Problem Set 5 (due Wednesday, March 30)

1. (7 + 7 = 14 points) Project management

Suppose you are a high-level manager in a software firm and you are managing n software projects. You are asked to assign m of the programmers in your firm among these n projects. Assume that all of the programmers are equally competent.

After some careful thought, you have figured out how much benefit i programmers will bring to project j. View this benefit as a number. Formally put, for each project j, you have computed an array $A_j[0..m]$ where $A_j[i]$ is the benefit obtained by assigning i programmers to project j. Assume that $A_j[i]$ is nondecreasing with increasing i.

(a) Nonincreasing marginal benefits: Make the economically sound assumption that the marginal benefit obtained by assigning an ith programmer to a project is nonincreasing as i increases. Thus, for all j and $i \geq 1$, $A_j[i+1] - A_j[i] \leq A_j[i] - A_j[i-1]$.

Design a greedy algorithm to determine how many programmers you will assign to each project such that the total benefit obtained over all projects is maximized. Justify the correctness of your algorithm and analyze its running time.

(b) General benefits: Now *remove* the assumption made above that the marginal benefit obtained by assigning an ith programmer to a project is nonincreasing as i increases. Design a dynamic programming algorithm to determine how many programmers you will assign to each project such that the total benefit obtained over all projects is maximized. Analyze its running time.

2. (8 points) Stacking stones

Consider a set of n rectangular paving stones where the ith stone is l_i units long and w_i units wide, $l_i \geq w_i \geq 1$. Assume that paving stone i can be stacked on top of paving stone j iff $l_i \leq l_j$ and $w_i \leq w_j$. Give an efficient dynamic programming algorithm for computing the maximum number of paving stones that can be stacked together. Briefly justify its correctness and analyze the asymptotic running time of your algorithm.

3. (2 + 8 = 10 points) Evolutionary trees

Problem 6.30 of text.

4. (8 points) Finding an optimal seating arrangement

Several families go out to dinner together. To increase their social interaction, they would like to sit at tables so that no two members of the same family are at the same table. Show how to formulate the problem of finding a seating arrangement that meets this objective as a maximum flow problem.

Assume that the dinner contingent has n families and the ith family has a_i members. Also assume that m tables are available and the jth table has a seating capacity of b_j.