Formal Methods in Software Engineering WS 2019/20 Tuesday 5$^{\text{th}}$ November, 2019

Exercise sheet 1

Submission deadline: Tuesday 12$^{\text{th}}$ November, 2019

Submission: by e-mail to paul.wild@fau.de. Feel free to ask questions!

Fixpoints in CTL

1. Let S be a set and $F\colon 2^S \to 2^S$ be a monotone function and let νF be its greatest fixpoint. Show that the function
$$G\colon 2^S \to 2^S, \quad G(A) = S \setminus F(S \setminus A)$$
(where $S \setminus A$ denotes the complement of A in S) has a least fixpoint μG, and moreover
$$\mu G = S \setminus \nu F.$$

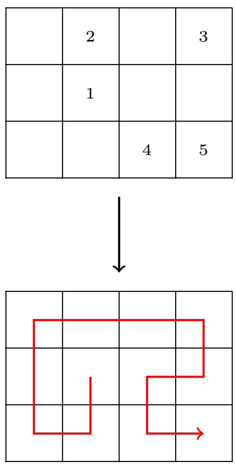
2. Use the Knaster-Tarski theorem directly to show that for any model $\mathcal{M}$ and any pair of CTL formulas φ and ψ, $[\![\mathsf{A}[\varphi \,\mathsf{R}\, \psi]]\!]^{\mathcal{M}}$ is the greatest fixpoint of
$$F\colon 2^S \to 2^S, \quad F(A) = [\![\psi]\!]^{\mathcal{M}} \cap ([\![\varphi]\!]^{\mathcal{M}} \cup (\mathsf{AX})A).$$

3. Using the logical equivalence $\mathsf{E}[\varphi \,\mathsf{U}\, \psi] \equiv \neg\mathsf{A}[\neg\varphi \,\mathsf{R}\, \neg\psi]$, prove the fixpoint characterization of $[\![\mathsf{E}[\varphi \,\mathsf{U}\, \psi]]\!]^{\mathcal{M}}$ from the lecture (November 4, slide 9) as the least fixpoint of
$$G\colon 2^S \to 2^S, \quad G(A) = [\![\psi]\!]^{\mathcal{M}} \cup ([\![\varphi]\!]^{\mathcal{M}} \cap (\mathsf{EX})A).$$

Number puzzle

Consider a board of 8×6 fields, where some fields are labelled with numbers ranging from 1 to 11, like depicted here on the left:

	A	B	C	D	E	F	G	H
6		10						
5		11						
4		9		1		6	7	
3	3	4			8			
2			5					
1				2				



The goal of this game is to cover the whole field with stones, at the same time respecting the sequence of numbers on the board. One starts with a single stone on the field with the lowest number. In each step, one can then put a new stone on a field adjacent to the previous one. E.g., in the field above, one starts on the field D4 and then one can proceed to either D3, C4, D5, or E4. However, the next *labelled* field to be covered needs to be D1, and so on. A full solution for a smaller problem is found in the right-hand side of the picture.

Your task is to model the problem above (for the initial field given in the picture on the left) in Promela and find a solution. Some hints:

- The easiest way to create a multi-dimensional array is probably to just create a onedimensional array and do some index arithmetics. You can use a #define for this. Multidimensional arrays as described in the Spin manual do not seem to work with ltsmin.
- In general, you can use #defines to significantly shorten your code. Just beware that spins is not always happy to compile your code if you use macros with arguments. Renaming some variables/arguments seems to help.
- Please describe how to obtain the solution from your model, ideally with some grep command that should be used on the output of ltsmin-printtrace.