LSINF1252 - Rapport

Vermeulen Lucas, Houyoux Joachim

May 11, 2018

1 Introduction

Pour ce projet, il nous a été demandé de compléter un programme pouvant calculer et afficher des fractales. La librairie étant déjà créée, nous devions réaliser le reste du programme afin que, sur base d'un ou plusieurs fichiers, le programme calcule les fractales et affiche celle dont la moyenne des points est la plus élevée dans le fichier mentionné en dernier lieu lors de l'appel.

2 Structure du programme

Afin de réaliser ce programme sur base d'un modèle de multi-threading, nous nous sommes basés sur le concept des producteurs-consommateurs. En effet, nous initialisons un buffer global de structures fractales qui accueillera toutes les fractales que le producteur créera et dont le consommateur se servira.
Le producteur a pour rôle de de prendre un fichier en argument, de distinguer dans ce fichier les lignes correspondantes à l'unique format valable de description de fractale, de créer les fractales décrites par ces lignes et de les mettre dans le buffer. L'insertion d'une fractale dans le buffer partagé est bien entendu une section critique qu'un seul thread à la fois peut modifier. Il est donc nécessaire de la protéger avec un mutex.
Le consommateur quant à lui, a pour rôle de prendre les fractales dans le buffer, de les calculer, et de comparer si la dernière fractale calculée possède une valeur moyenne plus ou moins élevée qu'une fractale globale MM (Meilleure Moyenne) et de changer cette fractale globale par la dernière fractale si cette dernière a une moyenne plus élevée. L'accès au buffer et le changement de la fractale globale MM sont des sections critiques protégées par des mutex.
Enfin, le main s'occupe dans un premier temps de créer les threads de lecture utilisant la fonction liée au producteur et les threads de calcul utilisant la fonction liée au consommateur. A la fin du main, on attend simplement que tous les threads de calcul soient terminés avant d'afficher la fractale globale MM possédant la meilleure moyenne.

3 Choix d'implémentation

Premièrement, afin de de régler pas mal de segmentation fault,nous avons initialisé une variable globale isEmpty de type int. Celle-ci s'incrémente d'une unité dans la section critique du producteur, lorsqu'une fractale est insérée dans le buffer. De plus, isEmpty est réduit de une unité dans la section critique du consommateur, lorsque qu'une fractale est retirée du buffer. Ainsi nous pouvons dire aux threads de calculs de s'arrêter lorsque isEmpty est égal à 0 et qu'il n'y a donc plus de fractales dans le buffer. Nous avons également initialisé un isReading qui compte le nombre de thread de lecture qui lisent encore un fichier, lorsque isReading est égal à 0, cela veut donc dire qu'aucun producteur n'est encore actif et que le buffer ne se remplira plus.

4 Commentaires sur le projet

Tout d'abord, nous tenons à dire que nous n'avons pas été aussi loin que demandé. En effet, nous ne traitons pas le cas où 2 fractales ont le même nom, ni le cas où 2 fractales possèdent la même meilleure moyenne. Nous ne faisons pas non plus le traitement du "-" dans le terminal, nous ne lisons donc pas de fractales sur l'entrée standard. Nous avons eu aussi une agréable surprise en essayant de mettre plus de threads de calcul que de fractales. En effet, les threads "inutiles" sont bloqués dans une boucle infini, même lorsque tout le buffer a été vidé. Après un nombre incalculable de tests afin de régler ce problème, nous nous avons trouvé une sorte d'alternative, mais il fallait spéculer sur la vitesse des threads, le programme fonctionnait donc par intermittence. Nous avons donc décidé de ne pas accepter cette option de la part de l'utilisateur.

5 Conclusion

En conclusion, notre programme souffre d'un nombre de bugs conséquent, probablement dû à son implémentation peu modulable. Les tests nous permettent pourtant de dire que le fait de charger des fichiers contenant des fractales et les calculer fonctionne, ainsi que la récupération de la meilleure moyenne et l'affichage des bonnes fractales. De plus l'option -d et –maxthreads fonctionne parfaitement .