
CS 473: Undergraduate Algorithms, Spring 2009

HBS 3

1. Change your recursive solutions for the following problems to efficient algorithms (Hint: use dynamic programming!).
 - (a) Consider two horizontal lines l_1 and l_2 in the plane. There are n points on l_1 with x -coordinates $A = a_1, a_2, \dots, a_n$ and there are n points on l_2 with x -coordinates $B = b_1, b_2, \dots, b_n$. Design an algorithm to compute, given A and B , a largest set S of non-intersecting line segments subject to the following restrictions:
 - i. Any segment in S connects a_i to b_i for some $i(1 \leq i \leq n)$.
 - ii. Any two segments in S do not intersect.
 - (b) Given a string of letters $Y = y_1 y_2 \dots y_n$, a segmentation of Y is a partition of its letters into contiguous blocks of letters (also called words). Each word has a quality that can be computed by a given oracle (e.g. you can call *quality("meet")* to get the quality of the word "meet"). The quality of a segmentation is equal to the sum over the qualities of its words. Each call to the oracle takes linear time in terms of the argument; that is *quality(S)* takes $O(|S|)$.
Using the given oracle, give an algorithm that takes a string Y and computes a segmentation of maximum total quality.
2. Give a polynomial time algorithm which given two strings A and B returns the longest sequence S that is a subsequence of A and B .
3. Consider a rooted tree T . Assume the root has a message to send to all nodes. At the beginning only the root has the message. If a node has the message, it can forward it to one of its children at each time step. Design an algorithm to find the minimum number of time steps required for the message to be delivered to all nodes.