CS 1510 Midterm 3
Fall 1996

1. (40 points)

   (a) Briefly explain how to find the logical AND of n bits in time $O(\log n)$ using n processors on a EREW PRAM.

   (b) Briefly explain how to find the logical AND of n bits in time $O(1)$ using n processors on a CRCW PRAM.

   (c) Define the efficiency of a parallel algorithm.

   (d) The input for the simplest variant of the knapsack problem is n positive integers $x_1, \ldots, x_n$ and an integer L. In class and in the book, an $O(nL)$ time algorithm was given for the knapsack problem. Briefly explain why this is not a polynomial time algorithm.

   (e) Your boss tells you that you must come up with an algorithm for a problem C. You know that C is NP-hard. How does this fact effect your search for the best possible algorithm for C?

2. (20 points) Consider the following 2Clique problem:

   INPUT: A undirected graph G and an integer k.

   OUTPUT: 1 if G has two vertex disjoint cliques of size k, and 0 otherwise.

   Show that this problem is NP-hard. Use the fact that the clique problem in NP-complete. The input to the clique problem is an undirected graph H and an integer j. The output should be 1 if H contains a clique of size j and 0 otherwise. Note that a clique is a mutually adjacent collection of vertices. Two cliques are disjoint if they do not share any vertices in common.

3. (20 points) The input to this problem is a collection of positive integers $x_1 \ldots x_n$ and an integer L. The problem is to determine if there are two disjoint subsets S and T of the x_i's that both sum to L. Give an algorithm for this problem whose running time is polynomial in n and L. Note that S and T being disjoint means that the intersection of S and T is empty. HINT: Find a dynamic programming algorithm.

4. (20 points) We consider the problem of computing F_n, the nth Fibonacci number, given an integer n as input. Show how to solve this problem in time $O(\log n)$ on a EREW PRAM with n processors. Make the unrealistic assumption that F_n will fit within one word of memory for all n, that is, assume that all arithmetic operations take constant time. Recall that F_n is defined by the following recurrence: $F_0 = F_1 = 1$, and $F_n = F_{n-1} + F_{n-2}$ for $n > 1$.

   HINT: Note that for $j > 0$

$$\begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} F_j \\ F_{j-1} \end{bmatrix} = \begin{bmatrix} F_{j+1} \\ F_j \end{bmatrix}$$