Université de Limoges | FACULTÉ DES SCIENCES ET TECHNIQUES

Licence 3ème année

UE Parallélisme I

janvier 2012

Durée : 2h — Documents autorisés

Threads & Sémaphores – (10 points)

1 – 5pts Soient les 4 threads suivantes, s'exécutant chacune une seule fois :

```
Thread A {          Thread B {          Thread C {          Thread D {
   P(s2);              P(s2);              P(s1);              P(s1);
   x = x + y;          y = x * y;          y = y * 3;          P(s3);
   V(s2);  }           V(s2);  }           V(s3);  }           x = x + 2;
                                                               V(s2);  }
```

Elles utilisent pour leur synchronisation trois sémaphores `s1`, `s2` et `s3`.
La valeur initiale de `s1` est 2, celle de `s2` est `0` et celle de `s3` est `0`.

a. Quelles sont les différentes valeurs possibles des variables `x` et `y` à la fin de l'exécution complète des quatres *threads*, si `x` est initialisé à `0` et `y` à `1` ?

b. Qu'est-ce que cela change si la valeur initiale de `s1` est 1, celle de `s2` est `0` et celle de `s3` est `1` (`x` reste à `0` et `y` à `1`) ?

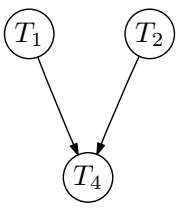
2 – 5pts Écrire un algorithme pour résoudre le problème suivant (vous pouvez vous inspirer de la notation de l'exercice 1) :

Un programme a été décomposé en 9 threads $\{T_1, T_2, T_3, T_4, T_5, T_6, T_7, T_8, T_9\}$.
L'analyse du parallélisme pouvant être exploité entre ces différentes threads est la suivante :
- les threads $\{T_2, T_3\}$, $\{T_4, T_5, T_6\}$ et $\{T_7, T_8\}$ peuvent être réalisées en parallèle ;
- la thread T_4 ne peut être faite qu'après la thread T_2 ;
- les threads T_5 et T_6 ne peuvent être exécutées qu'après la thread T_3 ;
- la thread T_7 ne peut être faite qu'après les threads $\{T_4, T_5\}$;
- les threads T_2 et T_3 ne peuvent être exécutées qu'après la thread T_1 ;
- les threads T_7 et T_8 peuvent s'exécuter en parallèle ;
- la thread T_9 ne peut s'exécuter qu'après les threads T_7 et T_8 ;
- la thread T_8 ne peut s'exécuter qu'après la thread T_6.

Questions :

a. Donnez le graphe de précédence du travail de ces 9 threads ;

Vous utiliserez la notation suivante :



$\Longrightarrow$ *Ce graphe de précédence correspond à la description suivante :*
- $\{T_1, T_2\}$ *peuvent être réalisées en parallèle ;*
- T_4 *ne peut être faite qu'après les threads* T_1 *et* T_2.

b. Combien de Sémaphores faudra-t-il pour réaliser ce travail ?

c. Donnez l'algorithme qui permet de synchroniser les threads entre elles.

Création de processus avec fork – (4 points)

3 – Soit le programme C suivant :

4pts

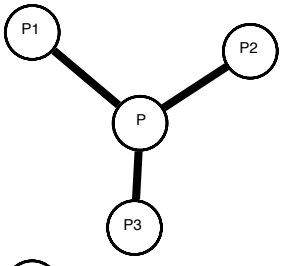
```
#include <stdio.h>

void enfant(int);

int main()
{  int resultat = 0;
   resultat = fork();

   if (resultat)
   {
      enfant(1);
   }
}

void enfant(int parametre)
{  int i=0; int resultat = 0;

   for(i=0; i <3; i++)
   {
      resultat = fork();
      if (resultat)
      {
         parametre = parametre * 3;
         printf("Je suis le processus 1 et parametre=%d\n", parametre);
      }
      else
      {
         parametre = parametre + 2;
         printf("Je suis le processus 2 et parametre=%d\n", parametre);
      }
   }
}
```

Que va-t-il afficher lors de son exécution ?

Signaux & tubes – (6 points)

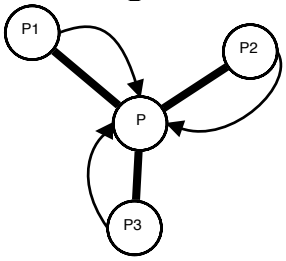
4 – Programmation :

6pts a. Écrire le programme `C` réalisant le travail suivant :



1. un processus P crée 3 fils $\{P_1, P_2, P_3\}$;
2. il crée un tube avec chacun de ses fils permettant de communiquer du père vers le fils ;
3. chaque fils va :
   - ⋆ lire une valeur depuis le père et l'afficher en indiquant son numéro :

```
Je suis le fils 1 et j'ai reçu la valeur 1
```



   - ⋆ envoyer un **signal au père** ;
   - ⋆ recommencer ;
4. lorsque le père reçoit 3 signaux consécutifs (un de chacun de ses fils), il envoie une nouvelle valeur à chacun de ses fils.

Vous utiliserez la fonction `pid_t getppid(void);` *pour obtenir le pid du père. Le père enverra des valeurs à partir d'un compteur :* $\{0, 1, 2, 3, ..\}$

b. Indiquez comment le père peut terminer l'ensemble des processus fils.