C-008

GPU を用いた Head-Gordon-Pople アルゴリズムの高速化
A GPU Implementation of Head-Gordon-Pople Algorithm

鈴木 寛太[1)] 伊藤 靖朗[1)] 藤井 晴斗[1)] 横川 宜矢[1)] 辻 聡樹[1)2)] 中野 浩嗣[1)] 笠置 明彦[2)]
Kanta Suzuki Yasuaki Ito Haruto Fujii Nobuya Yokogawa Satoki Tsuji Koji Nakano Akihiko Kasagi

1 はじめに

量子化学計算は電子状態を基に分子のエネルギーなどを計算する手法であるが，計算コストが非常に大きいという課題が存在する．中でも二電子反発積分と呼ばれる処理は，計算時間の多くを占めるボトルネックとなる．本研究では並列処理が可能な GPU を用いて，二電子反発積分の計算アルゴリズムの 1 つである Head-Gordon-Pople アルゴリズム (HGP アルゴリズム)[1] を高速化する手法を提案する．

2 二電子反発積分

2.1 Head-Gordon-Pople アルゴリズム

電子の軌道を表す際に用いる基底関数 $\chi(\boldsymbol{r})$ 及びガウス型軌道 $G(\boldsymbol{r})$ はそれぞれ

$$\chi(\boldsymbol{r}) = \sum_{i}^{K} d_i G_i(r) \tag{1}$$

$$G_i(\boldsymbol{r}) = N x_A^{a_x} y_A^{a_y} z_A^{a_z} \exp\left(-\alpha_i r_A^2\right) \tag{2}$$

と定義される．さらに，4 つの基底関数 $\chi_\mu, \chi_\sigma, \chi_\lambda, \chi_\sigma$ に関する二電子反発積分 $(\mu\nu|\lambda\sigma)$ は

$$(\mu\nu|\lambda\sigma) = \iint \chi_\mu(\boldsymbol{r}_1)\chi_\nu(\boldsymbol{r}_1)\frac{1}{r_{12}}\chi_\lambda(\boldsymbol{r}_2)\chi_\sigma(\boldsymbol{r}_2)d\boldsymbol{r}_1 d\boldsymbol{r}_2 \tag{3}$$

と定義される．量子化学計算では，あらゆる 4 つ基底関数の組に対して二電子反発積分を計算する. 即ち対象となる分子が M 個の基底関数で表されるとき，M^4 回の二電子反発計算が必要となる．

また，式 (3) に式 (1) を適用することで，$(\mu\nu|\lambda\sigma)$ は

$$(\mu\nu|\lambda\sigma) = \sum_{a=1}^{K_\mu}\sum_{b=1}^{K_\nu}\sum_{c=1}^{K_\lambda}\sum_{d=1}^{K_\sigma} d_{\mu a} d_{\nu b} d_{\lambda c} d_{\sigma d} [ab|cd] \tag{4}$$

$$[ab|cd] = \iint G_{\mu a}(\boldsymbol{r}_1)G_{\nu b}(\boldsymbol{r}_1)\frac{1}{r_{12}}G_{\lambda c}(\boldsymbol{r}_2)G_{\sigma d}(\boldsymbol{r}_2)d\boldsymbol{r}_1 d\boldsymbol{r}_2 \tag{5}$$

と式変形でき，各基底関数を構成する 4 つのガウス型軌道 $G_{\mu a}, G_{\sigma b}, G_{\lambda c}, G_{\sigma d}$ に関する二電子反発積分 $[ab|cd]$ の線形和となる．ここで，基底関数に対し定義される積分 $(\mu\nu|\lambda\sigma)$ を縮約積分と呼び，一方でガウス型軌道に対し定義される積分 $[ab|cd]$ を原始積分と呼ぶ．

二電子反発積分を数値的に計算する手法として，これまで様々なアルゴリズムが開発されてきた．本研究で高速化を行う Head-Gordon-Pople アルゴリズム (HGP アルゴリズム)[1] はその 1 つであり，Horizontal Recurrence Method(HRR) および Vertical Recurrence Relation(VRR) の二種類の漸化式を導入することで，原始積分の結果を再帰的に求める．ここで，本研究ではこれらの漸化関係をすべて展開し，さらに式を整理したソースコードを用いて，原始積分の計算を行っている．

1） 広島大学大学院 先進理工系科学研究科
2） 富士通株式会社 コンピューティング研究所

2.2 先行研究

二電子反発積分の GPU 実装に関する先行研究として，Ivan らによる研究があげられる [2]．Ivan らは，縦・横軸にそれぞれ primitive-shell ペアを対応させた行列によって，二電子反発積分計算をモデル化した．ここで primitive-shell とは，同じ原子核座標 $\{x_A, y_A, z_A\}$ および指数部 α を持ち，かつ角運動量和 $l = a_x + a_y + a_z$ が等しいガウス型軌道の集合として定義される．行列によるモデル化の例を図 1 に示す．

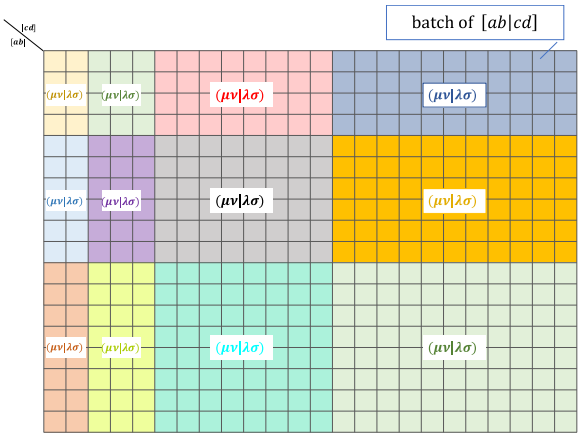

図 1 行列を用いた二電子反発積分のモデル化

行列の各要素について，その行番号と列番号から，4 つの primitive-shell が得られる．Ivan らは 4 つの primitive-shell のそれぞれの要素であるガウス型軌道の組み合わせに対する原始積分をまとめて batch と定義した．そして，二電子反発積分の GPU 実装アルゴリズムとして，1 つの batch に対し 1 つのスレッドを割り当てる 1T1B(1 Thread ⟷ 1 Batch) を提案した [2]. 即ち，図 1 の行列において各要素が 1 つの batch に対応している．

ここで，行列内で同じ色で塗られた要素 (batch) 内の原始積分 $[ab|cd]$ は，同じ縮約積分 $(\mu\nu|\lambda\sigma)$ に寄与するものとする．すると二電子反発積分は，行列の全要素を計算し，同じ縮約積分に寄与する要素ごとに総和を求める処理に帰着できる．この時，同じ縮約積分に寄与する要素に割り当てられた $K_\mu \cdot K_\nu \cdot K_\lambda \cdot K_\sigma$ 個のスレッドは，自身の結果をそれぞれ Global Memory 上の結果配列の同じ要素に対し排他的に加算を行うことで，原始積分 $[ab|cd]$ の値の集約を行う．

1T1B アルゴリズムの流れを Algorithm 1 に示す．本手法は非常に有力であり，Ji らが行った近年の研究 [3] にも用いられている．一方で，Global Memory 上の結果配列の各要素に対し，$K_\mu \cdot K_\nu \cdot K_\lambda \cdot K_\sigma$ 個のスレッドが排他的に加算を行うため，原始積分の値の集約にかかるオーバーヘッドが大きくなるという課題が存在する．

Algorithm 1 既存の GPU 実装 (1T1B アルゴリズム)

```
for each 行列の各要素 (batch) in parallel with thread do
    for each [ab|cd] do
        HGP アルゴリズムによる [ab|cd] の計算
        結果配列の対応する箇所に [ab|cd] の値を排他的に加算
    end for
end for
```

3 提案手法

提案手法では，モデル化した行列を 32 × 32 の小行列に分割する．この 32 × 32 の小行列をタイルとし，1 タイルに 1 ワープ (32 スレッド) を割り当てて計算を行う．提案手法の概要を図 2 に，アルゴリズムを Algorithm 2 に示す．示す．ただし簡略化のため，図 2 では 6 × 6 のタイルに分割した例を示している．

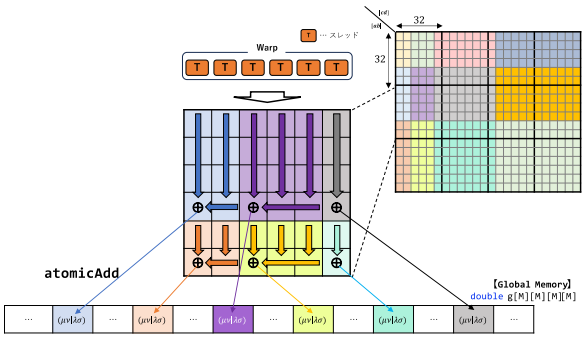

図 2 行列モデルにおける提案手法の概要

Algorithm 2 提案手法

```
for each 行列の 32 × 32 要素 (タイル) in parallel with warp do
    for each タイルの各列 in parallel with thread do
        for each タイルの各行 do
            for each [ab|cd] do
                HGP アルゴリズムによる [ab|cd] の計算
                [ab|cd] の値を，自身のレジスタ変数に加算
                if 同じ縮約積分に寄与する矩形の最終行 or タイルの最終行 then
                    Warp Shuffle を用いて，レジスタ変数の値を縮約積分ごとに集約
                    if 値が集約されたスレッド then
                        結果配列の対応する箇所に [ab|cd] の値を排他的に加算
                    end if
                    レジスタ変数に 0.0 を設定
                end if
            end for
        end for
    end for
end for
```

ワープ内の 32 スレッドはタイルの各列に割り当てられ，担当する列の 32 要素を逐次的に計算する．この時，各要素の計算結果をレジスタ変数に加算していくことで，原始積分に関する縦方向の集約を行う．加えて，寄与する縮約積分が変わる，即ち Global Memory への書込み先のインデックスが変わる時点で Warp Shuffle 命令を実行し，レジスタ変数の値を書込み先のインデックスが同じスレッドごとに横方向に集約する．この時，レジスタ変数の値は縮約積分ごとに 1 つのスレッドに集められ，そのスレッドのみが Global Memory への排他的加算を行う．この操作は，タイルの最終行においても実行される．

提案手法ではこの処理をタイルの最終行まで繰り返し，各ワープが縦・横方向の集約を繰り返しながら，タイル内の全要素を計算していく．これにより，Global Memory への排他的加算という高コストな処理が，一部レジスタ変数への加算や Warp Shuffle といった低コストの処理で置き換えられ，原始積分 $[ab|cd]$ の値の集約にかかるオーバーヘッドを大幅に削減することができる．

4 性能評価

本章では様々な分子に対して，CPU 実装，既存 GPU 実装 [2] 及び提案 GPU 実装の 3 つの手法について性能評価を行う．なお本研究において，基底関数及びガウス型軌道のパラメータ値は Pritchard らが公開する値 [4] を用いる．また，GPU には NVIDIA A100 を，CPU には AMD EPYC 7702 を用いた．

様々な分子に対する，二電子反発積分の実行時間を表 1 に示す．この結果より，提案した GPU 実装は，行列の全マスを逐次的に計算する CPU 実装と比較して最大 1819.80 倍，既存の GPU 実装と比較して最大 2.14 倍の高速化を達成したことが分かる．

表 1 二電子反発積分計算の実行時間 [ms]

テストケース	CPU	GPU		高速化率	
		既存 [2]	提案	CPU/提案	既存/提案
$(H_2O)_{10}$ / 6-31G	90507.96	102.14	83.32	1086.27	1.23
$C_{12}H_{10}N_2$ / STO-6G	458063.54	514.17	292.64	1565.28	1.76
$(MgCl_2)_4$ / STO-6G	1283380.98	1185.09	705.23	1819.80	1.68
K_2SO_4 / ANO-R0	1297618.25	1872.85	876.51	1480.44	2.14

5 まとめ

本研究では，量子化学計算において計算時間の多くを占める二電子反発積分と呼ばれる処理について，Head-Gordon-Pople アルゴリズムを用いた新たな並列計算アルゴリズムを提案し，GPU 上で実装を行った．その結果，CPU を用いた逐次計算と比較して最大 1819.80 倍，既存の GPU 実装 [2] と比較して最大 2.14 倍の高速化を達成した．

参考文献

[1] Martin Head-Gordon and John A. Pople. A method for two-electron Gaussian integral and integral derivative evaluation using recurrence relations. *The Journal of Chemical Physics*, Vol. 89, No. 9, pp. 5777–5786, 1988.

[2] Ivan S. Ufimtsev and Todd J. Martínez. Quantum chemistry on graphical processing units. 1. Strategies for two-electron integral evaluation. *Journal of Chemical Theory and Computation*, Vol. 4, No. 2, pp. 222–231, 2008.

[3] Ji Qi, Yingfeng Zhang, and Minghui Yang. A hybrid CPU/GPU method for Hartree- Fock self-consistent-field calculation. *The Journal of Chemical Physics*, Vol. 159, No. 10, p. 104101, 09 2023.

[4] Benjamin P. Pritchard, Doaa Altarawy, Brett Didier, Tara D. Gibson, and Theresa L. Windus. New Basis Set Exchange: An open, up-to-date resource for the molecular sciences community. *Journal of Chemical Information and Modeling*, Vol. 59, No. 11, pp. 4814–4820, 2019.