Priority Queues

4-19-2002

Opening Discussion

- What did we talk about last class?
- Do you have any questions about the assignments? The designs for assignment #6 are supposed to be due on Monday, but since I won't be here you can have that extended until Wednesday.
- What is a priority queue and what do we use them for?

Motivation

- Consider the following problems.
  - You are writing software to control what jobs get sent to a printer. Different jobs have different priorities so that things that matter more should get printed first. How do you set this up in an efficient way?
  - You are writing some type of "event" handling system. Every so often you are given a task and a time when it needs to be started. You need to start them in the proper order.

Operations and Requirements

- For these types of problems we need a container with only three operations. One is to be able to get the minimum or maximum element. The others are to be able to add new elements and remove the minimum.
- The idea would be O(1) for both operations, but that generally isn't attainable. We can get O(1) for finding and O(log n) for the adding/removing.

Heaps

- The data structure we want to use to achieve this is called a heap. A heap is a tree type structure, but it isn't sorted in the same way as the binary trees we discussed. Instead it uses the "Heap-Order Property" where all the nodes below a given node have values higher than it.
- Recursively we can just say, the root is the minimum value in the tree.

Complete Trees as Arrays

- When a tree is complete we can easily represent it as an array. In a complete tree, the left most nodes in a generation get their children first, and nodes get a left child before a right child.
- Note that for a binary tree, the path to a leaf can be expressed as a binary number. If we prefix it with a 1 we get a unique encoding where the first element is 1. This also allows up to quickly find parents.

Inserting to a Heap

- To insert into a binary heap, we put a "bubble" node at the next open space and let it move up the heap until it moves into place.
- At each step it gets swapped with a value greater than it so the heap-order property is maintained.
- This operation takes O(log n) time because the heap is a complete tree.

Removing from a Heap

- To remove the smallest element we simply make the "bubble" node at the top and this time the last element in the heap is what will wind up filling it. We let it "sink" down through the tree. At each step we swap it with the smaller of the two children assuming that it is larger than them. If it isn't it can stop there.

Minute Essay

- I want you to draw the heaps that result from adding the following numbers into an empty heap then removing two of them. 4,8,6,2,3,7
- There will be no class on Monday. On Wednesday my flight arrives at 8:30am so there is a non-zero chance I could miss class that day as well. I will call the office if I'm running late so someone will come in here to let you know.