

## TP n° 1 : Mesures de temps d'exécution.

### 0. Introduction

Les objectifs de ce TP sont les suivants :

- se familiariser avec l'utilisation du compilateur C d'Intel (ICC) sous Linux, notamment pour mesurer de manière relativement précise les temps d'exécution de programmes.
- Observer l'impact de certaines options de compilation
- Mettre en évidence l'influence des caches sur les temps d'exécution

Ce TP est à effectuer sur des PC équipés de processeurs Intel

On demande un compte rendu de TP par binôme à rendre le mercredi 4 Octobre.

### 1. Copie de tableaux

#### Accès ligne ou accès colonne

Soit un programme C qui effectue la copie d'un tableau d'entiers 32 bits  $N \times N$  dans un autre en utilisant successivement les deux algorithmes suivants :

```
Version ij
for (i=0; i<N; i++)
    for (j=0; j<N; j++)
        Y[i][j]=X[i][j];
```

```
Version ji
for (j=0; j<N; j++)
    for (i=0; i<N; i++)
        Y[i][j]=X[i][j];
```

Donner le temps d'exécution/élément (c'est-à-dire le temps d'exécution total divisé par  $N*N$ ) pour chacune des versions pour  $N=10$ ,  $N=50$ ,  $N=100$ ,  $N=500$ .

Avec l'option de compilation `-O2` et sans (puis avec) l'option de compilation `-xW`

#### Copie d'octets et copie de mots de 32 bits

Soient deux programme C qui effectuent la copie d'un tableau  $N \times N$  octets et de  $N \times N$  « floats » dans un autre

Donner le temps d'exécution/élément (c'est-à-dire le temps d'exécution total divisé par  $N*N$ ) pour chacune des versions pour  $N=10$ ,  $N=50$ ,  $N=100$ ,  $N=500$ .

Avec l'option de compilation `-O2` et sans (puis avec) l'option de compilation `-xW`

### 2. Produit scalaire

En utilisant successivement des données entières (int) et des données flottantes (float), donner le temps d'exécution/élément (c'est-à-dire le temps d'exécution total divisé par  $N$ ) pour chacune des versions pour  $N=10$ ,  $N=50$ ,  $N=100$ ,  $N=500$ .

Avec l'option de compilation `-O2` et sans (puis avec) l'option de compilation `-xW`

### 3. Produit de matrices avec des données flottantes (float)

#### Produit de matrices ijk

Donner le temps d'exécution/élément (c'est-à-dire le temps d'exécution total divisé par  $N*N$ ) pour chacune des versions pour  $N=10$ ,  $N=50$ ,  $N=100$ ,  $N=500$ .

- Avec l'option de compilation `-O2` et sans (puis avec) l'option de compilation `-xW`

#### Produit de matrices ikj

Donner le temps d'exécution/élément (c'est-à-dire le temps d'exécution total divisé par  $N*N$ ) pour chacune des versions pour  $N=10$ ,  $N=50$ ,  $N=100$ ,  $N=500$ .

- Avec l'option de compilation `-O2` et sans (puis avec) l'option de compilation `-xW`

#### 4. Produits de matrices avec des données entières (int)

Refaire la partie 3 en utilisant des matrices « entières » au lieu de matrices « flottantes ».

#### 5. Annexe : Mesures de temps

Les mesures de temps sous Linux sont données par les fonctions suivantes, qui donnent le nombre de cycles d'horloge.

Pour obtenir les temps d'exécution, on exécute les programmes un certain nombre de fois, et on fait la moyenne des temps obtenus en enlevant les valeurs « aberrantes » (très supérieures aux autres).

```
double dtime();
long long readTSC ();

long long readTSC ()
{
    long long t;
    asm volatile (".byte 0x0f,0x31" : "=A" (t));

    return t;
}
double dtime()
{
    return (double) readTSC();
}
```

#### Mesure du temps d'exécution

```
double t1,t2 ;//déclaration des variables

t1 = dtime();

//Partie du programme dont on mesure le temps d'exécution.

t2 = dtime();

dt = t2-t1; // Nombre de cycles d'horloge processeur
```

#### 6. Annexe : utilisation de ICC

Documentation :

[http://www.intel.com/software/products/compilers/clin/docs/main\\_cls/index.htm](http://www.intel.com/software/products/compilers/clin/docs/main_cls/index.htm)

Les programmes à utiliser et modifier sont disponibles dans la page Web du M1

<http://www.lri.fr/~de/ArchiM1-0809.htm>