P03 – Constraint Satisfaction Problems & Datalog

This lab has two independent tasks. In the first part, you will formulate the popular puzzle game Sudoku as a CSP and solve it using a provided library. During the second task, you will use the logic programming language Datalog to analyze a social graph.

1. Sudoku as a CSP

Use the provided template called „*sudoku_TASK.py*“ together with the library *“python-constraint-1.2”* to solve a Sudoku. Sudoku is a popular number puzzle, consisting of a 9 by 9 grid, which is made up by nine sub-squares.

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

The goal of the puzzle is to fill the remaining white spaces with numbers 1 to 9, while not violating any of these three rules:

- Each number can only be used once per row.
- Each number can only be used once per column.
- Each number can only be used once per sub-square.

There already is a Sudoku as well as some helper variables defined in the template. Formulating the Sudoku as a CSP is now up to you. All the necessary methods to solve this task are contained in this basic example:

```
>>> from constraint import *
>>> problem = Problem()
>>> problem.addVariable("a", [1,2,3])
>>> problem.addVariable("b", [4,5,6])
>>> problem.getSolutions()
[{'a': 3, 'b': 6}, {'a': 3, 'b': 5}, {'a': 3, 'b': 4},
 {'a': 2, 'b': 6}, {'a': 2, 'b': 5}, {'a': 2, 'b': 4},
 {'a': 1, 'b': 6}, {'a': 1, 'b': 5}, {'a': 1, 'b': 4}]

>>> problem.addConstraint(lambda a, b: a*2 == b,
                          ("a", "b"))
>>> problem.getSolutions()
[{'a': 3, 'b': 6}, {'a': 2, 'b': 4}]

>>> problem = Problem()
>>> problem.addVariables(["a", "b"], [1, 2, 3])
>>> problem.addConstraint(AllDifferentConstraint())
>>> problem.getSolutions()
[{'a': 3, 'b': 2}, {'a': 3, 'b': 1}, {'a': 2, 'b': 3},
 {'a': 2, 'b': 1}, {'a': 1, 'b': 2}, {'a': 1, 'b': 3}]
```

(taken from http://labix.org/python-constraint)

For a detailed Documentation of the python-constraint library see:
http://labix.org/doc/constraint/

2. Reasoning over Data using Datalog

For this task you need to install the python packages “pandas” using `conda` and “pyDatalog” using `pip` (if you are not using Anaconda, your package manager is `pip` anyway, not `conda`; on unix-based systems you might have to run: “export PATH=~/anaconda/bin:$PATH”).

In this task, you will apply your AI skills in order to help solve a crime. There have been charges of insider trading against Katarina Quandt. The board of BigCompany AG has decided on 12.2.2017 to launch a large share-buyback program in February of 2017. On 23.2.2017 Katarina Quandt has bought vast amounts of shares of the BigCompany AG, while only the company board should have known about the buyback program at that point. Is it possible that somebody gave her a hint?

You have been provided with information about calls and text messages that have been issued in February 2017. The data is in triplet form, each row contains a caller, a callee and a date.

First, have a look at this tutorial *https://sites.google.com/site/pydatalog/Online-datalog-tutorial* to get a feeling for the `pyDatalog` syntax, and then complete the template "*datalog_crimefighting_TASK.py*" to find communication links between the suspect and the company board.