Polytech Sorbonne

Compte rendu TP2 - HPC

Ensemble de Mandelbrot

Romaric Kanyamibwa

MAIN4

Année 2017 - 2018

Table des matières

1 Introduction

L'objectif du ce TP est de mettre en œuvre le calcul parallèle des ensembles de Mandelbrot et la comparaison du temps d'exécution avec la version séquentielle. Avec ce TP nous allons démontrer que avec un programme parallèle nous pouvons accélérer de façon considérable nos programmes séquentielles.

2 Ensemble de Mandelbrot

On définit comme ensemble de Mandelbrot les points du plan complexe $\mathbb{C}$ vérifiant la suite récurrente de nombres complexes suivante:

$$\begin{cases} z_0 = 0 \\ z_{n+1} = z_n^2 + c \end{cases}$$

quand z_n converge.

3 Programme Séquentielle vs Programme Parallèle

Pour comparer la performances de versions parallèle et de la version séquentielle nous allons dessiner l'ensemble sur le plan complexe.

Nous allons commencer avec la version non-parallélise qui se trouve dans le fichier *mandel.c* et ensuite nous allons la rendre parallèle sur le fichier *mandelbrot.c*.

Dans le graphe ci-dessous nous allons travaillé avec les points z définit sur le domaine D=$\{z = x + iy$, $(x, y) \in$ [-1.5,-0.1]x[-1.3,0.1]$\}$
Dans notre exemple l'Incrément définit par $x_{inc} = \frac{x_{max} - x_{min}}{w-1}$ et $y_{inc} = \frac{y_{max} - y_{min}}{h-1}$ sera : x_{inc} =0.000250313 , y_{inc} =0.000250313
Le profondeur est la chose que nous allons faire varier pour avoir les différents temps de calcul: 10-1000
Nous allons générer des images de 800x800

Figure 1 – *Prof=10*

Figure 2 – *Prof=15*

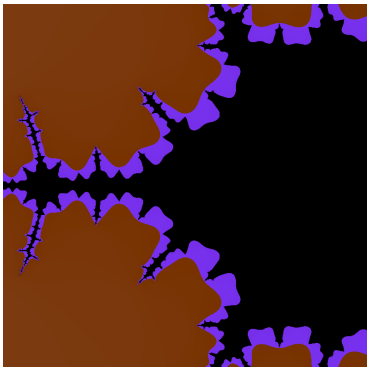

Figure 3 – *Prof=20*

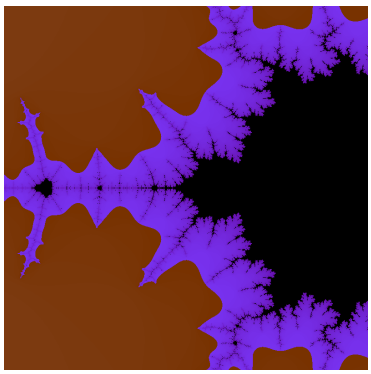

Figure 4 – *Prof=50*

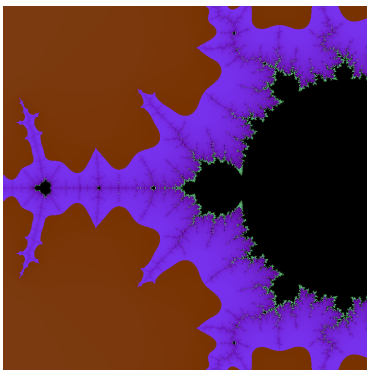

Figure 5 – *Prof=100*

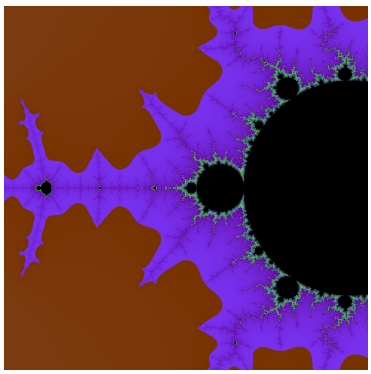

Figure 6 – *Prof=1000*

A part le profondeur nous allons aussi faire varier le nombre de processeurs ($NP \in \{1, 10, 20\}$) pour mettre en évidence l'accélération qui le parallélisme va offert à nos calculs.

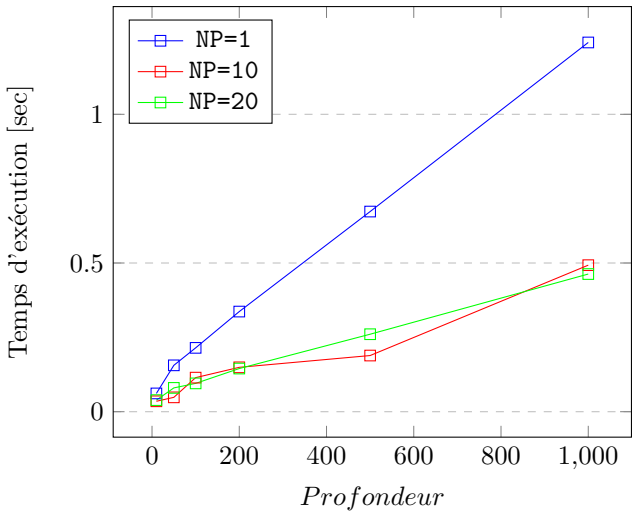

Figure 7 – *Temps d'exécution du programme pour de profondeurs différents*

4 Conclusion

En faisant varier le profondeur et en changeant le nombre de processeurs on remarque qu'on arrive à faire le programme 2,5 fois plus rapide rien qu'on utilisant 10 processeurs.Néanmoins on voit que la parcellarisation a une limite quant au nombre de processeurs. Dans le graphe on voit que avec 10 processeurs ou avec 20 processeurs le temps de calcul reste quasiment le même et de fois on a même un temps de calcul plus grand avec 20 processus que avec 10. En fait ce ralentissement est doué aux messages que les 20 processus doivent échanger entre eux pour transmettre leur calculs. Ce pour cela que quand on fait du parallélisme il faut trouver le nombre optimal de processus qu'on utilisera pour que la communication inter processus ne nous pénalise pas.