Architecture des microprocesseurs (ECE_4ES01_TA)

TD/TP5 : Analyse de performances de multicoeurs

Microprocesseur multicoeurs

Auteurs (quadrinôme)

CHACÓN GÓMEZ José Daniel

DE MACENA BARRETO Prénom 2

EMMANUEL DA COSTA Lucas

ZHE Chen

Encadrant / Chargé de TD : SIDEM Antoine

Responsable de la matière : HAMMAMI Omar

École Nationale des Techniques Avancées (ENSTA Paris)

Février 2026

Table des matières

1 Introduction

2 Exercice 3 — A completer

2.1 Questions

2.1.1 Question 1

> **Énoncé (Q1).** n considérant que chaque thread s'exécute sur un processeur dans une architecture de type multicoeurs à base de bus et 1 niveau de cache (comme décrit Figure 21), décrivez le comportement de la hiérarchie mémoire et de la cohérence des caches pour l'algorithme de multiplication de matrices. On supposera que le thread principal se trouve sur le processeur d'indice 1.

```
// (...)

  #pragma omp parallel for
  for(int x = 0; x < size; x++) {
    for (int y = 0; y < size; y++) {
      int64_t tot;
      for (int m = 0; m < size; m++) {
        tot += A[x*size + m]*B[m*size + y];
      }
      C[x*size + y] = tot;
    }
  }

// (...)
```

Listing 1 – Utilisation de openMP pour multiplication matricial

Dans l'algorithme de multiplication de matrices présenté dans le Listing 1, chaque thread s'exécute sur un processeur dans une architecture multicoeurs. Chaque thread accède à la mémoire pour lire les éléments des matrices A et B, et pour écrire les résultats dans la matrice C. La hiérarchie mémoire dans cette architecture multicoeurs comprend un niveau de cache partagé entre les processeurs. Lorsque les threads accèdent aux éléments des matrices, ils peuvent bénéficier de la localité de références, ce qui signifie que les données récemment utilisées sont susceptibles d'être réutilisées prochainement. Cependant, si plusieurs threads accèdent simultanément à des éléments de matrice qui ne sont pas contigus en mémoire (par exemple, en accédant à des éléments de la matrice B), cela peut entraîner des cache misses, ce qui ralentit les performances, sachant que les access à la matrice B sont dans l'ordre de la colonne, ce qui n'est pas optimal pour la localité de référence.

2.1.2 Question 2 - Comparaison de perrformances

> **Énoncé (Q2).** Examinez le fichier de déclaration d'un élément de type « processeur superscalaire out-of-order », et présentez sous forme de tableau cinq paramètres configurables

Paramètre	Description	Valeur par défaut
Nº d'instr. que peut récupérer par cycle	fetchWidth	4
Nº d'instr. que peut décoder par cycle	decodeWidth	4
Nº d'instr. que peut émettre par cycle	issueWidth	4
Nº d'instr. que peut valider par cycle	commitWidth	4
Nº d'entrées dans la Reorder Buffer (ROB)	numROBEntries	192

TABLE 1 – Paramètres configurables d'un processeur superscalaire out-of-order

de ce type de processeur avec leur valeur par défaut. Choisissez de préférence des paramètres étudiés lors des séances TD/TP précédentes. Le fichier à consulter est le suivant : $GEM5/src/cpu/o3/O3CPU.py

2.1.3 Question 3 - Localité de références

Énoncé (Q3). Examinez le fichier d'options de la plateforme se.py, puis déterminez et présentez sous forme de tableau les valeurs par défaut des paramètres suivants :

— Cache de données de niveau 1 : associativité, taille du cache, taille de la ligne
— Cache d'instructions de niveau 1 : associativité, taille du cache, taille de la ligne
— Cache unifié de niveau 2 : associativité, taille du cache, taille de la ligne

Le fichier **commons.py** est presenté dans le Code 2. Ces parametres sont définis dans la section "Cache Options" du fichier d'options de la plateforme se.py. En examinant ce fichier, on peut identifier les options suivantes qui correspondent aux paramètres demandés :

```
    # (...)
    # Cache Options
    parser.add_option("--external-memory-system", type="string",
                      help="use external ports of this port_type for caches")
    parser.add_option("--caches", action="store_true")
    parser.add_option("--l2cache", action="store_true")
    parser.add_option("--fastmem", action="store_true")
    parser.add_option("--num-dirs", type="int", default=1)
    parser.add_option("--num-l2caches", type="int", default=1)
    parser.add_option("--num-l3caches", type="int", default=1)
    parser.add_option("--l1d_size", type="string", default="64kB")
    parser.add_option("--l1i_size", type="string", default="32kB")
    parser.add_option("--l2_size", type="string", default="2MB")
    parser.add_option("--l3_size", type="string", default="16MB")
    parser.add_option("--l1d_assoc", type="int", default=2)
    parser.add_option("--l1i_assoc", type="int", default=2)
    parser.add_option("--l2_assoc", type="int", default=8)
    parser.add_option("--l3_assoc", type="int", default=16)
    parser.add_option("--cacheline_size", type="int", default=64)
    # (...)
```

Listing 2 – Extrait du fichier d'options de la plateforme se.py

Dans l'extrait du fichier d'options, on trouve les paramètres suivants avec leurs valeurs par défaut sont définis comme suit dans le Tableau 2 :

Cache	Associativité	Taille du cache	Taille de ligne
L1 Data (L1D)	2-way	64 kB	64 B
L1 Instruction (L1I)	2-way	32 kB	64 B
L2 Unifié	8-way	2 MB	64 B

TABLE 2 – Paramètres par défaut des caches

2.1.4 Question 4 - Localité de références

Question 4 : Déterminez quel est le processeur exécutant toujours le plus grand nombre de cycles. Expliquez pourquoi. expliquez également pourquoi l'analyse du nombre de cycles sur ce processeur revient à analyser le nombre total de cycles d'exécution de l'application.