# Contrôle continu – Modélisation et spécification

## Master Informatique

10 novembre 2015

**Durée :** 2h.
**Documents autorisés :** Une feuille A4 manuscrite recto-verso.

Chaque exercice est sur **5 points**.

*Modélisation d'algorithmes*

**Exercice 1 :**

On considère l'algorithme suivant pour deux processus P1 et P2 avec variables partagées. Notez que la ligne v1=max++ signifie que de façon atomique on met dans v la valeur courante de max et ensuite on incrémente la variable max.

```
int max=0; // variable partagée
```

```
— Processus P1
int v1=0;
while(true){
    v1=max++;
    while(v1!=max-1){}
    /*section critique*/
    max=max-1;
    v1=0;
}
```

```
— Processus P2
int v2=0;
while(true){
    v2=max++;
    while(v2!=max-1){}
    /*section critique*/
    max=max-1;
    v2=0;
}
```

1. Donnez une caractérisation du processus P1 par un graphe de programmes $PG1$ et donnez une caractérisation du processus P2 par un graphe de programmes $PG2$. Quelles locations correspondent aux sections critiques ?
2. On considère maintenant le système de transitions étiqueté $TS(PG1|||PG2)$ associé au graphe de programmes $(PG1|||PG2)$. Quelle est la forme des états de ce système. Dessinez une partie de ce système en représentant au moins 10 états.
3. Donnez la formule LTL qui permet d'exprimer le fait qu'à aucun moment les deux processus sont en section critique en même temps.
4. Est-ce-que $TS(PG1|||PG2)$ satisfait la formule que vous avez définie précédemment ? Justifiez votre réponse.
5. On considère maintenant un troisième processus P3 qui a le comportement suivant :

```
— Processus P3
int v3=0;
while(true){
    v3=max++;
    while(v3!=max-1){}
    /*section critique*/
    max=max-1;
    v3=0;
}
```

Donnez une caractérisation du processus P3 par un graphe de programmes $PG3$. Dans le système de transitions étiqueté $TS(PG1|||PG2|||PG3)$, la variable max prend elle un nombre fini de valeurs ? Justifiez votre réponse.

*Accès contrôlé*

**Exercice 2 :**

On considère un système fait de deux processus P1 et P2 et d'un contrôleur.
Le contrôleur est au début dans un état initial depuis lequel il peut recevoir une demande d'accès à la section critique soit du processus P1 soit du processus P2. Si il accepte une demande d'un processus, il attend que celui-ci lui-envoie un message de sortie de section critique avant de retourner dans l'état initial. Attention, le contrôleur doit donc se rappeler de quel processus il a accepté une demande.
Les deux processus ont ensuite le même comportement. En boucle, il demandent l'accès à la section critique et rentrent en section critique, puis ils quittent la section critique en envoyant un message de fin au contrôleur. On notera Init1 la proposition atomique pour signaler que le processus P1 est dans l'état initial, SC1 pour signaler que P1 est

dans sa section critique et pour P2 on utilisera les mêmes types de propositions atomiques que l'on notera Init2 et SC2.

On souhaite modéliser ce système par un système de transitions étiqueté. Pour cela on considèrera l'alphabet suivant pour les actions :

- `A-i` (pour i valant 1 et 2) : le processus Pi demande à accéder à la section critique
- `Q-i` (pour i valant 1 et 2) : le processus Pi quitte la section critique
- `DA-i` (pour i valant 1 et 2) : le contrôleur reçoit une demande d'accès à la section critique du processus Pi
- `DQ-i` (pour i valant 1 et 2) : le contrôleur reçoit un message de sortie de section critique de Pi

On supposera de plus que dans l'état initial les deux verrous sont libres.

1. Donnez une modélisation sous forme de système de transitions étiqueté du contrôleur.
2. Donnez une modélisation sous forme de système de transitions étiqueté des processus P1 et P2.
3. Donnez une définition du système de transitions étiqueté S représentant le système dans son ensemble.
4. Exprimez en LTL la propriété suivante : toujours, tout processus dans son état initial finit par être en section critique.
5. Le système S vérifie-t-il cette dernière propriété ? Justifiez votre réponse.

---

**Rappel : Syntaxe et sémantique de LTL.** Les formules de LTL se construisent selon la grammaire : $\phi ::= P \mid \phi \vee \psi \mid \phi \wedge \psi \mid \neg\phi \mid \mathbf{X}\phi \mid \phi\,\mathbf{U}\,\phi\mathbf{X}^{-1}\phi \mid \phi\,\mathbf{S}\,\psi$ où $P$ appartient à un ensemble de propositions atomiques $AP$. On utilise aussi les abréviations pour le futur $\mathbf{F}\phi \stackrel{def}{=} \top\,\mathbf{U}\,\phi$, $\mathbf{G}\phi \stackrel{def}{=} \neg\mathbf{F}\neg\phi$, et celles pour le passé : $\mathbf{F}^{-1}\phi \stackrel{def}{=} \top\,\mathbf{S}\,\phi$, $\mathbf{G}^{-1}\phi \stackrel{def}{=} \neg\mathbf{F}^{-1}\neg\phi$

Elles s'interprètent sur une position $i \geq 0$ le long d'une séquence infinie d'états étiquetés par des propositions atomiques $\rho$ : $\rho(i)$ est le $i$-ème état et $\ell(\rho(i))$ est l'ensemble des propositions vraies en $\rho(i)$. Les positions $0, \ldots, i-1$ sont le passé, et celles au delà de $i$ sont le futur. On :

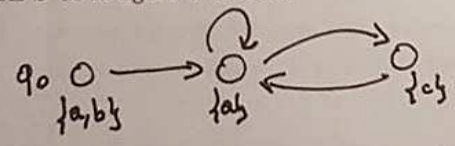
- $\rho, i \models P$ ssi $P \in \ell(\rho(i))$
- $\rho, i \models \mathbf{X}\phi$ ssi $\rho, i+1 \models \phi$
- $?, i \models \phi\,\mathbf{U}\,\psi$ ssi $\exists j \geq$ it.q.$(\rho, j \models \psi) \wedge (\forall k \in [i, j[,$ on a $: \rho, k \models \phi\ )$
- $\rho, i \models \mathbf{X}^{-1}\phi$ ssi $i > 0$ et $\rho, i-1 \models \phi$
- $\rho, i \models \phi\,\mathbf{S}\,\psi$ ssi $\exists j \geq$ it.q.$(\rho, j \models \psi) \wedge (\forall k ?[i, j[$on a $: \rho, k \models \phi\ )$

Plan :

*Logique temporelle LTL*

**Exercice** 3 :

On considère le STE $S$ de la figure ci-dessous.




1. Décrire l'ensemble des exécutions de $S$ depuis l'état $q_0$. Soit $\mathcal{E}$ cet ensemble.
2. Pour chacune des formules suivantes, dire si la formule est vraie pour toutes les exécutions de $\mathcal{E}$ (on justifiera les réponses) :

$$a\,\mathbf{U}\,c \qquad b\,\mathbf{U}\,a \qquad a\,\mathbf{U}\,b \qquad b\,\mathbf{U}\,c \qquad (a\,\mathbf{U}\,c \wedge Ga) \qquad \mathbf{G}(a\,\mathbf{U}\,c \wedge \mathbf{G}a)$$

**X** *Logique temporelle LTL*

**Exercice** 4 :

Pour chacune des formules suivantes, donner DEUX modèles différents (significativement !) pour lesquels la formule est vraie et DEUX modèles où elle est fausse :

$$\mathbf{G}(\texttt{problème} \Leftrightarrow \texttt{alarme}) \qquad \mathbf{G}\Big(a \Leftrightarrow (b\,\mathbf{U}\,c)\Big) \qquad \mathbf{G}\Big(a \Leftrightarrow \mathbf{X}b\Big)$$

**XX** *Logique temporelle LTL*

**Exercice** 5 :

Ecrire en logique temporelle, les propriétés suivantes qui concernent un protocole de communication où des messages sont émis (prop : `E`), reçu (prop : `R`), et des erreurs sont signalées (prop : `Err`)

1. Toute réception est précédée d'un envoi.
2. Si un message est envoyé, il est reçu ou une erreur est signalée.
3. Si un envoi n'est pas reçu après un premier envoi, le second envoi est toujours reçu.
4. Il ne peut y avoir deux erreurs le long d'une exécution.