Verification of Cyber-Physical System
Fall 2017

---

Exercice Sheet 7

Author : Sylvain Julmy

---

Professor : Ultes-Nitsche Ulrich
Assistant : Prisca Dotti

---

## Exercice 1

The two timed automaton from figures 1 and 2 are an implementation with `UPPAAL` of the `Promela` model given in exercice sheet :

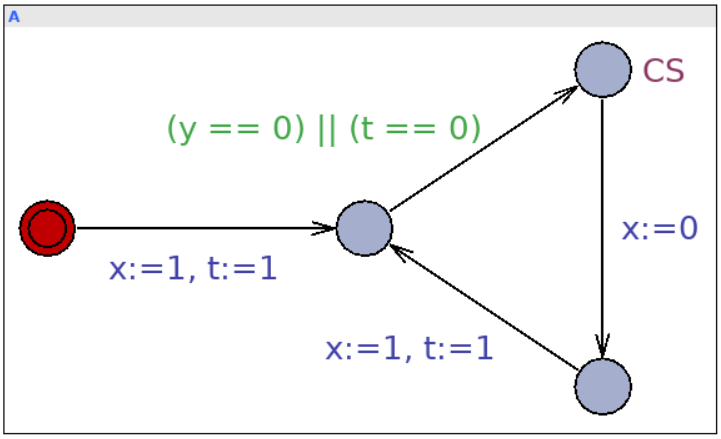


Figure 1: Process A

Both process use the same concept and only the variable used are different. At first, we start in an state that is unreachable from the other states of the automaton, this state is used in order to initiliase the automaton variables.

The second state reached is representing the wait of a process, and only when the corresponding property for each process holds, the transition can be taken.

The state named $CS$ is representing the critical section for each process, and none of those can access the critical section at the same time.

The incoming state from $CS$ is representing the transition from `x=false` and `x=true` in process $A$, we have to use such an additional state because we have to represent the possibility that $B$ is making progress between the two assignment `x=false` and `x=true` (the same holds for process $B$ w.r.t. process $A$).

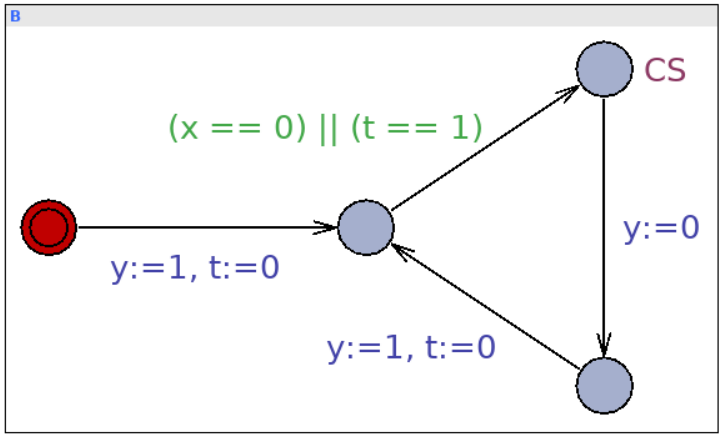


Figure 2: Process B

We also use the following declaration (listing 1) and composition (listing 2) for the system :

```
bool x,y,t;
```

Listing 1: Declarations

```
// Place template instantiations here.
A = ProcessA();
B = ProcessB();
// List one or more processes to be composed into a system.
system A,B;
```

Listing 2: Composition

In order to verify our system, we could use the verifier with the formula

`A[] not(A.CS and B.CS)`

which verify that for all possible path (`A`), this is always the case (`[]`) that `A` and `B` are not in the critical section at the same time (`p.CS` where `p` is the corresponding process).

## Exercice 2

$x$ is a clock of the system, at each step, the clock would either be 0 or greater than 2, the transition from $Idle$ to $Taken$ of process $B$ is taken only when receiving something from chanel $reset$, then the only possible transition after receiving something is the one from $Taken$ to $Idle$, where the clock is reset.

The only constraint we have is that $x$ have to be greater or equals to 2 in order to send/receive from the $reset$ chanel.

Then, we want to modify the model in order to have the clock reset only when the clock value is either 2 or 3.

### Model proposition 1

With the first proposition, we add the invariant $x \leq 3$ to the state *loop*, so the system have to leave the state when $x \leq 3$ would become $false$.

This proposition meet the requirement, because when $x$ would become higher than 3, the system is forced to leave the $Idle$ state and no deadlock is possible.

### Model proposition 2

With the second proposition, we modified the guard of the *reset* send from $x \geq 2$ to $x \geq 2$ && $x \leq 3$. It means that the clock can be reset only if $x$ is between 2 and 3.

This proposition does not meet the requirement, because when $x$ would become higher than 3, the system is not forced to leave the $Idle$ state. If the state is not leave, the system will entire in a deadlock because there is no available transition.