Hashing and Indexing

Saptarshi Pyne
Assistant Professor
Department of Computer Science and Engineering
Indian Institute of Technology Jodhpur, Rajasthan, India 342030

CSL4030 Data Engineering Lectures 25, 26
October 9th, 11th, 2023

What we discussed in the last class

(Network) directory access protocols

- Lightweight directory access protocol (LDAP)
  - LDAP data interchange format (LDIF)

Hashing

- Hash map aka. hash table
  - (key, byte offset)
  - Hash collision resolution strategies (e.g., linked lists)

- Databases where number of keys fit into the main memory but per key updates are too frequent
  - Log file and log segment file
  - Compaction and Merger of log segment files
  - In-memory hash map of each log segment file

Hashing (contd.)

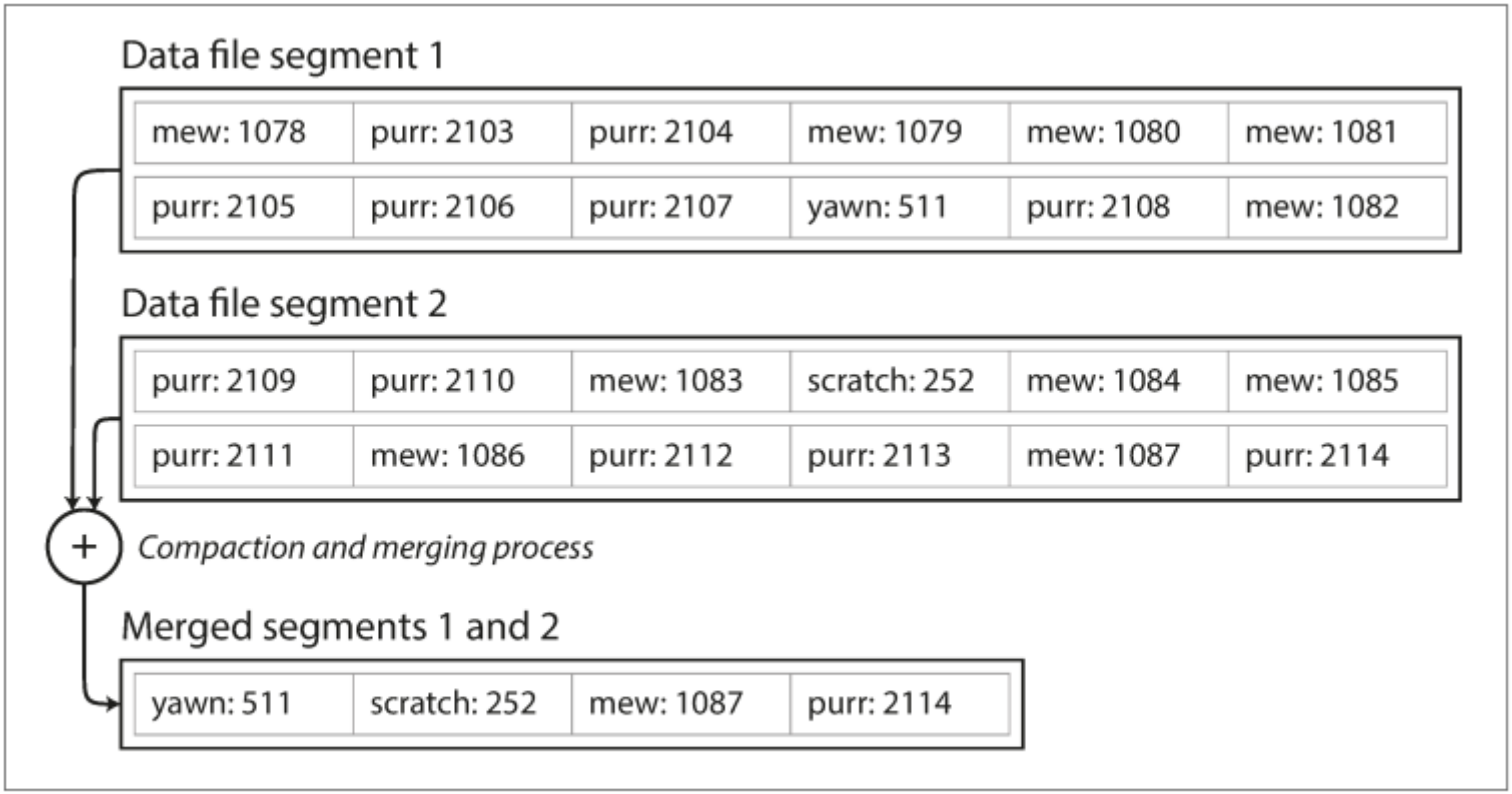


Figure 3-3. Performing compaction and segment merging simultaneously.

Hashing (contd.)

- Searching for a key
- Deleting a key and logging its tombstone
- Crash recovery and database restarts
  - Rebuilding vs. snapshotting

Limitations of hashing

- A database where the number of keys **does not** fit into the main memory
- Does not allow range queries (e.g., ‘B21AI001’… ‘B21AI058’)
- Resolutions
  - B-tree
  - SSTable and LSM-tree

B-tree (Bayer and McCreight, Boeing Research Labs, 1970)

- The most widely used database indexing strategy

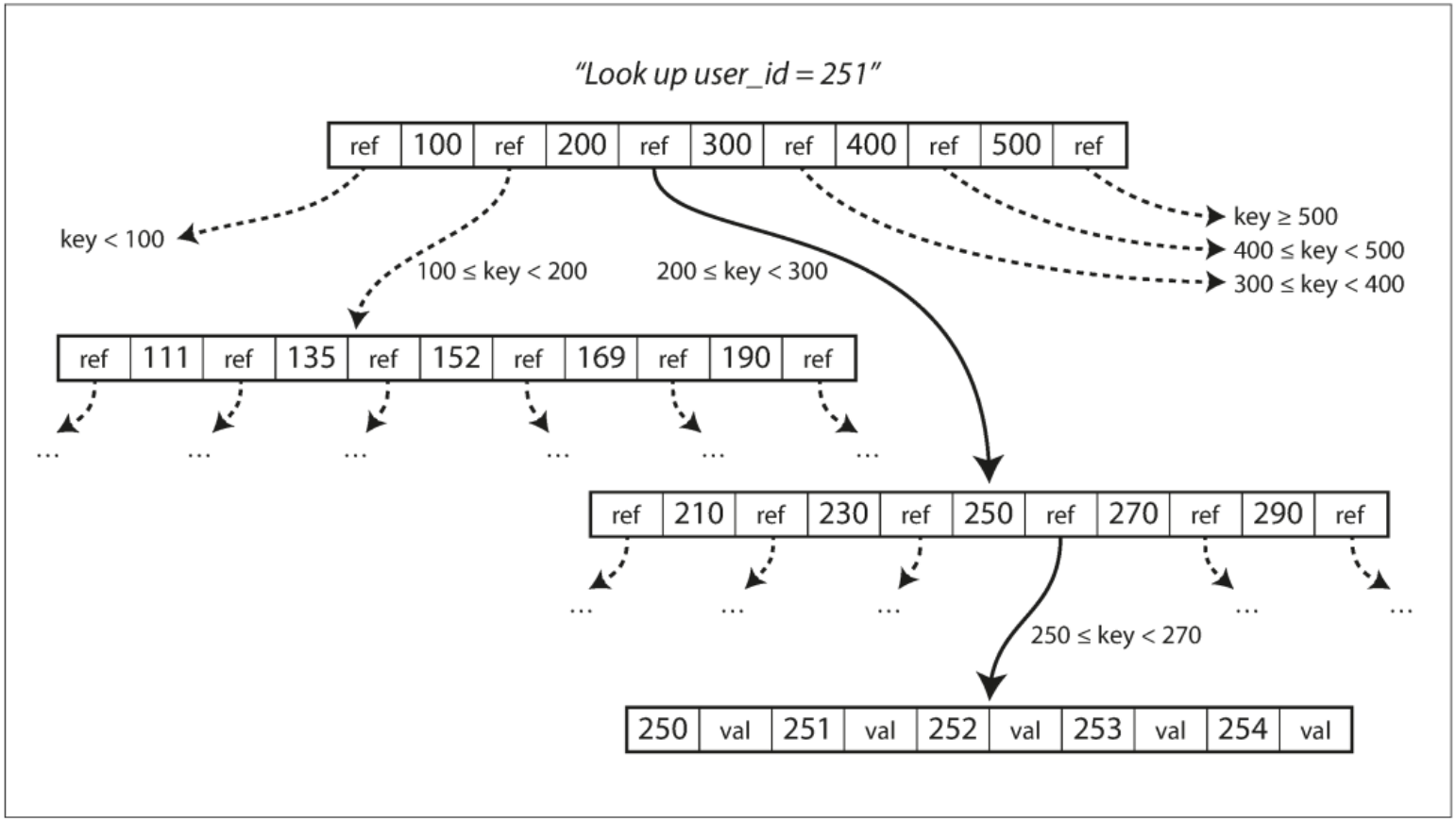


Figure 3-6, Kleppmann

B-tree (contd.)

- Branching factor: The number of references to child pages

Shorted string table (SSTable)

- Introduced in the **Google BigTable** paper: Fay Chang, **Jeffrey Dean (Chief Scientist, Google and Lead, Google AI)**, Sanjay Ghemawat, et al., 'Bigtable: A Distributed Storage System for Structured Data', 7th USENIX Symposium on Operating System, Design and Implementation (OSDI), November **2006**.

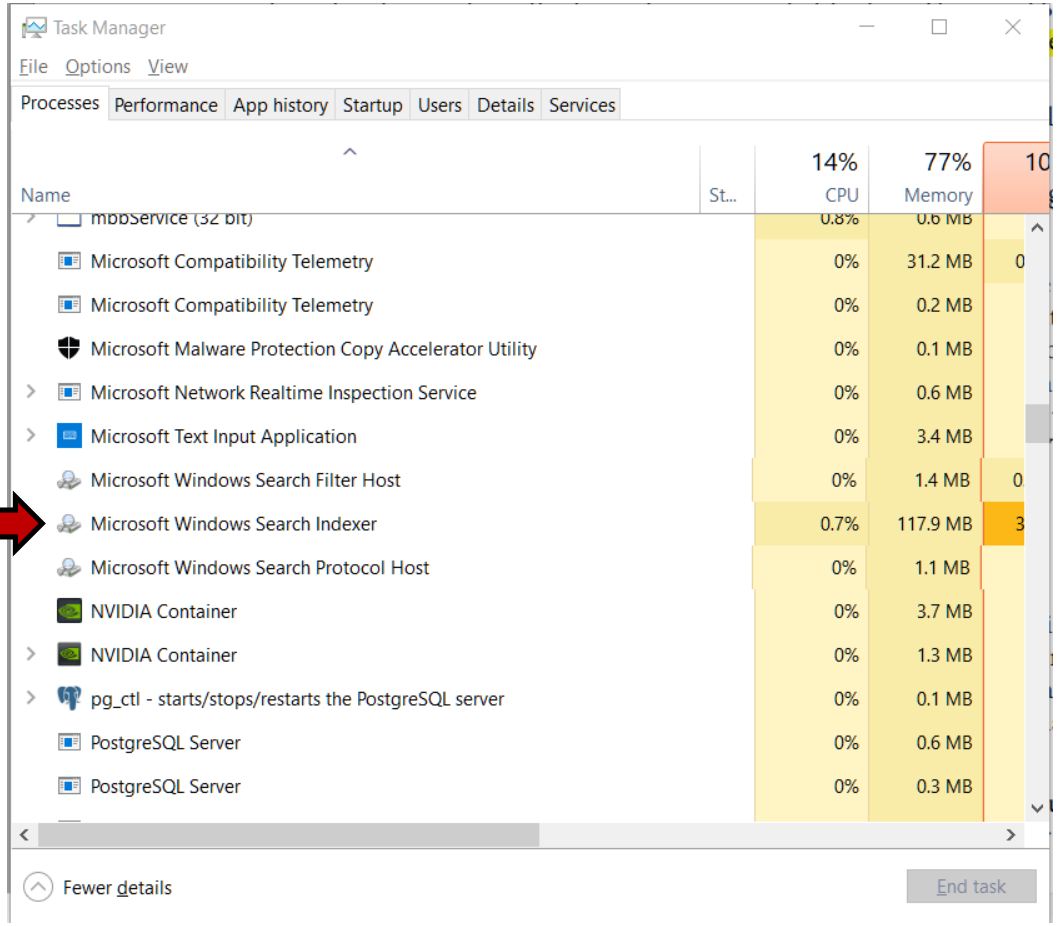
- Assumptions:
  - A key can appear at most once in a log segment file (ensured through compaction)
  - When merging multiple log segment files, only adjacent files are merged

SSTable: Operations

- Sorting by key using **in-memory balanced tree data structures** that allow insertion of keys in any order and reading them back only in the sorted order, e.g., red-black tree, AVL tree.
  - The in-memory sorted key tree is sometimes called a **memtable**.
  - When a memtable exceeds a predecided threshold (usually a few MBs), move it to the disk as an SSTable. Meanwhile, the insertion of new keys happens in a new memtable.

SSTable: Operations (contd.)

- Periodically merging (and compacting) multiple log segment files i.e. SSTables in the background



SSTable: Operations (contd.)

- Searching through in-memory sparse indices

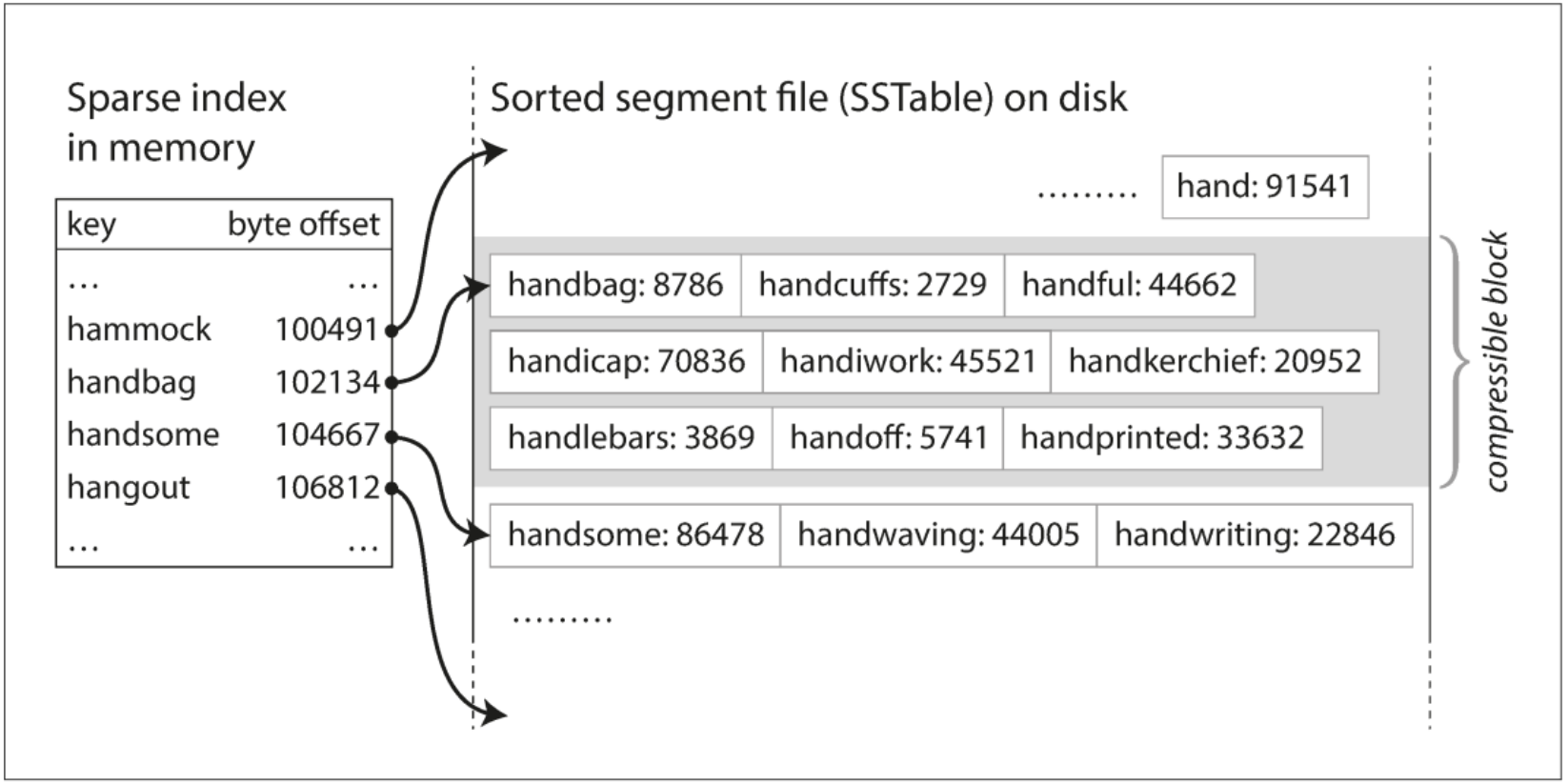


Figure 3-5. An SSTable with an in-memory index.

Limitations of SSTables

- If a memtable crashes (e.g., a database crash or system reboot), the index tree is lost.
  - Resolution
    - Maintain a separate log file in the disk for the current memtable.
    - As soon as a new key is inserted into the memtable, append it to the log. There is no need to sort this log since its only purpose is to restore the memtable in case of a crash.
    - When the current memtable is written to the disk, empty the log. Start anew for the new memtable.
    - **Concern:** Too frequent disk I/Os.

Log-structured merge-tree (LSM-tree)

- Patrick O'Neil, Edward Cheng, Dieter Gawlick, and Elizabeth O'Neil: 'The LogStructured Merge-Tree (LSM-Tree)', ***Acta Informatica***, volume 33, number 4, pages 351–385, June **1996**. doi:10.1007/s002360050048
  - A collaboration between University of Massachusetts-Boston + Digital Equipment Corporation + Oracle corp.
  - Designed for full-text search. Key = a word. Value = The list of IDs of all the documents where that word appears.

What if the searched key does not exist?

- Search all the log segment files only to realize the key does not exist. Wastage of time.
  - Resolution: Add a **Bloom filter**, a memory-efficient data structure that can quickly tell whether a given element is a member of a given set or not.
    - First, run the searched key through a Bloom filter. If the key passes through the Bloom filter, then only search for it in the log segment files.

Burton H. Bloom: "Space/Time Trade-offs in Hash Coding with Allowable Errors," *Communications of the ACM*, volume 13, number 7, pages 422–426, July 1970. doi:10.1145/362686.362692

References

- M. KLEPPMANN (2017), Designing Data-Intensive Applications The Big Ideas Behind Reliable, Scalable, and Maintainable Systems, O'Reilly.
  - Pages 69-90, Chapter 3: Storage and Retrieval

Thank you