UNIVERSITÉ CÔTE D'AZUR

MASTER INFORMATIQUE

Rapport de TP

Travaux Pratiques sur la Hiérarchie Mémoire

Réalisé par
Francois Flandin

Encadré par
Pr Sid Touati

Master Informatique
à Université Côte d'Azur
Premier semestre de l'année 2024-2025

Table des matières

1 Introduction

Ce rapport présente le travail réalisé dans le cadre d'un TP visant à explorer les performances de la hiérarchie mémoire d'un processeur. L'objectif était d'analyser le comportement des caches et de la mémoire centrale à travers des tests pratiques, tout en utilisant des outils simples et adaptés.

En partant des caractéristiques techniques du processeur de test, on a mis en place des expériences pour mesurer les temps d'accès à la mémoire et la bande passante. Les données obtenues ont ensuite été analysées pour mieux comprendre les mécanismes en jeu, comme le prefetch matériel ou les optimisations logicielles.

Ce document retrace les étapes principales de cette étude, les méthodes utilisées, et les enseignements tirés des résultats.

Environnement Expérimental

Micro-Architecture

Dans cette partie sera detaillée la micro-architecture de la machine de tests.

Nom de modèle : 11th Gen Intel(R) Core(TM) i5-1135G7 @ 2.40GHz
Taille des adresses : 39 bits physical, 48 bits virtual
Coeurs physiques : 4
Taille de ligne de cache : 64 octets

Graphical Topology				
Coeurs	0 4	1 5	2 6	3 7
Cache L1	48 kB	48 kB	48 kB	48 kB
Cache L2	1MB	1MB	1MB	1MB
Cache L3	8 MB			

TABLE 1 – Topologie de la machine de tests

Environnement Logiciel

Dans cette partie sera détaillée la partie logiciel de la machine de test, les fichiers scripts pour changer entre machine allégée et machine classique sont fournis dans l'archive.

Distribution : Fedora v40 WorkStation
Compilateur utilisé : `gcc` version 14.2.1 20240912 avec option `-O0`
Outil de mesure des temps d'exécution : fonction `gettimeofday` de C de la librairie `sys/time.h`.

Configuration expérimentale

Processus en activité

```
> ps -x
PID TTY      STAT   TIME COMMAND
```

```
1263 ?        Ss     0:00 /usr/lib/systemd/systemd --user
1265 ?        S      0:00 (sd-pam)
1284 tty1     Ss     0:00 -zsh
1898 tty1     R+     0:00 ps -x
```

Voici ce que font les processus en cours d'exécution.

systemd : C'est un système d'initialisation et de gestion de services sous Linux, conçu pour démarrer, arrêter et superviser les processus, tout en offrant une gestion centralisée des sessions et des ressources.
sd-pam : *systemd-PAM* C'est un utilitaire de `systemd` pour gérer les sessions.
zsh : C'est le shell qui est lancé pour exécuter les commandes.
ps -x : C'est la commande qui a été lancée pour voir les processus.

Méthodologie de récolte des données expérimentales

Pour la premiere partie, après avoir mesuré les differents temps moyen d'accès, le programme C l'affiche sur la sortie standard afin d'exécuter la commande `./benchmark > bench.csv` pour rassembler toutes les sorties dans un fichier. De plus, le programme affiche les données sous format `csv` pour permettre un meilleur traitement avec le logiciel `gnuplot`.
Pour la deuxieme partie, le programme affiche simplement le résultat sur la sortie standard, il nous reste donc juste a récupérer le résultat avec un `./benchmark > res.data`, exécuté 4 fois ou plus avec une boucle `for`.
Pour la dernière partie, l'outil utilisé se charge lui-même de mettre les résultats dans des fichiers `.data`, que l'on pourra plot avec les fichiers `.gp` et `gnuplot`.

2 Tester les performances de la hiérarchie mémoire d'un CPU avec un micro-benchmark séquentiel

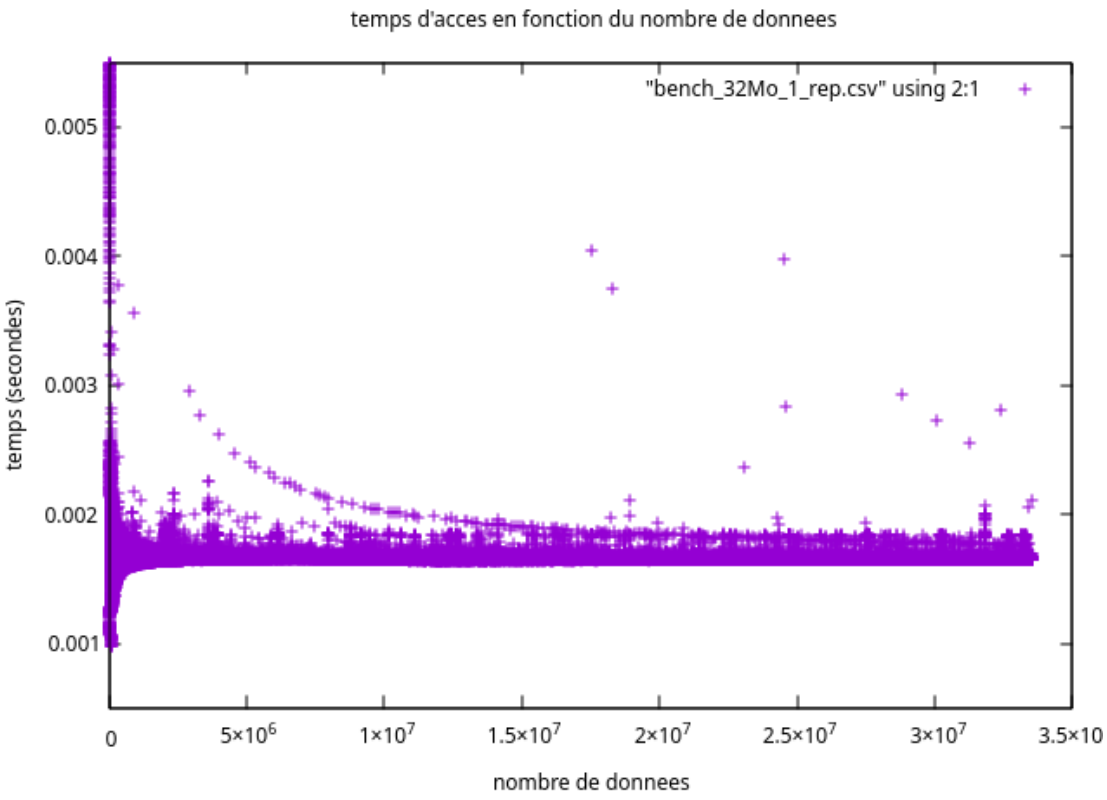

FIGURE 1 – Résultats du benchmark avec un tableau de taille 32Mo et une répétition par accès.

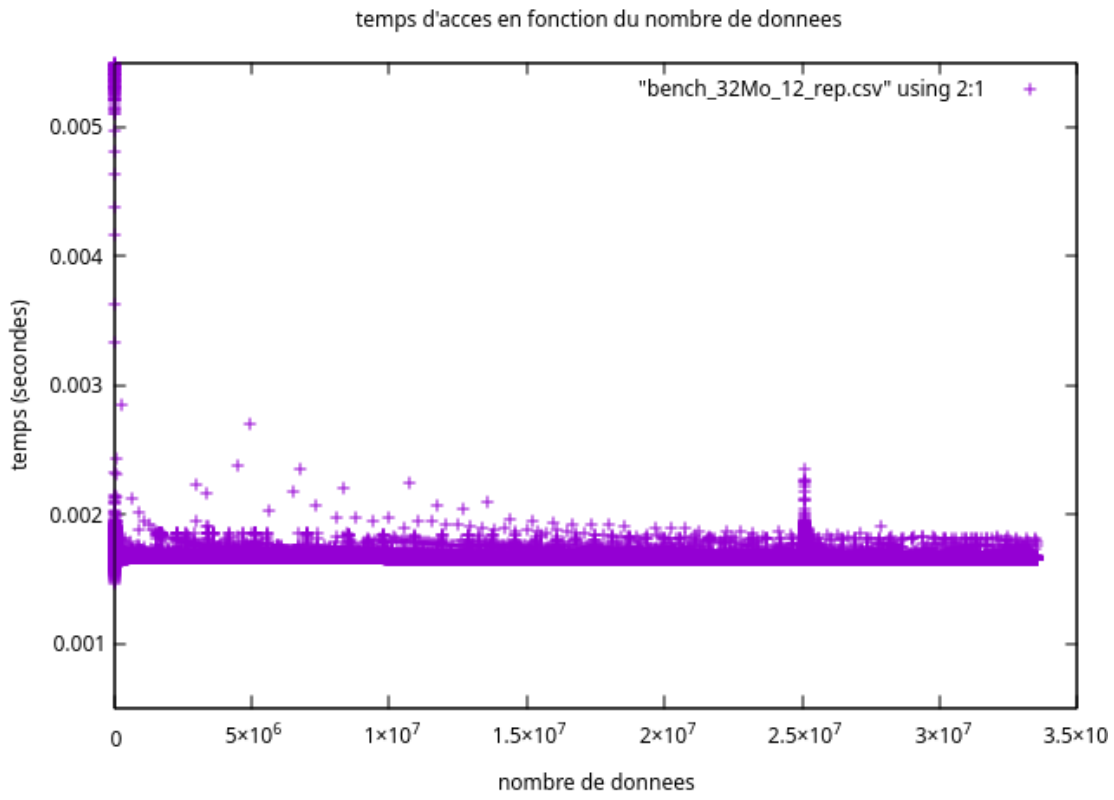

FIGURE 2 – Résultats du benchmark avec un tableau de taille 32Mo et 12 répétitions par accès.

Ce que l'on remarque sur ces graphiques c'est qu'aucun "palier" n'apparaît, cela pourraît être dû au prefetch matériel, on remarque même que les temps d'accès dimininuent avec le temps. On observe aussi qu'avec 12 répétitions, les temps d'accès sont plus stables.

3 Evaluation de la bande passante entre le processeur et la mémoire centrale.

Dans cette partie sera expliqué comment à été évaluée la bande passante entre le processeur et la mémoire centrale, une brève analyse des résultats sera aussi proposée.

3.1 Méthode

Pour calculer la bande passante entre le processeur et la mémoire centrale, il est d'abord nécessaire de calculer le temps moyen d'accès a la mémoire, et de connaître un peu notre machine. Les données qui sont importantes a connaître sont, la taille du cache L3 ainsi que la taille de la ligne de cache. Ils vont servir a déterminer la taille du tableau ainsi que la taille du pas pour éviter qu'une donnée soit préchargée en mémoire, on doit donc prendre 2 fois la taille du cache L3 et un pas suffisament grand pour être sûr de dépasser le prefetch matériel ainsi que le préchargement des données dans un cache.
La taille du cache L3 de la machine de test est de 8Mo, on a donc $(2*8*1024*1024)/sizeof(int)$, ce qui donne 4194304 élements, quand à la taille du pas, on choisira un pas très grand, 4 fois et 8 fois la taille de la ligne de cache, réalisés avec des `bit shift` pour s'assurer qu'aucune donnée ne soit préchargée, on remarquera que plus on augmente le pas, plus le temps d'accès moyen à la mémoire augmente.

3.2 Résultats

Multiplicateur de pas	Temps d'acces moyen (en ns)	Bande passante (en Mo/s)
4	62,49237075	985,63519275
8	116,783142	563,27429225

TABLE 2 – Résultats moyens des tests de performance mémoire sur 4 exécutions du benchmark.

De plus, si on regarde les résultats inclus dans l'archive, on remarque que plus on effectue d'exécution consécutives, plus le temps d'accès a la mémoire diminue et le débit augmente, cela pourrait venir du système d'exploitation qui optimiserait par lui-même, ou en augmentant la vitesse du processeur.

4 Etude de l'outil Calibrator

Les résultats du benchmark réalisé avec `Calibrator` sont laissés dans l'archive, l'objectif ici est de comprendre pourquoi cet outil est mieux que les benchmarks réalisés précédemment.
Si on s'en tient a la documentation du logiciel, `Calibrator` permet de mesurer les différents niveaux de cache, avec pour chaque cache, sa taille, sa taille de ligne de cache, on remarque d'ailleurs sur les résultats que la taille augmente selon le niveau de cache, c'est sans doute une des raisons pour lesquelles on n'observait pas de "palier" au premier benchmark, et aussi la latence d'accès ou d'échec au niveau de cache.
L'outil permet aussi de mesurer les différents niveaux du `TLB`, qui est un cache mémoire qui permet de stocker les dernières traductions adresse virtuelle/adresse physique, évitant ainsi d'avoir à les chercher dans la mémoire centrale. Ce qui est une très grande différence avec les benchmarks réalisés plus tôt, qui nous permettaient de calculer uniquement les temps d'accès aux niveaux de cache ainsi que la bande passante entre le processeur et la mémoire centrale. L'analyse des niveaux de TLB viennent avec la mesure de sa capacité, de la taille d'une page mémoire, ainsi que la latence d'échec.
L'outil peut aussi mesurer la latence d'accès à la mémoire centrale, en additionnant simplement les *latences d'échec* de tous les niveaux de cache. Les différences avec les précédents benchmarks ne s'arrête pas là, il y a aussi la façon dont l'outil réalise ses benchmarks, il change plusieurs fois pendant le benchmark la taille du pas, appelée `stride` en anglais, tandis que sur nos benchmarks, la taille du pas était fixe.

5 Conclusion

Dans ce travail pratique, les performances de la hiérarchie mémoire d'un processeur ont été étudiées à travers plusieurs tests et mesures. En regardant les temps d'accès moyen et la bande passante, on a pu voir comment des paramètres comme la taille des tableaux et le pas de lecture influencent les résultats.
Plusieurs détails intéressants en sont sortis, comme premièrement l'absence de "paliers", probablement à cause du prefetch matériel, et une amélioration des performances après plusieurs exécutions, sûrement grâce à des optimisations du système.
Ces tests montrent à quel point il est important de bien comprendre le fonctionnement des caches et de la mémoire pour optimiser les performances. Cela ouvre aussi la porte à des études plus poussées, sur les effets des réglages du système ou sur les optimisations matérielles par exemple.