Due: Day 11

When presenting an algorithm, make sure you have:

- A clear (pseudo-code) description of the algorithm
- A proof that it is correct (loop invariants may help)
- An analysis of its runtime (stating the runtime isn’t sufficient; provide a proof or justification).

For full credit, you need all three pieces.

Exercises

1. 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 \( T \).

2. Suppose now that you’re given an \( n \times n \) grid graph \( G \). (An \( n \times n \) grid graph is just the adjacency graph of an \( n \times n \) chessboard. To be completely precise, it is a graph whose node set is the set of all ordered pairs of natural numbers \( (i, j) \), where \( 1 \leq i \leq n \) and \( 1 \leq j \leq n \); the nodes \( (i, j) \) and \( (k, l) \) are joined by an edge if and only if \( |i - k| + |j - l| = 1 \).)

We use some of the terminology of the previous question. Again, each node \( v \) is labeled by a real number \( x_v \); you may assume that all these labels are distinct. Show how to find a local minimum of \( G \) using only \( O(n) \) probes to the nodes of \( G \). (Note that \( G \) has \( n^2 \) nodes.)
3. CLRS, exercise 7-6

4. For two distinct points \( p_1 = (x_1, y_1) \) and \( p_2 = (x_2, y_2) \) in the plane, we say that \( p_1 \) is subordinate to \( p_2 \) if \( x_2 \geq x_1 \) and \( y_2 \geq y_1 \). Design an \( O(n \log n) \) algorithm that, given a set of \( n \) points, finds all points that are not subordinate to any other point.

5. Netflix gives movie suggestions based on how close your movie ratings are to others. In order to see how close your ratings are, the folks at Netflix compare your ranking of \( n \) movies compared to another subscriber’s ranking of those same numbers. Rather than just compare the number of matching elements, they compare the number of out-of-order pairs.

For example, if your ranking of movies 1-5 is \(<1, 3, 5, 2, 4>\) and another’s subscriber’s rating is \(<2, 3, 5, 1, 4>\), the number of out-of-order pairs is 5: \((1, 3), (1, 5), (1, 2), (3, 2), (5, 2)\).

Clearly, the least number of out-of-order pairs is 0. Given two ranking of \( n \) movies, what is the greatest number of out-of-order pairs?

Give an \( O(n \log n) \) algorithm that determine the number of out-of-order pairs in an input consisting of two rankings, each a permutation of 1..\( n \).