LSINF1252 : Systèmes Informatiques
Projet Fractales

de Villenfagne Clothilde - 33991600
de Voghel Brieuc - 59101600

10 mai 2018

CECI EST UN PREMIER CROQUIS

TODO : adding multiple means

TODO : -v verbose

TODO : talking about tests

1 Introduction

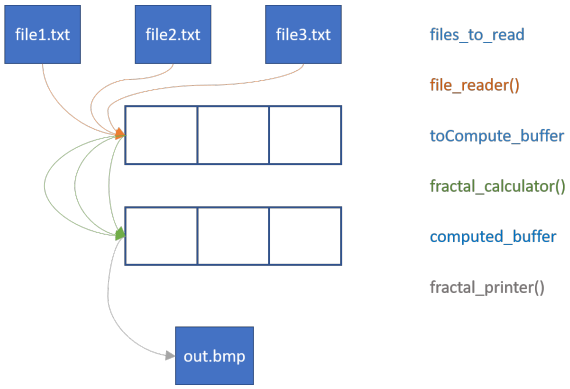
Dans le cadre du cours de Systèmes Informatiques, il nous a été demandé d'implémenter un programme en C, qui prend comme argument des fichiers contenant une liste de fractales. Le programme doit retourner la fractale dont la moyenne des valeurs en chaque point est la plus élevée sous forme d'une image BMP. Dans ce rapport nous allons tout d'abord décrire nos choix d'implémentation. Ensuite, nous allons faire une évaluation qualitative et quantitative de notre travail et comparer notre programme dans son efficacité de multithreading.

2 Choix d'implémentation

Le premier point auquel nous avons porté notre attention est la structure de la fractale. Comme nous ne voulions pas perdre du temps à parcourir plusieurs fois chaque pixel pour en premier calculer sa valeur et puis après encore repasser pour calculer la moyenne, nous avons décidé que lors du calcul de la valeur d'un pixel, cette valeur soit additionnée à une valeur totale. Ce total, il ne suffira plus qu'à le diviser par la surface de la fractale pour obtenir la moyenne. Ceci a directement été implémenté dans `fractal_set_value()`.

Notre deuxième point d'importance est le fait d'avoir implémenté les buffers sous forme d'une pile (liste chaînée avec `pop` et `push`). De ce fait, il nous semblait d'être plus facile d'accéder au buffer. Ce buffer pourrait d'ailleurs avoir une taille virtuellement infinie, mais nous limitons ceci quand-même grâce aux sémaphores. Comme l'ordre de calcul des fractales n'a pas d'importance, le fait d'avoir implémenté une pile LIFO (last in, first out) n'est pas dérangeant.

Nous avons implémenté une librairie externe (similaire à `libfractal`) pour cette liste chaînée : `libstack`. Elle est compilable séparément et nous l'importons donc dans notre main.



Afin de de pas calculer deux fractales ayant le même nom, nous mettons à jour un string contenant tous les noms des fractales précédemment calculées. Ainsi, nous pouvons vérifier si une fractale a déjà été

calculée. Le choix de stocker les noms dans un string nous avait l'air le plus facile et rapide à l'exécution, pas de perte de temps avec `malloc` par exemple.

3 Piste d'amélioration

Le premier élément que nous avons identifié comme étant incomplet (et donc éventuellement complétable à partir de l'état actuel), est le fait que si plusieurs factales ont la même moyenne et que cette moyenne est la plus haute moyenne de toutes les fractales, seule une fractale est sortie sous `FichierOut.bmp`. Nous pouvons imaginer réutiliser une pile de fractale où nous pourrions stocker les fractales ayant la plus haute moyenne du moment.

4 Évaluation qualitative et quantitative

Nous avons effectué plusieurs tests avec notre programme. Notamment sur des fichiers plus larges contenant jusqu'à 1 millions de fractales. Ces tests, nous les avons d'abord fait avec un seul, puis plusieurs threads de calcul différents. Les résultats étaient surprenants. La!!!figure X!!! en donne un aperçu.

5 Conclusion