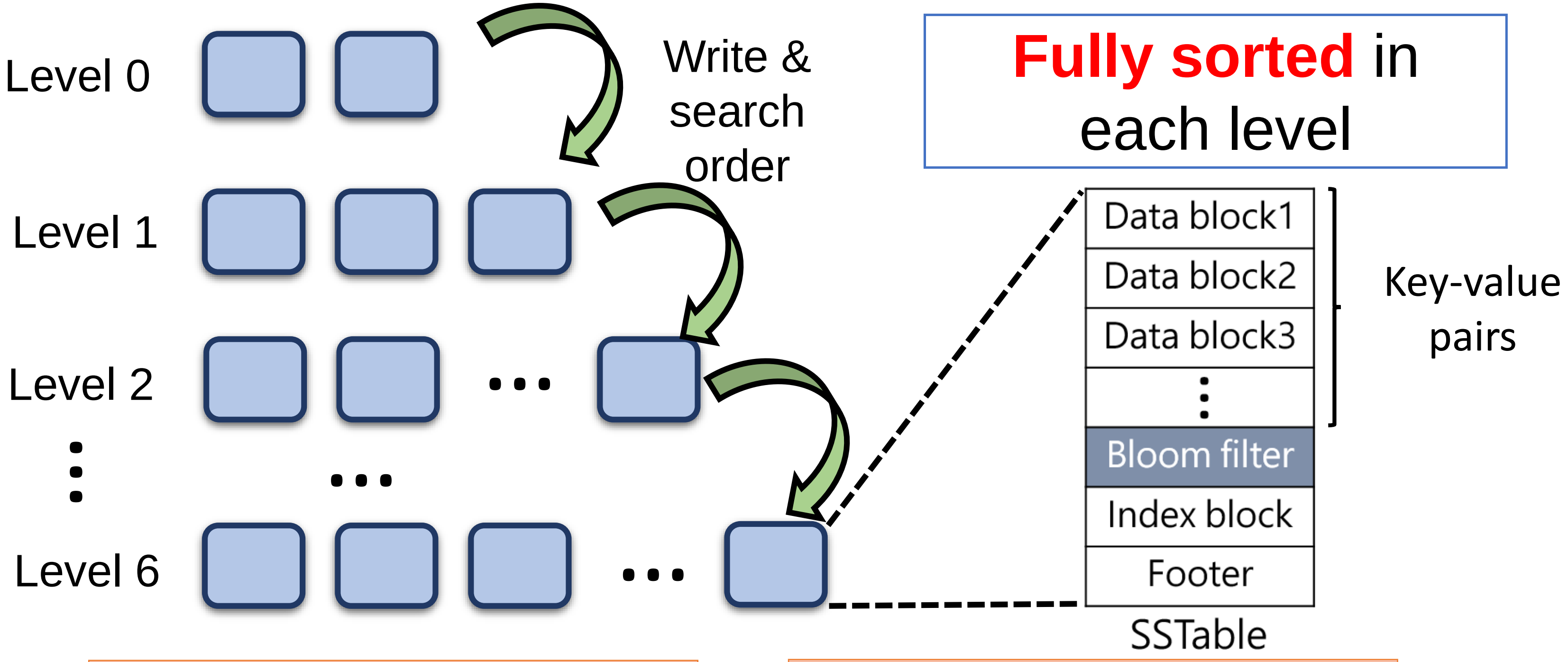
Yongkun Li, Chengjin Tian, Fan Guo, Cheng Li, Yinlong Xu

University of Science and Technology of China

LSM-tree-based KV Stores

- The most common design of KV stores is based on LSM-tree (log structured merge tree)

Fully sorted in each level



Read Amplification
(searching multiple SSTables is needed)

Bloom Filters
(also cached in mem. to improve read)

Motivation

- Bloom filters suffer from false positive rate
  - ❑ False positive rate (FPR): 0.6185^b (b: Bits-per-key)

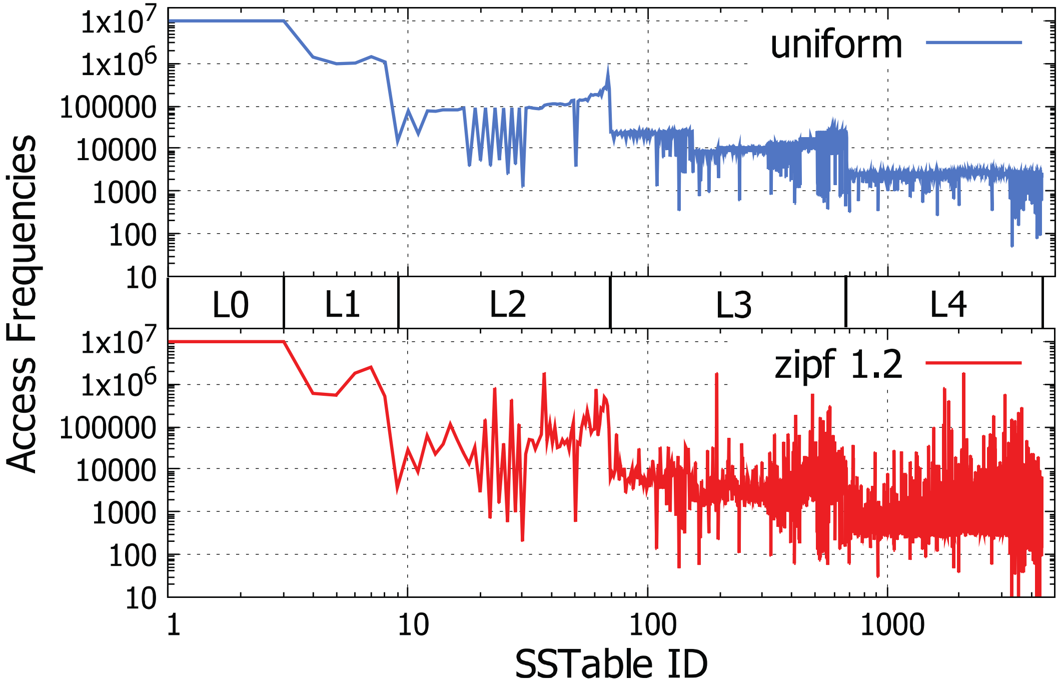
Bits-per-key	2bits	3bits	4bits	5bits	6bits
FPR	40%	23.7%	14.7%	9.2%	5.6%

  - ❑ How to reduce false positive rate?
    - ❖ Allocate more bits for each key
    - ❖ Incur large memory overhead

Question: how to improve the Bloom filter design with limited memory consumption?

Main Idea

- ElasticBF: locality-aware elastic scheme
  - **Observation**: unevenness of access frequencies (between levels/SSTables)



Hot SSTables
More bits/key
Lower FPR

Cold SSTables
Fewer bits/key
Limited mem. usage

Challenges & Design Highlights

- BF allocation: Immutable data organization and heterogeneous accesses even within an SSTable

BF separability + Fine-grained allocation

- Writes in mixed workloads lead to reset the hotness info. (as compaction creates new files)

Hotness inheritance

- BF adjustment requires smart decision with small memory overhead to realize the elastic feature

Cost-benefit analysis + in-memory multi-queue DS

For more detailed design and evaluations, welcome to our talk!!

ATC2019, 4:35 pm–5:55 pm, Track II, on July 11th