MA/CSSE 473 Day 24

2-3 Trees

Heaps and HeapSort

Problem Reduction

MA/CSSE 473 Day 24

- Late "day" for **HW 9** extends to Friday noon.
- **HW 10** due next Tuesday, Oct 19
- Don't forget the **Convex Hull** implementation problem. It is due **24 days** after I assigned it (due October 21) ; **15** of those days have already passed.
- **HW 11** due Saturday, Oct 23. It is smaller than most assignments.
- **HW 12** due Tuesday, Oct 26.
- **Take-home exam** available Oct 29 (Friday) at 9:55 AM, due Nov 1 (Monday) at 8 AM.
- **Student Questions**
- 2-3 trees
- Binary heaps and heapsort.

Recap: AVL Trees

- Named for authors of original paper, **A**delson-**V**elskii and **L**andis (1962).
- An AVL tree is a height-balanced Binary Search Tree.
- A BST T is **height balanced** if T is empty, or if
  - $| \text{height}(T_L) - \text{height}(T_R) | \leq 1$, and
  - T_L and T_R are both height-balanced.
- Show: Maximum height of an AVL tree with N nodes is $\Theta(\log N)$.
- How do we maintain balance after insertion?
- **Exercise:** Given a pointer to the root of an AVL tree with N nodes, find the height of the tree in log N time.
- Details on balance codes and various rotations are in the CSSE 230 slides that are linked from the schedule page.

Q1-5

2-3 trees

- Another approach to balanced trees
- Keeps all leaves on the same level
- Some non-leaf nodes have 2 keys and 3 subtrees
- Others are regular binary nodes.

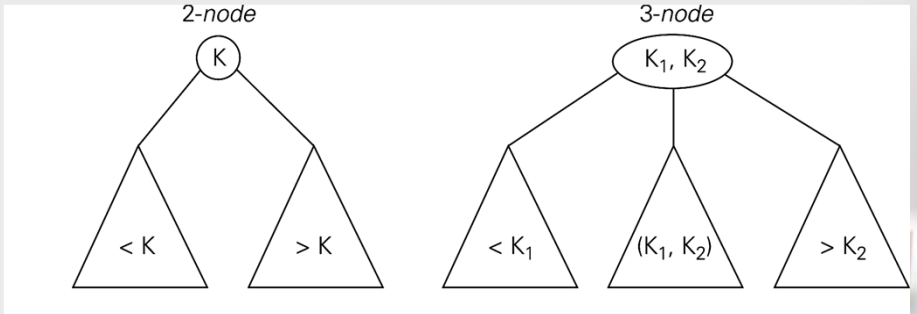


FIGURE 6.7 Two kinds of nodes of a 2-3 tree

Q6

Efficiency of 2-3 tree insertion

- Upper and lower bounds on height of a tree with n elements?
- Worst case insertion and lookup times is proportional to the height of the tree.

Q7

2-3 tree insertion example

- More examples of insertion:
  http://www.cs.ucr.edu/cs14/cs14_06win/slides/2-3_trees_covered.pdf
  http://slady.net/java/bt/view.php?w=450&h=300

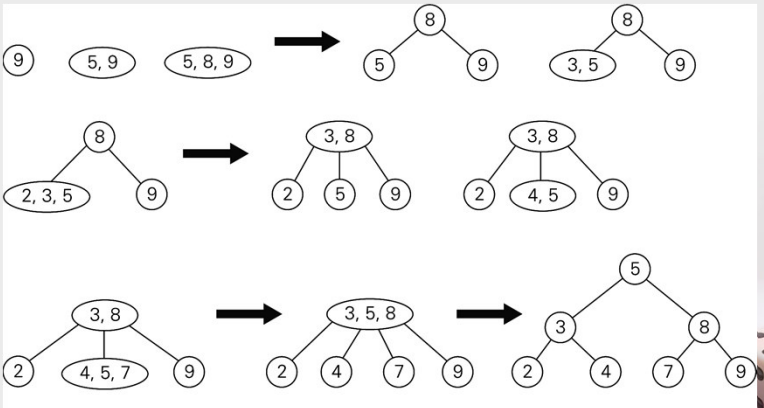
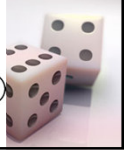
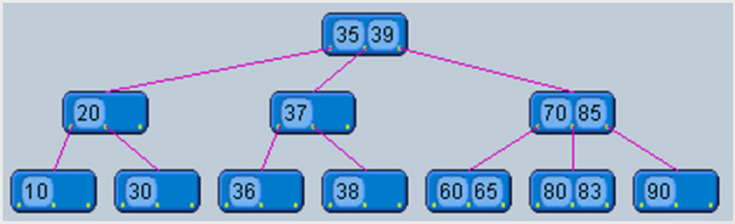


FIGURE 6.8 Construction of a 2-3 tree for the list 9, 5, 8, 3, 2, 4, 7



2-3 Tree insertion practice

- Insert 84 into this tree and show the resulting tree

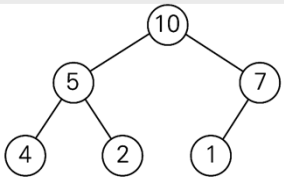


Q8

Binary (max) Heap Quick Review

See also Weiss, Chapter 21

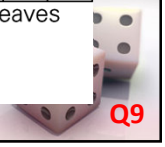
- An almost-complete Binary Tree
  - All levels, except possibly the last, are full
  - On the last level all nodes are as far left as possible
  - No parent is smaller than either of its children
  - A great way to represent a Priority Queue
- Representing a binary heap as an array:



the array representation

index	0	1	2	3	4	5	6
value		10	5	7	4	2	1
		parents			leaves		

FIGURE 6.10 Heap and its array representation



Q9

Insertion and RemoveMax

- Insertion:
  - Insert at the next position to maintain an almost-complete tree, then percolate up to restore heap property.
- RemoveMax:
  - Move last element of the heap to the root, then percolate down to restore heap property.
- Both operations are $\Theta(\log n)$.
- Demo: http://www.cs.auckland.ac.nz/software/AlgAnim/heaps.html

HeapSort

- Arrange array into a heap
- Starting from the root, remove each element from the heap and move to the end of the array.
- Animation: http://www.cs.auckland.ac.nz/software/AlgAnim/heapsort.html
- Faster heap building algorithm: **buildheap** http://students.ceid.upatras.gr/~perisian/data_structure/HeapSort/heap_applet.html

Q10-12