CS 141
Advanced Topics in Algorithms
Spring 2004
Problem Set 7b
Due Monday, March 8 in class

1. **[15 Points] Majority Circuits!** A majority circuit with n boolean inputs $x_1, x_2, \ldots, x_n$ outputs a 1 if $x_1 + x_2 + \ldots + x_n > n/2$ and outputs 0 otherwise. Describe a circuit for an n-input majority function that has depth $O(\lg n)$. Describe and analyze your circuit carefully. What is the size of your circuit?

2. **[15 Points] More fun with Norelco Shavers!** Give efficient EREW algorithms to compute the preorder, inorder, and postorder numberings for an arbitrary binary tree with n nodes. For each of the three parts of this problem (preorder, inorder, and postorder), describe the values that must be placed in processors A, B, C, explain where the solution is found, and briefly explain why your algorithm works.

3. **[15 Points] Parallel Matrix Multiplication!** In this problem we'll investigate the problem of finding the product of two $n \times n$ matrices using a shared-memory parallel computer with n^3 processors.

   (a) Show that this can be done in $O(\lg n)$ time if the machine has CREW. Is this algorithm work-efficient? No!

   (b) Now describe the fastest algorithm you can find for this problem that operates on a machine with EREW.

4. **[15 Points] Hurts Car Rental!** Hurts Car Rental provides a variety of maps and travel information for its customers. Customers often complain that certain travel routes are boring because they involve long stretches of road with no attractions in between. Hurts would like to advise travelers of the boringness of certain trips.

   Let G be a directed graph with n vertices (representing cities, meat processing plants, and other major attractions) and m weighted edges (representing roads - potentially one-way roads- and their lengths). The *boringness* of a path between vertices u and v in G is defined to be the edge of maximum weight on that path. The *cob (coefficient of boringness)* of a pair (u, v) is the minimum boringness over all paths from u to v.

   Devise a parallel algorithm which computes the cob of all pairs of vertices in the network simultaneously. Notice that there are n^2 pairs of vertices. You may assume that the vertices are numbered 1 through n and the edge distances are specified in a $n \times n$ matrix in which the entry at location (i, j) is the weight of the edge from i to j if there is one and is ∞ otherwise. The algorithm leaves the computed cobs in a $n \times n$ matrix as well.

   Your algorithm must work in $O(n)$ time. How many processors do you use? What model (EREW, CREW, or CRCW) does your algorithm require? (*H*int: First find a standard one-processor algorithm for this problem. Then parallelize it.)

5. **[15 Points] Sorting in Parallel!** Give a parallel algorithm for sorting n numbers in $O(\log n)$ time using n^2 processors. What model (EREW, CREW, or CRCW) does your algorithm require? (Use the "weakest" model that you can! EREW is weaker than CREW is weaker than CRCW.)