Homework 1

To hand in on September 27th at the beginning of the exercise session, or by mail (before 14:00) at `marie.fortin@lsv.fr`.

Exercise 1 (Mutual Exclusion)**.**

1. The following program is a mutual exclusion protocol for two processes due to Pnueli. There is a shared boolean variable s, initialized to 1, and two shared boolean variables y_i, i in $\{0, 1\}$, initialized to 0. Each process P_i can read the values of s, y_0, and y_1, but only write a new value in s and y_i. Here is the code of process P_i in C-like syntax:

   while (true)
   {
   /* *1: Noncritical section.* */
   atomic { $y_i = 1$; $s = i$; };
   /* *2: Wait for turn.* */
   wait until (($y_{1-i} == 0$) || (s != i));
   /* *3: Critical section.* */
   $y_i = 0$;
   }

   Draw the transition system of each process, and construct their parallel composition. Label the states appropriately using the atomic propositions w_i and c_i, holding when process P_i is waiting or in the critical section, respectively.

2. Does the algorithm ensure *mutual exclusion*, i.e. that the two processes can never be simultaneously inside the critical section?

3. Does the algorithm ensure *starvation freedom*, i.e. that every waiting process will eventually access the critical section, provided that the other process does not stay forever inside the critical section?

Exercise 2 (Vending Machines)**.** Let $C \subseteq \mathbb{N}$ be a finite set of *coin denominations* (for instance, $C = \{5, 10, 20, 50, 100, 200\}$), and P a finite set of *products* (for instance, $P = \{\text{coffee}, \text{tea}\}$). We call *vending machine* any program m following the syntax below:

$$
\begin{array}{rll}
m ::= & \texttt{deliver}(p) & /*\ \textit{Deliver a product}\ */\\
& \mid \texttt{count} := \texttt{count} - c & /*\ \textit{Return a coin}\ */\\
& \mid \texttt{req} := \bot & /*\ \textit{Erase last request}\ */\\
& \mid \texttt{if}\ \textit{cond}\ \{m\} \mid \texttt{while}\ \textit{cond}\ \{m\} \mid m\ ;\ m & \\
\textit{cond} ::= & \texttt{true} \mid \texttt{count} \geq n & \\
& \mid \texttt{req} = p & /*\ \textit{Button for p pressed}\ */\\
& \mid \texttt{req} = \texttt{cancel} & /*\ \textit{Cancel button pressed}\ */\\
& \mid \texttt{req} = \bot & /*\ \textit{No button pressed}\ */
\end{array}
$$

where $p \in P$, $c \in C$, $n \in \mathbb{N}$. A *user* is a program

$$
\begin{array}{rll}
u ::= & \texttt{count} := \texttt{count} + c & /*\ \textit{Insert a coin}\ */\\
& \mid \texttt{req} := p \mid \texttt{req} := \texttt{cancel} & /*\ \textit{Push a button}\ */\\
& \mid \texttt{await}(p) & /*\ \textit{Wait until}\ \texttt{deliver}(p)\ \textit{is performed}\ */\\
& \mid u\ ;\ u &
\end{array}
$$

1. Show that any vending machine m can be modeled by a transition system with variables (see lecture notes) M_m with a finite set of states, one integer variable `count`, and one variable `req` with a finite domain.

2. Give a transition system M_{users} modeling the set of all executions of *all* possible users: for all users u, any execution of u should correspond to some execution of M_{users}, and conversely, any execution of M_{users} should correspond to an execution of some user u.

3. (a) Let $M = (S, \Sigma, (\texttt{count}, \texttt{req}), (\mathbb{N}, P \uplus \{\texttt{cancel}\}), T, I, \mathrm{AP}, \ell)$ be a transition system with one integer variable `count`, one variable `req` with domain $P \uplus \{\texttt{cancel}\}$, a finite number of states and transitions, and such that all guards on `count` appearing in M are of the form $\texttt{count} \geq i$ or $i \leq \texttt{count} \leq j$ for some constants $i, j \in \mathbb{N}$, and all updates of `count` are of the form $\texttt{count} := \texttt{count} - c$ or $\texttt{count} := \texttt{count} + c$ for some $c \in C$. Show that the following problem is decidable:
   Input: M as described above, and $s \in S$.
   Question: Does there exist ν such that (s, ν) is reachable in M?

   (b) Deduce that the satisfaction of the following safety property by a given vending machine m is decidable: "a coffee cannot be delivered if less than 50c have been inserted".

4. (Bonus) Show that the following problem is decidable: given a vending machine m, is it always the case that if the coffee button is pressed after exactly 50c have been inserted, and no other button is pressed later, then eventually the machine gives out coffee?

5. Write a program m satifying these two properties.