Problem Set 7

1. Let G be a directed graph with vertices numbered 1 through n. Suppose that all edges of G are of the form (i, j) with $i < j$ (so edges always point from lower numbered vertices to higher vertices).

   (a) Show how to find the longest path in G in $O(m + n)$ time. The path can start and end at any vertex.

   (b) Show how to turn the Longest Increasing Subsequence problem into an instance of part (a).

2. At each of n equally spaced points along a highway you're considering purchasing a billboard. For each position i, you've projected that putting up a billboard in position i will net you a revenue of $R[i]$. However, due to local regulations, billboards must be spaced at least k positions apart.

   (a) Give an $O(n)$ algorithm to find the maximum revenue that you can make.

   (b) Suppose that in addition to the rule above, you're only able to put up M billboards total. Give an $O(Mn)$ algorithm to find the maximum revenue that you can make.

3. Say that a sequence $a_1, a_2, \ldots, a_k$ is "zigagging" if either

   $$a_1 \leq a_2 \geq a_3 \leq \ldots$$

   or

   $$a_1 \geq a_2 \leq a_3 \geq \ldots$$

   Given an array A of length n, give an $O(n^2)$ algorithm to find the length of the longest zigzagging subsequence.

4. Let G be an undirected graph. The Hamiltonian Path Problem asks whether there is a path in G that visits every vertex exactly once.

Such a path is called a Hamiltonian Path. We will soon see that there (probably) no polynomial time algorithm for this problem. But we can still do a little better than brute force.

(a) Consider the following brute force algorithm. Iterate over all permutations of the vertices. For each permutation, check if the permutation corresponds to a Hamiltonian path. What is the runtime of this algorithm?

(b) Give a dynamic programming algorithm that runs in $O(n^2 2^n)$ time.

(c) Show that this is an asymptotic improvement over the algorithm in part (a).

1 Optional

1. In a Candy-Crush-like game you're given a linear array consisting of gems of various colors. You can blow up any consecutive sequence of k gems with the same color earning k^2 points. These gems are then removed from the array. You can keep blowing up sequences of gems until there are no gems left. Give a polynomial time algorithm to find the maximum score that you can achieve.