Systèmes Concurrents

1h45, documents autorisés

19 décembre 2012

Les exercices sont indépendants. Barème sous réserve.

1 Cours : conception d'une application concurrente (2 points)

Soit une application concurrente avec beaucoup de variables partagées, accédées principalement en lecture et rarement en écriture. On souhaite assurer la cohérence de séquences de code (i.e. tel que le résultat soit similaire à une exécution purement séquentielle). Les possibilités d'implantation sont :

1. exécuter chaque séquence de code en exclusion mutuelle vis-à-vis des autres séquences (utilisation d'un mutex) ;
2. protéger l'accès à chaque variable par un verrou spécifique, chaque séquence de code acquérant les verrous nécessaires ;
3. implanter l'application en non-bloquant ;
4. utiliser une mémoire transactionnelle.

Pour chaque choix, argumenter en deux ou trois phrases l'intérêt et l'inconvénient de chacun.

2 Java Threads (4 points)

Écrire un programme utilisant l'API Java Thread (ou si préféré l'API C Posix Thread) et qui a le fonctionnement suivant :

- trois activités sont créées par l'activité initiale ;
- l'activité 1 incrémente de 1 à 10 un compteur puis se termine ;
- l'activité 2 incrémente ce même compteur jusqu'à 20 à partir du moment où l'activité 1 a fini son travail ;
- l'activité 3 affiche un message chaque fois que le compteur prend une valeur impaire ;
- le programme se termine après que l'activité 2 est terminée ;
- une synchronisation adéquate assurera le bon ordonnancement.

3 Sémaphore (4 points)

Dans une région africaine reculée, une forêt est divisée par un canyon infranchissable. Il y a cependant une liane en travers. Cette liane permet aux chimpanzés de traverser le canyon. Cependant, si deux chimpanzés allant en direction opposée se trouvent sur la liane, ils se battent et tombent dans le canyon. En outre, la liane ne peut supporter que le poids d'au plus cinq chimpanzés.

Les opérations de synchronisation sont `s_engager_sur_la_liane` et `quitter_la_liane` (l'opération `parcourir_la_liane` est purement applicative). Un chimpanzé exécute le « code » suivant :

```
s_engager_sur_la_liane(sens); // sens = GD ou DG (gauche-droite ou droite-gauche)
parcourir_la_liane;
quitter_la_liane();
```

Questions

1. Donner le code des deux opérations de manière à assurer :
   - quand un chimpanzé a commencé à traverser, il est certain de pouvoir finir la traversée sans croiser un autre chimpanzé ;
   - il n'y a jamais plus de cinq chimpanzés simultanément sur la liane.
2. Votre solution présente-t-elle un risque de famine, i.e. un flux infini de chimpanzés venant dans un sens empêche-t-il les chimpanzés de passer en sens inverse ? Si oui, proposez une modification à votre solution pour corriger ce défaut.

4 Moniteur (6 points)

On considère un pub où de la bière est servie et consommée selon les règles suivantes :

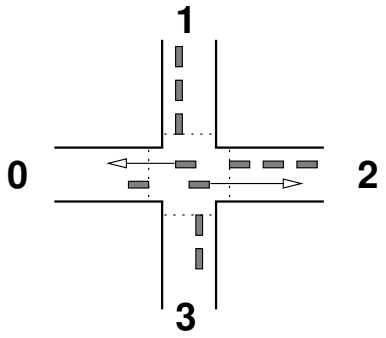
1. il n'y a qu'une seule boisson, servie au verre ;
2. des consommateurs peuvent entrer à n'importe quel moment ;
3. un consommateur n'a jamais plus d'un verre à la main ;
4. les consommateurs demandent à être servis, un à la fois, par le patron ;
5. quand un consommateur a fini son verre, il le pose sur une table quelconque et peut aller demander un nouveau verre (plein !), ou quitter l'établissement ;
6. un assistant du patron circule de temps en temps dans la salle pour ramasser les verres sales et les rapporter au comptoir (il en prend un nombre arbitraire) ;
7. le patron utilise des verres propres pour servir les consommateurs (ouf) ;
8. quand il n'y a pas de consommateurs en attente ou qu'il n'y a plus de verres propres, le patron lave des verres sales en attente de nettoyage ;
9. le nombre de verres est fini (N) ;
10. il n'y a pas de contraintes sur le nombre de consommateurs ou la taille des tables.

Questions

1. Donner le code des processus « consommateur », « patron » et « assistant ».
2. En déduire les opérations de synchronisation (l'interface) et développer un moniteur assurant le bon fonctionnement de ce pub. Bien identifier les variables d'état nécessaires.

5 Processus communicants (4 points)

On considère un carrefour à quatre voies. Le trafic peut provenir des quatre directions et ne peut être que traversant (un véhicule ne peut que traverser le carrefour mais ne peut pas tourner à droite ou à gauche). Par ailleurs au plus trois véhicules non croisant peuvent se trouver simultanément dans le carrefour. Ces trois véhicules peuvent traverser dans le même sens ou en sens opposé. Par contre, ils ne peuvent pas venir de deux voies perpendiculaires.



Note : vous pouvez répondre in(différemment en CSP (`canal?message...`) ou en Ada (`accept rendezvous(...)`). Il est cependant conseillé d'utiliser CSP, plus léger et suffisant pour l'exercice.

Questions

1. Donner l'interface de l'activité de synchronisation (quels canaux/quels rendez-vous).
2. Donner le code de l'activité de synchronisation, sans se préoccuper de la famine.
3. Proposer une solution simple permettant d'éviter la famine.