SAARLAND UNIVERSITY ARTIFICIAL INTELLIGENCE
Dr. Álvaro Torralba and Prof. Wolfgang Wahlster
Dr. Cosmina Croitoru, Daniel Gnad, Marcel Steinmetz
Yannick Körber, Michael Barz
Christian Bohnenberger, Sophian Guidara, Alexander Rath,
El Khansa Rekik, Julia Wichlacz, Anna Wilhelm

Practical Sheet 1.

Solutions due Tuesday, **May 22**, 23:59, in the Moodle system.

Exercise 1. (20 Points)

In this sheet, your task is to model a Sudoku puzzle (as shown below) as a CSP problem, and to solve it using a CSP solver. To do so, you must use the Z3 theorem prover.[1][2] To run Z3, you have to create a text file in Z3's input language, which you can then pass as command line argument to the Z3 executable. The Z3 executable is available in the VM through typing `z3` in a terminal. Printing additional information about the solving process can be done through the `-st` option, e.g., `z3 -st path/to/csp.z3`. The result of Z3 (`sat` or `unsat`) will be printed to the console.

An overview of the, for this sheet, relevant Z3 language fragments is shown in Table 1. As the input language of this solver allows to represent strictly more general problems than CSPs, **you must not use any statement that is not shown in this table**. An example file (the Coloring Australia example from the lecture) is available in Moodle.

Encode the following *generalized Sudoku* puzzle as CSP using the Z3 language fragment depicted in Table 1. The goal of this puzzle is to fill the empty cells in the board with numbers from 1 to 9 complying with the constraints listed below. We use $\langle i, j \rangle$ to denote the value of the cell with $x = i$ and $y = j$.

1. Typical Sudoku constraints:
   (a) Numbers cannot be repeated in any row, column, or 3x3 square
   (b) Cells whose values are already specified must be assigned to the respective values.
2. Top middle 3x3 square: The numbers must comply with the arithmetic expressions drawn in the figure:
   (a) $\langle 3, 0 \rangle + \langle 3, 1 \rangle + \langle 3, 2 \rangle = 15$

[1] https://github.com/Z3Prover/z3

[2] An introduction can be found in http://rise4fun.com/z3/tutorial/guide.

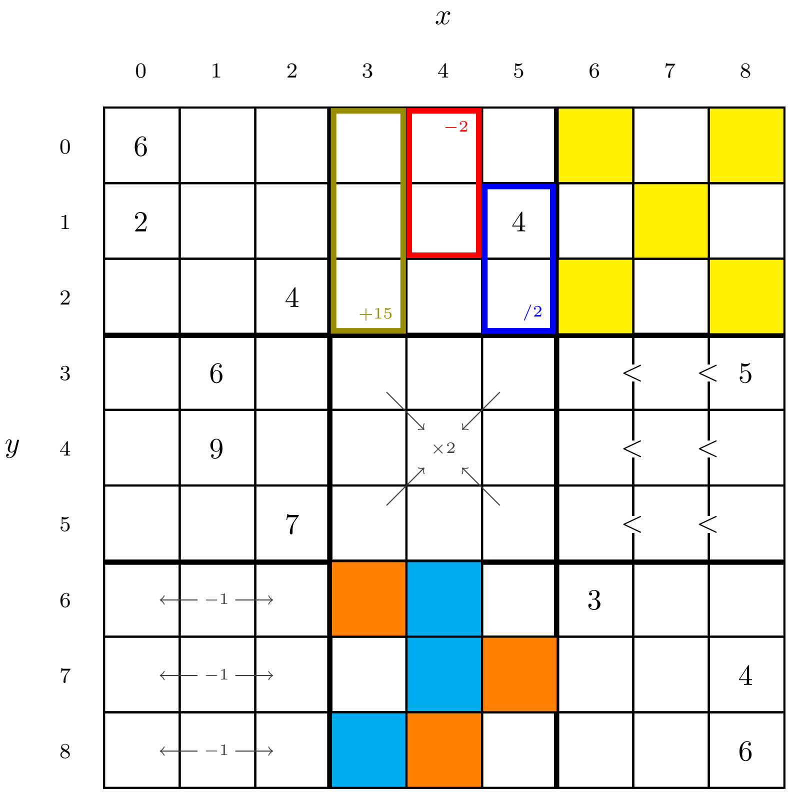

Figure 1: Illustration of the generalized Sudoku puzzle.

   (b) $\langle 4,1\rangle - \langle 4,0\rangle = 2$

   (c) $\frac{\langle 5,1\rangle}{\langle 5,2\rangle} = 2$

3. Top right 3x3 square: at most one of the yellow cells may contain a value larger than 4.

4. Center square: The sum of the corners of the center square must be equal to twice the center cell of the center square. In other words: $\langle 3,3\rangle + \langle 5,3\rangle + \langle 3,5\rangle + \langle 5,5\rangle = 2 \times \langle 4,4\rangle$

5. Middle right square: Numbers must comply with the inequalities. In particular:

   (a) $\langle 6,3\rangle < \langle 7,3\rangle < \langle 8,3\rangle$

   (b) $\langle 6,4\rangle < \langle 7,4\rangle < \langle 8,4\rangle$

   (c) $\langle 6,5\rangle < \langle 7,5\rangle < \langle 8,5\rangle$

6. Bottom left square: For every cell in this square with $x = 1$, one of the horizontally adjacent cells must equal the value minus 1. For example, for the cell $\langle 1,6\rangle$, if $\langle 1,6\rangle = 4$, then either $\langle 0,6\rangle = 3$ or $\langle 2,6\rangle = 3$.

7. Bottom middle square: The values of the orange cells must be either all odd or all even. Similarly for the blue cells. Moreover, if the orange cells contain odd numbers, then the blue cells must contain even numbers. If the orange cells contain even numbers, then the blue cells must contain odd numbers.

Your task is to:

(a) Implement the constraints into the template file `sudoku.z3` that we provide. There, we already included the basic definition of the Sudoku board, and print the values of all cells before Z3 terminates. You can refer to the value of the cell $\langle i, j\rangle$ through `(Board x`*i* `y`*j*`)` (as examplified in the `get-value` statements).

(b) Run Z3 on your file. Save the output in a file called "sudoku.log". Note: If the output of the solutions takes extremely long, or there is no solution, it is very likely that you have a bug in your model.

IMPORTANT Please add comments to your model, i.e., for each of the constraint types (1-7), you should have a different section in your model. Briefly explain in the comments how you encoded each of the constraints. **We will subtract points if you do not put any comments into your model, or your model is difficult to understand with the given comments.**

Submission instructions Put your `sudoku.z3` and `sudoku.log` files into an archive called "name1-name2-name3.zip", where "name1", "name2", "name3" are the family names of all authors. Additionally, **add a file called "authors.txt"** to the archive that contains one line per author, detailing the full name and matriculation number. To upload the archive to the Moodle system, go to the corresponding assignment, click "Add submission", and upload it. Note that only one author per group needs to do the submission.

Statement	Description
General	
`(check-sat)`	Checks whether the CSP defined up to this point is satisfiable.
`(declare-const var Int)`	Declares a new variable with name `var`.
`(assert E)`	Adds boolean expression `E` as constraint.
`(get-value (E))`	Prints the value of `E`, where `E` can be an arbitrary expression such as constant, variable, function, or mathematical or boolean combination thereof (must occur after `(check-sat)`).
`(get-model)`	Prints all variable assignments (must occur after `(check-sat)`).
`(echo "message")`	Prints `message` to the console.
`; This is a comment`	Commenting.
Mathematical Expressions	
c	Constants $c \in \mathbb{Z}$.
`var`	Evaluates to the value of variable `var`.
`(fn 1 2)`	Evaluates to the value $\texttt{fn}(1, 2)$ of the function with name `fn`.
($\circ$ E_1 ... E_n)	Evaluates to $E_1 \circ E_2 \circ \cdots \circ E_n$, where $\circ$ can be any of `+`, `-`, and `*`.
Boolean Expressions	
`true`	Constant for true.
`false`	Constant for false.
(`not` E)	Negation of the boolean expression E.
(`and` E_1 ... E_n)	Conjunction over the boolean expressions E_1 to E_n.
(`or` E_1 ... E_n)	Disjunction over the boolean expressions E_1 to E_n.
($\circ$ E_1 ... E_n)	Is true iff for the evaluation of the expressions E_1 to E_n, it holds that $E_1 \circ E_2$ and $E_2 \circ E_3, \ldots,$ and $E_{n-1} \circ E_n$, where $\circ$ can be any of `<`, `<=`, `>`, `>=`, and `=`.
(`distinct` E_1 ... E_n)	Is true iff every expression E_1 to E_n evaluates to a different value.

Table 1: Z3 input language fragment you may use for the CSP modeling exercises.