Technische Universität München **Winter term 2017/18**
I7
Prof. J. Esparza / Dr. M. Blondin **16.12.2017**

Automata and Formal Languages — Homework 9

Due 19.12.2017

Exercise 9.1

Suppose there are n processes being executed concurrently. Each process has a critical section and a non critical section. At any time, at most one process should be in its critical section. In order to respect this mutual exclusion property, the processes communicate through a channel c. Channel c is a queue that can store up to m messages. A process can send a message x to the channel with the instruction $c \,!\, x$. A process can also consume the first message of the channel with the instruction $c \,?\, x$. If the channel is full when executing $c \,!\, x$, then the process blocks and waits until it can send x. When a process executes $c \,?\, x$, it blocks and waits until the first message of the channel becomes x.

Consider the following algorithm. Process i declares its intention of entering its critical section by sending i to the channel, and then enters it when the first message of the channel becomes i:

```
1 process(i):
2     while true do
3         c ! i
4         c ? i
5         /* critical section */
6         /* non critical section */
```

(a) Sketch an automaton that models a channel of size $m > 0$ where messages are drawn from some finite alphabet Σ.

(b) Model the above algorithm, with $n = 2$ and $m = 1$, as a network of automata. There should be three automata: one for the channel, one for `process`(0) and one for `process`(1).

(c) ★ Use `Spin` to simulate and verify `naive_mutex.pml`.

(d) Construct the asynchronous product of the network obtained in (b).

(e) Use the automaton obtained in (d) to show that the above algorithm violates mutual exclusion, i.e. the two processes can be in their critical sections at the same time.

(f) Design an algorithm that makes use of a channel to achieve mutual exclusion for two processes ($n = 2$). You may choose m as you wish.

(g) Model your algorithm from (f) as a network of automata.

(h) ★ Model your algorithm from (f) in Promela. Use `Spin` to simulate and verify the algorithm.

(i) Construct the asynchronous product of the network obtained in (g).

(j) Use the automaton obtained in (i) to show that your algorithm achieves mutual exclusion.

Exercise 9.2

In the previous question, we have seen that mutual exclusion can be achieved by communicating through channels. Let us now consider processes communicating through shared variables. Suppose there are two processes sharing a variable x initialized to 0. Mutual exclusion can be achieved using the following algorithm:

```
1 process(i):
2     while true do
3         while x = 1 − i do
4             skip
5         /* critical section */
6         x ← 1 − i
7         /* non critical section */
```

(a) Model the above algorithm as a network of automata. There should be three automata: one for the channel, one for `process`(0) and one for `process`(1).

(b) Construct the asynchronous product of the network obtained in (a).

(c) ★ Use `Spin` to simulate and verify `mutex.pml`.

(d) Use the automaton obtained in (b) to show that the algorithm achieves mutual exclusion.

(e) If a process wants to enter its critical section, is it always the case that it will eventually enter it? You should reason in terms of infinite executions.

Exercise 9.3

The following algorithm attempts to achieve mutual exclusion for two processes. The processes share variables b_0, b_1 and k initialized respectively to *false*, *false* and 0.

```
1  process(i):
2      while true do
3          b_i ← true
4          while k ≠ i do
5              while b_{1−i} do
6                  skip
7              k ← i
8          /* critical section */
9          b_i ← false
10         /* non critical section */
```

(a) Model the above algorithm as a network of automata.

(b) Find an execution where both processes end up in their critical sections at the same time.

(c) ★ Model the algorithm in Promela.

(d) ★ Can you find an execution violating mutual exclusion by simulating the algorithm with `Spin`?

(e) ★ Verify the algorithm with `Spin`.

Exercise 9.4

The following algorithm attempts to achieve mutual exclusion for two processes. These processes share variables x, y and z which are initialized to 0.

```
1  process(i):
2      while true do
3          /* non critical section */
4          x ← i + 1
5          if y ≠ 0 and y ≠ i + 1 then
6              goto 4
7          z ← i + 1
8          if x ≠ i + 1 then
9              goto 4
10         y ← i + 1
11         if z ≠ i + 1 then
12             goto 4
13         /* critical section */
```

(a) Model the above algorithm as a network of automata.

(b) ★ Find an execution where both processes end up in their critical sections at the same time.

(c) ★ Model the algorithm in Promela.

(d) ★ Can you find an execution violating mutual exclusion by simulating the algorithm with `Spin`?

(e) ★ Verify the algorithm with `Spin`.