15-850: Advanced Algorithms **CMU, Spring 2024**
Exercises with Homework #2

Exercises are for fun and edification, please do not submit. (You may discuss these exercises with others.) The ones below are grouped by topic and their subparts do not necessarily build on one another (for example, you do not need to do (a) to do (b)).

1. **Matrix Multiplication is Useful.** Given an undirected simple graph $G = (V, E)$, a triangle is just a clique of size 3; i.e., 3 vertices such that all 3 edges are present. Give algorithms for the following problems:
   - Find a triangle in G in time n^ω.
   - Find a $3k$-clique in a graph in time $n^{k\omega}$.
   - Find a triangle in G in time $m^{1.5}$. (This one is slightly harder—not an exercise—and does not use matrix multiplication.)

2. **Low-Diameter Decompositions for Simple Graphs.** Recall a β-low-diameter decomposition, given graph G and distance D, randomly breaks it into pieces of max-distance D, such that each pair x, y is separated with probability at most $\frac{d_G(x,y)}{D} \cdot \beta$.
   (a) Show that if each *edge* $(x, y) \in E(G)$ is cut with probability $\frac{d_G(x,y)}{D} \cdot \beta$, then so is any pair $x, y \in V$. Hence, if the graph only has unit-weight edges, each edge can be cut with probability at most β/D.
   (b) Show that (i) any path graph has an LDD with $\beta = 1$, (ii) any tree with $\beta = 2$, and (iii) the standard k-dimensional $(n^{1/k} \times n^{1/k} \times \cdots \times n^{1/k})$-grid with $\beta = k$.

3. **Approximation via Randomized Simplification.** In Lecture #5 we saw low-stretch trees, and used them to approximate APSP on general graphs. We explore this connection further, via the k-median problem we saw in HW#0: Given a graph G and k, the k-median problem asks you to find a set $C \subseteq V$ with $|C| = k$ to minimize $\Phi_G(C) := \sum_{v \in V} d_G(v, C)$.
   (a) Given an algorithm to solve k-median optimally on trees, show that the algorithm that samples a tree T from (random) low-stretch tree distribution with stretch α, solves k-median on T to get C_T, and outputs this set C_T, ensures that the expected cost $\mathbf{E}_T[\Phi_G(C_T)] \leq \alpha OPT$.
   (b) Show that if you perform $L := O(\frac{\log n}{\varepsilon})$ independent runs of the above algorithm to get sets $C_1, C_2, \ldots C_L$, and return the set with the least $\Phi_G(C_i)$ value from among these (call it C^*), then $\mathbf{Pr}[\Phi_G(C_T) > (1 + \varepsilon)\alpha OPT] \leq 1/\text{poly}(n)$.
   (c) Show that the expected weight of a low-stretch tree is at most $O(\alpha)$ times the MST.
   (d) What kinds of problems can you solve using the ideas in the above parts. E.g., does it work for APSP? How about the K-center problem? Or the K-means problem which wants to minimize $\Psi_d(C) := \sum_{v \in X}(d(v, C))^2$. Why or why not?
   (e) (Slightly non-trivial) Extend your dynamic-programming algorithm from HW#0 to solve k-median on an edge-weighted tree. Hint: first solve it on a binary tree. Then show how to reduce the problem to binary trees.

4. **Flows, Kings, and Halls.** Recall that König's theorem says for a bipartite graph G, the size of the maximum matching in G is equal to the size of the minimum vertex cover.

(a) Use the max-flow/min-cut theorem to prove Kőnig's theorem. (Recall that the max-flow/min-cut theorem says that in any flow network, the maximum s-t flow equals the minimum s-t cut. Moreover if the arc capacities are integers, then the max-flow is guaranteed to be integral.)

(b) Use Kőnig's theorem to prove Hall's theorem:

> In a bipartite graph $G = (L, R, E)$, for any set $S \subseteq L$, let $N(S) = \{r \in R \mid \exists \ell \in S, (\ell, r) \in E\}$ be the neighbors of S. Then G has a matching of size $|L|$ if and only if $|N(S)| \geq |S|$ for all $S \subseteq L$.

(c) Here is the perfect matching polytope on general graphs, given by the exponentially many inequalities:

$$K_{\text{pm-nonbip}} := \{x \in \mathbb{R}^m \mid \sum_{e \in \partial v} x_e = 1, \sum_{e \in \partial S} x_e \geq 1 \text{ for all odd sets } S, x \geq 0\}.$$

As before ∂S is the set of edges with one end the set S and the other end outside. Show that the "odd set inequalities" $\sum_{e \in \partial S} x_e \geq 1$ for odd cardinality sets S can be replaced by $\sum_{e \in E_S} x_e \leq \lfloor |S|/2 \rfloor$, where E_S is the set of edges both of whose endpoints lie in S.

5. **Matching Reductions.**

(a) Suppose you have an algorithm that solves max-weight perfect matchings (MaxWPM) for all non-negative weight functions. Give reductions that allow you to solve (a) min-weight perfect matchings (MinWPM) and (b) min-weight max-cardinality matchings (MinWMaxM)—i.e., among all matchings of size equal to $MM(G)$ find the one with least weight. Your reduction should only make a *single call* to the max-weight perfect matching oracle.

If the MinWPM and MinWMaxM instances are bipartite, ensure that the reductions give you MaxWPM instances that are bipartite too.

(b) Now suppose you are allowed to make multiple calls. Show how to devise an algorithm that tells you the maximum cardinality of a matching in a graph using an algorithm that tells you whether a graph has a perfect matching or not. You should use only $O(\log n)$ calls to the PM algorithm.

6. **Self-Reduction.** Suppose you have an algorithm A that takes a graph G and a number K, and outputs `yes` if the graph has a vertex cover of size at most K, and `no` otherwise. Give an algorithm for the search problem, i.e., one that takes G, K and outputs a vertex cover of size at most K (if one exists), using at most n calls to A.

7. **Practice taking Duals of Large LPs.** Recall that a (primal) linear program in standard form can be represented by

$$\begin{aligned} \text{minimize} \quad & \mathbf{c}^T \mathbf{x} \\ \text{subject to} \quad & A\mathbf{x} \geq \mathbf{b} \\ & \mathbf{x} \geq 0. \end{aligned}$$

Here, $\mathbf{c}$ and $\mathbf{x}$ are vectors in $\mathbb{R}^n$ where n is the number of variables, and $\mathbf{b}$ is a vector in $\mathbb{R}^m$ whre m is the number of constraints. The matrix $A \in \mathbb{R}^{m \times n}$ has m rows and n columns. The *dual* linear program is defined by

$$\begin{aligned} \text{maximize} \quad & \mathbf{b}^T\mathbf{y} \\ \text{subject to} \quad & A^T\mathbf{y} \leq \mathbf{c} \\ & \mathbf{y} \geq 0. \end{aligned}$$

When the primal linear program is not represented in standard form, it is easiest to first transform it to standard form (which may involve adding new variables or constraints) and then take the dual.

(a) Recall the maximum weight perfect matching LP for bipartite graphs with vertices (buyers) b on the left and vertices (items) i on the right:

$$\begin{aligned} \text{maximize} \quad & \sum_{bi} v_{bi}x_{bi} & \\ \text{subject to} \quad & \sum_{b} x_{bi} = 1 & \forall i \\ & \sum_{i} x_{bi} = 1 & \forall b \\ & x_{bi} \geq 0 & \forall bi \end{aligned}$$

Show that the dual is

$$\begin{aligned} \text{minimize} \quad & \sum_{i} p_i + \sum_{b} u_b & \\ \text{subject to} \quad & p_i + u_b \geq v_{bi} & \forall bi \\ & y_{bi} \text{ unconstrained} & \forall bi \end{aligned}$$

In case you are stuck, **I have written a sample solution for this problem at the end of this document.**

(b) Recall the minimum cost r-arborescence LP for a directed graph $G = (V, A)$:

$$\begin{aligned} \text{minimize} \quad & \sum_{a\in A} w_a x_a & \\ \text{subject to} \quad & \sum_{a\in\partial^+ S} x_a \geq 1 & \forall S \subseteq V - \{r\}, |S| > 1 \\ & \sum_{a\in\partial^+ v} x_a = 1 & \forall v \neq r \\ & x_a \geq 0 & \forall a \in A \end{aligned}$$

Show that the dual is

$$\begin{aligned} \text{maximize} \quad & \sum_{S\subseteq V-\{r\}} y_S & \\ \text{subject to} \quad & \sum_{S: a\in\partial^+ S} y_S \leq w_a & \forall a \in A \\ & y_S \geq 0 & \forall S \subseteq V - \{r\}, |S| > 1 \\ & y_{\{v\}} \text{ unconstrained} & \forall v \neq r \end{aligned}$$

(c) Consider the following (exponential-sized) LP for maximum s-t flow with edge capacities c_e: let $\mathcal{P}$ be the (exponentially large) set of (non-self-intersecting) paths from s to t. For each path $P \in \mathcal{P}$, declare a variable $f_P \geq 0$ which denotes the amount of flow sent along path P. We can write the maximum s-t flow LP as

$$\begin{aligned} \text{maximize} \quad & \sum_{P \in \mathcal{P}} f_P \\ \text{subject to} \quad & \sum_{P \in \mathcal{P}: e \in P} f_P \leq c_e && \forall e \in E \\ & f_P \geq 0 && \forall P \in \mathcal{P}. \end{aligned}$$

Note that there is a smaller, polynomial-sized max-s-t-flow LP that simply enforces flow conservation at each vertex outside s and t. However, the exponential-sized LP has many conceptual advantages, as we will see below.

i. Show that the dual is

$$\begin{aligned} \text{minimize} \quad & \sum_{e \in E} c_e \ell_e \\ \text{subject to} \quad & \sum_{e \in P} \ell_e \leq 1 && \forall P \in \mathcal{P} \\ & \ell_e \geq 0 && \forall e \in E \end{aligned}$$

Hint: in this case, the original s-t-flow LP already fits the dual LP very well. So it may be easier to encode it as the dual, and the new LP above as the primal. This works because taking the dual of the dual LP gives back the primal LP.

ii. How can we interpret the dual LP above? In particular, what does the constraint $\sum_{e \in P} \ell_e \leq 1$ for all $P \in \mathcal{P}$ mean conceptually? (Hint: think about the variables ℓ_e as edge *lengths*.)

iii. Now consider the LP for maximum *budget-constrained* s-t flow within a given budget $B \geq 0$. There is a cost/weight w_e on each edge $e \in E$ (unrelated to its capacity c_e), and for each path $P \in \mathcal{P}$, let $w_P = \sum_{e \in P} w_e$ be the total cost of edges in P. We require the total cost of the s-t flow to be at most B:

$$\begin{aligned} \text{maximize} \quad & \sum_{P \in \mathcal{P}} f_P \\ \text{subject to} \quad & \sum_{P \in \mathcal{P}: e \in P} f_P \leq c_e && \forall e \in E \\ & \sum_{P \in \mathcal{P}} w_P f_P \leq B \\ & f_P \geq 0 && \forall P \in \mathcal{P}. \end{aligned}$$

Compute the dual and interpret it in a similar way. (Hint: imagine increasing/decreasing the length of each edge by the same amount.)

Solution to (a): Write constraint $\sum_b x_{bi} = 1$ as $\sum_b x_{bi} \leq 1$ and $\sum_b x_{bi} \geq 1$. Multiply the latter inequality by -1 to get $\sum_b -x_{bi} \leq -1$. We can rewrite the LP as

$$\begin{aligned}
\text{maximize} \quad & \sum_{bi} v_{bi} x_{bi} & \\
\text{subject to} \quad & \sum_{b} x_{bi} \leq +1 & \forall i \\
& \sum_{b} -x_{bi} \leq -1 & \forall i \\
& \sum_{i} x_{bi} \leq +1 & \forall b \\
& \sum_{i} -x_{bi} \leq -1 & \forall b \\
& x_{bi} \geq 0 & \forall bi
\end{aligned}$$

The solution vector $\mathbf{x}$ has an entry x_{bi} for each edge bi. The objective vector $\mathbf{c}$ has coordinate v_{bi} for each edge bi. The constraint matrix A has:

(a) for each item i, a row named i^+ with $+1$ on entry bi for all b and other entries 0,

(b) for each item i, a row named i^- with -1 on entry bi for all b and other entries 0,

(c) for each buyer b, a row named b^+ with $+1$ on entry bi for all i and other entries 0,

(d) for each buyer b, a row named b^- with -1 on entry bi for all i and other entries 0.

The constraint vector $\mathbf{b}$ has value $+1$ on each coordinate i^+ and b^+, and value -1 on each coordinate i^- and b^-.

This LP is already in standard dual form, so its dual is the primal LP, where I use $\mathbf{y}$ instead of $\mathbf{x}$ since $\mathbf{x}$ is already taken:

$$\begin{aligned}
\text{minimize} \quad & \mathbf{b}^T\mathbf{y} \\
\text{subject to} \quad & A^T\mathbf{y} \geq \mathbf{c} \\
& \mathbf{y} \geq 0.
\end{aligned}$$

The vector $\mathbf{y}$ has the same dimension as $\mathbf{b}$, so it has coordinates i^+, i^- for each i and coordinates b^+, b^i for each b. The objective function $\mathbf{b}^T\mathbf{y}$ can be written as

$$\begin{aligned}
\mathbf{b}^T\mathbf{y} &= \sum_i b_{i^+} y_{i^+} + \sum_i b_{i^-} y_{i^-} + \sum_b b_{b^+} y_{b^+} + \sum_b b_{b^-} y_{b^-} \\
&= \sum_i (+1) y_{i^+} + \sum_i (-1) y_{i^-} + \sum_b (+1) y_{b^+} + \sum_b (-1) y_{b^-} \\
&= \sum_i (y_{i^+} - y_{i^-}) + \sum_b (y_{b^+} - y_{b^-}).
\end{aligned}$$

Next, we focus on the constraint $A^T\mathbf{y} \geq \mathbf{c}$. Recall that matrix A a column for each edge bi and a column for each i^+, i^-, b^+, b^-. So A^T has a row for each edge bi and a column for each i^+, i^-, b^+, b^-. Each row bi of the constraint $A^T\mathbf{y} \geq \mathbf{c}$ can be written as

$$\underbrace{\sum_i y_{i^+}}_{\text{columns } i^+} + \underbrace{\sum_i -y_{i^-}}_{\text{columns } i^-} + \underbrace{\sum_b y_{b^+}}_{\text{columns } b^+} + \underbrace{\sum_b -y_{b^-}}_{\text{columns } b^-} \geq v_{bi}.$$

So far, everything was mechanical: tedious but not insightful. Here comes the main insight: *define a new variable p_i to represent $y_{i+} - y_{i-}$ and a new variable u_b to represent $y_{b+} - y_{b-}$.* The variables p_i are unconstrained since $p_i = y_{i+} - y_{i-}$ where $y_{i+}, y_{i-} \geq 0$, and the variables u_b are unconstrained too. With this transformation, the objective function becomes $\sum_i p_i + \sum_b u_b$ and the constraints become $\sum_i p_i + \sum_b u_b \geq v_{bi}$, exactly as desired. Note that this transformation is equivalent: from any solution with variables $y_{i+}, y_{i-}, y_{b+}, y_{b-}$ we can transform it to a solution with p_i, u_b (by the way we defined p_i, u_b), and from any solution with p_i, u_b we can transform it to a solution with $y_{i+}, y_{i-}, y_{b+}, y_{b-}$ (since we can always find $y_{i+}, y_{i-} \geq 0$ whose difference is p_i, and same for u_b).