RAIRO. RECHERCHE OPÉRATIONNELLE

M. MARRAKCHI

Parallélisation de l'algorithme du chemin critique pour une machine à mémoire partagée

RAIRO. Recherche opérationnelle, tome 31, n° 4 (1997), p. 429-440

<http://www.numdam.org/item?id=RO_1997__31_4_429_0>

Recherche opérationnelle/Operations Research
(vol. 31, n° 4, 1997, pp. 429 à 440)

PARALLÉLISATION DE L'ALGORITHME DU CHEMIN CRITIQUE POUR UNE MACHINE À MÉMOIRE PARTAGÉE (*)

par M. MARRAKCHI ([1])

Communiqué par Jacques CARLIER

Résumé. – *Dans ce papier, nous considérons la parallélisation, sur une machine de type MIMD à mémoire partagée (Alliant FX/80), de l'algorithme du chemin critique appliqué au graphe 2-pas à tâches de durées égales. Ce type de graphe est obtenu notamment en décomposant l'algorithme séquentiel résolvant un système triangulaire linéaire par bloc. Nous présentons le temps parallèle y compris le temps des accès à la mémoire partagée de l'algorithme parallèle et nous déterminons théoriquement et expérimentalement la valeur optimale de la taille du bloc qui minimise le temps total d'exécution.*

Mots clés : Algorithmes parallèles, Alliant FX/80, résolution de système triangulaire, bloc optimal.

Abstract. – *In this paper, we consider parallelisation on a parallel computer with shared memory of the critical path algorithm for 2-steps graph with constant task cost. This graph occures in the parallelisation of block triangular linear system resolution. We present the parallel execution time including the communication cost of the critical path algorithm and we theoretically and practically determine the optimal value of the block size which minimises the parallel execution time.*

Keywords: Parallel algorithms, Alliant FX/80, triangular linear system resolution, optimal block.

1. INTRODUCTION

La parallélisation de l'algorithme résolvant un système triangulaire a été étudiée par plusieurs auteurs [4, 8, 9]. Dans ce papier, nous présentons la parallélisation de l'algorithme du chemin critique pour une machine parallèle à mémoire partagée. Nous présentons la décomposition de l'algorithme séquentiel résolvant un système linéaire triangulaire par bloc et nous montrons à l'aide d'un exemple que le graphe de précédence est le graphe 2-pas à tâches de durées égales. Ensuite, nous rappelons l'algorithme du chemin critique pour un problème de taille n et un nombre de processeurs égal à p où $1 \leq p \leq (n+2)/4$ [8]. Nous décrivons la parallélisation de

(*) Reçu en juin 1995.

([1]) Département Informatique, Faculté des Sciences, 3038 Sfax, Tunisie.

Recherche opérationnelle/Operations Research, 0399-0559/97/04/$ 7.00

l'algorithme du chemin critique pour une machine parallèle à mémoire partagée. Enfin, nous évaluons pour chaque tâche le temps théorique des accès à la mémoire partagée et nous calculons, théoriquement et expérimentalement la taille optimale du bloc r_{opt} qui minimise le temps d'exécution parallèle de l'algorithme du chemin critique (la matrice étant découpée en blocs $r \times r$). Nous montrons la bonne adéquation des résultats expérimentaux et théoriques. Le modèle de calcul parallèle utilisé est le modèle CREW-PRAM comprenant p processeurs identiques notés P_1, $P_2, ..., P_p$ et reliés à la mémoire, partagée en bancs, par l'intermédiaire d'un réseau d'interconnexion [4]. Dans ce modèle, un processeur ne peut lire-écrire et faire des calculs en même temps. Autrement dit, il n'y a pas de recouvrement des accès à la mémoire par des calculs. Comme exemple de machine parallèle nous avons utilisé un Alliant FX/80. Elle possède 8 processeurs vectoriels. Ces processeurs peuvent accéder à la mémoire centrale, organisée en quatre quadrants, au travers d'un cache partagé, lui-même organisé en quatre bancs. Les processeurs accèdent au cache via un réseau crossbar. Le cache est connecté à la mémoire partagée par un bus. Si on appelle W, X, Y, Z les quadrants du cache, les éléments successifs d'un vecteur sont distribués dans le cache suivant le schéma cyclique $W\,X\,X\,W\,Y\,Z\,Z\,Y$ [5].

2. PRÉSENTATION DU GRAPHE 2-PAS

Considérons le système $A\,x = b$, où $A = (a_{ij})\ 1 \leq i, j \leq N$ une matrice carrée triangulaire inférieure de taille N non singulière et x, b deux vecteurs de taille N. On suppose que $N = nr$ où r est la taille des blocs choisie *a priori* et on découpe cette matrice en blocs $r \times r$. La décomposition en tâches est la suivante :

Pour $i := 1$ **à** n **Faire**
  Exécuter $T^r_{i,i}$:< **Pour** $k := (i-1)\,r + 1$ **à** ir **Faire**
    Pour $m := (i-1)\,r + 1$ **à** $k - 1$ **Faire**
      $x_k := x_k - a_{k,m}\,x_m$
    FinPour
    $x_k := \frac{x_k}{a_{kk}}$
  FinPour>
  Pour $j := i + 1$ **à** n **Faire**
    Exécuter $T^r_{j,i}$:< **Pour** $k := (j-1)\,r + 1$ **à** jr **Faire**
      Pour $m := (i-1)\,r + 1$ **à** ir **Faire**
        $x_k := x_k - a_{k,m}\,x_m$
      FinPour
    FinPour>.

Nous avons alors :

$$T^r_{i,i} = \bigcup_{(i-1)\,r+1 \le k \le ir} \Big(\Big(\bigcup_{(i-1)\,r+1 \le m \le k-1} T^1_{k,m} \Big) \bigcup T^1_{k,k} \Big)$$

$$T^r_{j,i} = \bigcup_{(j-1)\,r+1 \le k \le jr} \Big(\bigcup_{(i-1)\,r+1 \le m \le ir} T^1_{k,m} \Big).$$

La figure 1 illustre le graphe d'ordonnancement pour $n = 6$. C'est un graphe 2-pas de taille 6. Un graphe 2-pas de taille n est un graphe comportant $2\,n-1$ niveaux numérotés de haut en bas. Le niveau de numéro impair $2\,i-1$ $(i = 1, ..., n)$ est constitué d'une seule tâche $T^r_{i,i}$, celui de numéro pair $2\,i\,(i = 1, 2, ..., n-1)$ est constitué des tâches $T^r_{j,i}$ pour $j = i+1, i+2, ..., n$. Les tâches sont reliées par les contraintes d'ordonnancement suivantes :

$$\text{(A)} \quad T^r_{i,i} \ll T^r_{j,i}, \qquad i+1 \le j \le n, \qquad 1 \le i \le n-1$$

$$\text{(B)} \quad T^r_{j,i} \ll T^r_{j,i+1}, \qquad 2 \le j \le n, \qquad 1 \le i \le j-1$$

$T \ll T'$ signifie que l'exécution de la tâche T doit être terminée avant que l'exécution de la tâche T' ne puisse commencer [2, 10]. C'est cette structure particulière qui est à l'origine de l'appellation « 2-pas ». $T^r_{i,i}$ correspond à la résolution d'un système triangulaire de taille r. $T^r_{j,i}\,(i < j)$ correspond quant à elle à l'élimination de r inconnues (déjà déterminées) dans les r équations associées (*fig.* 2).

Exemples :

1. Pour $r = 1$, une tâche $T^1_{i,i}$, correspond à :

$$\left\langle x_i := \frac{x_i}{a_{i,i}} \right\rangle$$

et une tâche $T^1_{j,i}$, correspond à :

$$\langle x_j := x_j - a_{j,i}\,x_i \rangle.$$

2. Pour $r = 2$, une tâche $T^2_{i,i}$, correspond à :

$$\left\langle x_{2\,i-1} := \frac{x_{2\,i-1}}{a_{2\,i-1,2\,i-1}} \right\rangle$$

$$\langle x_{2\,i} := x_{2\,i} - a_{2\,i,2\,i-1}\,x_{2\,i-1} \rangle$$

$$\left\langle x_{2\,i} := \frac{x_{2\,i}}{a_{2\,i,2\,i}} \right\rangle,$$

et une tâche $T^2_{j,i}$, correspond à :

$\langle x_{2j-1} := x_{2j-1} - a_{2j-1,2i-1}\, x_{2i-1} \rangle$ $\quad$ $\langle x_{2j-1} := x_{2j-1} - a_{2j-1,2i}\, x_{2i} \rangle$

$\langle x_{2j} := x_{2j} - a_{2j,2i-1}\, x_{2i-1} \rangle$ $\quad$ $\langle x_{2j} := x_{2j} - a_{2j,2i}\, x_{2i} \rangle$.

Pour la clarté du texte, nous notons dans la suite $T^r_{j,i}\,(i \geq j)$ par $T_{j,i}\,(i \geq j)$ sauf précision. Une unité de temps arithmétique τ_a est définie comme le temps mis pour effectuer soit une multiplication suivie d'une soustraction soit une division [6, 9]. Le temps arithmétique mis pour exécuter une tâche $T_{i,i}$ (resp. $T_{j,i}$) est $\mathrm{Ex}_a\,(T_{i,i}) = (r + r^2)\,\tau_a/2$ (resp. $\mathrm{Ex}_a\,(T_{j,i}) = r^2\,\tau_a$).

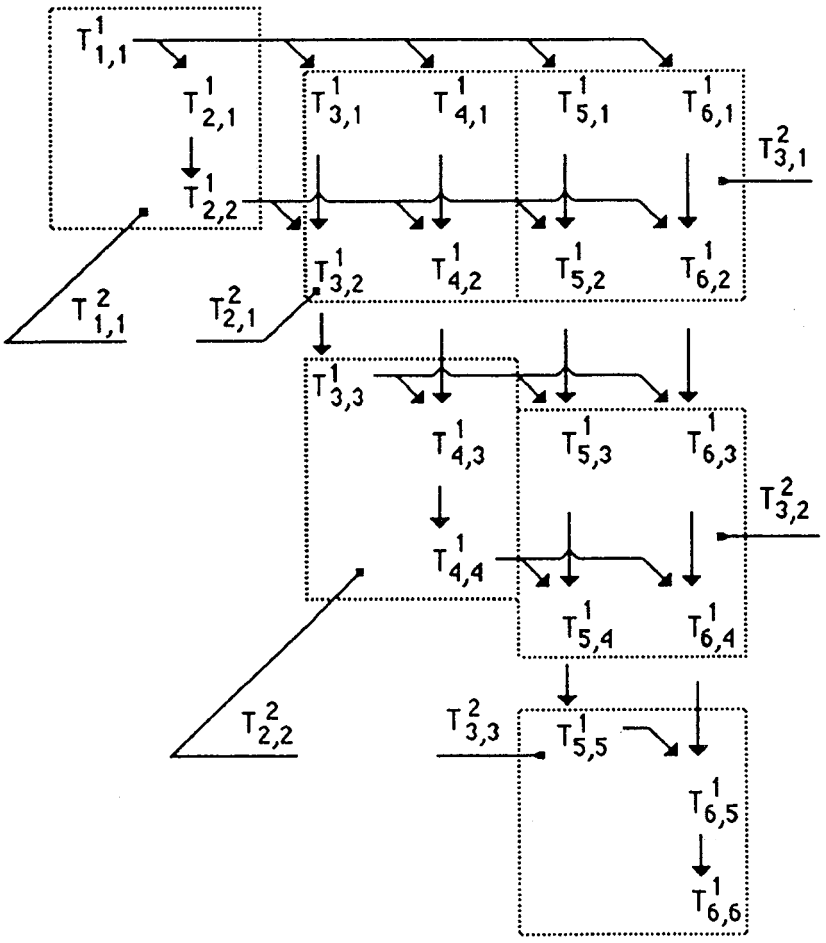

Figure 1. – Graphe 2-pas pour $n = 6$, $r = 1$ et représentation des tâches pour $n = 3$ et $r = 2$.

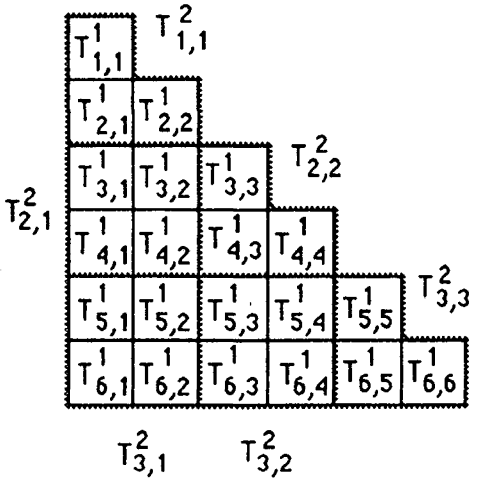

Figure 2. – Correspondance tâches sous-matrices.

Notons τ_c le coût d'une lecture (resp. écriture) d'une donnée de la mémoire (resp. en mémoire) [4]. Alors le temps d'accès à la mémoire d'une tâche $T_{j,i}$, $j > i$ (resp. $T_{i,i}$) est $\mathrm{Ex}_c(T_{j,i}) = (r^2 + 3r)\tau_c$ (resp. $\mathrm{Ex}_c(T_{i,i}) = (r^2 + 5r)\tau_c/2$). Comme le nombre de tâches $T_{j,i}$, $j > i$ est en $O(n^2/r^2)$ et le nombre de tâches $T_{i,i}$ est en $O(n/r)$, nous supposons que toutes les tâches ont le même coût arithmétique (resp. des accès à la mémoire) qui est égal à $r^2\,\tau_a$ (resp. $(r^2+3r)\,\tau_c$). Le coût total d'une tâche est égal à $\mathrm{Ex}(T_{j,i}) = \mathrm{Ex}_a(T_{j,i}) + \mathrm{Ex}_c(T_{j,i}) = r^2\,\tau_a + (r^2 + 3r)\,\tau_c$. Le temps séquentiel de l'algorithme est alors $\frac{nr(nr+1)}{2}(\tau_a + 4\tau_c)$. Nous rappelons que l'algorithme du chemin critique exécute parmi les tâches indépendantes (qui sont libérées des contraintes d'ordonnancement) d'abord celles qui ont le plus long chemin (la longueur du chemin d'une tâche T est définie comme la distance, *i.e.* nombre de tâches multiplié par le coût d'une tâche qui est constant, séparant la tâche T de la tâche terminale $T_{n,n}$) [8]. Ajoutons que l'algorithme du chemin critique appliqué au graphe 2-pas, à tâches de durées égales, exécute ce graphe en deux phases [8]. L'exécution de la phase (1) commence à l'instant $t = 0$ et se termine à l'instant $t = 4p - 1$. L'exécution de la phase (2) a lieu juste après la fin de l'exécution de la phase (1). Le temps de l'algorithme du chemin critique est :

$$T_p = (\lfloor((n-1)(n+2) - 2p^2 - 2)/2p\rfloor + 2p + 1)(r^2\,\tau_a + (r^2 + 3r)\,\tau_c).$$

L'algorithme du chemin critique est détaillé dans la figure 3 pour $n = 10$ et $p = 3$.

Comme dans [7], on note $L(T_{k,j}) = \{T_{k,j}; T_{k+1,j}; T_{k+2,j}; ...; T_{n,j}\}$, $L(T_{k,j}, T_{i,j}) = \{T_{k,j}; T_{k+1,j}; T_{i,j}\}$, où $i \geq k$. Si $i < k$, $L(T_{k,j}, T_{i,j})$ est vide. $D(T_{k,j}) = \{T_{k,j}; T_{k+1,j-1}; ...\}$, c'est l'ensemble de toutes les tâches $T_{a,b}$ telles que $a + b = k + j$ et $b \leq j$.

3. IMPLANTATION

Dans ce qui suit, nous présentons l'implantation de l'algorithme du chemin critique sur un Alliant FX/80 et des résultats expérimentaux et théoriques.

Implantation de la phase (1)
Exécuter $T_{1,1}$
Pour $i = 1$ **à** $2p - 1$ **Faire** /Boucle séquentielle/
  Pour $j = i + 1$ **à** $i + p$ **Faire en Parallèle** /Boucle parallèle/
    Si $j = i + 1$ **Alors Exécuter** $T_{i+1,i}$; $T_{i+1,i+1}$
    Sinon Si $j \leq p + 1$ **Alors Exécuter** $T_{j,i}$; $T_{j+p-1,i}$
      Sinon Si $j \leq 2p$ **Alors Exécuter** $T_{j,i}$; $T_{j+p-1,p+i-j+1}$
        Sinon Exécuter $T_{j,2p-j+i}$; $T_{j+p-1,p-j+i+1}$

A chaque valeur de j, chaque processeur exécute successivement 2 tâches. On vérifie aisément que l'ensemble des tâches de l'algorithme est : $L(T_{i+1,i}; T_{2p,i})$, $D(T_{2p+1,i-1})$, $\{T_{i+1,i+1}\}$ $(1 \leq i \leq 2p - 1)$. C'est la partie du graphe 2-pas privé de la tâche $T_{1,1}$ et bordée inférieurement (au sens strict) par $D(T_{2p+1,2p-1})$.

Implantation de la phase (2)

Dans cette phase, les p processeurs exécutent les tâches du graphe bordé supérieurement (au sens large) par $D(T_{2p+1,2p-1})$, c'est-à-dire, l'ensemble des tâches $D(T_{2p+1,2p-1})$, $D(T_{i+1,i})$ et $D(T_{i+1,i+1})$ avec $2p \leq i \leq n-1$. Pour pouvoir implanter cette phase, on numérote toutes les tâches dans l'ordre d'exécution donné par la contrainte (C) suivante [8] :

$$T_{k,j} \leq\leq T_{k+1,j-1}$$

où la notation $T \leq\leq T'$ signifie que l'exécution de la tâche T' ne peut débuter avant celle de la tâche T (mais la simultanéité est possible). Plus précisément, les processeurs exécutent les tâches dans l'ordre suivant :

$$T_{2p+1,2p-1} \leq\leq T_{2p+2,2p-2} \leq\leq ... \leq\leq T_{2p+1,2p} \leq\leq T_{2p+2,2p-1}$$
$$\leq\leq ... \leq\leq T_{k,j} \leq\leq T_{k+1,j-1} \leq\leq ...$$

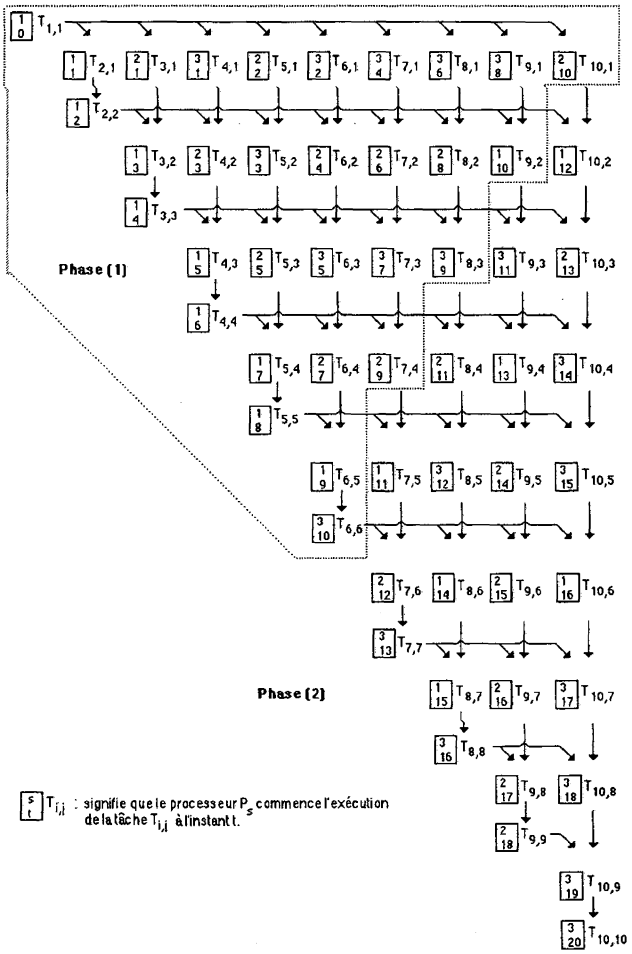

Figure 3. – Exécution du graphe 2-pas pour $n = 10$ et $p = 3$.

Pour celà, On rappelle que $N_{k,j}$ est le nombre de tâches exécutées avant $T_{k,j}$ (au sens strict) où $T_{k,j}$ est une tâche exécutée dans la phase (2) [4]. Nous posons $T_{k,j} \equiv \hat{T}_{v+1}$ où

1. $v = N_{k,j} = (i-2)(i+1)+k$ si $T_{k,j} \in D(T_{i+1,i})$ et $2p-1 \leq i \leq n/2$

2. $v = N_{k,j} = (i-1)(i+1) + k - 1$ si $T_{k,j} \in D(T_{i+1,i+1})$ et $2p - 1 \leq i \leq n/2$

3. $v = N_{k,j} = (n-1)(n+2)/2 - (n-i)^2 - n + k - 1$ si $T_{k,j} \in D(T_{i+1,i})$ et $i \geq n/2$

4. $v = N_{k,j} = (n-1)(n+2)/2 - (n-i)^2 - i - k - 1$ si $T_{k,j} \in D(T_{i+1,i+1})$ et $i \geq n/2$.

L'implantation de cette phase sera alors comme suit :

/Exécution de la partie du graphe située entre $D(T_{2p+1,2p-1})$ et $D(T_{n-p,n-p})$/
Pour $u = (4p-2)p$ **à** $(n-1)(n+2)/2 - p(p+1) - 1$, pas p **Faire**
/Boucle séquentielle de pas p, *i.e.* $u = (4p-2)p, (4p-1)p, 4p^2, .../$
  Pour $v = u+1$ **à** $u+p$ **Faire en Parallèle**
  /Boucle parallèle/
    Exécuter $\hat{T}_v$
/Synchronisation/
/Exécution de la partie du graphe située entre $D(T_{n-p+1,n-p})$ et $D(T_{n,n})$/
Pour $k = 1$ **à** $2p-1$, pas 2 **Faire**
/Boucle séquentielle de pas 2, *i.e.* $k = 1, 3, 5, .../$
  Pour $l = (k+1)/2$ **à** p **Faire en Parallèle**
  /Boucle parallèle/
    Exécuter $T_{n-p+l,n-p+k-l}$, $T_{n-p+l,n-p+k-l+1}$ /En séquentiel/

Pour mesurer les performances de l'algorithme, nous appelons efficacité expérimentale (resp. théorique) E le rapport entre le temps expérimental (resp. théorique) t_{seq} mis pour exécuter l'algorithme par un seul processeur et le produit du nombre de processeurs p par le temps expérimental (resp. théorique) t_p mis pour exécuter l'algorithme avec p processeurs ($E = t_{\text{seq}}/(p\,t_p)$). Pour $p = 8$ et $r = 1$, la figure 4 présente les courbes des efficacités expérimentale et théorique. La différence entre les efficacités

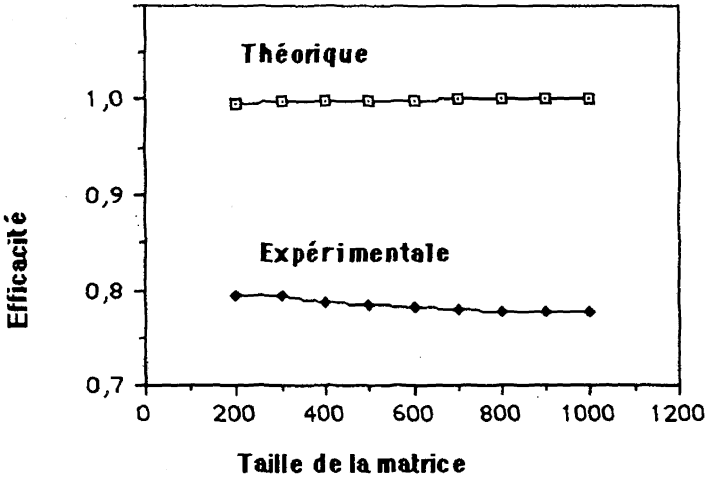

Figure 4. – Courbes des efficacités.

théorique et expérimentale est dûe au temps d'appel des tâches, l'utilisation de la boucle « Pour », la synchronisation et d'autres éléments du programme parallèle qui ne sont pas pris en compte dans la formule théorique.

4. DÉTERMINATION DE LA TAILLE OPTIMALE DU BLOC

Dans cette section, nous évaluons le temps des accès à la mémoire associé à une tâche pour pouvoir déterminer théoriquement la valeur optimale de la taille du bloc r_{opt} qui minimise le temps total d'exécution. Enfin, nous présentons les valeurs théorique et expérimentale de r_{opt}. Pour simplifier la présentation, nous posons $\tau_a = 1$. La valeur de τ_c, dépend fortement de l'architecture et plus précisément du réseau d'interconnexion reliant la mémoire partagée et les processeurs. En supposant que la traversée d'un commutateur prend un temps constant, nous déduisons que la valeur de τ_c est constante [4]. Dans ce cas, le coût total d'une tâche est égal à $\text{Ex}(T_{j,i}) = \text{Ex}_a(T_{j,i}) + \text{Ex}_c(T_{j,i}) = r^2 + (r^2 + 3r)\tau_c$. D'après le résultat présenté dans [7], le temps de l'exécution de l'algorithme du chemin critique est : $T_p(r) = (\lfloor((n-1)(n+2) - 2p^2 - 2)/2p\rfloor + 2p + 1)(r^2 + (r^2 + 3r)\tau_c)$ où $n = N/r$. En minimisant à la fois le coût des accès à la mémoire et le temps d'inactivité des processeurs nous obtenons un temps d'exécution minimum pour l'algorithme du chemin critique. En effet, si $r = 1$ c'est-à-dire que les blocs sont formés par des points, nous avons un temps des accès à la mémoire maximum et un temps d'inactivité des processeurs minimum. Si $r = n$, le coût des accès à la mémoire est minimum et le temps d'inactivité des processeurs est maximum. Entre ces deux valeurs il existe une valeur r_{opt} qui minimise le temps d'exécution de l'algorithme parallèle [1]. Le problème à résoudre consiste à déterminer la valeur r_{opt} pour laquelle le temps d'exécution est minimum. Nous avons donc :

$$\text{minimiser}(T_p(r) \qquad \text{tel que} \quad 1 \leq r \leq N).$$

Posons

$$F(r) = (((n-1)(n+2) - 2p^2 - 2)/2p + 2p + 1)(r^2 + (r^2 + 3r)\tau_c)$$

$$\text{où } n = N/r.$$

$$(\partial F/\partial r)(r) = [2(1+\tau_c)(2p^2 + 2p - 4)r^3 + ((1+\tau_c)N + 3\tau_c(2p^2 + 2p - 4))r^2 - 3\tau_c N^2]/2pr^2$$

TABLEAU I
Les valeurs théoriques de r_{opt}.

	$N = 1800$			$N = 3600$		
p	2	4	8	2	4	8
$r_{\text{opt}}\ (\tau_c = 0,5)$	37	29	21	54	44	32
$r_{\text{opt}}\ (\tau_c = 1)$	44	34	24	65	52	38
$r_{\text{opt}}\ (\tau_c = 2)$	49	38	27	74	58	42

En posant $a = 2p^2 + 2p - 4$ et $b = \tau_c/(1 + \tau_c)$ nous obtenons :

$$2pr^2\,(\partial F/\partial r)\,(r) = 2a\,(1 + \tau_c)\,G\,(r)$$

$$\text{où } G\,(r) = r^3 + ((3ab + N)/2a)\,r^2 - 3bN^2/2a$$

Il est aisé de prouver si $N > (1 + (1 + 12ab\,(2 + 3b))^{1/2})/6b$ que $G\,(1) < 0$ et $G\,(N) > 0$, autrement dit $G\,(1)\,G\,(N) < 0$. De plus $(\partial G/\partial r)\,(r)$ possède un signe constant sur $[1, N]$, donc G admet donc un seul minimum sur $[1, N]$. Par conséquent F admet un seul minimum sur $[1, N]$. C'est la valeur de r annulant G. $G(r) = 0$ entraîne que : $r^3 + ((3ab + N)/2a)\,r^2 - 3bN^2/2a = 0$.

En posant $u = (3ab + N)/6a$ et $v = bN^2/2a$, la racine y de l'équation $G\,(r) = 0$ située entre 1 et N s'écrit :

$$y = (2\cos(\theta/3) - 1)\,u \quad \text{où } \theta = \text{Arccos}\,(-1 + 3v/2u^3) \text{ si } -4u^3 + 3v \leq 0$$

$$y = (-u^3 + 3v/2 + ((-12u^3\,v + 9v^2)^{1/2})/2)^{1/3} + (-u^3 + 3v/2 - ((-12u^3\,v + 9v^2)^{1/2})/2)^{1/3} - u \text{ si } -4u^3 + 3v \geq 0.$$

On prendra pour r_{opt} l'une des deux valeurs $\lfloor y \rfloor$ ou $\lceil y \rceil$ minimisant $T_p\,(r)$. Pour avoir une idée claire sur r_{opt} théorique, il est nécessaire de fixer des valeurs de τ_c. Pour cela, nous choisissons trois valeurs de τ_c. Les deux premières valeurs sont $\tau_c = 0,5$ et 1, la troisième valeur est $\tau_c = 2$: valeur choisie par Cosnard *et al.* pour une machine à mémoire partagée [3]. Le tableau I illustre les valeurs théoriques de r_{opt} pour $N = 1800$ et 3600.

Pour Cosnard *et al.* [3], l'unité de temps est le temps mis pour exécuter une opération arithmétique ou pour lire un élément d'une matrice de la mémoire. Comme notre unité de temps (une multiplication suivie d'une soustraction) est deux fois plus grande que la leur alors $\tau_c = 2$ est la valeur correspondant au modèle qu'ils ont choisi pour une machine à mémoire partagée. Dans la pratique, ces auteurs ont précisé que la valeur $\tau_c = 2$ est la plus réaliste. Les

TABLEAU II
Temps parallèles pour différentes valeurs de r.

$N = 1\,800$						$N = 3\,600$					
r	$p = 2$	r	$p = 4$	r	$p = 8$	r	$p = 2$	r	$p = 4$	r	$p = 8$
30	2,9137	20	1,5981	20	0,9263	45	11,5421	30	6,3233	30	3,6783
60	2,8890	30	1,5779	25	0,9183	60	11,5017	50	6,2484	45	3,6457
90	2,9289	50	1,5779	30	0,9205	100	11,5080	90	6,2597	60	3,6511

TABLEAU III
Les valeurs expérimentales de r_{opt}.

	$N = 1\,800$			$N = 3\,600$		
p	2	4	8	2	4	8
r_{opt} expérimental	56	40	26	78	68	50

résultats des expériences faites sur un Alliant FX/80 sont présentés dans le tableau II (la valeur de r est donnée dans la colonne r et le temps parallèle est donné en secondes dans la colonne p).

Une interpolation quadratique a été utilisée pour ajuster les valeurs de r_{opt} expérimentales. Le tableau III illustre les valeurs expérimentales de r_{opt}.

Il est clair que r_{opt} théorique est inférieur à r_{opt} expérimental notamment lorsque p est assez petit. Cela est dû au fait que la formule théorique est calculée en comptant le nombre de pas de l'algorithme parallèle. Elle ne tient pas compte du temps utilisé par des éléments intervenant dans le programme parallèle. Ce temps augmente la valeur du temps parallèle expérimental. Ce qui entraîne que le temps expérimental d'exécution minimum est plus petit que celui trouvé. La valeur de r_{opt} dépend en fait de p et n. Elle croît avec n et décroît avec p. Si la valeur de p est assez grande, les valeurs théorique et expérimentale de r_{opt} sont très proches .Ceci est dû au fait que le coût des accès à la mémoire croît avec le nombre de processeurs.

5. CONCLUSION

Nous avons étudié les performances théorique et expérimentale de l'ordonnancement du chemin critique. Nous avons déterminé la valeur optimale r qui permet de minimiser le temps d'exécution du chemin critique.

Pour $\tau_c = 2$, nous signalons la bonne adéquation des résultats théorique et expérimental surtout si le coût des accès à la mémoire n'est pas élevé par exemple quand p est assez grand ($p = 8$) et le nombre des tâches du graphe est faible (n est assez petit). Dans le cas contraire, il n'est pas possible de prendre des mesures expérimentales exactes.

REMERCIEMENTS

Je remercie vivement le professeur Z. Mahjoub, les referees dont les remarques ont permis d'améliorer le contenu et la présentation de ce papier ainsi que le professeur I. Duff et ses collègues du CERFACS pour leur aide.

BIBLIOGRAPHIE

1. A. Gerasoulis et T. Yang, Efficient Algorithms and a Software Tool for Scheduling Parallel Computation, in Scheduling Theory and its Applications, P. Chretienne, E. G. Coffman Jr., J. K. Lenstra et Z. Liu, *John Wiley & Sons*, 1995, p. 111-143.
2. M. Cosnard, M. Marrakchi, Y. Robert et D. Trystram, Parallel Gaussian Elimination on an MIMD Computer, *Parallel Computing*, 1988, *6*, p. 275-296.
3. M. Cosnard, J. M. Muller, Y. Robert et D. Trystram, Communication Costs Versus Computation Costs in Parallel Gaussian Elimination, in Parallel Algorithms & Architectures, M. Cosnard, P. Quinton, Y. Robert et M. Tchuente, Proceedings of the International Workshop, Luminy, France, *North-Holland*, 1986, p. 19-29.
4. M. Cosnard et D. Trystram, Algorithmes et architectures parallèles, *InterEditions*, 1993.
5. M. J. Dayde, I. S. Duff, J. Y. L'excellent et L. Giraud, Évaluation d'ordinateurs vectoriels et parallèles sur un jeu de programmes représentatifs des calculs intensifs à la division avions de l'aérospatiale, Report FR/PA/93/19, April 1993.
6. R. E. Lord, J. S. Kowalik et S. P. Kumar, Solving Linear Algebraic Equations on an MIMD Computer, *J. A.C.M.*, 1983, *30*, 1, p. 103-117.
7. M. Marrakchi, Optimal Parallel Scheduling for the 2-steps Graph with Constant Task Cost, *Parallel Computing*, 1992, *18*, p. 169-176.
8. M. Marrakchi, Un algorithme parallèle optimal pour la résolution d'un système triangulaire, *RAIRO Rech. Opér.*, 1993, *27*, n° 3, p. 273-280.
9. N. M. Missirlis, Scheduling Parallel Iterative Methods on Multiprocessor Systems, *Parallel Computing*, 1987, *5*, p. 295-302.
10. Y. Robert, The Impact of Vector and Parallel Architectures on the Gaussian Elimination Algorithm, *Manchester University Press*, 1990.