Part 6. Trees (3)

CS 200 Algorithms and Data Structures

1

Outline

- **B-Trees**
- External Methods

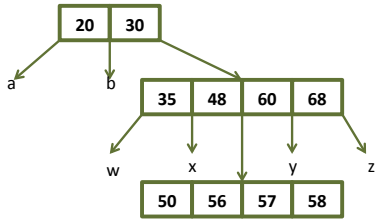
2

B-tree of degree m

- All leaves are at the same level
- Each node contains between m-1 and *floor((*m-2)/2) records (except for the root).
- Each internal node has one more child than it has records.
  - An exception to this rule is the root node
    - As few as one record
    - As few as two children
- A 2-3 tree is a B-tree of degree 3.

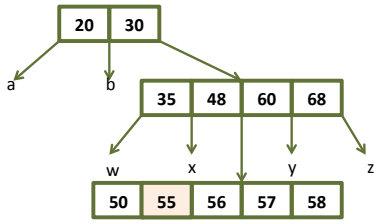
3

Example: A B-tree of degree 5



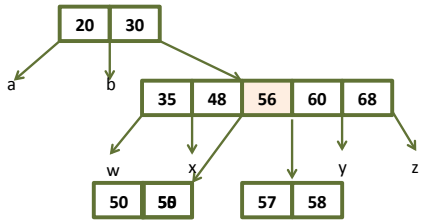
4

Inserting 55



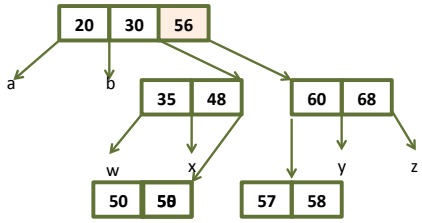
5

Inserting 55



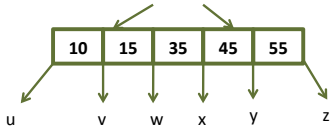
6

Inserting 55



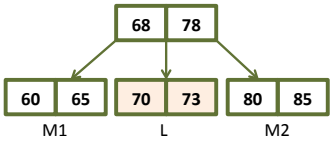
7

Splitting the Root node



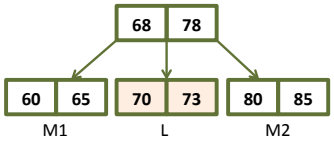
8

Deleting 73



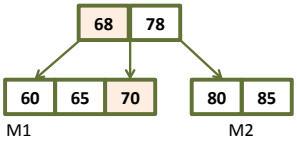
9

Deleting 73



10

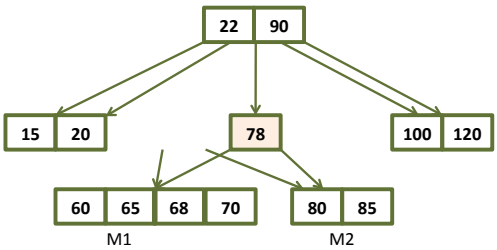
Deleting 73



No sibling has (depth -1) items
→ Merge leaves

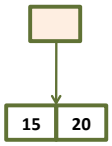
11

Recursive Merging



12

Remove empty root



13

Outline

- B-Trees
- **External Methods**

14

Why external storage?

- External storage is for reading and writing data to a file.
- Data stored in a file exists beyond the execution period of a software.
- More scalable than internal memory.

15

Physical Storage Media (1/3)

- Cache-fastest and most costly form of storage; volatile; managed by the hardware/operating system.
- Main memory:
  - general-purpose machine instructions operate on data resident in main memory
  - fast access, but generally too small to store the entire dataset
  - sometimes referred to as **core** memory
  - volatile — contents of main memory are usually lost if a power failure or system crash occurs

16

Physical Storage Media (2/3)

- Optical storage – non-volatile. CD-ROM most popular form.
- Write-once, read-many (WORM) optical disks used for archival storage.
- Tape storage – non-volatile, used primarily for backup (to recover from disk failure), and for archival data
  - **sequential-access** – much slower than disk
  - very high capacity
  - tape can be removed from drive ⇒ storage costs much cheaper than disk

17

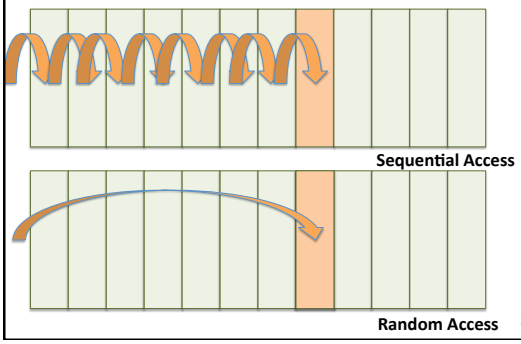
Physical Storage Media (3/3)

Records, Blocks, and Files

- The records of a file are organized into one or more blocks
  - Hardware configuration and the system software of the computer determine the size of the block
- Random accessing file
  - A program can read/write a given block from the file by specifying its block number
  - Reading or writing happens in the block level (not a records)
- Sequential accessing file
  - Records are accessed in a sequential order

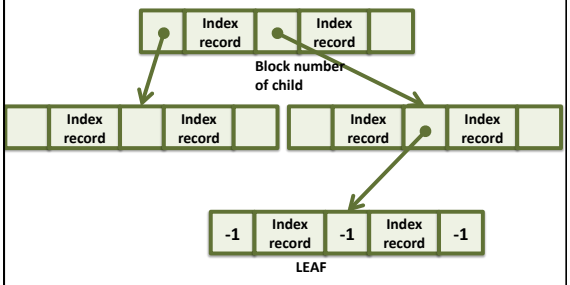
19

Sequential Access Vs. Random Access



20

Using B-Trees to organize blocks of an external file



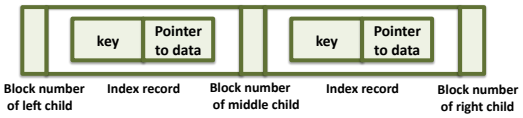
21

Indexing data blocks

- Organize blocks of an external file into a tree structure
  - Use block numbers for child pointers
- Contains up-to three child points (for 2-3 tree)
  - For the null pointer, a child pointer value uses -1

22

A single node of the 2-3 tree



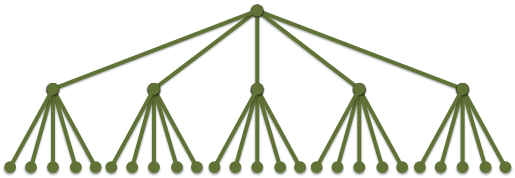
23

How many children the nodes of the search tree can have?

- How big can m be?
- Each node must be large enough to accommodate m child pointers and m-1 records of the form <key, pointer>
- M should be the largest possible integer
  - M child pointers, and m-1 records can fit into a single block of the file

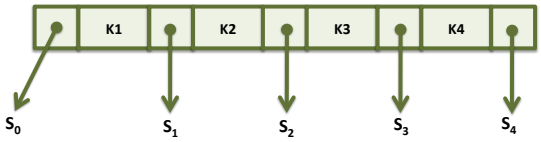
24

A full tree whose internal nodes have five children



25

The format of a single node



26

Insertion into a B-tree

1. Insert the data record into the data file
   - Find block p in the data file into which you can insert the new record
   - Block p is either any block with a vacant slot or a new block
2. Insert a corresponding index record into the index file

27

Deletion from a B-tree

1. Locate the index record in the ***index file***
2. Delete the data record from the ***data file***

28

Midterm Exam II