ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

Midterm Exam, Advanced Algorithms 2017-2018

- You are only allowed to have a handwritten A4 page written on both sides.
- Communication, calculators, cell phones, computers, etc... are 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. You may also refer to statements given in the exercise sheets and the homework sheet. You are however *not* allowed to refer to material from any specific solution to these exercises.
- **Do not touch until the start of the exam.**

Good luck!

Name: _____ **N° Sciper:** _____

Problem 1 / 20 points	Problem 2 / 20 points	Problem 3 / 20 points	Problem 4 / 20 points	Problem 5 / 20 points

Total / 100

1 *(consisting of subproblems **a**-**b**, 20 pts)* **Basic questions.** This problem consists of two subproblems each worth 10 points.

1a *(10 pts)* Suppose we use the Simplex method to solve the following linear program:

$$\begin{aligned} \textbf{maximize} \quad & 4x_1 - x_2 - 2x_3 \\ \textbf{subject to} \quad & x_1 - x_3 + s_1 = 1 \\ & x_1 + s_2 = 4 \\ & -3x_2 + 2x_3 + s_3 = 4 \\ & x_1,\, x_2,\, x_3,\, s_1,\, s_2,\, s_3 \geq 0 \end{aligned}$$

At the current step, we have the following Simplex tableau:

$$\begin{aligned} x_1 &= 1 + x_3 - s_1 \\ s_2 &= 3 - x_3 + s_1 \\ s_3 &= 4 + 3x_2 - 2x_3 \\ \hline z &= 4 - x_2 + 2x_3 - 4s_1 \end{aligned}$$

Write the tableau obtained by executing one iteration (pivot) of the Simplex method starting from the above tableau.

Solution:

1b *(10 pts)* Chef Baker Buttersweet just took over his family business - baking tasty cakes! He notices that he has m different ingredients in various quantities. In particular, he has $b_i \geq 0$ kilograms of ingredient i for $i = 1, \ldots, m$. His family cookbook has recipes for n types of mouthwatering cakes. A kilogram of cake of type j is worth c_j CHF. For each recipe j, the cookbook says how many kilograms of each of the ingredients are needed to make one kilogram of cake of type j. One kilogram of cake of type j, for $j = 1, \ldots, m$, needs precisely a_{ij} kilograms of ingredient i for all $i = 1, \ldots, m$.

Chef wants to make $x_j \leq 1$ kilograms of cake of type j. Having studied linear programming, he knows that the maximum revenue he can get is given by the following linear program, where $A \in \mathbb{R}_+^{m \times n}$, $b \in \mathbb{R}_+^m$ and $c \in \mathbb{R}_+^n$.

$$\begin{aligned} \textbf{Maximize} \quad & \sum_{j=1}^{n} c_j x_j \\ \textbf{subject to} \quad & Ax \leq b \\ & 1 \geq x_j \geq 0 \quad \forall j. \end{aligned}$$

Chef realizes that he can use Hedge algorithm to solve this linear program (approximately) but he is struggling with how to set the costs $m_i^{(t)}$ at each iteration. Explain how to set these costs properly.

*(In this problem you are asked to define the costs $m_i^{(t)}$. You do **not** need to explain how to solve the reduced linear program that has a single constraint. Recall that you are allowed to refer to material covered in the lecture notes.)*

Solution:

(This page is intentionally left blank.)

(Primal) LP Relaxation

$$\textbf{minimize} \quad \sum_{S \in \mathcal{T}} c(S) x_S$$

$$\textbf{subject to} \quad \sum_{S \in \mathcal{T}: e \in S} x_S \geq 1 \quad \text{for } e \in \mathcal{U}$$

$$x_S \geq 0 \quad \text{for } S \in \mathcal{T}$$

(Dual)

$$\textbf{maximize} \quad \sum_{e \in \mathcal{U}} y_e$$

$$\textbf{subject to} \quad \sum_{e \in S} y_e \leq c(S) \quad \text{for } S \in \mathcal{T}$$

$$y_e \geq 0 \quad \text{for } e \in \mathcal{U}$$

Figure 1. The standard LP relaxation of set cover and its dual.

2 *(20 pts)* Recall that a set cover instance is specified by a universe $\mathcal{U} = \{e_1, \ldots, e_n\}$, a family of subsets $\mathcal{T}$, and a cost function $c : \mathcal{T} \to \mathbb{R}_+$. The task is to find a collection $C \subseteq \mathcal{T}$ of subsets of minimum total cost that covers all elements. The natural LP relaxation and its dual (as seen in class) are given in Figure 1.

In the homework, we analyzed a primal-dual algorithm for vertex cover. In this problem you should design and analyze a primal-dual algorithm for set cover that has an approximation guarantee of f, where f is the maximum number of sets any element belongs to: $f = \max_{e \in \mathcal{U}} |\{S \in \mathcal{T} : e \in S\}|$. In other words, you should prove that your primal-dual algorithm returns a set cover of weight at most f times the weight of an optimal solution.

(In this problem you are asked to (i) design the primal-dual algorithm and (ii) show that it has an approximation guarantee of f. We remark that an answer that solves and rounds the LP relaxation is rewarded 0 points. Recall that you are allowed to refer to material covered in the lecture notes.)

Solution:

(This page is intentionally left blank.)

3 *(20 pts)* **LP-based algorithm for packing knapsacks.** Homer, Marge, and Lisa Simpson have decided to go for a hike in the beautiful Swiss Alps. Homer has greatly surpassed Marge's expectations and carefully prepared to bring n items whose total size equals the capacity of his and his wife Marge's two knapsacks. Lisa does not carry a knapsack due to her young age.

More formally, Homer and Marge each have a knapsack of capacity C, there are n items where item $i = 1, 2, \ldots, n$ has size $s_i > 0$, and we have $\sum_{i=1}^{n} s_i = 2 \cdot C$ due to Homer's meticulous preparation. However, being Homer after all, Homer has missed one thing: although the items fit perfectly in the two knapsacks fractionally, it might be impossible to pack them because items must be assigned integrally!

Luckily Lisa has studied linear programming and she saves the family holiday by proposing the following solution:

- Take *any* extreme point x^* of the linear program:

$$\begin{aligned} x_{iH} + x_{iM} &\leq 1 && \text{for all items } i = 1, 2, \ldots, n \\ \sum_{i=1}^{n} s_i x_{iH} &= C \\ \sum_{i=1}^{n} s_i x_{iM} &= C \\ 0 \leq x_{ij} &\leq 1 && \text{for all items } i = 1, 2, \ldots, n \text{ and } j \in \{H, M\}. \end{aligned}$$

- Divide the items as follows:
  - Homer and Marge will carry the items $\{i : x^*_{iH} = 1\}$ and $\{i : x^*_{iM} = 1\}$, respectively.
  - Lisa will carry any remaining items.

Prove that Lisa needs to carry at most one item.

(In this problem you are asked to give a formal proof of the statement that Lisa needs to carry at most one item. You are not allowed to change Lisa's solution for dividing the items among the family members. Recall that you are allowed to refer to material covered in the lecture notes.)

Solution:

(This page is intentionally left blank.)

4 *(20 pts)* **Balancing degrees.** A beautiful result by the Swiss mathematician Leonhard Euler (1707 - 1783) can be stated as follows:

> Let $G = (V, E)$ be an undirected graph. If every vertex has an even degree, then we can orient the edges in E to obtain a directed graph where the in-degree of each vertex equals its out-degree.

In this problem, we address the problem of correcting an imperfect orientation A to a perfect one A' by flipping the orientation of the fewest possible edges. The formal problem statement is as follows:

Input: An undirected graph $G = (V, E)$ where every vertex has an even degree and an orientation A of E. That is, for every $\{u, v\} \in E$, A either contains the directed edge (u, v) that is oriented towards v or the directed edge (v, u) that is oriented towards u.

Output: An orientation A' of E such that $|A' \setminus A|$ is minimized and

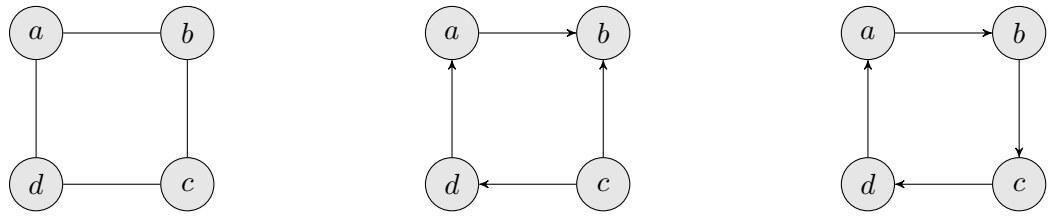
$$\underbrace{|\{u \in V : (u, v) \in A'\}|}_{\text{in-degree}} = \underbrace{|\{u \in V : (v, u) \in A'\}|}_{\text{out-degree}} \qquad \text{for every } v \in V.$$

Design and analyze a polynomial-time algorithm for the above problem.

(In this problem you are asked to (i) design the algorithm, (ii) analyze its running time, and (iii) show that it returns a correct solution. Recall that you are allowed to refer to material covered in the lecture notes.)

An example is as follows:

G $\qquad$ $A = \{(a, b), (c, b), (c, d), (d, a)\}$ $\qquad$ $A' = \{(a, b), (b, c), (c, d), (d, a)\}$



The solution A' has value $|A' \setminus A| = 1$ (the number of edges for which the orientation was flipped).

Solution:

(This page is intentionally left blank.)

5 *(20 pts)* **Comparing algorithms with little communication.** Two excellent students, Alice from EPFL and Bob from MIT, have both built their own spam filters. A spam filter is an algorithm that takes as input an email and outputs 1 if the email is spam and 0 otherwise. Alice and Bob now want to compare their two spam filters.

To perform the comparison, they both download the same huge data set consisting of n emails out of which some are spam. Alice then runs her spam filter on the data set to obtain $a_1, a_2, \ldots, a_n$ where $a_i \in \{0, 1\}$ is the output of her spam filter on the i:th email in the data set. Similarly, Bob runs his spam filter on the data set to obtain $b_1, b_2, \ldots, b_n$ where $b_i \in \{0, 1\}$ is the output of his spam filter on the i:th email in the data set. Their goal is then to determine whether their outputs are the same.

An issue that they face is that $a_1, a_2, \ldots, a_n$ are stored on Alice's computer and $b_1, b_2, \ldots, b_n$ are stored on Bob's computer. They thus need to transfer (or communicate) information to solve the problem. A trivial solution is for Alice to transfer all her outputs $a_1, a_2, \ldots, a_n$ to Bob who then performs the comparison. However, this requires Alice to send n bits of information to Bob; an operation that is very costly for a huge data set. In the following, we use randomization to achieve a huge improvement on the number of bits transfered between Alice and Bob.

Specifically, motivated by something called pseudo-random generators, we assume that Alice and Bob have access to the same randomness (called shared randomness). That is, Alice and Bob have access to the same infinite stream of random bits $r_1, r_2, \ldots$.

Your task is now to use this shared randomness to devise a randomized protocol of the following type:

- As a function of $a_1, a_2, \ldots, a_n$ and the random bits $r_1, r_2, \ldots$, Alice computes a message m that consists of only 2 bits. She then transmits this 2-bit message m to Bob.
- Bob then, as a function of $b_1, b_2, \ldots, b_n$, the message m, and the random bits $r_1, r_2, \ldots$, outputs EQUAL or NOT EQUAL.

Bob's output is correct if he outputs EQUAL when $a_i = b_i$ for all $i \in \{1, \ldots, n\}$ and NOT EQUAL otherwise. Your protocol should ensure that Bob outputs the correct answer with probability at least $2/3$, where the probability is over the random bits $r_1, r_2, \ldots$.

(In this problem you are asked to (i) explain how Alice computes the message m of 2 bits (ii) explain how Bob calculates his output, and (iii) prove that Bob's output is correct with probability at least $2/3$. A correct solution where Alice sends a message m of $O(\log n)$ bits is rewarded 12 points. Recall that you are allowed to refer to material covered in the lecture notes.)

An interesting fact (but unrelated to the exam) is that any correct deterministic strategy would require Alice and Bob to send n bits of information.

Solution:

(This page is intentionally left blank.)