CS 441T/539T Advanced Algorithms	Fall Semester, 2002

Exam 1

October 10, 2002

Please write all solutions clearly and legibly in the space provided. A list of available hints is on the chalkboard. If you are given a hint with cost of 3 points on a 10 point problem then the maximum score possible is 7 points.

1. (5 pts) Here we look at a nearly completed hand-simulation for the canoe rental problem from Homework 2. YOU SHOULD BE ABLE TO COMPLETE THIS PROBLEM IN 5 MINUTES. DO NOT SPEND MORE THAN 10 MINUTES HERE. Recall that $f_{i,j}$ is the fee for renting a canoe from post i to post j for $1 \leq i < j \leq n$. Let $m[i]$ be the rental cost for the best solution to go from post i to post n for $1 \leq i \leq n$. As proven in the HW 2 solutions,

$$m[i] = \begin{cases} 0 & \text{if } i = n \\ \min\limits_{i<j\leq n} (f_{i,j} + m[j]) & \text{otherwise} \end{cases}$$

A secondary array s will hold the choices made. In particular, $s[i] = j$ indicates that the optimal solution for going from post i to post n, made the next canoe change at post j. Consider the following input

```
    2  3  4  5  6  7  8
   ----------------------
1 | 4  3  7  5  8 11 10
2 |    4  2  3  1  5  9
3 |       3  1  4  8 12
4 |          4  3  2 11
5 |             4  8  6
6 |                3  7
7 |                   1
```

where the entry in row i, column j is the value of $f_{i,j}$. Below are the arrays m and s with all but the first entry already computed.

```
m[1] m[2] m[3] m[4] m[5] m[6] m[7] m[8]
---------------------------------------
       5    6    3    6    4    1    0


s[1] s[2] s[3] s[4] s[5] s[6] s[7] s[8]
---------------------------------------
       4    4    7    8    7    8    8
```

(a) Complete the missing two entries above (1 in each table). Very briefly show your work.

(b) Now use the above to give the optimal solution. Again, just very briefly explain how you have done this.

2. (15 pts) You are given an n by n grid where each grid square holds a real-valued number (which might be negative). Let grid square $(1, 1)$ correspond to the lower left corner and let (n, n) correspond to the upper right corner. Let $p_{i,j}$ be the value on grid square (i, j). You begin at grid square $(1, 1)$. You are to go to grid square (n, n). However, you are only allowed to move one square to the right or move one square up. Your goal is to find a path from $(1, 1)$ to (n, n) such that the sum of the values on the grid squares you go on is maximized.

   In this problem you MUST solve this problem using DYNAMIC PROGRAMMING. (While you could formulate this as a shortest path problem, you will get a more efficient solution using dynamic programming). Try to make your algorithm as efficient as you can. To help ensure you don't forget anything, a template has been provided.

   The general subproblem form that I will use is as follows:

   Here is the recursive definition:

Here is a proof that the recursive definition I gave will always yield an optimal solution to the given subproblem.

Here is my time complexity analysis (i.e. the time complexity of my solution, along with a brief explanation of how I derived it).

3. (20 pts) In each of the following two problems a greedy algorithm is suggested. You are to prove whether or not the given greedy algorithm computes an optimal solution.

   (a) You have n people who can work at a store, where person i will work from time b_i until time e_i. Also, you are given an opening time b and closing time e for the store. The goal is to find the minimum number of people to work for the day so that at least one person is at the store between times b and e.

   Here is the proposed greedy algorithm. Sort the input so that $e_1 \leq e_2 \leq \cdots \leq e_n$. Let P be the set of people who can work at time b. That is, $P = \{i : b_i \leq b \leq e_i\}$. Select the person j in P with the highest index (i.e the latest ending time). Schedule person j to work from b_j to e_j. Then repeat this procedure for people $p_{j+1}, \ldots, n$ to cover times from e_j to e.

(b) Consider the following scheduling problem. We have n jobs, which can be processed on a machine. (Only one job can be processed at a time on the machine.) Job i starts at time s_i, ends at time e_i, and results in a profit of p_i. The goal is to find a schedule that results in the maximum total profit.

Here is the proposed greedy algorithm. Sort the input so that $p_1 \geq p_2 \geq \cdots \geq p_n$. Consider the jobs in order of job 1 to job n. If job i doesn't conflict with the previously schedule jobs then schedule it from time s_i to e_i. Otherwise, job i is not scheduled.

4. (15 pts) In the previous problem, you should have found that one of the proposed algorithm was optimal and the other was not. If you didn't, consider taking a hint for problem 3 which will tell you which one is optimal.

   In this problem you are to use **dynamic programming** to give an algorithm guaranteed to yield an optimal solution for the problem of 3a or 3b in which the proposed greedy algorithm does not yield an optimal solutions. Try to make your algorithm as efficient as you can. To help ensure you don't forget anything, a template has been provided.

   The general subproblem form that I will use is as follows:

   Here is the recursive definition:

Here is a proof that the recursive definition I gave will always yield an optimal solution to the given subproblem.

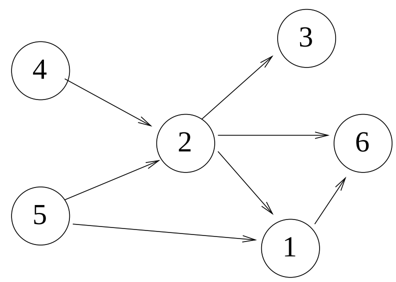
Here is my time complexity analysis (i.e. the time complexity of my solution, along with a brief explanation of how I derived it).

THE CS 441T EXAM ENDS HERE. (If a CS 441T student does the next problem we will note it as extra credit, but no points will be added to your total. Hence CS 441T students should do all they can on problems 1-4 before even considering problem 5.)

THIS PROBLEM IS REQUIRED ONLY FOR CS 539 STUDENTS.

5. (15 pts) Consider the following scheduling problem. There are n jobs to be processed by a single machine which can execute at most one job at a time.

   - Each job j requires a processing time of p_j and specifies a "anger" function $f_j(t)$ which how angry the customer will be if job j is not completed until time t. The "anger" function can be any non-decreasing function. That is, for $t_1 < t_2$, it must be $f_j(t_1) \leq f_j(t_2)$. In other words, the customer will only get angrier as it take longer. But these anger functions could look very different. One customer may not be angry at all until some time at which he because very angry. Another customer may get a little bit angrier over time.
   - As part of the input you are also given a set of precedence constraints of the form job i must be scheduled prior to job j. The precedence constraints form a directed acyclic graph. For example:



   - Your goal is as follows. Let S be a legal schedule (i.e. the precedence constraints are satisfied and only one job is processed at a time). For $1 \leq j \leq n$, let C_j be the time when job j is completed in S. The goal is to find the schedule S that **minimizes** $\max_{1 \leq j \leq n} (f_j(C_j))$. That is, you want to minimize the highest anger that will be incurred.

Give a greedy algorithm that optimally solves this problem and prove it is correct. You need NOT analyze the time complexity.

Additional Work for Problem ____________

Additional Work for Problem ______________

Problem	Points Possible	Points Received
1	5	
2	15	
3a	20	
3b		
4	15	
5*	15	
total	55 + 15	