

UNIVERSITÉ DE BRETAGNE OCCIDENTALE

MASTER 2 INFORMATIQUE  
DÉPARTEMENT INFORMATIQUE

2020/2021

COMPILATION & PARALLÉLISME

---

## Devoir de TP OpenMP

---

*Auteur :*  
William PENSEC

5 février 2021



# Sommaire

I	Analyse des programmes fournis . . . . .	2
I.1	Question 1 . . . . .	2
I.2	Question 2 . . . . .	2
I.3	Question 3 . . . . .	3
II	Expérimentations . . . . .	3
II.1	Hypothèse 1 . . . . .	3
II.2	Hypothèse 2 . . . . .	3
II.3	Hypothèse 3 . . . . .	4
III	Conception d'une meilleure version parallèle . . . . .	4
III.1	Question 1 . . . . .	4
III.2	Question 2 . . . . .	5
III.3	Question 3 . . . . .	6
	III.3.1 Hypothèse 1 . . . . .	6
	III.3.2 Hypothèse 2 . . . . .	6
	III.3.3 Hypothèse 3 . . . . .	6
	III.3.4 Résultats . . . . .	6

## I Analyse des programmes fournis

Les trois hypothèses dont on parlera dans cette partie sont :

	Pas	ITA	ITB	ITC	ITD	ITE
Hypothèse 1	200	8	8	7	2	1
Hypothèse 2	200	8	4	3	7	1
Hypothèse 3	200	2	5	4	5	1

### I.1 Question 1

Le temps théorique de la version séquentielle peut se calculer grâce à la formule :

$$T_{seq} = (ITA * PAS)^3 + (ITB * PAS)^3 + (ITC * PAS)^3 + (ITD * PAS)^3 + (ITE * PAS)^3$$

	$T_{seq}$ théorique
Hypothèse 1	11 008 000 000
Hypothèse 2	7 576 000 000
Hypothèse 3	2 584 000 000

### I.2 Question 2

Le temps théorique de la version parallèle  $V_1$  peut se calculer grâce à la formule :

$$T_{V_1} = (MAX(ITA; ITB; ITC) * PAS)^3 + (ITD * PAS)^3 + (ITE * PAS)^3$$

Cette version utilise donc 3 coeurs au niveau de la parallélisation de ITA, ITB, ITC.

	$T_{V_1}$ théorique	Accélération	Efficacité
Hypothèse 1	4 168 000 000	2.64	0.88
Hypothèse 2	6 848 000 000	1.11	0.37
Hypothèse 3	2 008 000 000	1.29	0.43

### I.3 Question 3

Le temps théorique de la version parallèle  $V_2$  peut se calculer grâce à la formule :

$$T_{V_2} = (MAX(ITA; ITD) * PAS)^3 + (MAX(ITB; ITC) * PAS)^3 + (ITE * PAS)^3$$

Cette version utilise donc 2 coeurs au niveau de la parallélisation de ITA, ITD et ITB, ITC.

	$T_{V_2}$ théorique	Accélération	Efficacité
<b>Hypothèse 1</b>	8 200 000 000	1.34	0.67
<b>Hypothèse 2</b>	4 616 000 000	1.64	0.82
<b>Hypothèse 3</b>	2 008 000 000	1.29	0.64

## II Expérimentations

Comme on peut voir dans les résultats suivants, les résultats des parties théoriques et expérimentales sont approximativement équivalents en terme d'efficacité et d'accélération. Il y a toutefois des écarts plus ou moins grands mais qui sont en général dû à l'ordinateur et aux tâches qui s'exécutent en même temps car j'obtiens parfois un delta de 5s sur les résultats. Maximum il y a environ 9% d'écarts ce qui est très correct.

### II.1 Hypothèse 1

	Temps expérimental	Accélération	Efficacité
<b>Version séquentielle</b>	65.57s		
<b>Version <math>V_1</math></b>	27.25s	2.41	0.80
<b>Version <math>V_2</math></b>	49.22s	1.33	0.67

### II.2 Hypothèse 2

	Temps expérimental	Accélération	Efficacité
<b>Version séquentielle</b>	44.24s		
<b>Version <math>V_1</math></b>	40.47s	1.09	0.36
<b>Version <math>V_2</math></b>	27.21s	1.63	0.81

## II.3 Hypothèse 3

	Temps expérimental	Accélération	Efficacité
Version séquentielle	13.98s		
Version $V_1$	11.47s	1.22	0.41
Version $V_2$	10.98s	1.27	0.64

## III Conception d'une meilleure version parallèle

### III.1 Question 1

La mise en oeuvre de cette nouvelle version est disponible dans le fichier de code source du TP.

Le code de cette nouvelle version parallèle se décline par :

---

```

1  #pragma omp parallel sections private(i, j, k)
2  {
3  #pragma omp section
4  {
5      //traitement A
6      for (i = 0; i < PAS * ITA; i++){
7          for (j = 0; j < PAS * ITA; j++){
8              for (k = 0; k < PAS * ITA; k++){
9                  A[i][j] = A[i][j] + M1[i][k] * M2[k][j];
10             }
11         }
12     }
13 }
14 #pragma omp section
15 {
16 #pragma omp parallel sections private(i, j, k)
17 {
18 #pragma omp section
19 {
20     //traitement C
21     for (i = 0; i < PAS * ITC; i++){
22         for (j = 0; j < PAS * ITC; j++){
23             for (k = 0; k < PAS * ITC; k++){
24                 C[i][j] = C[i][j] + M5[i][k] * M6[k][j];
25             }
26         }
27     }
28 }
29 }
30 #pragma omp section

```

```

31     {
32         //traitement B
33         for (i = 0; i < PAS * ITB; i++){
34             for (j = 0; j < PAS * ITB; j++){
35                 for (k = 0; k < PAS * ITB; k++){
36                     B[i][j] = B[i][j] + M3[i][k] * M4[k][j];
37                 }
38             }
39         }
40     }
41 }
42
43 //traitement D
44 for (i = 0; i < PAS * ITD; i++){
45     for (j = 0; j < PAS * ITD; j++){
46         for (k = 0; k < PAS * ITD; k++){
47             D[i][j] = D[i][j] + B[i][k] * C[k][j];
48         }
49     }
50 }
51 }
52 }
53
54 //traitement E
55 for (i = 0; i < PAS * ITE; i++){
56     for (j = 0; j < PAS * ITE; j++){
57         for (k = 0; k < PAS * ITE; k++){
58             E[i][j] = E[i][j] + A[i][k] * D[k][j];
59         }
60     }
61 }
62

```

---

### III.2 Question 2

Le temps théorique de la version parallèle  $V_3$  peut se calculer grâce à la formule :

$$T_{V_3} = \text{MAX}((ITA * PAS)^3; (\text{MAX}(ITB; ITC) * PAS)^3 + ((ITD * PAS)^3)) + (ITE * PAS)^3$$

Cette version utilise donc 3 coeurs au niveau de la parallélisation de ITA, ITB, ITC.

	$T_{V_3}$ théorique	Accélération	Efficacité
Hypothèse 1	4 168 000 000	2.64	0.88
Hypothèse 2	4 104 000 000	1.85	0.62
Hypothèse 3	2 008 000 000	1.29	0.43

### III.3 Question 3

#### III.3.1 Hypothèse 1

	Temps expérimental	Accélération	Efficacité
Version séquentielle	65.57s		
Version $V_1$	27.25s	2.41	0.80
Version $V_2$	49.22s	1.33	0.67
Version $V_3$	26.20s	2.50	0.83

#### III.3.2 Hypothèse 2

	Temps expérimental	Accélération	Efficacité
Version séquentielle	44.24s		
Version $V_1$	40.47s	1.09	0.36
Version $V_2$	27.21s	1.63	0.81
Version $V_3$	25.17s	1.76	0.59

#### III.3.3 Hypothèse 3

	Temps expérimental	Accélération	Efficacité
Version séquentielle	13.98s		
Version $V_1$	11.47s	1.22	0.41
Version $V_2$	10.98s	1.27	0.64
Version $V_3$	10.94s	1.27	0.42

#### III.3.4 Résultats

Comme on peut voir sur les résultats précédents, la version 3 est plus efficace ou au moins aussi efficace que les autres versions parallèles. En effet, on obtient des valeurs assez proche des meilleures solutions de  $V_1$  et  $V_2$  mais en étant dans une seule et même version  $V_3$ .