CS 482 Summer 2005

Homework Assignment #3

Out: July 15 Due: July 19

Question 1

Solve (and prove) the following recurrence equations:

(a) $T(n) = 16T(\frac{n}{4}) + O(n),\ T(1) = O(1)$

(b) $T(n) = T(\frac{n}{2}) + O(n^2),\ T(1) = O(1)$

Question 2

modified from exercise 5 in chapter 5 of the text

Hidden surface removal is a problem in computer graphics that scarcely needs an introduction: when Woody is standing in front of Buzz you should be able to see Woody but not Buzz; when Buzz is standing in front of Woody, . . . well, you get the idea.

The magic of hidden surface removal is that you can often compute things faster than your intuition suggests. Here's a clean geometric example to illustrate a basic speed-up that can be achieved. You are given n non-vertical lines in the plane, labeled $L_1, ..., L_n$, with the i^{th} line specified by the equation $y = a_i x + b_i$. We will make the assumption that no three of the lines all meet at a single point. We say line L_i is uppermost at a given x-coordinate x_0 if its y-coordinate at x_0 is greater than the y-coordinates of all the other lines at $x_0 : a_i x_0 + b_i > a_j x_0 + b_j$ for all $j \neq i$. We say line L_i is visible if there is some x-coordinate at which it is uppermost—intuitively, some portion of it can be seen if you look down from "$y = \infty$."

Give an algorithm that takes n lines as input, and in $O(n \log n)$ time returns all the visible ones **and where they intersect**. Figure 5.10 of the text gives an example.

Hint 1: Start by sorting the edges in order of increasing slope. Notice that the first and last lines in this order will always be visible.

Hint 2: If two lines are both visible, the region in which the line of smaller slope is uppermost lies to the left of the region in which the line of larger slope is uppermost.

Hint 3: When you combine two sub-solutions, consider each intersection point and consider which sub-solution has the uppermost line at this point.

Question 3

exercise 6 in chapter 5 of the text

Consider an n-node complete binary tree T, where $n = 2^d - 1$ for some d. Each node v of T is labeled with a real number x_v. You may assume that the real numbers labeling the nodes are all distinct. A node v of T is a local minimum if the label x_v is less than the label x_w for all nodes w that are joined to v by an edge. You are given such a complete binary tree T, but the labeling is only specified in the following implicit way: for each node v, you can determine the value x_v by probing the node v. Show how to find a local minimum of T using only $O(\log n)$ probes to the nodes of T.