Parallélisation du calcul *ab-initio* de l'énergie de corrélation électronique

Nicolas Maillard, Jean-Louis Roch[a], Pierre Valiron[b] *

[a]Nicolas.Maillard@imag.fr, Jean-Louis.Roch@imag.fr, LMC-IMAG, Institut Fourier BP 53X, 100 Rue des Mathématiques, 38041 Grenoble Cedex 9

[b]Pierre.Valiron@obs.ujf-grenoble.fr, Observatoire de Grenoble, U. Joseph Fourier BP 53 F-38041 Grenoble Cedex 9

Mots clef : chimie quantique, parallélisme, transformations OA-OM, MP2, PRAM

Résumé

Nous montrons sur l'exemple d'une application en chimie quantique (calcul *ab-initio* de l'énergie de corrélation électronique) comment la restructuration du schéma de calcul séquentiel permet la construction d'un algorithme parallèle dont la performance (temps parallèle, nombre de processeurs, espace mémoire et communication) peut être analysé sur un modèle théorique (LPRAM). L'implémentation de l'algorithme en MPI et son expérimentation permettent de confronter cette analyse à la parallélisation effective.

1. Introduction

En chimie quantique, on est amené à résoudre de manière approchée l'équation de Schrödinger sur des ensembles de particules pour en calculer l'énergie. Un grand nombre de méthodes utilisent la transformation des intégrales de répulsion électronique (ERI) de la base des orbitales atomiques (OA) à la base des orbitales moléculaires (OM), appelée *transformation à quatre indices*. Cette étude présente la parallélisation d'une telle transformation, que l'on utilise en particulier dans le cadre de la méthode des perturbations du deuxième ordre (MP2). Dans un souci de clarté nous exposerons la transformation telle qu'elle est utilisée dans MP2.

On trouve des tentatives de parallélisation de la transformation à quatre indices et de MP2 depuis quelques années [2] dans la communauté de "computational chemistry" : [4], [5] ou [3].

Dans la suite nous exposerons l'algorithme (section 2) selon notre formulation en terme de supermatrices. Nous expliquerons ensuite le parallélisme intrinsèque à cette formulation et étudierons la complexité de MP2 dans différents modèles (section 2.3). Nous étudions enfin une implémentation en MPI sur IBM SP1 en analysant les performances pratiques de la parallélisation proposée.

2. Parallélisation de l'algorithme de transformation à quatre indices

Cet algorithme est constitué de 4 phases de combinaison linéaire sur des cœfficients à quatre indices (ERI). Nous montrons comment on peut le réécrire à l'aide de produits matriciels indépendants ce qui présente l'intérêt d'une part de rendre possible l'utilisation de BLAS-3 et d'autre part de dégager une granularité pour le parallélisme.

Cette reformulation se place explicitement dans le cadre de deux restrictions importantes : d'abord on suppose pouvoir obtenir de manière indépendante chaque donnée initiale en un temps u négligeable par rapport aux autres phases du calcul ; ensuite on ne tient pas compte

* . Ce travail est supporté par le projet CNRS-IMAG-INPG-INRIA Apache et par l'observatoire de Grenoble.
2. Nous remercions Jérôme Devémy pour son aide précieuse dans l'établissement de la bibliographie sur MP2.

non plus des propriétés de symétrie dans les supermatrices qui permettent, dans les codes de référence, de diviser le temps séquentiel par un facteur (au mieux) de huit.

2.1. Formalisme et notations

Dans toute cette section on utilise des matrices de matrices (ou supermatrices) : M est une matrice formée de $N \times N$ blocs de taille $N \times N$. On notera $M_{(\alpha,\beta),(i,j)}$ le cœfficient d'indice $(\alpha-1)N+i,(\beta-1)N+j$ *i.e.* l'élément (i,j) du bloc (α,β) de M. On appellera superligne (respct. supercolonne) une sous-matrice $N \times N^2$ (respct. $N^2 \times N$) de M. On notera aussi $\mathbf{M}(p,q,r)$ la complexité du produit de deux matrices de dimensions $p \times q$ et $q \times r$. Sauf exception, on considérera que $\mathbf{M}(p,q,r) = pqr$.

2.2. Transformations des OA en OM et sommation MP2

Soit les supermatrices suivantes : M est la matrice des données initiales du calcul ; C_O et C_V sont des supermatrices diagonales par blocs, dont tous les blocs diagonaux valent la même matrice (respectivement c_O et c_V, denses toutes les deux)[3], P est la supermatrice de permutation dont l'action est la suivante : $(M \times P)_{(\alpha,\beta),(i,j)} = M_{(\alpha,j),(i,\beta)}$ *i.e.* le produit à droite (respt. à gauche) par P permute les indices de colonnes (respt. de lignes) entre blocs et éléments[4].

On montre facilement que le calcul MP2 (exposé par exemple sous sa forme classique dans [1]) se reformule alors en :

1. Effectuer la transformation à quatre indices, *i.e.* calculer la supermatrice
   $E = C_O.P.C_O.M.C_V.P.C_V$; (1)
2. calculer l'énergie MP2 : $\sum_{\alpha,j,\beta=1,N\, i\in I_b} \frac{(E_{(\alpha,\beta),(i,j)} - E_{(\alpha,j),(i,\beta)})^2}{\epsilon_i+\epsilon_j-\epsilon_\alpha-\epsilon_\beta}$ qui est une norme de la supermatrice $E - EP$ pondérée par des cœfficients ϵ connus à l'avance.

Une façon astucieuse de faire ce calcul est alors le suivant : on découpe l'intervalle $1 \ldots O$ en O/I intervalles de taille I. L'indice b ("Batches") parcourt dans la suite en parallèle ces O/I intervalles I_b, et i parcourt séquentiellement les valeurs d'un intervalle.

L'algorithme suivant décrit un parenthésage de l'expression (1) permettant d'évaluer l'énergie avec un espace mémoire en $I.N^3$:

Entrées : valeurs de M, supposées calculables à la volée en temps u.
Do $b = 1, O/I$ en parallèle :

1. **Calcul des lignes $i \in I_b$ de tous les blocs de C_OMC_V.** On obtient une supermatrice $\tilde{M}_b$ dont seules les I lignes $i \in I_b$ de chaque superligne sont non-nulles.
2. **Produit à gauche par C_OP.** On effectue ensuite la produit des I superlignes $i \in I_b$ de (C_OP) par $\tilde{M}_b$. Ceci se ramène à IN produits de matrices $N \times N$ (*cf.* figure 1).
3. **Produit à droite par PC_V.** Chaque bloc résultat est obtenu par produit de deux matrices $N \times N$ (*cf.* figure 1).
4. **Calcul de la norme de $E - EP$.** Comme les I superlignes de E_b sont stables par la multiplication à droite par P, on a tous les éléments nécessaires au calcul de (2).

$b := b + 1$
Fin Do

2.3. Complexité

Espace mémoire et nombre d'opérations : dans tout le calcul on ne manipule au plus que I superlignes à chaque itération, soit $I.N^3$ cœfficients. En terme d'opérations, en comptant O/I itérations, il vient successivement : le temps d'initialisation de M en N^4u

3. Dans un but de simplification, on n'introduira pas ici les nombres d'orbitales occupées et virtuelles. Ceci ne change en rien l'algorithme.

4. Soit $1 \le p,q \le N$ et soit $P_{p,q} = I_N - \Delta_{p,p} - \Delta_{q,q} + \Delta_{p,q} + \Delta_{q,p}$ où $\Delta_{p,q}$ est la matrice $N \times N$ nulle sauf l'élément (p,q) qui vaut 1. P s'écrit alors $P = \prod_{i<j=1}^{N} P_{(i,j),(j,i)}$.

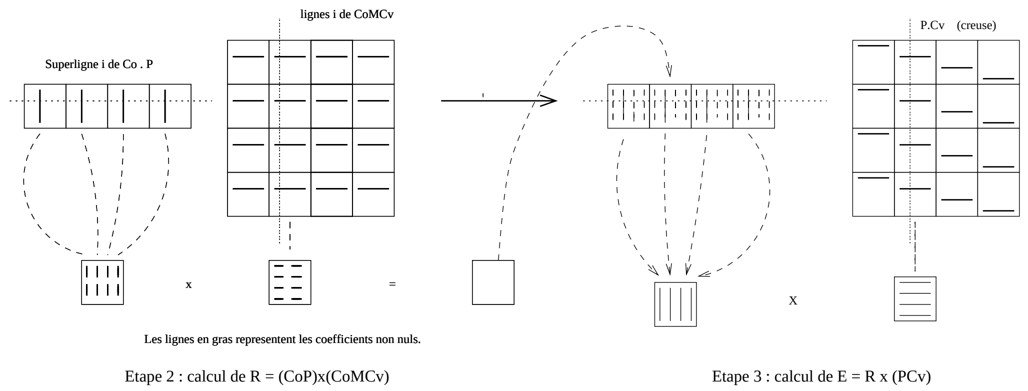

FIG. 1 –. Produits par $C_O P$ et PC_V. L'énergie finale sera obtenue par sommation des cœfficients de E.

que l'on néglige face à ON^4 ; La première étape qui implique $(O/I)N^2$ produits matriciels $N \times I$ par $I \times N$, soit $O.N^4$ opérations. Les autres étapes ont de même une complexité en $(O/I)IN\mathbf{M}(N,N,N) = O.N^4$. Il est à noter que cette complexité peut être réduite par l'utilisation de produits de matrices rapides, dont certains sont implémentés dans les BLAS. La sommation finale est quant à elle en ON^3.

Le paramètre I établit un compromis entre la mémoire dont on dispose et le nombre de calculs que l'on doit ré-effectuer à chaque itération : ceci est important dans le cas où N n'est pas très élevé car bien qu'asymptotiquement négligeable face au coût des étapes suivantes, il peut être prépondérant sur de petits systèmes.

Temps parallèle et communications: l'algorithme proposé est de temps parallèle $\log N$ si on choisit $I = O$ (sans recalcul) et de travail optimal (ON^4 opérations). Si l'on choisit une valeur donnée $1 \leq I \leq O$ du paramètre I qui optimise l'espace mémoire, on obtient un temps $(O/I)\log N$ avec un travail optimal ON^4.
Pour minimiser le coût des communications on rappelle qu'un regroupement par blocs bidimensionnels permet sur une PRAM à n processeurs d'effectuer un produit de matrices $n \times n$ en temps $\mathbf{O}(n^2)$ avec un volume de communications par processeur en $\mathbf{O}(n^{4/3})$ (soit $\mathbf{O}(n^{7/3})$ en travail total) en supposant les matrices réparties uniformément [2].

Dans ces conditions, avec un I fixé et en affectant N processeurs à chaque produit matriciel $N \times N$, le calcul MP2 s'effectue en temps $\mathbf{O}((O/I)N^2)$ sur $\mathbf{O}(IN^2)$ processeurs [5] ; le travail en ON^4 est optimal et le volume de communications est en $\mathbf{O}((O/I)N^{4/3})$ sur chaque processeur, soit un travail en communications en $\mathbf{O}(ON^{10/3})$.

Analyse sur $p < N$ processeurs: en pratique on se limite à un nombre fini $p < N$ de processeurs. On est donc amené à regrouper $(O/I)IN/p$ produits matriciels par nœud disponible. Le volume total de calculs reste inchangé et le volume de communication par processeur est en $(O/I)(IN/p).N^2$ (le produit par P distribuant uniformément les cœfficients), soit un travail total en communications en ON^3. La mémoire requise par processeur, grâce à la découpe en blocs, est en $I.N^3/p$.

3. Implémentation et résultats

L'étude algorithmique est donc prometteuse. L'un des buts de cette étude est de valider cette approche théorique du parallélisme et il nous faut maintenant prouver par la programmation que des résultats concrets peuvent en déboucher. De plus il faut tenir compte de l'importance pratique que peuvent avoir les constantes dans les complexités théoriques en

5. On remarque que les produits successifs conservent l'uniformité de la répartition d'origine : chaque processeur en début et fin d'étape possède N cœfficients.

les validant expérimentalement.

La courbe ci-après montre l'accélération obtenue pour un programme MPI (la partie séquentielle étant écrite en Fortran-77 et 90) sur l'IBM SP-1 du LMC-IMAG, avec un jeu de données pour lequel $N = 72$ et $O = 8$ et un nombre de processeurs allant de 1 à 32.

La courbe présentée ne comporte les mesures que sur les temps des transformations matri-

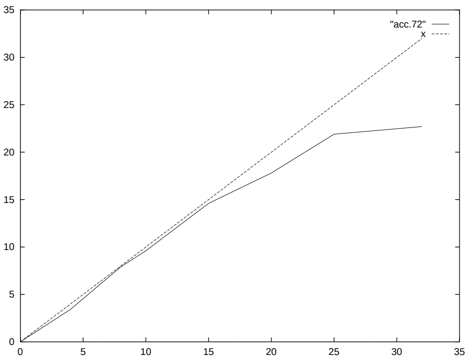

FIG. 2 –. Accélération sur la transformation à quatre indices

cielles au centre de cet article ; il est à noter que sur de tels jeux de données le temps de calcul des données initiales est en pratique loin d'être négligeable. Pour avoir un MP2 complètement parallèle il est alors nécessaire d'interfacer cette version parallèle de la transformation à quatre indices avec un générateur parallèle des ERI et donc un code SCF parallèle.

4. Conclusion et perspectives

Nous avons réécrit un algorithme de transformation à quatre indices selon un formalisme permettant d'en extraire un fort parallélisme. Dans le cadre de MP2 la complexité est dominée par la première transformation car les transformations suivantes ne concernent plus que les orbitales occupées. Il est alors utile de poursuivre ce travail par la parallélisation explicite du calcul des ERI et par la prise en compte des symétries associées. Pour les autres méthodes post-SCF en chimie *ab-initio*, une transformation complète à 4 indices (de complexité N^5 en calculs) est requise, et représente un goulot d'étranglement significatif en calcul et en mémoire sur les calculateurs séquentiels. Notre approche représente ainsi une étape importante en vue de leur parallélisation efficace.

Le but de ce travail à terme est double : nous souhaitons à la fois proposer une utilisation effective d'un programme parallèle calculant l'énergie de corrélation électronique grâce à ces transformations et obtenir une version plus élaborée qui traiterait des problèmes de régulation de charge, surtout sur réseaux de machines hétérogènes, à travers diverses plateformes : MPI, ATHAPASCAN qui seront ainsi testées sur un problème réel.

Bibliographie

1. M. Head-Gordon, J. Pople, and M. Frisch. MP2 energy evaluation by direct methods. *Chem. Phys. Lett.*, 153:503–506, 1988.
2. J. Jàjà. *An Introduction to Parallel Algorithms*. Addison-Wesley, 1992.
3. A. C. Limaye and S. R. Gadre. A general parallel solution to the Møller-Plesset energy evaluation. *J. Chem. Phys.*, 100(2):1303–1307, Jan. 1994.
4. J. D. Watts and M. Dupuis. Parallel computation of the Møller-Plesset second-order contribution to the electronic correlation energy. *J. Comp. Chem.*, 9(2):158–170, 1988.
5. R. A. Whiteside and al. Parallel algorithm for quantum chemistry : Integral transformation on a hypercube multiprocessor. *J. Phys. Chem.*, 86:2185–2193, 1987.