Sorbonne Université – Master Informatique – Spécialité STL – UE 5I553 – PPC

Paradigmes de programmation concurrente
TD 1 — (2 h)
Modélisation avec de réseaux de Petri

Romain Demangeon

Excercice 1 : Machine à café

Nous souhaitons modéliser le distributeur de boissons suivant :
— la machine accepte à tout moment des pièces,
— on peut appuyer sur deux types de boutons *thé* ou *café*,
— quand on appuie sur un bouton, si une pièce est presént, et si il n'y a pas de gobelet dans le reservoir, la machine sert la boisson requise,
— pour boire la boisson le client retire le gobelet de la machine.

Question 1

Proposer un réseaux de Petri qui modélise le distributeur de boissons. Donner sa description formelle et sa matrice d'incidence.

Question 2

Donner un vecteur initial correspondant à une machine vide et une personne avec 3 pièces.
Calculer des séquences totales possibles. Donner leur vecteurs caractéristiques.

Question 3

Expliquer formellement pourquoi, à tout moment, on ne peut avoir dans la poubelle plus de gobelets que de pièces que la personne possède initialement.

Question 4

Reprendre les questions 1 et 2 en supposant qu'un thé coûte une pièce et un café deux pièces.

Excercice 2 : Modèles de Programmes

Pour chaque question, proposer un réseau de Petri modélisant le système de threads. Identifier les transitions vivantes et les blocages.

Question 1

Un unique thread exécutant le programme suivant :

1. Tâche locale T_1
2. Création de deux threads fils exécutant chacun les tâches locales L_1 puis L_2
3. Attente de synchronisation des deux threads et destruction.
4. Retour à l'étape 1.

Question 2

Un thread t_1 exécutant le programme suivant :

1. Tâche locale T_1
2. Envoi d'un message asynchrone m à t_2
3. Tâche locale T_2.
4. Réception d'une réponse asynchrone r de t_2.
5. Tâche locale T_3.
6. Envoi d'un message m à t_2

et un thread t_2 adéquat (qui permet à t_1 de ne pas être bloqué).

Question 3

Trois threads exécutant chacun le programme suivant :

1. Tâche locale indépendante T_1
2. Utilisation d'une ressource partagée R
3. Tâche locale indépendante T_2
4. Retour à l'étape 1.

Question 4

Deux threads exécutant chacun le programme suivant :

1. Tâche locale indépendante T_1
2. Prise d'une ressource partagée R_1
3. Prise d'une ressource partagée R_2
4. Tâche locale T_2
5. Relâche de la ressource R_2
6. Relâche de la ressource R_1
7. Retour à l'étape 1.

Question 5

Trois threads réalisant le diner des philosophes.

Question 6

Quatre threads accédant régulièrement à une base de données n'acceptant que deux connexions simultanées.

Question 7

Un thread implémentant le programme p suivant :

1. Tâche locale indépendante T_1
2. Création d'un nouveau thread exécutant p
3. Tâche locale indépendante T_2
4. Choix non-déterministe entre i) terminaison ou ii) retour à l'étape 1

Excercice 3 : Réseaux Bornés

Une place dans un réseau de Petri est k-bornée quand, dans tous les marquages atteignables depuis le marquage initiale, elle contient au plus k jetons. Elle est *safe* quand elle est 1-bornée. Un réseau est k-borné (resp. *safe*) quand toutes ses places sont k-bornées (resp. *safe*)
Donner des exemples de réseaux simples 1) *safe* 2) k-bornés mais non-*safe* 3) non k-bornés.
Donner les places et les réseaux k-bornés et *safe* des exemples précédents.

Excercice 4 : Modelisation

A l'aide d'un logiciel de votre choix (e.g. *pneditor*) modélisez et testez des réseaux de Petri pour :

— The barbershop problem
— The Santa Claus problem