Programmation Système
TP no 2 : Threads et Synchonisations

Guillaume Mercier
email : mercier@enseirb.fr

7/8 Novembre 2019

1 La théorie ... et la pratique

Récupérez le code du programme `test_mutex_types.c` disponible ici : `/net/ens/mercier/IF210/TP2/Codes/test_mutex_types.c`. Compilez-le et exécutez-le. Que constatez-vous ? Allez faire un tour sur la page de manuel des fonctions `pthread_mutex_lock/unlock` et `pthread_mutexattr_init` pour confirmer ou infirmer vos hypothèses.

2 Passage d'arguments

Soit la fonction suivante :

```
int fonction(int a , char b, int *ptr){
return a + (int)b + *ptr;
}
```

Créez un programme avec des threads qui éxécutent cette fonction. Les arguments de la fonction devront être passés au thread qui l'exécutent et pas autrement (i.e. pas en variable globale, etc.).

3 Sémaphores et verrous

On suppose disposer des outils suivants :

1. des verrous, de type `pthread_mutex_t` et qui possèdent trois opérations :
   — `pthread_mutex_init(pthread_mutex_t *mon_mutex)` qui permet d'initialiser un objet de type verrou ;
   — `pthread_mutex_lock(pthread_mutex_t *mon_mutex)` qui permet de prendre un verrou ;
   — `pthread_mutex_unlock(pthread_mutex_t *mon_mutex)` qui permet de libérer un verrou ;
2. des variables de condition, de type `pthread_cond_t` et qui possèdent quatre opérations :
   — `pthread_cond_init(pthread_cond_t *ma_condition)` qui permet d'initialiser un objet de type condition ;

— `pthread_cond_wait(pthread_cond_t *ma_condition, pthread_mutex_t *mon_mutex)` qui permet d'attendre qu'une condition soit réalisée ;
— `pthread_cond_signal(pthread_cond_t *ma_condition)` qui permet de signaler à *un* thread/processus en attente qu'une condition est vérifiée ;
— `pthread_cond_broadcast(pthread_cond_t *ma_condition)` qui permet de signaler à *tous* les threads/processus en attente qu'une condition est vérifiée ;

Question 1 Implémentez des *sémaphores*, c'est-à-dire les opérations `init`, `P()` et `V()` en utilisant uniquement des verrous et éventuellement des variables de condition.

Question 2 Reprenez des exercices du cours mettant en jeu des sémaphores et remplacez-les par les sémaphores que vous venez d'implémentez. Vérifiez que le comportement des exemples n'est pas modifié.

Question 3 Maintenant, implémentez des verrous à l'aide de sémaphores, i.e les opérations `lock` et `unlock` à l'aide des opérations `P()` et `V()`. **NB :** je ne demande pas d'implémenter des variables de condition ...

4 Calculatrice parallèle

Reprenez le code de la calculatrice préfixe de l'année dernière (une version est disponible à `/net/ens/mercier/IF210/TP2/Codes/tree_norm.c`) et parallélisez le **calcul** de l'expression avec des threads.

5 Calcul de maximum

Reprenez le code du programme qui calcule le maximum d'un tableau (version avec le tableau en mémoire partagée, disponible ici : `/net/ens/mercier/IF210/TP1/Correction/exo1-2.c`) et transformez le programme pour remplacer des processus par des threads. Par quoi faut-il remplacer le tube de communication qui servait d'outil de synchronisation ?