TP Synchronisation

Bastien Zigmann, Gaëtan Sorin

2 novembre 2018

1 Lecteurs - Rédacteurs

1.1 Question 1

Ce programme initialise un nombre de thread lecteur et rédacteur selon les valeurs passées en argument.
Cependant, tous les problèmes de synchronisation ne sont pas détectés. En effet si un rédacteur écrit la même valeur que la valeur précédente, notamment lors d'une lecture simultanée, une incohérence peut ne pas être détectée.

1.2 Question 2

Cette solution revient à exécuter les threads en séquence, un à la fois. On perd alors l'intérêt du multithreading.

1.3 Question 3-4

Les différentes solutions se trouvent dans leurs répertoires respectif. Chaque version peut-être compilée depuis son répertoire avec `make`.
Il est également possible de compiler toutes les versions à l'aide du script bash `build.sh`. Les exécutables sont alors placés dans le répertoire `dist`.
Un programme de test de la fifo est disponible dans `test_fifo.c`.

1.3.1 Implémentations

Les version priorité lecteur et rédacteur sont implémentées à l'identique hormis les condition de mise en attente et de signal des threads.

- Quand la priorité est donnée aux lecteurs, un lecteur prend se met en attente que si la donnée est en écriture. Un fois la lecture finie, si aucun autre lecteur n'est en lecture, il signale un rédacteur.
  Un rédacteur, quant à lui, se mettra en attente si la donnée est en écriture ou en lecture, mais aussi si un ou plusieurs lecteurs sont en attente.
  Au moment de rendre la main le rédacteur, si des lecteurs attendent, ils sont signalés à l'aide d'un broadcast sur leur condition.

- Quand la priorité est donnée aux rédacteurs, un rédacteur se mettra en attente lorsque la donnée est en lecture ou en écriture.
  Au moment de rendre la main si un ou plusieurs rédacteurs sont en attente, il en signale un. Sinon il signale tous les lecteurs en attente.
  Un lecteur se mettra en attente si la donnée est en écriture ou si au moins un rédacteur est en attente. Une fois la lecture effectuée, et si aucun autre lecteur n'est en lecture, le lecteur signale un rédacteur en attente.

Pour version avec file d'attente, on a choisi d'utiliser une file de structures composées d'une condition et d'un entier. La condition sert à mettre le thread en attente et à le réveillé.

- Un lecteur se met dans la file, si elle n'est pas vide ou si la donnée est en écriture.
  Après son attente, il reveille le lecteur suivant si il y'en a un et le sort de la file, et effectue la lecture.
  À la fin de sa lecture, si il était le dernier en lecture, il reveille le prochain dans la file et l'en sort.

- Un rédacteur s'ajoute dans la queue si la file n'est pas vide, si la donnée est utilisée (lecture ou écriture). Puis à son tour, effectue son écriture.
  Une l'écriture terminée, si la file n'est pas vide, il en sort le prochain et le réveille.