Exercise cpc-08: Example 5: Configuration of track indicators

Exercise-08b: Configuration of track indicators

This exercise is an alternative to 08a!
You need not implement both, but only one …

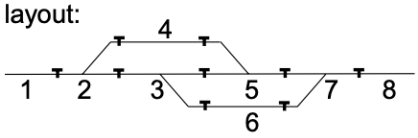
Model the track indication problem from **example 5** in cpc-examples.pdf in MiniZinc.

Required result: PDF file named surname(s)-08b.pdf containing the contents of following files
- tracks.mzn: generic program
- *.dzn: various examples

Be prepared to run your MiniZinc program during your presentation!

Additional questions for discussion in the lesson:
- Differences of solving times for different layouts

Based on a (much more complicated) example from the railroad domain:
- Example track layout:



- T-separators divide tracks into sections (e.g. 1-8)
- Each section is monitored by a track indicator
- Each track indicator has a type (A, B, C, D)
- Neighboring indicators are subject to restrictions:
  - A allows only B as neighbors
  - B allows only A or C as neighbors
  - C allows only B or D as neighbors
  - D allows only C as neighbors

Tasks:
- Model - specific for the example track layout, e.g. single mzn file with variables, domains, constraints, test with all solutions
- Generic model for arbitrary track layouts, e.g. array of variables for sections in an mzn file, table for neighborship in a dzn file (for each layout)
- Test with various layouts and compare solving times e.g. duplicate example layout (and optionally connect the two)
- Define an unsatisfiable layout
- Implement **optimization** alternatives:
  - Minimize cost: A=1, B=2, C=3, D=4
  - Ensure equal distribution: All 4 types shall be selected as equally as possible
  - Pareto front of both alternatives (criteria)

Generic model (tracks.mzn):

```
enum Type = {A, B, C, D};
array[Type,Type] of 0..1: allows = [|
% A, B, C, D
  0, 1, 0, 0 | % A
  1, 0, 1, 0 | % B
  0, 1, 0, 1 | % C
  0, 0, 1, 0   % D
|];

int: NrSections;

set of int: Sections = 1..NrSections;

array[Sections,Sections] of 0..1: neighborship;

array[Sections] of var Type: indicators;

constraint forall (i,j in Sections where i<j)
  (neighborship[i, j]==1 -> allows[indicators[i],indicators[j]] == 1);

solve satisfy;
```

Original Example (8 sections):

```
NrSections = 8;

% only upper half is used because of i<j constraint
neighborship = [|
% 1, 2, 3, 4, 5, 6, 7, 8
  0, 1, 0, 0, 0, 0, 0, 0 | %1
  0, 0, 1, 1, 0, 0, 0, 0 | %2
  0, 0, 0, 0, 1, 1, 0, 0 | %3
  0, 0, 0, 0, 1, 0, 0, 0 | %4
  0, 0, 0, 0, 0, 0, 1, 0 | %5
  0, 0, 0, 0, 0, 0, 1, 0 | %6
  0, 0, 0, 0, 0, 0, 0, 1 | %7
  0, 0, 0, 0, 0, 0, 0, 0   %8
|];
```

Output Sample:

```
Running tracks.mzn, sample8_1.dzn                                    169msec
indicators = [A, B, A, A, B, B, A, B];
% time elapsed: 169msec
----------
%%%mzn-stat: failures=0
%%%mzn-stat: initTime=0.00036
%%%mzn-stat: nodes=5
%%%mzn-stat: peakDepth=4
%%%mzn-stat: propagations=40
%%%mzn-stat: propagators=9
%%%mzn-stat: restarts=0
%%%mzn-stat: solutions=1
%%%mzn-stat: solveTime=9.3e-05
%%%mzn-stat: variables=8
%%%mzn-stat-end
Finished in 169msec.
```

Unsatisfiable Example:

```
NrSections = 8;

% only upper half is used because of i<j constraint
neighborship = [|
% 1, 2, 3, 4, 5, 6, 7, 8
  0, 1, 0, 0, 0, 0, 0, 0 | %1
  0, 0, 1, 1, 0, 0, 0, 1 | %2
  0, 0, 0, 0, 1, 1, 0, 0 | %3
  0, 0, 0, 0, 1, 0, 0, 0 | %4
  0, 0, 0, 0, 0, 0, 1, 0 | %5
  0, 0, 0, 0, 0, 0, 1, 0 | %6
  0, 0, 0, 0, 0, 0, 0, 1 | %7
  0, 0, 0, 0, 0, 0, 0, 0   %8
|];
```

Output Sample:

```
Running tracks.mzn, sample8_1_unsatisfiable.dzn                    190msec
=====UNSATISFIABLE=====
%%%mzn-stat: failures=4
%%%mzn-stat: initTime=0.000384
%%%mzn-stat: nodes=7
%%%mzn-stat: peakDepth=1
%%%mzn-stat: propagations=124
%%%mzn-stat: propagators=10
%%%mzn-stat: restarts=0
%%%mzn-stat: solutions=0
%%%mzn-stat: solveTime=0.000129
%%%mzn-stat: variables=8
%%%mzn-stat-end
Finished in 190msec.
```

Double-sized Example:

```
NrSections = 16;

% only upper half is used because of i<j constraint
neighborship = [|
% 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 11 12 13 14 15 16
  0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0 | %1
  0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0 | %2
  0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0 | %3
  0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0 | %4
  0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0 | %5
  0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0 | %6
  0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1 | %7
  0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0 | %8
  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 | %9
  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 | %10
  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0 | %11
  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 | %12
  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 | %13
  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 | %14
  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1 | %15
  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 | %16
|];
```

Output Sample:

```
Running tracks.mzn, sample8_1_16t.dzn                                    193msec
indicators = [A, B, A, A, B, B, A, B, A, B, A, A, B, B, A, B];
% time elapsed: 161msec
----------
%%%mzn-stat: failures=0
%%%mzn-stat: initTime=0.000436
%%%mzn-stat: nodes=9
%%%mzn-stat: peakDepth=8
%%%mzn-stat: propagations=98
%%%mzn-stat: propagators=21
%%%mzn-stat: restarts=0
%%%mzn-stat: solutions=1
%%%mzn-stat: solveTime=0.00013
%%%mzn-stat: variables=16
%%%mzn-stat-end
Finished in 193msec.
```

Larger Example (32 sections):

```
NrSections = 32;

% only upper half is used because of i<j constraint
neighborship = [|
% 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32
  0, 1, 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, 0 | %1
  0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0 | %2
  0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 1, 0 | %3
  0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0 | %4
  0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1 | %5
  0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0 | %6
  0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0 | %7
  0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0 | %8
  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0 | %9
  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0 | %10
  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0 | %11
  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1 | %12
  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0 | %13
  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0 | %14
  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0 | %15
  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0 | %16
  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0 | %17
  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0 | %18
  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0 | %19
  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0 | %20
  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1 | %21
  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0 | %22
  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0 | %23
  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0 | %24
  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0 | %25
  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0 | %26
  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0 | %27
  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0 | %28
  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0 | %29
  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1 | %30
  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 | %31
  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 | %32
|];
```

Output Sample:

```
Running tracks.mzn, sample8_1_16t copy.dzn                                                197msec
indicators = [A, B, A, A, B, B, A, B, A, B, A, A, B, B, A, B, A, B, A, A, B, B, A, B, B, B, A, A,
B, B, A, B];
% time elapsed: 166msec
----------
%%%mzn-stat: failures=0
%%%mzn-stat: initTime=0.00052
%%%mzn-stat: nodes=16
%%%mzn-stat: peakDepth=15
%%%mzn-stat: propagations=179
%%%mzn-stat: propagators=39
%%%mzn-stat: restarts=0
%%%mzn-stat: solutions=1
%%%mzn-stat: solveTime=0.000133
%%%mzn-stat: variables=32
%%%mzn-stat-end
Finished in 197msec.
```

Solver Performance Comparison:

Size	Average solveTime	Avg. overall time
8	0.000093	169ms
16	0.000130	161ms
32	0.000133	166ms