Problem set 3

Electronic submission to Gradescope due **11:59pm Friday 3/1**. Form a group of 2-3 students — that is, submit one homework with all of your names.

*You may discuss these problems with your classmates, but please do not look for answers to these problems on the Internet. Your submission must be the original work of you and your partners, and you must understand everything that is written on your submission. We strongly suggest that you type up your solutions in LaTeX. A template is provided with the class notes.*

1. **Problem 1**

The Minimum Vertex Cover problem in general undirected graphs is to choose the smallest vertex cover of the graph (recall that a vertex cover is a subset $S$ of $V$ such that for every edge, at least one of its endpoints is in $S$).

1. Write this as a mathematical program which allows binary constraints on variables (i.e. you can restrict variables to be either 0 or 1) in addition to linear constraints and a linear objective function.

2. Relax the binary constraints by allowing your decision variables to lie in the range $[0, 1]$. Does the optimum solution to the resulting linear program provide an upper bound or a lower bound on the optimum solution to the original mathematical program? Explain.

3. Show that every basic feasible solution to the resulting linear program is half-integral, i.e., the decision variables take on values 0, 1/2, or 1.

4. Show how to obtain a 2-approximation for the Minimum Vertex Cover problem, i.e., a polynomial time algorithm that is guaranteed to find a vertex cover that is no larger than twice the smallest vertex cover.

2. **Problem 2**

Recall the multicommodity flow problem from Exercise 18. Recall the input consists of a directed graph $G = (V, E)$, $k$ “commodities” or source-sink pairs $(s_1, t_1), \ldots, (s_k, t_k)$, and a positive capacity $u_e$ for each edge.

Consider also the multicut problem, where the input is the same as in the multicommodity flow problem, and feasible solutions are subsets $F \subseteq E$ of edges such that, for every commodity $(s_i, t_i)$, there is no $s_i$-$t_i$ path in $G = (V, E \setminus F)$. (Assume that $s_i$ and $t_i$ are distinct for each $i$.) The value of a multicut $F$ is just the total capacity $\sum_{e \in F} u_e$.

---

1Thanks to Tim Roughgarden for letting us reuse some of the problems from his course.
(a) Formulate the multicommodity flow problem as a linear program with one decision variable for each path \( P \) that travels from a source \( s_i \) to the corresponding sink \( t_i \). Aside from nonnegativity constraints, there should be only be \( m \) constraints (one per edge).

[Note: this is a different linear programming formulation than the one asked for in Exercise 18.]

(b) Take the dual of the linear program in (a). Prove that every optimal 0-1 solution of this dual — i.e., among all feasible solutions that assign each decision variable the value 0 or 1, one of minimum objective function value — is the characteristic vector of a minimum-value multicut.

c) Show by example that the optimal solution to this dual linear program can have objective function value strictly smaller than that of every 0-1 feasible solution. In light of your example, explain a sense in which there is no max-flow/min-cut theorem for multicommodity flows and multicuts.

**Problem 3**

This problem gives a linear-time (!) randomized algorithm for solving linear programs that have a large number \( m \) of constraints and a small number \( n \) of decision variables. (The constant in the linear-time guarantee \( O(m) \) will depend exponentially on \( n \).)

Consider a linear program of the form

\[
\max c^T x
\]

subject to

\[
Ax \leq b.
\]

For simplicity, assume that the linear program is feasible with a bounded feasible region, and let \( M \) be large enough that \( |x_j| < M \) for every coordinate of every feasible solution. Assume also that the linear program is “non-degenerate,” in the sense that no feasible point satisfies more than \( n \) constraints with equality. For example, in the plane (two decision variables), this just means that there does not exist three different constraints (i.e., halfplanes) whose boundaries meet at a common point. Finally, assume that the linear program has a unique optimal solution.\(^2\)

Let \( C = \{1, 2, \ldots, m\} \) denote the set of constraints of the linear program. Let \( B \) denote additional constraints asserting that \(-M \leq x_j \leq M \) for every \( j \). The high-level idea of the algorithm is: (i) drop a random constraint and recursively compute the optimal solution \( x^* \) of the smaller linear program; (ii) if \( x^* \) is feasible for the original linear program, return it; (iii) else, if \( x^* \) violates the constraint \( a_i^T x \leq b_i \), then change this inequality to an equality and recursively solve the resulting linear program.

More precisely, consider the following recursive algorithm with two arguments. The first argument \( C_1 \) is a subset of inequality constraints that must be satisfied (initially, equal to \( C \)). The second argument is a subset \( C_2 \) of constraints that must be satisfied with equality (initially, \( \emptyset \)). The responsibility of a recursive call is to return a point maximizing \( c^T x \) over all points that satisfy all the constraints of \( C_1 \cup B \) (as inequalities) and also those of \( C_2 \) (as equations).

\(^2\) All of these simplifying assumptions can be removed without affecting the asymptotic running time; we leave the details to the interested reader.
Linear-Time Linear Programming

**Input:** two disjoint subsets $C_1, C_2 \subseteq C$ of constraints;

**Base case #1:** if $|C_2| = n$, return the unique point that satisfies every constraint of $C_2$ with equality;

**Base case #2:** if $|C_1| + |C_2| = n$, return the point that maximizes $c^T x$ subject to $a_i^T x \leq b_i$ for every $i \in C_1$, $a_i^T x = b_i$ for every $i \in C_2$, and the constraints in $B$;

**Recursive step:**
choose $i \in C_1$ uniformly at random;
recurse with the sets $C_1 \setminus \{i\}$ and $C_2$ to obtain a point $x^*$;
if $a_i^T x^* \leq b_i$ then
return $x^*$
else
recurse with the sets $C_1 \setminus \{i\}$ and $C_2 \cup \{i\}$, and return the result

(a) Prove that this algorithm terminates with the optimal solution $x^*$ of the original linear program.
[Hint: be sure to explain why, in the “else” case, it’s OK to recurse with the ith constraint set to an equation.]

(b) Let $T(m, s)$ denote the expected number of recursive calls made by the algorithm to solve an instance with $|C_1| = m$ and $|C_2| = s$ (with the number $n$ of variables fixed). Prove that $T$ satisfies the following recurrence:

$$T(m, s) = \begin{cases} 1 & \text{if } s = n \text{ or } m + s = n \\ T(m - 1, s) + \frac{n-s}{m} \cdot T(m - 1, s + 1) & \text{otherwise} \end{cases}$$

[Hint: you should use the non-degeneracy assumption in this part.]

(c) Prove that $T(m, 0) \leq n! \cdot m$.
[Hint: it might be easiest to make the variable substitution $\delta = n - s$ and proceed by simultaneous induction on $m$ and $\delta$.]

(d) Conclude that, for every fixed constant $n$, the algorithm above can be implemented so that the expected running time is $O(m)$ (where the hidden constant can depend arbitrarily on $n$).

**Problem 4**

You are given an arm $A$ which can be in one of a finite set of states $S$. If you play the arm in state $a$, it goes to state $b$ with probability $P(a, b)$ which is given to you, and provides expected reward $0 \leq r(a) \leq 1$ where $r(a)$ is also given to you. Assume that you can observe the state of the arm $A$ at all times, and the arm is in state $0$ at time $t = 0$. Recall that the reference arm $R_p$ is defined as an arm that gives an expected reward $p$ whenever it is played. Recall that we defined the Gittins Index $g$ of arm $A$ as $g = p/(1 - \theta)$ where $0 < \theta < 1$ is the discount factor, and where $p$ is chosen such that the optimum multi-armed bandit algorithm given arm $A$ (starting in state 0) and arm $R_p$ is indifferent between which arm it plays at time $t = 0$.

Provide a linear program to compute $g$. The linear program should be of size polynomial in $|S|$. 3
Problem 5

This problem considers a variant of the online decision-making problem. There are \( n \) “experts,” where \( n \) is a power of 2.

**Combining Expert Advice**

At each time step \( t = 1, 2, \ldots, T \):

- each expert offers a prediction of the realization of a binary event (e.g., whether a stock will go up or down);
- a decision-maker picks a probability distribution \( p^t \) over the possible realizations 0 and 1 of the event;
- the actual realization \( r^t \in \{0, 1\} \) of the event is revealed;
- a 0 or 1 is chosen according to the distribution \( p^t \), and a mistake occurs whenever it is different from \( r^t \);

You are promised that there is at least one omniscient expert who makes a correct prediction at every time step.

(a) Prove that the minimum worst-case number of mistakes that a deterministic algorithm can make is precisely \( \log_2 n \).

(b) Prove that the minimum worst-case expected number of mistakes that a randomized algorithm can make is precisely \( \frac{1}{2} \log_2 n \).

Problem 6

The follow-the-leader (FTL) algorithm is a natural online decision-making algorithm, which at time step \( t \) chooses the action \( a \) with maximum cumulative reward \( \sum_{u=1}^{t-1} r^u(a) \) so far.

(a) Show that the worst-case regret of the algorithm can grow linear in \( T \). And more generally every deterministic algorithm, can have regret that grows linearly with \( T \).

Now consider a randomized variant of FTL, the follow-the-perturbed-leader (FTPL) algorithm, with worst-case regret comparable to that of the multiplicative weights algorithm. In the description of FTPL, we define each probability distribution \( p^t \) over actions implicitly through a randomized subroutine.
Follow-the-Perturbed-Leader (FTPL) Algorithm

for each action $a \in A$ do
  independently sample a geometric random variable with parameter $\eta$\(^3\) denoted by $X_a$;
for each time step $t = 1, 2, \ldots, T$ do
  choose the action $a$ that maximizes the perturbed cumulative reward $X_a + \sum_{u=1}^{t-1} r^u(a)$ so far;

For convenience, assume that, at every time step $t$, there is no pair of actions whose (unperturbed) cumulative rewards-so-far differ by an integer.

(b) Prove that, at each time step $t = 1, 2, \ldots, T$, with probability at least $1 - \eta$, the largest perturbed cumulative reward of an action prior to $t$ is more than 1 larger than the second-largest such perturbed reward.

[Hint: Sample the $X_a$’s gradually by flipping coins only as needed, pausing once the action $a^*$ with largest perturbed cumulative reward is identified. Resuming, only $X_{a^*}$ is not yet fully determined. What can you say if the next coin flip comes up “tails?”]

(c) As a thought experiment, consider the (unimplementable) algorithm that, at each time step $t$, picks the action that maximizes the perturbed cumulative reward $X_a + \sum_{u=1}^{t} r^u(a)$ over $a \in A$, taking into account the current reward vector. Prove that the regret of this algorithm is at most $\max_{a \in A} X_a$.

[Hint: Consider first the special case where $X_a = 0$ for all $a$. Iteratively transform the action sequence that always selects the best action in hindsight to the sequence chosen by the proposed algorithm. Work backward from time $T$, showing that the reward only increases with each step of the transformation.]

(d) Prove that $E[\max_{a \in A} X_a] \leq b\eta^{-1} \ln n$, where $n$ is the number of actions and $b > 0$ is a constant independent of $\eta$ and $n$.

[Hint: use the definition of a geometric random variable and remind yourself about “the union bound.”]

(e) Prove that, for a suitable choice of $\eta$, the worst-case expected regret of the FTPL algorithm is at most $b\sqrt{T \ln n}$, where $b > 0$ is a constant independent of $n$ and $T$.

Problem 7

This problem fills in some gaps in our proof sketch of strong linear programming duality.

(a) For this part, assume this version of Farkas’s Lemma, that given $A \in \mathbb{R}^{m \times n}$ and $b \in \mathbb{R}^m$, exactly one of the following statements holds: (i) there is an $x \in \mathbb{R}^n$ such that $Ax = b$ and $x \geq 0$; (ii) there is a $y \in \mathbb{R}^m$ such that $y^T A \geq 0$ and $y^T b < 0$.

\(^3\)Equivalently, when repeatedly flipping a coin that comes up “heads” with probability $\eta$, count the number of flips up to and including the first “heads.”
Deduce from this a second version of Farkas’s Lemma, stating that for $\mathbf{A}$ and $\mathbf{b}$ as above, exactly one of the following statements holds: (iii) there is an $\mathbf{x} \in \mathbb{R}^n$ such that $\mathbf{A}\mathbf{x} \leq \mathbf{b}$; (iv) there is a $\mathbf{y} \in \mathbb{R}^m$ such that $\mathbf{y} \geq 0$, $\mathbf{y}^T \mathbf{A} = 0$, and $\mathbf{y}^T \mathbf{b} < 0$.

[Hint: note the similarity between (i) and (iv). Also note that if (iv) has a solution, then it has a solution with $\mathbf{y}^T \mathbf{b} = -1$.]

(b) Use the second version of Farkas’s Lemma to prove the following version of strong LP duality: if the linear programs

$$\begin{align*}
\text{max } & \mathbf{c}^T \mathbf{x} \\
\text{subject to } & \mathbf{A}\mathbf{x} \leq \mathbf{b} \\
\text{with } & \mathbf{x} \text{ unrestricted}, \\
\text{and } & \text{min } \mathbf{b}^T \mathbf{y} \\
\text{subject to } & \mathbf{A}^T \mathbf{y} = \mathbf{c}, \mathbf{y} \geq 0
\end{align*}$$

are both feasible, then they have equal optimal objective function values.

[Hint: Let $\gamma^*$ denote the optimal objective function value of the dual linear program. Add the constraint $\mathbf{c}^T \mathbf{x} \geq \gamma^*$ to the primal linear program and use Farkas’s Lemma to show that the feasible region is non-empty.]