ALGORITHM DESIGN

Exercise Sheet 1

Lecturer: Prof. Stefano Leonardi
Teaching Assistants:
Philip Lazos, Rebecca Reiffenhäuser

Academic Year 2021/2022
November 18, 2021
Due Date: December 5, 2021

General Terms: You can work on the exercises in teams of two people, and hand them in accordingly. However, make sure that each one of you has fully understood every solution you present. Do not copy any work from other students, the internet or other sources, and do not share your work with others outside your team. If at any point, any part of the exercises you hand in is apparent to be a copy of other work, this will result in the following consequences: All of your exercises, previous as well as upcoming ones, will be treated as if you did not hand them in at all, and you will have to participate in the written exam to make up for this. Please note that there will be no exception made, even if you are the original author of work someone else copied, or if your exercise partner is the one responsible. Therefore, please make sure to only choose a partner that you trust, and do not hand out your exercise solutions to others.
Late Policy: If you hand in your exercises after the due date, each day that you are late will result in a discount to your score, i.e. you will only receive 90% of the points if you are one day late, 80% on the second day after the due date, and 70% on the third. If you are late by more than three days, the assignment will get zero points in total.
Formal Requirements: Please typeset your solutions (11 pt at least), and state the names of both team members at the beginning of each sheet you hand in. Make sure to name the exact exercise each part of the solution refers to, otherwise it will not be graded. Please start a new page for each main exercise in the assignment (i.e. exercise 1, 2, etc., but not for each subquestion in them). Make sure your solution takes no more than one page for each main exercise (plus at most one extra page for the code, if an implementation is required). Everything after the first page will not be taken into account. So, for example, when the assignment has four exercises, please hand in four pages, one for each exercise. All solutions have to be sent via email to

lazos@diag.uniroma1.it **or** rebeccar@diag.uniroma1.it

Office Hours: There will be special hours for questions about the exercises announced via the piazza site. Presumably, this will take place in form of a zoom meeting due to the Covid19-measures.

Exercise 1

Your birthday is coming up. You have n relatives R, who would each be willing to spend at most some value $v_r \in \mathbb{R}_+$ on your gift, for any $r \in R$. All of them are asking you for your wishes.

However, these people have different proximity to you in the family tree, given by the length $dist(r)$, $r \in R$ of a path from the vertex representing you, to the one representing the respective relative r. (You can assume you are the root of the tree, your closest relatives will be connected to you via an edge (length 1), the second-closest via a path of two edges (length 2), and so on.) Now, to not be untasteful, if $dist(r) < dist(r')$ you should never ask relative r' for a more expensive gift than r, i.e. the prices p_r, $r \in \{1, \dots, n\}$ of presents that you suggest should satisfy the following: if $dist(r) < dist(r')$, then also $p_r \geq p_{r'}$. To make sure relative r will actually buy the present you suggest for price p_r, of course you must also ensure that $p_r \leq v_r$ - if not, r will not buy the gift, and you collect value 0.

a) Give an algorithm (short description!) computing the maximum-possible, accumulated value of all your gifts in time polynomial in n. Prove its correctness and analyze its running time.

b) Solve the problem via a short(!) Python program, and provide the according code as well as the output number for the following instance:

- Distance 1: $v_1 = 15.2$, $v_2 = 9.4$, $v_3 = 11.1$
- Distance 2: $v_4 = 9.7$, $v_5 = 12.2$, $v_6 = 7.5$, $v_7 = 10.8$
- Distance 3: $v_8 = 5.2$, $v_9 = 6.3$, $v_{10} = 5.6$, $v_{11} = 10.2$
- Distance 4: $v_{12} = 4.9$, $v_{13} = 2$, $v_{14} = 3.5$, $v_{15} = 2.9$
- Distance 5: $v_{16} = 1$, $v_{17} = 6.4$, $v_{18} = 4.6$, $v_{19} = 2.1$

Exercise 2

You are the owner of a large chain of franchise shops, and you would like to expand to a new city. The blocks in your city make an $n \times n$ grid. However, although your products are awesome and in high demand, the city will not allow you to open a shop in each block. Instead, for every row of blocks i, you are are given a number r_i that limits the maximum number of shops opened there - and for every column j, there also is a maximum number c_j.

a) Find the maximum number of franchise shops you can legally open in the city. To do so, model the problem as a flow network. Then, describe how to get the right answer using Ford-Fulkerson, and prove the correctness of your construction.

b) To really become known, you would like to make sure that people driving through the city have a good chance at seeing at least one of your shops. Therefore, at least one shop should be placed every few streets. More exactly, for row $\{1, \ldots, 5\}$ of the grid, there should be at least one block in these rows that has a shop, and the same should hold for row $\{6, \ldots, 10\}$, et cetera. Conversely, for every five columns of the grid, you want to ensure also at least one shop. Adjust your network such that it can be used to determine whether this requirement can be fulfilled. Here, you are allowed to not also pose upper bounds on the capacity on each edge, but also state a lower bound on the minimum amount of flow you want on the edge. Give a variation of the Ford-Fulkerson algorithm that solves the problem for this richer type of network. Prove that together, your network and the algorithm solve the decision problem described above, while also maximizing the number of shops in case of a 'yes'-instance.

Exercise 3

You are tasked with creating a competitive programming squad, selecting students from schools around Italy. There are n schools in total and every school i has $g_i \geq 0$ girls and $b_i \geq 0$ boys. You need to select exactly k students from each school, but be careful: the total number of girls and boys need to be as close as possible. In particular, you need to select $\hat{b}_i$ boys and $\hat{g}_i$ girls from each school i, respecting that $0 \leq \hat{g}_i \leq g_i$, $0 \leq \hat{b}_i \leq b_i$ and $\hat{g}_i + \hat{b}_i = k$, in order to minimize:

$$\left| \sum_{i=1}^{n} \hat{g}_i - \sum_{i=1}^{n} \hat{b}_i \right|.$$

You are guaranteed that for every school i, we have $b_i + g_i \geq k$.

a) Provide an algorithm that computes the $\hat{g}_i$ and $\hat{b}_i$ in polynomial time. Prove its correctness and analyze its run time.

Exercise 4

The grinch is planning to make all the city's kids sick the day before christmas, by overfeeding them with sweets. Each kid has a set of sweets they like, and the grinch would like to give them all of those. He needs to give each kid some bags of sweets from Santa's supply that cover their needs, but also wants to minimize the number k of times he has to steal the most popular bag: if more than k of the same kind are missing, Santa might notice... More formally, from a given set S of sweets, bag types $B_1, B_2, \ldots, B_n \subseteq S$ and preferred sets of sweets $K_1, K_2, \ldots, K_m$, you need to find a set of bags $\hat{B}(j) \subseteq \{B_1, B_2, \ldots, B_n\}$ for each kid i such that:

$$K_i \subseteq \cup_{B \in \hat{B}(j)} B.$$

Moreover, for all bags B_i, you need to make sure that

$$|\left\{\hat{B}(j) | B_i \in \hat{B}(j), j \in \{1, \ldots, m\}\right\}| \leq k.$$

Planning this is taking the grinch a long time. Prove that indeed, the problem is NP-complete.
Hint: For your reduction, use a version of SAT where every clause has either all positive or all negative literals, which is an NP-Complete problem (i.e., all clauses are of the form $(x_1 \vee x_2 \vee \ldots)$, or $(\overline{x_1} \vee \overline{x_2} \vee \ldots)$).