TD #9 – Mémoire partagée entre threads

Entrée dans la section critique

Implémentez un programme qui va créer `n` threads ayant chacun pour but d'incrémenter un compteur global accessible par chaque thread 10 fois. Ce compteur doit donc être protégé par une section critique. Tout comme dans le dernier exercice du TD8, la manière la plus simple d'obtenir une variable "globale" dont la valeur de sortie doit être récupérer est de la placer dans une liste, et de donner cette liste en argument à la fonction de thread.

Petits mots doux discrets

L'objectif de l'exercice est de faire passer un message (entier) d'un thread à l'autre en passant par `n` interlocuteurs. Chaque interlocuteur `i` peut transmettre/recevoir un message à/de ses voisins directs `(i-1)` et `(i+1)`.

Soit un nombre choisi aléatoirement entre 1 et 100 par le thread 1, faites-en sorte que le dernier thread le reçoive et l'affiche. L'attente des threads `i` doit être fait à l'aide de conditions.

Mécanisme de barrière

L'objectif de l'exercice est d'offrir à l'utilisateur un mécanisme de barrière entre `n` threads réutilisable, ceci en utilisant uniquement des `Locks` et des `Conditions`. Il est proposé de l'implémenter en suivant plusieurs étapes :

Barrière avec 3 threads

La synchronisation est effectuée sur 3 threads, les deux premiers attendent que le troisième les libère.

Barrière avec n threads

Les `(n-1)` premiers threads attendent que le dernier les libère.

Ré-utilisabilité

La barrière peut être utilisée plusieurs fois dans le programme, avec le même nombre de threads se synchronisant.