# Knowledge Representation and Reasoning

Exercise Session 4

Teaching assistant: Robbe Van den Eede (robbe.vandeneede@kuleuven.be)

IDP-Z3 is a knowledge representation and reasoning system based on the representation language FO(·). In this exercise session, we use FO(·) to represent a number of model expansion problems, and IDP-Z3 to solve them. A model expansion problem is the problem of, given a logical theory and a partial interpretation/structure, extending the partial interpretation to a complete assignment such that it satisfies the theory.

IDP-Z3 is available online at: `https://interactive-consultant.idp-z3.be/IDE`

To learn the IDP-Z3 syntax:

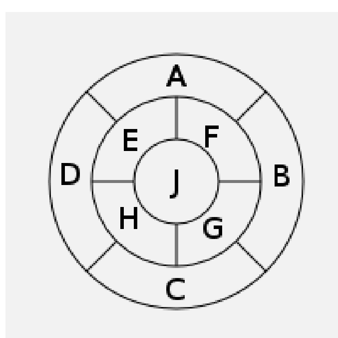
- You can play with the default knowledge base
- The reference manual is available from the Help menu (top right), and here: `https://docs.idp-z3.be/en/stable/`
- A cheat sheet is available here: `https://docs.idp-z3.be/en/stable/summary.html`

Hints for the exercise:

- You can start from the accompanying template files.
- Begin by declaring the vocabulary. Hint: consider using functions instead of predicates, to represent uniqueness conditions.
- Translate the knowledge of the problem domain into FO(·).
- Use a partial structure to specify the problem instance.

## 1 4-Colouring of a graph

Given the following graph, assign a colour to each node, such that neighbouring nodes are assigned different colors. By the four colour theorem, it suffices to consider four colours.



| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 8 | 5 | | | | 2 | 4 | | |
| 7 | 2 | | | | | | | 9 |
| | | 4 | | | | | | |
| | | | 1 | | 7 | | | 2 |
| 3 | | 5 | | | | 9 | | |
| | 4 | | | | | | | |
| | | | | 8 | | | 7 | |
| | 1 | 7 | | | | | | |
| | | | | 3 | 6 | | 4 | |

## 2 Sudoku

Many puzzles and riddles can be easily solved by modelling them in a KR framework and doing model expansion.

1. Fill in the template to solve the following Sudoku problem:

   Each row, column, and $3 \times 3$ block should contain all the different digits from 1 to 9.

   (Note that by specifying the problem instance in a partial structure and not in the theory, the modelling is easily adapted to other sudoku puzzles as well.)

2. Can you easily adapt your solution to bigger sudoku puzzles? How would you adapt your modelling to the following puzzle?

| | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| D | 3 | B | E | 6 | 8 | 1 | 4 | F | A | | 0 | | 5 | | |
| 4 | 1 | | F | | | | | | | | 7 | | | | |
| | | 9 | 0 | E | | 5 | | | 1 | 6 | | | 3 | 2 | |
| 6 | C | 5 | | A | | D | | | | | 9 | E | 1 | | |
| | D | | | | | 4 | | 6 | | B | 2 | | | | |
| | | E | | 7 | B | | | | F | 0 | C | D | | | |
| | | 6 | | | | | | 9 | | 4 | | 1 | | | |
| A | 7 | | | | | | | E | D | | | B | | | |
| 9 | E | 7 | | 0 | | B | | | | | | | | | |
| | | | 6 | D | | | F | C | | | | | 8 | 9 | 0 |
| | 2 | | B | C | | | | | 8 | D | A | 6 | | | |
| | | 4 | | | | | | | | | 1 | | 7 | 5 | |
| | | | | | | 2 | 6 | | | | | 5 | 0 | 7 | |
| | 8 | 3 | | 9 | 7 | A | D | | C | | | F | | | |
| F | 6 | | | | E | | | | | 3 | | | | 8 | C |
| 5 | 0 | 1 | A | | | | | | B | | F | | | E | D |

## 3 Travelling salesman problem

Given is directed graph in which the vertices represent cities and the edges roads. (Since the graph is directed, roads might by one-way.) A travelling salesman plans a tour to visit every cities exactly once, and want to minimize its total travelled distance.

Fill in the template to specify a correct tour. This tour is modelled by means of a constant $start \in Vertex$ representing the starting city, and a function $next : Vertex \rightarrow Vertex$ representing the order of cities visited in the tour. Make sure to return to the starting city after visiting the final city.

You do not have to take into account (minimization of) distances yet. This will be considered in the next exercise session.

## 4 N-queens

In chess, a queen can move as far as she pleases, horizontally, vertically, and diagonally. A chess board has $n$ rows and $n$ columns. The standard $n \times n$ Queen's problem asks "how to place $n$ queens on an $n \times n$ chess board so that no queen can attach another queen in one move?".

1. Create an N-queens theory which models the problem, starting from the template.
2. Solve the problem in IDP-Z3 and compute its models. Try for $n = 4, 5$ and $8$ and interpret the results. You should obtain 92 solutions for $n = 8$.
3. What is the difference between this theory and a standard pure Prolog program?