

Rapport Projet OHBPC

Présenté par :

Hamza Amine DEMIGHA

**Année universitaire : 2023/2023**

Table des matières

[**Environnement des tests** 3](#_Toc119276794)

[**Comparaison des tests** 3](#_Toc119276795)

[**A – DGEMM :** 3](#_Toc119276796)

[**1 – Comparaison entres les versions :** 4](#_Toc119276797)

[**2 – Comparaison entres les flags d’optimisation :** 4](#_Toc119276798)

[**3 – Comparaison entres les compilateurs :** 5](#_Toc119276799)

[**Conclusion :** 5](#_Toc119276800)

[**B – DOTPROD :** 6](#_Toc119276801)

[**1 – Comparaison entres les versions :** 6](#_Toc119276802)

[**2 – Comparaison entres les flags d’optimisation :** 7](#_Toc119276803)

[**3 – Comparaison entres les compilateurs :** 7](#_Toc119276804)

[**Conclusion :** 7](#_Toc119276805)

[**C – REDUC :** 8](#_Toc119276806)

[**1 – Comparaison entres les versions :** 8](#_Toc119276807)

[**2 – Comparaison entres les flags d’optimisation :** 9](#_Toc119276808)

[**3 – Comparaison entres les compilateurs :** 9](#_Toc119276809)

[**Conclusion :** 9](#_Toc119276810)

# **Environnement des tests**

Pour tous ses mesures de performance j’ai utilisé le cartable numérique connecté au secteur, avec le CPU qui tourne à une fréquence stable.

Les informations sur le CPU sont sauvegardées dans les fichiers **« cpuinfo.txt »** et **« lscpu.txt »** à l’aide des commandes :

* **$lscpu**
* **$cat /proc/cpuinfo**

Les informations sur les caches de données :

**/sys/devices/system/cpu/cpu0/cache/index0/\*** ---> L1.txt

**/sys/devices/system/cpu/cpu0/cache/index2/\*** ---> L2.txt

**/sys/devices/system/cpu/cpu0/cache/index3/\*** ---> L3.txt

On pinne le processus sur un cœur de calcul numéro 1 :

**cpupower -c 1 frequency-set -g performance**

Et on exécute le programme a l’aide de la commande **taskset** :

**taskset -c -1 ./fichier\_exe n r**

Pour tout ces tests on fixe taille de matrice **n = 100** et kernel répétitions **r = 100**

Les compilateurs utilisées sont : **{ GCC, CLANG, ICX }**

Les flags d’optimisations sont : **{ -O1, -O2, -O3 }**

Les graphes de comparaison sont faits avec **GNUPlot**

# **Comparaison des tests**

Pour la discussion de mesures de performance générés par ce programme, on compare entre les versions avec différents compilateurs et flags d’optimisations par leurs unités de débit de données (Mib/s) :

## **A – DGEMM :**

Ce programme donne les mesures de performance de différentes versions de calcul multiplication matrice \* matrice.

Les versions sont : **IJK , IKJ , IEX , UNROLL4 , CBLAS**

On a ajouté une autre version de déroulage x8 **UNROLL8** à ce programme.

Une image contenant table

Description générée automatiquement

### **1 – Comparaison entres les versions et les flags d’optimisation :**

* + 1. Avec CC = GCC, OFLAGS = -O3

//1

* + 1. Avec CC = CLANG, OFLAGS = -O3

//4

* + 1. Avec CC = ICX, OFLAGS = -O3

//7

**Discussion :**

A travers les résultats obtenus, on peut dire que la meilleure version de calcul de multiplication matrice \* matrice est la version : **CBLAS** car son Mib/s est le grand avec tous les compilateurs et les différents flags d’optimisation.

c’est clair que le flag d’optimisation -O1 n’est pas performant aussi que les flags -O3 et -O2 car c’est le niveau d’optimisation le plus basique.

Pour les flags -O2 et -O3 les résultats presque les mêmes, mais le flag -O3 > -O2.

* -O1 : C'est le niveau d'optimisation le plus basique. Le compilateur va essayer de produire un code plus rapide et plus compact sans prendre trop de temps de compilation. C'est très basique mais ça fait toujours le travail.
* -O2 : Un échelon au-dessus de -O1 . C'est le niveau *recommandé* d'optimisation si vous n'avez de besoin spécifique. -O2 active quelques options de plus que -O1 . Avec -O2 , le compilateur va essayer d'augmenter la performance sans compromettre la taille et sans prendre trop de temps en compilation.
* -O3 : C'est le plus haut niveau d'optimisation possible. Il active des optimisations qui sont coûteuses en terme de temps de compilation et d'usage de la mémoire. Compiler tous vos paquets avec -O3 ne garantit pas une amélioration de la performance. En réalité, dans de nombreuses situation, cela ralentit le système à cause des binaires plus volumineux qui réclament plus de mémoire. De plus cette option est réputé casser de nombreux paquets. C'est pourquoi utiliser -O3 n'est pas recommandé.

### **3 – Comparaison entres les compilateurs :**

* + 1. Avec OFLAGS = -O1

//1

* + 1. Avec OFLAGS = -O2

//2

* + 1. Avec OFLAGS = -O3

//3

**Discussion :**

A travers les résultats obtenus, on peut dire que le meilleur compilateur pour ce programme est ICX. Les 2 autres compilateurs GCC et CLANG leurs mesures de performance sont proches.

### **Conclusion :**

Au final, pour avoir les meilleures mesures de performance du calcul de la multiplication matrice \* matrice la version recommandée est **CBLAS** avec le compilateur **ICX** et le flag d’optimisation **-O3**.

## **B – DOTPROD :**

Ce programme donne les mesures de performance de différentes versions de calcul produit scalaire.

Les versions sont : **base**

On a ajouté une autre version de déroulage x8 **UNROLL8** à ce programme.

Une image contenant texte

Description générée automatiquement

### **1 – Comparaison entres les versions et les flags d’optimisation :**

* + 1. Avec CC = GCC, OFLAGS = -O1, -O2, -O3

//1

* + 1. Avec CC = CLANG, OFLAGS = -O1, -O2, -O3

//2

* + 1. Avec CC = ICX, OFLAGS = -O1, -O2, -O3

//3

**Discussion :**

### **2 – Comparaison entres les compilateurs :**

* + 1. Avec OFLAGS = -O1

//1

* + 1. Avec OFLAGS = -O2

//2

* + 1. Avec OFLAGS = -O3

//3

**Discussion :**

### **Conclusion :**

## **C – REDUC :**

Ce programme donne les mesures de performance de différentes versions de calcul somme d’un tableau.

Les versions sont : **base**

On a ajouté une autre version de déroulage x8 **UNROLL8** à ce programme.

Une image contenant texte

Description générée automatiquement

### **1 – Comparaison entres les versions et les flags d’optimisation :**

* + 1. Avec CC = GCC, OFLAGS = -O1, -O2, -O3

//1

* + 1. Avec CC = CLANG, OFLAGS = -O1, -O2, -O3

//2

* + 1. Avec CC = ICX, OFLAGS = -O1, -O2, -O3

//3

**Discussion :**

### **2 – Comparaison entres les compilateurs :**

* + 1. Avec OFLAGS = -O1

//1

* + 1. Avec OFLAGS = -O2

//2

* + 1. Avec OFLAGS = -O3

//3

**Discussion :**

Ce qu’il faut en conclure, à mon sens, c’est que GCC et CLang sont des solutions comparables. Il y a deux compilateurs libres de très grande qualité l’un comme l’autre. De plus, la compétition informelle dans laquelle ils se trouvent fait que l’un et l’autre s’améliore. Tout cela au bénéfice des utilisateurs.  
  
Du coup, impossible de dire qui est le meilleur, ça vaut le coup de tester les deux. Dans un projet concret, hé bien, le choix de l’un plutôt que l’autre est probablement surtout une question d’habitude.

### **Conclusion :**