Technische Universität München (I7)
J. Esparza / M. Blondin / P. Ashok

Summer Semester 2018
26.04.2018

Model Checking – Exercise sheet 3

General tips on Spin

Use the command `spin filename.pml` to simulate the model and `spin -a filename.pml` to generate the verifier `pan.c`. This can be compiled into an executable `pan` by using the command `gcc -o pan pan.c` and can be run using the command `./pan`. If you want to check LTL properties using `pan`, then you also need to use the option `-a` (find acceptance cycles) and sometimes the option `-mN` (where `N` is the maximum search depth, eg. `100000`). If you have multiple LTL properties, they can be chosen using the option `-N` followed by the name of the property (eg. `p1`).

An important thing to keep in mind is that warnings, errors and so on appear at the top of the messages printed by `pan`. In order to simulate a trail, use `ispin`'s simulation tab. Press the 'Re(run)' button followed by 'Rewind'. After that you may use the 'Step forward/backward' buttons to navigate the counter-example.

Exercise 3.1

The general solution for this exercise can be downloaded from the 'Exercise' section of the course page. The following are some useful tips/hits for each sub-question.

1. C-style `printf` can be used to print out values.
2. A `do` loop can be used as shown in the sample solution, but you may also use a `select (n : 1 .. 30000)` statement to non-deterministically choose `n`.
3. Labels are useful features when performing LTL model checking and you may use them to 'label' lines or sections of code. Refer to `http://spinroot.com/spin/Man/labels.html` for details.
4. In Promela, expressions such as `a == b` can also be statements. Execution would be blocked at such statements until the expression is satisfied. We use this as well as the fact that processes always interleave in order to implement the solution for this exercise.
5. As mentioned by a student in class, the LTL next operator (`X`) is not available by default in Spin as its use makes the partial order reduction technique invalid. For details, see `http://spinroot.com/spin/Man/ltl.html`. The solutions are

```
ltl p1 { [] (odd@update -> X((!odd@update) U even@update)) }
ltl p2 { [] (even@update -> X((!even@update) U odd@update)) }
ltl p3 { [] (original > 0 -> (original * original) > n) }
ltl p4 { (n%2 == 0) U (n == 1) }
```

   The last property could be interpreted in various ways and has various solutions.

Comment: Those who attended the tutorial may remember that `p4` *was being satisfied by the sample program. The reason happens to be that* `n` *is a global variable which is initialized to 1 inside the* `init { ... }` *block, because of which, from the beginning,* `n == 1` *was satisfied which implies that* `p4` *is satisfied. Mystery solved!*

Exercise 3.2

As this exercise was not completed in class, we recommend you try to solve it yourself at home. The sample solution would be uploaded at a later point of time.