Announcements

MP5 available, due 3/31, 11:59p.

AVL tree analysis:

Since running times for Insert, Remove and Find are O(h), we'll argue that h = O(log n).

- Defn of big-O:

- Draw two pictures to help us in our reasoning:

- An upper bound on the height for a tree of n nodes is the same as a lower bound on the number of nodes in a tree of height h.

AVL tree analysis:

Putting an upper bound on the height for a tree of n nodes is the same as putting a lower bound on the number of nodes in a tree of height h.

- Define N(h):

- Find a recurrence for N(h):

- We simplify the recurrence:

- Solve the recurrence: (guess a closed form)

AVL tree analysis: prove your guess is correct.

- Thm: An AVL tree of height h has at least $2^{h/2}$ nodes, _____.

Consider an arbitrary AVL tree, and let h denote its height.

Case 1: _____ _____ _____ _____ _____ ____

Case 2: _____ _____ _____ _____ _____ ____

Case 3: _____ then, by an Inductive Hypothesis that says

_____, and since

_____, we know that

_____.

Punchline:

Classic balanced BST structures:

- Red-Black trees – max ht $2\log_2 n$.

  Constant # of rotations for insert, remove, find.

- AVL trees – max ht $1.44\log_2 n$.

  O(log n) rotations upon remove.

Balanced BSTs, pros and cons:

- Pros:
  - Insert, Remove, and Find are always O(log n)
  - An improvement over:
  - Range finding & nearest neighbor
- Cons:
  - Possible to search for single keys faster
  - If data is so big that it doesn't fit in memory it must be stored on disk and we require a different structure.

B-trees (the only “out of core” data structure we’ll discuss)

Can we always fit data into main memory?

So where do we keep the data?

Big-O analysis assumes uniform time for all operations.

But…

The Story on Disks

2GHz machine gives around 2m instructions per _____.

Seek time around _____ for a current hard disk.

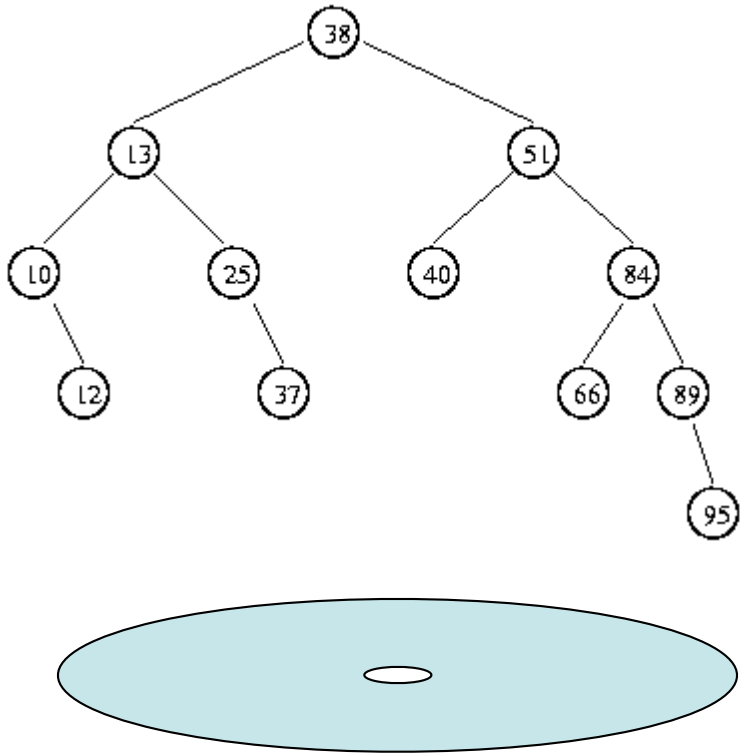
Imagine an AVL tree storing US driving records.

- How many records?
- How much data, in total?
- How deep is the AVL tree?

- How many disk seeks to find a record?

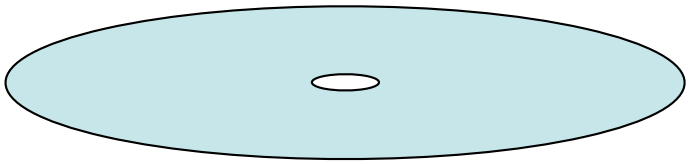
B Trees

Suppose we weren't careful…



B Tree of order m

12	18	27	52	58	63	77	89



Goal: Minimize the number of reads from disk

- Build a tree that uses 1 disk block per node
  - Disk block is the fundamental unit of transfer
- Nodes will have more than 1 key
- Tree should be balanced and shallow
  - In practice branching factors over 1000 often used

http://people.ksp.sk/~kuko/bak/big/