CS 222 Fall 2005 Midterm

Open book and notes - but only the books and notes used in this class. The exam may be long. Write answers that are sufficient but to the point. The points for each question are written in parens.

1. (10 pts) Suppose you have a black box that can take in a set of distinct numbers and return the median of the numbers (or one median in the case that the set is of even size), plus a partioning of the remaining numbers into two groups: the group of numbers less than the chosen median, and the group of numbers greater than the chosen median.

Describe an algorithm that uses the black box to sort an unsorted set of n numbers. The algorithm cannot compare two numbers - only the black box can do that. However, the algorithm can count the number of elements in a set. Don't worry about low level programming details, just the algorithmic essentials.

Let $O(f(k))$ denote the time the black box takes to process a set of size k. State (without detail) the best $f(k)$ you know. Using that $f(k)$, write out the recurrence relation for the total time this algorithm uses to sort n numbers. Solve the recurrence. Explain your work sufficiently.

2. (15 pts) A directed graph G is a graph where the edges are directed. A rooted directed tree T is a tree with directed edges so that from every node v in T, there is a directed path from v to the root node of T. A spanning directed tree of G is a rooted directed tree in G that contains every node of G.

Consider the problem of determining whether an unknown directed graph contains a spanning directed tree. The class of permitted algorithms is modeled as follows: the graph is held in a black box which is inacessable to the algorithm; The algorithm learns about the graph by repeatedly specifying two nodes i, j and asking the box if the graph contains a directed edge from i to j. Note that i may be bigger or smaller than j. By asking questions of this type, the algorithm must determine whether or not the graph in the box contains a spanning directed tree. Describe an adversary strategy that shows that any algorithm in the above class must, in worst case, ask $n(n-1)/2$ questions.

3. (10 pts) Given two strings S_1 and S_2 whose lengths add to n, we want to find every suffix of S_1 that exactly matches a prefix of S_2. Show how to use the Z-algorithm to find all these suffixes using $O(n)$ total operations.

4. (10 pts) Suppose T is a rooted tree where each node has at most two children. There is a message sitting at the root of T and we want to transmit copies of the message to all the nodes in the tree. A node can only send a copy of the message if it has already received a copy from its parent node. If a node in the tree has already received the message, then in each successive time unit, it can pass a copy of the message to *one* of its children. So in the first time-unit, the root can send the message to one of its children, say node v, and in the next time unit both the root and node v can send a copy of the message, assuming the root has a child other than v and that v has a child. (You can think of this as a phone-tree where each person who has received the message can call one other person in each time unit.)

The problem is to determine the least amount of time (number of time units) needed to transmit a copy of the message to all the nodes in the tree.

I don't think the problem can be solved by a concise strategy such as: A node that has received a copy of the message always sends copies of the message to its children in order of the number of nodes below each child, largest number first. The example below shows that this strategy does not work.

Problem: Give a DP algorithm to solve the problem and analyze the running time.

5. (15 pts) In the RNA folding problem discussed in class, a symbol is not permitted to be matched to one of its immediate neighbors, left or right. The folding problem was solved by DP and ran in $O(n^3)$ time. Now suppose we permit any symbol to match a neighbor, provided of course that the type of the neighbor is the opposite of the type of the symbol. Does this allow a faster solution? If so, what is the resulting time bound? Justify your answer.

Hint: Relate this to other problems you have seen.