Concurrence – OCaml

Florian Bourse

1 Threads, Mutex, Semaphores en OCaml

Pour compiler un programme utilisant un des modules `Thread`, `Mutex` ou `Semaphore`, il faut ajouter à sa commande de compilation `-I +threads unix.cma threads.cma` (ou `.cmxa` si on compile avec `ocamlopt`). Par exemple, pour compiler le fichier `test.ml` et créer un exécutable `test` :

```
ocamlc -I +threads unix.cma threads.cma test.ml -o test
```

Thread

Nous allons principalement utiliser les deux fonctions suivantes :

— `Thread.create : ('a -> 'b) -> 'a -> Thread.t`
`Thread.create f x` créer un nouveau fil d'exécution qui applique la fonction `f` à la valeur `x` et renvoie l'identifiant du nouveau fil.

— `Thread.join : Thread.t -> unit`
`join th` attends que le fil `th` se termine.

Mutex

Les 3 fonctions principales pour l'utilisation de mutex sont les suivantes :

— `Mutex.create : unit -> Mutex.t`
`Mutex.create ()` créé un nouveau mutex et renvoie son identifiant.

— `Mutex.lock : Mutex.t -> unit`
`Mutex.lock l` verrouille le mutex `l`. Si le mutex est déjà verrouillé, l'exécution est mise en pause en attendant qu'il soit déverrouillé.

— `Mutex.unlock : Mutex.t -> unit`
`Mutex.unlock l` déverrouille le mutex `l`. Si un autre fil attendait pour le verrouiller, il est réveillé.

Semaphore.Counting

Les 3 fonctions principales pour l'utilisation de sémpahores sont les suivantes :

— `Semaphore.Counting.make : int -> Semaphore.Counting.t`
`Semaphore.Counting.make n` créé un nouveau sémaphore, avec pour valeur initiale `n` qui doit être positive ou nulle.

— `Semaphore.Counting.acquire : Semaphore.Counting.t -> unit`
`Semaphore.Counting.acquire s` met en pause l'exécution jusqu'à ce que la valeur de `s` soit non nul, puis la décrémente et reprend l'exécution. (correspond à l'opération P)

— `Semaphore.Counting.release : Semaphore.Counting.t -> unit`
`Semaphore.Counting.release s` incrémente la valeur de `s` et réveille un fil en attente si besoin.

Exercice 1

Soient les deux processus P1 et P2 suivants. Ils se partagent deux sémaphores, S1 et S2, initialisés à 0.

```
P1 {                          P2 {
  procedure A1;                 procedure A2;
  V(S2);                        V(S1);
  P(S1);                        P(S2);
  procedure B1;                 procedure B2;
}                             }
```

Question 1. Quelle synchronisation a-t-on imposée sur les exécutions des procédures A1, A2, B1 et B2 ?

Question 2. Écrire le code afin d'imposer la même synchronisation pour N processus en utilisant N sémaphores.

Question 3. On peut résoudre le problème pour N processus avec uniquement deux sémaphores (et un compteur). Donner le code des processus dans ce cas.

Question 4. On peut résoudre le problème pour N processus avec uniquement deux sémaphores (et un compteur). Donner le code des processus dans ce cas.

Exercice 2

Soient 3 processus qui exécutent les programmes suivants :

```
P1 : loop A1 end_loop
P2 : loop A2 end_loop
P3 : loop A3 end_loop
```

Utiliser des sémaphores pour synchroniser ces 3 processus de telle manière que :

Question 5. les actions Ai ne soient jamais simultanées.

Question 6. les actions Ai ne soient jamais simultanées et se déroulent toujours dans l'ordre A1A2A3A1A2A3...

Question 7. les actions Ai ne soient jamais simultanées et se déroulent toujours dans l'ordre A1(A2 ou A3)A1(A2 ou A3)...

Question 8. les actions Ai se déroulent toujours séquentiellement mais dans un ordre quelconque, par exemple : (A2A1A3)(A2A1A3)...

La salle de bain

La samme de bain commune d'une pension mixte est utilisée par les filles et les garçons. Les règles suivantes sont observées :

1. au plus MAX filles utilisent simultanément la salle de bain ;
2. au plus MAX garçons utilisent simultanément la salle de bain ;
3. la salle de bain n'est jamais utilisé simultanément par des garçons et des filles ;
4. les filles ont priorité sur les garçons pour utiliser la salle de bain ;
5. les garçons ne peuvent entrer dans la salle de bain que si aucune fille n'utilise ou n'attend pour utiliser la salle de bain.

Question 9. À quelle condition une fille peut-elle entrer dans la salle de bain ? Et un garçon ?

Question 10. Identifier les événements qui peuvent bloquer un processus (fille ou garçon).

Question 11. Donner le code des quatre fonctions :

```
void boy_wants_to_use_bathroom () ;
void boy_leaves_bathroom () ;
void girl_wants_to_use_bathroom () ;
void girl_leaves_bathroom () ;
```

Algorithme de la boulangerie de Lamport

Implémenter l'algorithme de la boulangerie de Lamport pour afficher les 1000 premiers nombres en utilisant 8 fils d'exécution en remplaçant les multiples de 3 par Fizz, les multiples de 5 par Buzz et les multiples de 15 par FizzBuzz. Chaque "nombre" sera précédé de l'identifiant du fil qui l'a affiché, et sur une nouvelle ligne.

Producteurs/Consommateurs

Implémenter une version des producteurs/consommateurs où les producteurs produisent des nombres aléatoires, et les consommateurs les ajoutent à une variable si ils sont pairs ou les retirent si ils sont impairs.

2 variantes : avec une file (module `Queue`) ou avec un tableau de taille finie.