Verification Techniques 2007, Homework 4

The homework till Friday, June 1 is to hand in exercises A, B.

1.

**Since.** Consider the new binary temporal operator $\text{S}$, pronounced “since”. Intuitively, since is the “backwards” analogue of (strong) until. That is, $\phi_1 \text{S} \phi_2$ means that the last occurrence of $\phi_2$ was followed by a period of $\phi_1$ up to the present from the state after that where $\phi_2$ held. The formal semantics can be described as

- $(\sigma, i) \models \phi_1 \text{S} \phi_2$ iff $\exists j \leq i : (\sigma, j) \models \phi_2$ and $\forall k : j < k \leq i (\sigma, k) \models \phi_1$

Your problem is the following:

a) Express $\square (p \Rightarrow p \text{S} q)$ as a formula containing $p$, $q$, and the other temporal operators that we have used ($\diamond$, $\square$, $\triangledown$, $\lozenge$, $\cup$, $\omega$).

b) Draw a Büchi automaton that accepts the language that satisfies $\square (p \Rightarrow p \text{S} q)$.

c) Make a **never**-claim in PROMELA that will check whether a program satisfies $\square (p \Rightarrow p \text{S} q)$.

2.

Below is, in pseudocode, a mutual algorithm, due to Burns, for solving the mutual exclusion problem between an arbitrary number of processes. The processes are number from 1 to $N$. Below is the pseudocode executed by process $i$. Each process $i$ has a variable $\text{flag}(i)$, which can be written by process $i$, and read by any process.

```
enter the trying section
L  \text{flag}(i) := 0
   for j from 1 upto i - 1 do
      if $\text{flag}(j) = 1$ then goto L
      $\text{flag}(i) := 1$
   for j from 1 upto i - 1 do
      if $\text{flag}(j) = 1$ then goto L
M  for j from i + 1 upto N do
   if $\text{flag}(j) = 1$ then goto M
   *** critical region ***
   $\text{flag}(i) := 0$
goto beginning
```

Your task is to model this algorithm in Promela, using only small atomic statements (either reading or writing to a local variable) and to use SPIN to investigate whether the algorithm satisfies the following properties:
• mutual exclusion, i.e., that at most one process can be in its critical region at any time

• freedom of starvation, i.e., that whenever a process has entered the protocol, it will at some later point reach $M$. This property might be satisfied differently for different process indices $i$.

Your solution should contain beautiful Promela models, describing how you added checks, and results of SPIN runs. If the property is satisfied, use a not too small value of $N$. If the property is not satisfied, show a shortest error-trace leading to a bad state or a bad loop with as few processes as possible.