Université Paris Diderot – Master 1 II

Théorie et pratique de la concurrence

Partiel du 30 mars 2010

Durée : 1h30. Tous les documents sont autorisés. Le barème est indicatif.

Question 1 : (7 points)

Soit le programme Promela suivant :

```
int n=0;

active proctype P() {
int ptmp, i;
p1: i=0;
p2: do :: i < K ->
p3:           ptmp = n;
p4:           n = ptmp + 1;
p5:           i++
p6:    :: else -> skip
p7: od
}

active proctype Q() {
int qtmp, j;
q1: j=0;
q2: do :: j < K ->
q3:           qtmp = n;
q4:           n = qtmp - 1;
q5:           j++
q6:    :: else -> skip
q7: od
}
```

1. Si `K` est positif, quelles sont les valeurs possibles de `n` ?
2. Ecrire en LTL les propriétés suivantes en explicitant les propositions atomiques utilisées :
   (a) "`n` ne prend jamais la valeur `2K`"
   (b) "`n` prend la valeur 0 un nombre fini de fois"
   (c) "si `n` vaut 1, il prendra la valeur 2 un jour"
   (d) "`n` prend la valeur 1 après avoir eu la valeur 0"
   (e) "`n` prend toutes les valeurs entre 0 et 2"
3. Pour chaque propriété ci-dessus, indiquer sa classe (vivacité, sûreté ou les deux) et sa valeur de vérité pour le programme considéré.

Question 2 : (7 points)

Supposons qu'une machine fournit l'instruction atomique suivante :

```
TS(int common, int local) :
      atomic { local = common;
               common = 1 }
```

Cette instruction est utilisée pour implementer l'exclusion mutuelle entre deux processus comme suit :

```
int common = 0; /* variable partagee */

-- Processus 1:                    -- Processus 2:
int local1;                        int local2;
loop forever :                     loop forever :
l1:  section NC                    l1:  section NC
     repeat                             repeat
l2:    TS(common, local1);         l2:    TS(common, local2);
l3:  until local1 = 0;             l3:  until local2 = 0;
l4:  section critique              l4:  section critique
l5:  common = 0;                   l5:  common = 0;
```

1. Expliquer (10 lignes maximum) comment fonctionne cet algorithme.
2. Montrer, en raisonnant sur l'algorithme, que cet algorithme satisfait les propriétés d'exclusion mutuelle et d'absence d'inter-blocage.
3. Qu'en est-il des propriétés d'absence de famine et d'attente bornée ?
4. Y-a-t-il un processus privilégié ?

Question 3 : (7 points)

Soit l'algorithme suivant :

```
semaphore S=1, T=0;

active proctype P() {              active proctype Q() {
p1: wait(S);                       q1: wait(T);
p2: printf("p");                   q2: printf("q");
p3: signal(T)                      q3: signal(S);
}                                  }
```

1. Quels sont les affichages possibles de ce programme ?
2. Quels sont ces affichages si on commente l'instruction `wait(S)` ?
3. Quels sont ces affichages si on commente l'instruction `wait(T)` ?
4. Y-a-t-il un changement de comportement si les deux sémaphores sont des sémaphores binaires ?
5. Proposer une implementation en Promela pour les sémaphores (non binaires) généralisés.