Méthodes Numériques - TP02 : Multiplications de Matrices

Abonnenc Alicia - Gilles Bonhoure

Mars 2016

1 Présentation du Sujet

Le but de ce TP était de réaliser divers calculs matriciels afin de définir lequel était le plus optimisé. Nous avons donc réalisé trois types de multiplications de matrices différents ainsi que des opérations complémentaires :

1. Une multiplication calculant ligne par ligne la matrice résultat
2. Une multiplication calculant colonne par colonne la matrice résultat
3. Une multiplication calculant "bloc" par "bloc" la matrice résultat (on découpe la matrice résultat puis on calcule ligne par ligne ces sous-matrices)
4. Une addition de matrices
5. Une multiplication de matrice par un vecteur
6. Gaxpy

2 Résultats des tests

Voici quelques traces des résultats obtenus à partir de notre programme :

```
Calculs sur 5 matrices
Dimension des matrices : 1024
Nombre de threads : 4
Dépassement du cache : oui
// MULTIPLICATIONS //
Multiplication | Lignes de la matrice de sortie
time = 17132.173ms
MFLOPS : 626.739968
Multiplication | Colonnes de la matrice de sortie
time = 19503.172ms
MFLOPS : 550.547264
Multiplication | Par blocs de 32 valeurs de la matrice de sortie
time = 16715.935ms
MFLOPS : 642.346304
// SOMME //
time = 14.384ms
MFLOPS : 364.493888
// MULT MATxVECT //
time = 10.208ms
```

```
MFLOPS : 513.604992
// GAXPY //
time = 10.127ms
MFLOPS : 518.218624
```

3 Analyse des résultats

On remarque que le temps de calcul par ligne ou par colonne est assez variable, et l'un n'est pas sensiblement meilleur que l'autre. (Quand il n'y a pas de dépassement du cache). Mais avec un dépassement de cache, la méthodes utiklisant le calcul par lignes est plus performante en terme de mégaflops.
On voit également que découper la matrices résultat par bloc pour la calculer est très intéressant. Cela s'explique par le fait que l'on peut stocker ces blocs dans le cache, et donc que l'on y accède plus rapidement lors des calculs, contrairement aux autres calculs.