Projet Fractales - Rapport final

Gerard Julien
Zanella Maxime

May 2018

1 Introduction

Après la réflexion sur l'architecture de notre programme, nous nous sommes lancés dans l'implémentation. Bien sûr, nous avons du faire des choix de conception, décrits ci-dessous. Malgré quelques points à améliorer, notre programme renvoie la ou les fractale(s) voulue(s).

2 Choix d'implémentations

Pour réaliser ce programme, nous avons du prendre certaines décisions. Tout d'abord, pour l'architecture globale nous avons implémenté 2 producteurs-consommateurs (nécessitant donc 2 mutex et4 sémaphores) : des producteurs lisant les fichiers (un thread de lecture par fichier ajouté à cela un thread pour lire l'entrée standard) et créant les strucutres fractales. Ensuite, des consommateurs qui vont reprendre ces structures pour calculer chaque fracale (un nombre fixe de threads de calcul, initialement à 4 si, on entre pas d'arguments indiquant le nombre de threads). A ce sujet nous avons décidé d'introduire une variable contenant la moyenne des valeurs de la fractale et calculée directement par les threads de calculs ci-dessus. En effet, cela rendait le code plus efficace car il suffisait d'ncrémenter la valeur avec celle renvoyée par méthode itérative, évitant ainsi de parcourir 2 fois le même tableau.
Comme énoncé précédemment, nous avons un deuxième système de producteur-consommateur, le principe est toujours le même sauf qu'ici les producteurs sont les consommateurs du premier système de producteur-consommateur et le consommateur est un unique thread comparant les fractales au fur et à mesure qu'elles sont calculées. Nous avons décidé de mettre un unique thread pour cette dernière partie car cela simplifiait la coordination et surtout la condition d'arrêt du thread pour enfin par la suite créer le fichier bitmap. De plus, la moyenne ayant déjà été calculée, le nombre d'opérations est largement inférieur au calcul des fractales.

Au niveau de l'implémentation par rapport aux consignes, nous n'avons pas eu le temps d'implémenter une partie permettant de vérifier si les noms de fractales sont bien tous différents. De plus, nous avons du faire face à de nombreuses erreurs de segmentation fault lors de l'appel à certaines fonctions de la librairie, nous avons donc été contraints de rajouter ces quelques fonctions dans notre main même si cela n'est évidemment pas la meilleure implémentation.
De plus, nous n'avons pas eu le temps d'implémenter plusierus tests CUnit.

3 Evalutation générale du projet

Dans l'ensemble notre programme nous parait assez efficace même si certaines choses sont à améliorée (détection des noms semblables, etc.). De plus, faire plus de tests CUnit aurait été bienvenu, surout pour débugguer le code. Certaines erreurs subsistent, par exemple selon l'exécution, il arrive qu'un fichier bmp ne se crée pas comme voulu alors que tous les autres sont corrects. Il serait possible aussi d'améliorer la lecture sur l'entrée standard, en effet notre programme lit ce que l'utilisateur rentre dans la main, empêchant le lancement des threads en fond et donc le calcul des fractales contenues dans les fichers mis en argument avant de lancer tous les threads.
Enfin, nous n'avons pas eu l'occasion de vraiment nous servir du github, car nosu avons commencé à travailler sur un seul ordinateur.
Il reste donc des améliorations à y apporter.