ALGORITHM DESIGN

Exercise Sheet 2

Lecturer: Prof. Stefano Leonardi

Teaching Assistants:

Philip Lazos, Rebecca Reiffenhäuser

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, because 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.

Please use the subject line "AD 2020/2021 Exercise 2" and include a *single* pdf file containing all the solutions (as well as your names, exercise set, course etc) using the filename "lastname1_lastname2_ex2.pdf".

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

Santa is in a hurry: he has n presents still to deliver to n houses H, and $m < n$ elves E to help him with the task. He wants to take a nice nap now, so he wants his elves to do the deliveries. Unlike himself, the elves unfortunately are not skilled enough to be capable of delivering any present: one cannot climb chimneys, another is too loud and would wake people who have a light sleep, and so on. Santa knows for each $e \in E$ which houses H_e are doable, and since elves are playful creatures, he believes each delivery will take each elf one full hour. As christmas is soon to be over, Santa would like to deliver all presents, but also minimize the time until the last elf is finished.

a) Help Santa write the problem as an ILP with 0-1-variables. Then relax this to an LP.

b) Give an algorithm that gets as input an optimal solution to your LP, and outputs a feasible solution to Santa's problem which is, in expectation, just as good as the LP optimum.

c) To take away Santa's worries, show that your algorithm is unlikely to deliver very bad results: prove that with high probability, i.e. at least $1 - 1/m$, the resulting solution is no worse than $O(\ln m + OPT)$, where OPT denotes the finishing time of the optimal LP solution.
   Hint: If X is the sum of independent $\{0, 1\}$-variables and $\mathbb{E}[x] = \mu > 0$, then for any $\epsilon > 0$:

$$\mathbb{P}[X \geq (1 + \epsilon)\mu] < exp\left(-\mu \, \min\{\epsilon, \epsilon^2\}/3\right)$$

Exercise 2

Santa has a real challenge this year: 187 of his elves were quarantined with Covid 19 during the christmas time, and are now clinically depressed because they missed all of the fun. He wants to give each of them a perfect present, and since they have always envied Santa for his magical sleigh, his idea is to give each of them his own toy sleigh.

Now in the shed, he has a set P of n pieces that will clearly be sufficient to build the sleighs, each of which is available in large amounts. The pieces are such that you will never need more than copy of any $p \in P$ for the same sleigh. Santa, as an expert, knows what a good sleigh needs: For example, they might each need a front light, and

there might be different ones $(p_1, \ldots, p_l)$ in P, so you must use p_1 or $\ldots$ or p_l. Also, it is important for everything to look great, so he is planning to use at least one red part in each gift and one green one. Actually, Santa has a long list of all kinds of different constraints on the sleigh assembly, which he is writing down similarly as above.

As the new intern, Santa is calling you into his office and while you are eating a nice sugar cane, he tasks you with the following: Tomorrow, it is your job to write and run a polynomial-time algorithm that gets as input a formal depiction of Santa's constraints, and outputs all 187 different sets of parts for the sleighs (if possible, otherwise return 'no'). Santa promises the assembly will be done by more experienced workers - all you have to take care of is that no two sleighs are exactly the same, so the gifts still feel special enough.

Despite the sugar cane, the assignment leaves a bitter taste in your mouth. However, not wanting to ruin your own christmas, you don't protest right away: that would look just lazy. Come up with a mathematical proof that in general, this problem is clearly a hard one and might not be solvable for you in a day.

Exercise 3

We have an undirected graph $G = (V, E)$ with $|V|$ players, in which each player i controls exactly one, distinct vertex $v_i \in V$. Each edge $e \in E$ is associated with a *nonnegative* weight w_e. A *cut* is a partition of the set of vertices into two disjoint sets $LEFT$ and $RIGHT$. Each player selects the set in which his controlled vertex will be, i.e. the strategy space of each player i is $\alpha_i = \{LEFT, RIGHT\}$ and let $\alpha = (\alpha_1, \ldots, \alpha_n)$ be the corresponding strategy profile.

Let $CUT(\alpha)$ be the set of edges that have one endpoint in each set and let N_i be the set of neighbours of v_i in the graph G. The payoff of player i is defined as

$$u_i(\alpha) = \sum_{e \in CUT(\alpha) \cap N_1} w_e.$$

1. Design a cut game in which the Price of Anarchy is 2. *Hint:Use a graph with four vertices and unit weights.*

2. Prove that the Price of Anarchy of cut games is at most 2 in the general case, not only for the game with 4 vertices. *Hint: First argue that in a pure Nash equilibrium, the total weight of the neighbouring vertices of a vertex v_i in the cut is at least half the total weight of all the neighbouring vertices of the vertex.*

Exercise 4

Suppose that there is a graph $G = (V, E)$ where the vertices can be of two types: there is a set A of possible antennae locations and a set S of cities, with $V = A \cup S$. The graph is complete, i.e. for any pair $\{x, y\} \subseteq V$ we also have that $(x, y) \in E$. Moreover, the weight of each each edge (x, y) is given by the distance function $d : V^2 \to \mathcal{R}$; it satisfies the properties $d(x, y) = d(y, x) \geq 0$, $d(x, x) = 0$ and $d(x, z) + d(z, y) \geq d(x, y)$, for all $x, y, z \in V$. Our goal, is to select k antennae such that the maximum distance from a city to any chosen antenna is minimized. Specifically, we need to find a $U \subset A$ such that $|U| = k$ and

$$\max_{x \in S} \min_{y \in U} d(x, y)$$

is minimized.

1. Show that minimizing this objective is NP-hard.

2. Find a 3-approximation algorithm.

3. Show that finding an a-approximation with $a < 3$ is also NP-hard. If you do this, there is no need to include an answer to the first sub-question.