PROGRAMMATION PARALLÈLE

PROJET

20 mars 2017

Kilian Fatras

Table des matières

1 PARALLÉLISME

1.1 Par ligne

Dans un premier temps, j'ai parallélisé le programme avec l'aide de la bibliothèque MPI. Je me suis concentré sur une parallélisation par ligne de l'image avec l'aide d'un schéma maître-esclave. Le schéma est le suivant : le maître envoie une ligne qui n'a pas encore été calculé à un de ses esclaves, qui vient de terminer le calcul d'une ligne précédente, l'esclave calcul donc la ligne et renvoie le résultat au maître. Le maître prend alors en compte les résultats et renvoient une nouvelle ligne à l'esclave tant que toutes les lignes traitées.

1.2 Par bloc

J'ai par la suite voulu envoyer un bloc de lignes à mon esclave et non pas une ligne. J'envoie donc des blocs de 16 lignes à mes esclaves et ils calculent chaque ligne du bloc avant de renvoyer le résultat.

1.3 Open MP

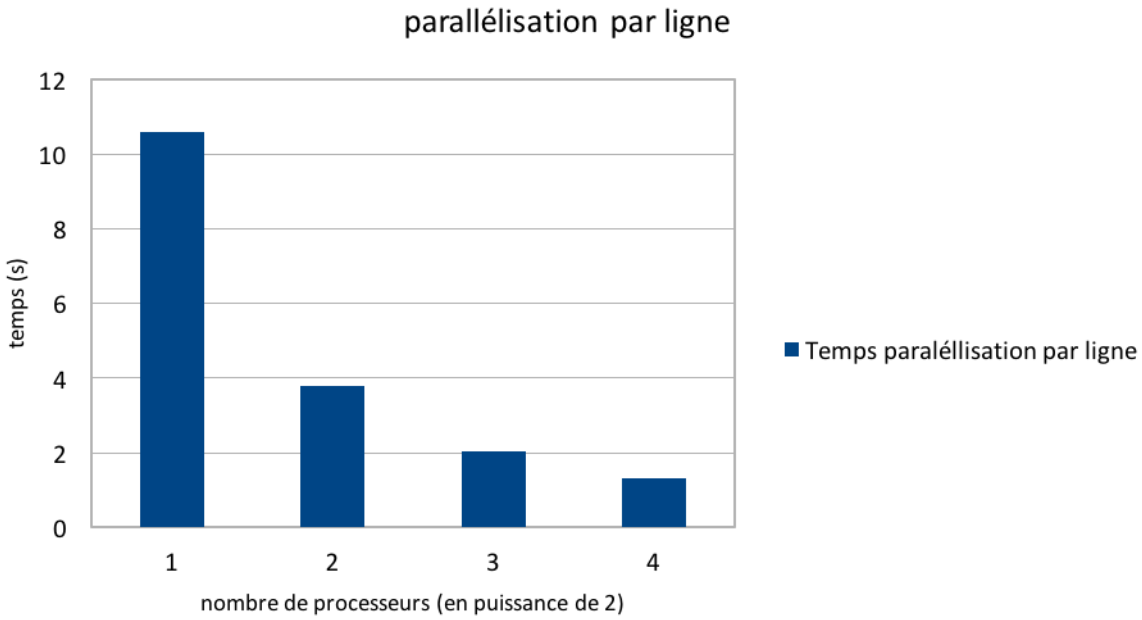
Enfin, lorsque j'ai mis en place l'interface maître-esclave, il me restait à paralléliser la boucle for qui appelait la fonction trace. En effet, cette fonction est extrêmement lourde a appelé, c'est pourquoi nous parallélisons cette boucle avec l'aide d'Open MP.

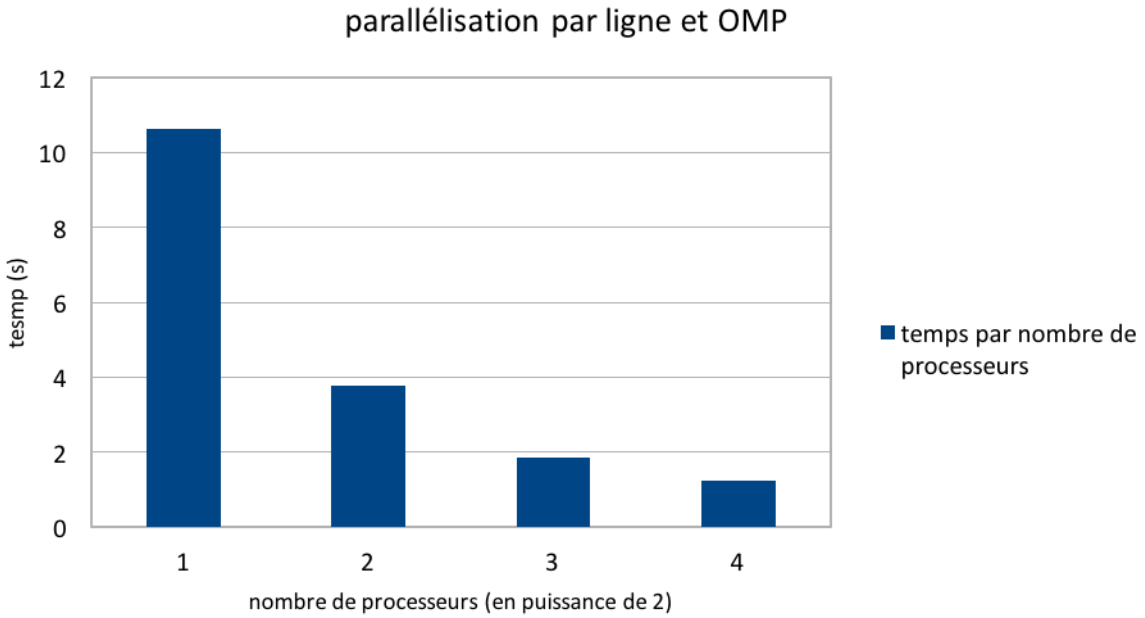
Lors de la parallélisation par ligne, j'ai utiliser une parallélisation Open MP dynamique sur chaque pixel de la ligne. Lorsque je parallélise le programme par bloc de lignes, je parallélise le calcul des lignes du bloc et non pas chaque pixel de chaque ligne.

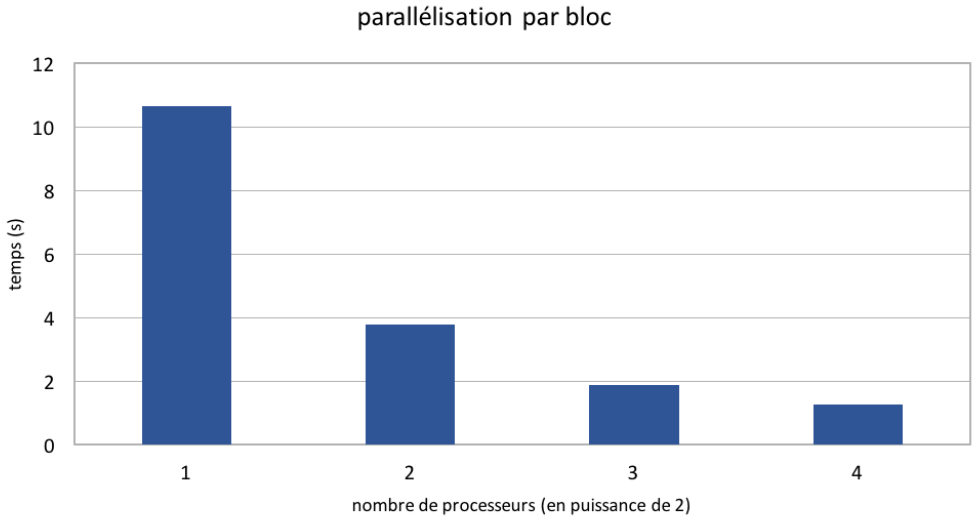
2 RÉSULTAT

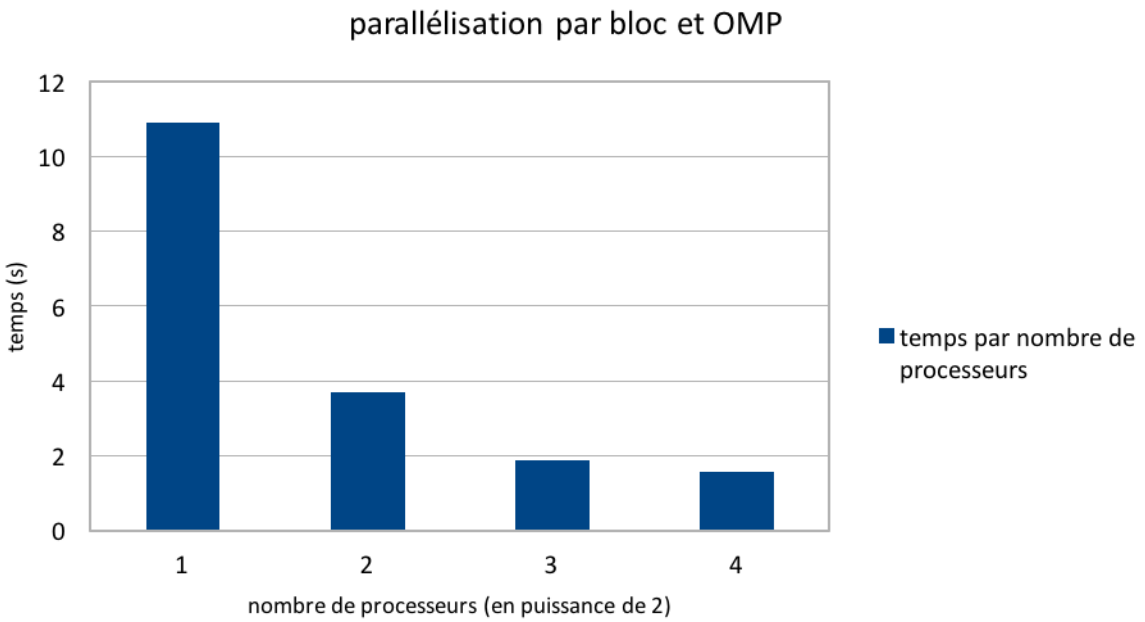
Lors de l'exécution du programme, j'affiche le temps d'exécution de la fonction 'render' sur chaque processeur. Vous pouvez voir les graphiques suivants montrant le temps d'exécution de cette fonction sur un processeur en fonction du nombre total de processeur. J'ai travaillé avec 2, 4, 8 et 16 processeurs.

2.1 Graphique




parallélisation par ligne et OMP
12
10
8
6
4
2
0
tesmp (s)
1
2
3
4
nombre de processeurs (en puissance de 2)
temps par nombre de processeurs


parallélisation par bloc
12
10
8
6
4
2
0
temps (s)
1
2
3
4
nombre de processeurs (en puissance de 2)



2.2 Interprétation

Nous Observons que la parallélisation par ligne est plus efficace que la parallélisation par bloc lorsque l'on ne fait pas appel à la bibliothèque Open MP. En effet, le nombre important de créations et de destructions de threads ralentissent le programme.

Si on ne fait pas appel à la bibliothèque Open MP, on observe une légère accélération de la parallélisation avec une stratégie de parallélisme par bloc.