INMAS 2021, Modeling & Optimization Problems: Session 2.

1. Show that the following integer programs all have the same set of feasible solutions:

   (a)
$$\begin{aligned} 97x_1 + 32x_2 + 25x_3 + 20x_4 &\le 139 \\ x_i &\in \{0,1\} \end{aligned}$$

   (b)
$$\begin{aligned} 2x_1 + x_2 + x_3 + x_4 &\le 3 \\ x_i &\in \{0,1\} \end{aligned}$$

   (c)
$$\begin{aligned} x_1 + x_2 + x_3 &\le 2 \\ x_1 + x_2 + x_4 &\le 2 \\ x_1 + x_3 + x_4 &\le 2 \\ x_i &\in \{0,1\} \end{aligned}$$

   Solution. Verify the following statements:

   (a) Any solution in which at most two variables are equal to 1 is feasible under all three formulations.
   (b) Any solution in which $x_1 = 1$ and at least two other variables are equal to 1 is infeasible in all three formulations.
   (c) The solution in which $x_1 = 0$ and all other variables are equal to 1 is feasible in all three formulations.

   Note that every solution falls into one of these three cases, so the three formulations have exactly the same sets of feasible points.

   □

2. Suppose that you are interested in choosing a set of investments $\{1, \ldots, 7\}$. Model the following constraints:

   (a) You cannot invest in all of them.

   Solution. $\sum_{i=1}^{7} x_i \le 6$ □

   (b) You must choose at least one of them.

   Solution. $\sum_{i=1}^{7} x_i \ge 1$ □

   (c) Investment 1 cannot be chosen if investment 3 is chosen.

Solution. $x_3 \leq 1 - x_1$. □

(d) Investment 4 can be chosen only if investment 2 is also chosen.

Solution. $x_4 \leq x_2$. □

(e) You must choose either both investments 1 and 5 or neither.

Solution. $x_1 = x_5$. □

(f) You must choose either at least one of the investments 1,2,3 or at least two investments from 2,4,5,6.

Solution. This can be written as:

$$2(x_1 + x_2 + x_3) \geq 2 - (x_2 + x_4 + x_5 + x_6)$$

Alternatively, we could use a disjunction formulation:

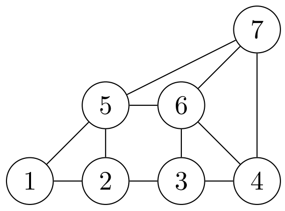
$$x_1 + x_2 + x_3 \geq 1 \text{ or } x_2 + x_3 + x_5 + x_6 \geq 2$$

Applying the disjunction recipe, this would lead to the formulation:

$$\begin{aligned} y_1 + y_2 &= 1 \\ x_1^1 + x_2^1 + x_3^1 &\geq y_1 \\ x_2^2 + x_3^2 + x_5^2 + x_6^2 &\geq 2y_2 \\ x_i &= x_i^1 + x_i^2 \\ 0 \leq x_i^1 &\leq y_1 \\ 0 \leq x_i^2 &\leq y_2 \\ &\text{all variables binary.} \end{aligned}$$

□

3. A graph (V, E) is called a planar graph if it can drawn in a two-dimensional plane so that the edges E intersect only at the vertices V of the graph. A *graph coloring* is an assignment of colors to vertices such that the endpoints of each edge must be different colors. There is a theorem that states that any planar graph has a graph coloring using at most 4 colors. Formulate an integer program that identifies a coloring of a planar graph that uses the minimum number of colors. Implement this IP in Gurobi, and apply your implementation to the following graph:



Solution. Decision Variables:

$$\begin{aligned}
x_{vc} &: \text{binary variable; 1 if vertex } v \text{ is colored } c \\
& \text{ for } v \in V \text{ and } c \in \{1,2,3,4\} \\
y_c &: \text{binary variable; 1 if color } c \text{ is used for } c \text{ in } \{1,2,3,4\}
\end{aligned}$$

Constraints: The endpoints of edges must be different colors.

$$x_{uc} + x_{vc} \leq 1 \qquad \text{for } \{u, v\} \in E, c \in \{1,2,3,4\}$$

No nodes can be color c if color c is not used:

$$x_{vc} \leq y_c \qquad \text{for } v \in V, c \in \{1,2,3,4\}$$

Every node must be colored:

$$\sum_{c \in \{1,2,3,4\}} x_{vc} = 1 \qquad \text{for } c \in \{1,2,3,4\}$$

Objective:

$$\sum_{c \in \{1,2,3,4\}} y_c$$

For a Gurobi implementation, see the script "graphcolor.py". There is a solution that uses three colors: nodes 1,3, and 7 receive the same colors; nodes 2 and 6 receive the same colors; nodes 4 and 5 receive the same colors. □

4. Formulate the following as constraints to mixed integer programs:

   (a) $u = \max\{x_1, x_2\}$ assuming that $0 \leq x_j \leq C$ for $j = 1, 2$.

   Solution. Using a disjunctive constraint, we can write this as the following system of constraints:

$$\begin{aligned}
&u \geq x_1 \\
&u \geq x_2 \\
&u \leq x_1 \text{ or } u \leq x_2
\end{aligned}$$

   We introduce binary variables y_1 and y_2, and continuous variables $x_1^1, x_1^2, x_2^1, x_2^2, u^1, u^2$. Then, applying the disjunction recipe would give us the formulation

$$\begin{aligned}
u &\geq x_1 \\
u &\geq x_2 \\
y_1 + y_2 &= 1 \\
0 &\leq u^1 \leq Cy_1 \\
0 &\leq u^2 \leq Cy_2 \\
0 &\leq x_1^1 \leq Cy_1 \\
0 &\leq x_1^2 \leq Cy_2 \\
0 &\leq x_2^1 \leq Cy_1 \\
0 &\leq x_2^2 \leq Cy_2 \\
u^1 &\leq x_1^1 \\
u^2 &\leq x_2^2 \\
x_1 &= x_1^1 + x_1^2 \\
x_2 &= x_2^1 + x_2^2 \\
u &= u^1 + u^2
\end{aligned}$$

A simpler formulation following a similar idea would be:

$$\begin{aligned}
u &\geq x_1 \\
u &\geq x_2 \\
y_1 + y_2 &= 1 \\
0 &\leq u^1 \leq Cy_1 \\
0 &\leq u^2 \leq Cy_2 \\
u^1 &\leq x_1 \\
u^2 &\leq x_2 \\
u &= u^1 + u^2
\end{aligned}$$

$\square$

(b) $v = |x_1 - x_2|$ assuming that $0 \leq x_j \leq C$ for $j = 1, 2$.

Solution. This has a similar idea. We can rewrite the constraint $v = |x_1 - x_2|$ as

$$v = \max\{x_1 - x_2, x_2 - x_1\}$$

So it almost looks like we can apply the formulation from the previous part. However, there is one obstacle: an assumption in the previous formulation was that both of the terms within the max are bounded below by 0. In this case, that is not true, since $x_1 - x_2$ could be any number between $-C$ and C. To fix this, let's write:

$$v = \max\{x_1 - x_2 + C, x_2 - x_1 + C\} - C$$

Now, we have essentially reduced the problem to that we addressed in the previous part, which gives us the formulation:

$$\begin{aligned}
v &= u - C \\
u &\geq x_1 - x_2 + C \\
u &\geq x_2 - x_1 + C \\
y_1 + y_2 &= 1 \\
0 &\leq u^1 \leq C y_1 \\
0 &\leq u^2 \leq C y_2 \\
u^1 &\leq x_1 - x_2 + C \\
u^2 &\leq x_2 - x_1 + C \\
u &= u^1 + u^2
\end{aligned}$$

□

(c) Let x be an n-dimensional vector of integer decision variables, and let $x*$ be an integral point in $\mathbb{R}^n$; model the constraint $x \neq x^*$ under the assumption that $0 \leq x_j \leq C$ for each j.

Solution. First, consider the case that $x^* \not\geq 0$. In this case, x^* is already excluded by the constraints $x \geq 0$. So, we may assume without loss of generality that $x \geq 0$. Similarly, we can assume without loss of generality that $x_j^* \leq C$ for each j. Then, we can write the condition $x \neq x^*$ as

$$\sum_{i=1}^{n} |x_i - x_i^*| \geq 1$$

By introducing a vector of variables v, we can rewrite this condition as:

$$\begin{aligned}
v_i &= |x_i - x_i^*| \\
\sum_{i=1}^{n} v_i &\geq 1
\end{aligned}$$

Then, applying the previous part, this gives us the formulation:

$$\begin{aligned}
\sum_{i=1}^{n} v_i &\geq 1 \\
v_i &= u_i - C \\
u_i &\geq x_i - x_i^* + C \\
u_i &\geq x_i^* - x_i + C \\
y_i^1 + y_i^2 &= 1 \\
0 \leq u_i^1 &\leq C y_i^1 \\
0 \leq u_i^2 &\leq C y_i^2 \\
u_i^1 &\leq x_i - x_i^* + C \\
u_i^2 &\leq x_i^* - x_i + C \\
u_i &= u_i^1 + u_i^2
\end{aligned}$$

□

5. Suppose that you have a single machine, and you have several tasks to perform on this machine. The machine can only process one task at a time, but you may choose the order in which the tasks are performed. Each of these tasks has a deadline. Let t_j be the time required to complete task j, let d_j be the deadline for task j. There is a penalty for missing deadlines. Specifically, if task j is completed before time t_j, there is no penalty, while if task j is completed at some time $\tau > d_j$, then the penalty is $p_j(\tau - d_j)$ for some constant p_j.

   Hint: define variables:

$$x_j = \begin{cases} 1 & \text{the time at which job } j \text{ is completed.} \\ 0 & \text{otherwise.} \end{cases}$$

   Then, note that either $x_j \geq x_i + t_j$ or $x_i \geq x_j + t_i$; where the former holds if task i is placed before task j and the latter holds if task i is placed after task j.

   (a) Formulate an integer program that identifies the optimal sequence of tasks.

   Solution. Let us define continuous variables:

$$x_j = \text{the time at which job } j \text{ is completed.}$$

   It is clear that we can include the constraints:

$$x_j \geq t_j$$

because if job i is placed first, this job would be completed at time t_i.
Next, we will add some disjunctive constraints. For any task i and j, either task i is placed before task j, or task i is placed after j. If task i is placed before j, then $x_j \geq x_i + t_j$, which is to say that the time at which task j is completed is greater than the time at which task i is completed plus the amount of time required to complete task j. If task j is placed before task i, then $x_i \geq x_j + t_i$. Thus, we have the disjunction:

$$x_j \geq x_i + t_j \text{ or } x_i \geq x_j + t_i$$

There are $\binom{n}{2}$ such disjunctions. Let's fix some i and j, and look at the resulting disjunction. We introduce binary variables y^i_{ij} and y^j_{ij}, where

$$y_{ij} = \begin{cases} 1 & \text{if task } i \text{ is placed before } j \\ 0 & \text{otherwise} \end{cases}$$

and

$$y_{ji} = \begin{cases} 1 & \text{if task } j \text{ is placed before } i \\ 0 & \text{otherwise} \end{cases}$$

We will also introduce continuous variables z^{ij}_i, z^{ij}_j, z^{ji}_j, z^{ji}_j where

$$z^{ij}_i = \begin{cases} \text{time at which job } i \text{ completed} & \text{if } i \text{ before } j \\ 0 & \text{otherwise} \end{cases}$$

Here, the subscript i denotes that we are referring the completion time of job i, and the superscript ij denotes that we are considering the case in which job i is placed before job j. Then, the disjunction recipe gives us the constraints:

$$\begin{aligned} y_{ij} + y_{ji} &= 1 \\ 0 \leq z^{ij}_i &\leq My_{ij} \\ 0 \leq z^{ji}_i &\leq My_{ji} \\ 0 \leq z^{ij}_j &\leq My_{ij} \\ 0 \leq z^{ji}_j &\leq My_{ji} \\ z^{ij}_j &\geq z^{ij}_i + t_j y_{ij} \\ z^{ji}_i &\geq z^{ji}_j + t_i y_{ji} \\ x_i &= z^{ij}_i + z^{ji}_i \\ x_j &= z^{ij}_j + z^{ji}_j \end{aligned}$$

If there are n tasks, then gathering up and organizing these constraints for every possible value of i and j gives us:

$$\begin{aligned}
x_j &\geq t_j \text{ for } j \in [n] \\
y_{ij} + y_{ji} &= 1 \text{ for } i, j \in [n] \text{ s.t. } i < j \\
0 \leq z_i^{ij} &\leq M y_{ij} \text{ for } i, j \in [n] \text{ s.t. } i \neq j \\
0 \leq z_i^{ji} &\leq M y_{ji} \text{ for } i, j \in [n] \text{ s.t. } i \neq j \\
z_j^{ij} &\geq z_i^{ij} + t_j y_{ij} \text{ for } i, j \in [n] \text{ s.t. } i \neq j \\
x_i &= z_i^{ij} + z_i^{ji} \text{ for } i, j \in [n] \text{ s.t. } i \neq j \\
x, z &\text{ continuous} \\
y &\text{ binary}
\end{aligned}$$

These constraints are a valid formulation for the vector of completion times. That is, if there is a feasible solution (x, y, z) to this set of constraints, then it is possible to process the jobs in such a manner that job i is completed at time x_i. Conversely, given a plan for processing the jobs, it is possible to find a feasible solution (x, y, z) such that y encodes the sequence of jobs and x encodes the job completion times. This may not be immediately obvious, and you may want to confirm this yourself. For the value of the big M, we can use the total processing time of all items $\sum_{i=1}^n t_i$.
However, it is difficult to express the objective value solely in terms of these constraints. Let us introduce continuous variables:

$$\ell_j = \begin{cases} x_j - d_j & \text{if } x_j > d_j \\ 0 & \text{otherwise} \end{cases}$$

That is ℓ_j records the lateness of job j; if job j is early, then ℓ_j is zero, while if job j is late, the value of ℓ_j is equal to the number of minutes late. We can achieve this with the constraints:

$$\begin{aligned}
&\ell_j \geq x_j - d_j \text{ for } j \in [n] \\
&\ell_j \geq 0 \text{ for } j \in [n] \\
&\ell \text{ continuous}
\end{aligned}$$

And then we can write the objective as:

$$\min_{x,y,z,\ell} \sum_{j=1}^n p_j \ell_j$$

Note that here, we are allowing some feasible solutions in which the variables ℓ_j that record lateness are strictly greater than the actual lateness. However, it can be shown that in every optimal solution, the lateness variables are exactly equal to the actual lateness, so this formulation is valid (proof left to the reader). □

(b) Using Gurobi, solve the following instance:

Task	Processing time	Deadline	Penalty (per minute late)
1	5 minutes	60 minutes	\$10
2	10 minutes	20 minutes	\$18
3	17 minutes	32 minutes	\$14
4	8 minutes	27 minutes	\$89
5	9 minutes	44 minutes	\$22
6	14 minutes	52 minutes	\$37
7	28 minutes	77 minutes	\$46

Solution. For an implementation in Python using Gurobi, see the file "machineschedule.py". A solution is as follows. The jobs are arranged in order 4, 2, 6, 5, 1, 7, 3. These jobs are started at times 0, 8, 18, 32, 41, 46, and 74 respectively and completed at times 8, 18, 32, 41, 46, 74 and 91 respectively. The total penalties incurred amount to \$826. □