Technische Universität München **Summer Semester 2014**
I7
J. Esparza / A. Durand-Gasselin **11.04.2014**

Model Checking – Exercise sheet 1

Exercise 1.1

Alice and Bob play a tournament of heads and tails: they toss a coin repeatedly, for each toss whoever wins is awarded one point. The tournament ends when one of them has a lead of 3 points. For all tosses Alice calls heads and Bob call tails. Draw the corresponding transition system.

Exercise 1.2

We give the following C function:

```
1 void syra(char n) {
2   while (n > 1) {
3     if (n % 2)
4        n = 3* n +1;
5     else
6        n = n /2;
7   }
8 }
```

1. Draw the transition system for the states reachable from 7, 11 and 13
2. Does this program always terminate ?

Exercise 1.3

1. Download and install Spin from `http://www.spinroot.com/`.
2. Run Spin, following instructions from `Spin/Test/README_tests.txt`, tests 1 to 5.
3. Run spin on the same programs using ispin (from `http://spinroot.com/spin/Src/ispin.tcl`)

   You will need `wish` which is part of tcl/tk which you can download from: `http://www.tcl.tk/software/tcltk/8.5.html`

Remark: The following links contain most of the information you might need about promela (Spin's modeling language):
`http://spinroot.com/spin/Man/Manual.html`
`http://spinroot.com/spin/Man/Quick.html`
`http://spinroot.com/spin/Man/grammar.html`

Exercise 1.4

1. Write in promela the function given in exercise 1.2 as a process.
2. How to check for termination of the function ?
3. Write in promela using a global variable n and two proesses: one that handles odd values and one that handles even values.

Exercise 1.5

We give the following program written in Promela. Explain the behaviour of this program (provided all assertions are satisfied). Are all assertions always holding ? Propose a way to ensure those.

```
#define N 50
int pos;
bool tokenv[N];

proctype moveLeft () {
  int i;
  do ::
    i = pos + N-1 % N;
    printf("Moving left\n");
    tokenv[pos] = false;
    tokenv[i] = true;
    pos = i;
  od;
}
proctype moveRight () {
  int i;
  do ::
    i = pos + 1 % N;
    printf("Moving right\n");
    tokenv[pos] = false;
    tokenv[i] = true;
    pos = i;
  od;
}
```

```
proctype display () {
  do ::
    assert tokenv[pos];
    printf("Token at %d\n", pos);
  od;
}
init {
  pos = 1;
  tokenv[pos] = true;
  run moveLeft ();
  run moveRight ();
  run display ();
}
```