Une méthode de décomposition de domaines multifrontale multiniveaux

Yves Escaig* — Michel Vayssade — Gilbert Touzot***

** INSA de Rouen, Institut de mécanique*
BP 08, 76131 Mont-Saint-Aignan cedex

*** Université de technologie de Compiègne*
BP 649, 60206 Compiègne cedex

Travail effectué en collaboration entre l'Université de Compiègne et la division MMN de la Direction des Etudes et Recherches d'EDF dans le cadre d'une bourse cifre.

RÉSUMÉ. Le développement des ordinateurs multiprocesseurs a provoqué un regain d'intérêt pour les méthodes de décomposition de domaines. Le but de cet article est de présenter une méthode de décomposition de domaines multiniveaux pour la méthode des éléments finis avec résolution des problèmes interfaces à chaque niveau par une méthode directe. Pour réduire le coût de calcul des inverses des matrices de rigidité des sous-domaines, on utilise une méthode multifrontale. Après un bref rappel de la méthode du complément de Schur et son implantation classique, la méthode multifrontale est exposée en détail. Une comparaison entre ces deux implantations, en terme de nombre d'opérations théoriques et de temps d'exécution mesurés, est ensuite proposée. Dans la deuxième partie de l'article, l'influence de deux paramètres est étudiée : le nombre de sous-domaines et le nombre de niveaux de décomposition. Enfin, une comparaison entre la méthode de décomposition de domaines multifrontale proposée et une méthode sans décomposition est effectuée.

ABSTRACT. The developpement of multiprocessor computers has led to a new interest in domain decomposition methods. The goal of this article is to present a multilevel domain decomposition algorithm for the finite element method using a direct solver on the interface problems at each level. In order to reduce the cost of building the inverse of the rigidity matrices of each subdomain, a multifrontal method is used. After a brief description of the Schur complement method and its classical implementation, the multifrontal method is presented in detail. A comparison of theoretical number of operations and measured execution times between these two implementations is given. In the second part of the article, the influence of two parameters is studied : the number of subdomains and the number of decomposition levels. Finally, a comparison between the multifrontal domain decomposition method and a method without decomposition is provided.

MOTS-CLÉS : méthode des éléments finis, décomposition de domaines, décomposition multiniveaux, méthode multifrontale.
KEY WORDS : finite element method, domain decomposition, multilevel decompisition, multifrontal method.

1. Introduction

Il est aujourd'hui admis que l'avenir du calcul scientifique passera par l'utilisation des machines multiprocesseurs, et plus particulièrement des machines à architecture MIMD (Multiple Instruction Multiple Data) où la mémoire est physiquement répartie entre les divers processeurs. Or la programmation de ce type d'ordinateur est un véritable défi : il faut concevoir de nouvelles méthodes numériques qui contiennent le plus possible de parallélisme, tout en réduisant au maximum les communications entre les entités de calcul. Au vu de ces deux impératifs, pour la méthode des éléments finis, les stratégies de décomposition de domaines semblent particulièrement intéressantes.

En dehors des problèmes spécifiques liés à la parallélisation, les méthodes de décomposition de domaines offrent deux difficultés supplémentaires généralement reconnues : elles sont lourdes à mettre en œuvre [Debruyne 83] et elles présentent souvent des performances séquentielles moindres que des méthodes globales sans décomposition [Schrem 89]. Afin de proposer un système de modélisation par éléments finis, adapté aux machines multiprocesseurs grâce à l'utilisation des méthodes de décomposition de domaines et exploitable dans la pratique, il est nécessaire de surmonter ces deux difficultés.

Une solution aux problèmes de la difficulté d'utilisation a été proposée dans [Escaig 93]. Le but du présent article est de présenter les méthodes numériques qui ont été développées afin d'obtenir des méthodes de décomposition de domaines dont les performances soient au moins comparables et si possible meilleures que celles des méthodes sans décomposition, même dans le cas d'une utilisation séquentielle. On peut remarquer que de telles méthodes de décomposition de domaines auront en plus les avantages reconnus à cette classe de méthodes : parallélisation efficace, modularité, ...

Dans la suite de cet article, nous présentons tout d'abord deux méthodes de décomposition de domaines avec résolution directe du problème interface : la méthode classique des sous-structures et la méthode nouvelle que nous proposons. Puis, après une comparaison de ces deux méthodes, nous étudierons l'influence du nombre de sous-domaines sur les performances des méthodes numériques, ainsi que l'influence du nombre de niveaux de décomposition. L'implantation en parallèle des méthodes proposées dans le présent article est exposée dans [Escaig 94].

2. Les méthodes de décomposition de domaines

De manière très schématique, les méthodes de décomposition de domaines consistent à partager le domaine de départ W en un certain nombre de sous-domaines W_i. Ce découpage peut être effectué avec ou sans recouvrement des sous-domaines. Nous nous plaçons ici dans le cas où il n'y a pas recouvrement. Dans ce cas, le problème de départ peut être résolu de manière indépendante sur chaque sous-domaine, sous réserve de raccorder de manière adéquate les différentes solutions au travers des frontières entre sous-domaines. Pour effectuer ces raccordements, on peut envisager plusieurs stratégies. L'une d'entre elles consiste à calculer directement la valeur de la solution sur les frontières des sous-domaines à partir des informations recueillies au cours d'un calcul partiel des solutions sur l'intérieur des sous-domaines

(phase de condensation) : c'est la méthode bien connue du complément de Schur [Imbert 79]. Concrètement, si l'on choisit la méthode des éléments finis comme technique de discrétisation, on peut écrire le système d'équations linéaires final à résoudre K u = f sous la forme :

$$\begin{bmatrix} k_{11} & 0 & k_{13} \\ 0 & k_{22} & k_{23} \\ k_{31} & k_{32} & k_{33} \end{bmatrix} \begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix} = \begin{pmatrix} f_1 \\ f_2 \\ f_3 \end{pmatrix}$$

dans le cas où le domaine de départ est découpé en deux sous-domaines, et en renumérotant les équations de manière à faire apparaître d'abord les inconnues internes au premier sous-domaine puis les inconnues internes au second sous-domaine puis les inconnues se trouvant sur la frontière entre les sous-domaines. En utilisant les deux premières équations, on peut éliminer les inconnues u_1 et u_2 de la $3^{ième}$ équation et on obtient le système suivant :

$$\begin{bmatrix} k_{11} & 0 & k_{13} \\ 0 & k_{22} & k_{23} \\ 0 & 0 & S \end{bmatrix} \begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix} = \begin{pmatrix} f_1 \\ f_2 \\ \tilde{f}_3 \end{pmatrix}$$

$$[S] = [k_{33}] - [k_{31}][k_{11}]^{-1}[k_{13}] - [k_{32}][k_{22}]^{-1}[k_{23}]$$

$$\{\tilde{f}_3\} = \{f_3\} - [k_{31}][k_{11}]^{-1}\{f_1\} - [k_{32}][k_{22}]^{-1}\{f_2\}$$

Dans le cas d'un opérateur coercif, les blocs K_{ii} sont inversibles car ce sont les matrices associées au problème de départ avec des conditions aux limites de Dirichlet homogènes sur l'interface.

La matrice S est appelée matrice du complément de Schur. Le problème à résoudre sur l'interface, appelé problème global ou problème interface, s'écrit donc :

$$S\, u_3 = \tilde{f}_3 \text{ sur } \Gamma \qquad (1)$$

Pour résoudre ce problème global, on peut procéder de deux manières.

La première manière consiste à construire explicitement la matrice S et le second membre $\tilde{f}_3$, puis à résoudre de manière directe le système linéaire ainsi obtenu. C'est la stratégie qui est employée dans la méthode des sous-structures, bien connue en mécanique des solides.

Cependant, la construction explicite de la matrice S peut s'avérer très coûteuse car elle nécessite l'inversion des matrices K_{ii}. D'où l'idée d'utiliser, pour résoudre le problème (1), des méthodes itératives qui ne nécessitent pas le calcul explicite de la matrice S, mais uniquement celui du produit de cette matrice par des vecteurs [Le Tallec 90].

Pour la suite du travail présenté ici, nous avons choisi d'approfondir les méthodes directes. Ce choix peut paraître discutable, étant donné que, dans les méthodes directes, on calcule explicitement la matrice du complément de Schur. Ce cacul entraîne d'une part une augmentation du temps d'exécution, ainsi que la place mémoire nécessaire. Plusieurs raisons motivent ce choix.

Tout d'abord, les méthodes directes sont robustes, critère déterminant, surtout dans le cadre d'applications de mécanique des solides non linéaire où les matrices tangentes sont particulièrement mal conditionnées.

De plus, les méthodes directes sont applicables telles quelles à tous les types de problèmes et de géométries. Pour les méthodes itératives, le choix du préconditionneur optimal qui conduira aux meilleures performances est souvent lié à la nature du problème ou à la géométrie. Ce point est important en vue d'une utilisation industrielle.

Un autre avantage des méthodes directes est qu'elles permettent une estimation précise du temps calcul en fonction de la géométrie du problème alors que ceci est difficile et parfois impossible pour une méthode itérative. Là encore, dans un cadre industriel, cet avantage est important car il permet de décider d'effectuer ou non une modification ou une étude supplémentaire en fonction des coûts de calcul entraînés.

D'autre part, les méthodes directes permettent d'envisager l'utilisation de décompositions de domaines multiniveaux. Cette possibilité, théoriquement possible pour les méthodes itératives, semble peu envisageable dans la pratique. En effet, l'utilisation de solveurs itératifs pour les problèmes locaux dans le cas d'un seul niveau de décomposition apparaît déjà très inefficace [Roux 89]. Donc, pour des problèmes industriels comportant jusqu'à dix niveaux de décomposition, les méthodes itératives ne semblent pas bien adaptées. En revanche, il est toujours possible d'utiliser une méthode directe pour traiter les sous-domaines jusqu'à la racine, puis une méthode itérative pour le problème racine [Rogé 93].

Enfin, les méthodes directes permettent une plus grande modularité dans l'utilisation des méthodes de décomposition de domaines. En effet, il est possible, dans certains cas, de réutiliser des sous-domaines déjà calculés auparavant, ce qui peut conduire à la création de bibliothèques de sous-structures.

3. La méthode des sous-structures

La méthode des sous-structures, bien connue en mécanique des solides, reprend pas à pas la méthode du complément de Schur [Imbert 79]. L'enchaînement des opérations effectuées peut être résumé de la manière suivante, dans la cas où l'arbre des sous-structures ne comporte qu'un seul niveau de décomposition :

a - Condensation des sous-structures

pour chaque sous-structure faire :

1 - calcul et assemblage des matrices élémentaires dans les matrices de rigidité $[k_{ii}]$, $[k_{if}]$, $[k_{ff}]$

2 - factorisation de la matrice $[k_{ii}]$

3 - calcul du produit $[\Phi] = [k_{ii}]^{-1}.[k_{if}]$

pour ind = 1, nombre de nœuds frontières

résoudre $[k_{ii}] \cdot [\Phi]_{ind} = [k_{if}]_{ind}$

4 - calcul de la matrice de rigidité condensée

$$[\bar{k}_{ff}] = [k_{ff}] - [k_{fi}] [k_{ii}]^{-1} [k_{if}]$$

5 - calcul du second membre condensé

résoudre $[k_{ii}] \{q\} = \{fi\}$

calculer $\{\bar{f}_f\} = \{f_f\} - [k_{fi}] \{q\}$

6 - assemblage de la matrice $[\bar{k}_{ff}]$ barre dans la matrice de rigidité condensée globale $[\bar{K}]$

7 - assemblage du second membre condensé dans le second membre condensé global $[\bar{F}]$

b - Résolution du problème interface

1 - factorisation de la matrice $[\bar{K}]$

2 - résolution

c - Restitution des résultats

pour chaque sous-domaine faire

1 - calcul de $\{u_i\} = \{q\} + [\Phi] \cdot \{u_f\}$

Cette implantation, classique, de la méthode des sous-structures présente l'avantage de pouvoir être facilement dérivée d'une implantation existante de la méthode des éléments finis. En effet, à l'exception du module d'assemblage des matrices de rigidité $[k_{ii}]$, $[k_{if}]$, $[k_{ff}]$ qui est particulier, tous les autres modules sont

déjà présents. Cela permet donc de construire rapidement un système de sous-structuration dans un programme éléments finis. Néanmoins, cette méthode numérique présente un inconvénient majeur, à savoir le coût très important de l'étape d'inversion de la matrice $[k_{ij}]$, coût que l'on ne retrouve pas dans la méthode des éléments finis sans décomposition. Les premiers résultats de performances le confirment. Ces premières mesures ont été obtenues en résolvant l'équation de Poisson en 2D, sur un carré discrétisé régulièrement à l'aide de triangles à trois nœuds (élément P1). Dans cet exemple, le domaine carré a été découpé en quatre sous-structures identiques, mais cette identité n'a pas été exploitée lors de la résolution afin de pouvoir faire des comparaisons avec une méthode sans sous-structuration. Les deux méthodes, avec et sans sous-structuration, sont programmées de la manière la plus proche possible pour faciliter les comparaisons. Elles utilisent toutes les deux le même gestionnaire de mémoire, les mêmes calculs élémentaires et le même solveur (algorithme de Choleski). Les résultats, obtenus sur une station de travail IBM RS/6000, sont indiqués dans la figure 1.

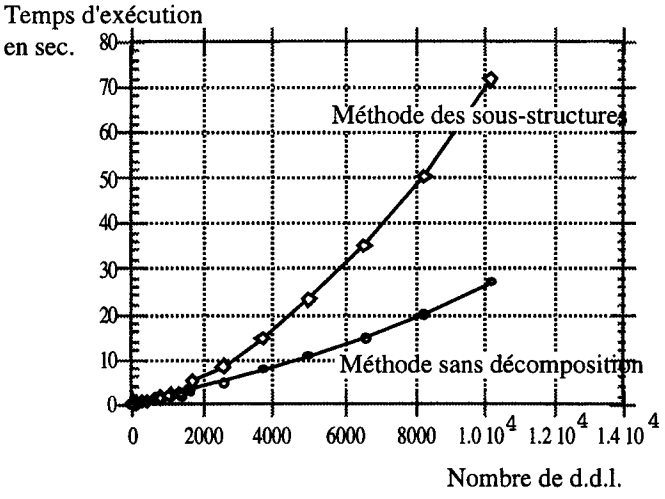


Figure 1. *Comparaison entre la méthode classique des sous-structures et une méthode sans décomposition*

Si l'on mesure plus finement le temps d'exécution des différentes étapes de l'algorithme, on s'aperçoit que l'étape d'inversion de la matrice $[k_{ij}]$ consomme environ 75% du temps total de résolution. Donc, afin de rendre compétitives les méthodes de décomposition de domaines avec résolution directe du problème interface, il est nécessaire de réduire, voir de faire disparaître, cette étape d'inversion explicite de la matrice $[k_{ij}]$. Pour cela, nous proposons d'utiliser une méthode multifrontale.

4. La méthode multifrontale

Rappels sur la méthode frontale

Introduite par Irons en 1969 [Irons 69], la méthode frontale est une méthode qui permet de tenir compte de la topologie des équations issues de la méthode des éléments finis, afin de réduire l'encombrement mémoire. Pour cela, les étapes d'assemblage et d'élimination sont imbriquées.

Au cours du processus d'assemblage de la méthode des éléments finis, à un instant donné, seuls quelques nœuds sont actifs, c'est-à-dire que ces nœuds sont déjà apparus lors de l'assemblage d'un élément, et qu'ils apparaîtront plus tard dans d'autres éléments. Ces nœuds actifs constituent, à un instant donné, ce que l'on appelle un front. En revanche, les autres nœuds soit ne sont pas encore apparus, soit ont déjà été totalement assemblés et ont pu être déjà éliminés. Il n'est donc pas nécessaire de leur allouer une ligne et une colonne dans la matrice de rigidité globale. Dans la méthode frontale, dès qu'un degré de liberté n'est plus actif, il est éliminé de la matrice de rigidité, appelée matrice frontale car elle ne contient que les nœuds actifs présents dans le front. On comprend donc que la taille de la matrice frontale va "osciller", c'est-à-dire tantôt augmenter avec l'assemblage des degrés de liberté de nouveaux nœuds actifs, tantôt diminuer avec l'élimination des degrés de liberté de nœuds qui ne sont plus actifs.

Afin d'éliminer un degré de liberté de la matrice frontale, on effectue une étape de la méthode de Gauss (ou de Choleski) sur cette matrice frontale en prenant comme ligne pivot, la ligne du degré de liberté à éliminer.

Ensuite, on retire cette ligne de la matrice et on la sauvegarde dans une structure de données spéciale, que l'on peut soit conserver en mémoire centrale, soit plus généralement dans une mémoire secondaire, sur disque par exemple.

A la fin de ce processus, lorsque tous les éléments ont été assemblés, et tous les degrés de liberté éliminés, la matrice de rigidité globale factorisée se trouve dans la structure de données de sauvegarde. Il ne reste plus qu'à parcourir celle-ci, dans l'ordre inverse où les degrés de liberté ont été éliminés, afin d'obtenir la solution du système.

La méthode frontale apparaît comme la méthode la plus souple en ce qui concerne l'ordre d'élimination des variables. Il est en effet très facile de décider à quel moment telle ou telle variable doit être éliminée. Il suffit pour cela de la conserver dans la matrice frontale, et de retarder son élimination.

De plus, compte tenu de la taille limitée de la matrice frontale, elle s'adapte très facilement à la taille de mémoire centrale disponible, en évitant le découpage des matrices en blocs et donc de nombreux mouvements d'entrée/sortie. Il est possible de mettre en place des stratégies de choix des fronts qui consistent à conserver des d.d.l. dans le front plus longtemps que nécessaire avant de les éliminer pour occuper ainsi toute la mémoire disponible.

Enfin, comme nous le verrons dans le paragraphe suivant, la méthode frontale s'adapte très bien aux méthodes de sous-structuration, sous la forme de la méthode que nous appelons multifrontale.

La méthode multifrontale

Dans la littérature, on utilise le nom de méthode multifrontale pour désigner deux méthodes numériques proches, mais néanmoins différentes. En effet, pour [Duff] par exemple, la méthode multifrontale est utilisée afin de factoriser une matrice, souvent creuse, déjà existante. On utilise pour cela un arbre d'élimination afin de créer des fronts de variables pouvant être éliminées en parallèle. Cette approche consiste uniquement à dégager un ordre particulier d'élimination des variables. Elle est complètement déconnectée de la méthode des éléments finis, et peut être utilisée pour n'importe quelle matrice.

Pour notre part, dans la suite du document, nous appellerons méthode multifrontale une méthode qui couple la méthode frontale décrite au paragraphe précédent, et une méthode de décomposition de domaine.

Du point de vue du principe de l'algorithme, la méthode multifrontale est identique à la méthode des sous-structures c'est-à-dire qu'il y a condensation des matrices de rigidité, assemblage des matrices condensées, résolution pour les nœuds de liaison puis restitution sur les nœuds internes. La différence apparaît dans la manière d'effectuer ces calculs.

L'idée de base de la méthode multifrontale utilisée est d'appliquer la méthode frontale à chaque sous-structure, en assemblant normalement tous les éléments de la sous-structure, c'est-à-dire en insérant dans la matrice frontale des nœuds internes et des nœuds frontières. En revanche, on s'interdit d'éliminer les degrés de liberté (d.d.l.) attachés aux nœuds frontières. Donc, à la fin du processus d'assemblage-élimination au sein de chaque sous-structure, il ne reste plus dans la matrice frontale que les termes correspondant aux degrés de liberté portés par les nœuds frontières. Or, ces équations ont été modifiées par l'élimination successive de tous les degrés de liberté internes de la sous-structure. Ceci correspond exactement au processus de condensation. On retrouve cette idée par exemple dans [Keunings 92].

La matrice frontale qui reste à la fin du processus d'assemblage-élimination est donc exactement la matrice de rigidité condensée calculée dans la méthode des sous-structures (matrice $\left[\bar{k}_{ff}\right]$ de l'étape a-4).

Les étapes de la condensation des sous-structures sont donc rigoureusement les mêmes que celles d'une méthode frontale classique. Les phases explicites d'inversion de la matrice $[k_{ii}]$, de calcul de la matrice [F] et de la matrice de rigidité condensée ont donc complètement disparu. On peut donc espérer obtenir de bien meilleures performances avec la méthode multifrontale qu'avec la méthode des sous-structures présentée précédemment.

5. Comparaison entre la méthode des sous-structures et la méthode multifrontale

La méthode des sous-structures et la méthode multifrontale peuvent toutes les deux être vues comme deux implantations différentes de la méthode d'élimination de Gauss par blocs [Imbert 79]. Elles conduisent donc aux mêmes résultats numériques, bien que les opérations effectuées soient différentes. On peut voir cela

comme le déroulement du même algorithme sur deux permutations d'une même matrice.

Dans la méthode des sous-structures, les opérations effectuées sur les inconnues qui correspondent à des degrés de liberté portés par des nœuds intérieurs à la sous-structure, que nous appellerons simplement inconnues internes dans la suite de cet article, sont complètement séparées de celles effectuées sur les inconnues correspondant à des degrés de liberté portés par des nœuds de la frontière de la sous-structure, que nous appellerons simplement inconnues frontières dans la suite de cet article. En effet, on commence par factoriser la matrice de rigidité qui correspond aux inconnues internes, puis de manière séparée, on répercute ces opérations sur la matrice de rigidité des inconnues frontières, en utilisant pour cela la matrice de couplage $[k_{if}]$.

Dans la méthode multifrontale, les opérations effectuées sur les inconnues internes et les inconnues frontières ne sont plus séparées, mais effectuées en même temps au cours du processus d'assemblage-élimination frontal. Ceci permet de faire disparaître la matrice de couplage. Il n'est donc plus nécessaire de calculer explicitement le produit $[k_{ii}]^{-1}[k_{if}]$, ce qui évite la résolution, très coûteuse de *nfr* systèmes linéaires, où *nfr* est le nombre d'inconnues frontières. Ces calculs sont faits implicitement puisque les inconnues internes et les inconnues frontières sont présentes simultanément dans le système frontal.

On peut donc s'attendre à de meilleures performances pour la méthode multifrontale que pour la méthode des sous-structures. Pour le vérifier, un modèle théorique du nombre d'opérations effectuées par chacune des méthodes a été développé.

Nombre d'opérations de la méthode des sous-structures

Soient les notations suivantes :

- lb : largeur de bande de la matrice k_{ii}
- ntr : nombre d'inconnues internes
- nfr : nombre d'inconnues frontières
- a : facteur de remplissage de la matrice de couplage $[k_{if}]$ (typiquement a = 0.01)

Si on ne considère que l'étape de condensation de la matrice de rigidité de la sous-structure par la méthode des sous-structures, les opérations effectuées sont :

- la factorisation de la matrice de rigidité $[k_{ii}]$. Cette matrice a pour dimension (ntr × ntr) avec une largeur de bande lb. La factorisation requiert *ntr* étapes de l'algorithme de Gauss, soit :

$$N_{fac} = ntr\ (2 \quad lb^2)$$

• le calcul du produit $\Phi = [k_{ii}]^{-1} [k_{if}]$ (étape a-3), ce qui revient à *nfr* résolutions de systèmes linéaires. Une étape de descente nécessite (2 ntr lb) opérations, soit au total :

$$N_M = \text{ntr } (4 \text{ nfr lb})$$

• le calcul du produit matriciel $[k_{fi}]$ F (étape a-4), soit (2 ntr^2 nfr) opérations. Mais la matrice $[k_{fi}]$ étant creuse, chaque ligne ne contient que (a ntr) termes non nuls qui correspondent aux nœuds internes voisin des nœuds frontières. D'où :

$$N_p = \text{ntr } (2\ \alpha^2\ \text{nfr ntr})$$

Ceci donne un coût total pour la condensation par la méthode des sous-structures de :

$$N_{\text{sous-structure}} = \text{ntr } (2 \quad lb^2 + 4 \text{ nfr lb} + 2 \quad \alpha^2 \text{ nfr ntr}) \tag{2}$$

Nombre d'opérations pour la méthode multifrontale

Dans la méthode multifrontale, la condensation de la matrice de rigidité est totalement effectuée au cours du processus d'assemblage-élimination frontal, en même temps que la factorisation de la matrice de rigidité des inconnues internes.

Soit *lfrt* la largeur frontale maximale. Pour une matrice pleine de dimension *lfrt*, une étape de l'algorithme de Gauss requiert $2.lfrt\ (lfrt - 1) \approx 2\,lfrt^2$ opérations. On élimine toutes les inconnues internes, ce qui représente *ntr* étapes de l'algorithme de Gauss. D'où :

$$N_{mf} = \text{ntr } (2. \quad lfrt^2) \tag{3}$$

Cette valeur N_{mf} est une borne supérieure du nombre d'opérations. En réalité, le nombre d'opérations de la méthode multifrontale dépend de la largeur de front instantanée *lfrinst* à chaque étape de l'algorithme. Pour prendre ceci en compte, on utilise dans les formules précédentes non plus la largeur frontale maximale *lfrt* mais une largeur frontale moyenne *lfrm*, que l'on peut définir à partir des largeurs frontales instantanées par [Sloan 89] :

$$lfrm = \sqrt{\frac{\sum_{i=1}^{N} lfrinst_i^2}{N}} \tag{4}$$

Cette moyenne, qui correspond à une moyenne quadratique, se justifie dans notre cas car elle entraîne un décompte exact du nombre d'opérations si celui-ci est estimé par la formule :

$$N_{op} = N\ lfrm^2$$

En effet, le coût exact de l'élimination en fonction de la largeur frontale instantanée s'écrit :

$$N_{ex} = \sum_{i=1}^{n} lfrinst_i^2$$

On obtient donc la formule suivante pour le nombre d'opérations de la méthode multifrontale :

$$N_{mf} = ntr\ (2.\ \ lfrm^{\ 2}) \qquad (5)$$

Pour une géométrie quelconque, les variables *lb*, *lfrt*, *lfrm*, *nfr* et *a* ne sont pas calculables à priori. En revanche, dans le cas d'une géométrie simple, maillée régulièrement, ces variables peuvent être estimées, ce qui permet d'étayer la validité des formules précédentes. Comme pour la comparaison entre la méthode des sous-structures et une méthode sans décomposition, on considère le problème de la plaque carrée découpée en quatre sous-domaines. Les trois courbes du nombre d'opérations effectuées en fonction du nombre de d.d.l. pour la méthode des sous-structures (Ns-s), la méthode multifrontale avec la largeur frontale maximale (Nmf-max)(3) la méthode multifrontale avec la largeur frontale moyenne (Nmf-moy)(5) sont présentées sur la figure 2.

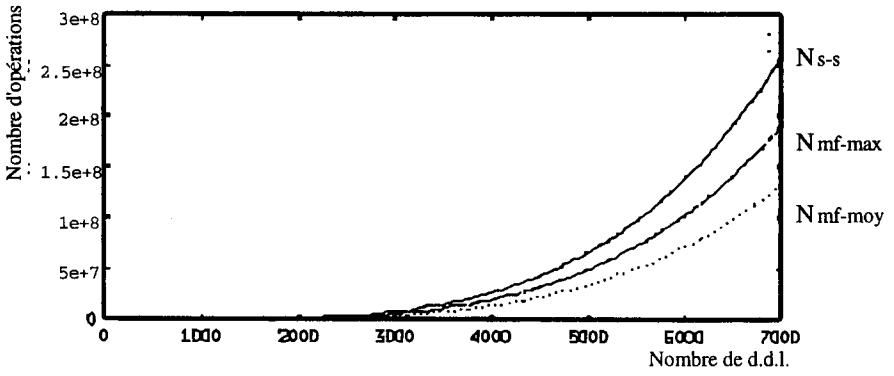


Figure 2. *Courbes du nombre d'opérations estimées des méthodes des sous-structures et multifrontale*

Bien qu'il soit très difficile de relier directement le nombre d'opérations arithmétiques au temps calcul, il est néanmoins possible de les comparer qualitativement. La figure 3 indique les temps calculs obtenus pour les deux méthodes sur un superordinateur Cray Y-MP/4-128. Ces résultats ont été obtenus pour l'application entière, et contiennent donc d'autres opérations non comptabilisées ici, mais qui sont les mêmes pour les deux méthodes.

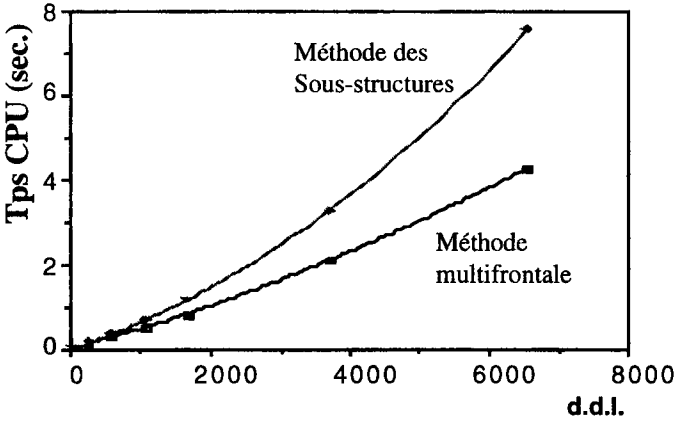


Figure 3. *Temps calcul d'une résolution complète pour les deux méthodes en fonction de d.d.l.*

On peut remarquer que les courbes des temps calcul et des nombres d'opérations évoluent sensiblement de la même manière.

Analyse des résultats

Dans la méthode multifrontale, la complexité de l'algorithme est pilotée par la largeur frontale moyenne *lfrm*. Or, nous avons vu que, dans la matrice frontale, étaient présentes simultanément les inconnues frontières et les inconnues internes actives à ce moment de l'algorithme. Soit *lfmi* la largeur frontale moyenne (respectivement *lfrti* la largeur frontale maximale) due aux inconnues internes, indépendamment des inconnues frontières. On peut donc écrire :

$$\text{lfrm} < \text{nfr} + \text{lfmi} \quad \text{et} \quad \text{lfrt} < \text{nfr} + \text{lfrti}$$

Si on utilise la valeur limite de lfrm égale à nfr + lfmi dans la formule (5), on obtient :

$$N_{mf} = \text{ntr}\,(2\ \ \text{lfmi}^2 + 4\ \text{nfr lfmi} + 2\ \ \text{nfr}^2)$$

On constate que cette formule est équivalente à celle du nombre d'opérations de la méthode des sous-structures, sous réserve que *lfmi* et *lb* soient comparables. Or, il a été montré [Sloan 89] que *lfmi* et *lb* étaient du même ordre de grandeur. Donc, dans le cas où lfrm = nfr+lfmi, la méthode multifrontale et la méthode des sous-structures, exécutent le même nombre d'opérations.

Mais, si au cours du processus d'assemblage-élimination frontal, les inconnues frontières n'apparaissent qu'à la fin, les inconnues internes peuvent prendre la place des inconnues frontières dans la matrice frontale. A la limite, on a :

$$lfrm = lfmi \quad et \quad lfrt = nfr$$

On peut donc donner les encadrements de *lfrm* et *lfrt* pour la méthode multifrontale :

$$lfmi < lfrm < nfr + lfmi$$
$$nfr < lfrt < nfr + lfrti$$

Interprétation qualitative

Dans le cas d'une équation aux dérivées partielles elliptique, la solution en n'importe quel endroit du domaine dépend de l'ensemble des inconnues du domaine. Lorsque le domaine est découpé en éléments finis, il est possible de faire diffuser cette influence à tout le domaine étape par étape, en remarquant qu'un noeud du maillage n'est directement influencé que par ses voisins : c'est le principe de localité de la méthode des éléments finis. Cette diffusion étape par étape est effectuée lors de la résolution du système d'équations linéaires final, ce système d'équations linéaires ayant une structure creuse particulière. C'est également ce qui se passe avec les fronts successifs de la méthode frontale.

Au cours du déroulement de la méthode multifrontale, les inconnues frontières ne sont pas éliminées. En revanche, les lignes de la matrice frontale qui leur correspondent sont modifiées par l'élimination de toutes les inconnues internes. Ces lignes "portent" donc toute l'influence de la sous-structure. C'est le phénomène de condensation. Si les inconnues frontières sont assemblées en dernier dans la matrice frontale, les lignes de la matrice qui leur correspondent vont être uniquement modifiées par l'élimination des inconnues internes présentes dans la matrice à ce moment, ce qui préserve le caractère local de la méthode, et donc la structure creuse du système d'équations linéaires. En revanche, si les inconnues frontières sont présentes dès le début dans la matrice frontale, toutes les inconnues internes vont venir directement modifier les lignes qui correspondent aux inconnues frontières. On obtient évidement les mêmes résultats que précédemment, mais on a par ce fait perdu ce principe de localité pour les inconnues frontières, et donc la structure creuse. Ceci entraîne un nombre important d'opérations supplémentaires.

Dans la méthode des sous-structures, on commence par factoriser un système d'équations linéaires correspondant aux inconnues internes, d'une manière classique, qui respecte la localité des opérations et donc la structure creuse de la matrice. Puis on effectue le produit suivant :

$$M = [k_{fi}][k_{ii}]^{-1}[k_{if}]$$

Ceci peut être interprété comme la diffusion de l'influence de toutes les inconnues internes sur les inconnues frontières. Cette opération viole donc le principe de localité décrit précédemment et s'avère donc très coûteuse. C'est la raison pour laquelle la méthode des sous-structures est moins performante que la méthode multifrontale, sous réserve que, pour la méthode multifrontale, les éléments soient éliminés dans un ordre optimal. Afin de générer cet ordre optimal, un optimiseur de largeur de bande multifrontale a été développé [Escaig 92].

6. Influence du nombre de sous-structures

Lorsque l'on utilise une méthode de décomposition de domaines, le problème du choix du découpage en sous-domaines apparaît immédiatement. Si ce découpage n'est pas guidé de manière naturelle par la physique du problème, l'utilisateur est confronté au dilemme suivant : faut-il créer beaucoup de petits sous-domaines, ou très peu de gros sous-domaines. En d'autres termes, quelle est la taille optimale à donner aux sous-domaines ?

Dès qu'il est question d'optimalité, il est nécessaire de définir le critère d'optimalité, ainsi que les contraintes imposées au système.

Dans notre cas, la seule contrainte qui s'exerce sur l'utilisateur est celle de la place mémoire disponible. En effet, la taille des sous-domaines doit être telle que leur calcul puisse s'effectuer en mémoire centrale. Nous ferons l'hypothèse ici que le problème peut être entièrement calculé en mémoire centrale, quel que soit le nombre de sous-domaines. Si tel n'était pas le cas, ceci imposerait une borne inférieure au nombre de sous-domaines.

Différents critères d'optimalité peuvent être envisagés. On peut penser à l'équilibrage des tâches lors d'une exécution en parallèle. On peut également essayer d'optimiser la place mémoire. Nous avons choisi dans cette analyse le temps d'exécution, qui est directement lié au nombre d'opérations.

La méthode des sous-structures est composée de deux parties consommatrices de temps machine : la condensation des sous-structures et le calcul des inconnues frontières. La partie de restitution de la valeur des inconnues frontières demande un temps d'exécution négligeable.

Les phases de condensation et de calcul des inconnues frontières agissent à la manière de vases communiquants : si on augmente le nombre de sous-structures, on diminue le temps global de condensation, mais on augmente le nombre d'inconnues frontières, et donc le temps de leur calcul.

Voyons tout d'abord comment évolue le temps de condensation en fonction du nombre de sous-domaines.

D'après (5), le nombre d'opérations effectuées lors de la condensation d'une sous-structure par la méthode multifrontale est :

$$N_{mf} = ntr\ (2.\ \ lfrm^{2})$$

Cette formule est linéaire en *ntr*. Donc, le fait de diminuer le nombre des nœuds internes de chaque sous-structure sera exactement compensé par l'augmentation du nombre de sous-structures. En revanche, le nombre d'opérations diminue avec le carré de la largeur frontale moyenne. Tout va donc dépendre de la diminution de la largeur frontale moyenne, due à une augmentation du nombre des sous-domaines.

La deuxième composante significative du temps d'exécution dans les méthodes de décomposition de domaines est la résolution du système d'équations linéaires final correspondant aux inconnues frontières. La matrice associée à ce système est pleine, ce qui implique un nombre d'opérations, si *nlng* est la taille de la matrice :

$$N_{fac} = \frac{2\ nlng^{3}}{3}$$

Ce nombre d'opérations est toujours croissant avec la taille du système. Le nombre total d'opérations pour la méthode multifrontale, en utilisant la largeur frontale moyenne, est donc :

$$N_{mf} = \sum_{i=1}^{nss} 2\ ntr_i\ lfrm_i^2 + \frac{2\ nlng^3}{3} \qquad (6)$$

Afin de valider ce modèle, on peut choisir un problème avec une géométrie simple sur laquelle on va pouvoir déterminer a priori la valeur des différents paramètres. On considère donc une plaque carrée bi-dimensionnelle, maillée régulièrement à l'aide de triangles à 3 nœuds, sur laquelle on résout l'équation de Laplace.

Découpage en bandes

Cette plaque est découpée en bandes afin de former les sous-domaines. Pour cet exemple, on obtient les courbes du nombre d'opérations en fonction du nombre de sous-domaines indiqués sur la figure 4.

La courbe du nombre total d'opérations étant strictement croissante, cela veut dire que la diminution de la largeur frontale ne compense à aucun moment l'augmentation du nombre d'opérations due aux inconnues frontières. On peut expliquer cela par la forme des sous-domaines. Ils sont très allongés, donc le pourcentage d'inconnues frontières par rapport aux inconnues internes est grand. On peut juste observer une croissance plus faible du nombre total d'opérations entre 2 et 10 sous-structures. Les parts respectives de la condensation et de la résolution du problème frontière sont également présentées sur la figure 4.

Comparons maintenant ces résultats avec l'expérience : la figure 5 montre l'évolution des temps de calcul en fonction du nombre de sous-structures, pour un problème à 3721 d.d.l. Les tests ont été réalisés sur une station SUN Sparc 1.

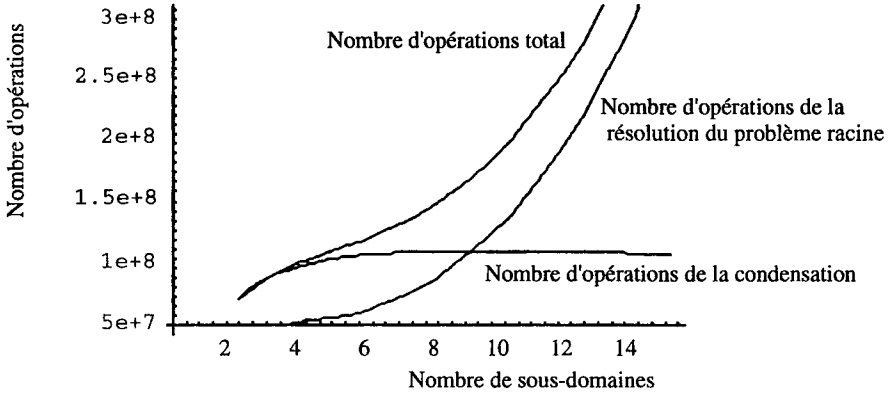


Figure 4. *Evolution du nombre d'opérations estimées en fonction du nombre de sous-domaines*

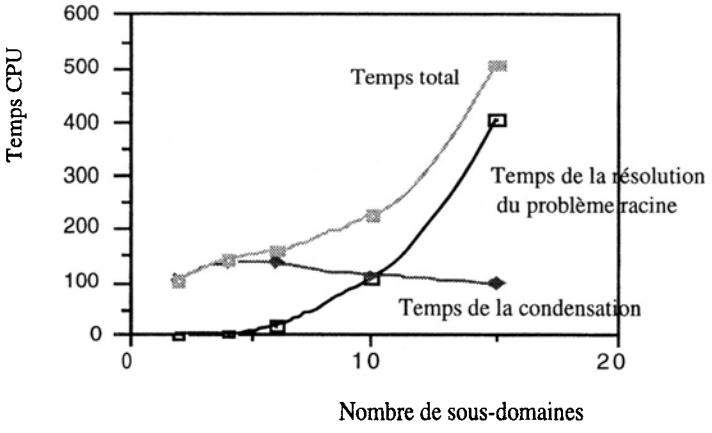


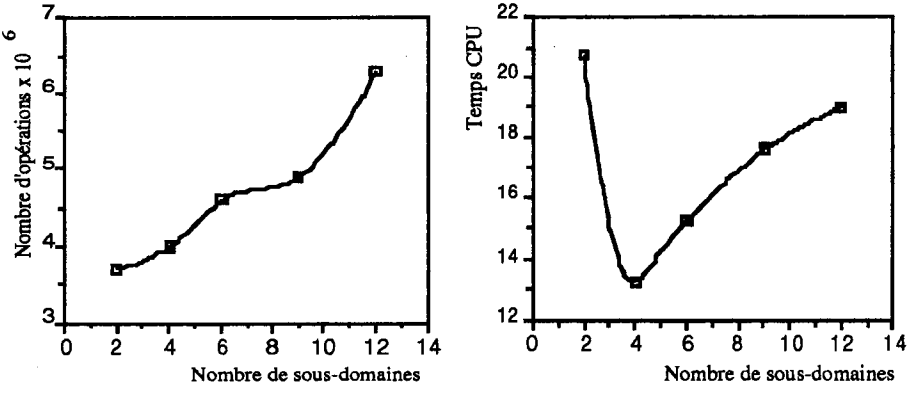
Figure 5. *Evolution du temps d'exécution en fonction du nombre de sous-domaines*

Qualitativement, les courbes des figures 4 et 5 sont très similaires, ce qui tend à valider les modèles proposés.

Découpage en carrés

Le découpage en bandes que nous venons de voir n'entraîne pas de réduction du nombre d'opérations. Ceci est dû au grand nombre de nœuds frontières crée par ce découpage. Pour remédier à cela, on effectue maintenant un découpage en carrés de la plaque.

En faisant le même type de calculs qu'au paragraphe précédent, on obtient une courbe du nombre d'opérations en fonction du nombre de sous-structures (cf figure 6-a).



(a) Décompte du nombre d'opérations en utilisant la largeur frontale maximum

(b) Temps CPU effectivement obtenus

Figure 6. *Evolution du nombre d'opérations estimées et du temps d'exécution en fonction du nombre de sous-domaines pour le découpage en carrés*

En revanche, les temps de calcul observés sont très différents (voir la figure 6-b). Cette différence est due à l'utilisation de la largeur frontale maximale à la place de la largeur frontale moyenne. En effet, la largeur frontale maximale est au moins égale au nombre d'inconnues frontières. Or, ce nombre d'inconnues frontière reste constant par sous-domaines, que la plaque soit découpée en deux ou quatre sous-domaines. On n'observe donc pas de décroissance dans la courbe du nombre d'opérations théorique si on utilise la largeur frontale maximale. En revanche, la largeur frontale moyenne diminue avec le découpage en sous-domaines. L'évolution de la largeur frontale instantanée au cours de la condensation d'un sous-domaine est donnée sur la figure 7 pour un découpage en deux et en quatre sous-domaines de la plaque de dimension n = 60.

Si on utilise dans les équations du modèle les valeurs expérimentales de la largeur frontale moyenne, calculée à l'aide de la formule (4), on retrouve l'allure de la courbe 6-b, c'est-à-dire une décroissance du nombre d'opérations entre 2 et 4 sous-domaines, puis une remontée entre 4 et 12 (cf figure 8).

Le problème de la modélisation plus fine de la largeur frontale moyenne reste ouvert, particulièrement dans le cas de géométries complexes. Ceci est notamment important pour l'estimation a priori du temps d'exécution d'un calcul. Néanmoins, on peut envisager d'effectuer une condensation symbolique, donc peu coûteuse, des sous-domaines pour déterminer précisément la largeur frontale moyenne.

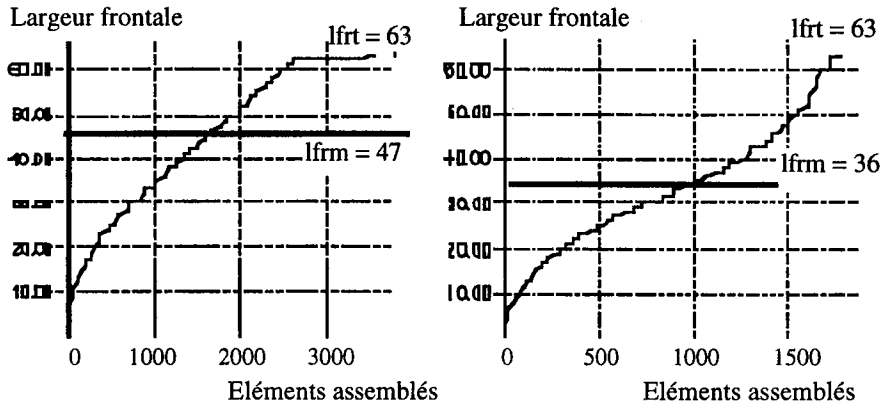


Figure 7. *Evolution de la largeur frontale instantanée d'un sous-domaine pour un découpage en deux et quatre sous-domaines*

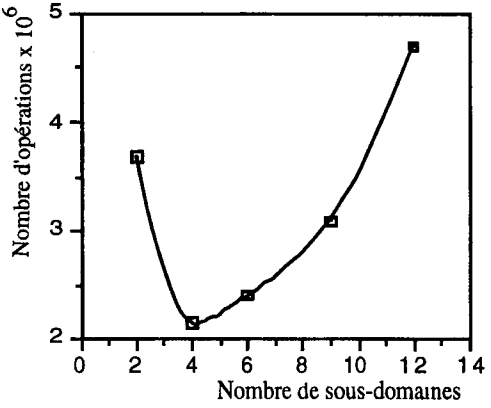
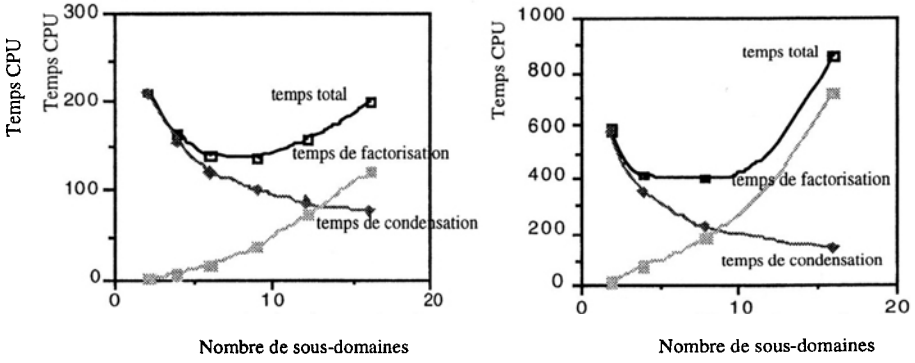


Figure 8. *Evolution du nombre d'opérations en fonction du nombre de sous-domaines en tenant compte de la largeur frontale moyenne*

L'exemple précédent montre que, dans certains cas de découpage, on observe un minimum dans la courbe d'évolution du temps CPU en fonction du nombre de sous-domaines. Pour vérifier cela, nous avons effectué deux autres séries de tests, en faisant varier le nombre de sous-structures. La première est basée sur une plaque rectangulaire, maillée régulièrement à l'aide de quadrangles Q8 sur laquelle on résout un problème d'élasticité linéaire à 2 degrés de liberté par nœud. La plaque est découpée en sous-domaines carrés. La seconde est une poutre tridimensionnelle maillée à l'aide de briques H8, découpée en sous cubes. Les résultats de ces deux séries de tests, effectués sur une station de travail SUN Sparc 1, sont donnés sur la figure 9.



(a) Plaque 2D 24 x 24 (3650 d.d.l.)

(b) Poutre 3D 8 x 8 x 8 (2187 d.d.l.)

Figure 9. *Evolution des temps d'exécution en fonction du nombre de sous-domaines avec un découpage en carrés et en cubes*

On retrouve la présence d'un minimum pour les deux courbes. Pour le cas 3D, la courbe ne comporte que 4 points car la plus petite discrétisation qui permettrait un découpage en 2, 4, 8, 12, 16, 24 sous-domaines devrait avoir au moins 12 éléments finis dans chaque direction, ce qui conduit à un problème de taille trop importante.

D'autre part, on peut noter que la courbe croît de manière plus forte dans le cas 3D que dans le cas 2D. Les deux courbes supplémentaires du temps de condensation et de factorisation de la matrice du sous-domaine racine permettent de l'expliquer. En effet, on voit que les deux courbes des temps de condensation se comportent de la même manière, alors que le temps de factorisation dans le cas 3D augmente beaucoup plus vite. Ceci est dû aux interfaces 2D entre les sous-domaines 3D, qui font croître plus vite la taille du problème racine que dans le cas 2D aux interfaces 1D.

Les deux types de découpage précédents en carrés et en cubes sont optimisés quant à la taille de l'interface et à la forme des sous-domaines. Si l'on étudie les performances obtenues avec l'algorithme de découpage automatique, on observe un comportement semblable, bien que la décroissance soit moins forte (cf figure 10). Cela est dû au plus grand nombre de nœuds frontières engendrés par le découpage automatique. Ceci montre que l'approche avec découpage automatique, transparente pour l'utilisateur, peut réduire le nombre total d'opérations, et ceci pour un effort nul de la part de l'utilisateur.

Tous les cas présentés ici ont été calculés sur une machine à processeurs scalaires, une station de travail SUN Sparc 1. Dans le cas de processeurs vectoriels, le temps de résolution du problème frontière est diminué car la matrice associée est pleine, ce qui permet d'obtenir de très longs vecteurs, et donc des performances vectorielles très élevées. Par exemple, sur le cas 3D à 6591 d.d.l., et sans optimisation particulière, on obtient 160 Mflops sur un Cray Y-MP. Le temps d'exécution de la phase de factorisation aura donc tendance à augmenter moins vite que sur une machine à processeurs scalaires. La figure 11 présente l'évolution du

temps de factorisation pour le problème de la plaque 2D et du cube 3D, calculés sur un ordinateur Cray Y-MP.

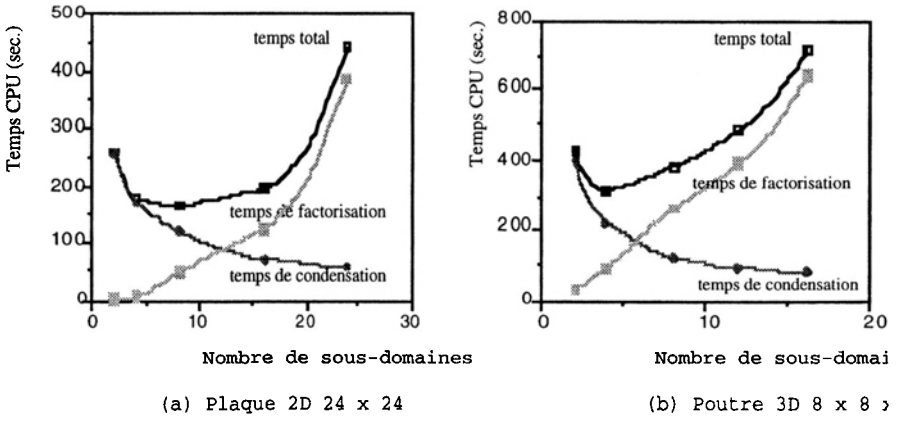


Figure 10. *Evolution des temps d'exécution en fonction du nombre de sous-domaines avec un découpage automatique*

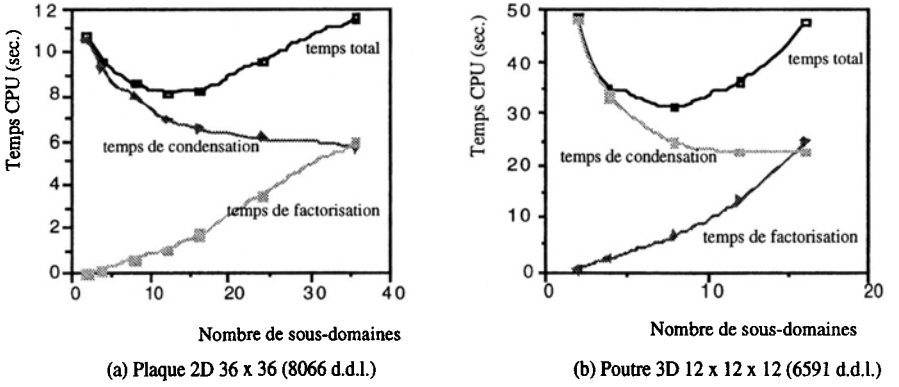


Figure 11. *Evolution des temps d'exécution en fonction du nombre de sous-domaines avec un découpage en carrés et en cubes sur un processeur vectoriel*

7. Découpage multiniveaux

L'une des possibilités les plus intéressantes qu'offrent les méthodes de décomposition de domaines est sans doute la création récursive de sous-domaines à partir d'autres sous-domaines et non plus uniquement d'éléments finis. Dans ce cas, un problème est représenté par un arbre de sous-domaines. Cette possibilité de décomposition multiniveaux permet d'une part de rendre la méthode encore plus

modulaire, et donc de traiter des problèmes à géométrie très complexe. D'autre part, cela permet également de réduire de manière arbitraire la taille des problèmes interfaces que l'on doit résoudre.

Afin de traiter un problème qui comporte plusieurs niveaux de décomposition, on applique de manière récursive la méthode de condensation décrite précédemment, à savoir dans le cas de l'arborescence de la figure 12 :

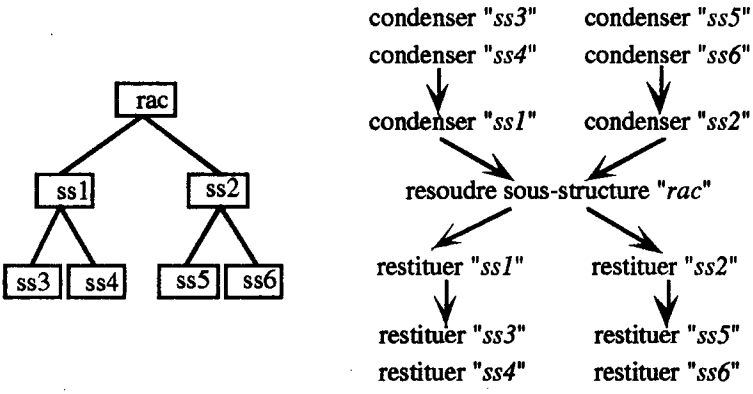


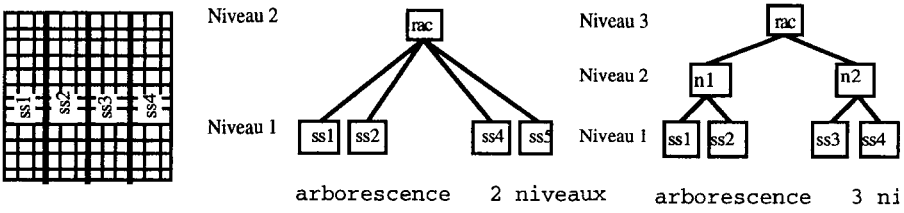
Figure 12. *Exemple de calcul par sous-domaines : détail des étapes*

Les matrices de rigidité condensées sont considérées comme les matrices élémentaires des sous-domaines que l'on va assembler afin de former la matrice de rigidité des sous-domaines de niveau supérieur, qui à leur tour seront condensées. Pour effectuer cette étape de condensation des sous-domaines de niveau supérieur à un, on choisit la méthode multifrontale, pour les raisons exposées au paragraphe précédent. En dehors de la notion de connectivité d'un élément fini qui devient l'ensemble des numéros des nœuds qui composent les sous-domaines de niveau supérieur, toutes les étapes d'assemblage et d'élimination frontale peuvent être reprises dans le cadre d'un sous-domaine (parfois appelé super-élément). La principale difficulté qui apparaît lorsque l'on utilise des décompositions multiniveaux réside dans la gestion des différentes numérotations des nœuds qui sont susceptibles d'appartenir à plusieurs sous-domaines de plusieurs niveaux. Cet aspect des choses est traité dans [Escaig 93]. La gestion automatisée des différents niveaux de l'arborescence des sous-domaines proposée dans cette référence qui a été effectivement mise en place, nous a permis d'étudier de manière simple l'influence du nombre de niveaux dans l'arbre des sous-domaines sur les temps de résolution du problème.

Dans la partie précédente, nous avons étudié l'influence du nombre de sous-domaines sur le nombre total d'opérations effectuées, et nous avons montré que, dans certains cas, ce découpage entraîne une réduction du nombre d'opérations. Mais, à partir d'une certaine limite, ce découpage conduit à un plus grand nombre d'opérations à cause de la taille du problème frontière. Afin de remédier à cela, on peut appliquer de nouveau un découpage au problème frontière en créant plusieurs niveaux dans l'arborescence des sous-domaines. Cela ne modifie en rien les sous-

domaines de base. Cette opération est rendue envisageable et facile à exécuter grâce à un ensemble des commandes mises ici à la disposition de l'utilisateur [Escaig 93].

Pour faire apparaître la diminution du nombre d'opérations, prenons encore une fois l'exemple de la plaque rectangulaire que l'on découpe en bandes pour plus de simplicité.



La figure 13 montre en détail de quoi sont constitués les différents sous-domaines.

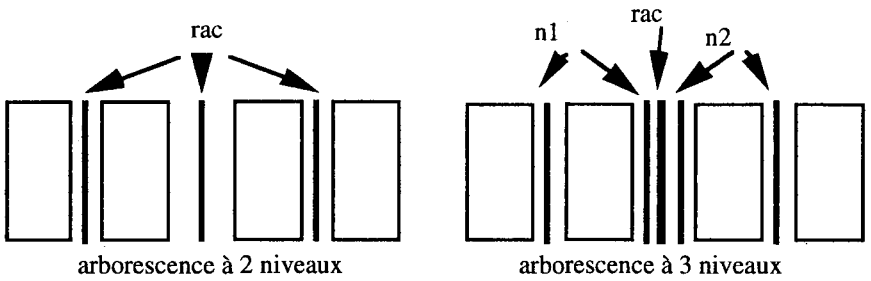


Figure 13. *Composition des différents sous-domaines*

Soit p le nombre de nœuds du maillage dans une direction, et nlng le nombre de nœuds du problème racine. Le nombre d'opérations nécessaires à la factorisation du problème racine, dans le cas où il n'y a qu'un seul niveau de sous-structuration, est égal à :

$$nlng = 3\,(p+1)$$

$$N_{op} = \frac{2\ nlng^{\,3}}{3} = \frac{2\,[3\,(p+1)\ ]^3}{3} = \boxed{18\ (p+1)^{\,3}}$$

Dans le cas de deux niveaux de sous-structuration (soit 3 niveaux dans l'arbre), la méthode frontale est appliquée de nouveau sur le deuxième niveau. Il faudrait donc, en toute rigueur, calculer le nombre d'opérations effectuées au cours de cette élimination frontale. Dans notre exemple, il n'y a que 2 sous-domaines de base par sous-domaine de niveau 2. Donc, la méthode frontale n'est constituée que d'une seule étape, ce qui nous permet d'utiliser en première approximation la formule précédente. De plus, le but ici est plus de démontrer qualitativement la réduction du nombre d'opérations que la quantifier exactement.

Pour la condensation de chacun des sous-domaines *n1* et *n2*, nous obtenons :

$$\text{nlng} = 2\,(p+1)$$

$$N_{n1} = \frac{2\ \text{nlng}_{n1}{}^3}{3} = \frac{2\,[2\,(p+1)\ ]^3}{3} = \frac{16\ (p+1)^{\ 3}}{3}$$

Pour la factorisation de la racine, on a :

$$\text{nlng} = (p+1)$$

$$N_{rac} = \frac{2\ \text{nlng}_{rac}{}^3}{3} = \frac{2\ (p+1)^{\ 3}}{3}$$

soit un nombre total d'opérations :

$$N_{op} = \frac{16\ (p+1)^{\ 3}}{3} + \frac{16\ (p+1)^{\ 3}}{3} + \frac{2\ (p+1)^{\ 3}}{3} = \frac{34\ (p+1)^{\ 3}}{3} \approx \boxed{14{,}7\ \ (p+1)^{\ 3}}$$

Par rapport au cas à un niveau de sous-structuration, le nombre total d'opérations est réduit d'un facteur 1,6.

En augmentant encore le nombre de niveaux dans l'arborescence, le nombre d'opérations continue de diminuer, mais de manière moins importante. Cette moindre diminution est dû au recouvrement de plus en plus grand entre les sous-domaines de niveaux supérieurs.

Pour vérifier ces calculs, nous avons considéré le cas d'une plaque carrée bi-dimensionnelle, maillée en Q8 sur laquelle on résout un problème d'élasticité. Pour une discrétisation de 24 éléments dans chaque direction (soit 3650 d.d.l.), nous avons découpé la plaque en 24 sous-domaines, en utilisant le découpage en carré. L'arbre de ces sous-domaines est constitué successivement de 1, 2, 3 et 4 niveaux de sous-structuration. Les temps d'exécution obtenus sur une station de travail SUN Sparc 1 sont présentés sur la figure 14.

On constate une réduction du temps total d'exécution encore supérieure aux prévisions. Entre 1 et 2 niveaux de sous-structuration, le facteur est de 2,8.

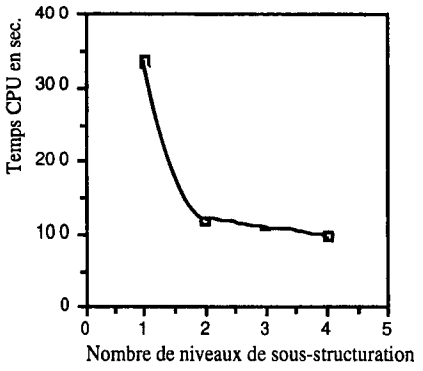


Figure 14. *Evolution des temps d'exécution en fonction du nombre de niveaux de sous-structuration*

Cette constatation est importante puisqu'elle permet d'atténuer les effets d'un découpage en un grand nombre de sous-domaines. En effet, un nombre important de sous-domaines a tendance à faire croître le temps de factorisation de la matrice du sous-domaine racine alors qu'un découpage en plusieurs niveaux de sous-structuration le fait baisser.

D'autre part, cela montre que la modularité des méthodes de décomposition de domaines où l'on découpe de manière récursive le problème de départ en sous-problèmes de complexité moindre ne va pas à l'encontre de l'efficacité numérique.

De plus, ce fait peut être un argument important en faveur des méthodes directes de résolution du problème frontière par rapport aux méthodes itératives qui ne sont pas bien adaptées au découpage multiniveaux. De plus, on peut souligner qu'une sous-structuration multiniveaux efficace et facile à mettre en œuvre permet de réduire à volonté la taille du problème racine, donc son coût en place mémoire, tout en réduisant le plus souvent le temps de calcul total.

8. Comparaison avec une méthode sans décomposition

Dans les deux paragraphes précédents, nous avons montré d'une part que la méthode multifrontale est plus performante que la méthode des sous-structures, et d'autre part que le nombre d'opérations, et donc le temps d'exécution, dépend du nombre de sous-domaines. Il est intéressant maintenant de comparer les méthodes de décomposition de domaines que nous venons de présenter avec les méthodes classiques, sans décomposition.

La première difficulté d'une telle comparaison réside dans la multitude des algorithmes et des implantations qui existent pour les méthodes classiques. Il apparaît donc très difficile, sinon impossible de répondre de manière globale à la question : les méthodes de décomposition de domaines sont-elles plus ou moins efficaces que les méthodes classiques ? Une comparaison significative n'est donc possible que lorsque les algorithmes et les implantations sont proches.

Le seconde difficulté provient du fait que les temps d'exécution pour les méthodes de décomposition de domaines dépendent du nombre de sous-domaines, et plus

généralement du découpage en sous-domaines. Il faut donc comparer les temps d'exécution de la méthode classique avec l'évolution des temps d'exécution des méthodes de décomposition de domaines en fonction du nombre de sous-domaines, pour un type de découpage donné.

La figure 15 présente l'évolution du temps d'exécution de la méthode multifrontale en fonction du nombre de sous-domaines par rapport au temps d'exécution de la méthode globale pour un problème 2D à un d.d.l. par noeud comportant 3721 d.d.l. au total.

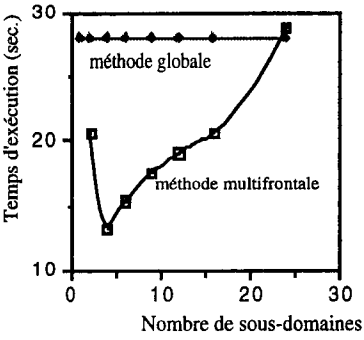


Figure 15. *Temps d'exécution de la méthode globale par rapport à la méthode multifrontale*

Sans parler de parallélisme, mais uniquement pour des exécutions séquentielles, le découpage jusqu'en 24 sous-domaines entraîne une diminution du temps calcul par rapport à une méthode sans décomposition.

Pour des problèmes 2D et 3D d'élasticité linéaire, on retrouve le même type de résultats (figure 16).

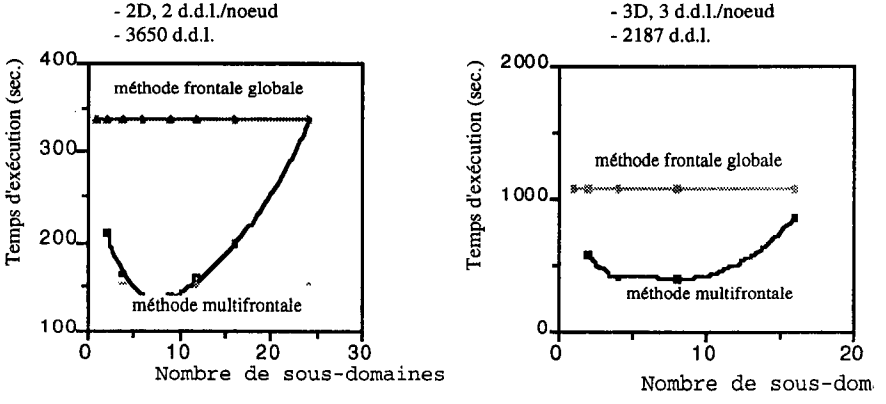


Figure 16. *Comparaison entre les temps d'exécution d'une méthode sans décomposition et de la méthode multifrontale*

Les résultats que nous venons de donner pour la méthode multifrontale ont été obtenus pour un seul niveau de sous-structuration. Or, le paragraphe précédent a montré que l'on peut diminuer le temps de calcul en augmentant le nombre de niveaux de sous-structuration. La figure 17 reprend la courbe du temps d'exécution de la figure xx, en faisant figurer le temps d'exécution obtenu avec la méthode frontale sur l'ensemble du problème.

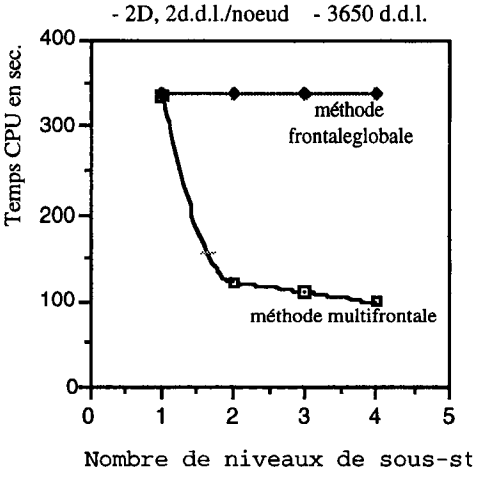


Figure 17. *Comparaison entre les temps d'exécution d'une méthodes sans décomposition et de la méthode multifrontale lorsque le nombre de niveaux de sous-structuration varie*

9. Conclusion

Dans cet article, nous avons présenté une étude des méthodes de décomposition de domaines avec résolution directe du problème interface. Après avoir décrit la méthode classique des sous-structures, nous avons proposé une nouvelle méthode : la méthode multifrontale que nous avons étendue aux cas multiniveaux.

Ensuite, nous avons montré que la méthode multifrontale est plus performante que la méthode des sous-structures.

Nous avons également montré que les performances des méthodes de décomposition de domaines dépendent du nombre de sous-domaines que l'on crée à partir du domaine de départ, et que, dans la plupart des cas, la méthode multifrontale proposée permet une réduction du nombre d'opérations par rapport aux méthodes sans décomposition.

Grâce à la méthode multifrontale multiniveaux développée, nous avons pu mettre en évidence que l'ajout de niveaux de sous-structuration conduit à une réduction du nombre total d'opérations effectuées, tout en permettant de réduire de manière arbitraire la taille des problèmes interfaces.

Du point de vue des performances numériques, la méthode de décomposition de domaines multiniveaux proposée apparaît donc comme une alternative possible aux méthodes sans décomposition, même dans le cas de résolutions séquentielles.

10. Bibliographie

[Debruyne 83] G. DEBRUYNE, Y. WADIER, "Barrage de Garrabet : un calcul tridimensionnel par éléments finis", note E.D.F. Hi/4374-07

[Duff] I. Duff, A. ERISMANN, J. REID, *Direct methods for sparse matrices,* Clarandon Press Oxford

[Escaig 92] Y. ESCAIG, Compte rendu de la rencontre avec M. DELCEY de FRAMASOFT+CSI à Lyon, compte rendu interne E.D.F. n°866-07

[Escaig 93] Y. ESCAIG, M. VAYSSADE, G. TOUZOT, "Automatisation de la décomposition de domaines dans un logiciel de calcul par éléments finis", *Actes du Colloque National en calcul des Structures,* Vol. 2, pp 847-873, Hermes (1993)

[Escaig 94] Y. ESCAIG, M. VAYSSADE, G. TOUZOT, "Parallelization of a multilevel domain decomposition", à paraître dans *Computing Systems in Engineering*

[Imbert 79] J.F. IMBERT, *Analyse des structures par éléments finis*, Ecole Nationale Supérieure de l'Aéronautique et de l'espace, Cepadues Editions

[Irons 69] B.M. IRONS "A frontal solution program for finite element analysis", Int. *Jour. Num. Meth. Eng.*, Vol 2, pp 5-32 (1970)

[Keunings 92] R. KEUNINGS, O. ZONZ, R. AGGARWAL "Parallel algorithms in computational Rheology", in *XIth International Congress on Rheology,* Brussels, Ed P. Moldenaers, R. Keunings, Elsevier, 274-276 (1992)

[Lesoinne 91] M. LESOINNE, C. FARHAT, M. GERADIN, "Parallel/Vector improvements of the frontal method", *Int. J. Numer. Meth. Engrg.*, Vol 32, pp1267-1282 (1991)

[Le Tallec 90] P. LE TALLEC, Y-H. De ROECK, M. VIDRASCU "Domain decomposition methods for large linearly elliptic three dimensional problems", Rapport de recherche de l'INRIA n°1182 (1990)

[Rogé 93] C. ROGÉ, F-X ROUX "Méthodes de calcul des structures composites par sous-domaines sur calculateurs MIMD", *Actes du Colloque National en calcul des Structures,* Vol. 2, pp 920-927, Hermes (1993)

[Roux 89] F-X. ROUX "Méthode de décomposition de domaine à l'aide de multiplicateurs de Lagrange et application à la résolution en parallèle des équations de l'élasticité linéaire", Thèse de l'Université de Paris 6 (1989)

[Sloan 89] S. SLOAN, "A Fortran program for profile and wavefront reduction", *Int. J. Numer. Meth. Engrg.*, Vol 28, pp 2651-2679 (1989)

[Schrem 89] E. SCHREM *Handbook for linear static analysis*, INTES Publication UM 404 Rev C (1989)