Advanced Topics in Algorithms
Computer Science 182b
Spring 2011
Homework 4, Due Wednesday, February 16

1. [**20 Points**] **The Wall Problem Revisited.** Prof. I. Lai of the Pasadena Institute claims that the online algorithm that we saw in class for the robot "Wall Problem" would work just as well if we got rid of the nuisance window and simply did the sweeping step. Show that he's wrong (as usual) by showing that without the nuisance window, there exist request sequences for which the algorithm does *asymptotically* worse than $\sqrt{n}$ times optimal.

2. [**25 Points**] **The Wall Problem ReRevisited!** Consider the Wall Problem again. Now assume that all of the rectangles are squares. As before, the robot cannot see an obstacle until it touches it. You may assume that when the robot touches a square, it can immediately detect the distance to the the corners to its north and south. You should also assume that square obstacles may straddle the goal line. Show that a simpler algorithm than the one we examined in class can achieve a *constant* competitive ratio in this case (rather than $O(\sqrt{n})$ ratio)! Your algorithm and its competitive analysis should be described clearly and you should give the best possible constant competitive ratio that you can find.

3. [**15 Points**] **Union-Find with Partial Path Compression.** In class we examined the union-find data structure using union-by-size and path compression. One disadvantage of path compression is that it is a two-phase process: First we must "climb" the path to find the root. Then, we climb the path a second time and set the parent of each node on that path to be the root.

   The famous Shmorbodian theoretical computer scientist, Professor Y. Woerksohaard, proposes the following simpler approach called *partial path compression*: As we "climb" a path from a node to the root of its tree, each node on the path has its parent pointer changed to point to its grandparent. This process can be done in just one pass as we climb up the path. Professor Woerksohaard claims that this approach still allows us to perform any sequence of n MAKESET, UNION, and FIND operations in time $O(n \log^* n)$ time. Is this true or false? If true, give a short but precise explanation (you may cite any part of the proof that we did in class without replicating it). If false, give a short but precise counter-example.

4. [**25 Points**] **Union-Find Again.** Consider again the union-find data structure using union-by-size and path compression. Assume that we will perform

a sequence of n MAKESET, UNION, and FIND operations where all of the MAKESETs are performed before the UNIONs and all of the UNIONs are done before the FINDs. Show that the total running time is now asymptotically even better than $\Theta(n \log^* n)$.

5. [**20 Points**] **Double-Rotations are the Secret to All Happiness!** Recall that splay trees use two types of double-rotation ("zig-zig" and "zig-zag") and one type of single-rotation ("zig"). Using these rotation rules, we were able to show that any sequence of n tree operations (such as INSERT, FIND, and DELETE) can take at most $O(n \log n)$ time. Show that if we simply used a sequence of standard single-rotations to splay a node to the root when we perform a tree operation then there exists a sequence of n tree operations that take take $\Omega(n^2)$ time. (Using just INSERT and FIND as your tree operations will suffice to show this.) Aha! Double-rotations really are the *secret to all happiness*!