

Exam 1

October 7, 2003

Please write all solutions clearly and legibly in the space provided.

1. (20 pts) Here we consider the problem of optimally deciding where to put line breaks when formatting text with the goal of *minimizing the number of lines used* in the layout. Let M be the maximum number of characters that can be placed on a line. The input is the value of M and sequence of n words w_1, w_2, \dots, w_n of lengths $\ell_1, \ell_2, \dots, \ell_n$, measured in characters where $\ell_i \leq M$ for all i .

(5 pts) Give a **greedy** algorithm that will optimally solve this problem. Describe your algorithm as succinctly as possible in the space provided. No pseudo-code or implementational details are needed.

(15 pts) Now prove that your greedy algorithm will minimize the number of lines in the layout.

2. (35 points) Give a dynamic programming solution to the following problem. The input to this problem is a complete binary tree T with root r where each edge in T is directed towards the leaves and has a real-valued weight which may be negative, zero, or positive. You are to give a linear time algorithm to find a maximum weight directed path in T .

Note: A complete binary tree is one in which every internal node has exactly two children. Here's some notation for you to use.

- For x a node in the tree, let $T(x)$ be the subtree rooted at node x . So $T = T(r)$.
- For x a node in the tree, let $\text{Leaf}(x)$ be a boolean that is true exactly when x is a leaf.
- For x an internal node in the tree, let $\text{Left}(x)$ be the node that is x 's left child, and let $\text{LeftEdgeWeight}(x)$ be the weight on the edge from x to $\text{Left}(x)$.
- For x an internal node in the tree, let $\text{Right}(x)$ be the node that is x 's right child, and let $\text{RightEdgeWeight}(x)$ be the weight on the edge from x to $\text{Right}(x)$.

(8 pts) Write the general subproblem form for your solution. You may find it necessary to do some additional computation to compute the final answer

(2 pts) The cost of the maximum weight path in T is computed from the subproblem solutions as follows:

(12 pts) Write the recursive definition:

(8 pts) Prove that your recursive definition is correct.

(5 pts) Provide a time complexity analysis for your solution, along with a brief explanation of how you derived it. Let n be the number of nodes in the tree T .

3. (25 pts) Consider the following problem. Given a square grid of n rows and n columns, integers r_1, \dots, r_n , and c_1, \dots, c_n , is it possible to place pebbles on the board so that for $1 \leq i \leq n$, row i has exactly r_i pebbles and column i has exactly c_i pebbles.

Here is a greedy algorithm that solves this problem. While there is some $r_i > 0$ and some $c_j > 0$, repeat the following: Let x be the maximum index among all rows i for which $r_i > 0$. (That is, row x is the last row that needs a pebble.) Let column y be such that $c_y \geq c_j$ for all j . (That is, column y needs at least as many pebbles as any other column.) Place a pebble at square (x, y) , decrement r_x by 1 and decrement c_y by 1. When the loop terminates, if all r_i and c_i are 0 then return “yes” (a solution exists) and otherwise return “no.”

Let (x, y) be the first pebble placed by the above algorithm. For this exam question you need just prove that: **If there is a solution to this problem, then there exists a solution in which a pebble is placed at square (x, y) .**

More Space for Problem

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 1-3 before even considering problem 4.)

THIS PROBLEM IS REQUIRED ONLY FOR CS 539 STUDENTS.

4. (5 pts) Complete the proof that the algorithm from Problem 3 is correct.

Problem	Points Possible	Points Received
1	20	
2	35	
3	25	
4*	5	
total	80 + 5	

Available Hints:

1. Problem 1: 5 pts to give the greedy algorithm
2. Problem 1: 2 pts to say whether a greedy algorithm you've written is correct.
3. Problem 1: 3 pts for some definitions to help get the greedy choice proof started.
4. Problem 2: 8 pts to give the general subproblem form
5. Problem 2: 3 pts to say whether a general subproblem form you've written will work.
6. Problem 2: 10 pts to be given the complete recursive definition
7. Problem 3: 10 pts to be given a pretty good start on the proof.