Calcul de capacités parasites dans le logiciel InCa3D

1 Introduction à la programmation dans InCa3D

Dans cette partie, nous présenterons les travaux développés au cours de cette thèse pour la prise en compte des capacités parasites dans le logiciel InCa3D [2] co-développé par le G2Elab et CEDRAT.

Les langages de programmation principaux dans le logiciel InCa3D sont le fortran 77, pour les parties de description géométrique et post-processing, et le langage java pour l'intégration et la résolution du problème ainsi que l'IHM. InCa3D peut être piloté par des fichiers de commandes en python.

2 Modèle de donnée des régions capacitives

Une région capacitive, appelée « Region Capa » dans InCa3D, est définie à partir (Fig. B.1) :

- d'une liste de faces externes appartenant à un conducteur unidirectionnel ou bidirectionnel,
- d'une valeur de potentiel,
- de paramètres de maillage automatique des faces (optionnel).

Par défaut, toutes les faces externes d'un conducteur sont enregistrées. Il est également possible de ne choisir que certaines faces si l'on veut par exemple négliger l'épaisseur d'un conducteur très fin. Par conséquent, on peut définir plusieurs Régions Capas par conducteur. Le potentiel sert uniquement dans le cas d'une résolution de charges (car pour le calcul des capacités, celui-ci sera imposé à 0 ou 1 V) ou en post-processing. Les cinq options de maillage sont celles décrites dans la section concernant le mailleur automatique (section 4.4.2, p. 65). Celles-ci sont prioritaires par rapport à celles fixées par défaut pour l'ensemble des conducteurs et des diélectriques (correspondant au numéro « 4 » dans la figure B.1).

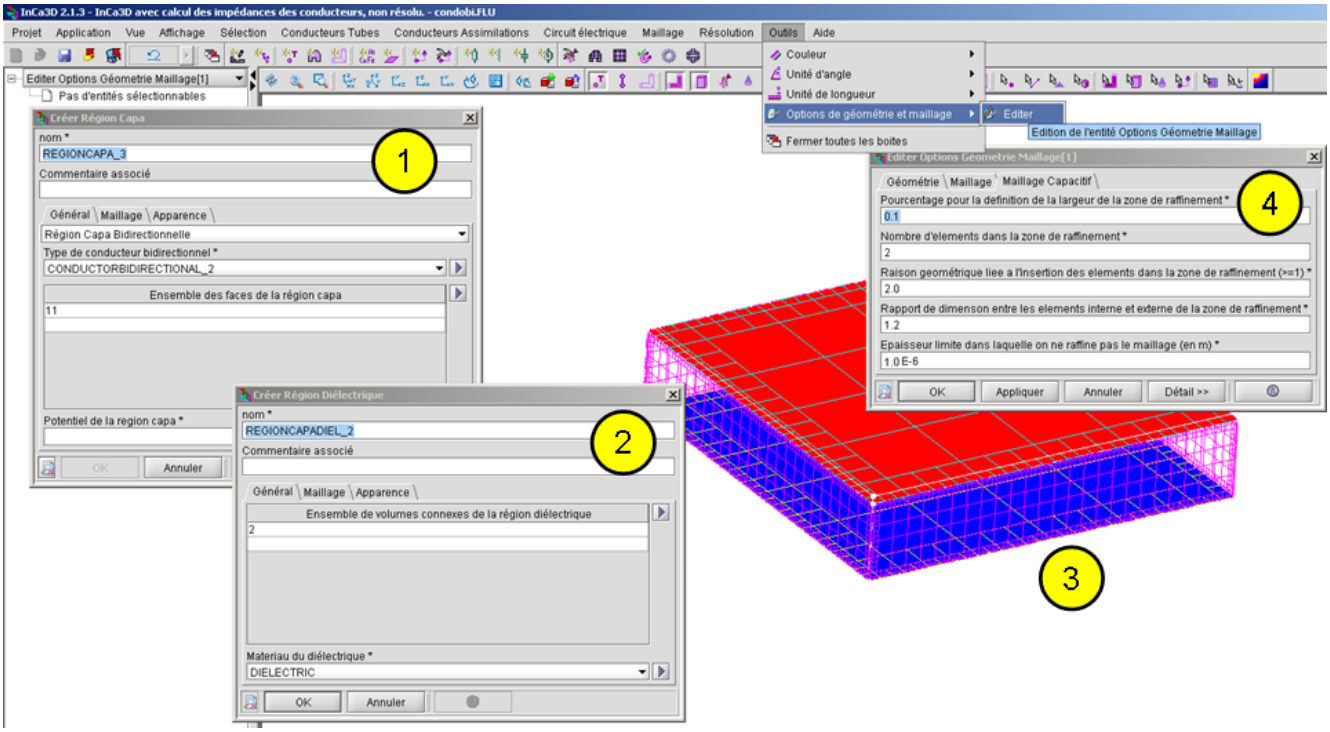

Figure B.1 – Exemple d'IHM dans InCa3D pour la prise en compte des capacités parasites - 1) fenêtre de création d'une Région Capa ; 2) de même pour une Région Diélectrique ; 3) Visualisation des maillages capacitifs et diélectriques ; 4) fenêtre d'options générales du maillage surfacique électrostatique

Lors de la résolution, une ligne (ou colonne) de la matrice des capacités sera attribuée à chaque Région Capa (en respectant l'ordre de création de ces régions).

3 Modèle de donnée des régions diélectriques

Une région diélectrique, appelée « Région Diélectrique », est définie à partir (Fig. B.1) :

- d'une liste de volumes connexes,
- d'un matériaux diélectrique,
- de paramètres de maillage automatique des faces (optionnel).

Dans un premier temps, le groupe de volume choisi doit être connexe afin de déterminer facilement les normales sortantes de chaque élément de maillage à l'aide de l'isobarycentre des éléments du diélectrique. Cette normale sortante est nécessaire pour le calcul du champ normal à l'interface diélectrique-diélectrique. Il est prévu de traiter ultérieurement des volumes diélectriques convexes en calculant et orientant automatiquement les normales sortantes de chaque face externe des volumes diélectrique.

La sélection des faces externes du volume du diélectrique se fait au moment du lancement d'une résolution de charges ou de capacités. Si le diélectrique est collé à un conducteur, les faces communes sont alors enlevées de la liste de faces externes du volume diélectrique.

4 Calcul de charges et de matrices de capacités parasites

Une fois le maillage des Régions Capas et Diélectriques réalisé à l'aide du mailleur automatique, le calcul des charges et/ou des capacités peut alors être lancé. Les données utiles codées en fortran (maillage, valeurs des potentiels, permittivité du diélectrique) sont traduites en java pour être utiliser dans le fichier « .jar » contenant toute l'intégration et la résolution du problème électrostatique. Pour l'instant, les paramètres liés à la formulation (choix de la méthode de collocation ou de Galerkin), à l'intégration (analytique, numérique avec x points de Gauss) et à la résolution (paramètres du GMRES(m) ou du BiCGSTAB) sont pilotés en externes via un fichier texte.

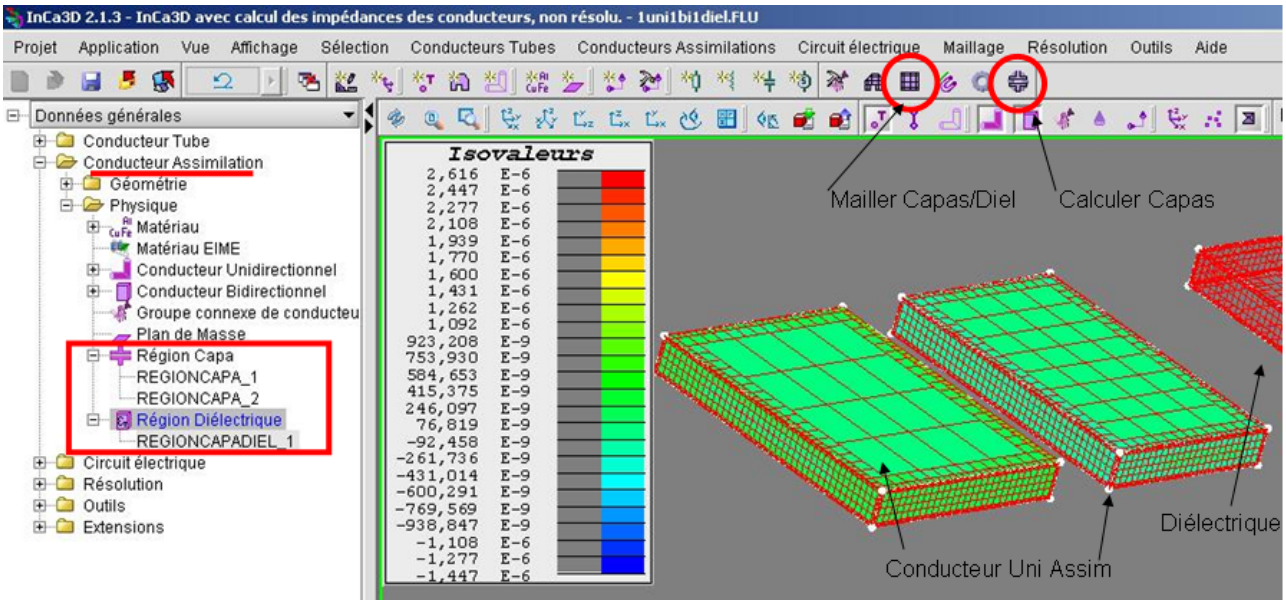

Figure B.2 – Exemple de post-processing après résolutions de charges dans InCa3D

Ensuite, en post-processing, la valeur des charges surfaciques peut être affichées dans l'IHM (Fig. B.2). Cela peut permettre de vérifier la bonne répartition des charges sur le maillage et d'améliorer éventuellement ce dernier. On prévoit la possibilité d'utiliser un algorithme de remaillage adaptatif.

5 Connexion des capacités aux éléments RLM

Dans un premier temps, la connexion des capacités aux éléments RLM est effectué via un fichier de commande python. Ce fichier de commande est créé automatiquement (via une routine java) à partir des noms des bornes PEEC choisies (nœuds électriques de connexions présents dans les conducteurs). Plusieurs options de connexions sont possibles pour créer soit des cellules PEEC en Π (demi-capacités), en T ou en Γ (voire section 2.2, p. 127). Pour toutes ces connexions, on utilise un algorithme permettant de connecter les capacités entre deux bornes PEEC les plus proches (en prenant en compte la distance).

Le lancement de ces fichiers de commande permet soit de créer et connecter automatiquement la matrice de capacités à la modélisation résistive-inductive soit de modifier la valeur des capacités parasites.

6 Création du macro-bloc capacitif pour un solveur circuit

De la même manière que pour la section précédente, un macro-bloc capacitif codé en langage cicuit spice est généré (via une routine java) à partir des noms des bornes PEEC choisies. Le type de connexion interne des capacités est également optionnel comme dans la section précédente. Ainsi, il est possible de générer un macro-bloc capacitif avec les mêmes noms de bornes PEEC que ceux du macro-bloc RLM.

7 Application de ces outils à un cas test de Schneider-Electric du projet O2M

Dans le cadre du projet d'**O2M**, des cas tests ont été identifiés. Un cas test de Schneider-Electric : un hacheur élévateur multi-couches réalisé avec des sous-parties du variateur de vitesse de STIE 3 (p. 153) a été modélisé avec tous les outils développés dans InCa3D. Ce convertisseur est présenté dans la figure B.3. Le temps de description de la géométrie et de la physique (inductive et capacitive) dans la version de développement du logiciel InCa3D, ainsi que les temps de calcul et la connexion des macro-blocs RLM et C comprennent environ deux journées de travail. Le temps passé pour ce dispositif est beaucoup plus rapide que celui passé par Jérémie Aimé sur le variateur de vitesse traité dans le chapitre cinq (section 3, p. 153) qui se compte en plusieurs semaines.

Une modélisation CEM conduite et rayonnée grâce au circuit équivalent de ce convertisseur est actuellement en cours de test au sein de Schneider-Electric.

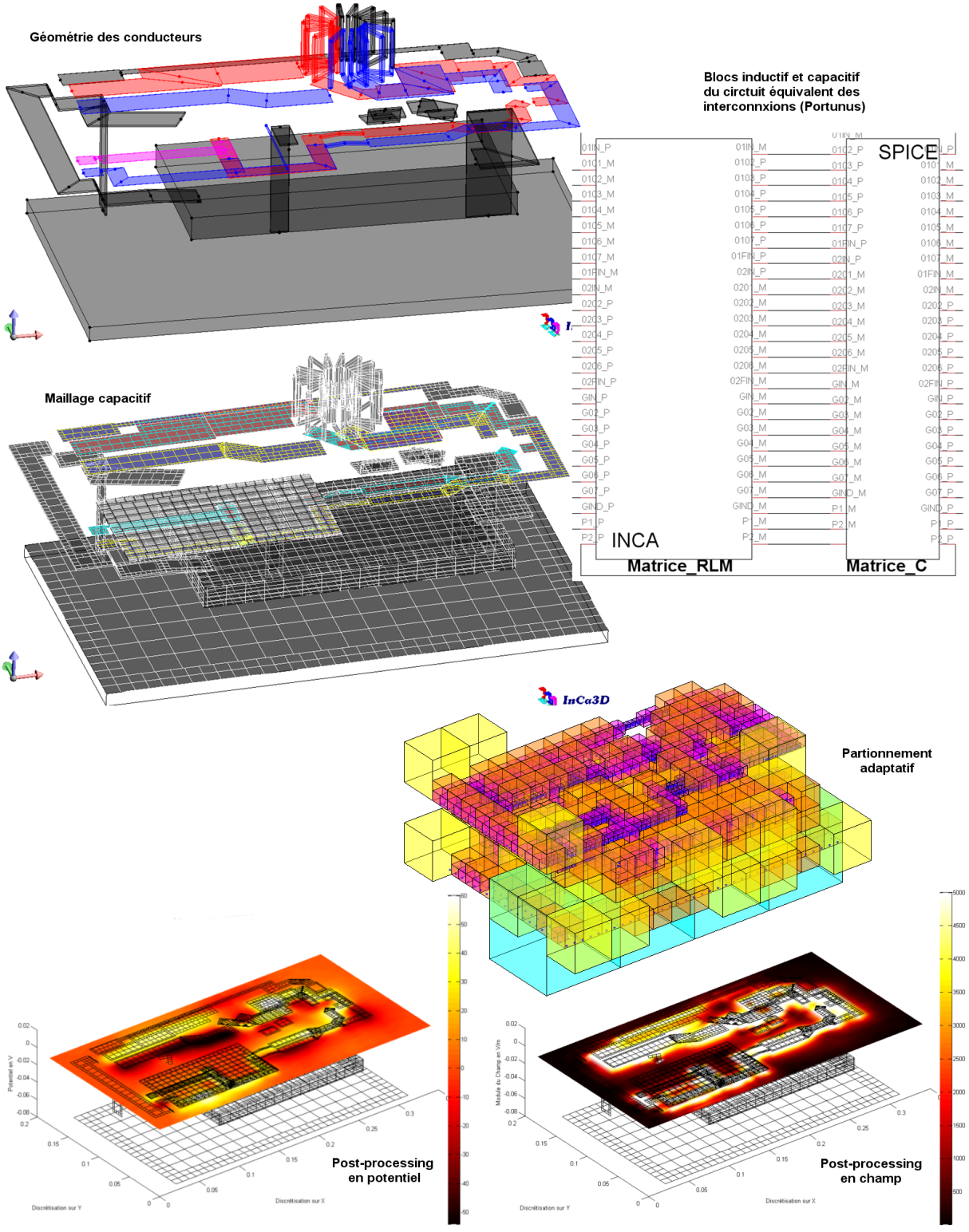

Figure B.3 – Modélisation dans InCa3D et Portunus d'un hacheur : cas test de Schneider-Electric du projet**O2M** - visualisation du maillage capacitif, du partitionnement adaptatif au niveau 6 et de post-processings en potentiel et en champ électrique

Annexe C

Calcul parallèle multi-processeur

1 Introduction

Les travaux présentés dans cette annexe ont été réalisés juste avant la soutenance de thèse et n'ont pas été analysés par les rapporteurs.

Aujourd'hui, les nouvelles générations d'ordinateurs intègrent au moins deux processeurs. Nous avons souhaité bénéficier de cette technologie pour accélérer encore plus les routines de calcul en les parallélisant. Ainsi, cette partie traite du calcul parallèle sur un ordinateur équipé de plusieurs processeurs et non sur un ensemble d'ordinateurs ou cluster.

Les méthodes intégrales contrairement aux méthodes des éléments finis (matrice creuse) sont plus facilement parallélisables. Nous avons parallélisé des routines d'intégration et de résolution.

2 Mise en œuvre

Les routines d'intégration en interactions proches (remplissage des petites matrices pleines) ont été parallélisées. Nous intégrons ainsi en parallèle les matrices d'interactions proches des cubes feuilles et de leurs cubes adjacents (section 4.2, p. 97).

L'inversion des blocs (inverse des matrices d'interactions proches, section 5.5, p. 108) dans le processus de préconditionnement a également été parallélisée.

Enfin, dans le solveur itératif GMRES(m), les calculs de potentiels et de champs lointains (avec la FMM) et proches, ainsi que le préconditionnement, ont été parallélisés. Pour ces deux derniers calculs, il s'agit simplement de paralléliser des produits matrice-vecteur. Pour les calculs de potentiel ou de champ avec la FMM, le processus de parallélisation est plus complexe à mettre en place et ne sera pas détaillé ici.

3 Performances en temps

La figure C.1 montre la comparaison du temps de résolution en fonction du nombre de processeur (un ou deux) dans le cas des deux plaques parallèles maillées entre 500 et 250 000 éléments environ. Le temps de résolution de ces problèmes peut être détaillé en trois parties : le partitionnement (routine pas encore parallélisée), l'intégration proche (et lointaine, appelée « Potentiel FMM » dans la figure C.1) et la résolution avec le solveur itératif GMRES préconditionné (le temps de préconditionnement est noté « Précond. Mat. Diag. »). Le temps noté « GMRES(30) » est le temps total d'une résolution avec une taille de sous-espace de Krylov fixée a 30 ; il comprend toutes les routines du GMRES (non parallélisées) et les routines parallélisées de préconditionnement, de calcul de potentiel (ou de champ) proche et lointain.

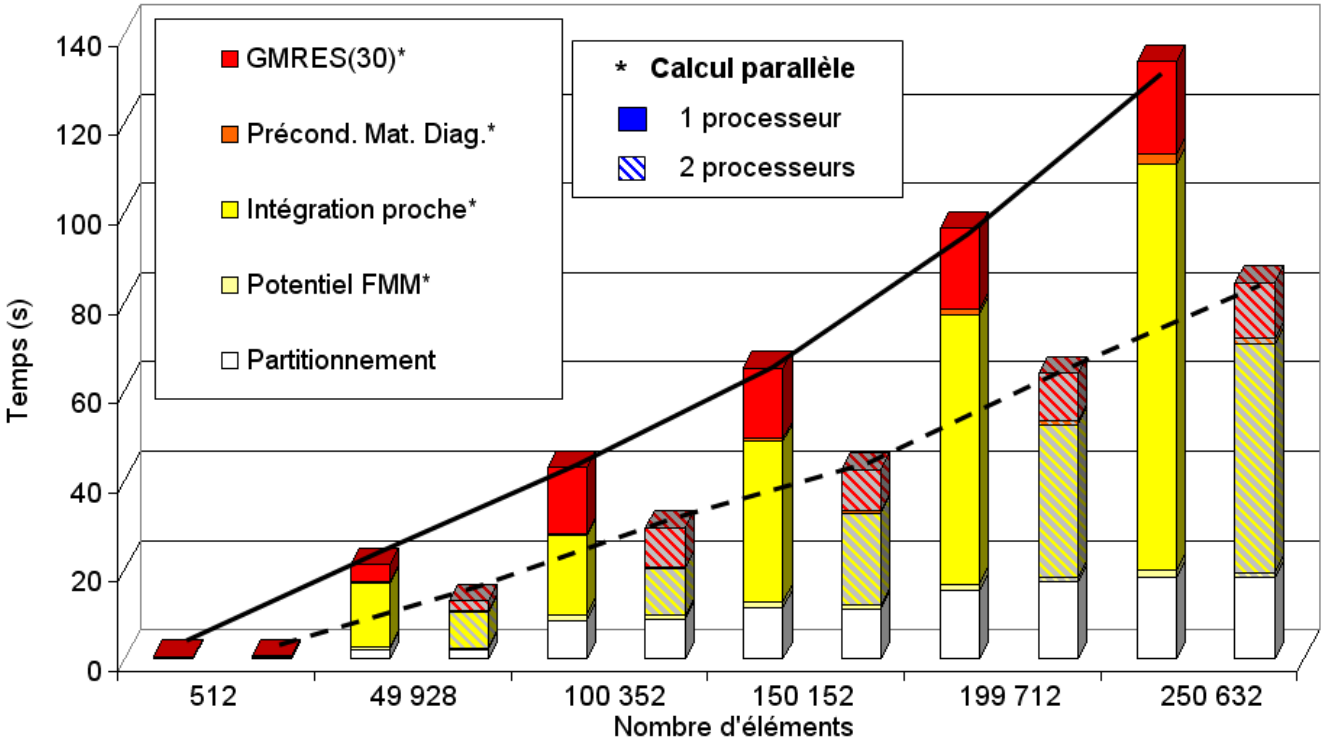

Figure C.1 – Comparaison en temps des routines de calcul avec 1 ou 2 processeurs

La figure C.2 montre pour le cas des deux plaques parallèles maillées en 300 000 éléments, le temps d'intégration des matrices d'interactions proches en fonction du nombre de processeurs. On peut remarquer que la décroissance du temps d'intégration en fonction du nombre de processeurs suit bien une loi en $\mathcal{O}(1/N)$ si N est le nombre de processeur.

Le tableau C.1 montre les facteurs de division en temps moyens, obtenus pour plusieurs processeurs, de l'ensemble des routines de calcul parallélisées. Ces données sont représentées dans la figure C.3 pour visualiser l'écart entre l'accélération en temps idéale et celle obtenue par la simulation. La courbe notée « Accélération idéale » est linéaire : avec N processeurs, on espère accélérer les temps par un facteur N. Or, ce n'est pas le cas en pratique car les temps de communi-

cation avec la mémoire virtuelle java ne sont pas idéaux et dépendent également de l'architecture physique de la mémoire cache [81]. Ainsi, les résultats obtenus sont conformes aux attentes.

Tous ces résultats de comparaison montrent des gains en temps très importants pour peu de complexité supplémentaire ajoutée aux algorithmes : sur n'importe quel ordinateur muni de deux processeurs, on divise quasiment le temps de traitement d'un problème par deux !

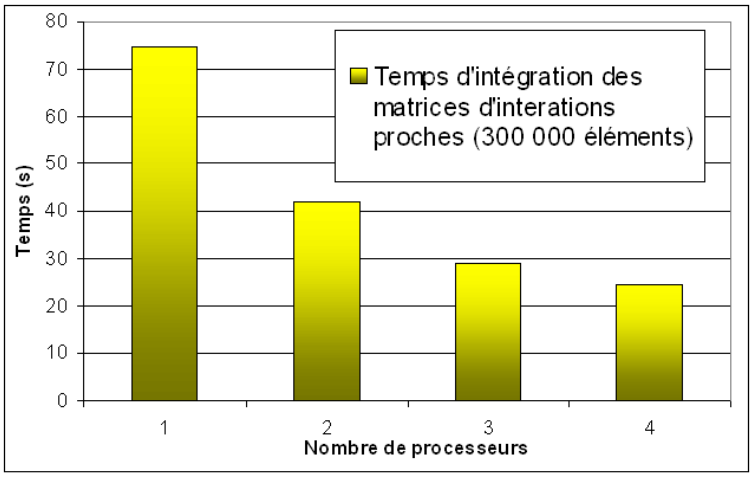

Figure C.2 – Comparaison des temps d'intégration des matrices d'interactions proches entre 1 et 4 processeurs

Nombre de processeurs	2	3	4
Facteur de division en temps	1,8	2,6	3,1

Tableau C.1 – Facteurs moyens de division en temps pour plusieurs processeurs pour l'ensemble des routines de calcul parallélisées

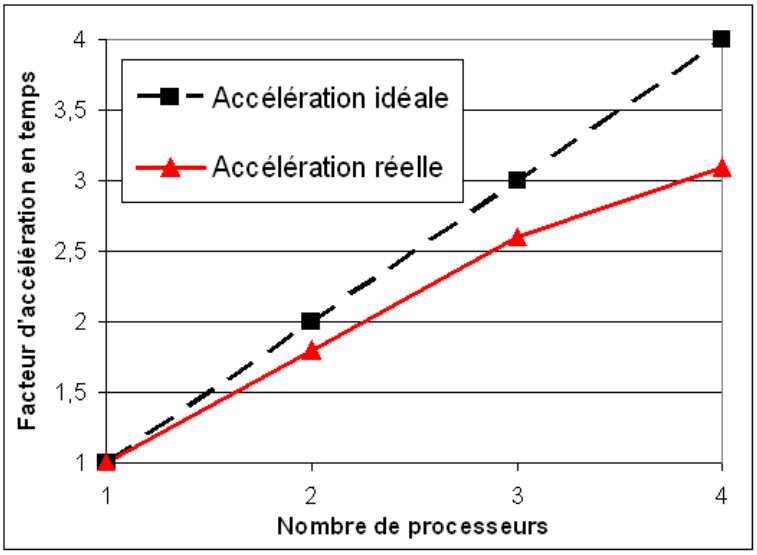

Figure C.3 – Courbes d'accélération en temps avec plusieurs processeurs