Examen de parallélisme

Maîtrise, UVSQ

22 janvier 2003

Notes. L'examen dure 2 heures (le temps, subjectif, de chaque sujet est donné aussi). Les seuls documents autorisés sont le cours et les TDs/TPs. Lisez bien les sujets. N'écrivez pas l'énoncé du sujet sur votre papier, juste son numéro. N'écrivez pas beaucoup, mais bien (*non multa, sed multum*).

1 Questions du cours (30')

a) Donnez les 2 principaux buts/avantages du parallélisme (≈4 lignes pour chaque but).

b) Même chose pour les inconvénients.

c) Expliquez ce que c'est le faux partage (« false sharing ») et comment on peut l'éviter. Faire un schéma.

d) Spécifiez comment un programme peut avoir une accélération supérieure à 2 pour 2 processeurs.

2 Interblocage à la piscine (20')

À une piscine il y a 25 paniers et 5 cabines (avec leurs clés). Toute personne qui va à la piscine fait, en ordre :

- prendre la clé d'une cabine ;
- prendre un panier ;
- ... (aller à la cabine pour se mettre en maillot de bain)...
- rendre la clé de la cabine ;
- ... (se baigner, réfléchir à son examen de parallélisme)...
- prendre la clé d'une cabine ;
- ... (aller à la cabine et se rhabiller) ;
- rendre la clé de la cabine ;
- rendre le panier.

a) Écrivez le morceau de code multi-threading (en pseudocode) qui gère tout cet enchaînement d'actions. (En pseudocode, on peut utiliser directement les primitives de synchronisation : mutex (lock(l), unlock(l)), sémaphore (wait(s), notify(s)), barrière (barrier()) etc.)

b) Dans l'algorithme présenté il peut y avoir un interblocage. Donnez une situation faisant apparaître l'interblocage. Parmi les 4 conditions d'interblocage de Coffman et al., trouvez celles qui sont remplies. Enfin, spécifiez ce qu'il faut faire pour éviter tout interblocage.

3 Un peu de littérature (20')

Écrivez en pseudocode l'algorithme du lecteur/écrivain *avec priorité à l'écrivain.* (Dès qu'il y a un écrivain en attente, tous les lecteurs *suivants* sont mis en attente, ou bien un lecteur n'entre pas dans la section critique s'il y a 1 ou plusieurs écrivains en attente.)

4 Algorithmique parallèle (50')

Nous avons un tableau bidimensionnel d'entiers $N \times N$, avec N très grand (e.g. 10000), tableau déjà initialisé. Il faut faire une modification dans ce tableau : dans chaque élément il faut mettre la moyenne entre lui-même et ses 4 éléments voisins : haut, bas, droite, gauche (comme dans le cas d'une image, pour la rendre floue). Pour les bords, on ne prend en considération que ses 2–3 voisins.

Malheureusement, comme cette matrice occupe déjà quasiment toute la mémoire de notre multi-processeur, il est très inefficace pour faire ce calcul d'allouer une nouvelle matrice intermédiaire (ou de faire de la récursion sur toute la matrice). Cependant, chaque thread a à sa disposition un vecteur de N entiers, ainsi qu'une variable locale spécifiant son identificateur :

```
int tab[N][N];
int v[NBTHREADS][N];
int thid;  // déjà initialisé avec une valeur entre 0 et NBTHREADS-1
```

Écrivez où est la difficulté du problème et *clairement* comment la résoudre dans les cas suivants :

a) 1 thread (i.e. en séquentiel) ; dans ce cas, écrire aussi le code (en pseudocode) exécuté par chacun des threads ;

b) 4 threads avec la décomposition de domaine suivante :

```
+---------+
|---------|
|---------|
|---------|
+---------+
```

c) 4 threads avec la décomposition de domaine suivante :

```
+----+----+
|    |    |
|----+----|
|    |    |
+----+----+
```