TP1
Threads

GIF 4104
Programmation parallèle et distribuée

Pier-Luc Auger (PLAUG2 - 910 098 011)
Mathieu Garon (MAGAR220 - 111 005 164)

Université Laval

Rapport TP1

Mise en contexte

Dans ce tp, nous devions arriver à synchroniser des fils, de sorte que les thread pairs s'exécute d'abord dans l'ordre croissant, puis les fils de numéros impairs dans l'ordre décroissant. Pour ce faire, nous avons entre autre utilisé la librairie thread du standard C++11.

Méthode utilisée

Pour synchroniser les fils, deux variables de conditions sont nécessaires: condition pair et condition impair. Le problème est décrit de la manière suivante:

- Les fils impairs ne peuvent pas être éxecuter avant que tous les fils pair ne soient terminés.
- Les fils pair doivent être exécutés en ordre croissant.
- Les fils impairs doivent être exécutés en ordre décroissant.

Pour palier à la première condition, un booléen est utilisé pour indiquer que les fils pair sont tous exécutés. De cette manière, une fois les fils lancés, il suffit d'attendre la fin d'exécution des fils pair (thread[pair].join()) et de changer le booléen pour ensuite notifier la variable de condition des fils impairs.

Pour synchroniser les fils pairs, une variable contenant la plus petite valeur paire de fils exécuté est partagée. Si l'identifiant du fil est le même, il est exécuté et la valeur est incrémenté de deux. Il est important de garder le mutex lors de la modification de la variable. Ensuite la condition paire est notifiiée à tous les fils pour exécuter le prochain si applicable.

Finalement, la même procédure est utilisé pour les fils impairs. La plus haute valeur impaire est partagée et décrémentée lorsqu'un fil est exécuté.

Les deux conditions pour la solution du problème sont donc:

- tid >= this->HighestOdd && this->OddReady;
- tid <= this->LowestEven;

Finalement, le fil principal s'assure de bien récupèrer la mémoire des fils avant de terminer. De plus, un mutex est utilisé pour faire l'acquisition de std::cout pour assurer l'écriture cohérente sur le terminal.

Lancement de l'exécutable

Pour utiliser notre programme, il s'agit d'utiliser `cmake` pour générer le makefile. Cela permettra ensuite d'utiliser la commande `make` pour compiler le programme. Le programme prend en entrée le nombre de fils à faire exécuter.