RWTH Aachen University **WS 2010/11**
Lehrgebiet Theoretische Informatik Tutorial 8
Kneis—Langer—Sikdar 07.12.2010

Tutorial Exact Algorithms

This tutorial is aimed primarily at practice in dynamic programming. The problems are all polynomial-time solvable.

Exercise T15 Maximum Value Contiguous Subsequence. You are given a sequence $a_1, \ldots, a_n$ of n integers (positive and negative). You are required to find a *contiguous* subsequence for which the sum of the elements in the subsequence is maximized. Design a DP algorithm for this problem that does this in $O(n)$ time. Note that in the absence of negative integers, the answer is trivially the whole sequence.

Solution
For $1 \leq i \leq n$, define OPT$[i]$ to be the maximum sum over all contiguous subsequences that end at a_i. Then OPT$[1] = a_1$ and for $i \geq 2$, the optimal sequence either begins at some $j < i$ and ends at i or it begins at i itself. Hence

$$\text{OPT}[i] = \max\{\text{OPT}[i-1] + a_i, a_i\}.$$

One can evaluate OPT$[n]$ in linear time.

Exercise T16 Box Stacking. You are given n types of boxes, where the ith box has height h_i and a base of length l_i and width w_i. Assume without loss of generality that $w_i \leq l_i$. You want to create a stack of boxes that is as tall as possible, but you can stack box i above box j iff $l_j > l_i$ and $w_j > w_i$. Design a DP algorithm for this problem.

Solution First arrange the boxes in order of decreasing base area. Then no box in the sequence can be below any box preceding it. For $1 \leq j \leq n$, define OPT$[j]$ to be the height of the tallest stack of boxes with box j is at the top. Then

$$\text{OPT}[j] = \max_{\substack{i<j \\ w_i > w_j; l_i > l_j}} \{\text{OPT}[i] + h_j\}.$$

The time taken to compute OPT$[i]$ is $O(i)$ and hence the total time taken is $O(n^2)$. The optimal solution is given by

$$\text{OPT} = \max_i \{\text{OPT}[i]\},$$

as any box in the sequence could have been on top.

Homework Assignment H15 (10 Points) Longest Increasing Subsequence. You are given a sequence of n integers $a_1, \ldots, a_n$ and are required to determine a subsequence (not necessarily contiguous) of maximum length such that the integers in the subsequence form a strictly increasing sequence. For example, if the input sequence is $14, 2, 1, 19, 4, 5$, then an optimal subsequence is $1, 4, 5$. Another optimal solution is $2, 4, 5$.

Solution For $1 \leq j \leq n$, define OPT[j] to be the length of the optimal subsequence that ends at j. Then OPT[1] $= 1$ and for all $j > 1$,

$$\text{OPT}[j] = \max_{\substack{i<j \\ a_i<a_j}} \{\text{OPT}[i] + 1\}.$$

The optimal subsequence that ends at j has its second-last element at some position $i < j$. Since the subsequence is strictly increasing, we also have that $a_i < a_j$. The time taken to compute OPT[i] is $O(i)$ and hence the total time taken is $O(n^2)$. The optimal solution is

$$\text{OPT} = \max_{1 \leq j \leq n} \{\text{OPT}[j]\},$$

since the optimal solution could have ended anywhere.

Homework Assignment H16 (10 Points) Building Bridges. Consider a 2-D map with a horizontal river passing through its center. There are n cities on the southern bank with x-coordinates $a_1 < a_2 < \cdots < a_n$ and n cities on the northern bank with x-coordinates $b_1, b_2, \ldots, b_n$. You want to connect city i on the northern bank with city i on the southern bank by building bridges for as many north-south pairs of cities as possible with the restriction that no two bridges must cross. Design a DP algorithm to find out the maximum number of bridges that can be constructed.

Solution This is similar to the Longest Increasing Subsequence problem. For if cities $a_{i_1} < a_{i_2} < \cdots < a_{i_p}$ on the southern bank are connected by bridges in an optimal solution then the sequence $b_{i_1}, b_{i_2}, \ldots, b_{i_p}$ is a longest increasing subsequence of $b_1, \ldots, b_n$.