Algorithms
Computer Science 140 & Mathematics 168
Spring 2009
Homework 12b
Due Thursday, April 30

This is the last assignment of the semester! Note that it is due on Thursday, rather than Tuesday. If you have an algorithms "ruble" remaining, you may redeem it for an extension until Friday at 5 PM.

1. **[10 Points] "Implementing" the DC Line Algorithm.** In class we examined the online DC algorithm for the real line with the Euclidean distance metric. Recall that in this algorithm if the request is outside the convex hull of the servers, we shuttle the nearest server to serve the request. If the request is inside the convex hull, we shuttle the two servers adjacent to the request towards the request at equal speed (at least one of the two servers then arrives to serve the request). (As we proved in class, we could use a "lazy" version of this algorithm in which only one server moves on any request.)

   Imagine that we have k servers and we wish to construct a centralized algorithm that dispatches robots according to the DC algorithm. Our objective is to minimize the *total computation time*, over a sequence of n requests, that the algorithm incurs in making its dispatch decisions. Describe in detail the data structure (or structures) that you would use to implement the centralized dispatch algorithm, how these data structures would be used in your implementation, and derive the total computation time for computing the n dispatch decisions. (The run-time will be a function of n and k.) For full credit, try to make your algorithm very fast.

2. **[35 Points] DC-Tree!** You've been hired by `Nile.com`, a new online bookstore. The floor of a Nile warehouse has tracks embedded in the floor and a collection of k robots move along these tracks picking up books, etc. The network of tracks form a rectilinear tree: There are no cycles and each piece of track is straight. Notice that the tracks are continuous so a robot can be located at any point along the track. In other words, any (graph theoretic) tree in which the edges are drawn as straight line segments forms a legitimate network of track except that the robots can be located anywhere along any edge of the tree.

   Generalize the *DC* algorithm for the real line to an algorithm for rectilinear

trees. Describe the algorithm very carefully. Then show that your algorithm is still k-competitive. How cool is that!?!

3. **[25 Points] A Neat Application of the DC Tree Algorithm.** Now consider a finite metric space which is a weighted graph with N vertices. That is, the requests and the servers can only be located at the N vertices. The edge weights, of course, satisfy the triangle inequality. In this problem you will show that there is a $(N-1)k$-competitive online algorithm for this k-server problem.

   (a) First we will need a little technical lemma. Consider a weighted graph G and a minimum spanning tree T for G. Let (u, v) be an edge of weight d in G. Note that edge (u, v) may or may not be in T. However, prove that there must be a path from u to v in T whose total weight does not exceed $(N-1)d$.

   (b) Now describe a $(N-1)k$-competitive online algorithm for this problem. Remember, the metric space is finite! The servers and requests must be at the vertices of the graph.