Midterm Exam, Algorithms II 2023-2024

Do not turn the page before the start of the exam. This document is double-sided, has 6 pages, the last ones possibly blank. Do not unstaple.

- The exam consists of three parts. The first part consists of multiple-choice questions, the second part consists of a short open question, and the last part consists of three open-ended questions.
- For the open-ended questions, your explanations should be clear enough and in sufficient detail that a fellow student can understand them. In particular, do not only give pseudocode 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 lectures including algorithms and theorems (without reproving them). You are however *not* allowed to simply refer to material covered in exercises/homework.

Good luck!

Problem 1: Multiple Choice Questions (24 points)

For each question, select the correct alternative. Note that each question has **exactly one** correct answer. Wrong answers are not penalized with negative points.

1a. Matroids *(8 points).* Consider the ground set $E = \{a, b, c, d\}$. Select a collection $\mathcal{I}$ of independent sets from below such that $(E, \mathcal{I})$ is a matroid.

A. $\{\{\}, \{a\}, \{b\}, \{c\}\ \{a, b\}, \{a, c\}, \{a, d\}\}$

B. $\{\{\}, \{a\}, \{b\}, \{c\}, \{a, b\}\}$

C. $\{\{\}, \{a\}, \{b\}, \{c\}\}$

D. $\{\{\}, \{a\}, \{b\}, \{c\}, \{d\}, \{a, b\}, \{a, c\}, \{b, c\}, \{b, d\}, \{a, b, c\}, \{b, c, d\}\}$

E. $\{\{\}, \{a\}, \{b\}, \{c\}, \{a, b, c\}\}$

Solution. The correct answer is option C.
The set in option C satisfies the conditions of a matroid. You don't need to check the other options since there is always **one correct answer**, but let's see why they fail:

- Option A can not be a matroid because it's not downwards closed: $\{a, d\} \in \mathcal{I}$ and $\{d\} \subset \{a, d\}$ but $\{d\} \notin \mathcal{I}$
- Option B does not satisfy the extension property: $\{c\} \in \mathcal{I}$, $\{a, b\} \in \mathcal{I}$ and $|\{c\}| < |\{a, b\}|$, but you can not add any element from $\{a, b\} \setminus \{c\}$ to $\{c\}$ and still get an independent set.
- Option D is not downwards closed: $\{b, c, d\} \in \mathcal{I}$ and $\{c, d\} \subset \{b, c, d\}$ but $\{c, d\} \notin \mathcal{I}$
- Option E also is not downwards closed: $\{a, b, c\} \in \mathcal{I}$ and $\{a, b\} \subset \{a, b, c\}$ but $\{a, b\} \notin \mathcal{I}$

1b. Vertex-Cover relaxation *(8 points).* Consider the minimum vertex cover problem, and consider the linear programming (LP) relaxation for vertex cover you saw in class. For a graph G, denote by $\mathrm{OPT}(G)$ the cost of an optimum vertex cover for G, and by $\mathrm{OPT}_{\mathrm{LP}}(G)$ the cost of an optimal solution of the LP relaxation. Which one of the following statements is true?

A. There is a graph G so that $\mathrm{OPT}(G) \geqslant 4 \cdot \mathrm{OPT}_{\mathrm{LP}}(G)$.

B. For all n-vertex graphs G, we have $\mathrm{OPT}_{LP}(G) \geqslant n/2$.

C. If an n-vertex graph G has $\mathrm{OPT}(G) = 3n/4$, then $\mathrm{OPT}_{\mathrm{LP}}(G) \geqslant 3n/4$.

D. For all graphs G, it holds that $\mathrm{OPT}(G) \geqslant \mathrm{OPT}_{\mathrm{LP}}(G)$.

E. There exists a graph such that $\mathrm{OPT}_{\mathrm{LP}}(G) > 2 \cdot \mathrm{OPT}(G)$.

Solution. The correct answer is option D. The best integral optimal value is always bigger or equal to the best fractional optimal value (in a minimization problem). Let's also see why the rest of the choices are incorrect

- Option A is incorrect because since $\mathrm{OPT}(G) \leqslant 2 \cdot \mathrm{OPT}_{\mathrm{LP}}(G)$. This holds because we can always round a fractional solution of the Vertex Cover LP and lose at most a factor of 2.

- Option B is also incorrect. To see this, let G be an n-vertex star. Then taking the central node is a valid vertex cover. Therefore $\mathrm{OPT}_{\mathrm{LP}}(G) = 1$.
- Option C is also incorrect. Let G be a 4-vertex clique. Then $\mathrm{OPT}(G) = 3n/4 = 3$, but $\mathrm{OPT}_{\mathrm{LP}}(G) = 2$ by putting $1/2$ on every vertex.
- Option E is incorrect because we know that $\mathrm{OPT}_{\mathrm{LP}}(G) \leqslant 1/2\mathrm{OPT}(G)$

1c. Weighted Majority *(8 points).* We apply the weighted majority algorithm to aggregate the (binary) answers of 17 experts. At every step, we *divide by* 2 the weights of those experts that provided a wrong answer. Assume that c experts always provide the correct answer. What is the smallest value of c for which the total number of mistakes we make is *at most* 1, independently of the answers given by the other $17 - c$ experts?

A. 3

B. 4

C. 5

D. 6

E. 7

Solution. The correct answer is option E, $c := 7$.
Initially, the total weight is 17 and the "good" experts have total weight c. We claim $c = 7$ such experts suffice for us to make at most one mistake. As soon as one mistake is made, from the lecture we have that the total new weight can be *at most* $\frac{3 \cdot 17}{4} = 12.75$. The weight of the "good" experts is still c as they made no mistake. Notice that:

$$c = 7 > \frac{1}{2} \cdot 12.75,$$

so making a new mistake is impossible.

If we make c smaller, e.g., $c = 6$, we might already run into trouble. Assume in the first step 9 experts are wrong (so we make a mistake). Then, after this step, the total weight is $8 + 4.5 = 12.5$. For the second step, assume all $17 - c = 11$ experts excepting the good ones are wrong. Their total weight contribution is $12.5 - 6 = 6.5 > 6$, so we will follow their advice and be wrong again.

Problem 2: Short Open Question (10 points)

Write the dual linear program of the following linear program. No explanation is needed for your answer.

$$\begin{aligned} \min \quad & 3x_1 + 5x_2 + x_3 \\ \text{s.t.} \quad & x_1 + x_2 + x_3 \geqslant 1 \\ & x_2 - 2x_3 \leqslant 3 \\ & x_1, x_2, x_3 \geqslant 0 \end{aligned}$$

Solution.

$$\begin{aligned} \max \quad & y_1 - 3y_2 \\ \text{s.t.} \quad & y_1 \leqslant 3 \\ & y_1 - y_2 \leqslant 5 \\ & y_1 + 2y_2 \leqslant 1 \\ & y_1 \geqslant 0 \\ & y_2 \geqslant 0 \end{aligned}$$

Problem 3: Extreme Point Structure (22 points)

Given a graph $G = (V, E)$ with edge-weights $w : E \to \mathbb{R}$, consider the matching problem where we wish to select a matching of maximum weight consisting of exactly k edges. We can adapt the linear program seen in class to obtain the following relaxation:

$$\begin{aligned} \text{maximize} \quad & \sum_{e \in E} x_e \cdot w(e) \\ \text{subject to} \quad & \sum_{e \in \delta(v)} x_e \leqslant 1 \qquad && \forall v \in V \\ & \sum_{e \in E} x_e = k \\ & x_e \geqslant 0 && \forall e \in E \end{aligned}$$

where $\delta(v)$ denotes the set of edges incident to vertex v.

Let x^* be an extreme point of the above linear program. Consider the graph G' which is the subgraph of G that contains only the edges with $x^*_e > 0$. **Prove that G' contains no cycles of even length.** A cycle has an even length if it has an even number of edges.

Solution. Let x^* be an extreme point and let $G' = \{e \in E : x^*_e > 0\}$. Suppose for contradiction that G' contains an even cycle $e_1, e_2, \ldots, e_{2l}$. We will show that there exists feasible solutions $y \neq z$ such that $x = \frac{1}{2}y + \frac{1}{2}z$.

Let $\epsilon = \min\{x^*_{e_1}, x^*_{e_2}, \ldots, x^*_{e_{2l}}\} > 0$. Define y and z by

$$y_e = \begin{cases} x^*_e + \epsilon, \text{ if } e \in \{e_1, e_3, \ldots, e_{2l-1}\} \\ x^*_e - \epsilon, \text{ if } e \in \{e_2, e_4, \ldots, e_{2l}\} \\ x^*_e, \text{ otherwise} \end{cases}$$

and

$$z_e = \begin{cases} x^*_e - \epsilon, \text{ if } e \in \{e_1, e_3, \ldots, e_{2l-1}\} \\ x^*_e + \epsilon, \text{ if } e \in \{e_2, e_4, \ldots, e_{2l}\} \\ x^*_e, \text{ otherwise.} \end{cases}$$

Notice that y and z are feasible. Indeed,

$$\sum_{e \in \delta(v)} y_e = \sum_{e \in \delta(v)} z_e = \sum_{e \in \delta(v)} x^*_e \leqslant 1 \qquad \forall v \in V$$

(if v is on the cycle then we add and substract ϵ on its neighbourhood, and if v is not on the cycle then we don't change anything), and

$$\sum_{e \in E} y_e = \sum_{e \in E} z_e = \sum_{e \in E} x^*_e + \epsilon l - \epsilon l = k.$$

Finally, $z_e, y_e \geqslant 0$ for all $e \in E$ by definition of ϵ.

We now have $x^* = \frac{1}{2}(y + z)$, and clearly $y \neq z$ (since $\epsilon > 0$). So x^* is a convex combination of feasible solutions, contradicting the assumption that x^* is an extreme point.

Problem 4: Prize-Collecting Vertex Cover and Duality (22 points)

The prize-collecting vertex cover problem is a generalization of vertex cover in which we are not obligated to cover all edges, but must pay a penalty for those left uncovered. A formal definition is as follows:

Input: An undirected graph $G = (V, E)$ with a penalty $p_e \geqslant 0$ for every edge $e \in E$.

Output: A subset $C \subseteq V$ of vertices so as to minimize $|C| + \sum_{e \in E : e \cap C = \emptyset} p_e$.

To formulate a linear programming relaxation, we associate a variable x_v for every vertex $v \in V$, and a variable z_e for every edge $e \in E$. The intended meaning of these variables is that x_v indicates whether $v \in C$ and z_e indicates whether e pays a penalty, i.e., is not covered by C. We then arrive at the following linear programming (LP) relaxation and its dual:

(Primal) LP Relaxation

$$\begin{array}{lll}
\textbf{minimize} & \displaystyle\sum_{v \in V} x_v + \sum_{e \in E} p_e \cdot z_e & \\
\textbf{subject to} & x_u + x_v + z_e \geqslant 1 & \text{for } e = \{u, v\} \in E \\
& x_v \geqslant 0 & \text{for } v \in V \\
& z_e \geqslant 0 & \text{for } e \in E
\end{array}$$

(Dual)

$$\begin{array}{lll}
\textbf{maximize} & \displaystyle\sum_{e \in E} y_e & \\
\textbf{subject to} & \displaystyle\sum_{e \in \delta(v)} y_e \leqslant 1 & \text{for } v \in V \\
& y_e \leqslant p_e & \text{for } e \in E \\
& y_e \geqslant 0 & \text{for } e \in E
\end{array}$$

Recall that $\delta(v)$ denotes the set of edges incident to vertex $v \in V$.

We will analyze a simple and very fast primal-dual algorithm for the prize-collecting vertex cover problem. The algorithm maintains a feasible dual solution y initially set to $y_e = 0$ for every $e \in E$. It then iteratively improves the dual solution until every edge $e \in E$ not covered by the set $C = \{v \in V : \sum_{e \in \delta(v)} y_e = 1\}$ corresponds to a tight constraint $y_e = p_e$. Note that C consists of those vertices whose constraints in the dual are tight and the algorithm only stops when the edges not covered by C correspond to tight dual constraints. The formal description of the algorithm is as follows:

1) Initialize the dual solution y to be $y_e = 0$ for every $e \in E$.
2) While there is an edge e with $y_e < p_e$ and that is *not* covered by $C = \{v \in V : \sum_{e \in \delta(v)} y_e = 1\}$, i.e., $e \cap C = \emptyset$:
   - Increase y_e until one of the dual constraints (corresponding to u, v or e) becomes tight.
3) Return $C = \{v \in V : \sum_{e \in \delta(v)} y_e = 1\}$.

Prove that the primal-dual algorithm has an approximation guarantee of 2. That is, show that the returned set C has value

$$|C| + \sum_{e \in E | e \cap C = \emptyset} p_e$$

at most twice the value of an optimal solution. **Partial credits will be given to solutions that bound the approximation guarantee by 3.**

Solution. We show that

$$|C| + \sum_{e \in E: e \cap C = \emptyset} p_e \leqslant 2\ \mathrm{OPT}_{\mathrm{LP}},$$

where $\mathrm{OPT}_{\mathrm{LP}}$ is the value optimal linear programming solution. Note that since the linear program is a relaxation this would imply that

$$|C| + \sum_{e \in E: e \cap C = \emptyset} p_e \leqslant 2\ \mathrm{OPT},$$

We have the following equations.

$$\begin{align}
&|C| + \sum_{e \in E: e \cap C = \emptyset} p_e \tag{1}\\
&= \sum_{v \in V: \sum_{e \in \delta(v)} y_e = 1} 1 + \sum_{e: e \cap C = \emptyset} p_e \tag{2}\\
&= \sum_{v \in V: \sum_{e \in \delta(v)} y_e = 1} \left(\sum_{e \in \delta(v)} y_e \right) + \sum_{e: e \cap C = \emptyset} y_e \tag{3}\\
&\leqslant \sum_{e: e \cap C \neq \emptyset} 2 y_e + \sum_{e: e \cap C = \emptyset} y_e \tag{4}\\
&\leqslant 2\ \mathrm{OPT}_{\mathrm{LP}} \tag{5}
\end{align}$$

The first term in line (**??**) follows by our choice of C. The second term in equation (**??**) follows because whenever $e \cap C = \emptyset$ we have $y_e = p_e$. The first term in line (**??**) follows because y_e appears $|y_e \cap C|$ many times in the first term in line (**??**). Finally line (**??**) follows by weak duality.

Problem 5: Edge-Disjoint Spanning Trees (22 points)

Given a graph $G = (V, E)$, **design and analyze a polynomial-time algorithm** that does the following: Construct three spanning trees of G that share no edges, or report that this task is impossible (i.e., that G does not have three edge-disjoint spanning trees).

Solution. In this problem given a graph $G = (V, E)$ we are asked to find 3 edge-disjoint spanning trees if they exist or report that they do not, in polynomial time. We will solve this problem via matroid intersection.

We will first create 3 copies of the graph G, $G_1 = (V_1, E_1), G_2 = (V_2, E_2), G_3 = (V_3, E_3)$, and consider the graph $G' = (V_1 \cup V_2 \cup V_3, E_1 \cup E_2 \cup E_3)$. Thus G' is a forest with 3 edge and vertex disjoint copies of G. Our first matroid M_1 will be the graphic matroid on G'. Our second matroid $M_2 = (E', \mathcal{I})$ will be the following partition matroid

$$\mathcal{I} = \{X \subseteq E' : |X \cap \{e_1^i, e_2^i, e_3^i\}| \leqslant 1 \quad \forall e^i \in E\},$$

where for every edge $e^i \in E$, $e_j^i \in E_j$ denotes the j^{th} copy of e^i for $j \in \{1, 2, 3\}$. Now we consider finding the maximum cardinality independent set in $M_1 \cap M_2$, this can be found in polynomial time using matroid intersection algorithm.

If the size of the output is $3(|V|-1)$, then we have found 3 edge disjoint spanning trees. This is simply because the maximum size of an acyclic subgraph in G_j is at most $|V| - 1$ when it is a spanning tree for all $j \in \{1, 2, 3\}$, and the second matroid ensures we only take any edge once in our solution. Thus our solution must be the union of 3 edge disjoint spanning trees. If the size of solution is strictly less than $3(|V| - 1)$ then we output that there is no solution.