Algorithms M2–IF TD 5

November 8, 2021

1 Weighted Independent Set on Paths

(Exercise 6.3 of [DPV]) We are considering opening restaurants along a highway from city A to city B. The possible locations are given to us as an array $D[1 \dots n]$, where $D[i]$ is the distance of location i from A. Each location has an expected profit $P[i]$. We have unlimited budget, however, we do not want to open two restaurants which are at distance at most k kilometers. Given this constraint, describe a polynomial-time (in n) algorithm that selects the locations that maximize the total expected profit.

2 Max Sum Sub-interval

We are given an array $A[1 \dots n]$ of positive and negative integers. We want to calculate two integers a, b such that $\sum_{i=a}^{b} A[i]$ is maximized.

- Observe that this problem is trivial to solve in $O(n^3)$.
- Give an algorithm that solves this problem in time $O(n)$.

3 Longest Common Subsequence

We are given two strings over the alphabet $\{A, B, C, \dots, Z\}$. For example, the strings $s_1 =$`ILOVEALGORITHMS` and $s_2 =$`VACANCESDENOEL`. A subsequence of a string s is a string we can obtain by deleting some of the letters of s. For example, `LOVE` is a subsequence of s_1, `LOLO` is also a subsequence of s_1 (it doesn't matter that its letters are not consecutive is s_1), but `AGORA` is not a subsequence of s_1.

1. Given two strings s_1, s_2, give a polynomial-time algorithm which decides if s_2 is a subsequence of s_1. Prove the correctness of your algorithm.
2. Given two strings s_1, s_2, give a polynomial-time algorithm which calculates the length of the longest string s' which is a subsequence of both s_1, s_2.

4 The Gas Station Problem

We want to drive from city A to city B along a highway of length k kilometers. Our car has a tank with a capacity of L liters and we start out with a full tank from city A. To simplify things, suppose that our car uses 1 liter per kilometer of driving.

We are given two arrays $D[1 \dots n]$ and $P[1 \dots n]$. The first array contains the positions of gas stations along the way (that is, the distance of each gas station from A). The second array has the price of gas for each gas station. So, if $D[i] = d_i$ and $P[i] = p_i$, this means that the i-th gas station is at d_i kilometers from A and sells gas for p_i euros per liter.

Give a polynomial-time algorithm which selects a set of gas stations to stop at and the amount of gas to buy at each one in order to minimize the total cost of driving from A to B. You may assume that it's OK to arrive at B with an empty tank. Your algorithm should run in time polynomial in n, L.

5 Inventory Problem

You own a company that sells cars. For this exercise we will try to minimize the inventory cost of your company, that is, the cost of storing the cars in a garage.

You have a garage with space for S cars. Storing a car for a week in this garage costs C euros **per car**. When your garage is running low you can order more cars from the factory. This has a delivery cost of K, independent of how many cars you ordered.

You are given an array which estimates the expected demand for cars in the next few weeks: the array $D[1 \dots n]$ has value $D[i]$ for the number of cars you would like to have available in your garage to sell during week i.

We want to calculate an ordering and storage schedule which satisfies the following:

- We never store more than S cars in the garage.
- At the beginning of each week i, the number of cars in the garage is at least $D[i]$. In other words, if at the beginning of week i the garage contains fewer than $D[i]$ cars, we **must** order more cars (and pay K for the delivery) so that this week's demand is satisfied.
- The total storage cost is minimized.

You may assume that the garage contains $S_0 < S$ cars in the beginning. Your algorithm should run in time polynomial in n and S.