Contrôle INF 223

23 Novembre 2011

CORRIGE

1h30 – Sans document

1 Concurrence

1.1 Threads Posix (2 points)

Expliquer précisément le fonctionnement du couple `pthread_cond_wait` et `pthread_cond_broadcast`.

1.2 Threads Java (2 points)

Si S1 et S2 sont des sémaphores initialisés ainsi :

```
Semaphore S1  = new Semaphore (n);
Semaphore S2  = new Semaphore (0);
```

La gestion suivante d'un schéma producteur/consommateur où put et get sont des méthodes associées à un même objet, est-elle correcte ? Pourquoi ? Corriger en justifiant.

```
synchronized put(Object o){
  S1.acquire();
  last=(last+1)%size;
  buffer[last] = o;
  S2.release();
  }
```

```
synchronized Object get(){
   S2.acquire();
   Object o = buffer[first];
   first=(first+1)%size;
   S1.release();
   return o;
}
```

Correction

```
Object get(){
   fullSlots.acquire();
   synchronized (this) {
      Objet o = b[first];
      first=(first+1)%size;
   }
   emptySlots.release();
   return o;
}
```

```
void put (Object o) {
   emptySlots.acquire();
   synchronized (this) {
      last=(last+1)%size;
      b[last] = o;
   }
fullSlots.release();
}
```


2 Communications

2.1 Question 1 Sockets (2 points)

Expliquer précisément ce qui se passe lors de l'exécution des instructions suivantes : quel est le rôle de `Port`, comment sera débloqué le serveur qui attend dans `accept`, et que fera-t-il après avoir été libéré ?

```
ServerSocket sock1 = new ServerSocket(Port);

Socket sock2 = sock1.accept();
```

2.2 Question 2 Sockets (2 points)

Quel problème peut se poser lors d'un échange de données, par exemple du type `int`, entre deux processus distants qui communiquent en utilisant des sockets (que ce soit avec UDP ou TCP) ? Donner une solution.

3 Cas d'étude

3.1 Version threads (2 points)

Dans le cas d'étude en version Thread, expliquer quelle construction est utilisée pour établir les communications entre threads et indiquer quelles propriétés cette construction apporte en termes de synchronisation.
Correction : boite aux lettres ou tampon circulaire. Blocage lorsque vide pour get et lorsque plein pour put.

3.2 Version sockets (2 points)

Dans le cas d'étude en version Socket, expliquer comment est constitué le graphe de communication complètement maillé afin de minimiser le nombre de sockets ouvertes.
Correction : on brise la symétrie en fonction du numéro de noeud pour n'ouvrir qu'une seule socket.

3.3 Version CORBA (2 points)

Dans le cas d'étude en version CORBA, expliquer comment les nœuds arrivent à prendre contact alors qu'ils n'ont aucune information au départ les uns entre les autres. Indiquer une façon alternative pour obtenir le même résultat, en se basant sur une solution présentée dans les notes du cours.
Correction : on stocke les références dans des fichiers. On aurait pu utiliser un serveur de noms.