# Midterm Exam, Advanced Algorithms 2020-2021

- You are allowed to consult lectures notes of the course, but **no outside material**.
- **Communication is not allowed.**
- Your explanations should be clear enough and in sufficient detail that a fellow student can understand them. In particular, do not only give pseudo-code without explanations. A good guideline is that a description of an algorithm should be such that a fellow student can easily implement the algorithm following the description.
- You are allowed to refer to material covered in the lecture notes including theorems without reproving them.

**Good luck!**

**Name:** ______________________ **N° Sciper:** ______________

| Problem 1 / 22 points | Problem 2 / 30 points | Problem 3 / 20 points | Problem 4 / 28 points |
|---|---|---|---|
| | | | |

| **Total / 100** |
|---|
| |

**1** *(22 pts)* **LP duality.** Write down the duals of the linear programs below together with complimentary slackness conditions.

**1a**

$$\begin{aligned} \text{minimize} \quad & 3x_1 + 4x_2 \\ \text{s. t.} \quad & x_1 + 2x_2 \geq 1 \\ & 2x_1 + 4x_2 \geq 5 \\ & x_1, x_2 \geq 0. \end{aligned}$$

**1b**

$$\begin{aligned} \text{maximize} \quad & 3x_1 + 4x_2 + x_3 \\ \text{s. t.} \quad & x_1 + 2x_2 - x_3 = 0 \\ & x_1 + 2x_3 \leq 3 \\ & 5x_1 + x_2 \leq 2 \\ & x_1, x_2, x_3 \geq 0. \end{aligned}$$

**2** *(30 pts)* **Optimal vertex removal.** Suppose that you are given a directed graph $G = (V, E)$ together with an assignment of costs to vertices $c : V \to \mathbb{R}_+$ and a set of vertices $S \subseteq V$ that form an independent set (i.e. none of the vertices in $S$ are neighbors in $G$). We say that a subset $R \subseteq V \setminus S$ disconnects $S$ if no vertex in $S$ can reach another vertex in $S$ when vertices in $R$ are removed together with all their edges. Your task is to find the cheapest set of vertices $R \subseteq V \setminus S$ that disconnects $S$, where the cost of a set $R \subseteq V$ is defined as $\sum_{v \in R} c_v$.

**2a** *(8 pts)* Let $\mathcal{P}$ denote the set of simple[1] paths in $G$ connecting pairs of vertices in $S$ (here a path $P = (u_1, u_2, \ldots, u_k)$ is a sequence of vertices of $G$ such that for every $i = 1, \ldots, k-1$ one has $(u_i, u_{i+1}) \in E$; a path is simple if it contains no repeated **vertices**). Consider the linear program below:

$$\begin{aligned} \text{minimize} \quad & \sum_{v \in V \setminus S} c_v d_v \\ \text{s. t.} \quad & \sum_{v \in P, v \in V \setminus S} d_v \geq 1 \qquad \text{for every } P \in \mathcal{P} \\ & d_v \geq 0 \qquad\qquad\qquad \text{for all } v \in V \setminus S. \end{aligned}$$

Prove that for every $R \subseteq V \setminus S$ that disconnects $S$ there exists a feasible solution $(d_v)_{v \in V \setminus S}$ to the LP above with cost bounded by the cost of $R$.

**2b** *(11 pts)* Write down the dual of the LP in **(a)** together with complimentary slackness conditions.

**2c** *(11 pts, half ⋆)* Show how, given a candidate solution $d = (d_v)_{v \in V \setminus S}$, one can in polynomial time check whether $d$ is feasible for the LP above, and find a violated constraint if $d$ is not feasible.

[1] We call a path simple if it contains no repeated **vertices**.

**Solution to problem 2c**

**3** *(20 pts)* **Collaborative basis.** In the collaborative basis problem $d \geq 2$ participants are given $d \times n$ matrices $A_1, \ldots, A_d$ for some $n \geq 1$. Their task is to select one column from each of the matrices $A_i, i = 1, \ldots, d$, so that the selected columns form a basis for the entire space $\mathbb{R}^d$. Give an efficient algorithm that, given matrices $A_1, \ldots, A_d$, outputs **YES** if such a collection of columns exists and **NO** otherwise.

Example 1. Suppose that $d = 2, n = 4$, and matrices $A_1, A_2$ are given by

$$A_1 = \begin{pmatrix} 5 & 2 & 1 & 2 \\ 5 & 1 & 5 & 0 \end{pmatrix} \quad \text{and} \quad A_2 = \begin{pmatrix} 1 & 0 & 2 & 3 \\ 1 & 0 & 2 & 3 \end{pmatrix}.$$

Then taking the second column of $A_1$ and the first column of $A_2$, we obtain a basis for $\mathbb{R}^2$. Indeed, the matrix

$$\begin{pmatrix} 2 & 1 \\ 1 & 1 \end{pmatrix}$$

is full rank, so the answer is **YES**.
Example 2. Suppose that $d = 3, n = 2$, and matrices are given by

$$A_1 = \begin{pmatrix} 1 & 2 \\ -2 & 2 \\ 1 & 4 \end{pmatrix}, A_2 = \begin{pmatrix} -1 & 3 \\ 2 & 0 \\ -1 & 5 \end{pmatrix} \text{ and } A_3 = \begin{pmatrix} -2 & 6 \\ 4 & 0 \\ -2 & 10 \end{pmatrix}$$

Here one notes that columns of $A_3$ are just a scaled version of columns of $A_2$, and columns of $A_2$ can be obtained by taking a linear combination of columns of $A_1$, so it is not possible to choose one column from each matrix to obtain a basis for $\mathbb{R}^3$.

*Hint: use matroids. You may use the fact that, given a collection of $k$ vectors in $\mathbb{R}^d$, one can check if the vectors are linearly independent (i.e., if the matrix whose columns are these vectors has rank $k$) in time polynomial in $k$ and $d$.*

**Solution to problem 3**

**4** *(28 pts)* **Matrix reconstruction.** A well known advertisement agency decided to hire you to find the right advertisement placement strategy for their clients. Suppose that there are $n$ types of billboards in Lausanne and $m$ clients, and you used a linear program to determine how many billboards of each type every client should use. Specifically, your LP solver produced an $n \times m$ matrix $A$, where for $i = 1, \ldots, n$ and $j = 1, \ldots, m$ the $(i, j)$'th entry of the matrix shows how many billboards of type $i$ the $j$'th client should use. There is a problem though: the entries in the matrix are not integers. At the very least they are non-negative, however, and every row sum as well as every column sum is an integer. You will design an efficient algorithm to round the matrix entries to integers with the following constraints:

- Any non-integer element $x$ in the matrix can only be replaced by $\lfloor x \rfloor$ or $\lceil x \rceil$.
- In the output matrix, for any row (or column), the sum of entries in that row (or column) should remain the same as in the initial matrix.

*Note: there might be many correct output matrices for a given initial matrix, and you only need to output one of them. You should design the algorithm, prove its correctness and establish runtime bounds.*

*(Hint: use ideas developed in class for a problem on graphs)*

Example 1: The matrix

$$\begin{pmatrix} 1 & 2.2 & 1 & 5.8 \\ 3 & 0 & 1 & 2 \\ 2 & 1.8 & 3 & 0.2 \end{pmatrix}$$

can be rounded to the matrix

$$\begin{pmatrix} 1 & 2 & 1 & 6 \\ 3 & 0 & 1 & 2 \\ 2 & 2 & 3 & 0 \end{pmatrix}.$$

Example 2: The matrix

$$\begin{pmatrix} 0.3 & 0.3 & 0.3 & 0.1 \\ 0.2 & 0.5 & 0.1 & 0.2 \\ 0.5 & 0.2 & 0.6 & 0.7 \end{pmatrix}$$

can be rounded to the matrix

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix}.$$

**Solution to problem 4**