File Organization

B-Trees II

Reminder

- Final exam
  - The date for the Final has been decided:

  - Saturday, November 16th
  - 8am – 10am
  - 01-2000

Project Notes

- Farmer returned today!!!
  - Look for comments on memory management!

- Parking Lot Problem: due Nov 11th

New plan

- Today: Files 3

- Monday: Ethics
- Tuesday: Final Review
- Then we are done!

Before we start

- Any questions

File Organization

- How to find stuff in a file
  - Improve access time by imposing a defined structure on a file
  - Database applications
  - Advanced searching strategies

Going up a level

- The OS must also:
  - Organize it’s file system for maximum performance (I.e. low access time)
  - File System
- If the file is a searchable database
  - This optimization can be improved by imposing an additional structure on a file.

Organization Strategies

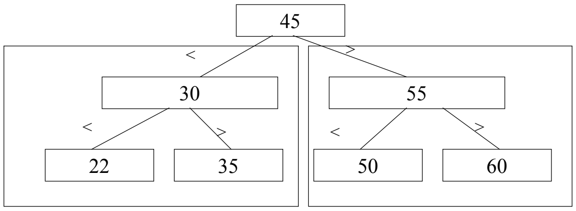
- Sequential
- Indexed
- Hashing
- Tree-based Organization

Recall from CS2

- Binary Search Tree
  - Efficient storage for search / retrieval of “sortable” data.
  - Basic idea
    - Left subtree contains nodes with data less than data at node
    - Right subtree contains nodes with data greater than data at node

Binary Search Trees

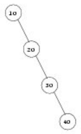
- Branching can also imply ordering of node data



All values less than 45 All values greater than 45

Binary Search Trees

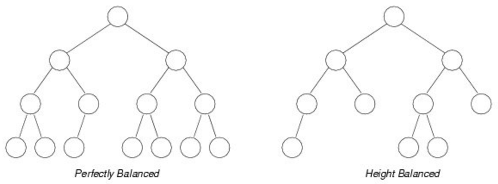
- Searches using binary trees can be done in O(log n) time.
- However, to achieve this, trees must be balanced.
  - Consider:



Binary Search Trees

- Balanced trees
  - A binary tree is perfectly balanced if the number of nodes in the left and right subtrees is no more than 1 for each node.
    - Difficult to insert into a perfectly balanced tree and maintain it’s “perfect” status
  - A binary tree is height balanced if the height of the left and right subtrees of every node (at each level) differs by no more than one.
    - Height balancing is easier to achieve than perfect balancing

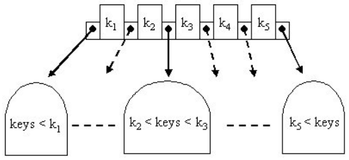
Binary Search Trees



Multiway Search Trees

- A generalization of binary search trees
- Each node has at most m children
- If a node has k children, it has k-1 keys
- The tree is ordered

Multiway Search Trees



B-trees

- A B-Tree (of order m) is a multiway search tree with:
  - All leaves at the same level
  - Each interior node has between m and m/2 non-empty children
  - Each leaf node holds between m/2 to m - 1 keys
  - The root has between 2 and m non-empty children

B-trees

- Example
  - A Btree of order 9, a node may have
    - 9 pointers to children
    - 8 keys
  - Unless it’s the root node, must have
    - At least 5 pointers to children (half or 9, rounded up)
      - For leaf nodes, pointers to children are null.
    - At least 4 key values

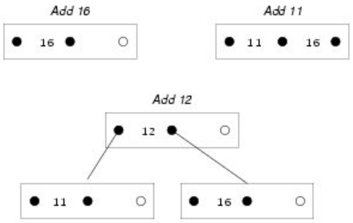
B-trees

- For simplicity we say nodes have pointers and keys.
- Since they are to used for file indexing:
  - Nodes contain key / file location pairs
  - Trees are sorted on key values.
- Questions

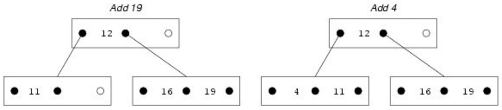
Insertion

- Insertion into a B-Tree is from leaf up rather than from root down
  - Find leaf where new key should be placed and attempt to place it there.
  - If the node is full (overflow)
    - Split the node into 2 nodes
    - Promote the median key up to the orig node's parent
    - Repeat as needed (e.g. promotion causes an overflow)

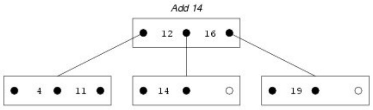
Insertion



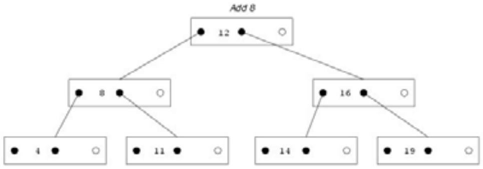
Insertion



Insertion



Insertion

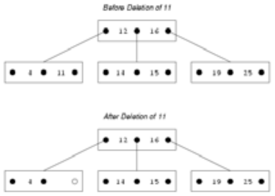


Removal

- A bit more complicated than insertion
  - If a node becomes too empty (underflow), must combine with a sibling node.
  - Process
    - Find node with a given key
    - remove the key
    - Readjust tree to maintain properties

Removal – readjustment

- Case 1
  - Deletion from leaf node – no underflow
    - Simply remove the key from the leaf node

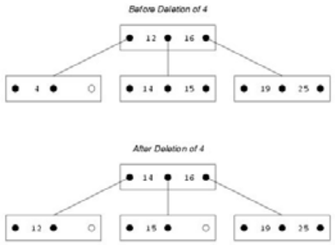


Removal – readjustment

- Case 2
  - Deletion from leaf node – underflow – adjacent sibling non-minimal
    - Steal value from adjacent sibling

Removal – readjustment

- Case 2

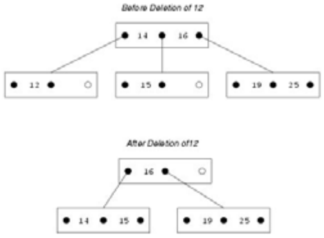


Removal – readjustment

- Case 3 – Deletion from minimal node with both siblings also miminum
  - Combine underflow node with sibling and entry from parent

Removal – readjustment

- Case 3

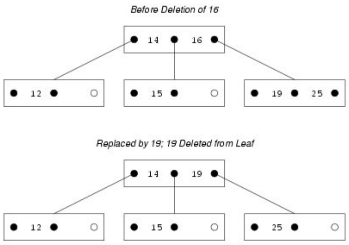


Removal – readjustment

- Case 4 – Remove interior node
  - Replace removed key with immediate successor or predecessor

Removal – readjustment

- Case 4



Before Deletion of 16

Replaced by 19; 19 Deleted from Leaf

Let’s see this in action

- http://sky.fit.qut.edu.au/~maire/baobab/baobab.html

Summary

- Balanced Trees
- Multiway Search Trees
- B-Trees

- Questions?