MAC0325/5781 COMBINATORIAL OPTIMIZATION: ASSIGNMENT 1

Submit:	Exercises 6, 9, 12, and 16; grad students must additionally submit Exercise 20
Deadline:	28/Sep/2022
Instructions:	Follow **CAREFULLY** the submission instructions listed in Moodle.

The superscripts beside each exercise number indicate *approximately* the lecture number after which you can start working on that exercise.

This version of the assignment has references to commit `c960e82` of the lecture notes. As the latter get revised, actual reference numbers below may be updated. Hence, you should often check back for updated versions of this problem set.

Exercise 1.$^{\text{L04}}$ Run the Bellman-Ford Algorithm 4.1 from the lecture notes on the instance (D, c, r, s) for the minimum-cost walk problem shown in Figure 1 (page 6). The arc costs are described alongside the arcs. Specify the outcome and corresponding certificate found.

Exercise 2.$^{\text{L05}}$ Let $(D, c, r, s) \in \mathcal{I}_{\text{MinWalk}}$. Unpack $D =: (V, A, \varphi)$. Assume that $r \rightsquigarrow v \rightsquigarrow s$ for each $v \in V$. Prove that the (MinWalk) and the optimization problem

$$\begin{aligned} &\text{Maximize} && y(s) - y(r) \\ &\text{subject to} && y(v) - y(u) \leq c(a) \qquad \text{for each } a \in A, \text{ where } uv \coloneqq \varphi(a), \end{aligned} \tag{2.1}$$

have the same optimal value.

Comments. All the results needed to solve this problem were already laid out in the lectures. This problem just tests if you learned your puzzle pieces well. Your job here is just to put them together.

As in the case for arcs with negative costs in the minimum-cost walk problem, you might consider at first that the optimization problem (2.1) and feasible potentials in general are useful *only* to provide certificates of optimality for the minimum-costs walk problem. Exercise 3 shows otherwise.

Exercise 3 (CCPS Ex. 2.35)**.**$^{\text{L05}}$ Suppose that we are given tasks $t_1, \ldots, t_k$. Each task t_i has a processing time p_i. For certain pairs (i, j), task t_i must precede t_j, that is, the processing of t_j cannot begin until the processing of t_i is completed. We wish to schedule the processing of the tasks so that all of the tasks are completed as soon as possible. Formulate this problem in the format of (2.1), and briefly describe an algorithm to solve the optimization problem.

Comments. The statement of the problem should make it very natural to setup an instance for (MinWalk). Verify how the values of feasible potentials for such digraph can be interpreted in terms of this scheduling problem.

Exercise 4 (CCPS Ex. 2.22)**.**$^{\text{L06}}$ Let $D = (V, A, \varphi)$ be a digraph, let $c\colon A \to \mathbb{R}$ be a cost function, and let $R, S \subseteq V$ be nonempty subsets. Consider the optimization problem

$$\begin{aligned} &\text{Minimize} && c(W) \\ &\text{subject to} && W \text{ is an } (r,s)\text{-walk in } D, \\ &&& r \in R, \\ &&& s \in S. \end{aligned} \tag{4.1}$$

That is, we seek a minimum-cost walk from *any* vertex in R to *any* vertex in S. Denote the set of instances of (4.1) by $\mathcal{I}$; each element of $\mathcal{I}$ has the form (D, c, R, S).

Describe an efficiently computable function $\rho\colon \mathcal{I} \to \mathcal{I}_{\text{MinWalk}}$ such that, for each instance $(D, c, R, S) \in \mathcal{I}$, the optimization problem (4.1) is (homomorphically) equivalent to the optimization problem $(\text{MinWalk}(\rho(D, c, R, S)))$.

Hint. Adapt the construction from the proof of Corollary 5.6 from the lecture notes.

Exercise 5 (CCPS Ex. 2.37).$^{\text{L06}}$ Let $G = (V, E, \varphi)$ be a graph, let $c\colon E \to \mathbb{R}$ be a cost function, and let $r, s \in V$. Consider the minimum-cost path problem for graphs:

$$\begin{aligned} &\text{Minimize} && c(P) \\ &\text{subject to} && P \text{ is an } (r,s)\text{-path in } G. \end{aligned} \tag{5.1}$$

Suppose that $c \geq 0$, i.e., $c(e) \geq 0$ for each $e \in E$. Denote the set of instances of (5.1) with $c \geq 0$ by $\mathcal{I}_+$; each element of $\mathcal{I}_+$ has the form (G, c, r, s).

Describe an efficiently computable function $\rho\colon \mathcal{I}_+ \to \mathcal{I}_{\text{MinWalk}}$ such that, for each instance $(G, c, r, s) \in \mathcal{I}_+$, the optimization problem (5.1) is (homomorphically) equivalent to the optimization problem $(\text{MinWalk}(\rho(G, c, r, s)))$. When costs are allowed to be negative, what difficulty arises?

Exercise 6 (CCPS Ex. 2.38).$^{\text{L06}}$ Let $(D, c, r, s) \in \mathcal{I}_{\text{MinWalk}}$. Unpack $D =: (V, A, \varphi)$. Consider the following parity-constrained variation of the minimum-cost walk problem:

$$\begin{aligned} &\text{Minimize} && c(W) \\ &\text{subject to} && W \text{ is an } (r,s)\text{-walk in } D, \\ &&& W \text{ has even length.} \end{aligned} \tag{6.1}$$

(Alternatively, one might consider the odd version of (6.1).) Suppose that $c \geq 0$, i.e., $c(a) \geq 0$ for each $a \in A$. Denote the set of instances of (6.1) with $c \geq 0$ by $\mathcal{I}_+$.

Describe an efficiently computable function $\rho\colon \mathcal{I}_+ \to \mathcal{I}_{\text{MinWalk}}$ such that, for each instance $(D, c, r, s) \in \mathcal{I}_+$, the optimization problem (6.1) is (homomorphically) equivalent to the optimization problem $(\text{MinWalk}(\rho(D, c, r, s)))$.

Comment. Naturally, you must provide the homomorphisms that prove the equivalence, and you must prove that they are indeed homomorphisms.

Hint. Split each vertex $v \in V$ into two "copies", say v_0 and v_1.

Exercise 7 (CCPS Ex. 2.39).$^{\text{L02}}$ Let $(D, c, r, s) \in \mathcal{I}_{\text{MinWalk}}$. Unpack $D =: (V, A, \varphi)$. Consider the bottleneck version of the minimum-cost walk problem:

$$\begin{aligned} &\text{Minimize} && \max_{a \in A(W)} c(a) \\ &\text{subject to} && W \text{ is an } (r,s)\text{-walk in } D. \end{aligned} \tag{7.1}$$

Assume that $m \geq n$. Describe a $O(m \log m)$ algorithm that solves this problem.

Hint. For each arc $a \in A$, how quickly can you determine if the subdigraph having only arcs of cost $\leq c(a)$ has an rs-walk?

Comment. For each walk $W \coloneqq \langle v_0, a_1, v_1, \ldots, a_\ell, v_\ell \rangle$, one may form the vector $i \in [\ell] \mapsto c(a_i)$ in $\mathbb{R}^\ell$. Denote such vector by c_W. Recall that the 1-norm of a vector $x \in \mathbb{R}^n$ is $\|x\|_1 = \sum_{i \in [n]} |x_i|$. Hence, in the minimum-cost walk problem, one seeks to minimize $\|c_W\|_1$. The ∞-norm of a vector $x \in \mathbb{R}^n$ is $\|x\|_\infty = \max_{i \in [n]} |x_i|$. Hence, (7.1) seeks to minimize $\|c_W\|_\infty$.

Exercise 8 (Arbitrage).$^{\text{L03}}$ Suppose V is a (finite) set of currencies. You may think of the Brazilian Real, US dollar, Euro, as well as other usual suspects. Let $r, s \in V$. You want to exchange 1 unit of currency r for the maximum possible amount of units of currency s. In order to do that, you may use several financial services, that sometimes tax you some extra fee on top of the "official"/market exchange rate, or provide you with a discount[1] if they are desperate to get new customers. For

[1]You are welcome to join me in laughing about this possibility. After you solve this question, you will find one of the reasons why this never happens.

simplicity, assume that, if $u, v \in V$ are distinct currencies, you have already shopped for the best exchange rate you can get on the market from u to v, and it allows you to exchange 1 unit of currency u for $c_{uv} > 0$ units of currency v. Hence, for instance, if $v \in V$ is a currency distinct from r and s, you might perform your desired exchange by making 1 unit of currency r into c_{rv} units of the intermediate currency v, which you then exchange for $c_{rv} \cdot c_{vs}$ units of the target currency s.

Formulate the optimization problem of maximizing the number of units of currency s that you can get by paying 1 unit of currency r. Your formulation should use the minimum-cost walk problem. Is it possible for the corresponding minimum-cost walk problem to be unbounded?

Hint. Maximizing a product of positive numbers can be reduced to minimizing the reciprocal of the product of these numbers (why?); the reciprocal of the product is the product of the reciprocals. Which mathematical tool/function did you learn in high school that allows you to "transform products into sums"?

Comments. Your (expected) solution to this question should convince you that negative costs on the arcs for the minimum-cost walk problem may make sense. Remember our discussion in lecture about interpreting the minimum-cost walk problem as a useful "optimization model".

Exercise 9.$^{\text{L06}}$ Let $\mathcal{O} := (X, f, \alpha)$ and $\mathcal{P} := (Y, g, \alpha)$ be optimization problems. Let $\varphi \colon X \to Y$ be a function. Prove that φ is a homomorphism (that is, for each $\mu \in \mathbb{R}$, one has $\varphi(L_{\mathcal{O}}(\mu)) \subseteq L_{\mathcal{P}}(\mu)$) if and only if, for each $x \in X$, one has $\alpha g(\varphi(x)) \geq \alpha f(x)$.

Comments. Equation (6.4) in the lecture notes states this. In this exercise, you are supposed to prove that, **without** using the fact stated in the lecture notes.

Hint. Follow your nose. Maybe break into cases $\alpha = \pm 1$ for better understanding, but the proof should be written for both values of α at the same time.

Exercise 10 (Exercise 6.5 from the lecture notes)**.**$^{\text{L06}}$ Let $\mathcal{O}$ and $\mathcal{P}$ be equivalent optimization problems. Prove that $\operatorname{opt}(\mathcal{O}) = \operatorname{opt}(\mathcal{P})$ and that precisely one of the following holds:

(i) $\mathcal{O}$ and $\mathcal{P}$ are infeasible;
(ii) $\mathcal{O}$ and $\mathcal{P}$ are unbounded;
(iii) $\mathcal{O}$ and $\mathcal{P}$ have (the same) finite optimal values and no optimal solutions;
(iv) $\mathcal{O}$ and $\mathcal{P}$ have optimal solutions; moreover, if $\mathcal{O} \xrightarrow{\varphi} \mathcal{P}$ and x^* is an optimal solution for $\mathcal{O}$, then $\varphi(x^*)$ is an optimal solution for $\mathcal{P}$.

Hint. In the case with finite optimal values, it might be useful to first consider the "moreover" statement.

Exercise 11.$^{\text{L06}}$ A **partial order** $\preccurlyeq$ on a set P is a binary relation on P if

(i) $\preccurlyeq$ is reflexive;
(ii) $\preccurlyeq$ is **antisymmetric**, i.e., $x \preccurlyeq y$ and $y \preccurlyeq x$ imply that $x = y$, for each $x, y \in P$;
(iii) $\preccurlyeq$ is transitive.

In this case, it is usual to refer to the ordered pair $(P, \preccurlyeq)$ as a **poset**, from "partially-ordered set". For $x, y \in P$, we write $x \prec y$ to mean that $x \preccurlyeq y$ and $x \neq y$. An element x of a subset $S \subseteq P$ is **maximal** (in S) if there is no $y \in S$ such that $x \prec y$.

Let $G = (V, E, \varphi)$ be a graph. Consider the poset $(P, \preccurlyeq)$ where P is the set of all subgraphs of G and we write $F \preccurlyeq H$ if F is a subgraph of H. (You should verify that $\preccurlyeq$ is indeed a partial order, though this is not part of the exercise.) Let S be the set of subgraphs of G that are connected. Recall the definition of connected graphs, components, and the equivalence relation $\sim_G$ in §6.11 from the lecture notes. Prove that the components of G (as induced subgraphs) are the maximal elements of S. In other words, the components of G are the maximal connected subgraphs of G, i.e., the maximal elements in the set of connected subgraphs of G, with respect to the subgraph partial order.

Comment. This problem is not very hard. The only (and perhaps, very real) difficulty is in loading in your brain all the relevant (formal) concepts.

Exercise 12 (CCPS Ex. 3.16).$^{\text{L08}}$ Run Kőnig's Algorithm 8.1 from the lecture notes on the graph from Figure 2 in page 6 to find a maximum matching and a minimum vertex cover. You must follow **precisely** the instructions below.

Instructions. For each iteration of the **for** loop in Kőnig's Algorithm, you must write the augmenting path found and the new matching; when writing the augmenting path, it is okay to list only vertices, i.e., skip the edges in the sequence. At the end of the algorithm, you must list the vertices of the vertex cover found. You must **not** write anything else in your solution!

The algorithm presented allows for many arbitrary choices. We will add further specifications so that **there is only one correct sequence** of augmenting paths and intermediate matchings for this exercise. Set U to be the vertices at the top of Figure 2, i.e., the lowercase letters. When performing a search for an augmenting path at each iteration of the **for** loop, you must always find a **minimum-length** M_t-augmenting path from a vertex in $U \setminus V_{M_t}$ to a vertex in $W \setminus V_{M_t}$ and, among all such minimum-length augmenting paths, you must choose **the lexicographically smallest** one. For instance, the first augmenting path found is $\langle a, A\rangle$.

One way to achieve this is to perform BFS (see [2, Ch. 22]) starting with the M_t-free vertices from U in the queue, in alphabetic order, and whenever the list of outgoing arcs from a vertex is traversed, the "out-neighbors" should be traversed in alphabetic order. Alternatively, one may adapt the Bellman-Ford Algorithm 4.1 from the lecture notes to find the desired paths.

Note that the vertices in the graph from Figure 2 were relabeled from the corresponding picture that appears in [1].

Exercise 13.$^{\text{L08}}$ Let G be a (U, W)-bipartite graph. For each subset $S \subseteq U$, denote $N(S) := \{\, w \in W : w \text{ has a neighbor in } S\}$. Prove that precisely one of the following holds:

(i) G has a matching that saturates U, or

(ii) there is $S \subseteq U$ such that $|S| > |N(S)|$.

Hint. First prove that both options cannot hold. Next assume (i) fails, and prove (ii). For that, rely on Kőnig's matching theorem. From a vertex cover $C = C_U \cup C_W$ with $C_U \subseteq U$ and $C_W \subseteq W$, consider the set $S := U \setminus C_U$. Can there be an edge between $U \setminus C_U$ and $W \setminus C_W$?

Comment. Interpret this result in terms of certificates.

Exercise 14 (CCPS Ex. 5.2).$^{\text{L07}}$ Let $G = (V, E, \varphi)$ be a graph, and let M be a matching in G. Let $\nu(G)$ denote the maximum size of a matching in G. Adapt the proof method of Berge's Theorem to show that there are at least $\nu(G) - |M|$ pairwise vertex-disjoint M-augmenting paths; here we call two paths P and Q **vertex-disjoint** if $V(P) \cap V(Q) = \varnothing$.

Exercise 15.$^{\text{L02}}$ Let $(S, \preccurlyeq)$ be a poset with S finite; see Exercise 11 for the definition. Elements $x, y \in S$ are **comparable** if $x \preccurlyeq y$ or $y \preccurlyeq x$; otherwise, x and y are **incomparable**. A **chain** is a subset of pairwise comparable elements of S, and an **antichain** is a subset of pairwise incomparable elements of S. Prove that the optimal values of the optimization problems below are equal:

$$\begin{aligned} &\text{Maximize} && |C| \\ &\text{subject to} && C \subseteq S \text{ is a chain,} \end{aligned} \tag{15.1}$$

and

$$\begin{aligned} &\text{Minimize} && |\mathcal{A}| \\ &\text{subject to} && \mathcal{A} \subseteq \mathcal{P}(S) \text{ is a collection of antichains,} \\ &&& \textstyle\bigcup \mathcal{A} = S. \end{aligned} \tag{15.2}$$

Comments. This minmax relation is known as Mirsky's theorem.

Hint. Can a chain and an antichain intersect in more than one element? Use your answer to show that, if C is feasible for (15.1) and $\mathcal{A}$ is feasible for (15.2), then $|C| \le |\mathcal{A}|$. To find feasible solutions where equality holds, define the *height* of $s \in S$ as the maximum size of a chain in S with maximal element s. Can two elements of the same height be comparable?

Exercise 16.$^{\text{L08}}$ Let $(S, \preccurlyeq)$ be a poset with S finite; we use the terminology described in Exercise 15. Prove that the optimal values of the optimization problems below are equal:

$$\begin{aligned} &\text{Maximize} && |A| \\ &\text{subject to} && A \subseteq S \text{ is an antichain,} \end{aligned} \tag{16.1}$$

and

$$\begin{aligned} &\text{Minimize} && |\mathcal{C}| \\ &\text{subject to} && \mathcal{C} \subseteq \mathcal{P}(S) \text{ is a collection of chains,} \\ &&& \bigcup \mathcal{C} = S. \end{aligned} \tag{16.2}$$

Comments. This minmax relation is known as Dilworth's decomposition theorem.

Hint. Can a chain and an antichain intersect in more than one element? Use your answer to show that, if A is feasible for (16.1) and $\mathcal{C}$ is feasible for (16.2), then $|A| \leq |\mathcal{C}|$. To prove that equality may be achieved, build a bipartite graph having two copies of each element of S, and apply Kőnig's matching theorem.

Exercise 17.$^{\text{L08}}$ Let G be a regular bipartite graph with at least one edge. Prove that G has a perfect matching.

Hint. First consider whether it is possible to have color classes of different sizes. Next, verify that one of the options in Exercise 13 is not possible. For a subset $S \subseteq V$ of vertices of a graph $G = (V, E, \varphi)$, the cut determined by S (in G) is $\delta_G(S) := \{\, e \in E : \varphi(e) \cap S \neq \varnothing \neq \varphi(e) \cap \overline{S} \}$, where $\overline{S} := V \setminus S$. How are $\delta(S)$ and $\delta(N(S))$ related? Also, how is $|\delta(T)|$ related to $|T|$ and the valency of G, if G is regular?

Exercise 18.$^{\text{L07}}$ Let $G = (V, E, \varphi)$ be a (U, W)-bipartite graph. Let $M, N \subseteq E$ be maximum matchings. Show that there is a maximum matching that saturates $(V_M \cap U) \cup (V_N \cap W)$.

Hint. Components of $(V, M \mathbin{\Delta} N, \varphi\restriction_{M \Delta N})$.

Exercise 19.$^{\text{L08}}$ Let $G = (V, E)$ be a graph. Call $S \subseteq V$ a **clique** if $G[S] = (S, \binom{S}{2})$, and call $S \subseteq V$ a **stable** or **independent** set if $G[S] = (S, \varnothing)$. The maximum size of a clique in G is denoted $\omega(G)$. A **coloring** of G is a partition of V into stable sets of G. The minimum size of a coloring of G is denoted $\chi(G)$, called the **chromatic number** of G. The graph G is called **perfect** if, for each $S \subseteq V$, one has $\omega(G[S]) = \chi(G[S])$. The **complement** of G is the graph $\overline{G} := (V, \binom{V}{2} \setminus E)$. Prove that, if G is bipartite, then $\overline{G}$ is perfect.

Exercise 20.$^{\text{L05}}$ Design a polynomial-time algorithm for the following problem. Given a digraph $D = (V, A, \varphi)$, vertices $r, s \in V$, and a cost function $c\colon A \to \mathbb{R}$ such that D has no negative cycles, find the set

$$\bigcup \{\, A(W) : W \text{ is a min-cost } rs\text{-walk in } D \}.$$

That is, one wants to find all the arcs that lie in *some* min-cost rs-walk. Prove correctness and determine the running time of your algorithm.

References

[1] W. J. Cook, W. H. Cunningham, W. R. Pulleyblank, and A. Schrijver. *Combinatorial optimization.* Wiley-Interscience Series in Discrete Mathematics and Optimization. A Wiley-Interscience Publication. John Wiley & Sons, Inc., New York, 1998, pages x+355 (cited on page 4).

[2] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to algorithms.* Third. MIT Press, Cambridge, MA, 2009, pages xx+1292 (cited on page 4).

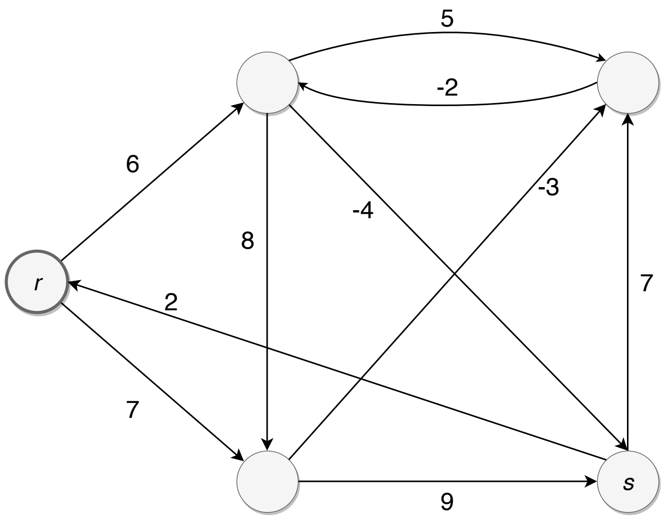

Figure 1. The input instance for Exercise 1.

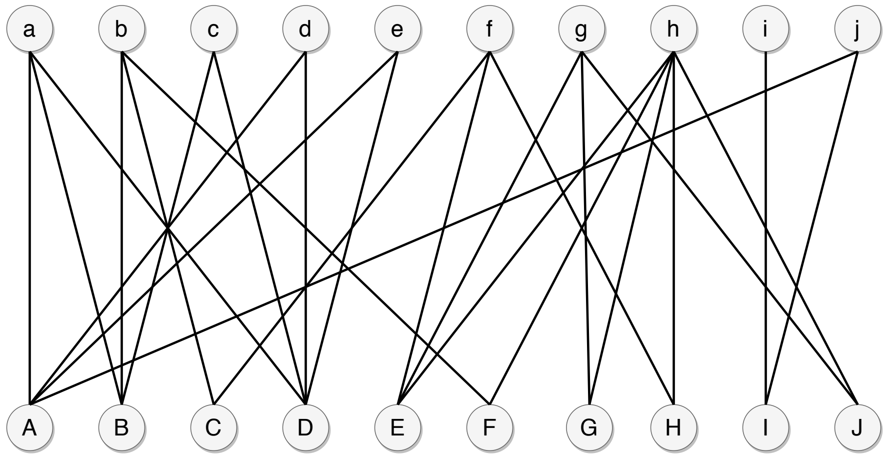

Figure 2. The input instance for Exercise 12.