1.3 Beispiel: Raumzuordnung

Situation:

- n Leute sollen auf n Büros verteilt werden.
- Jede Person i gibt eine Rangfolge der Büros an (r_{ij} ist der Rang von Büro j in der Liste von Person i).
- Aufgabe: Finde eine Zuteilung, so dass die Summe der erhaltenen Ränge minimal ist.

[7]

AMPL-Modell (assign.mod)

```
set PEOPLE;
set OFFICES;

param rank {PEOPLE,OFFICES};

var X {PEOPLE,OFFICES} binary;

minimize Total_Cost:
   sum {i in PEOPLE, j in OFFICES}
     rank[i,j] * X[i,j];

subject to assign_p {i in PEOPLE}:
   sum {j in OFFICES} X[i,j] = 1;

subject to assign_o {j in OFFICES}:
   sum {i in PEOPLE} X[i,j] = 1;
```

AMPL-Daten (`myassign.dat`)

```
set PEOPLE := Coullard Daskin Hazen Hopp
              Iravani Linetsky Mehrotra Nelson
              Smilowitz Tamhane White ;

set OFFICES := C118 C138 C140 C246 C250 C251 D237 D239
               D241 M233 M239;

param rank:
        C118 C138 C140 C246 C250 C251 D237 D239 D241 M233 M239:=
Coullard   6    9    8    7   11   10    4    5    3    2    1
Daskin    11    8    7    6    9   10    1    5    4    2    3
Hazen      9   10   11    1    5    6    2    7    8    3    4
Hopp      11    9    8   10    6    5    1    7    4    2    3
Iravani    3    2    8    9   10   11    1    5    4    6    7
Linetsky  11    9   10    5    3    4    6    7    8    1    2
Mehrotra   6   11   10    9    8    7    1    2    5    4    3
Nelson    11    5    4    6    7    8    1    9   10    2    3
Smilowitz 11    9   10    8    6    5    7    3    4    1    2
Tamhane    5    6    9    8    4    3    7   10   11    2    1
White     11    9    8    4    6    5    3   10    7    2    1;
```

Lösen mit AMPL

```
ampl: model assign.mod;
ampl: data myassign.dat;
ampl: option solver gurobi;
ampl: solve;
Gurobi 8.1.0: optimal solution; objective 28
26 simplex iterations
ampl: display X;
X [*,*]
:          C118 C138 C140 C246 C250 C251 D237 D239 D241 M233 M239
Coullard      1    0    0    0    0    0    0    0    0    0    0
Daskin        0    0    0    0    0    0    0    0    1    0    0
Hazen         0    0    0    1    0    0    0    0    0    0    0
Hopp          0    0    0    0    0    0    1    0    0    0    0
Iravani       0    1    0    0    0    0    0    0    0    0    0
Linetsky      0    0    0    0    1    0    0    0    0    0    0
Mehrotra      0    0    0    0    0    0    0    1    0    0    0
Nelson        0    0    1    0    0    0    0    0    0    0    0
Smilowitz     0    0    0    0    0    0    0    0    0    1    0
Tamhane       0    0    0    0    0    1    0    0    0    0    0
White         0    0    0    0    0    0    0    0    0    0    1
;
```

1.4 Beispiel: Paarungen

Situation:

- $n = 2k$ Leute sollen paarweise auf k Büros verteilt werden.
- Für jedes Paar $\{i, j\}$ haben wir einen Parameter $w_{\{i,j\}} \in \mathbb{R}$, der einschätzt, wie gut die beiden zusammen arbeiten können (je kleiner desto besser).
- Aufgabe: Finde eine Paarung, für welche die Summe der w-Werte minimal ist.

[8]

AMPL-Modell (paar.mod)

```
set PEOPLE;
set PAIRS within PEOPLE cross PEOPLE;
# data file: define for i>j only

param weight {PAIRS};

var x {PAIRS} binary;

minimize Total_Weight:
sum {(i,j) in PAIRS} weight[i,j] * x[i,j];

s.t. DegreeConstraint {i in PEOPLE}:
sum {(i,j) in PAIRS} x[i,j]
  + sum {(j,i) in PAIRS} x[j,i] = 1;
```

AMPL-Daten (paar.dat)

```
set PEOPLE :=  1 2 3 4 5 6 7 8 9 10 ;
param: PAIRS: weight:
      1   2   3   4   5   6   7   8   9  10  :=
  1   .   .   .   .   .   .   .   .   .   .
  2  -5   .   .   .   .   .   .   .   .   .
  3  -6   3   .   .   .   .   .   .   .   .
  4  -3  -3  -4   .   .   .   .   .   .   .
  5  -2  -9   6  -5   .   .   .   .   .   .
  6   3  -9   1  -8  -4   .   .   .   .   .
  7   7   0  10  -3   8  -2   .   .   .   .
  8   9  -9  -9  -4  -3  -2   4   .   .   .
  9 -10   0  -9   2  10 -10  -4  -6   .   .
 10  -1   0   7   3  -5   9  -4  -5   0   . ;
```

Lösen mit AMPL

```
ampl: model paar.mod;
ampl: data paar.dat;
ampl: option solver gurobi;
ampl: solve;
Gurobi 8.1.0: optimal solution; objective -40
13 simplex iterations
ampl: display x;
x [*,*]
:    1   2   3   4   5   6   7   8   9    :=
2    0   .   .   .   .   .   .   .   .
3    0   0   .   .   .   .   .   .   .
4    0   0   0   .   .   .   .   .   .
5    0   1   0   0   .   .   .   .   .
6    0   0   0   1   0   .   .   .   .
7    0   0   0   0   0   0   .   .   .
8    0   0   1   0   0   0   0   .   .
9    1   0   0   0   0   0   0   0   .
10   0   0   0   0   0   0   1   0   0
;
```

1.5 Beispiel: Reihenfolgeplanung

Situation:

- n Jobs müssen nacheinander auf einer Maschine ausgeführt werden.
- Es sind Umrüstzeiten $t_{i,j}$ bekannt, die benötigt werden, wenn Job j unmittelbar nach Job i ausgeführt wird.
- Zur Vereinfachung: Job 1 muss als erster ausgeführt werden, Job n als letzter.
- Aufgabe: Finde eine Reihenfolge der Jobs, welche die Summe der Umrüstzeiten minimiert.

[9]