HOMEWORK #8 SOLUTION

Yihua Guo (yhguo@umich.edu), Qi Chen (alfchen@umich.edu), Tianyi Ma (mtianyi@umich.edu)

1. Exercise 8.1 Task scheduling, continued

Problem: Consider again the "task scheduling" Exercise 2.4. Take the dual of the linear-optimization problem that you formulated. Explain how this dual can be interpreted as a kind of network problem. Using `AMPL`, solve the dual of the example that you created for Exercise 2.4 and interpret the solution.

Solution: Let $p_{i,j}$ represent the precedence matrix, i.e.

$$p_{i,j} = \begin{cases} 1, & \text{if job } j \text{ precedes job } i; \\ 0, & \text{otherwise.} \end{cases} \tag{1.1}$$

Then the task scheduling problem can be formulated mathematically as follows:

$$\text{(P')} \qquad \begin{array}{lrcll} \min & t_{n+1} & & & \\ & p_{i,j}(t_j + d_j) & \le & t_i, & \text{for } p_{i,j} = 1; \\ & t_0 & = & 0\ , & \\ & t_i & \ge & 0, & \forall i\ . \end{array}$$

Let m be the nunmber of pair (i, j) such that $p_{i,j} = 1$. Transfering (P') into the standard-form problem (P):

$$\text{(P)} \qquad \begin{array}{lrcll} \min & t_{n+1} & & & \\ & t_i - t_j - s_k & = & d_j, & \text{for } p_{i,j} = 1; \\ & t_0 & = & 0\ , & \\ & t_i & \ge & 0, & \forall i\ , \\ & s_k & \ge & 0, & \forall k\ , \end{array}$$

which is in standard form of the following

$$\text{(P)} \qquad \begin{array}{lrcl} \min & c'x & & \\ & Ax & = & b\ ; \\ & x & \ge & \mathbf{0}\ , \end{array}$$

where

$$x = \begin{pmatrix} t_0 & \cdots & t_n & t_{n+1} & s_1 & \cdots & s_m \end{pmatrix}',$$
$$c = \begin{pmatrix} 0 & \cdots & 0 & 1 & 0 & \cdots & 0 \end{pmatrix}',$$
$$b = \begin{pmatrix} b_1 & \cdots & b_m & 0 \end{pmatrix}',$$

A is a $(m+1) \times (n+m+2)$ matrix. The first m lines of A are parameters for m constraints for $p_{i,j} = 1$, and the last line of A is for $t_0 = 0$. The dual of (P) is

$$\text{(D)} \qquad \begin{array}{lrcl} \max & y'b & & \\ & y'A & \le & c'\ , \end{array}$$

Let

$$y = \begin{pmatrix} y_1 & \cdots & y_m & y_{m+1} \end{pmatrix}',$$

the objective function can be written as

$$\sum_{i=1}^{m} y_i b_i$$

Date: April 25, 2014.

$y'A \leq c'$ represents $(m+n+2)$ constraints. Considering the structure of matrix A, the first $(n+1)$ constraints have the following form (l denote the column number of matrix A starting from 0, which also corresponds to t_0 to t_n)

$$\sum_{(l,i) \text{ assoc. } j:p_{l,i}=1} y_j - \sum_{(i,l) \text{ assoc. } j:p_{i,l}=1} y_j \leq 0$$

For the $(n+2)$th constraint (which corresponds to the column of t_{n+1}):

$$\sum_{(n+1,i) \text{ assoc. } j:p_{n+1,i}=1} y_j - \sum_{(i,n+1) \text{ assoc. } j:p_{i,n+1}=1} y_j \leq 1$$

For last m constraints:

$$-y_i \leq 0.$$

Then the problem (D) is equivalent to

$$(\text{D'}) \quad \begin{array}{llclcll} \max & \sum_{i=1}^m y_i b_i & & & & & \\ & \sum_{(i,l) \text{ assoc. } j:p_{i,l}=1} y_j & - & \sum_{(l,i) \text{ assoc. } j:p_{l,i}=1} y_j & = & u_l \ , & l = 0,...,n, \\ & \sum_{(i,n+1) \text{ assoc. } j:p_{i,n+1}=1} y_j & - & \sum_{(n+1,i) \text{ assoc. } j:p_{n+1,i}=1} y_j & = & u_{n+1} - 1 \ , & \\ & & & y_i & \geq & 0, & \\ & & & u_j & \geq & 0. & \end{array}$$

Based on the task dependency graph, we can actually see that $\sum_{(i,l) \text{ assoc. } j:p_{i,l}=1} y_j$ can be viewed as all the outgoing net flow from node l, and $\sum_{(l,i) \text{ assoc. } j:p_{l,i}=1} y_j$ can be viewed as all the incoming net flow to node l. The u_l can be viewed as the net supply at node l. For node 0 to n, the outgoing net flow, minus the incoming net flow, is equal to the net supply. However, node $n+1$ absorbs the net flow by 1. Each dependency arc has the cost and the objective is to find the possible maximum cost for the flow. This is how the dual can be interpreted as a kind of network problem. The dual actually gives the critical path of the task scheduling problem, the primal. We will next show a concrete example.

Let $n = 5$, $m = 8$, $d(i = 0..5) = (0, 2, 1, 6, 4, 2)$,

$$(p_{i=0..6,j=0..6}) = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 \end{pmatrix}'.$$

The primal has the following constraints for $p_{i,j} = 1$ (in the same order for matrix A):

$$\begin{array}{ccccc} t_1 & - & t_0 & \geq & 0 \\ t_2 & - & t_1 & \geq & 2 \\ t_4 & - & t_1 & \geq & 2 \\ t_3 & - & t_2 & \geq & 1 \\ t_5 & - & t_3 & \geq & 6 \\ t_5 & - & t_4 & \geq & 4 \\ t_6 & - & t_5 & \geq & 2 \\ t_6 & - & t_4 & \geq & 4 \end{array}$$

Then the dual problem (D)

$$(\text{D}) \quad \begin{array}{lcccccccccc} \max & & 2y_2 & +2y_3 & +y_4 & +6y_5 & +4y_6 & +2y_7 & +4y_8 & & \\ & -y_1 & & & & & & & & \leq & 0, \\ & y_1 & -y_2 & -y_3 & & & & & & \leq & 0, \\ & & y_2 & & -y_4 & & & & & \leq & 0, \\ & & & & y_4 & -y_5 & & & & \leq & 0, \\ & & & y_3 & & & -y_6 & & -y_8 & \leq & 0, \\ & & & & & y_5 & +y_6 & -y_7 & & \leq & 0, \\ & & & & & & & y_7 & +y_8 & \leq & 1, \\ & & & & & & & & y_i & \geq & 0. \end{array}$$

Through `AMPL`, we get the optimal solution for (D): $\hat{y} = (0,1,0,1,1,0,1,0)'$, and the optimal objective value is 11. By examining the constraints, we can see that the '1's in the optimal solution correspond to the following task dependency edge: $(1 \to 2), (2 \to 3), (3 \to 5), (5 \to 6)$. By comparing this solution to the primal optimal solution, we see that the primal and the dual has the same optimal value, and the dual solution actually indicates the critical path of finishing the task. That is, it indicates the longest necessary path through the network of tasks.

2. Exercise 8.2 Pivoting and total unimodularity

Problem: A pivot in an $m \times n$ matrix A means choosing a row i and column j with $a_{ij} \neq 0$, subtracting $\frac{a_{kj}}{a_{ij}}$ times row i from all other rows $k (\neq i)$, and then dividing row i by a_{ij}. Note that after the pivot, column j becomes the i-th standard-unit column. Prove that if A is TU, then it is TU after a pivot.

Proof:

Let A_p be the matrix after applying the pivot to A by choosing a row i and column j with $a_{ij} \neq 0$. After the pivot, A^p will be something like this:

$$\begin{array}{cc} & \begin{array}{ccc} & j & \end{array} \\ i & \left(\begin{array}{c|c|c} & 0 & \\ \cdots & \vdots & \cdots \\ & 0 & \\ \hline \cdots & 1 & \cdots \\ \hline & 0 & \\ \cdots & \vdots & \cdots \\ & 0 & \end{array}\right) \end{array}$$

. So our goal is to prove that A^p is also a TU when A is TU.

$\because$ A is TU, and according to Theorem 8.6 (iii), appending standard-unit columns to A leaves A TU

$\therefore$ $A^1 = [A, \mathbf{I}_m]$ is TU

$\therefore$ by definition every square nonsingular submatrix B of A^1 has $det(B) = \pm 1$

$\therefore$ every square nonsingular submatrix B of A^1 is unimodular

$\because$ every basis matrix of A^1 is square nonsingular submatrix

$\therefore$ every basis matrix of A^1 is unimodular

Then we apply the a_{ij} pivot to A^1, and get $A^2 = \begin{pmatrix} A^p & D \end{pmatrix}$, where D will be something like this:

$$\begin{array}{cc} & \begin{array}{ccc} \phantom{\mathbf{I}_{i-1}} & i & \phantom{\mathbf{I}_{m-i}} \end{array} \\ i & \left(\begin{array}{c|c|c} \mathbf{I}_{i-1} & \vdots & 0 \\ \hline 0 & 1 & 0 \\ \hline 0 & \vdots & \mathbf{I}_{m-i} \end{array}\right) \end{array}.$$

$\because$ the pivot is a set of elementary row operations on the basis matrix of A^1, and each element of A is either 1 or (-1)

$\therefore$ $\frac{a_{kj}}{a_{ij}} = \pm 1$ and $a_{ij} = \pm 1$ in these elementary row operations

$\therefore$ the determinants of every basis matrix of A^1 and the corresponding basis matrix of A^2 using the same basis are ± 1

$\therefore$ every basis matrix of A^2 is also unimodular

We switch the j-th column of A^p and the i-th column of D in A^2. $A_j = e^i$, so D become $\mathbf{I}_m$, and A^p becomes $\begin{pmatrix} A_1 & \ldots & A_{j-1} & D_i & A_{j+1} & \ldots & A_n \end{pmatrix}$.

Now we consider the matrix $A^3 = \begin{pmatrix} A^q & \mathbf{I}_m \end{pmatrix}$, where $A^q = \begin{pmatrix} A_1 & \ldots & A_{j-1} & A_{j+1} & \ldots & A_n \end{pmatrix}$.

$\because$ A^3 consists of all columns in A^2 except D_i

$\therefore$ every basis matrix of A^3 is also basis matrix of A^2

$\therefore$ every basis matrix of A^3 is unimodular

$\therefore$ using Theorem 8.5, A^q is TU

$\therefore$ $A^{q1} = \begin{pmatrix} A_1 & \ldots & A_{j-1} & A_{j+1} & \ldots & A_n & e^i \end{pmatrix}$ is also TU using Theorem 8.6 (iii)

$\because$ $A_j = e^i$

$\therefore$ $A^{q1} = \begin{pmatrix} A_1 & \ldots & A_{j-1} & A_{j+1} & \ldots & A_n & A_j \end{pmatrix}$ is TU

Note that by switching columns of A^{q1}, we can get A^p. So if switching columns does not change the TU property, A^p is then proved to be TU. So next we prove that switching columns does not change the TU property.

For TU matrix $A = \begin{pmatrix} A_1 & \ldots & A_i & \ldots & A_j & \ldots & A_n \end{pmatrix}$, suppose that we switch column A_i and A_j and get A^1. For a square nonsingular submatrix B^1 of A^1, there are 3 cases:

(1) B^1 does not have any overlap with column i or j in A^1. So after the switching, B^1 also has the same determinant value as in A, thus $det(B^1) = \pm 1$;

(2) B^1 has overlap with one of column i or j in A^1. Let's say with column i in A^1. Suppose B^1 is a $r \times r$ matrix and column m of B^1 overlaps with column i of A^1, so B^1_m was in column j of A. So B^1 can be obtained by switching several columns of A's square submatrix $B = \begin{pmatrix} B^1_1 & ... & B^1_{m-1} & B^1_{m+1} & ... & B^1_r & B^1_m \end{pmatrix}$. Since B^1 is nonsingular and B just changes the order of columns compared to B^1, B is also nonsingular. Considering that A is TU, $det(B) = \pm 1$. So $det(B^1) = (-1)^s det(B) = (-1)^s(\pm 1) = \pm 1$, where s is the time of switching.

(3) B^1 has overlap with both column i and j in A^1. So B_1 can be obtained by switching two columns of a nonsingular submatrix B in A.Thus, $det(B^1) = \pm 1$.

Thus, for all square nonsingular matrix B^1 of A^1, $det(B^1) = \pm 1$, which means that all of them are unimodular. So we just prove that switching two columns of TU matrix A still results in a TU matrix.

Thus, after switching columns of TU matrix A^{q1}, we get A^p, which is also a TU matrix.

■

3. Exercise 8.3 Comparing formulations for a toy problem

Problem: Consider the systems:

$$\begin{aligned} S_1 : \quad 2x_1 + 2x_2 + x_3 + x_4 &\le 2; \\ x_j &\le 1; \\ -x_j &\le 0. \end{aligned} \tag{3.1}$$

$$\begin{aligned} S_2 : \quad x_1 + x_2 + x_3 &\le 1; \\ x_1 + x_2 + x_4 &\le 1; \\ -x_j &\le 0. \end{aligned} \tag{3.2}$$

$$\begin{aligned} S_3 : \quad x_1 + x_2 &\le 1; \\ x_1 + x_3 &\le 1; \\ x_1 + x_4 &\le 1; \\ x_2 + x_3 &\le 1; \\ x_2 + x_4 &\le 1; \\ -x_j &\le 0. \end{aligned} \tag{3.3}$$

Notice that each system has precisely the same set of integer solutions. Compare the feasible regions S_i of the continuous relaxations, for each pair of these systems. Specifically, for each choice of pair $i = j$, demonstrate whether or not the solution set of S_i is contained in the solution set of S_j. HINT: To prove that the solution set of Si is contained in the solution set of S_j, it suffices to demonstrate that every inequality of S_j is a nonnegative linear combination of the inequalities of S_i. To prove that the solution set of S_i is not contained in the solution set of S_j, it suffices to give a solution of S_i that is not a solution of S_j.

Solution: Comparing the feasible regions S_i of the continuous relaxations: (1) Comparing S_1 and S_2. $\forall x \in S_2$, we have

$$x_1 + x_2 + x_3 \le 1 \ , \ x_1 + x_2 + x_4 \le 1 \ , \ -x_j \le 0.$$

Then,

$$\begin{gathered} (x_1 + x_2 + x_3) + (x_1 + x_2 + x_4) \le 1 + 1, \\ (x_1 + x_2 + x_3) + (-x_2) + (-x_3) \le 1, \\ (x_1 + x_2 + x_3) + (-x_1) + (-x_3) \le 1, \\ (x_1 + x_2 + x_3) + (-x_1) + (-x_2) \le 1, \\ (x_1 + x_2 + x_4) + (-x_1) + (-x_2) \le 1, \\ -x_j \le 0, \end{gathered}$$

which are equivalent to

$$2x_1 + 2x_2 + x_3 + x_4 \le 2 \ , \ x_j \le 1 \ , \ -x_j \le 0.$$

So $x \in S_1$, and $S_2 \subset S_1$. S_1 is not contained in S_2 because $(0, 0.5, 1, 0)' \in S_1$ but $(0, 0.5, 1, 0)' \notin S_2$.

(2) Comparing S_2 and S_3. $\forall x \in S_2$, we have

$$x_1 + x_2 + x_3 \le 1 \ , \ x_1 + x_2 + x_4 \le 1 \ , \ -x_j \le 0.$$

Then,

$$(x_1 + x_2 + x_3) + (-x_3) \le 1,$$

$$(x_1 + x_2 + x_3) + (-x_2) \leq 1,$$
$$(x_1 + x_2 + x_4) + (-x_2) \leq 1,$$
$$(x_1 + x_2 + x_3) + (-x_1) \leq 1,$$
$$(x_1 + x_2 + x_4) + (-x_1) \leq 1,$$
$$-x_j \leq 0,$$

which are equivalent to

$$x_1 + x_2 \leq 1\ ,\ x_1 + x_3 \leq 1\ ,\ x_1 + x_4 \leq 1\ ,\ x_2 + x_3 \leq 1\ ,\ x_2 + x_4 \leq 1\ ,\ -x_j \leq 0.$$

So $x \in S_3$, and $S_2 \subset S_3$. S_3 is not contained in S_2 because $(0.5, 0.5, 0.5, 0.5)' \in S_3$ but $(0.5, 0.5, 0.5, 0.5)' \notin S_2$.

(3) Comparing S_1 and S_3. S_3 is not contained in S_1 because $(0.5, 0.5, 0.5, 0.5)' \in S_3$ but $(0.5, 0.5, 0.5, 0.5)' \notin S_1$. S_1 is not contained in S_3 because $(0, 0.5, 1, 0)' \in S_1$ but $(0, 0.5, 1, 0)' \notin S_3$.

4. Exercise 8.4 Comparing facility-location formulations

Problem: We have seen two formulations of the forcing constraints for the uncapacitated facility-location problem. We have a choice of the mn constraints: $y_i + x_{ij} \leq 0$, for $i = 1, ..., m$ and $j = 1, ..., n$, or the m constraints: $ny_i + nx_{ij} \leq 0$, for $i = 1, ..., m$. Which formulation is stronger? That is, compare (both *computationally* and *analytically*) the strength of the two associated continuous relaxations (i.e., when we relax $y_i \in \{0, 1\}$ to $0 \leq y_i \leq 1$, for $i = 1, ..., m$). In Appendix A.3, there is AMPL code for trying computational experiments.

Solution: Consider the choice of the mn constraints:

$$S_1 : -y_i + x_{ij} \leq 0, \text{for } i = 1, ..., m \text{ and } j = 1, ..., n,$$

and the m constraints:

$$S_2 : -ny_i + \sum_{j=1}^{n} x_{ij} \leq 0, \text{for } i = 1, ..., m.$$

Using `AMPL`, we first random generate parameters for the uncapacitated facility-location problem and using this set of parameters to compare the two set of constraints with integer constraint and continuous relaxations. We omit the parameters here because of limited space.

For integer programming, S_1 and S_2 give the same optimal solution with total cost of 456 ($y = (1, 0, 1, 0, 1, 0, 1, 0, 1, 0)'$). x are also exactly the same (we don't show x here as its dimension is large). When considering the continuous relaxation, S_1 still gives the same optimal solution, while S_2 gives a different optimal solution with the total cost of 301.76 ($y = (0.2, 0.12, 0.16, 0.12, 0.16, 0.04, 0.08, 0.08, 0.04, 0)'$), which is lower than the optimal cost for integer programming. From this example, we see that the formulation of S_1 is stronger.

We next prove it analytically. $\forall (x, y) \in S_1$,

$$-y_i + x_{ij} \leq 0, \text{for } i = 1, ..., m \text{ and } j = 1, ..., n,$$

then,

$$\sum_{j=1}^{n} (-y_i + x_{ij}) \leq 0, \text{for } i = 1, ..., m,$$

which is equivalent to

$$-ny_i + \sum_{j=1}^{n} x_{ij} \leq 0, \text{for } i = 1, ..., m.$$

Thus $(x, y) \in S_2$, so $S_1 \subset S_2$. However, S_2 is not contained in S_1. It's possible to construct an feasible solution (x, y) for S_2, such that,

$$y_1 = \frac{1}{n}, x_{1j} = 0, \text{for } j = 1, ..., n-1, x_{1n} = \frac{2}{n},$$
$$y_2 = 1, x_{2j} = 1, \text{for } j = 1, ..., n-1, x_{2n} = 1 - \frac{2}{n},$$

and other variables is equal to zero. It is obvious that this $(x, y) \notin S_1$ $(-y_1 + x_{1n} = 1/n > 0)$. So the formulation of S_1 is stronger.

5. Exercise 8.5 Comparing piecewise-linear formulations

Problem: We have seen that the adjacency condition for piecewise-linear univariate functions can be modeled by

$$\begin{aligned}&\lambda_1 \le y_1;\\ &\lambda_j \le y_{j-1}+y_j, \quad \text{for } j=2,...,n-1;\\ &\lambda_n \le y_{n-1}.\end{aligned} \tag{5.1}$$

An alternative formulation is

$$\begin{aligned}&\textstyle\sum_{i=1}^{j} y_i \le \sum_{i=1}^{j+1}\lambda_i, \quad \text{for} j=1,...,n-2;\\ &\textstyle\sum_{i=j}^{n-1} y_i \le \sum_{i=j}^{n}\lambda_i, \quad \text{for} j=2,...,n-1.\end{aligned} \tag{5.2}$$

Explain why this alternative formulation is valid, and compare its strength to the original formulation, when we relax $y_i \in \{0,1\}$ to $0 \le y_i \le 1$, for $i=1,...,n1$. (Note that for both formulations, we require $\lambda_i \ge 0$, for $i=1,...,n$, $\sum_{i=1}^{n}\lambda_i = 1$, and $\sum_{i=1}^{n-1} y_i = 1$).

Solution:

When we have $y_i \in \{0,1\}$, the alternative formulation (5.2) is equivalent to formulation (5.1). In (5.1), the formulation means that if y_k is 1, for some k ($1 \le k \le n$), and necessarily all of the other y_j are 0, then only λ_k and λ_{k+1} can be positive. It is easy to see that in (5.2), the meaning is the same: if y_k is 1, $\sum_{i=1}^{k} y_i = 1 \le \sum_{i=1}^{k+1}\lambda_i$, and $\sum_{i=k}^{n-1} y_i = 1 \le \sum_{i=k}^{n}\lambda_i$, so considering that $\sum_{i=1}^{n}\lambda_i = 1$, thus $\sum_{i=1}^{k+1}\lambda_i = \sum_{i=k}^{n}\lambda_i = 1$, thus $\lambda_k + \lambda_{k+1} = \sum_{i=1}^{k+1}\lambda_i - (\sum_{i=1}^{n}\lambda_i - \sum_{i=k}^{n}\lambda_i) = 1-(1-1) = 1$, thus only λ_k and λ_{k+1} can be positive.

After continuous relaxations, the alternative formulation (5.2) and formulation (5.1) are no longer equivalent. In fact, the solution set of (5.2) is contained in the solution set of (5.1). Next, we will first prove that every inequality of (5.1) is a nonnegative linear combination of the inequalities of (5.2), and there exists a solution of (5.1) that is not a solution of (5.2).

We first prove that every inequality of (5.1) is a nonnegative linear combination of the inequalities of (5.2).

For the inequality $\lambda_1 \le y_1$ of (5.1), we let $j=2$ in the 2nd inequality of (5.2), so we have:
$\sum_{i=2}^{n-1} y_i \le \sum_{i=2}^{n}\lambda_i$
$\Rightarrow \sum_{i=1}^{n-1} y_i - y_1 \le \sum_{i=1}^{n}\lambda_i - \lambda_1$
$\Rightarrow 1 - y_1 \le 1 - \lambda_1$
$\Rightarrow y_1 \ge \lambda_1$

For the inequality $\lambda_n \le y_{n-1}$ of (5.1), we let $j=n-2$ in the 1st inequality of (5.2), so we have:
$\sum_{i=1}^{n-2} y_i \le \sum_{i=1}^{n-1}\lambda_i$
$\Rightarrow \sum_{i=1}^{n-1} y_i - y_{n-1} \le \sum_{i=1}^{n}\lambda_i - \lambda_n$
$\Rightarrow 1 - y_{n-1} \le 1 - \lambda_n$
$\Rightarrow y_{n-1} \ge \lambda_n$

For the inequality $\lambda_j \le y_{j-1}+y_j,$ for $j=2,...,n-1$ of (5.1), we reform the 1st inequality of (5.2) to $\sum_{i=1}^{j-2} y_i \le \sum_{i=1}^{j-1}\lambda_i, \text{for} j=3,...,n$ and the 2nd inequality of (5.2) to $\sum_{i=j+1}^{n-1} y_i \le \sum_{i=j+1}^{n}\lambda_i, \text{for} j=1,...,n-2$, and add them as follows:
$\sum_{i=1}^{j-2} y_i + \sum_{i=j+1}^{n-1} y_i \le \sum_{i=1}^{j-1}\lambda_i + \sum_{i=j+1}^{n}\lambda_i$
$\Rightarrow \sum_{i=1}^{n-1} y_i - y_{j-1} - y_j \le \sum_{i=1}^{n}\lambda_i - \lambda_j$
$\Rightarrow 1 - y_{j-1} - y_j \le 1 - \lambda_j$
$\Rightarrow y_{j-1} + y_j \ge \lambda_j$

Thus, every inequality of (5.1) is a nonnegative linear combination of the inequalities of (5.2).

Next, we prove that there exists a solution of (5.1) that is not a solution of (5.2).

To construct such situation of (5.1), we choose $y_j = \frac{1}{n-1}, j = 1...n-1$. Then for $\lambda_j, j=1...n$, we assign $\lambda_n = \frac{1}{n-1}$, and $\lambda_j = \frac{2}{n-1}$ for $j=n-1$, $j=n-2$,... until $j=m$ in which $(\sum_{j=n-1}^{m}\frac{2}{n-1}) + \frac{1}{n-1} = 1$. After than, for $j=1...m-1$, $\lambda_n = 0$. This solution satisfies (5.1). By constructing the solution in this way, when $n \ge 4$, one can always find $y_1 \le \sum_{i=1}^{2}\lambda_i$, which does not satisfy the 1st equation of (5.2) when $j=1$.

To see that, let's take a look at the case when $n=4$. Table 1 show the y_j and λ_j of our constructed solution for (5.1) when $n=4$. It is clear to see that this solution satisfies (5.1), but violates $\sum_{i=1}^{j} y_i \le \sum_{i=1}^{j+1}\lambda_i$ when $j=1$ since $y_1 = 1/3 \ge 0 = \sum_{i=1}^{2}\lambda_i$.

Thus, we just prove that every inequality of (5.1) is a nonnegative linear combination of the inequalities of (5.2) and there exists a solution of (5.1) that is not a solution of (5.2), which proves that the solution set of (5.2) is contained in the solution set of (5.1) when we relax $y_i \in \{0,1\}$ to $0 \le y_i \le 1$, for $i=1,...,n1$.

■

j	y_j	λ_j
1	1/3	0
2	1/3	0
3	1/3	2/3
4	/	1/3

TABLE 1. y_j and λ_j in the constructed solution for (5.1) when $n = 4$

6. EXERCISE 8.6 GOMORY CUTS

Problem: Consider the standard form problem having

$$A := \begin{pmatrix} 7 & 1 & 1 & 0 & 0 \\ -1 & 3 & 0 & 1 & 0 \\ -8 & -9 & 0 & 0 & 1 \end{pmatrix},\ b := \begin{pmatrix} 28 \\ 7 \\ -32 \end{pmatrix},\ c := (-2, -1, 0, 0, 0)'.$$

An optimal basis is $\beta := (1, 2, 5)$, $\eta := (3, 4)$. Suppose that all of the variables are constrained to be integer. Derive all (Gom) cuts for this basis and compare their strength on the continuous relaxation of the feasible region. HINT: View everything in the space of the variables x_η.

Can you get a stronger (GMI) cut (with respect to this basis) by ignoring the integrality on some variables?

Solution: The Gom cuts for the given basis are:

$$\frac{3}{22}x_3 + \frac{21}{22}x_4 \geq 0.5,$$
$$\frac{1}{22}x_3 + \frac{7}{22}x_4 \geq 0.5,$$
$$\frac{1}{2}x_3 + \frac{1}{2}x_4 \geq 0.5.$$

They strengthen the feasible region for continuous relaxation. The comparison is shown in Figure 1 and Figure 2.

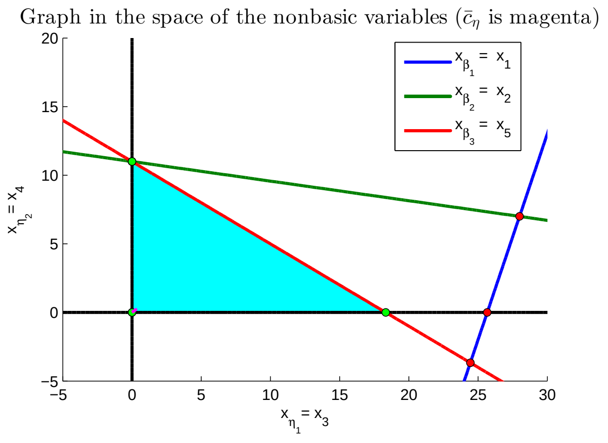

FIGURE 1. Feasible region projected into the space of (x_3, x_4) (original)

7. EXERCISE 8.7 MAKE AMENDS

Problem: Find an *interesting applied problem*, model it as a pure- or mixed-*integer* linear-optimization problem, and test your model with AMPL. Credit will be given for *deft* modeling, *sophisticated* use of AMPL, testing on meaningfully-*large* instances, and *insightful* analysis. Try to play with CPLEX *integer* solver options (they can be set through AMPL) to get better behavior of the solver.

Solution:

We construct the following problem as an example. Suppose Martin wants to make money and he decides to open several restaurants in the city he lives. There are 4 types of restaurants,

- Japanese restaurant needs 20 employees each and has a profit of \$ 420.5 per day;
- Chinese restaurant needs 17 employees each and has a profit of \$ 325.6 per day;
- Italian restaurant needs 27 employees each and has a profit of \$ 507.2 per day;
- French restaurant needs 30 employees each and has a profit of \$ 486.3 per day;

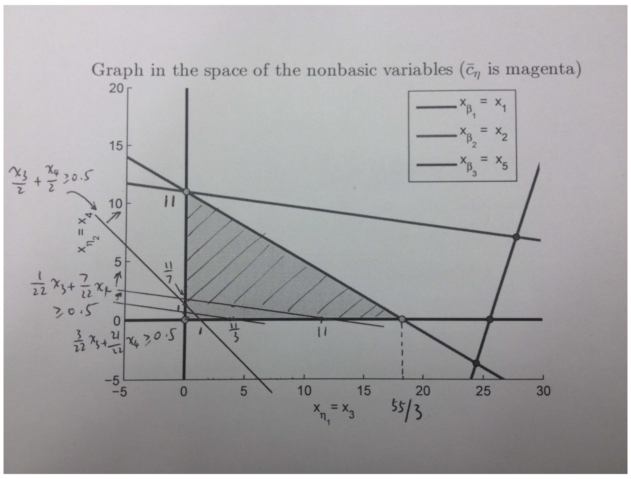

FIGURE 2. Feasible region projected into the space of (x_3, x_4) (Gom cuts)

Martin would like to open at least 77 restaurants but he only has 813 people to serve in restaurants. For unemployed people, he needs to pay \$ 5.7 per person. The goal is to maximize the profit of all restaurants.

This is an optimization problem and can be formalized as follows:

$$
\begin{array}{lr}
\max & 420.5x_1 + 325.6x_2 + 507.2x_3 + 486.3x_4 \\
\text{s.t.} & x_1 + x_2 + x_3 + x_4 \geq 77; \\
& 20x_1 + 17x_2 + 27x_3 + 30x_4 \leq 813; \\
& x_1 \geq 0 \text{ and integer}; \\
& x_2 \geq 0 \text{ and integer}; \\
& x_3 \geq 0 \text{ and integer}; \\
& x_4 \geq 0 \text{ and integer}.
\end{array}
\tag{7.1}
$$

The AMPL file ex8_7.mod is attached at follows,

```
# ex8_7.mod
#
param n integer > 0; # NUMBER OF COLUMNS
set COLS := {1..n}; # DEFINING THE SET OF COLUMNS INDICES

param employeenum;
param restaurantnum;
param wage0;
param profit {COLS};
param wage {COLS};
param employeenumperrestaurant {COLS};

var x {j in COLS} >= 0; # DEFINING THE (NONNEGATIVE) VARIABLES

maximize totalprofit:
sum {j in COLS} (profit[j] * x [j] - wage[j]*x[j]*employeenumperrestaurant[j]) -  (employeenum - sum {j in COLS}

subject to Constraints_resnum:
sum {j in COLS} x[j] <= restaurantnum;


subject to Constraints_empnum:
sum {j in COLS} employeenumperrestaurant[j] * x[j] <= employeenum;
```

The AMPL file ex8_7.dat is attached at follows,

```
# ex8_7.dat
#
param n := 4;
param employeenum := 813;
param restaurantnum := 77;
param wage0 := 5.7;
param profit :=
  1 420.5
  2 325.6
  3 507.2
  4 486.3;

param wage :=
  1 12.5
  2 20.6
  3 17.2
  4 22.3;

param employeenumperrestaurant :=
  1 20
  2 17
  3 27
  4 30;
```

The AMPL file ex8_7.run is attached at follows,

```
# ex8_7.run
#
reset; # CLEAR FROM ANY PREVIOUS SOLVE

option show_stats 1; # SET AMPL OPTION TO CONTROL OUTPUT
option presolve 10; # COMMENT OUT TO SOLVE BIGGER MODELS
option relax_integrality 0; # COMMENT OUT TO SOLVE BIGGER MODELS
option solver cplex; # AMPL OPTION TO CHOOSE AN AVAILABLE SOLVER
#option cplex_options 'display=2 presolve=0'; # SETTING SOLVER OPTIONS
#option cplex_options 'relax_integrality=0 presolve=10'; # SETTING SOLVER OPTIONS
       # DELETING "presolve=0" MAY HELP TO SOLVE BIGGER MODELS

model ex8_7.mod; # READ IN THE MODEL
data ex8_7.dat; # READ IN THE DATA AND INSTANTIATE THE MODEL
solve;  # PASS THE INSTANCE TO THE SOLVER

display totalprofit > ex8_7.out; # WRITE THE OPTIMAL OBJECTIVE VALUE TO A FILE
                      # NOTE THAT '>' DELETES THE FILE FIRST, IF IT EXISTS
for {j in COLS}
  { # USING CSTYLE COMMANDS TO OUTPUT OPTIMAL SOLUTION TO FILE
    printf "x(%i) = %f\n", j, x[j] >> ex8_7.out;
                    # NOTE THAT '>>' APPENDS TO THE FILE, IF IT EXISTS
  };
```