UFR 919 Informatique – Master Informatique

Spécialité STL – UE 5I553 – PPC

Paradigmes de programmation concurrente
TME 4 — (2 h)
Uppaal

Carlos Agon

1 Question 1 : Tutoriel Uppaal

UPPAAL est l'un des environnements les plus populaires pour la modélisation, vérification et spécification de systèmes réactifs. Il a été développé et maintenu par un équipe mixte des universités de UPPsala et AALborg.

Dans un premier temps, en suivant le tutoriel de la question 1 vous allez découvrir cet environnement, pour après vous épanouir en modélisant les questions numéro 2 et 3.

1.1 Une machine à cafe

Pour découvrir Uppaal, reprenons l'exercice 1 du TD2. Il s'agissait de modéliser un machine à cafe avec la spécification suivante :

— la machine accepte les pièces quand elle est disponible,
— on peut appuyer sur un des deux boutons de la machine : thé ou café,
— quand on appuie sur un bouton, si une piece est présente, et si aucun gobelet n'est présent, la machine sert la boisson requise,
— une personne peut boire la boisson et jeter le gobelet, cela rend la machine disponible.

Bien sûr, vous savez déjà modéliser une telle machine, mais comment faire en Uppaal ?

1. Dans un premier temps, vous devez explorer l'interface de **`Editor`**, ce qui vous permettra de définir le LTS de la Figure 1. Renommez ce template par 'machine'.
2. Apres avoir construit l'automate, cliquez sur le fichier *System declarations* et écrivez le code suivant : `system machine;`. Cela indique que notre système est défini par un seul automate : `machine`.
3. Maintenant cliquez sur le bouton **`Simulator`**, si vous n'avez pas d'erreur de syntaxe vous allez voir une fenêtre comme celle de la figure 2.

   Explorez les différentes formes de navigations et ses représentations graphiques.

Nous allons maintenant définir un processus 'user', et le faire interagir avec notre machine.

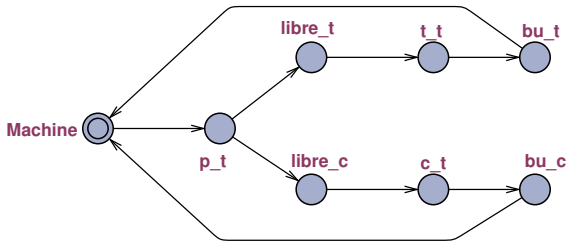

FIGURE 1 –

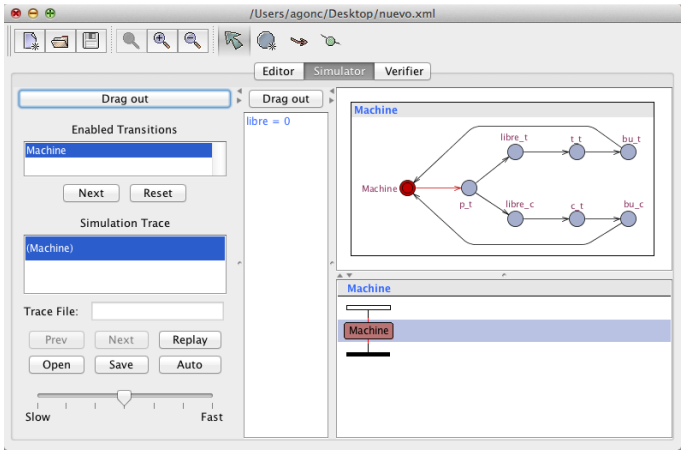

FIGURE 2 –

1. Créer un template 'user' correspondant au processus p!.(pt!.tea?.bu! + pc!.coffee?.bu!)
2. Pour indiquer l'action d'une transition, cliquez sur la transition et écrivez le nom de l'action sur la champ 'sync'.
   Remplissez les transitions pour 'user' et pour 'machine'.
   Vous devez aussi déclarer l'ensemble de toutes les actions dans le fichier `Declarations`.
   Entrez le code suivant : `chan p, pt ,pc ,bu, tea, coffee;`
3. Pour faire une nouvelle simulation, écrivez dans *System declarations* le code suivant :

```
Machine = machine ();
User1 = user ();
User2 = user ();
system Machine, User1, User2;
```

Finalement vous pouvez faire quelques vérifications dans l'éditeur **`Verifier`**

1. Il n'y a pas deadlock
   (A[] not deadlock)
2. Les deux utilisateurs peuvent boire du café au même temps
   (E<> (User1.c_t && User2.c_t)
3. Si user1 est prêt pour boire du thé alors la machine est dans l'état bu_t (A[] (User1.bu_t imply Machine.bu_t))

Pour tester d'autres formules vous pouvez vous en servir des symboles suivants : && (and), `||` (or), == (equalite), `imply` (implication), `not` (négation)

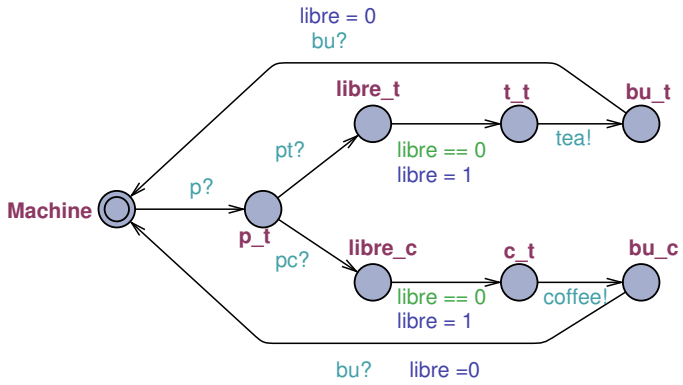

FIGURE 3 –

Nous allons maintenant gérer l'état libre de la machine en utilisant une variable. Pour cela, on définit la variable `libre` dans le fichier `Declarations`, ainsi : `int [0,1] libre`

En double cliquant sur une transaction vous verrez d'autres champs à remplir, en particuliers les champs `Guard` et `Update`. Une condition sur le champ Guard permet de tester par une variable si la transition peut être effectuée. Le champ Update permet de modifier les variables une fois la transition franchie. Ainsi finalement le LTS de notre machine est montré dans la figure 3.

2 Question 2 : Un dîner chic

Un couple souhaite dîner dans un restaurant chic où l'on sert :
-champagne (CH)
-foie gras (FG)
-soupe de tortue (ST)

Mais malheureusement à table il n'y a qu'une fourchette et une cuillère. Cela ne pose pas de problème pour le champagne, on peut le boire sans contrainte (mais avec mesure). La soupe bien sûr demande l'utilisation de la cuillère, tandis que pour le foie gras on peut utiliser soit la fourchette soit la cuillère.

Le serveur apporte, dans n'importe quel ordre, les différents mets et boissons, mais comme le restaurant est un peu cher le couple prendra au maximum 10 mets ou boissons et après ils arrêtent de manger.

Supponsons que le serveur apporte dans l'ordre :
$ST \rightarrow$ FG $\rightarrow$ ST $\rightarrow$ ST $\rightarrow$ FG $\rightarrow$ CH $\rightarrow$ CH $\rightarrow$ FG $\rightarrow$ FG $\rightarrow$ FG.

1. Faîtes un système qui modélise le dîner.
2. Prouver que il n'y a pas d'interblocage.

3 Question 3 : Algorithme d'exclusion mutuelle

Modélisez le fameux algorithme de Petterson d'exclusion mutuelle

Process 1	Process 2
req1 = 1 ;	req2= 1 ;
turn = 2 ;	turn = 1 ;
while (turn != 1 && req2 != 0) ;	while (turn != 2 && req1 != 0) ;
-section critique-	-section critique-
job1() ;	job2() ;
req1=0 ;	req2 = 0 ;

Montrez que les deux processus ne peuvent pas rentrer dans la section critique en même temps.