
Delphine Longuet, Prof. Burkhart Wolff
Parc Orsay Université
4, rue Jacques Monod
Building H / Room 012

TD 1

Date : 22.09.2009, Durée : 3 heures

Exercice 1 (Modèles de Variables et de Programmes)

Soit le programme P suivant :

```
while true do
  if not B then
    B := true;
    proc;
    B := false;
  end if
end while
```

Questions :

1. Donner l'automate AP modèle de ce programme.
2. Donner l'automate AB modèle d'une variable booléenne B.
3. Construire le produit synchronisé de ces automates. On suppose que B est initialisée à false.

Note. Pour construire un automate, on commence par établir la liste de ses transitions, et donc de ses états; c'est seulement dans les cas très simples qu'on peut le dessiner directement.

Exercice 2 (Modèles de Variables et de Programmes)

Cet algorithme, un classique, assure l'exclusion mutuelle entre deux processus. Il utilise trois variables globales : deux variables booléennes D0 et D1 initialisées à **false** et une variable entière TOUR qui ne peut prendre que les valeurs 0 et 1, initialisée à 0.

Le processus P0 exécute le code suivant :

```
PROCESS P0 :  
while true do  
  // section non critique  
  D0 := true ;  
  TOUR := 0 ;  
  attendre (D1 = false or TOUR = 1) ;  
  // section critique  
  D0 := false ;  
end while
```

Le processus P1 exécute le code symétrique obtenu en permutant 0 et 1 :

```
PROCESS P1 :  
while true do  
  // section non critique  
  D1 := true ;  
  TOUR := 1 ;  
  attendre (D0 = false or TOUR = 0) ;  
  // section critique  
  D1 := false ;  
end while
```

Questions :

1. Donner les automates AP0 et AP1. Préciser leur état initial.
2. Rappeler les automates AD0 et AD1 ; donner l'automate ATOUR.
3. Quelles sont les contraintes de synchronisation sur ces cinq automates ? Construire le produit synchronisé.
4. Donner une variante des automates AP0 et AP1 qui modélise une attente active.

Exercice 3 (Algorithme d'exclusion mutuelle naif)

Soit l'algorithme d'exclusion suivant :

```
flag[0] = 0 ;  
flag[1] = 0 ;
```

```
PROCESS P0
```

```
A : flag[0] = 1 ;
```

```
B :
```

```
while flag[1] do
```

```
    ; // do nothing
```

```
end while
```

```
C : critical section
```

```
D : flag[0] = 0 ;
```

```
PROCESS P1
```

```
A : flag[1] = 1 ;
```

```
B :
```

```
while flag[0] do
```

```
    ; // do nothing
```

```
end while
```

```
C : critical section
```

```
D : flag[1] = 0 ;
```

Question : Cet algorithme respecte t'il :

1. l'exclusion mutuelle ?
2. l'absence d'interblocage ?
3. l'absence de famine ?