Exercises for Mathematical Programming

186.835 VU 3.0 – SS 2021

Fabian Holzberger 11921655

Exercise 1 *Network Design* ***(1 point)***

We are given a directed graph $G = (V, A)$ and a value $b_i \in \mathbb{R}$ for each node $i \in V$ which denotes a demand ($b_i < 0$) or a supply ($b_i > 0$), such that $\sum_{i \in V} b_i = 0$. There are two types of costs: transportation costs c_{ij} of shipping one unit from node i to node j, and building costs of establishing a direct link (i, j) from node i to j. A link on arc (i, j) can be built (i) with costs d^1_{ij} and capacity u^1_{ij}, or (ii) with costs d^2_{ij} and capacity u^2_{ij}. Assume that $d^1_{ij} < d^2_{ij}$ and $u^1_{ij} < u^2_{ij}$ and that at most one option can be chosen on each arc. A network has to be built that satisfies all demands and minimizes the total building and transportation costs. Formulate the problem as a (mixed) integer linear program.
Answer: Assume we model an arc of type 1 between (i, j) by the variable $x^1_{ij} \in \{0, 1\}$ and of type 2 respectively by $x^2_{i,j} \in \{0, 1\}$. Then the objective function is given by:

$$g(x^1, x^2) = \sum_i \sum_j \left[x^1_{ij}(c_{ij} u^1_{ij} + d^1_{ij}) + x^2_{ij}(c_{ij} u^2_{ij} + d^2_{ij})\right] \tag{1}$$

Therefore, we get the following MILP:

$$\min_{x_1, x_2} g(x^1, x^2) \tag{2}$$
$$\sum_j (x^1_{ij} u^1_{ij} + x^2_{ij} u^2_{ij}) \leq b_i \quad \forall i \tag{3}$$
$$\sum_j (x^1_{ij} u^1_{ij} + x^2_{ij} u^2_{ij}) \geq b_i \quad \forall i \tag{4}$$
$$x^1_{ij} \geq 0 \quad \forall i, j \tag{5}$$
$$x^2_{ij} \geq 0 \quad \forall i, j \tag{6}$$
$$x^1_{ij} + x^2_{ij} \leq 1 \quad \forall i, j \tag{7}$$

In (3) and (4) we ensure that the demand in every node is satisfied. The constraint (7) only allows that either a type 1 or type 2 connection is build.

Exercise 2 *Scheduling* ***(1 point)***

A factory consists of m machines $M_1, \ldots, M_m$. Each job $j = 1, \ldots, n$ needs to be processed on each machine in the specific order $M_{j(1)}, \ldots, M_{j(m)}$ (permutation of the machines). Machine M_i takes time p_{ij} to process job j. A machine can only process one job at a time, and once a job is started on any machine, it must be processed to completion. The objective is to minimize the **sum of the completion times** of all the jobs. The completion time of job j is the time when the last subtask on machine $M_{j(m)}$ is finished. Formulate the problem as a (mixed) integer linear program.
Answer: Assume the variable x_{kj} defines the start time of a job j on machine k. Jobs cant overlap on a machine, and we want that the condition $(x_{kj} - x_{ki} \geq p_{ki}) \vee (x_{ki} - x_{kj} \geq p_{kj})$ holds. One can express this condition with aid of a new variable y_{kji}, that shall be 1 if job j is starting before i on machine M_k and 0 else, as done in [1]. Therefore let $y_{kij} \in \mathbb{N}$ and $0 \leq y_{kij} \leq 1$. Next define the number $T_\infty := \sum_{k,i} p_{ki} \geq \sup_{i,j} |x_{li} - x_{lj}| \quad \forall l$. Then we have the conditions:

$$(T_\infty + p_{ki})(1 - y_{kji}) + (x_{kj} - x_{ki}) \geq p_{ki} \quad \forall i, j, k \tag{8}$$
$$(T_\infty + p_{kj}) y_{kji} + (x_{ki} - x_{kj}) \geq p_{kj} \quad \forall i, j, k \tag{9}$$

Note that the conditions imply that if $(x_{ki} - x_{kj}) > 0$ and (8) holds then $y_{kji} = 0$ and on the other side if $(x_{ki} - x_{kj}) \leq 0$ and (9) hold then $y_{kji} = 1$, as required. Further, we see that the two conditions ensure that start times can not overlap. To ensure that a job j is executed in its right order $M_{j(1)}, \ldots, M_{j(m)}$ we introduce the variable z_{hkj} that is 1 if the job j needs to be executed on machine M_h before it can be executed on M_k and 0 otherwise. We can precompute the value of z_{hkj} and therefore add the last condition:

$$z_{hkj}(x_{hj} + p_{hj}) \leq x_{kj} \quad \forall j \tag{10}$$

The full MILP is therefore:

$$\min C_{max} \tag{11}$$

$$(T_\infty + p_{ki})(1 - y_{kji}) + (x_{kj} - x_{ki}) \geq p_{ki} \quad \forall i, j, k \tag{12}$$

$$(T_\infty + p_{kj})y_{kji} + (x_{ki} - x_{kj}) \geq p_{kj} \quad \forall i, j, k \tag{13}$$

$$z_{hkj}(x_{hj} + p_{hj}) \leq x_{kj} \quad \forall j \tag{14}$$

$$x_{kj} \geq 0 \quad \forall k, j \tag{15}$$

$$ykij \geq 0 \quad \forall k, i, j \tag{16}$$

$$ykij \leq 1 \quad \forall k, i, j \tag{17}$$

$$\tag{18}$$

Exercise 3 *Sports League (**3 points**)*

The season in a sports league with n teams has started. Each team plays against each other team at home and away, so each team plays exactly $2(n-1)$ games. In case of a win, a team receives three points, in case of a draw one point, and in case of a loss zero points. After the season the k worst teams with respect to the total number of achieved points get relegated. The task is now to determine the minimum number of points a team has to achieve to have a guarantee that it is not relegated in any season outcome.

- Formulate the problem as a (mixed) integer linear program. (**1 point**)
- Solve the problem for $n = 18$ and $k = 3$ with an MILP solver (e.g., CPLEX, Gurobi, Excel Solver). (**2 points**)

Answer: To model the outcome of the games we create three variables indexed by the games x^w_{ij}, x^d_{ij}, $x^l_{ij} \in \{0, 1\}$ where the first is one is 1 if team i wins against j at home and 0 otherwhide, the second one is 1 if the game is a draw and the third 1 if i looses abd therefore j wins. Note that we only create variables for $i \neq j$ and therefore can model a game at home for team i by the index ij and a game away by ji. A team can either win, loose or have a draw with another team. Therefore, we have the constraint:

$$x^w_{ij} + x^d_{ij} + x^l_{ij} = 1 \forall i, j \text{ with } i \neq j \tag{19}$$

The points of a team can be described by the points earned in home games and away games:

$$p_t = \sum_{j, j \neq t} (3x^w_{tj} + x^d_{tj} + 3x^l_{jt} + x^d_{jt}) \tag{20}$$

Note that for an away game of theam t we have the indices jt such that x^w_{jt} is 1 if j wins but x^l_{jt} if one if t wins, which is the reason we include it above. Next we want to order the points from $t = 0$ with the lowest points to $t = n$ that acheaves the highest points. Therefore let $p_t \leq p_{t+1}$ for $t = 0, 1, ..., n-1$. The objective is then to maximize p_k such that we then know that if a team has $_k + 1$ points it is surely not relegated in any season outcome. The MILP taken from [2] is therefore:

$$\max p_k = \max \left[\sum_{j, j \neq k} (3x^w_{kj} + x^d_{kj} + 3x^l_{jk} + x^d_{jk}) \right] \tag{21}$$

$$x^w_{ij} + x^d_{ij} + x^l_{ij} = 1 \quad \forall i, j \text{ with } i \neq j \tag{22}$$

$$\sum_{j, j \neq t} (3x^w_{tj} + x^d_{tj} + 3x^l_{jt} + x^d_{jt}) \leq \sum_{j, j \neq t+1} (3x^w_{t+1j} + x^d_{t+1j} + 3x^l_{jt+1} + x^d_{jt+1}) \quad \forall t \in \{0, 1, ..., n-1\} \tag{23}$$

$$\tag{24}$$

We want to solve the the MILP for $n = 18$ and $k = 3$ by the Python MIP package (link).

Listing 1:

```
import pip
import numpy as np
pip.main(['install', 'mip'])
```

```python
from mip import Model, xsum, maximize, BINARY

nt = 18
k  = 2

m = Model("Sports League")

xw = []
xd = []
xl = []

for i in range(nt):
  xw.append([])
  xd.append([])
  xl.append([])
  for j in range(nt):
    if i!=j:
      xw[i].append(m.add_var(var_type=BINARY))
      xd[i].append(m.add_var(var_type=BINARY))
      xl[i].append(m.add_var(var_type=BINARY))
      m += (xw[i][j] + xd[i][j] + xl[i][j]) == 1
    else:
      xw[i].append([])
      xd[i].append([])
      xl[i].append([])

for t in range(nt-1):
  m +=    xsum( 3*xl[i][t]   + xd[i][t]   +
                3*xw[t][i]   + xd[t][i]  for i in range(nt) if i!=t) <=\
          xsum( 3*xl[i][t+1] + xd[i][t+1] +
                3*xw[t+1][i] + xd[t+1][i]  for i in range(nt) if i!=(t+1))

m.objective = maximize(xsum( 3*xl[i][k] + xd[i][k] +
                             3*xw[k][i] + xd[k][i]  for i in range(nt) if i!=k))
m.optimize()

pk = np.sum([3*int(xl[i][k].x) + int(xd[i][k].x) +
  3*int(xw[k][i].x) + int(xd[k][i].x) for i in range(nt) if i!=k])

print("Team k={} has {} points and will still relegate.".format(k,pk)+\
  "Therefore a team is safe if it has at least {} points".format(pk+1))
```

The result is that the team 0 and 1 loose all games except when they play against each other they have a draw. Therefore, team 0 and 1 have 2 points and all other teams 57 points, including team 2. Therefore, a team will never relegate if it reaches 58 points.

References

[1] A. S. Manne, "On the job-shop scheduling problem," *Operations Research*, vol. 8, no. 2, pp. 219–223, 1960.

[2] C. Raack, A. Raymond, A. Werner, and T. Schlechte, "Integer programming and sports rankings," 2013.