M2AN. MATHEMATICAL MODELLING AND NUMERICAL ANALYSIS
- MODÉLISATION MATHÉMATIQUE ET ANALYSE NUMÉRIQUE

L. GIRAUD
P. SPITERI

Résolution parallèle de problèmes aux limites non linéaires

M2AN. Mathematical modelling and numerical analysis - Modélisation mathématique et analyse numérique, tome 25, n° 5 (1991), p. 579-606

<http://www.numdam.org/item?id=M2AN_1991__25_5_579_0>

MATHEMATICAL MODELLING AND NUMERICAL ANALYSIS
MODÉLISATION MATHÉMATIQUE ET ANALYSE NUMÉRIQUE
(Vol. 25, n° 5, 1991, p. 579 à 606)

RÉSOLUTION PARALLÈLE DE PROBLÈMES AUX LIMITES NON LINÉAIRES (*)

L. GIRAUD (1), P. SPITERI (1)

Communiqué par F. ROBERT

Résumé. — *Dans le présent travail, nous considérons les méthodes parallèles de relaxation synchrones et asynchrones par sous domaines pour la résolution numérique de problèmes d'équations aux dérivées partielles non linéaires. Ces algorithmes parallèles ont été implantés sur deux types d'architecture, un multiprocesseur à mémoire distribuée et un multiprocesseur à mémoire partagée. L'analyse de la convergence des algorithmes considérés s'effectue en mettant en évidence une propriété de contraction en norme vectorielle de l'opérateur de point fixe associé au problème. Enfin, on expose les résultats expérimentaux obtenus, et en particulier, on compare les performances des algorithmes parallèles par rapport à celles des méthodes séquentielles.*

Abstract. — *In this work we consider the subdomain parallel synchronous and asynchronous relaxation methods for solving non-linear boundary value problems. These parallel algorithms have been implemented on two kinds of architecture : a loosly coupled multiprocessor and a shared memory multiprocessor. The convergence analysis of the algorithms is proved thanks to a norm vectorial property of the fixed point application associated to the problem. Lastly, we present the experimental results, and in particular, we compare the performance of the parallel algorithms with these of the sequential methods.*

INTRODUCTION

De plus en plus nombreux sont les problèmes de mathématiques appliquées dont la résolution met en œuvre des traitements numériques volumineux. Compte tenu des caractéristiques des machines séquentielles

(*) Received March 1990.

(1) E.N.S.E.E.I.H.T., I.R.I.T. U.A. C.N.R.S. n° 1399, 2, rue Camichel, 31071 Toulouse Cedex.

traditionnelles les plus puissantes, ces traitements sont difficilement envisageables à cause du coût prohibitif des calculs requis. L'un des domaines d'application les plus sensibles à ce type de contraintes est la résolution de systèmes algébriques de grandes tailles issus de la discrétisation de problèmes d'équations aux dérivées partielles. Afin de diminuer les temps de restitution des traitements informatiques, le parallélisme apparaît comme une des solutions envisageables.

Parmi les multiples méthodes numériques parallèles nous nous sommes plus spécialement intéressés, dans le présent travail, aux algorithmes de relaxation parallèles synchrones et asynchrones. Dans ce contexte et, pour la résolution numérique d'équations aux dérivées partielles, la parallélisation considérée repose sur une décomposition en sous domaines disjoints de l'ouvert Ω sur lequel le problème est défini. Chaque sous domaine, ainsi construit, est alors pris en charge par un processeur. Ce type d'algorithme a été introduit à l'origine par D. Chazan et M. Miranker [11] pour la résolution des systèmes linéaires. Dans le cadre de systèmes non linéaires de nombreux travaux ont ensuite été effectués par :

— F. Robert, M. Charnay et F. Musy [36] dans le cas d'algorithmes chaotiques synchrones,

— J. C. Miellou [28] dans le cas d'algorithmes chaotiques à retards où les retards sont bornés,

— G. M. Baudet [3] dans le cas d'algorithmes parallèles asynchrones, où les résultats de J. C. Miellou ont été étendus au cas où les retards ne sont plus bornés ce qui autorise une implantation tolérante aux pannes sur une architecture multiprocesseur.

Du point de vue de l'analyse de ces méthodes, la convergence de ces algorithmes a été étudiée dans divers contextes mathématiques :

— dans un premier temps, des résultats de convergence ont été établis par D. Chazan et M. Miranker [11], F. Robert [35] et J. C. Miellou [28] lorsque l'application de point fixe associée à l'opérateur à inverser est contractante en norme vectorielle ; en particulier J. C. Miellou, P. Spiteri [30] (*cf.* [39]) ont mis en évidence dans ce contexte une condition suffisante de convergence ;

— dans un second temps, des résultats de convergence en ordre partiel ont été obtenus par J. C. Miellou [29], C. Jacquemard [24] et M. N. El Tarazi [16] en utilisant le principe du maximum discret ;

— enfin M. N. El Tarazi [17] a établi un résultat de convergence pour une norme scalaire appropriée.

Signalons également les travaux de D. Bertsekas, J. Tsitsiklis [8] et J. C. Miellou, Ph. Cortey-Dumond, M. Boulbrachene [31] qui permettent de

présenter dans un contexte unifié les études théoriques précédentes. Il convient également de signaler les travaux de A. Chine [12] et F. Robert [34] sur les itérations discrètes.

Sur le plan expérimental, des essais numériques ont été réalisés par G. M. Baudet [3] sur C.M.M.P., G. Authié [1] sur un prototype de multiprocesseur à mémoire partagée, et *via* des simulations d'exécutions parallèles par J. L. Rosenfeld [37], J. Juliand, G. R. Perrin, P. Spiteri [25] et M. N. El Tarazi [16].

Sur le plan informatique et dans l'état actuel des technologies, deux types d'architectures parallèles prédominent, qui se différencient par leur gestion mémoire : les multiprocesseurs à mémoire distribuée et les multiprocesseurs à mémoire partagée.

En vue de résoudre numériquement des problèmes aux dérivées partielles non linéaires, nous avons étudié dans le présent travail, d'une part, l'analyse de la convergence des algorithmes de relaxation parallèles synchrones et asynchrones et d'autre part, l'implantation de ces méthodes sur des machines multiprocesseurs à mémoire partagée et à mémoire distribuée. Nous nous sommes plus particulièrement intéressés à la résolution du problème de l'obstacle, du problème des équations d'Hamilton-Jacobi-Bellman discrétisées et linéarisées, ainsi qu'à la résolution d'un problème de diffusion non linéaire. Pour ces différents problèmes, il est bien connu que les méthodes de relaxation séquentielles sont particulièrement bien adaptées [39] ; il nous apparaissait alors opportun de les paralléliser. Dans ce contexte, l'analyse de la convergence peut être effectuée soit par des techniques d'ordre partiel [29], soit en mettant en évidence une propriété de contraction en norme vectorielle de l'application de point fixe associée au problème. Or, cette dernière propriété est difficile à vérifier ; c'est pourquoi il a été nécessaire, dans le cadre de la résolution numérique de problèmes d'équations aux dérivées partielles non linéaires discrétisées, de caractériser, en terme d'accrétivité, des matrices à diagonale dominante qui contribuent à vérifier aisément cette propriété de contraction.

Le présent article se subdivise en cinq paragraphes. Dans le premier paragraphe, nous rappelons la formulation des algorithmes parallèles synchrones et asynchrones ainsi que le résultat de convergence en norme vectorielle que nous utilisons pour l'étude des applications. Dans le second paragraphe, en relation avec la notion d'accrétivité [2], nous caractérisons les matrices à coefficients diagonaux strictement positifs et à forte dominance diagonale. Dans le but de résoudre numériquement les problèmes modèles, ces résultats sont ensuite utilisés au paragraphe suivant pour l'étude de la convergence des algorithmes parallèles synchrones et asynchrones. Au paragraphe IV, nous décrivons l'implantation des algorithmes parallèles asynchrones sur un IBM 3090VF, architecture à mémoire partagée, ainsi que le principe d'implantation de ces algorithmes sur un

réseau multi-Transputers ; pour ce dernier type d'architecture, la difficulté majeure résidait dans l'inadéquation existante entre les communications asynchrones entre processeurs, requises par ce type de méthodes, et le mode de communication synchrone, avec émissions et réceptions bloquantes de messages, exploité sur les Transputers. Enfin, le paragraphe V est consacré à l'exposé des résultats expérimentaux obtenus sur les deux types d'architecture utilisées.

I. RAPPEL DE LA FORMULATION DES ALGORITHMES

Soit E un espace de Banach réflexif et β un entier naturel. Pour k appartenant à $\{1, ..., \beta\}$, soit $\{E_k\}$ une famille d'espaces de Banach réflexifs. On note $|.|_k$ une norme de l'espace E_k. On considère également la décomposition de E, en un produit fini de β espaces de Banach du type :

$$E = \prod_{k=1}^{\beta} E_k .$$

Soit X un élément de E, que l'on décompose en :

$$X = (X_1, ..., X_\beta), X_k \in E_k , \quad \forall k \in \{1, ..., \beta\} .$$

Soit F une application de $D(F)$ inclus dans E à valeurs dans $D(F)$ telle que :

$$D(F) = \varnothing \tag{1}$$

on considère, également, la décomposition de F compatible avec celle de E :

$$F(X) = (F_1(X), ..., F_k(X), ..., F_\beta(X)) .$$

On considère le problème de point fixe suivant :

$$\left[\begin{array}{c}\text{Déterminer } X^* \text{ appartenant à } D(F) \text{ tel que} \\ X^* = F(X^*) .\end{array}\right. \tag{2}$$

On se propose de déterminer X^* par des algorithmes de relaxation parallèles synchrones ou asynchrones, dont on rappelle ci-dessous la formulation (D. Chazan, M. Miranker [11], F. Robert *et al.* [36], J. C. Miellou [28], G. M. Baudet [3], D. Bertsekas et J. Tsitsiklis [8]).

Notons $\mathcal{N}$ l'ensemble des entiers naturels.

DÉFINITION 1 : *Une* stratégie *s est définie par une suite $(s(p))$ telle que*

$$\forall p \in \mathcal{N} \quad \{1, ..., \beta\} \supset s(p), s(p) \neq \varnothing \tag{3}$$

$$\forall k \in \{1, ..., \beta\} \textit{ l'ensemble } \{p \in \mathcal{N} / k \in s(p)\} \textit{ est infini} \tag{4}$$

DÉFINITION 2 : *Une suite de* retards r *est définie par une suite* $(r(p))$ *telle que :*

$$\forall p \in \mathcal{N},\ r(p) = (r_1(p), ..., r_k(p), ..., r_\beta(p)) \in \mathcal{N}^\beta \text{ et}$$
$$\forall p \in \mathcal{N},\ \forall k \in \{1, ..., \beta\} \text{ l'application :}$$
$$p \to \rho_k(p) = p - r_k(p)$$

est une fonction non décroissante de p *et vérifie de plus*

$$\rho_k(p) \geqslant 0 \quad et \quad \rho_k(p) = p \quad \forall k \in s(p) \tag{5}$$
$$\lim \rho_k(p) = +\infty \tag{6}$$

Compte tenu des définitions précédentes, les algorithmes parallèles de relaxation asynchrones peuvent alors être définis comme suit.

DÉFINITION 3 : *Soit* $X^{(0)} \in D(F)$; *on considère alors la suite* $(X^{(p)})$ *d'éléments de* E *définie par induction :*

$$\forall p \in \mathcal{N},\ \forall k \in \{1, ..., \beta\},\quad X_k^{(p+1)} = \begin{cases} X_k^{(p)} & si \quad k \notin s(p) \\ F_k(w) & si \quad k \in s(p) \end{cases} \tag{7}$$

où $w = \left\{X_1^{p-r_1(p)}, ..., X_i^{p-r_i(p)}, ...\right\}$, $w \in E$.

Remarque 1 :

1° La notion de stratégie correspond aux numéros des composantes sur lesquelles on travaille. La relation (4) signifie qu'une composante n'est jamais abandonnée définitivement au cours du calcul. La notion de retards rend compte de l'asynchronisme avec lequel est traitée chacune des composantes du vecteur X ; lorsque les retards sont identiquement nuls, la formulation (7) correspond alors aux algorithmes de relaxation synchrones [35] ; si de plus pour tout $p \in \mathcal{N}$ on a :

— $s(p) = \{1, 2, ..., \beta\}$, (7) modélise l'algorithme de Jacobi par blocs.

— $s(p) = p \bmod (\beta) + 1$, (7) modélise l'algorithme de Gauss-Seidel par blocs.

D'une manière générale l'algorithme défini par (7) modélise une méthode de relaxation, où les calculs sont effectués avec les valeurs d'interactions disponibles à l'itération considérée.

2° La relation (5) précise que pour le processus k on ne considère pas de retard dans le calcul de la k-ième composante ; de plus la relation (6) prend en compte des retards infinis [3] ce qui rend compte de situations où l'un des processeurs de calcul est en panne.

On rappelle ci-dessous un résultat général de convergence des algorithmes parallèles asynchrones (*cf.* [3], [28] et [35]).

PROPOSITION 1 : *Soit F une application de $D(F)$ inclus dans E à valeur dans $D(F)$.*

L'hypothèse (1) *étant vérifiée, on suppose en outre que :*

$$F \text{ admet un point fixe } X^* \in D(F) \tag{8}$$

$$F \text{ est contractante pour la norme vectorielle } q \text{ c'est-à-dire} \tag{9}$$

$\exists$ *une matrice de contraction* $\mathfrak{J}$ *de type* (β, β) *non négative telle que* $\rho(\mathfrak{J}) < 1$ *et vérifiant :*

$$\forall W \in D(F) \quad q(F(X^*) - F(W)) < \mathfrak{J} \cdot q(X^* - W)$$

où $q(X) = \{|X_1|_1, \ldots, |X_k|_k, \ldots, |X_\beta|_\beta\}$ *et* $\rho(\mathfrak{J})$ est le rayon spectral de $\mathfrak{J}$.

Alors (7) *définit* $X^{(p)}$ *quel que soit* $p \in N$, $X^{(p)} \in D(F)$ *et* $\{X^{(p)}\}$ *converge fortement vers* X^* *point fixe de* F.

II. QUELQUES PROPRIÉTÉS UTILES POUR L'ANALYSE DE LA CONVERGENCE DES ALGORITHMES

II.1. Le contexte mathématique général

Soit E un espace de Banach réel et E^* son dual ; on note respectivement $|.|$ et $|.|^*$ les normes définies sur E et E^* et $\langle , \rangle$ la forme bilinéaire qui met en dualité E et E^*. Pour chaque $X \in E$ on considère l'opérateur de dualité G associé à E, et qui est un opérateur de E vers E^* défini par

$$\forall X \in E, \quad G(X) = \{g \in E^* \text{ t.q. } |g|^* = |X|, \langle g, X \rangle = |X|^2\} . \tag{10}$$

Grâce au théorème de Hahn-Banach on montre que $G(X)$ est une multi-application non vide, fermée, qui coïncide avec le sous-différentiel de la fonction $1/2\,|X|^2$ (*cf.* [2]).

On rappelle également la notion d'opérateur accrétif (*cf.* [2], [5]).

DÉFINITION 4 : *Un opérateur multivoque $\mathscr{A}$ de E dans E est un opérateur* accrétif (*resp.* fortement accrétif) *si :*

$$\forall (X, Y) \in \mathscr{A}, \forall (X', Y') \in \mathscr{A}, \quad \exists g \in G(X - X') \text{ tel que : } \langle Y - Y', g \rangle \geqslant 0$$

(resp. $\forall (X, Y) \in \mathscr{A}, \forall (X', Y') \in \mathscr{A}, \exists g \in G(X - X')$ *et* $\exists m \in \mathbb{R}^+$ *tel que :*

$$\langle Y - Y', g \rangle \geqslant m |X - X'|^2) .$$

Remarque 2 : Si E est un espace de Hilbert, la notion d'opérateur accrétif

coïncide avec celle d'opérateur monotone, puisque si $X \neq X'$, on vérifie aisément que $g = X - X'$.

PROPOSITION 2 : *Soit Λ^d une multi-application non décroissante. Alors Λ^d est accrétive.*

Preuve : Supposons que pour tout $(X, Y) \in \Lambda^d$ et $(X', Y') \in \Lambda^d$, on ait :

$$X \geqslant X' \Rightarrow Y \geqslant Y' .$$

Alors pour tout nombre réel σ positif on obtient

$$X - X' + \sigma (Y - Y') \geqslant X - X' \geqslant 0 .$$

Donc

$$| X - X' + \sigma (Y - Y') | \geqslant | X - X' | .$$

Par conséquent, en utilisant les propriétés classiques des opérateurs accrétifs (V. Barbu [2]), la précédente inégalité a pour conséquence que Λ^d est un opérateur accrétif.

II.2. Caractérisation des matrices fortement accrétives dans $\mathbb{R}^n$:

Soit $n \in \mathcal{N}$ un entier naturel ; on suppose ici que A de coefficients a_{lk} est une matrice carré réelle de dimension $n \times n$. De plus on note x_k les composantes du vecteur $X \in \mathbb{R}^n$.

PROPOSITION 3 : *Une condition nécessaire et suffisante pour qu'une matrice A soit fortement accrétive (resp. accrétive) dans $\mathbb{R}^n$ muni de la norme euclidienne, est que A soit une matrice fortement définie positive (resp. semi-définie positive), c'est-à-dire qu'il existe un nombre réel positif m tel que :*

$$\begin{aligned} &\langle A . X, X \rangle \geqslant m \, | X |_2^2 , \quad \forall X \in \mathbb{R}^n, X \neq 0 \\ &(\textit{resp.} \ \langle A . X, X \rangle \geqslant 0 , \quad \forall X \in \mathbb{R}^n, X \neq 0) \end{aligned} \tag{11}$$

où $\langle ., . \rangle$ désigne le produit scalaire dans $\mathbb{R}^n$ et $| . |_2$ la norme euclidienne dans $\mathbb{R}^n$.

Preuve : Soit $X \in \mathbb{R}^n$; classiquement on sait que le sous-différentiel de l'application $X \to 1/2 | X |^2$ est défini par :

$$\forall X \in \mathbb{R}^n , \ G(X) = \{ g \in \mathbb{R}^n \text{ t.q. } g = X \text{ si } X \neq 0 \text{ et } g = 0 \text{ si } X = 0 \ \} . \tag{12}$$

1° Supposons que A soit une matrice fortement accrétive ; on a alors

$$\langle A . X, g \rangle \geqslant m | X |_2^2 , \ \forall X \in \mathbb{R}^n , \ X \neq 0 \text{ et } g \in G(X) . \tag{13}$$

Compte tenu de (12), on obtient bien l'inégalité (11) et A est donc une matrice fortement définie positive.

2° Réciproquement, si l'inégalité (11) est vraie pour tout $X \in \mathbb{R}^n$ $(X \neq 0)$ alors, en utilisant (12), on obtient bien l'inégalité (13) et A est une matrice fortement accrétive.

De plus, si X est identiquement nul, l'inégalité (13) est trivialement vérifiée.

PROPOSITION 4 : *Une condition nécessaire et suffisante pour qu'une matrice A soit fortement accrétive (resp. accrétive) dans* $\mathbb{R}^n$ *muni de la norme* l_∞, *est qu'il existe un nombre réel positif m tel que pour tout* $k \in \{1, ..., n\}$:

$$a_{kk} \geqslant m \tag{14}$$

$$a_{kk} - \sum_{\substack{j=1 \\ j \neq k}}^{n} |a_{kj}| \geqslant m \tag{15}$$

(*resp.* $a_{kk} \geqslant 0$, $a_{kk} - \sum_{\substack{j=1 \\ j \neq k}}^{n} |a_{kj}| \geqslant 0$).

Preuve : 1° Supposons tout d'abord que A est une matrice fortement accrétive et posons $\bar{g} = g/|X|_\infty$, $g \in G(X)$; donc $|\bar{g}|^* = 1$.

Considérons à présent l'ensemble J_M défini par :

$$J_M = \{i \in \{1, 2, ..., n\} \text{ tel que } |x_i| = |X|_\infty\}$$

où $|.|$ est la valeur absolue ; dans ce cas on peut définir l'application de dualité comme suit :

$$g_i = \begin{cases} \theta_i \text{ sign } (x_i) & \text{si} \quad i \in J_M \\ 0 & \text{si} \quad i \notin J_M \end{cases}$$

Avec $0 \leqslant \theta_i \leqslant 1$, $\forall i \in J_M$ et $\sum_{i \in J_M} \theta_i = 1$.

La matrice A étant fortement accrétive, on a :

$$\langle A \,.\, X, \bar{g} \rangle = \sum_{i \in J_M} \theta_i \sum_{j=1}^{n} a_{ij} x_j \text{ sign } (x_i) \geqslant m |X|_\infty . \tag{16}$$

Considérons d'abord l'inégalité (16) pour $X = e_k$ où e_k est le k-ième vecteur de la base canonique de $\mathbb{R}^n$; nous avons alors

$$\theta_k = 1 \quad \text{et} \quad \theta_j = 0 , \quad \forall j \neq k .$$

De plus, dans ce cas, les composantes de X étant toutes nulles sauf la k-ième

l'inégalité (16) devient :

$$\langle A \,.\, X, g \rangle = a_{kk} \geqslant m \,. \tag{17}$$

La relation étant vérifiée pour tous les vecteurs de la base canonique, on obtient bien l'inégalité (14).

On peut écrire l'inégalité (16) comme suit :

$$\langle A \,.\, X, \bar{g} \rangle = \sum_{i \in J_M} \theta_i \left(a_{ii} x_i + \sum_{\substack{j=1 \\ j \neq i}}^{n} a_{ij} x_j \right) \text{sign } (x_i) \geqslant m \, |X|_\infty \,.$$

Soit $k \in \{1, 2, ..., n\}$ un indice fixé, et $X^{(k)} \in \mathbb{R}^n$ un vecteur de composantes :

$$x_k^{(k)} = 1 \quad \text{et} \quad x_j^{(k)} = - \text{sign } (a_{kj}) \,, \quad \forall j \neq k \,.$$

Pour ce choix particulier de X, nous avons :

$$\text{sign } (x_k^{(k)}) = 1$$

et il est aussi possible de choisir :

$$\theta_k = 1 \quad \text{et} \quad \theta_i = 0 \,, \quad \forall i \neq k \,, \quad i \in J_M - \{k\} \,.$$

Dans ces conditions nous obtenons :

$$a_{kk} - \sum_{\substack{j=1 \\ j \neq k}}^{n} |a_{kj}| \geqslant m$$

et comme $k \in \{1, 2, ..., n\}$, nous vérifions bien l'inégalité (15).

2° Réciproquement, supposons que les inégalités (14) et (15) soient vérifiées. On peut, par ailleurs, facilement vérifier l'inégalité suivante :

$$- |a_{ij}| \,.\, |x_j| \leqslant a_{ij} x_j \,.\, \text{sign } (x_i) \,, \quad \forall X \in \mathbb{R}^n \,. \tag{18}$$

Alors, le membre de gauche de la relation (16) peut s'écrire :

$$\langle A \,.\, X, \bar{g} \rangle = \sum_{i \in J_M} \theta_i \left(a_{ii} |x_i| + \sum_{\substack{j=1 \\ j \neq i}}^{n} a_{ij} x_j \,.\, \text{sign } (x_i) \right) \,, \quad \forall X \in \mathbb{R}^n$$

et compte tenu de la relation (18) nous obtenons :

$$\langle A \,.\, X, \bar{g} \rangle \geqslant \sum_{i \in J_M} \theta_i \left(a_{ii} |x_i| - \sum_{\substack{j=1 \\ j \neq i}}^{n} |a_{ij}| \, |x_j| \right) \,, \quad \forall X \in \mathbb{R}^n \,.$$

Or, si $i \in J_M$ on a :

$$|x_i| \geqslant |x_j| , \quad \forall j \in \{1, 2, ..., n\} , \quad i \neq j , \quad \forall X \in \mathbb{R}^n .$$

Donc nous obtenons :

$$\langle A \cdot X, \bar{g} \rangle \geqslant \sum_{i \in J_M} \theta_i \left(a_{ii} - \sum_{\substack{j=1 \\ j \neq i}}^{n} |a_{ij}| \right) |x_i| , \quad \forall X \in \mathbb{R}^n .$$

Donc (14) et (15) étant vérifiées, nous avons donc :

$$\langle A \cdot X, \bar{g} \rangle \geqslant m \left(\sum_{i \in J_M} \theta_i \right) |X|_\infty = m |X|_\infty , \quad \forall X \in \mathbb{R}^n$$

et, en posant $g = \bar{g} \cdot |X|_\infty$, nous vérifions bien que A est une matrice fortement accrétive.

PROPOSITION 5 : *Une condition nécessaire et suffisante pour qu'une matrice A soit fortement accrétive (resp. accrétive) dans* $\mathbb{R}^n$ *muni de la norme* l_1, *est qu'il existe un nombre réel positif m tel que pour tout* $k \in \{1, ..., n\}$ *on ait :*

$$a_{kk} \geqslant m \tag{19}$$

$$a_{kk} - \sum_{\substack{j=1 \\ j \neq k}}^{n} |a_{jk}| \geqslant m \tag{20}$$

(*resp.* $a_{kk} \geqslant 0$, $a_{kk} - \sum_{\substack{j=1 \\ j \neq k}}^{n} |a_{jk}| \geqslant 0$).

Preuve :

1° Supposons, d'abord, que la matrice A est fortement accrétive et soit $\bar{g} = g/|X|_1$, tel que $|\bar{g}|^* = 1$. Dans ce contexte, l'application de dualité est à présent définie par :

$$\bar{g}_i = \text{sign } (x_i) , \quad \forall i \in \{1, 2, ..., n\}$$

et on a :

$$\langle A \cdot X, \bar{g} \rangle = \sum_{i=1}^{n} \sum_{j=1}^{n} a_{ij} x_j \text{ sign } (x_i) \geqslant m |X|_1 , \quad \forall X \in \mathbb{R}^n . \tag{21}$$

Considérons l'inégalité (21) pour $X = e_k$ où e_k est le k-ième vecteur de la

base canonique de $\mathbb{R}^n$; on obtient alors :

$$\langle A \,.\, X,\ \bar{g} \rangle = \sum_{i=1}^{n} a_{ik} \operatorname{sign}(x_i) \geqslant m \,. \tag{22}$$

Puisque $\operatorname{sign}(x_k) = 1$ et $\operatorname{sign}(x_i) \in [-1, 1]$ pour $i \neq k$, on peut choisir :

$$\operatorname{sign}(x_i) = 0 \quad \text{pour} \quad i \neq k$$

et la relation (22) devient :

$$\langle A \,.\, X,\ \bar{g} \rangle = a_{kk} \geqslant m$$

et comme $k \in \{1, ..., n\}$, on obtient bien l'inégalité (19).

De plus, pour le même choix de X, on peut écrire :

$$\begin{cases} \operatorname{sign}(x_i) = -1 & \text{si} \quad a_{ik} > 0 \quad \text{et} \quad i \neq k \\ \operatorname{sign}(x_i) = +1 & \text{si} \quad a_{ik} < 0 \quad \text{et} \quad i \neq k \end{cases}$$

de telle sorte que :

$$a_{ik} \,.\, \operatorname{sign}(x_i) = -|a_{ik}|\,, \quad \forall i \neq k,\ k \in \{1, ..., n\}\,.$$

L'inégalité (22) devient alors :

$$a_{kk} - \sum_{\substack{i=1 \\ i \neq k}}^{n} |a_{ik}| \geqslant m \,.$$

Enfin on peut recommencer le même travail pour tous les vecteurs de la base canonique et on obtient bien l'inégalité (20).

2° Réciproquement, supposons que les inégalités (19) et (20) soient valables ; l'inégalité (21) peut alors être écrite comme suit :

$$\langle A \,.\, X,\ \bar{g} \rangle = \sum_{j=1}^{n} \left(a_{jj} |x_j| + \sum_{\substack{i=1 \\ i \neq j}}^{n} a_{ij}\, x_j \operatorname{sign}(x_i) \right), \quad \forall X \in \mathbb{R}^n \,.$$

Or l'inégalité (18) étant encore vraie ici, on obtient :

$$\langle A \,.\, X,\ \bar{g} \rangle \geqslant \sum_{j=1}^{n} \left(a_{jj} - \sum_{\substack{i=1 \\ i \neq j}}^{n} |a_{ij}| \right) |x_j|\,, \quad \forall X \in \mathbb{R}^n \,.$$

Compte tenu de l'inégalité (20), on obtient alors :

$$\langle A \,.\, X,\ \bar{g} \rangle \geqslant m\, |X|_1\,, \quad \forall X \in \mathbb{R}^n \,.$$

En posant $g = \bar{g}\, |X|_1$, A est donc une matrice fortement accrétive.

III. APPLICATION A L'ANALYSE DE LA CONVERGENCE D'ALGORITHMES PARALLÈLES POUR RÉSOUDRE DES PROBLÈMES AUX LIMITES NON LINÉAIRES

III.1. Le cas général

On considère à présent le système algébrique non linéaire de la forme :

$$A \cdot X + \Lambda^d(X) - b \ni 0 \tag{23}$$

où A est une matrice de type $\nu \times \nu$, X et b sont deux vecteurs de $\mathbb{R}^\nu$ et Λ^d est un opérateur diagonal, en général, multivoque.

Remarque 3 : En général, un système du type (23) est issu de la discrétisation d'équations ou d'inéquations aux dérivées partielles, ν représentant le nombre de points de discrétisation.

On considère à présent une décomposition du problème (23) en β blocs, du type :

$$A_{kk} \cdot X_k + \Lambda_k^d(X_k) - b_k + \sum_{j \neq k} A_{kj} \cdot X_j \ni 0, \quad \forall k \in \{1, ..., \beta\}. \tag{24}$$

PROPOSITION 6 : *Sous les hypothèses et notations suivantes :*

— $\forall k \in \{1, ..., \beta\}$, *la sous-matrice diagonale* A_{kk} *de la matrice* A *est fortement accrétive, la constante d'accrétivité étant* m_{kk}, (25)

— $\forall (k, j) \in \{1, ..., \beta\}^2$, $j \neq k$, *soit* m_{kj} *la norme matricielle du bloc matrice* A_{kj}, (26)

— $\forall k \in \{1, ..., \beta\}$, Λ_k^d *est une application croissante,* (27)

— *la matrice* $\mathfrak{J}$ *de coefficient diagonaux nuls et hors diagonaux égaux à* m_{kj}/m_{kk} *est une matrice de contraction,* (28)

alors les algorithmes parallèles synchrones et asynchrones de relaxation définis par (7) *et associés à la décomposition par blocs du problème* (23) *convergent vers* X^* *solution de ce problème.*

Preuve : Soit X^* la solution du problème de point fixe associé au problème (23). A l'itération $(p+1)$ soit W les valeurs rendues disponibles par les β processeurs. On peut alors écrire :

$$A_{kk} \cdot X_k^{(p+1)} + \Lambda_k^d(X_k^{(p+1)}) \ni b_k - \sum_{j \neq k} A_{kj} \cdot W_j, \quad \forall k \in \{1, ..., \beta\}$$

$$A_{kk} \cdot X_k^* + \Lambda_k^d(X_k^*) \ni b_k - \sum_{j \neq k} A_{kj} \cdot X_j^*, \quad \forall k \in \{1, ..., \beta\}.$$

Soustrayons membre à membre ces relations et multiplions par $g_k \in G_k(X_k^{(p+1)} - X_k^*)$; compte tenu des hypothèses (25) et (27), l'opérateur $A_{kk} + \Lambda_k^d$ est un opérateur fortement accrétif de constante d'accrétivité

m_{kk}. De plus, le second membre peut être majoré par la quantité :

$$\sum_{j \neq k} m_{kj} \cdot |W_j - X_j^*|_j \, |X_k^{(p+1)} - X_k^*|_k$$

on a donc

$$m_{kk} |X_k^{(p+1)} - X_k^*|_k \leq \sum_{j \neq k} m_{kj} \cdot |W_j - X_j^*|_j , \quad \forall k \in \{1, ..., \beta\}$$

d'où

$$|X_k^{(p+1)} - X_k^*|_k = |F_k(W) - F_k(X^*)|_k \leq \sum_{j \neq k} \frac{m_{kj}}{m_{kk}} \cdot |W_j - X_j^*|_j , \quad \forall k \in \{1, ..., \beta\}$$

soit matriciellement

$$q(F(X^*) - F(W)) \leq \mathfrak{J} \cdot q(X^* - W)$$

et compte tenu de l'hypothèse (28), on obtient bien la convergence des algorithmes parallèles synchrones et asynchrones associés à la décomposition par blocs du problème (23) en appliquant le résultat de la proposition 1.

Remarque 4 : En pratique, il suffit de vérifier que la matrice $\mathfrak{J}$ a un rayon spectral inférieur à 1, puisque $\mathfrak{J}$ est une matrice non négative. Cette condition sera vérifiée si, par exemple, la matrice $\bar{M}$ de coefficient diagonaux m_{kk} et hors diagonaux $(-m_{kj})$ est une M-matrice (*cf.* [30], [39]).

Dans le cas où le nombre β de blocs est grand devant le nombre de processeurs, on a intérêt à considérer des algorithmes parallèles associés à une décomposition plus grossière du problème (23) en α blocs ($\alpha < \beta$) de dimensions supérieures ; ainsi on regroupe un ensemble de blocs adjacents de la décomposition initiale pour constituer un nouveau bloc ; celui-ci sera traité par un algorithme du type (7) sur l'un des α processeurs. On appelle ce type d'algorithme, algorithme parallèle synchrone ou asynchrone associé à la décomposition en sous-domaines du problème (23). On peut formuler ces algorithmes de façon analogue à (7), où ici $\bar{r}(p) \in \mathcal{N}^\alpha$ et $\{1, ..., \alpha\} \supset \bar{s}(p)$; grâce à un résultat de [30] et [39] on peut alors énoncer :

COROLLAIRE 1 : *Sous les hypothèses de la proposition 6, les algorithmes parallèles synchrones et asynchrones associés à la décomposition en sous-domaines du problème* (23) *convergent vers* X^* *solution de ce problème.*

III.2. Applications

Soit Ω un domaine ouvert borné inclus dans $\mathbb{R}^2$ (ou $\mathbb{R}^3$) et Γ la frontière de Ω.

a) *Problème n° 1 : un problème de diffusion fortement non linéaire*

Soit $a \in \mathbb{R}^+$ et $f \in L^2(\Omega)$. On considère le problème suivant :

$$\begin{cases} \text{Déterminer } u \text{ solution de} \\ -\Delta u + e^{au} = f \quad \text{dans} \quad \Omega \\ u/\Gamma = 0 . \end{cases} \tag{29}$$

On discrétise ce problème par le schéma classique aux différences finies à cinq points avec numérotation lexicographique des points du maillage ou numérotation rouge-noir par blocs. Dans les deux cas, on obtient un système algébrique du type (23) où ici, A est un bloc matrice, chaque bloc diagonal étant lui-même tri-diagonal ; classiquement, A est une M-matrice. De plus, compte tenu du fait que a est un nombre réel positif, l'opérateur $\Lambda^d(X) = \text{diag}\ (e^{ax_i})$ est un opérateur diagonal croissant, donc accrétif.

Pour étudier la convergence des algorithmes parallèles synchrones et asynchrones appliqués à la résolution du problème discrétisé précédent, on considère la décomposition par points du problème. Si $\mathbb{R}^v$ est normé par la norme l_1 ou l_∞, les coefficients diagonaux de la matrice A sont les coefficients d'accrétivité forte ; de plus, les valeurs absolues des coefficients hors diagonaux constituent pour cette décomposition, les normes des matrices d'interaction. Dans ces conditions, pour les topologies induites par les normes l_1 et l_∞, la matrice $\mathfrak{J}$ est la matrice de Jacobi associée à la matrice A. Or la matrice A étant une M-matrice, $\mathfrak{J}$ est une matrice de contraction et on est dans les conditions d'application du résulat de la proposition 6 ; de plus on peut également appliquer le résultat du corollaire 1, et on obtient ainsi la convergence des algorithmes parallèles synchrones et asynchrones pour toute décomposition plus grossière, en sous-domaines, du problème.

b) *Problème n° 2 : le problème de l'obstacle*

Soit K le cône convexe positif et $f \in L^2(\Omega)$; soit $\mathscr{A}$ l'opérateur elliptique défini par :

$$\mathscr{A}u = -\Delta u - \varepsilon \frac{\partial^2 u}{\partial x\, \partial y} + \theta \frac{\partial u}{\partial x} + \gamma \frac{\partial u}{\partial y} + \mu u$$

où ε, θ, γ et μ sont des constantes réelles et de plus μ est positive.

On considère l'inéquation variationnelle suivante :

$$\begin{cases} \text{Déterminer } u \in K \text{ telle que} \\ a(u, v - u) \geqslant \langle f, v - u \rangle , \quad \forall v \in K \end{cases} \tag{30}$$

où $a(.,.)$ est la forme bilinéaire classique associée à l'opérateur $\mathscr{A}$. Or le problème précédent peut être formulé comme suit :

$$\begin{cases} \text{Déterminer } u \text{ solution de :} \\ \mathscr{A}u - f + \partial\psi_K(u) \ni 0 \end{cases} \tag{31}$$

où $\partial\psi_K$ est le sous-différentiel de la fonction indicatrice ψ_K du convexe K.

La discrétisation du problème précédent par éléments finis du premier ordre conduit à un système algébrique du type (23), où Λ^d correspond à la discrétisation de l'opérateur $\partial\psi_K$. Compte tenu du fait que Λ^d est un opérateur diagonal non décroissant, et pour des valeurs convenables des coefficients ε, θ, γ et μ [38] qui assurent la diagonale dominance stricte des blocs diagonaux de la matrice A, on peut de la même façon montrer la convergence des algorithmes parallèles synchrones et asynchrones par sous-domaines.

c) *Problème n° 3 : le problème d'Hamilton-Jacobi-Bellman discrétisé et linéarisé*

On considère le problème suivant :

$$\begin{bmatrix} \text{Déterminer } X \text{ solution de} \\ \text{Max } (A^1 . X - b^1, A^2 . X - b^2) = 0 \end{bmatrix} \tag{32}$$

où A^1, A^2 sont deux matrices de type $\nu \times \nu$ et b^1, b^2 deux vecteurs de $\mathbb{R}^\nu$. En fait A^1 et A^2 correspondent respectivement à la discrétisation par différences finies classique des opérateurs $\mathscr{A}^1$ et $\mathscr{A}^2$ définis par :

$$\begin{cases} \mathscr{A}^1 = -\left(\dfrac{\partial^2}{\partial x^2} + \dfrac{\partial^2}{\partial y^2} + \dfrac{1}{2}\dfrac{\partial^2}{\partial y\, \partial x} \right) \\ \mathscr{A}^2 = -\left(\dfrac{1}{2}\dfrac{\partial^2}{\partial x^2} + \dfrac{\partial^2}{\partial y^2} + \dfrac{1}{10}\dfrac{\partial^2}{\partial y\, \partial x} \right) . \end{cases}$$

Les matrices A^1 et A^2 ont toutes deux une structure triple diagonale par blocs, chaque bloc diagonal étant lui-même tridiagonal ; de plus les coefficients de ces sous-matrices ont les valeurs suivantes :

$$a^1_{ii} = 3{,}5 \qquad a^2_{ii} = 2{,}9$$

$$a^1_{i,i+1} = a^1_{i,i-1} = -0{,}75 \qquad a^2_{i,i+1} = a^2_{i,i-1} = -0{,}45$$

par ailleurs on vérifie aisément que les normes des sous-matrices hors diagonales, induites par les normes l_1 et l_∞ sont égales à l'unité ; compte tenu des propriétés précédentes, on vérifie que A^1 et A^2 sont des matrices irréductibles à diagonale dominante, donc A^1 et A^2 sont des M-matrices.

On linéarise le problème (32) en définissant une matrice linéarisante $A(X)$ et un vecteur $b \in \mathbb{R}^\nu$ comme suit : pour tout $i \in \{1, ..., \nu\}$ on convient de noter $(A^1 . X - b^1)_i$ la i-ième composante de ce vecteur ; si $(A^1 . X - b^1)_i$ est supérieur à $(A^2 . X - b^2)_i$, alors la i-ième ligne de $A(X)$ sera la i-ième ligne de A^1 et la i-ième composante de b sera

$(b^1)_i$; dans le cas contraire la i-ième ligne de $A(X)$ et la i-ième composante de b seront respectivement la i-ième ligne de A^2 et la i-ième composante de b^2.

On obtient donc le système linéarisé suivant :

$$A(X) \cdot X = b \,. \tag{33}$$

On peut à présent comme au paragraphe III.1 étudier la convergence des algorithmes parallèles synchrones et asynchrones appliqués à la résolution du problème d'Hamilton-Jacobi-Bellman discrétisé et linéarisé. On obtient, dans un premier temps, la convergence de ces algorithmes associés à la décomposition naturelle par blocs ; en effet, pour les topologies associées aux normes l_1 et l_∞, on détermine la matrice $\mathfrak{J}$ qui est une matrice triple diagonale de coefficients diagonaux nuls et de coefficients co-diagonaux égaux à 0,5 ; comme $\rho(\mathfrak{J}) = \cos((\beta + 1)^{-1})$, où β désigne le nombre de blocs de $A(X)$, on vérifie bien que $\mathfrak{J}$ est une matrice de contraction pour le problème de point fixe associé au problème (33). Dans un second temps, on montre la convergence des algorithmes parallèles synchrones et asynchrones associés à la décomposition par sous-domaines en appliquant le résultat du corollaire 1.

Remarque 5 : La matrice J associée à la décomposition par points n'est autre que la matrice de Jacobi associée à la matrice $A(X)$. Compte tenu des propriétés des matrices A^1 et A^2, pour tout $X \in \mathbb{R}^\nu$ la matrice linéarisante $A(X)$ est également une M-matrice, donc J est une matrice de contraction [20]. Ceci nous permet alors d'élargir le résultat précédent, puisqu'il assure la convergence des algorithmes parallèles asynchrones par sous-domaines quel que soit le découpage et quelle que soit la granularité de cette décomposition.

IV. IMPLANTATION DES ALGORITHMES SUR DES ARCHITECTURES PARALLÈLES

IV.1. Architecture à mémoire distribuée

La machine cible, sur laquelle nous avons implanté les algorithmes, est un réseau constitué de 16 Transputers.

Rappelons tout d'abord, brièvement, les principales caractéristiques architecturales d'un Transputer, ainsi que la philosophie de leur utilisation comme constituants de base pour la réalisation d'une architecture multiprocesseur à mémoire distribuée.

Le Transputer est un module incluant un processeur, une mémoire locale et quatre liens bidirectionnels qui lui permettent de communiquer avec quatre autres modules. La connexion des liens est à la charge de l'utilisateur, qui peut ainsi configurer son réseau au gré de l'application qu'il a à traiter.

Les synchronisations et les communications entre les processeurs sont basées sur des échanges de messages avec émissions et réceptions bloquantes, ce qui correspond aux primitives de synchronisation de type rendez-vous C.S.P. [22].

Nous allons maintenant montrer la faisabilité de l'implantation des algorithmes parallèles asynchrones, décrits au paragraphe I, sur un tel réseau en utilisant un mode de communication synchrone.

L'implantation de tels algorithmes sur un réseau de Transputers a nécessité la mise en œuvre d'un protocole de communication assurant l'adéquation entre la méthode numérique asynchrone et les communications en mode message. Ainsi nous avons décomposé le réseau en trois sous-ensembles de processeurs : un processeur dédié au test d'arrêt, des processeurs « arithmétique » et des processeurs « mémoire ». La topologie du réseau est un anneau avec alternativement un processeur « arithmétique » et un processeur « mémoire ». Ce dernier, connecté à deux processeurs « arithmétique » se comporte comme un banc mémoire double port série, gère les variables partagées et les ordres de communication, ce qui autorise un fonctionnement asynchrone entre les traitements numériques. L'anneau est terminé par le processeur dédié au test d'arrêt.

Pour la gestion des données partagées deux stratégies ont été implantées, eu égard au fait que pour le calcul d'un ensemble de composantes C effectué par un processeur donné interviennent des composantes D calculées par les processeurs voisins. La première de ces stratégies consiste à utiliser, pour la mise à jour de cet ensemble C, les valeurs disponibles D calculées par les autres processeurs ; la seconde ne réalise des communications que si les valeurs D ont été mises à jour depuis le précédent calcul de C ; dans le cas contraire les composantes C ne sont pas recalculées et le processeur ne réalise une relaxation que sur les composantes restantes qu'il doit traiter. Par la suite, nous désignerons la première stratégie (resp. la seconde) par méthode sans saut de composantes (resp. par méthode avec

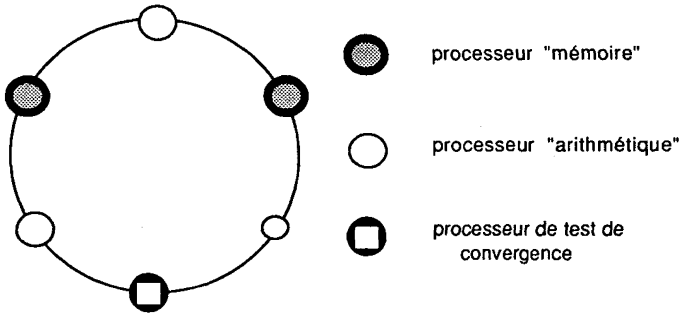

Figure 1. — Topologie du réseau.

saut de composantes). Dans tous les cas, les processeurs de calcul respectent le principe de Gauss, c'est-à-dire qu'un nouveau résultat est immédiatement transmis et que les valeurs les plus récentes sont utilisées par les processeurs.

Nous avons également implanté des méthodes de relaxation parallèles synchrones qui ont le même comportement numérique que les méthodes séquentielles correspondantes. Dans le cas des méthodes avec multicoloriage nous avons exploité le parallélisme naturel introduit par ces numérotations ; dans le cas de la numérotation lexicographique la parallélisation a été construite suivant le principe de la méthode des hyperplans [26], avec un réveil en cascade des processeurs [19].

Remarque 6 : Signalons également qu'il est possible d'implanter ce type d'algorithme sur un réseau de Transputers en utilisant les possibilités multitâche du Transputer [15] ; bien que cette seconde solution présente l'avantage d'utiliser deux fois moins de processeurs, nous l'avons écarté [19]. En effet, cette méthode présente, à notre avis certains désavantages. En premier lieu notre but était non seulement d'étudier le comportement informatique et numérique des algorithmes asynchrones mais aussi de comparer leurs performances avec celles des méthodes synchrones pour lesquelles le recours au multitâche est inutile ; pour faire une comparaison valable il fallait nous placer dans les mêmes conditions expérimentales. En second lieu, l'utilisation du multitâche a conduit à des performances inférieures d'environ 5 % à celles que nous avons obtenues en débanalisant les Transputers ; de plus la différence de performances entre les deux méthodes n'était pas constante mais dépendait de la décomposition en sous-domaines considérée : dans ces conditions il nous était difficile, dans l'analyse des résultats, de distinguer l'effet du multitâche et celui de l'asynchronisme sur le comportement des algorithmes.

IV.2. Architecture à mémoire partagée

La machine à mémoire partagée sur laquelle nous avons implanté les algorithmes est l'IBM 3090-400VF du CNUSC de Montpellier, qui, dans sa configuration actuelle, possède trois unités vectorielles. L'implantation d'algorithmes parallèles asynchrones sur ce type d'architecture consiste simplement à supprimer toutes les primitives de synchronisations utilisées pour les algorithmes parallèles synchrones.

V. RÉSULTATS EXPÉRIMENTAUX

Les résultats présentés ci-après ont été obtenus sur deux types d'architectures différentes. Dans ces expérimentations nous avons résolu le problème n° 1 discrétisé, sur un domaine polygonal, par différences finies avec 12 000 points de discrétisation, ainsi que les problèmes n^{os} 2 et 3 discrétisés,

sur le carré unité, respectivement par éléments finis de type P1 et différences finies classiques avec 20 000 points de discrétisation.

Le problème n° 1 est de plus résolu par un algorithme de relaxation non linéaire par blocs avec numérotation rouge-noir des blocs ; dans ce cas, la convergence des algorithmes peut également être étudiée de la même façon qu'au paragraphe III (*cf.* [39], [40]). Signalons, qu'en séquentiel, un algorithme de gradient conjugué préconditionné SSOR s'est avéré légèrement plus performant que la méthode de relaxation non linéaire utilisée ; par contre, l'algorithme du gradient conjugué préconditionné SSOR s'avère être un solveur peu efficace au niveau d'un sous-domaine en mode parallèle [21]. Par ailleurs, les problèmes n^{os} 2 et 3 sont résolus par un algorithme de relaxation par blocs avec numérotation lexicographique ; des expérimentations séquentielles antérieures (*cf.* [16], [39]) ont montré l'efficacité des méthodes de relaxation non linéaires comparées à des méthodes de gradient conjugué et gradient conjugué préconditionné pour la résolution de ces deux derniers problèmes.

Les résultats expérimentaux sont présentés sous forme de tableaux qui rassemblent les meilleures performances des algorithmes synchrones et asynchrones.

Dans ces tableaux les notations ont la signification suivante :

— nb proc. : nombre de processeurs mettant en œuvre les calculs,

— découpage : α-uplet représentant le nombre de blocs adjacents alloués à chaque processeur,

— min/max : nombre d'itérations minimal, respectivement maximal, effectué par l'un des processeurs,

— Tps restitution : temps de restitution exprimé en secondes obtenu sur machine dédiée,

— Acc. : accélération introduite par la parallélisation, définie comme le rapport du temps séquentiel et du temps de restitution en parallèle,

— Eff. : efficacité de l'algorithme, définie comme le rapport de l'accélération et du nombre de processeurs.

V.1. Résultats sur l'architecture multi-Transputers

Les résultats présentés ci-après ont été obtenus sur une configuration de seize Transputers de type T800, dont au maximum huit sont utilisés pour les calculs (*cf.* tableaux 1, 2, 3).

Nous pouvons faire quelques commentaires spécifiques de l'architecture multi-Transputers.

1° On obtient parfois, tant pour les algorithmes parallèles synchrones qu'asynchrones, des efficacités supérieures à 1. Dans le cas des algorithmes synchrones, et bien que le comportement numérique soit identique au

TABLEAU 1

Problème n° 1 : problème de diffusion non linéaire.

Nb proc.	Mode	Découpage	min/max	Tps restitution	Acc.	Eff.
1	séquentiel	113	58	62,68	/	/
2	synchrone	56-57	58	32,95	1,90	0,95
	asynchrone	56-57	60/59	33,21	1,89	0,94
4	synchrone	28-28-28-29	58	18,10	3,46	0,87
	asynchrone	30-25-28-30	58/63	18,25	3,43	0,86
8	synchrone	14-14-14-14 14-14-14-15	58	10,75	5,83	0,73
	asynchrone	14-14-14-14 14-14-14-15	64/72	11,32	5,54	0,69

TABLEAU 2

Problème n° 2 : problème de l'obstacle.

Nb proc.	Mode	Découpage	min/max	Tps restitution	Acc.	Eff.
1	séquentiel	143	241	258,12	/	/
2	synchrone	71-72	241	135,00	1,91	0,96
	asynchrone	69-74	229/242	126,72	2,04	1,02
4	synchrone	35-36-36-36	241	68,97	3,74	0,94
	asynchrone	35-36-36-36	238/244	65,52	3,94	0,99
8	synchrone	17-18-18-18 18-18-18-18	241	35,63	7,24	0,91
	asynchrone	17-18-18-18 18-18-18-18	252/274	35,90	7,19	0,90

TABLEAU 3

Problème n° 3 : problème d'Hamilton-Jacobi-Bellman.

Nb proc.	Mode	Découpage	min/max	Tps restitution	Acc.	Eff.
1	séquentiel	143	239	375,19	/	/
2	synchrone asynchrone	71-72 71-72	239 244/247	175,73 178,88	2,14 2,10	1,07 1,05
4	synchrone asynchrone	35-36-36-36 39-32-33-39	239 241/291	91,50 96,78	4,1 3,88	1,03 0,97
8	synchrone	17-18-18-18 18-18-18-18	239	47,57	7,89	0,99
	asynchrone	14-16-20-21 22-20-16-14	145/239	34,44	10,89	1,36

TABLEAU 4

Pourcentage des coûts de communications et des attentes par rapport au temps de restitution.

Nb proc.	Mode	Coût des communications	Coût des attentes
2	synchrone asynchrone	0,67 % 0,66 %	1,40 % 0,01 %
4	synchrone asynchrone	3,94 % 3,88 %	8,00 % 0,06 %
8	synchrone asynchrone	18,16 % 17,55 %	16,00 % 0,32 %

séquentiel, l'obtention des efficacités supérieures à 1 se justifie par la présence de deux types de mémoires sur un Transputer : d'une part une mémoire rapide implantée sur la puce, d'autre part, une mémoire plus lente que le processeur situé hors du boîtier. Ainsi, lors d'une exécution

séquentielle, si la taille des données devient importante, celles-ci ne tiennent pas toutes dans la mémoire rapide et par conséquent les accès mémoire ralentissent les calculs. En revanche, lors des exécutions parallèles les données sont distribuées sur les différents processeurs, si bien que la taille des données par processeurs est plus faible, les accès à la mémoire lente sont moins fréquents, ce qui contribue à accélérer le temps de restitution.

Dans le cas des algorithmes asynchrones, ce phénomène explique aussi partiellement les efficacités supérieures à 1, et pour certains découpages, s'y ajoute la diminution du nombre d'itérations qui contribue également à l'accélération de l'algorithme parallèle.

2° Nous avons indiqué au paragraphe IV les diverses variantes d'algorithmes parallèles asynchrones : les méthodes avec ou sans saut de composantes. On peut signaler que les sauts de composantes ont, en général, un effet accélérateur, qui croît avec le nombre de processeurs.

3° On constate que, pour les variantes synchrones et asynchrones, le nombre d'itérations pour un découpage donné sont du même ordre, donc les coûts des communications sont quasiment identiques. Par contre les temps d'attente imposés aux processeurs les moins chargés en mode synchrone sont nettement plus importants que les coûts des conflits d'accès aux variables partagées en mode asynchrone. De plus on constate évidemment que ces coûts augmentent avec le nombre de processeurs ; par contre, pour un nombre de processeurs donné, leurs proportions diminuent par rapport au temps de calcul, lorsque la taille du problème augmente.

Le tableau 4 donne un ordre de grandeur de ces coûts ; les valeurs rassemblées dans ce tableau ont la signification suivante :

— coût des communications : proportion du cumul des temps de communication de chacun des processeurs de calcul par rapport au temps de restitution,

— coût des attentes : proportion, par rapport au temps de restitution, du cumul des temps passés en conflits d'accès pour les méthodes parallèles asynchrones, et du cumul des temps d'attente pour les méthodes synchrones.

4° On peut également noter l'influence de la vitesse de communication sur les performances des algorithmes parallèles. En effet, des expérimentations [20], [21] ont également été réalisées sur un réseau de seize Transputers T414, dont au maximum huit ont été utilisés pour les calculs ; rappelons que ce type de Transputer ne possède pas de coprocesseur arithmétique. Cette différence architecturale par rapport au T800 se traduit par des taux (vitesse de transfert d'une donnée)/(vitesse d'une opération sur cette donnée) différents pour les deux types de réseaux. En double précision ce rapport est de 7,48 pour les T800, alors qu'il n'est que de 1,16 pour les T414. Sur le plan informatique, pour ce type de réseau, les temps de

communications sont réduits et leurs effets sur le temps de restitution sont diminués. C'est pourquoi les algorithmes parallèles synchrones et asynchrones implantés sur le réseau de T414 sont légèrement plus performants en terme d'efficacité, sans que, pour autant, l'une des deux méthodes (synchrone ou asynchrone) s'avère supérieure à l'autre.

V.2. Résultats sur le 3090-400VF

Il sont présentés dans les tableaux 5 et 6.

TABLEAU 5

Problème n° 2 : problème de l'obstacle.

Nb proc.	Mode	Découpage	min/max	Tps restitution	Acc.	Eff.
1	séquentiel	143	241	19,91	/	/
2	synchrone	69-74	241	10,38	1,92	0,96
	asynchrone	69-74	241/251	10,06	1,98	0,99
3	synchrone	47-48-48	241	7,42	2,68	0,89
	asynchrone	45-52-46	248/277	6,93	2,87	0,96

TABLEAU 6

Problème n° 3 : problème d'Hamilton-Jacobi-Bellman.

Nb proc.	Mode	Découpage	min/max	Tps restitution	Acc.	Eff.
1	séquentiel	143	241	21,17	/	/
2	synchrone	71-72	241	10,86	1,95	0,97
	asynchrone	71-72	251/254	11,03	1,92	0,96
3	synchrone	47-48-48	241	7,55	2,80	0,94
	asynchrone	49-44-50	244/275	7,41	2,86	0,95

Comme au paragraphe V.1 nous pouvons formuler quelques remarques spécifiques aux résultats obtenus sur l'IBM 3090VF.

1° Contrairement à l'architecture multi-Transputers, il nous a été impossible de mesurer le coût des conflits d'accès pour les algorithmes parallèles

asynchrones et nous n'avons pas mesuré les temps d'attente en mode parallèle synchrone.

2° De plus, l'IBM 3090VF offre des possibilités de parallélisation et de vectorisation. Pour le type de problèmes non linéaires étudiés, les algorithmes implantés sur cette architecture présentent un faible taux de vectorisation puisque celle-ci n'améliore que d'environ 30 % les performances scalaires.

V.3. Eléments de synthèse

Les résultats des expérimentations révèlent le bon comportement des algorithmes de relaxation parallèles par sous-domaines, qui produisent par rapport aux algorithmes séquentiels de bonnes performances aussi bien en mode parallèle synchrone qu'en mode parallèle asynchrone. Cependant l'analyse des résultats expérimentaux est très complexe et il paraît difficile de tirer des conclusions générales. Néanmoins il apparaît des tendances en fonction :

— du nombre de processeurs disponibles,
— de la taille du problème à résoudre,
— de la nature linéaire ou non linéaire du problème à résoudre,
— du mode de numérotation des points considéré.

Cette analyse apparaît d'autant plus ardue que les systèmes multiprocesseurs que nous avons utilisés ne nous donnent pas, systématiquement, strictement les mêmes performances sur un même cas test. En effet, pour deux exécutions successives, d'une part pour les algorithmes parallèles asynchrones nous n'obtenons pas le même nombre d'itérations, et d'autre part, pour les algorithmes synchrones, les temps de restitution varient. Dans le cas de l'IBM 3090VF ce phénomène peut s'expliquer par le fait que, même en mode dédié, une exécution est perturbée par l'activation de processus du système d'exploitation. Dans le cas de l'architecture multi-Transputers, cette non-reproductibilité des exécutions est d'autant moins explicable que la machine n'est utilisable qu'en mode dédié et que nous avions pris soin de ne pas exploiter les possibilités multitâche disponibles sur les Transputers pour éviter, autant que possible, des phénomènes de cette nature.

L'ensemble des résultats expérimentaux présentés dans les tableaux précédents représentent une partie des résultats sur lesquels se fondent les remarques que nous allons formuler ci-dessous ; cependant nous renvoyons le lecteur à (*cf.* [6], [18], [20], [21]) pour un complément d'information.

1° Lorsque le nombre de processeurs augmente, pour une taille du problème donnée, les performances, en terme d'efficacité, des algorithmes synchrones et asynchrones ont tendance à diminuer. Il existe cependant des

exceptions à cette tendance dans la mesure où, pour le problème d'Hamilton-Jacobi-Bellman résolu sur huit processeurs de calcul utilisés en mode asynchrone, on obtient une efficacité de 1,36. De plus, toujours pour une taille de problème fixée, il existe un nombre de processeurs critique à ne pas dépasser ; ceci est d'autant plus sensible que la taille du problème est petite.

2° Les expérimentations montrent l'influence de la décomposition en sous-domaines d'une part sur la vitesse de convergence des algorithmes parallèles asynchrones, d'autre part, sur le paramètre de relaxation optimal associé. Signalons qu'à notre connaissance, il n'existe aucun résultat théorique pour déterminer a priori le découpage optimal, et pour calculer le paramètre optimal associé à une décomposition donnée. Toutefois il existe certains critères heuristiques qui permettent de guider dans le choix du découpage en sous-domaines. En première approximation on peut dégager les principes suivants :

— minimisation des interactions entre les différents sous-domaines,

— définition de processus dont les charges ne soient pas trop disproportionnées les unes par rapport aux autres, l'excès d'asynchronisme donnant des résultats peu probants.

3° Au vu des résultats, il n'apparaît pas que l'architecture de la machine multiprocesseur (mémoire distribuée ou partagée) joue un rôle déterminant sur l'efficacité des algorithmes. En effet, pour un nombre de processeurs donné on retrouve sensiblement les mêmes efficacités sur les deux types d'architecture que nous avons considérées. Signalons toutefois que le faible nombre de processeurs disponibles sur l'IBM 3090VF du C.N.U.S.C. ne nous a pas permis de tester de manière approfondie le comportement des algorithmes parallèles sur ce type d'architecture à mémoire partagée.

4° Au vu de l'ensemble des expérimentations effectuées, on constate que l'on obtient, pour un nombre de processeurs donné, des temps de restitution du même ordre pour les deux méthodes synchrone ou asynchrone, sans que l'une d'elles ne s'avère plus performante que l'autre, sauf, peut-être, dans le cas du problème de l'obstacle.

Sur un plan informatique, les algorithmes parallèles asynchrones, dans la mesure où on est assuré de leur convergence, permettent de s'affranchir des contraintes de gestion des primitives de synchronisation. De plus, dans la mesure du possible, ils présentent l'avantage de garantir la convergence des méthodes de relaxation parallèles quel que soit le découpage considéré et la stratégie algorithmique (synchrone ou asynchrone) adoptée. Par contre, à notre connaissance, il n'existe aucun critère théorique pour déterminer a priori laquelle des stratégies synchrone et asynchrone sera la plus performante.

Enfin, signalons qu'un déséquilibrage raisonnable de la taille des sous-domaines ne dégrade pas systématiquement le comportement des algo-

rithmes parallèles asynchrones, alors qu'il augmente toujours les temps de restitution des algorithmes parallèles synchrones, en introduisant des temps d'attente supérieurs lors des synchronisations.

5° Il est à noter que, comme pour les méthodes de relaxation séquentielles, l'utilisation du multicoloriage des points a un effet accélérateur sur la convergence des algorithmes parallèles asynchrones ; par ailleurs cet effet a tendance à diminuer, voire à disparaître, lorsque le nombre de processeurs de calcul augmente.

TABLEAU 7

Effet de la numérotation sur la vitesse des algorithmes parallèles asynchrones pour le problème n° 1.

Nb. proc.	1	2	4	8
Numérotation lexicographique	92 s	47 s	23 s	13 s
Numérotation Rouge Noir par blocs	75 s	38 s	20 s	14 s

BIBLIOGRAPHIE

[1] G. AUTHIÉ, *Contribution à l'optimisation de flots dans les réseaux. Un multiprocesseur expérimental pour l'étude des itérations asynchrones.* Thèse de Doctorat ès Sciences, Université Paul Sabatier, Toulouse, 1987.

[2] V. BARBU, *Non linear semi-groups and differential equations in Banach spaces.* Noordhoff International Publishing, 1976.

[3] G. M. BAUDET, *Asynchronous iterative methods for multi-processor.* J. Ass. Comput. Mach. *25,* 226-244, 1978.

[4] F. L. BAUER, *On the field of values subordinate to a norm.* Numer. Math. *4,* 103-113, 1962.

[5] Ph. BENILAN, *Equation d'évolution dans un espace de Banach quelconque et applications.* Thèse de Doctorat ès Sciences, Orsay, 1972.

[6] S. BENJELLOUN, P. SPITERI, G. AUTHIÉ, *Parallel algorithms for solving the obstacle problem.* Computational Mechanics Publ., Springer-Verlag, *2,* 275-281, 1989.

[7] A. BENSOUSSAN, J. L. LIONS, *Applications des inéquations variationnelles en contrôle stochastique.* Dunod, Paris, 1978.

[8] D. BERTSEKAS, J. TSITSIKLIS, *Parallel and distributed computation.* Numerical Methods. Prentice Hall, 1989.

[9] F. F. BONSALL, J. DUNCAN, *Numerical ranges of operators on normed spaces and elements of normed algebras.* London Math. Soc. Lecture Note Ser. 2, Cambridge University Press, 1971.

[10] H. BREZIS, L. C. EVANS, *A variational inequality approach to the Bellmann-Dirichlet equation for two elliptic operators.* Arch. Rat. Mech. Anal. *71,* 1-14, 1979.

[11] D. CHAZAN, M. MIRANKER, *Chaotic relaxation.* Linear algebra and its appl., *2,* 199-222, 1969.

[12] A. CHINE, *Etude de la convergence globale et locale des itérations discrètes asynchrones.* Rapport technique 35, Informatique et Mathématiques Appliquées de Grenoble (IMAG), mars 1988.

[13] P. COMTE, J. C. MIELLOU, P. SPITERI, *La notion d'accrétivité, applications.* C. R. Acad. Sci. Paris, t. 283, 655-658, 1976.

[14] Ph. CORTEY DUMOND, *Analyse numérique de problèmes à frontières libres.* Thèse de Doctorat ès Sciences, Université Pierre-et-Marie Curie, Paris VI, 1985.

[15] D. EL BAZ, *Mise en œuvre d'algorithmes itératifs asynchrones sur un réseau de Transputers.* La lettre du Transputer, n° 3, 31-40, 1989.

[16] M. N. EL TARAZI, *Contraction et ordre partiel pour l'étude d'algorithmes synchrones et asynchrones en analyse numérique.* Thèse de Doctorat ès Sciences, Université de Besançon, 1981.

[17] M. N. EL TARAZI, *Some convergence result for asynchronous algorithms.* Numer. Math. *39,* 325-340, 1982.

[18] L. GIRAUD, P. SPITERI, Ph. BERGER, *Parallel asynchronous and synchronous 2D Poisson equation solvers on a processor network.* Computational Mechanics Publ., Springer-Verlag, *2,* 265-271, 1989.

[19] L. GIRAUD, P. SPITERI, Ph. BERGER, *Implantation d'algorithmes parallèles synchrones et asynchrones sur un réseau multi-Transputers.* Rapport E.N.S.E.E.I.H.T.-I.R.I.T., 1989.

[20] L. GIRAUD, P. SPITERI, *Résolution parallèle des équations d'Hamilton-Jacobi-Bellman discrétisées et linéarisées sur un calculateur distribué.* Publications Mathématiques de Besançon, 31-46, 1989.

[21] L. GIRAUD, P. SPITERI, *Résolution parallèle de problèmes d'équations aux dérivées partielles sur une architecture à mémoire distribuée.* Rapport E.N.S.E.E.I.H.T.-I.R.I.T., 1989.

[22] C. A. R. HOARE, *Processus Séquentiels Communicants.* Masson, Paris, 1987.

[23] HOWARD, *Dynamic programming and Markov process.* M.I.T., 1960.

[24] C. JACQUEMARD, *Contribution à l'étude d'algorithmes de relaxation à convergence monotone.* Thèse 3e cycle, Université de Besançon, 1977.

[25] J. JULIAND, G. R. PERRIN, P. SPITERI, *Simulation d'exécutions parallèles d'algorithmes numériques asynchrones.* 1st Conference A.M.S.E., Lyon, 1981.

[26] L. LAMPORT, *The hyperplane method for an array computer.* Sagamore Computer Conference, 1974.

[27] P. L. LIONS, *Sur quelques classes d'équations aux dérivées partielles non linéaires et leur résolution numérique.* Thèse de Doctorat ès Sciences, Paris VI, 1979.

[28] J. C. MIELLOU, *Algorithmes de relaxation chaotiques à retards.* R.A.I.R.O., R-1, 55-82, 1975 et C.R.A.S., t. 278, pp. 957-960, 1974.

[29] J. C. MIELLOU, *Asynchronous iterations in order intervals.* Parallel algorithms & architectures, 85-96, North-Holland, Eds. M. Cosnard *and al.,* 1986.

[30] J. C. MIELLOU, P. SPITERI, *Un critère de convergence pour des méthodes générales de point fixe.* R.A.I.R.O. Modél. Math. Anal. Numér., 645-669, 1985.

[31] J. C. MIELLOU, Ph. CORTEY-DUMOND, M. BOULBRACHENE, *Perturbation of fixed point iterative methods.* Advances in parallel processing. Vol. 1, pp. 81-122, 1990.

[32] N. NIRSCHL, H. SCHNEIDER, *The Bauer fields of values of a matrix.* Numer. Math., *6,* 355-365, 1964.

[33] J. M. ORTEGA, W. C. RHEINBOLD, *Iterative solution of non linear equations in several variables.* Academic Press, 1970.

[34] F. ROBERT, *Discrete iterations.* Springer Series in Comput. Math., *6,* 1986.

[35] F. ROBERT, *Contraction en norme vectorielle : convergence d'itérations chaotiques.* Linear algebra and its applications, *13,* 19-35, 1975.

[36] F. ROBERT, M. CHARNAY, F. MUSY, *Itérations chaotiques série parallèle pour des équations non linéaires de point fixe.* Apl. Mat., *20,* 1-38, 1975.

[37] J. L. ROSENFELD, *A case study on programing for parallel processors.* I. B. M., Thomas J. Watson, Research Center Report, n° RC-64, U.S.A., 1967.

[38] P. SPITERI, *Simulation d'exécutions parallèles pour la résolution d'inéquations variationnelles stationnaires.* Revue E.D.F., série C, n° 1, 149-159, 1983.

[39] P. SPITERI, *Contribution à l'étude de grands systèmes non linéaires.* Thèse de Doctorat ès Sciences, Université de Besançon, 1984.

[40] P. SPITERI, *Parallel asynchronous algorithms for solving boundary value problems.* In Parallel Algorithms, Eds. M. Cosnard *and al.,* North-Holland, 73-84, 1986.

[41] K. TAUBERT, *Accretive operators with applications to numerical integration of ordinary differential equations.* Colloquia Mathematica Societatis Janos Bolyai. Numerical Methods, Miskolc, 211-225, 1986.