École Polytechnique Fédérale de Lausanne
Prof. F. Eisenbrand

March 29, 2019

Discrete Optimization (Spring 2019)

Assignment 6

Problem 1

Determine the dual program for the following linear programs:

1.
$$\begin{array}{rrcl}\max & 2x_1+3x_2-7x_3 & & \\ & x_1+3x_2+2x_3 & = & 4 \\ & x_1+x_2 & \leq & 8 \\ & x_1-x_3 & \geq & -15 \\ & x_1,x_2 & \geq & 0\end{array}$$

2.
$$\begin{array}{rrcr}\min & 3x_1+2x_2-3x_3+4x_4 & & \\ & 2x_1-2x_2+3x_3+4x_4 & \leq & 3 \\ & x_2+3x_3+4x_4 & \geq & -5 \\ & 2x_1-3x_2-7x_3-4x_4 & = & 2 \\ & x_1\geq 0 & & \\ & x_4\leq 0 & & \end{array}$$

Solution:

1.
$$\begin{array}{rrrrcr}\min & 4y_1 & +8y_2 & -15y_3 & & \\ & y_1 & +y_2 & +y_3 & \geq & 2 \\ & 3y_1 & +y_2 & & \geq & 3 \\ & 2y_1 & & -y_3 & = & -7 \\ & & y_2 & & \geq & 0 \\ & & & y_3 & \leq & 0\end{array}$$

2.
$$\begin{array}{rrcr}\max & 3y_1-5y_2+2y_3 & & \\ & 2y_1+2y_3 & \leq & 3 \\ & -2y_1+y_2-3y_3 & = & 2 \\ & 3y_1+3y_2-7y_3 & = & -3 \\ & y_1+y_2-y_3 & \geq & 1 \\ & y_1\leq 0 & & \\ & y_2\geq 0 & & \end{array}$$

Problem 2

In the setting of the matrix-game described in Section 5.1 of the lecture notes, show that for $A\in\mathbb{R}^{m\times n}$, one has
$$\max_i \min_j A(i,j) \leq \min_j \max_i A(i,j).$$

Solution:

Suppose $\max_i \min_j A(i,j) = A(e,f)$ and $\min_j \max_i A(i,j) = A(g,h)$ for some, not necessarily distinct, indices e, f, g and h.
Since $A(e,f)$ is the minimal entry of A in the e-th row, we get for any column j: $A(e,f) \leq A(e,j)$. In particular, this is true for the column h, which implies $A(e,f) \leq A(e,h)$.
Similarly, since $A(g,h)$ is the maximal entry of the column h, we get $A(g,h) \geq A(e,h)$. Combining these two inequalities leads to $A(e,f) \leq A(e,h) \leq A(g,h)$.

Problem 3

Consider the following linear program $\max\{c^T x : Ax \leq b\}$ and its dual $\min\{b^T y : A^T y = c, y \geq 0\}$. Suppose that both programs are bounded and feasible. Let x_0 and y_0 be feasible solutions of the primal, respectively the dual linear program. Show that the following are equivalent:

(i) x_0 and y_0 are optimal solutions of the primal, respectively the dual.

(ii) $c^T x_0 = b^T y_0$.

(iii) If a component of y_0 is positive, the corresponding inequality in $Ax \leq b$ is satisfied by x_0 with equality.

Solution:
The equivalence of (i) and (ii) follows directly from the theorem of strong duality (Theorem 5.2 in the lecture notes). To finish the proof, we show the equivalence of (ii) and (iii).
Since y_0 is feasible we have $c^T = y_0^T A$ and since x_0 is feasible we get $Ax_0 \leq b$. Combining these two identities gives:

$$c^T x_0 = y_0^T A x_0 \leq y_0^T b$$

It follows that (ii) is true if and only if $y_0^T A = y_0^T b$ which can be written as $y_0^T (b - Ax_0) = 0$. This is equivalent to saying that for each component with $(y_0)_i > 0$ we get $0 = (b - Ax_0)_i = b_i - a_i^T x_0$, where a_i is the i-th row of A. Thus (ii) is true if and only if (iii) is true.

Problem 4

For each of the following assertions, provide a proof or a counterexample.

(i) An index that has just left the basis B in the simplex algorithm cannot enter in the very next iteration.

(ii) An index that has just entered the basis B in the simplex algorithm cannot leave again in the very next iteration.

Solution:

(i) An index that has left the basis can enter in the very next iteration. An example is a triangle in the plane. Maybe the simplex method does not decide to walk to the neighboring optimal vertex in one step but makes a detour (while improving) via the other vertex. In this case, the inequality that has just left re-enters again.

(ii) We give a first proof that uses the fact that Simplex always chooses a direction that augments the objective function. Let B be a feasible basis and let Simplex move from B to $\tilde{B} = B \setminus \{i\} \cup \{j\}$, i.e., i leaves the basis and j enters it. Note that B and $\tilde{B}$ have $n-1$ common indices. Assume that j leaves the basis in the next iteration. Let d and $\tilde{d}$ be the directions that Simplex chooses to move from B to $\tilde{B}$ and away from $\tilde{B}$, respectively. Then, $d \cdot a_k = 0 = \tilde{d} \cdot a_k$ for $k \in B \setminus \{i\}$, *i.e.*, d and $\tilde{d}$ are perpendicular to each constraint that is in the basis except

a_i and a_j. Since the a_k ($k \in B \setminus \{i\}$) are $n-1$ linearly independent vectors and $d, \tilde{d} \in \mathbb{R}^n$ this means that $d \parallel \tilde{d}$. Since j entered the basis, this means that $a_j^{\mathrm{T}} d > 0$. Since j leaves the basis in the next step $a_j^{\mathrm{T}} \tilde{d} = -1$. Thus, $d = -\omega\, \tilde{d}$ for some constant $\omega > 0$. In particular, this means that the Simplex is moving in the opposite direction. Now, due to the choice of the direction in Simplex we know that $c^{\mathrm{T}} d > 0$ and $c^{\mathrm{T}} \tilde{d} > 0$. But, this yields a contradiction since $0 < c^{\mathrm{T}} \tilde{d} = -\omega\, c^{\mathrm{T}} d < 0$.

We give a second proof showing that λ_j is non-negative in the iteration succeeding the entrance of j into the basis, i.e., j cannot leave the basis in the next iteration. In the Simplex iteration at $\tilde{B}$ we have $\lambda^{\mathrm{T}} A_{\tilde{B}} = c^{\mathrm{T}}$. For ease of notation, assume that $\tilde{B} = \{a_1, \ldots, a_{n-1}, \tilde{a}\}$ where $\tilde{a} = a_j$. If $\lambda_n = 0$, then $\tilde{a}$ can not leave the basis in the next iteration (for this, λ_n would have to be strictly negative). Hence, we can assume $\lambda_n \neq 0$. Then, rewriting the above yields

$$\lambda_1 a_1 + \cdots + \lambda_{n-1} a_{n-1} + \lambda_n \tilde{a} = c \quad \Leftrightarrow \quad \frac{-\lambda_1}{\lambda_n} a_1 + \cdots + \frac{-\lambda_{n-1}}{\lambda_n} a_{n-1} + \frac{1}{\lambda_n} c = \tilde{a}.$$

Since j entered the basis we know $\tilde{a}^{\mathrm{T}} d > 0$ (d is still the direction from B to $\tilde{B}$). This gives

$$\begin{aligned} 0 < \tilde{a}^{\mathrm{T}} d &= \frac{-\lambda_1}{\lambda_n} a_1^{\mathrm{T}} d + \cdots + \frac{-\lambda_{n-1}}{\lambda_n} a_{n-1}^{\mathrm{T}} d + \frac{1}{\lambda_n} c^{\mathrm{T}} d \\ &= 0 + \cdots + 0 + \frac{1}{\lambda_n} c^{\mathrm{T}} d. \end{aligned}$$

Since $c^{\mathrm{T}} d > 0$ this implies $\lambda_n > 0$ which means that j can not leave the basis in the very next iteration.

Problem 5

We define two different norms on vectors. The infinity-norm is defined by $\|y\|_\infty = \max_i |y_i|$ and the 1-norm is defined by $\|y\|_1 = \sum_i |y_i|$.

Let A be an $m \times n$ matrix and let $b \in \mathbb{R}^m$ be a vector. Consider the problem of minimizing $\|Ax - b\|_\infty$ over all $x \in \mathbb{R}^n$.

Suppose that v is the optimal value of the problem.

(a) Let $p \in \mathbb{R}^m$ be a vector satisfying $\|p_i\|_1 \leq 1$ and $p^T A = 0$. Show that $p^T b \leq v$.

(b) To obtain the best possible lower bound of the form considered in (a), we construct the following linear program

$$\begin{array}{lrcl} \max & p^T b & & \\ & p^T A & = & 0 \\ & \sum_{i=1}^m |p_i| & \leq & 1. \end{array}$$

Using strong duality, show that the optimal solution of this problem is equal to v.

Solution:

(a) We have $p^T b = p^T b - (p^T A)x = p^T(b - Ax) = \sum_{i=1}^n p_i (b - Ax)_i$ for any $x \in R^n$, using the fact that $p^T A = 0$. Now we calculate:

$$p^T b = \sum_{i=1}^n p_i (b - Ax)_i \leq \sum_{i=1}^n |p_i| |(b - Ax)_i| \leq \sum_{i=1}^n |p_i| v = v \sum_{i=1}^n |p_i| = v \|p\|_i \leq v$$

Where the second inequality comes from the fact that $\|Ax - b\|_\infty = v$.

(b) We rewrite the problem of minimizing $\|Ax - b\|_\infty$ step by step to obtain an equivalent linear program, where a_i^T denotes the i-th row of A and $\mathbb{1} \in \mathbb{R}^m$ the all one vector.

$$\begin{aligned}
\min\{\|Ax - b\|_\infty : x \in \mathbb{R}^n\} &\Leftrightarrow \min\{v : \|Ax - b\|_\infty \leq v, v \geq 0\} \\
&\Leftrightarrow \min\{v : |(Ax - b)_i| \leq v \text{ for } i = 1, \dots, m \text{ and } v \geq 0\} \\
&\Leftrightarrow \min\{v : (Ax - b)_i \leq v, (Ax - b)_i \geq -v, v \geq 0\} \\
&\Leftrightarrow \min\{v : Ax - b \leq v \cdot \mathbb{1}, Ax - b \geq -v \cdot \mathbb{1}, v \geq 0\} \\
&\Leftrightarrow \min\{v : Ax - v \cdot \mathbb{1} \leq b, -Ax - v \cdot \mathbb{1} \leq -b, -v \leq 0\}
\end{aligned}$$

From the obtained linear program we take the dual linear program and try to bring it in the form we wished. We will split the variable y in three parts: y^+, y^- and s for the three sets of constraints given by $A^T x - v \cdot \mathbb{1} \leq b$, $-A^T x - v \cdot \mathbb{1} \leq -b$ respectively $-v \leq 0$.

$$\begin{aligned}
&\max\{-b^T y^+ + b^T y^- + 0s : A^T y^+ - A^T y^- = 0, -\mathbb{1}^T y^+ - \mathbb{1}^T y^- - s = -1, y^+, y^-, s \geq 0\} \\
\Leftrightarrow &\max\{b^T (y^- - y^+) : A^T (y^+ - y^-) = 0, -\mathbb{1}^T (y^+ + y^-) \geq -1, y^+, y^- \geq 0\} \\
\Leftrightarrow &\max\{b^T (y^- - y^+) : A^T (y^- - y^+) = 0, \sum_{i=1}^{m} (y_i^+ + y_i^-) \leq 1, y^+, y^- \geq 0\}
\end{aligned}$$

Setting $p = y^- - y^+$ we get that $1 \geq \sum_{i=1}^{n} (y_i^+ + y_i^-) \geq \sum_{i=1}^{m} |p_i|$ and equality is obtained exactly if at least one of the y_i^+, y_i^- is zero. This is for example possible if $y_i^+ = 0$, $y_i^- = p_i$ for each $p_i \geq 0$ and $y_i^+ = -p_i$, $y_i^- = 0$ for each $p_i < 0$. Thus the dual becomes:

$$\max\{b^T p : A^T p = 0, \sum_{i=1}^{m} |p_i| \leq 1\} \Leftrightarrow \max\{p^T b : p^T A = 0, \sum_{i=1}^{m} |p_i| \leq 1\}$$

Which is exactly the linear program considered.
Further, with strong duality we get that an optimal solution p of this linear program, is equal to the optimal solution of the primal linear program, which is equal to the optimal solution of the original problem.

Problem 6

Consider the following problem. We are given $B \in \mathbb{N}$, and a set of integer points

$$S = \{p \in \mathbb{Z}^n : \ 0 \leq p_i \leq B, \ \forall i = 1, \dots, n\},$$

whose points are all colored blue but one, which is red. We have an oracle that, given $i \in \{1, \dots, n\}$ and $\alpha \in \{0, \dots, B\}$, tells us whether there exists a red point $x^* \in S$ with $x_i^* \leq \alpha$. Give an algorithm to find the red point using $O(n \log(B))$ many oracle calls.

Solution:

We split the problem in n subproblems: for $i \in \{1, \dots, n\}$, we want to obtain the i-th component of the red point. This is a binary search problem, and we illustrate how to solve it for $i = 1$. We first call the oracle with $\alpha = \lfloor B/2 \rfloor$. If the answer is positive, we call the oracle with $\alpha = \lfloor B/4 \rfloor$. If it is negative, we know that the point is not in the interval $[0, \lfloor B/2 \rfloor]$, so we call the oracle with $\alpha = \lfloor 3B/4 \rfloor$. We continue the process in the same manner. In this way, we are guaranteed to find each component of the red point with $O(\log(B))$ oracle calls, for a total of $O(n \log(B))$ many calls.