

フラグメント分子軌道法プログラム ABINIT-MP における 2 電子積分ルーチンの高速化ならびに並列化と性能評価

稲 富 雄^{†1} 中 野 達 也^{†2}
北 浦 和 夫^{†3} 長 嶋 雲 兵^{†4}

Ab initio フラグメント分子軌道 (FMO) 法は、巨大分子の電子状態計算を目的として北浦ら (Kitaura, 1999) によって提唱された計算手法である。従来の方法では分子サイズの 4 乗に比例していた計算量が、FMO 法では 2 乗に比例するために、大幅に計算時間が短縮できる。この FMO 法においても、最も計算時間がかかるのが 2 電子積分の計算であるため、本研究では高速に 2 電子積分を行う計算ルーチンを作成し ABINIT-MP (Nakano, 2000) に組み込んで、その性能評価を行った。HITACHI SR8000 (512PU×1GB) で計算を行ったところ、2 電子積分の計算速度が 4 倍から 7 倍向上した。一方、フラグメント (ペア) 内の並列化 (細粒度並列化) を行った結果、本ルーチンが 90% 前後の高い並列化効率を達成していることが分かった。しかしながら、2 電子積分の計算速度が速くなったこともあり、計算コストに対する通信速度の重みが増してしまい、SR8000 では細粒度並列化による大幅な実行時間の短縮が観測されなかった。

Improvement, Parallelization and Performance Evaluation of Routines for Electron Repulsion Integrals in Fragment Molecular Orbital Method Program ABINIT-MP

YUICHI INADOMI,^{†1} TATSUYA NAKANO,^{†2} KAZUO KITAURA^{†3}
and UMPEI NAGASHIMA^{†4}

Ab initio fragment molecular orbital (FMO) method is a computational technique proposed by Kitaura, et al. (1999) to calculate the electronic structure for large molecule such as polypeptide. The computation cost proportionally to fourth power of the molecular size in the use of conventional method becomes proportional to its square when the FMO method is used, and the FMO method can dramatically shorten the computation time. Since the evaluation of the electron repulsion integrals (ERIs) is still most time-consuming step even in the FMO calculations, we implemented the routines for the fast ERI calculation, embedded them in the ABINIT-MP program (Nakano, 2000) and carried out the performance evaluation of them. As the results of the calculation on the HITACHI SR8000 (512PU×1GB), the ERI computation speed becomes faster by 4 ~ 7 times than original program. On the other hand, the parallel ERI computation (fine-grained parallelization) achieved high parallelization efficiency of more about 90%. However, the weight of the communication cost becomes relatively large to the computation cost because of the shortening of the ERI evaluation time, and the dramatic improvement of the computation time in employing the fine-grained parallelization could not be observed on the SR8000.

1. はじめに

原子・分子の電子状態を求めるために用いられる分子軌道法は、2 電子 (反発) 積分 (electron repulsion integrals, ERIs) の計算量が系の大きさの 4 乗に比例するために、従来の方法を用いた場合には適用範囲が 50 残基のタンパク程度までと限られていた。北浦ら^{1)~3)} によって提唱された *ab initio* フラグメント分子軌道 (FMO) 法は、さらに大きな巨大分子に対する分子軌道計算を行うために開発された計算手法であ

†1 株式会社富士総合研究所

Fuji Research Institute Corporation

†2 国立医薬品食品衛生研究所

National Institute of Health Science

†3 大阪府立大学

Osaka Prefecture University

†4 産業技術総合研究所

National Institute of Advanced Industrial Science and
Technology

り、計算精度をほとんど下げることなく 500 残基を超えるようなタンパクの電子状態を求めることが可能である。FMO 法では、分子をいくつかの小さなフラグメントに分割し、各フラグメントやフラグメントペアに対する分子軌道計算を行うことで、分子全体のエネルギーや電荷分布などを求める。フラグメントやフラグメントペアの電子状態は独立に計算することができるため、FMO 法は並列性が高い計算手法である。すでに Nakano ら³⁾ によって MPI ライブラリを用いて並列化した並列 FMO 計算プログラム ABINIT-MP が開発されており、並列計算機を用いた性能評価^{4),5)} でも高性能を発揮することが確かめられている。

しかしながら、FMO 法においても、最も計算コストがかかるのは ERI の計算であり、計算時間を短縮するためには、ERI 計算の高速化、ならびにフラグメント (ペア) 内の 2 電子積分の並列化が必要である。本研究では、ABINIT-MP に Head-Gordon ら⁶⁾ が提案した ERI 計算アルゴリズムを用いて ERI の高速計算ルーチンを作成し、さらに MPI を用いた並列化も行って、その性能を評価した。

2. 2 電子積分計算アルゴリズムの概要

ここでは ERI とその計算アルゴリズムに関する簡単な説明を、文献 6), 7) に従って行う。ほとんどの量子化学計算では、対象とする化学種の波動関数の展開で用いる基底関数 (ϕ_μ) として、次式で表される縮約デカルトガウス型関数 (contracted Cartesian Gaussian type function, CCGTF) を使用する。

$$\phi_\mu(\mathbf{r}; \mathbf{n}, \mathbf{R}) = \sum_i^{K_\mu} D_{i\mu} \varphi(\mathbf{r}; \zeta_i, \mathbf{n}, \mathbf{R}), \quad (1)$$

ここで、 $\mathbf{r} = (x, y, z)$ は電子座標、 $\mathbf{R} = (R_x, R_y, R_z)$ は関数の中心座標、 $D_{i\mu}$ は縮約係数、 ζ_i は軌道指数、それに K_μ は縮約長と呼ばれる線形結合の長さである。 $\mathbf{n} = (n_x, n_y, n_z)$ は角運動量指数であり、各成分の和 $\lambda = n_x + n_y + n_z$ は、関数の形を特徴づける指標となる。通常、 $\lambda = 0, 1, 2, \dots$ である関数は、それぞれ s, p, d, \dots 型の関数と呼ばれ、同じ中心、 λ 、および軌道指数の組 $\{\zeta_i\}$ を持つ CCGTF の組を contracted shell (CS) と呼ぶ。さらに、規格化されていない原始デカルトガウス型関数 (primitive Cartesian Gaussian type function, PCGTF) $\varphi(\mathbf{r}; \zeta_i, \mathbf{n}, \mathbf{R})$ は次のように定義されている。

$$\varphi(\mathbf{r}; \zeta, \mathbf{n}, \mathbf{R}) = (x - R_x)^{n_x} (y - R_y)^{n_y} (z - R_z)^{n_z} \times \exp[-\zeta(\mathbf{r} - \mathbf{R})^2]. \quad (2)$$

規格化されていない PCGTF に対する原始 2 電子

積分 $[\mathbf{a}_p \mathbf{b}_q, \mathbf{c}_r \mathbf{d}_s]$ は 6 次元積分であり、次のように表される。

$$[\mathbf{a}_p \mathbf{b}_q, \mathbf{c}_r \mathbf{d}_s] = \iint \varphi(\mathbf{r}_1; \zeta_p, \mathbf{a}, \mathbf{A}) \varphi(\mathbf{r}_1; \zeta_q, \mathbf{b}, \mathbf{B}) \frac{1}{|\mathbf{r}_1 - \mathbf{r}_2|} \times \varphi(\mathbf{r}_2; \zeta_r, \mathbf{c}, \mathbf{C}) \varphi(\mathbf{r}_2; \zeta_s, \mathbf{d}, \mathbf{D}) d\mathbf{r}_1 d\mathbf{r}_2. \quad (3)$$

一般の分子軌道計算で操作されるのは、この原始 2 電子積分ではなく縮約 2 電子積分 ($\mathbf{a}_\mu \mathbf{b}_\nu, \mathbf{c}_\rho \mathbf{d}_\sigma$) で、次式で定義される。

$$(\mathbf{a}_\mu \mathbf{b}_\nu, \mathbf{c}_\rho \mathbf{d}_\sigma) = \sum_p^{K_a} \sum_q^{K_b} \sum_r^{K_c} \sum_s^{K_d} D_{p\mu} D_{q\nu} D_{r\rho} D_{s\sigma} [\mathbf{a}_p \mathbf{b}_q, \mathbf{c}_r \mathbf{d}_s]. \quad (4)$$

以下の式の中では、特に断らない限り、添字 p, q, r, s および μ, ν, ρ, σ を省く。

Obara ら⁷⁾ は ERI の高速計算を行うために、式 (5) で表される VRR (vertical recurrence relation) と呼ばれる効率の良いアルゴリズムを開発した。

$$\begin{aligned} & [(\mathbf{a} + \mathbf{1}_i) \mathbf{b}, \mathbf{c} \mathbf{d}]^{(m)} \\ &= (P_i - A_i) (\mathbf{a} \mathbf{b}, \mathbf{c} \mathbf{d})^{(m)} \\ &+ (W_i - P_i) (\mathbf{a} \mathbf{b}, \mathbf{c} \mathbf{d})^{(m+1)} \\ &+ \frac{1}{2\zeta} N_i(\mathbf{a}) \left\{ [(\mathbf{a} - \mathbf{1}_i) \mathbf{b}, \mathbf{c} \mathbf{d}]^{(m)} \right. \\ &\quad \left. - \frac{\rho}{\zeta} [(\mathbf{a} - \mathbf{1}_i) \mathbf{b}, \mathbf{c} \mathbf{d}]^{(m+1)} \right\} \\ &+ \frac{1}{2\zeta} N_i(\mathbf{b}) \left\{ [\mathbf{a}(\mathbf{b} - \mathbf{1}_i), \mathbf{c} \mathbf{d}]^{(m)} \right. \\ &\quad \left. - \frac{\rho}{\zeta} [\mathbf{a}(\mathbf{b} - \mathbf{1}_i), \mathbf{c} \mathbf{d}]^{(m+1)} \right\} \\ &+ \frac{1}{2(\zeta + \eta)} N_i(\mathbf{c}) [\mathbf{a} \mathbf{b}, (\mathbf{c} - \mathbf{1}_i) \mathbf{d}]^{(m+1)} \\ &+ \frac{1}{2(\zeta + \eta)} N_i(\mathbf{d}) [\mathbf{a} \mathbf{b}, \mathbf{c}(\mathbf{d} - \mathbf{1}_i)]^{(m+1)}, \quad (5) \end{aligned}$$

求めたい原始 2 電子積分は $[\mathbf{a} \mathbf{b}, \mathbf{c} \mathbf{d}]^{(0)}$ であり、この式の中のパラメータは以下のように表される。

$$\begin{aligned} \mathbf{P} &= \frac{\zeta_a \mathbf{A} + \zeta_b \mathbf{B}}{\zeta_a + \zeta_b}, & \mathbf{Q} &= \frac{\zeta_c \mathbf{C} + \zeta_d \mathbf{D}}{\zeta_c + \zeta_d}, \\ \mathbf{W} &= \frac{\zeta \mathbf{P} + \eta \mathbf{Q}}{\zeta + \eta}, & \zeta &= \zeta_a + \zeta_b, \\ \eta &= \zeta_c + \zeta_d, & \rho &= \frac{\zeta \eta}{\zeta + \eta} \\ \mathbf{1}_i &= (\delta_{ix}, \delta_{iy}, \delta_{iz}), & N_i(\mathbf{n}) &= n_i. \end{aligned}$$

(for $i = x, y, z$)

ここで $\mathbf{1}_i$ は角運動量指数の単位ベクトルである。また $[\mathbf{a} \mathbf{b}, \mathbf{c} \mathbf{d}]^{(m)}$ ($m \neq 0$) は補助積分と呼ばれ、次式で定義される。

$$[\mathbf{ab}, \mathbf{cd}]^{(m)} = \frac{2}{\sqrt{\pi}} \int_0^\infty \left(\frac{u^2}{\rho + u^2} \right)^m (\mathbf{ab}|u|\mathbf{cd}) du \quad (6)$$

ここで

$$(\mathbf{ab}|u|\mathbf{cd}) = \int \varphi(\mathbf{r}_2; \zeta_c, \mathbf{c}, \mathbf{C}) \varphi(\mathbf{r}_2; \zeta_d, \mathbf{d}, \mathbf{D}) \times (\mathbf{a}|\mathbf{0}\mathbf{r}_2|\mathbf{b}) d\mathbf{r}_2 \quad (7)$$

また

$$(\mathbf{a}|\mathbf{0}\mathbf{r}_2|\mathbf{b}) = \int \varphi(\mathbf{r}_1; \zeta_a, \mathbf{a}, \mathbf{A}) \varphi(\mathbf{r}_1; \zeta_b, \mathbf{b}, \mathbf{B}) \times \exp[-u^2(\mathbf{r}_1 - \mathbf{r}_2)^2] d\mathbf{r}_1. \quad (8)$$

s -型の原始デカルトガウス型関数に対する補助積分の値は、次式で求めることができる。

$$\begin{aligned} & (\mathbf{0}_A \mathbf{0}_B, \mathbf{0}_C \mathbf{0}_D)^{(m)} \\ &= 2 \sqrt{\frac{\rho}{\pi}} (\mathbf{0}_A | \mathbf{0}_B) (\mathbf{0}_C | \mathbf{0}_D) F_m(T) \\ &= \frac{1}{\sqrt{\zeta + \eta}} K(\zeta_a, \zeta_b, \mathbf{A}, \mathbf{B}) \\ &\quad \times K(\zeta_c, \zeta_d, \mathbf{C}, \mathbf{D}) F_m(T), \quad (9) \end{aligned}$$

ここで

$$T = \rho(\mathbf{P} - \mathbf{Q})^2 \quad (10)$$

$$F_m(T) = \int_0^1 t^{2m} \exp[-Tt^2] dt. \quad (11)$$

また

$$K(\zeta, \zeta', \mathbf{R}, \mathbf{R}') = \frac{\sqrt{2\pi}^{5/4}}{\zeta + \zeta'} \exp\left[-\frac{\zeta\zeta'}{\zeta + \zeta'} (\mathbf{R} - \mathbf{R}')^2\right]. \quad (12)$$

$F_m(T)$ は、いわゆる不完全ガンマ関数である。

Head-Gordon ら⁶⁾ は $[\mathbf{a}(\mathbf{b} + \mathbf{1}_i), \mathbf{cd}]$ と式 (5) の差をとって、HRR (horizontal recurrence relation) と呼ばれる新たな漸化式を導いた。HRR は次のように表される。

$$[\mathbf{a}(\mathbf{b} + \mathbf{1}_i), \mathbf{cd}]^{(m)} = [(\mathbf{a} + \mathbf{1}_i)\mathbf{b}, \mathbf{cd}]^{(m)} + (A_i - B_i)[\mathbf{ab}, \mathbf{cd}]^{(m)} \quad (13)$$

この HRR は関数の中心座標にのみ依存し、軌道指数とは無関係であるために、縮約ステップの外側で適用することが可能である。

$$(\mathbf{a}(\mathbf{b} + \mathbf{1}_i), \mathbf{cd}) = ((\mathbf{a} + \mathbf{1}_i)\mathbf{b}, \mathbf{cd}) + (A_i - B_i)(\mathbf{ab}, \mathbf{cd}) \quad (14)$$

VRR と HRR を組み合わせると、縮約ステップ内では $[\mathbf{e0}, \mathbf{f0}]$ タイプの積分のみを計算すればよい

ため、式 (5) の展開項数が 8 項から 5 項へと減少する。縮約ステップの内側の計算量は縮約長の 4 乗に比例するが、外側での計算は縮約長に依存しないため、縮約ステップの内の計算量を減少させれば、特に縮約長の大きな基底関数において ERI 計算のステップ数が大幅に減少する。さらに式 (5) や式 (14) を見れば分かるように、VRR と HRR は簡単な積和で表されているために、ベクトル化も容易である。したがって、VRR と HRR の組合せは ERI の高速計算に非常に有効である。従来の ABINIT-MP プログラム中では、VRR のみを用いたルーチンが使用されていたので、今回新たに VRR と HRR を用いたルーチンを作成した。

3. 2 電子積分計算の処理の流れ

作成した 2 電子積分計算ルーチンの処理内容を表 1 に示す。以下の手順で計算を行う。

- (1) CS ペアを角運動量指数の和 λ が大きい順に並べ換える。
- (2) CS ペアが決まると計算できる量 (ζ, \mathbf{P} あるいは $K(\zeta, \zeta', \mathbf{R}, \mathbf{R}')$) を計算してメインメモリ上に保存する。
- (3) VRR の式 (5) を用いて $(\mathbf{e0}, \mathbf{f0})$ タイプの積分を計算する。
- (4) HRR の式 (13) を用いて目的とする積分 $(\mathbf{ab}, \mathbf{cd})$ を求める。

従来のルーチンとの最も大きな違いは、用いているアルゴリズムが異なることであるが、本ルーチンの最初で行っている CS の並べ換えも、計算効率の向上に寄与するものと考えられる。この並べ換えを行うことの利点は、遭遇する 2 電子積分のタイプが少なくなることである。たとえば、CS4 重対のうち 1 つが p -型の CS で残り 3 つが s -型である場合を考える。並べ換えをししないと、 $(ps, ss), (sp, ss), (ss, ps), (ss, sp)$ の 4 通りの積分タイプが現れるため、それに応じた 4 つの積分ルーチンを作成する必要がある。一方、並べ換えをした場合には、 (ps, ss) 型の積分ルーチンを作れば事足りる。また、並べ換えなしの場合、1 つの CS4 重対を決めるたびに積分計算ルーチンの呼び出しを行う必要があるために、呼び出しのためのオーバーヘッドが大きくなるが、並べ換えを行うと、CS4 重対の型の組合せが同じである積分を 1 回のサブルーチン呼び出しで一度に計算することが可能であるため、少ない呼出負荷で計算を実行できる。このように、CS の並べ換えはコーディングの労力を少なくするだけでなく、計算コストの減少にも寄与すると考えられる。

表 1 2 電子積分計算ルーチンの概要

Table 1 Outline of evaluation of the two-electron repulsion integrals.

(The first step)	
CALL SORTCS	sorting of contracted shell pairs
DO ICS=1,NCS	first loops for the contracted shells
DO JCS=1,ICS	second loops for the contracted shells
DO IPS=1,NGI	loops for the primitive shells in ICS
DO JPS=1,NGJ	loops for the primitive shells in JCS
The calculation of the parameters \mathbf{P} , ζ and $K(\zeta, \zeta', \mathbf{R}, \mathbf{R}')$ for each pair of primitive shells	
END DO JPS	
END DO IPS	
END DO JCS	
END DO ICS	
(The second step)	
DO ITP=1,CSTYPE	first loops for all CS pair type
DO JTP=1,ITP	second loops for CS pair type
DO IJCS=1,NPPS	loops for the first pair of the contracted shells
DO KLCS=1,IPPS	loops for the second pair of the contracted shells
DO NPQ=1,NGIJ	loops for the primitive pairs in IJCS
DO NRS=1,NGKL	loops for the primitive pairs in KLCS
The evaluation of the $(\mathbf{e0}, \mathbf{f0})$ -type ERIs using the VRR	
END DO NRS	
END DO NPQ	
The transformation from $(\mathbf{e0}, \mathbf{f0})$ to $(\mathbf{ab}, \mathbf{cd})$ using the HRR	
END DO KLCS	
END DO IJCS	
END DO JTP	
END DO ITP	

表 2 (Gly)₁₀ の計算における 2 電子積分の計算時間と速度向上比Table 2 Elapsed time and speedup ratio for the ERI calculation of (Gly)₁₀.

	fragment ERI		fragment pair ERI	
	original	this work	original	this work
Elapsed time (sec.)	358.6	81.4	2684.8	384.6
Speedup ratio*		4.4		7.0

*; Original/this work

4. 性能評価

4.1 従来のルーチンとの計算速度の比較

まず、従来のルーチンと本ルーチンとの計算時間を表 2 に示す。ここでの対象分子として、10 残基の Poly Gly ((Gly)₁₀, 73 原子) を用い、2 残基 1 フラグメントとして計算を行った。C 末端をカルボキシル基でキャップし、その際末端の OH 基を C 末端グリシンが属するフラグメントに含めているため、フラグメント数は 5、フラグメントペア数は $5 \times (5 - 1) / 2 = 10$ である。本論文の性能評価はすべて HITACHI SR8000 (512PU×1GB) 上で行った。また、1 フラグメント 1PU として計算した。表 2 を見ると、本ルーチンは従来に比べて 4 倍から 7 倍高速に 2 電子積分を計算できることが分かる。系の大きさに違いのある他のいくつ

かの系でも速度比較を行ったが、同様の結果が得られた。フラグメントに比べてサイズが大きいフラグメントペアの場合において、特に速度向上の程度が顕著なことが分かる。フラグメントとフラグメントペアの 2 電子積分は、その積分数のほかには何ら相違点がないので、この速度向上の程度の違いには、各積分計算のルーチンの呼び出しのオーバーヘッドが関与していると考えられる。表 3 に、各積分タイプのサブルーチンごとの計算速度 (MFLOPS 値) と速度向上比を示すが、間接的にはあるが、この結果と表 2 で示した計算時間短縮の度合との比較から呼出負荷の影響が見てとれる。まず、すべての積分タイプに対して速度の向上が得られていることが分かる。ERI のタイプによって大きくその程度は異なるが、(ss, ss) 型で約 4 倍、最も速度向上比の小さい (pp, pp) タイプでも約 1.2 倍の高

表 3 各種分タイプのサブルーチンの実行速度 (MFLOPS), ならびに速度向上比
Table 3 Execution speeds (MFLOPS) and speedup ratios for each ERI type.

	(ss, ss)	(ps, ss)	(ps, ps)	(pp, ss)	(pp, ps)	(pp, pp)
original	14.0	21.0	35.1	32.4	69.9	118.4
this work [†]	59.7	76.9	104.1	97.9	122.3	147.2
speedup ratio*	4.3	3.7	3.0	3.0	1.7	1.2
this work [‡]	56.5	73.1	102.0	95.8	116.9	139.7
speedup ratio*	4.0	3.5	2.9	3.0	1.7	1.2

*; original/this work †; for fragment ERI ‡; for fragment pair ERI

表 4 各種分タイプのサブルーチンの呼び出し回数
Table 4 Number of call time of subroutine for each ERI type.

	(ss, ss)	(ps, ss)	(ps, ps)	(pp, ss)	(pp, ps)	(pp, pp)
original	2.8×10^6	4.5×10^6	1.8×10^6	9.4×10^5	7.8×10^5	8.6×10^4
this work	12	12	12	12	12	12

速化がなされている。しかし、各種分タイプの速度向上比がたかだか 4 倍であるのに対して、表 2 で示したように、実際の計算時間で見た場合には ERI 計算の速度向上比が 4.4 倍から 7 倍と高く、積分計算速度の向上だけで、ERI 計算時間の短縮を単純に説明することはできない。そこで全体的な ERI 計算速度の向上は、ERI 計算アルゴリズムによるものだけでなく、プログラムの構造にも起因するものと考えた方が妥当である。従来の ERI 計算ルーチンと本ルーチンでのアルゴリズム以外での最も大きな違いは、先に述べた CS 並べ換えによるサブルーチン呼出回数的大幅な減少 (表 4 参照) であるために、呼出負荷の減少が速度向上に大きく寄与していると考えられる。

4.2 フラグメント (ペア) 内の計算における並列化効率

フラグメント (ペア) 内の 2 電子積分を並列化し、台数効果による速度向上の程度を調べた。以下では、フラグメント (ペア) 間の並列計算 (大粒度の並列化) に対して、フラグメント (ペア) 内の並列化を細粒度並列化と呼ぶ。

具体的な数値を示す前に、ここで、細粒度並列化の効率について議論する。計算の並列化効率を p 、1PU 使用時の計算コストを T 、2PU 間の通信コストを L と仮定する。この場合、 n PU 使用時の全計算コスト (W_{cal}) は次式で表される。

$$W_{\text{cal}} = T \left((1-p) + \frac{p}{n} \right), \quad (15)$$

また、今回の計算の通信コストの多くは reduction 処理によるものなので、全通信コスト (W_{com}) は次のようになる。

$$W_{\text{com}} = L \log_2 n, \quad (16)$$

したがって、全コスト (W_{tot}) は次のように書ける。

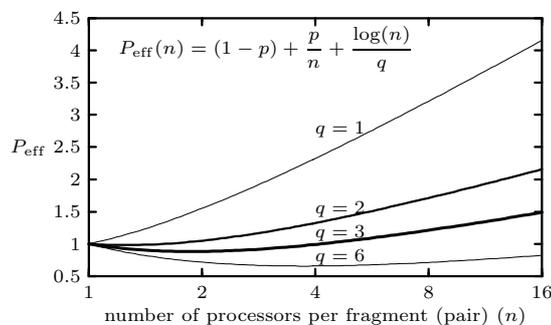


図 1 いろいろな q における、細粒度並列化による効率の変化
Fig. 1 Expected efficiencies using fine-grained parallelization with various ratio q .

$$W_{\text{tot}} = T \left((1-p) + \frac{p}{n} \right) + L \log_2 n. \quad (17)$$

ここで、 T と L の比 (T/L) としてパラメータ q を導入すると、並列化による実行効率 (P_{eff}) は、以下のように表される。

$$P_{\text{eff}} = (1-p) + \frac{p}{n} + \frac{\log_2 n}{q}. \quad (18)$$

図 1 に 1 フラグメント (ペア) あたりの PU 数 (n) に対する実行効率 (P_{eff}) の変化が、いくつかの q を用いてプロットしてある。その際、並列化効率 p の値として 0.9 を用いた。これを見ると、 q の値が 3 を超えるような場合でないと、細粒度並列化による実行時間の短縮はできないことが分かる。

実際に HITACHI SR8000 で実行させたときの 1PU あたりの通信時間や ERI の計算時間などを表 5 に示す。これは、C 末端をカルボキシル基でキャップした 15 残基の Poly Gly (Gly)₁₅、108 原子を対象分子として用いた結果である。表 5 の 1PU あたりの ERI 計算時間を式 (15) に最小自乗 fitting した結果、本ルーチンの ERI 計算そのものは約 93% の高い並列化効率

表 5 細粒度並列化を用いた場合の 1PU あたりの MPI 通信時間ならびに 2 電子積分計算時間
Table 5 MPI communication costs and ERI computation costs per processor unit
when fine-grained parallelization is used.

number of processors per fragment (pair)	1	2	4	8	16
communication(sec.)	—	1,217.5	1,269.9	1,394.9	1,787.0
ratio ^a	—	1.00	2.09	2.29	2.94
ERI calculation(sec.)	72.6	36.6	25.6	12.4	7.9
ratio ^b	1.00	0.50	0.35	0.17	0.11

a; the ratio of the communication cost to that between two processor units

b; the ratio of the computation cost to that with only one processor unit

を達成していることが分かった。一方、細粒度並列化を行った場合の通信時間の使用 PU 数に対する増加の程度は、式 (16) による理論値と同等か良い結果を示しているが、その絶対値が ERI の計算時間に比べてかなり大きいことが分かった。FMO 計算において細粒度並列化が可能な箇所の計算時間は、(Gly)₁₅ 分子の場合で 1PU あたり 7,113.6 秒という結果が得られ、表 5 の通信時間を式 (16) にあてはめて求めた 1PU あたりの通信時間が 506 秒なので、FMO 法の細粒度並列化におけるパラメータ q の値は約 1.76 と小さい。したがって、先に図 1 で示したように、フラグメント (ペア) あたりの PU 数の増加にともなう通信時間の増加が、細粒度並列化による計算時間の減少よりも大きくなるために、FMO 法での細粒度並列化による実行時間の短縮は SR8000 上ではあまり望めない。実際の計算においても、1 フラグメント (ペア) あたりの PU 数が増えるに従って、かえって実行時間が増加する、という結果になった。

5. 考 察

FMO 法におけるフラグメント (ペア) の MO 計算の並列化、すなわち大粒度並列化はすでに高い効率で達成されており、今後さらなる速度向上を目指すためには、2 電子積分などの分子積分や行列の対角化の高速化が必要である。今回、各フラグメント (ペア) の計算で最もコストのかかる 2 電子積分の高速化を行ったが、ほかにも高速化可能な部分がある。全体の速度向上をさせるためには、まずそれらを高速化するべきである。そしてすべての計算ルーチンの単体での高速化が終了した後、さらに高速化をはかる場合に、ようやくフラグメント (ペア) 内の並列化、つまり細粒度並列化の必要性が出てくる。しかし今回の結果は、通信速度がネックとなって、そう簡単には計算時間の短縮が行えないことを示している。特に、2 電子積分計算エンジン (MOE^{8)~11)} が ERI の超高速計算を行える場合などを考えると、 $q = T/L$ の分

子 T が小さくなるため、相対的に通信時間の重みが増してしまい、細粒度並列化をする意味が薄れてしまうことになる。

細粒度並列化の効率をあげたい場合には、計算コスト T を増やすか、通信時間 L を短くするかのどちらかで、それらの比 q を大きくする必要がある。通信コスト L を削減する手段としては、インターフェースの改良などにより、通信速度そのものを速くすることも 1 つの方針ではあるが、すぐに利用可能な選択肢として共有メモリの利用があげられる。この場合、通信速度そのものはメモリプロセッサ間の通信速度と等しいので、分散メモリ型の MPI による通信に比べて非常に高速である。1 フラグメント (ペア) あたりの PU 数にもよるが、メモリの使用量も減少するため、使用する計算機資源の削減にも貢献する。またアルゴリズムの改良により、通信量そのものを減らす工夫することも求められる。 q を大きくするもう 1 つの手段である計算時間を大きくすることについては、当然 FMO 計算に要する時間も長くなるため、考えなくてもよいと思われるが、計算精度やメモリ使用量の観点からは考慮する価値がある。まず、文献 4) でも触れられているとおり、Direct SCF 法を用いることで 2 電子積分をそのつど計算することを考える。この場合、たとえば SCF の収束に要するステップ数が 30 回だとすると、2 電子積分の計算時間は Direct 法でない場合に比べ単純に 30 倍になる。このように計算時間は大幅に大きくなるが、メモリや外部記憶装置上に巨大な 2 電子積分ファイルを保存しておく必要がないため、小規模な PC クラスタでも実行が可能になる。また、フラグメントのサイズを大きくして、分子積分の量そのものを増やすことも選択肢の 1 つである。先の Direct SCF 法と組み合わせると、メモリのサイズをそれほど気にせずにフラグメントのサイズを大きくできるし、そうすることにより近似の程度も良くなるために、計算精度も向上する。今後の FMO 法の高速化は、以上のことを考慮に入れて行う必要がある。また、

これらの考察から、フラグメントサイズが非常に大きな巨大分子に対する計算においては、細粒度並列化が有効であると考えられる。

6. おわりに

Ab initio FMO 法の高速化をはかるために、最も時間のかかる 2 電子積分計算のルーチンを VRR と HRR を組み合わせた方法と CS の並べ換えを導入して作成し、大きな速度向上を達成することができた。他のアルゴリズムを用いた 2 電子積分のさらなる高速化も考えられるが、2 電子積分以外の分子積分や行列対角化の高速化などを行えば、大粒度並列化による高速化はひとまず終了することになる。その次の FMO 法高速化のステップは、細粒度の高速化であるが、MPI による単純な並列化では、通信コストの増加から、積分自体が高い並列化効率を持つにもかかわらず、全体の計算時間の短縮にはつながらなかった。今後 FMO 法をより高速にし、短時間で巨大タンパクの電子状態を明らかにしていくためには、高速計算ルーチンの開発のみならず、通信コストの大幅な削減が求められる。また、フラグメントサイズが大きく、より大規模な系に対して、細粒度並列化が有効であることが示唆された。

謝辞 本研究の一部は、科学技術振興調整費の総合研究「科学技術計算専用ロジック組込み型プラットフォーム・アーキテクチャに関する研究」の支援による。また、すべての計算は、筑波先端情報計算センター (TACC) の協力による。

参 考 文 献

- 1) Kitaura, K., Sawai, T., Asada, T., Nakano, T. and Uebayasi, M.: Pair interaction molecular orbital method: An approximate computational method for molecular interactions, *Chemical Physics Letters*, Vol.312, p.319 (1999).
- 2) Kitaura, K., Ikeo, E., Asada, T., Nakano, T. and Uebayasi, M.: Fragment molecular orbital method: An approximate computational method for large molecules, *Chemical Physics Letters*, Vol.313, p.701 (1999).
- 3) Nakano, T., Kaminuma, T., Sato, T., Akiyama, Y., Uebayasi, M. and Kitaura, K.: Fragment molecular orbital: Application to polypeptides, *Chemical Physics Letters*, Vol.318, p.614 (2000).
- 4) 佐藤智之, 秋山 泰, 中野達也, 上林正巳, 北浦和夫: *Ab initio* ベア近似法による並列分子軌道計算プログラム ABINIT-MP の作成と性能評価, 情報処理学会誌: ハイパフォーマンスコンピュー

ティングシステム, Vol.41, p.104 (2000).

- 5) 中野達也, 高島 一, 長嶋雲兵: タンパク質の第一原理電子状態計算—生体化学反応の理論的解析を目指して, 日本シミュレーション学会誌: シミュレーション, Vol.19, No.4, p.37 (2000).
- 6) Head-Gordon, M. and Pople, J.A.: A method of two-electron Gaussian integral derivative evaluation using recurrence relations, *Journal of Chemical Physics*, Vol.89, p.5777 (1988).
- 7) Obara, S. and Saika, A.: Efficient recursive computation of molecular integrals over Cartesian Gaussian functions, *Journal of Chemical Physics*, Vol.84, No.7, p.3963 (1986).
- 8) 村上和彰, 小原 繁, 長嶋雲兵, 網崎孝志, 田辺和俊, 北尾 修, 北村一泰, 高島 一, 宮川宣明, 稲畑深二郎, 山田 想: *CICSJ bulletin*, Vol.16, p.6 (1998).
- 9) 小原 繁, 村上和彰, 長嶋雲兵, 網崎孝志, 田辺和俊, 北尾修, 北村一泰, 高島 一, 宮川宣明, 稲畑深二郎, 山田 想: *CICSJ bulletin*, Vol.16, p.2 (1998).
- 10) 稲畑深二郎, 山田 想, 大澤 拓, 沖野晃一, 富田裕人, 橋本浩二, 早川 潔, 宮川宣明, 村上和彰: 電子情報通信学会技術報告, Vol.IDC98-21, p.77 (1998).
- 11) Nagashima, U., Obara, S., Murakami, K., Yoshii, Y., Shirakawa, S., Amisaki, T., Kitamura, K., Takashima, H., Kitao, O. and Tanabe, K.: *Computational Materials Science*, Vol.14, p.132 (1999).

(平成 13 年 1 月 31 日受付)

(平成 13 年 5 月 10 日採録)



稲富 雄一

平成 10 年筑波大学大学院化学研究科博士課程修了。理学博士。同年同大学化学系技官, 平成 12 年(株)富士総合研究所入社, 現在に至る。日本化学会会員。



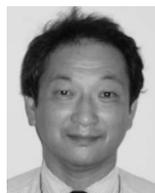
中野 達也

平成 4 年筑波大学大学院化学研究科博士課程修了。理学博士。同年厚生省国立衛生試験所(現国立医薬品食品衛生研究所)入所。平成 12 年国立医薬品食品衛生研究所化学物質情報部主任研究官。現在に至る。計算機を用いた生体系の理論的研究に従事。日本化学会, 日本薬学会各会員。



北浦 和夫

昭和 51 年大阪市立大学大学院理学研究科化学専攻博士課程修了。理学博士。同年日本学術振興会奨励研究員。昭和 52 年同特定領域奨励研究員。昭和 54 年分子科学研究所助手。昭和 56 年大阪市立大学理学部助手。平成元年岡崎国立共同研究機構分子科学研究所電子計算機センター助教授。平成 5 年大阪府立大学総合科学部教授。現在に至る。共役系分子の励起状態の理論的研究, 遷移金属錯体の構造と反応性の理論的研究, 分子間相互作用と分子間ポテンシャルの理論的研究, 分子シミュレーションによる水素結合性液体の構造・物性の研究等に従事。



長嶋 雲兵(正会員)

昭和 58 年北海道大学大学院理学研究科博士後期課程化学第二専攻単位取得満期退学, 理学博士。同年岡崎国立共同研究機構分子科学研究所電子計算機センター助手, 平成 4 年お茶の水女子大学理学部情報科学科助教授, 平成 8 年同大学教授。平成 10 年通産省工業技術院物質工学工業技術研究所基礎部理論化学研究室長, 平成 11 年同産業技術融合領域研究所計算科学研究グループ長, 平成 13 年経済産業省産業技術総合研究所先端情報計算センター情報基盤研究開発室室長, 現在に至る。専門分野は, 理論化学, 情報化学, 大規模数値計算, 広域分散並列処理, 計算型データベースシステム開発。日本化学会, 日本応用数理学会, IEEE, ACM 各会員。