

Analysis of Algorithms CS 3343 Lecture Eleven

Prof. William Winsborough
October 30, 2008

Business

- Turn in Homework 5
 - Problem 7.1 (p.159)
- Reading: 6.1-6.4
- Recall Homework 6 due today: 5.2-3, 5.2-4, 7.2-1, 7.2-4, 7.3-1, 7.3-2, 7.4-1
 - Also: Justify each inequality in the derivation of A.10 on p.1066
- Homework 7 due Thursday 11/6
 - 6.1-1, 6.1-2, 6.1-3 (hint: use induction on the height of the subtree), 6.1-7
 - 6.2-3, 6.2-4, 6.2-5

30 October 2008

Winsborough CS 3343 Lecture 11

2

Heaps and Heapsort

- A heap is a nearly complete binary tree
 - Lowest level may not be complete
- Represented as an array with two attributes:
 - $\text{heap-size}[A] \leq \text{length}[A]$
- Root is $A[1]$
- Navigation from a given node:
 - $\text{Parent}(i)$ return $\lfloor i/2 \rfloor$
 - $\text{Left}(i)$ return $2i$
 - $\text{Right}(i)$ return $2i+1$
 - Programming: bit shift implementations are done in line

30 October 2008

Winsborough CS 3343 Lecture 11

3

Terminology

- Max-heaps
 - Max-heap property: for all i , $A[\text{parent}(i)] \geq A[i]$
- Min-heaps
 - Min-heap property: for all i , $A[\text{parent}(i)] \leq A[i]$
- Height of a node is the number of edges in the longest downward path from the node to a leaf
 - Height of the heap is the height of its root $O(\lg n)$

30 October 2008

Winsborough CS 3343 Lecture 11

4

Max-Heapify

- Subroutine used for manipulating max-heaps
- Inputs: A, i
 - Assumes that binary trees rooted at $\text{Left}(i)$ and $\text{Right}(i)$ are max-heaps
 - But $A[i]$ may be smaller than its children
 - The routine moves such values down in the tree, making the tree rooted at i a max-heap
- Refer to p.130 for routine

30 October 2008

Winsborough CS 3343 Lecture 11

5

Runtime Complexity of Max-Heapify

- Constant time to organize top three nodes
- One recursive call with worst case input size $2n/3$
- $T(n) \leq T(2n/3) + \Theta(1)$
- $T(n) = O(\lg n)$

30 October 2008

Winsborough CS 3343 Lecture 11

6

Building a Heap

- Build the heap bottom up by using Max-Heapify (see p.133)
- Tight upper bound: $O(n)$
 - $O(n \lg n)$ is clear
 - An n -element heap has height $\lfloor \lg n \rfloor$ and at most $\lceil n/2^{h+1} \rceil$ nodes of height h
- Refer to p.135