

# Théorie et pratique de la concurrence – Master 1 Informatique

## TD 1 : Concurrency et LTL

### Exercice 1 :

*Algorithmes concurrents et leur propriétés*

Soit une fonction  $f$  pour laquelle il existe une valeur  $i$  telle que  $f(i) = 0$ . Les algorithmes concurrents ci-dessous cherchent une telle valeur  $i$  (appelée aussi le point zéro) en divisant l'espace de recherche en deux parties : les points zéro positifs  $i \geq 0$  et les points zéro négatifs  $i < 0$ .

Un algorithme est correct si, pour toutes les exécutions, les deux processus terminent après que l'un d'entre eux a trouvé un point zéro. Pour chaque algorithme, prouvez qu'il est correct en utilisant les diagrammes d'états ou trouvez une exécution incorrecte.

### Algorithme Zéro A.

```
boolean found;
process P()
    integer i:=0;
p1: found:=false;
p2: while not found
p3:   i:=i+1
p4:   found:=(f(i)==0)
process Q()
    integer j:=1;
q1: found:=false;
q2: while not found
q3:   j:=j-1
q4:   found:=(f(j)==0)
```

### Algorithme Zéro B.

```
boolean found:=false
process P()
    integer i:=0;
p1: while not found
p2:   i:=i+1
p3:   found:=(f(i)==0)
process Q()
    integer j:=1;
q1: while not found
q2:   j:=j-1
q3:   found:=(f(j)==0)
```

### Algorithme Zéro C.

```
boolean found:=false
process P()
    integer i:=0;
p1: while not found
p2:   i:=i+1
p3:   if (f(i)==0)
p4:     found:=true
process Q()
    integer j:=1;
q1: while not found
q2:   j:=j-1
q3:   if (f(j)==0)
q4:     found:=true
```

Dans les algorithmes suivants, on utilise la construction **await B then R** qui bloque l'exécution de R jusqu'à ce que la condition booléenne B soit vraie. De plus, l'exécution de R est faite atomiquement après le test de B. Si la condition B est évaluée (atomiquement) à faux, alors le processus qui exécute **await** est suspendu et il peut réessayer plus tard d'exécuter **await**.

### Exercice 2 :

*Sûreté et vivacité*

Pour chacune des phrases suivantes, indiquez quel type de propriété (sûreté ou vivacité) elle exprime en identifiant bien la partie qui y est intéressante :

1. Au plus 5 personnes sont dans l'ascenseur à un moment donné.
2. Les patrons sont servis dans l'ordre de leur arrivée.
3. Le coût de la vie ne décroît jamais.

### Algorithme Zéro D.

```
boolean found:=false
integer turn:=1
process P()
    integer i:=0;
p1: while not found
p2:   await turn==1 then turn:=2
p3:   i:=i+1
p4:   if (f(i)==0)
p5:     found:=true
process Q()
    integer j:=1;
q1: while not found
q2:   await turn==2 then turn:=1
q3:   j:=j-1
q4:   if (f(j)==0)
q5:     found:=true
```

### Algorithme Zéro E.

```
boolean found:=false
integer turn:=1
process P()
    integer i:=0;
p1: while not found
p2:   await turn==1 then turn:=2
p3:   i:=i+1
p4:   if (f(i)==0)
p5:     found:=true
p6:   turn:=2
process Q()
    integer j:=1;
q1: while not found
q2:   await turn==2 then turn:=1
q3:   j:=j-1
q4:   if (f(j)==0)
q5:     found:=true
q6:   turn:=1
```

4. Toute bonne chose a une fin.
5. Un livre s'améliore à chaque lecture.
6. Ce qui croit doit décroître.
7. Si deux processus attendent pour leur section critique, exactement un va y entrer.
8. Au plus une personne doit parler à un moment, les autres doivent écouter.
9. Si une interruption arrive, alors un message sera affiché.
10. Si une interruption arrive, alors un message sera affiché dans une seconde.

#### Exercice 3 :

#### *Equivalences de formules LTL*

Dire si les formules suivantes sont valides. Si oui prouvez-le, si non montrez un contre-exemple :

1.  $\Box\Box\varphi \Leftrightarrow \Box\varphi$
2.  $\Diamond\Diamond\varphi \Leftrightarrow \Diamond\varphi$
3.  $\Diamond\varphi \wedge \Diamond\psi \Leftrightarrow \Diamond(\varphi \wedge \psi)$
4.  $\Diamond\varphi \vee \Diamond\psi \Leftrightarrow \Diamond(\varphi \vee \psi)$
5.  $\Diamond\Box\Diamond\varphi \Leftrightarrow \Diamond\Box\varphi$
6.  $\Diamond\Box\Diamond\varphi \Leftrightarrow \Box\Diamond\varphi$

#### Exercice 4 :

#### *Formules fermées*

Une formule  $\varphi$  de LTL est dite fermée par préfixe si pour toute suite  $x$ , et pour toute suite finie  $z$ , si  $x \models \varphi$  alors  $zx \models \varphi$ . Une formule  $\varphi$  de LTL est dite fermée par coupure si pour toute suite  $x$ , et pour tout entier  $i$ , si  $x \models \varphi$  alors  $x, i \models \varphi$ . Soient  $p, q$  formules d'état. Dire, pour chacune des formules suivantes, se elle est fermée par préfixe ou/et par coupure.

1.  $\Box p$
2.  $\Diamond p$
3.  $\Box \Diamond p$

4.  $\Diamond \Box p$
5.  $\Diamond \Box p \Rightarrow \Box \Diamond q$