TP noté

22 janvier 2018

Vérification de l'algorithme de Szymanski en Cubicle

Le but de cet exercice est de modéliser l'algorithme d'exclusion mutuelle de Boleslaw K. Szymanski [1988] donné ci-dessous. Pour rentrer en section critique, chaque processus `x` exécute le code suivant :

```
L0 : Bx := true
L1 : await (forall y. x <> y => not Sy) then Bx := false
L2 : Wx := true ; Sx := true ;
L3 : if (exists y. x <> y /\ not By /\ not Wy)
       then Sx := false ; goto 4
       else Wx := false ; goto 5
L4 : await (exists y. x <> y /\ Sy /\ not Wy) then Wx := false ; Sx := true;
L5 : await (forall y. x <> y => not Wy)
L6 : await (forall y. y < x => not Sy)
L7 : {Critical section}
     Sx := false ; goto 0
```

Explications : chaque processus `x` dispose de trois variables booléennes locales `Bx` (lire « la variable `B` de `x` »), `Sx` et `Wx`. Les lignes $\texttt{L}_i$ correspondent à des points de programmes *atomiques*. Comme pour l'exercice 1, l'instruction `await p` signifie « attendre que `p` soit vraie pour continuer ». La variante `await p then i` attend que `p` soit vraie et, quand c'est le cas, exécute `i` de manière atomique.

Questions.

1. Écrire un programme Cubicle afin de modéliser l'algorithme de Szymanski.
2. Écrire une propriété pour vérifier que cet algorithme est sûr, c'est-à-dire qu'il n'y a jamais deux processus en section critique en même temps.
3. Modifier le fichier Cubicle afin de vérifier que l'arrêt d'un processus, en dehors de la section critique, n'empêche pas un autre processus d'y entrer.