Méthodes formelles de vérification (MFVerif)

Composition parallèle

Exercice 1 – *Caisses*

On considère un système de prix dans une caisse de supermarché. Le système permet de scanner les codes "barre" de produits et d'imprimer leurs prix. Le système est composé de trois composantes fonctionnant en parallèle :

1. Le processus de lecture de code est un processus itératif ayant deux actions : `Scanner_code` qui permet de lire un code barre, puis `Emission_code` qui émet le code lu.
2. Le processus de conversion de prix est un processus itératif ayant deux actions : `Recevoir_code` lui permettant de recevoir un code, puis `Emission_prix` qui émet le prix correspondant au code reçu.
3. Le processus d'impression de prix est un processus itératif ayant deux actions : `Recevoir_prix` lui permettant de recevoir un prix, puis `Imprimer_prix` permettant d'imprimer le prix reçu.

On fait abstraction des valeurs des codes et des prix (c'est-à-dire que l'on ne distinguera pas entre les actions selon les valeurs de leurs paramètres).

1. Donner les systèmes de transitions modélisant chacun des processus décrits ci-dessus.

2. Donner le modèle pour le système de saisie des prix basé sur une composition parallèle des trois processus ci-dessus. (Préciser les actions de synchronisation.)

3. Donner le système de transitions correspondant à ce modèle.
On considère que seules les actions `Scanner_code` et `Imprimer_prix` sont visibles (les autres sont considérées comme internes au système).

4. Donner le système de transitions modélisant le comportement visible du système de saisie. Combien d'objets peuvent être scannés avant qu'un prix soit imprimé ?

5. Est-il possible de montrer sur le modèle considéré la propriété suivante : "Les prix des objets sont affichés dans le même ordre que celui dans lequel ils ont été scannés".

Exercice 2 – *Exclusion mutuelle*

On veut formaliser le problème de l'exclusion mutuelle. Le système est composé de trois processus. Deux processus qui utilisent une ressource `r`, et un processus qui gère la ressource lors qu'il est libre.

Les processus qui utilisent la ressource ont les transitions suivantes :
- `Demander_res` qui demande la ressource en question, et
- `Rendre_res` qui rend la ressource après son utilisation.

Le processus représentant la gestion de la ressource a quatre transitions :
- `Accepter_proc1` et `Accepter_proc2` qui donnent accès à la ressource à chaque processus selon le cas, et
- `Recevoir_proc1` et `Recevoir_proc2` qui reprend la ressource de chaque processus (selon le cas) et la rend disponible.

1. Donnez des systèmes de transitions modélisant chacun des processus.

2. Donnez le modèle pour le système basé sur la composition parallèle des trois processus ci-dessus. (Préciser les actions de synchronisation.)

3. Donnez le système de transition correspondant à ce modèle.

4. Montrez que le système assure l'exclusion mutuelle des processus sur le ressource `r`.