Department of Mathematics and Computer Science
University of Southern Denmark, Odense

March 11, 2024
Marco Chiarandini

DM545/DM871 – Linear and integer programming

Sheet 5, Spring 2024

Exercises with the symbol $^+$ are to be done at home before the class. Exercises with the symbol * will be tackled in class and should be at least read at home. The remaining exercises are left for self training after the exercise class.

Solution:
Included. Note that for some exercises explanations of the models are not given. In the exam tests, however, explanations are expected in answers to the questions.

Exercise 1$^+$

Formulate the satisfiability problem as an ILP. Give the precise formulation for the following instance:

$$(x_1 \vee x_2 \vee x_3) \wedge (x_1 \vee \neg x_2) \wedge (x_2 \vee \neg x_3) \vee (x_3 \vee \neg x_1) \wedge (\neg x_1 \vee \neg x_2 \vee \neg x_3)$$

Solution:
Satisfiability problems ask to determine the truth assignment of variables of a formula in propositional logic. Each formula in propositional logic can be put in Conjunctive Normal Form:

$$\bigwedge_{j=1}^{m} \left(\bigvee_{i=1}^{n} x_i \right)$$

that is, conjunction of disjunction clauses. The formula in the problem resembleis a CNF except that it is a disjunction of two CNF formulas. Let's first model the feasibility region of each of the two CNF parts.

$$X_1 = \left\{ \begin{array}{rcl} x_1 + x_2 + x_3 & \geq & 1 \\ x_1 + (1 - x_2) & \geq & 1 \\ x_2 + (1 - x_3) & \geq & 1 \end{array} \middle| \, x_1, x_2, x_3 \in \{0, 1\} \right\}$$

$$X_2 = \left\{ \begin{array}{rcl} (1 - x_1) + x_3 & \geq & 1 \\ (1 - x_1) + (1 - x_2) + (1 - x_3) & \geq & 1 \end{array} \middle| \, x_1, x_2, x_3 \in \{0, 1\} \right\}$$

We can then model the whole formula as a disjunction between two feasibility regions (see slide 44 of lec08):

$$\begin{array}{lll} \max & 1 & \\ & x_1 + x_2 + x_3 & \geq 1 - y \\ & x_1 + (1 - x_2) & \geq 1 - y \\ & x_2 + (1 - x_3) & \geq 1 - y \\ & (1 - x_1) + x_3 & \geq 1 - (1 - y) \\ & (1 - x_1) + (1 - x_2) + (1 - x_3) & \geq 1 - (1 - y) \\ & x_1, x_2, x_3 \in \{0, 1\} & \\ & y \in \{0, 1\} & \end{array}$$

We could have also used the fact that every propositional formula can be put in CNF form. We can use logical equivalences. We can use the distributive law:

$$p \vee (q \wedge r) \equiv (p \vee q) \wedge (p \vee r)$$

Substituting the clauses in the original formula with new variables we have:

$$(p \wedge q \wedge r) \vee (t \wedge v)$$

and again in the same way:

$$(p \wedge q \wedge r) \vee z$$

thus:

$$(p \vee z) \wedge (q \vee z) \wedge (r \vee z)$$

$$(p \vee (t \wedge v)) \wedge (q \vee (t \wedge v)) \wedge (r \vee (t \wedge v))$$

Using the distributive law again...

Exercise 2*

(The following is exercise 6 from exam 2010) A car rental company at the beginning of each month wants to have a certain number of cars in each of the towns in which it operates. For the towns $A, B, C, \dots, G$ the number of cars desired is $30, 40, 55, 60, 80, 40, 55$, respectively. At the end of the current month there are instead in the stations in these towns $65, 90, 95, 15, 60, 10, 25$ cars, respectively. To move one car from one station to the other causes a cost that we may assume proportional to the distance between the two stations. The table indicates the distances (in hundreds of kilometers) between every city pair of stations.

from .. to..	D	E	F	G
A	5	6	10	9
B	9	11	9	15
C	12	10	14	15

Formulate the problem of deciding the cars to move while minimizing the costs in mathematical programming terms.

Solution:

Let $I = \{A, B, C, D, E, F, G\}$ be the set of towns and let f_i and d_i for $i \in I$ be the current and desired number of cars, respectively.

We can calculate $b_i = f_i - d_i$ for all i. We get: $\mathbf{b} = [65 - 30 = 35, 50, 40, -45, -20, -30, -30]$. Thus, we can form two groups of cities, those that have cars in excess $\mathcal{E} = \{A, B, C\}$ and those with lack of cars $\mathcal{L} = \{D, E, F, G\}$. We note that the total number of cars in excess is equal to the total number of lacking cars.

We can construct a bipartite graph with one node for each station in $\mathcal{E}$ in one partition and with one node for each station in $\mathcal{L}$ for the other partition. There are arcs from all nodes in $\mathcal{E}$ to all nodes in $\mathcal{L}$. Each arcs has associated a cost given by the table above. Let $x_{ij}, i \in \mathcal{E}, j \in \mathcal{L}$, be nonnegative integer variables that determine how many cars are to be moved from town $i \in \mathcal{E}$ to town $j \in \mathcal{L}$. The IP formulation of the problem is the following:

$$\min \sum_{i \in \mathcal{E}} \sum_{j \in \mathcal{L}} c_{ij} x_{ij} \tag{1}$$

$$\sum_{j \in \mathcal{L}} x_{ij} = b_i \qquad \forall i \in \mathcal{E} \tag{2}$$

$$\sum_{j \in \mathcal{E}} x_{ji} = -b_i \qquad \forall i \in \mathcal{L} \tag{3}$$

$$x_{ij} \geq 0 \text{ and integer} \qquad \forall i \in \mathcal{E}, j \in \mathcal{L} \tag{4}$$

Cars are moved from stations with positive excess (2) to stations with positive lack (3) and the objective function minimizes the total cost. A more general model without need for the creation of the two sets $\mathcal{E}$ and $\mathcal{L}$ can be formed as follows:

$$\min \sum_{i \in I} \sum_{j \in I} c_{ij} x_{ij} \tag{5}$$

$$\sum_{j \in I} x_{ij} - \sum_{j \in I} x_{ji} = f_i - d_i \qquad \forall i \in I \tag{6}$$

$$x_{ij} \geq 0 \text{ and integer} \qquad \forall i, j \in I \tag{7}$$

Constraints (6) ensure that the flow of cars is balanced. Cars are moved from towns with positive balance to towns with negative balance. Constraints (7) describe the domain of the variables. The objective minimizes the total cost of the movement.
The model (5)-(7) has n^2 variables and n constraints, where n is the number of towns.
Later in this course we will see that this problem is actually an instance of the *min cost flow problem* and it can therefore be solved by any direct algorithm for this problem. Alternatively, if all input data in the model are integer numbers, the solution to the linear relaxation of the problem will be integral because the constraint matrix is total unimodular. Hence, we could relax the integrality constraint.

Exercise 3*

Formulate the following manpower planning problem in mathematical programming terms.
Given a set of workers and a set of 15 working hours per day with a required staffing per hour. Determine the minimum number of people required to cover the workload requirements, knowing that a person works in shifts that cover 7 hours and a person starting in hour i contributes to the workload in hours $i, \dots, i+6$ (eg: a person starting in hour 3 contributes to the workload in hours 3,4,5,6,7,8,9).

Solution:

Decision Variables:

- $x_i \in \mathbb{N}_0$: number of people starting work in hour $i (i = 1, \dots, 15)$

Objective Function:

$$\min \sum_{i=1}^{15} x_i$$

Constraints:

- Demand:

$$\sum_{i=t-6}^{i=t} x_i \geq d_t \text{ for } t = 1, \dots, 15$$

- Bounds:

$$x_{-5}, \dots, x_0 = 0$$

It is a generalized set covering problem.

Exercise 4* Cutting-Stock Problem

Materials such as paper, textiles, cellophane, and metallic foil are manufactured in rolls of large widths. These rolls, referred to as *raws*, are later cut into rolls of small width, called *finals*. Each manufacturer produces raws of a few standard widths; the widths of the finals are specified by different customers and may vary widely. The cutting is done on machines by knives that slice through the rolls in much the same way as a knife slices bread. For example, a raw that is 100 cm wide may be cut into two finals with 31 cm width and one final with 36 cm width, with the 2 cm left over going to waste. When a complicated summary of orders has to be filled, the most economical way of cutting the existing raws into the desired finals is rarely obvious. The problem of finding such a way is known as the *cutting-stock problem*.
Model the problem in mathematical programming terms.

Solution:

We have at disposal an infinite number of raw stocks each of width L. We want to cut pieces of type i, each having width w_i and demand b_i. We want to satisfy the demands using the least possible raw stocks.
For example, let $L = 22$ and $w_1 = 5$, $b_1 = 7$ and $w_2 = 3$, $b_2 = 3$. Then, a feasible solution would be:

	type 1	type 2
raw stock 1 :	4	0
raw stock 2 :	0	7
raw stock 3 :	4	0

Let u_i, $i = 1, 2, 3$ be binary variables that indicate whether the raw stock i is used. Let x_{ij} be integer variables that indicate how many pieces of each type j are cut in the raw stock i.
For our small example, the model is:

$$\begin{array}{ll} \min & u_1 + u_2 + u_3 \\ & 5x_{11} + 3x_{12} \leq 22u_1 \\ & 5x_{21} + 3x_{22} \leq 22u_2 \\ & 5x_{31} + 3x_{32} \leq 22u_3 \\ & x_{11} + x_{21} + x_{31} \geq 7 \\ & x_{12} + x_{22} + x_{32} \geq 3 \\ & u_i \in \{0, 1\} \\ & x_{ij} \in \mathbb{Z}^+ \end{array}$$

An alternative formulation is possible by generating all feasible patterns for cutting a raw stock. For example:

$$\begin{array}{c|ccccc} & P_1 & P_2 & P_3 & P_4 & P_5 \\ \hline \text{type 1} & 4 & 0 & 1 & 2 & 3 \\ \text{type 2} & 0 & 7 & 5 & 4 & 2 \end{array}$$

Then the problem can be modeled by using one integer variable for each patter indicating how many raw stock must be cut with that pattern:

$$\begin{array}{ll} \min & \lambda_1 + \lambda_2 + \lambda_3 + \lambda_4 + \lambda_5 \\ & 4\lambda_1 + 0\lambda_2 + 1\lambda_3 + 2\lambda_4 + 3\lambda_5 \geq 7 \\ & 0\lambda_1 + 7\lambda_2 + 5\lambda_3 + 4\lambda_4 + 2\lambda_5 \geq 3 \\ & \lambda_1, \lambda_2, \lambda_3, \lambda_4, \lambda_5 \in \mathbb{Z}^+ \end{array}$$

This second formulation has an exponential number of variables. That is why it is usually generated by the computer. In delayed column generation, one does not introduce all variabels in the initial model rather they are introduced iteratively during the solutions process. The second formulation, in spite the exponential number of variables is often preferred because it provides a tighether formulation, ie, the lower bound found by linear relaxation is better for the second formulation than for the first one.

Exercise 5*

Write the following logical conditions as linear conditions on 0-1 variables:

$$X_1 \vee X_2$$
$$X_1 \rightarrow X_2$$
$$X_1 \wedge X_2$$
$$X_1 \leftrightarrow X_2$$
$$\neg X_1$$
$$X_1 \oplus X_2 \text{ xor}$$

Solution:

See Section 9.2 of chapter 9 from the book by Williams uploaded in ItsLearning, under Reading Material.
See also: `http://yetanothermathprogrammingconsultant.blogspot.dk/2016/02/xor-as-linear-inequalities.html`

Exercise 6*

Model by means of MIP the following constraint:

- $z = xy$, where $x, y \in \mathbb{B}$.

Solution:

z will be also a binary variable.

$$\begin{aligned} z &\geq x + y - 1 \\ z &\leq x \\ z &\leq y \end{aligned}$$

Verify it.

Exercise 7

Consider the following mathematical programming problem:

$$\max \left\{ \sum_{i,j:i<j} c_{ij} z_i z_j : z \in B^n \right\}$$

What type of programming problem is it? Put it in a form that we can solve with the methods seen so far in the course.

Solution:
This is similar to the problem above. We introduce z_{ij}. Since the problem is a maximization problem then we impose:

$$\begin{aligned} \max \ & \textstyle\sum_{i,j:i<j} c_{ij} z_{ij} \\ & z_{ij} \leq z_i \\ & z_{ij} \leq z_j \\ & z_i, z_j \in \{0,1\} \\ & z_{ij} \in \{0,1\} \end{aligned}$$

If it was a minimization problem then we would need :

$$\begin{aligned} \min \ & \textstyle\sum_{ij} c_{ij} z_{ij} \\ & z_{ij} \geq z_i + z_j - 1 \\ & z_{ij} \leq z_i \\ & z_{ij} \leq z_j \\ & z_i, z_j \in \{0,1\} \\ & z_{ij} \in \{0,1\} \end{aligned}$$

Exercise 8*

Reformulate the following problem into a 0-1 ILP:

$$\begin{aligned} \max x_1^3 + x_2^5 + x_3^2 + 5x_1x_2x_3^4 - 2x_1x_2 \\ 7x_1x_2^2 + 3x_1^2x_2 - 5x_1x_2 \geq 0 \\ -2x_1 + 3x_2 + x_3 \leq 3 \\ x_i \in \{0,1\}, i = 1,2,3 \end{aligned}$$

Solution:
$5x_1x_2x_3^4$ can be linearized by introducing the variable y_{123} and imposing:

$$\begin{aligned} y_{123} &\leq x_1 \\ y_{123} &\leq x_2 \\ y_{123} &\leq x_3 \\ y_{123} &\geq x_1 + x_2 + x_3 - 2 \end{aligned}$$

Exercise 9*

Show that the maximum matching in arbitrary graphs is a special sort of set packing problem.

Solution:

The set packing problem asks to find a maximum size subset of the columns of a matrix in such a way that no row is covered more than once. The maximum matching asks to find the maximum size subset of the set of edges such that no vertex is covered more than once. Hence, in the matching problem the incidence matrix has the same role as in the matrix of the set packing. It is a special sort of set packing because the incidence matrix has exactly two 1s in each column.

Exercise 10

A company producing and selling electricity has to plan the production for the next n days. It can choose among m types of fuels, e.g., cool, biomass, wind, etc.
A production unit can be on or off and this status can change on hourly basis. If a production unit uses fuel i, it generates a_i kilowatt of electricity at the cost of c_i. Due to limits on CO_2 emissions, the maximum number of production units per day burning the same fuel i equals to r_i.
The decision maker has to decide the type of fuel to use for each production unit on each hour for the next n days.
The further constraints apply: each fuel i can be used in more than one production unit per day but it can be used in at most p_i days; the costs of production per day must be non increasing during the time horizon, while the daily usage time must be non-decreasing during the production horizon. Let A be a set of pair of fuels. In each day and for each pair of fuels (h, k), if fuel h is used in at least one unit, then fuel k must be planned in at least a unit in the same day as well.
The objective is to maximize the electricity produced over the time horizon.
Model the problem as a MIP problem.

Solution:

First we consider whether we have actually something not trivial to decide. We want to maximize the production, hence we would tend to set the production units always on. There are however contraints limiting how much an unit can be on. This makes the problem not trivial and interesting to approach with MILP.

- $J = \{1, 2, \dots, p\}$ indexed by j set of production units
- $F = \{0, 1, 2, \dots, m\}$ indexed by i set of fuels. We denote by 0 the no-fuel, which means the plant is off
- $T = \{1, 2, ..., q\}$ indexed by t, set of time interval, $q = 24 \cdot n$
- $D = \{1, \dots n\}$ is the set of days and $T_d \subset T$ the set of hours belonging to day d. For example, to day 1 it corresponds $T_1 = \{1, 2, \dots, 24\}$.
- a_i, c_i, r_i, p_i as given
- $A = \{(h, k) \mid h \in F, k \in F\}$ set of pairs of fuels
- x_{ijt} decision variables, $x_{ijt} = 1$ if production unit j is on in hour t using fuel i, 0 if it does not use fuel i in time t. $x_{0jt} = 1$ if the plant is off at that time.
- y_{it} is an auxiliary variables that is 1 if fuel i is used by any plant at time t and 0 otherwise.

$$\max \sum_{i\in F\setminus\{0\}} \sum_{jt} a_i x_{ijt} \tag{8}$$
$$\sum_{i\in F} x_{ijt} = 1 \qquad \forall j \in J, t \in T \tag{9}$$
$$\sum_{j\in J}\sum_{t\in T_d} x_{ijt} = r_i \qquad \forall i \in F \setminus \{0\}, d \in D \tag{10}$$
$$\sum_{j\in J}\sum_{t\in T_d} x_{ijt} \leq 24\ p\ y_{id} \qquad \forall i \in F \setminus \{0\}, d \in D \tag{11}$$
$$\sum_{d\in D} y_{id} \leq p_i \qquad \forall i \in F \setminus \{0\} \tag{12}$$
$$\sum_{t\in T_d}\sum_{j\in J}\sum_{i\in F\setminus\{0\}} c_i x_{ijt} \geq \sum_{t\in T_{d+1}}\sum_{j\in J}\sum_{i\in F\setminus\{0\}} c_i x_{ijt} \qquad \forall d \in \{1,\dots,n-1\} \tag{13}$$
$$\sum_{t\in T_d}\sum_{j\in J}\sum_{i\in F\setminus\{0\}} x_{ijt} \geq \sum_{t\in T_{d+1}}\sum_{j\in J}\sum_{i\in F\setminus\{0\}} x_{ijt} \qquad \forall d \in \{1,\dots,n-1\} \tag{14}$$
$$\frac{1}{24p}\sum_{t\in T_d}\sum_{j\in J} x_{hjt} \leq \sum_{t\in T_d}\sum_{j\in J} x_{kjt} \qquad \forall d \in D, \forall (h,k) \in A \tag{15}$$
$$y_{id} \in \{0,1\} \qquad \forall i \in F, d \in D \tag{16}$$
$$x_{ijt} \in \{0,1\} \qquad \forall i \in F, j \in J, d \in D, t \in T \tag{17}$$

The objective function (8) maximizes the total of electricity produced by the plants when they are on.
The set of constraints (9) ensure that each plant is in exactly one state at each time, that is, it is using on using exactly one of the fuels or it is off.
The set of constraints (10) ensure that the maximum number of production units per day burning the same fuel i equals to r_i.
The set of constraints (11) force the variable y_{id} to be 1 when fuel i is used in day d. $24\ p$ is a constant larger than any value attainable on the left hand side. Then (12) impose that each fuel i is used in at most p_i days;
The set of constraints (13) force the costs of production per day must to be non increasing during the time horizon and constraints (14) the daily usage time to be non-decreasing during the production horizon.
The set of constraints (15) makes sure that in each day and for each pair of fuels (h,k), if fuel h is used in at least one unit, then fuel k must be planned in at least a unit in the same day as well. The fraction $1/24p$ makes sure that the left hand side never exceeds 1 while still being strictly positve when at least one unit uses the fuel k.