CENG 567 Design and Analysis of Algorithms Fall 2022

Homework 4

Released: Dec 14, 2022 Due: Dec 26, 2022

Instructions. You may work with other students, but you must individually write your solutions **in your own words**. If you work with other students or consult outside sources (such as Internet/book), cite your sources.

If you are asked to design an algorithm, provide:
(a) a description of the algorithm in English and, if helpful pseudocode,
(b) at least one worked example or diagram to show more precisely how your algorithm works,
(c) a proof (or indication) of the correctness of the algorithm,
(d) an analysis of the implementation of the algorithm.

Submissions. Submit a pdf file through odtuclass. LaTeX or Word typed submission is required.

1. Burning Calories
Suppose a friend of yours is planning to attend a workout program to lose weight. Each day he has to choose an exercise. The set of possible exercises is divided into those that are low-level and those that are high-level. The basic question each day is whether to choose a low or high level exercise.

If he selects a low-level exercise on day i, then he burns $l_i > 0$ calories; if he selects a high-level exercise he burns $h_i > 0$ calories. However, if he does high level activity on one day, it is required that he cannot choose any exercise on the next day. He has to rest. On the other hand, if he does a low-level exercise on one day, it is okay for him to select either type of exercise on the next day.

A plan is a list of choices of "low-level", "high-level" or "none" given a sequence of n days, with the condition that if "high-level" is chosen on a day $i > 0$ then "none" has to be chosen for day $i + 1$. Your friend can start with any choice on day 1. The value of the plan is the total calories he burns at the end of n days.

Design a dynamic programming algorithm that takes values for $l_1, l_2, ...l_n$ and $h_1, h_2, ...h_n$ and returns the plan of maximum calories. (Hint: Drawing the DAG corresponding to this problem may help to derive the recurrence.)

2. 2D Sheet Cutting
Consider an $m \times n$ sheet of metal where m, n are positive integers. You are given an array P with a real-valued price $P(i, j)$ for any integer-dimension rectangle of size $i \times j$. Assume $P(i, j) = P(j, i)$. You want to cut the sheet maximizing the total price of the resulting pieces. You can cut any piece, horizontally or vertically, with the cut going completely across.

Design an algorithm that indicates how to cut for maximum total price, producing instructions of the form "Cut a $i \times j$ piece at $x = k$ (or at $y = k$)".

3. Escape Problem (K&T Chp.7 Ex. 14)
Consider a directed graph $G = (V, E)$ and two disjoint sets of nodes $X, S \subset V$. A set of evacuation routes is a set of paths so that (i) each node in X is the start of one path, (ii) each path ends at a node in S, and (iii) the paths do not share any edges.

(a) Show how to decide in polynomial time whether a set of evacuation routes exists.
(b) Do the same when (iii) reads "the paths do not share any nodes".
(c) Give an example with the same G, X and S when (a) is possible but (b) not.

4. Enrollment Problem
There are n students in the CS program, and each of them must take the Algorithms course. There are m sections of Algorithms offered: $A_1, ..., A_m$. They begin at times $T_1, ..., T_m$. Each student can sign up for at most one section. Because students also have other courses to take, each student can only attend some subset of the Algorithms sections. This varies by student: some may only be free during one or two sections, while others might be able to attend all of them. In addition, each section A_i has a capacity C_i on the total enrollment, so you cannot put more than C_i students in that section.

Your goal is to assign students to sections so that as many students as possible are enrolled in a section of Algorithms (note that it might be impossible to enroll *everybody* into a section: this is fine).

Design an algorithm to solve this problem by using **network flows**. Note that it is not enough to only describe the graph to solve the problem. You should write an algorithm to describe how to use that graph to solve the given enrollment problem.