Thu. Jun 1, 2023

Room D

OS03 ポストムーア時代に重要となる計算技術：最新研究と将来展望

[D-09] OS03 ポストムーア時代に重要となる計算技術：最新研究と将来展望

座長:大島 聡史(九州大学)
2:30 PM - 3:45 PM Room D (2F Conference Room 201B)

[D-09-01] GPUを用いた非圧縮性流体シミュレーションにおけるデータ構造 AoSと SoAに着目した性能評価

*斉 済[1]、小野 謙二[1]、大島 聡史[1] （1. 九州大学）

2:30 PM - 2:45 PM

[D-09-02] 数値計算ライブラリの自動チューニングにおけるXAI適用の試み

青木 将太[1]、*片桐 孝洋[1]、大島 聡史[2,1]、永井 亨[1]、星野 哲也[1] （1. 名古屋大学、2. 九州大学）

2:45 PM - 3:00 PM

[D-09-03] 「不老」 Type II上で cuQuantum量子シミュレータを用いた相対論的量子化学計算の事例

杉﨑 研司[1,2,3]、Prasannaa V. S.[3]、大島 聡史[4]、片桐 孝洋[5]、森野 慎也[6]、*望月 祐志[7,8]、Sahoo B. K.[9]、Das B. P.[3,10] （1. 慶應義塾大学、2. JSTさきがけ、3. CQuERE (印)、4. 九州大学、5. 名古屋大学、6. エヌビディア合同会社、7. 立教大学、8. 東京大学、9. PRL (印)、10. 東京工業大学）

3:00 PM - 3:15 PM

[D-09-04] 量子アニーリングを用いた固体構造解析手法の開発

*本田 理央[1]、遠藤 克浩[2]、鈴木 雄大[1]、松田 佳希[3]、田中 宗[1]、村松 眞由[1] （1. 慶應義塾大学、2. 産業技術総合研究所、3. Fixstars Amplify）

3:15 PM - 3:30 PM

[D-09-05] Factorization Machineを用いた量子アニーリングによる Phase-fieldモデルのハミルトニアン補正項構築手法の開発

*青木 汐里[1]、遠藤 克浩[2]、松田 佳希[3]、関 優也[1]、田中 宗[1]、村松 眞由[1] （1. 慶應義塾大学、2. 産業技術総合研究所、3. Fixstars Amplify）

3:30 PM - 3:45 PM

OS03 ポストムーア時代に重要となる計算技術：最新研究と将来展望

[D-09] OS03 ポストムーア時代に重要となる計算技術：最新研究と将来展望

座長:大島 聡史(九州大学)

Thu. Jun 1, 2023 2:30 PM - 3:45 PM Room D (2F Conference Room 201B)

[D-09-01] GPUを用いた非圧縮性流体シミュレーションにおけるデータ構造 AoSとSoAに着目した性能評価

*斉 済[1]、小野 謙二[1]、大島 聡史[1] （1. 九州大学）

2:30 PM - 2:45 PM

[D-09-02] 数値計算ライブラリの自動チューニングにおける XAI適用の試み

青木 将太[1]、*片桐 孝洋[1]、大島 聡史[2,1]、永井 亨[1]、星野 哲也[1] （1. 名古屋大学、2. 九州大学）

2:45 PM - 3:00 PM

[D-09-03] 「不老」 Type II上で cuQuantum量子シミュレータを用いた相対論的量子化学計算の事例

杉﨑 研司[1,2,3]、Prasannaa V. S.[3]、大島 聡史[4]、片桐 孝洋[5]、森野 慎也[6]、*望月 祐志[7,8]、Sahoo B. K.[9]、Das B. P.[3,10] （1. 慶應義塾大学、2. JSTさきがけ、3. CQuERE (印)、4. 九州大学、5. 名古屋大学、6. エヌビディア合同会社、7. 立教大学、8. 東京大学、9. PRL (印)、10. 東京工業大学）

3:00 PM - 3:15 PM

[D-09-04] 量子アニーリングを用いた固体構造解析手法の開発

*本田 理央[1]、遠藤 克浩[2]、鈴木 雄大[1]、松田 佳希[3]、田中 宗[1]、村松 眞由[1] （1. 慶應義塾大学、2. 産業技術総合研究所、3. Fixstars Amplify）

3:15 PM - 3:30 PM

[D-09-05] Factorization Machineを用いた量子アニーリングによる Phase-fieldモデルのハミルトニアン補正項構築手法の開発

*青木 汐里[1]、遠藤 克浩[2]、松田 佳希[3]、関 優也[1]、田中 宗[1]、村松 眞由[1] （1. 慶應義塾大学、2. 産業技術総合研究所、3. Fixstars Amplify）

3:30 PM - 3:45 PM

計算工学講演会論文集 Vol. 28 (2023 年 5 月)
計算工学会

GPUを用いた非圧縮性流体シミュレーションにおけるデータ構造AoSとSoAに着目した性能評価

Performance Evaluation of Incompressible Fluid Simulation on GPUs focused on the Data Structures of AoS and SoA

斉済[1)]小野謙二[2)]大島聡史[3)]
Ji Qi, Kenji Ono and Satoshi Ohshima

[1)]九州大学 システム情報科学府（〒 819-0395 福岡県福岡市西区元岡 744, E-mail: qi.ji.558@s.kyushu-u.ac.jp）
[2)]九州大学 情報基盤研究開発センター（〒 819-0395 福岡県福岡市西区元岡 744, E-mail: keno@cc.kyushu-u.ac.jp）
[3)]九州大学 情報基盤研究開発センター（〒 819-0395 福岡県福岡市西区元岡 744, E-mail: ohshima@cc.kyushu-u.ac.jp）

Computational fluid dynamics codes make use of large arrays to represent multi-dimensional vector fields. On GPUs, memory accesses can be optimized using the memory coalescing technique, however, due to the three-dimensional nature of most CFD problems, it is impossible to arrange all memory accesses into continuous blocks. In this study, we evaluate the performance of using AoS and SoA to represent multi-dimensional vector fields in an in-house incompressible fluid simulation code on multi-GPUs, and estimate the influence of these data structures on different operations within the simulation.
Key Words : *CFD, HPC, GPU accelerated simulation, SoA and AoS*

1. はじめに

数値流体力学（CFD）は，微分方程式の数値解法を用いて流体の支配方程式を解き，流体の動きを解明する方法であり，風洞実験とともに様々な分野で大きな役割を持つ方法として知られている．

近年，GPU の性能向上，および開発・実行環境の整備に伴い，GPU を用いた CFD シミュレーションは増加している．しかし，不適切な実装手法を使えば，GPU の性能を十分に生かせない恐れがある．数値流体力学問題には速度などのベクトル場が多く存在し，ベクトル場の表現はプログラムのメモリー参照に大きな影響を与え，シミュレーションの性能に非常に重要だといえる．本研究は，ベクトル場の Array of Structures (AoS) と Structure of Arrays (SoA) 表現に着目し，それらのデータ構造の GPU を用いた CFD シミュレーションへの影響を評価する．

2. 非圧縮性流体シミュレーションの基礎

(1) 支配方程式

空気，水などの非圧縮性流体に対し，運動量保存を表すナビエストークス方程式は

$$\frac{\partial \boldsymbol{u}}{\partial t} + \boldsymbol{u}\nabla\cdot\boldsymbol{u} = -\nabla p + \frac{1}{Re}\nabla^2\boldsymbol{u} \tag{1}$$

となる．

質量保存を表す連続の式は

$$\nabla\cdot\boldsymbol{u} = 0 \tag{2}$$

となる．

(2) 数値計算法

本研究では，圧力・速度連成法である fractional step（FS）法 [1] を用いる．FS 法は，1 つの Δt での時間積分を 2 段階に分け．第 1 段階は，圧力の影響のない「擬似的な速度」を求める．

$$\boldsymbol{u}^* = \boldsymbol{u} + \Delta t(-\boldsymbol{u}\nabla\cdot\boldsymbol{u} + \frac{1}{Re}\nabla^2\boldsymbol{u}) \tag{3}$$

第 2 段階は，擬似的な速度場に圧力を加える「projection」操作を行う．

$$\boldsymbol{u}_{next} = \boldsymbol{u}^* - \Delta t\nabla p \tag{4}$$

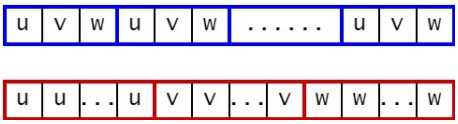
式 (4) に (2) を代入すると，圧力のポアソン方程式を導ける．

$$\nabla^2 p = \frac{\nabla\cdot\boldsymbol{u}^*}{\Delta t} \tag{5}$$

したがって，コロケート格子において，1 つの Δt での時間積分は，下記の流れになる．

Algorithm 1: FS time integration over Δt on collocated grid

1 Calculate $\boldsymbol{u}^*$;
2 Interpolation $\boldsymbol{u}^*$ from grid points to grid faces;
3 Calculate $\nabla\cdot\boldsymbol{u}^*$ using face values;
4 Solve Poisson equation;
5 Project p on both grid points and faces;



図–1 AoS（上）と SoA（下）

3. ベクトル場の SoA と AoS 表現

前述のように，FS 法には速度，擬似的な速度，および座標などのベクトル場が存在している．ベクトル場は多次元であるが，メモリーは 1 次元のアドレスで管理される．多次元ベクトル場のデータ構造は，メモリーでの並び方により，AoS と SoA に分けられる（図-1）．

速度ベクトル $\boldsymbol{u} = [u, v, w]$ を例として，AoS と SoA の構造を説明する．AoS は，1 つの計算格子の u, v, w 各成分を連続の要素で保存し，これを「structure」と呼ぶ．各格子の structure は連続に配置され，AoS になる．SoA は，u, v, w をそれぞれ 1 つの配列で保持し，u 配列には各計算格子の u 成分を連続に配置している．u, v, w 3 つの配列は SoA になる．

表-1 には，FS 法各ステップが参照するデータの種類を示す．すべてのステップは，ベクトルとスカラー 2 種類の変数をメモリーから読み出す．ポアソン方程式を用いた圧力の計算は，圧力場の値を変更するため，スカラー変数をメモリーに書き戻す．そのほかのステップは速度場，すなわちベクトル変数をメモリーに書き戻す．

表–1 一般座標系における FS 法が参照するデータの種類

	READ	WRITE
calculate $\boldsymbol{u}^*$	vector, scalar	vector
interpolate $\boldsymbol{u}^*$	vector, scalar	vector
poisson eq.	vector, scalar	scalar
project p at grid point	vector, scalar	vector
project p at grid face	vector, scalar	vector

4. GPU でのメモリー参照

(1) GPU 並列の仕組み

GPU は，大量の演算コアを持ち、多数のスレッドを同時に実行する．GPU のスレッドは並列処理の最小単位であるが，命令スケジューリングの最小単位は Warp である．Warp は 32 スレッドで構成され，それらのスレッドは常に同一命令を実行する．

GPU でのメモリー参照を効率的に行うには，下記の 2 つの点が重要である [3]：

a) Coalesced メモリー参照

Warp がメモリー参照命令を実行するとき，32 スレッドの参照先アドレスが連続であれば，その 32 回のメモリー参照を 1 回に coalescing でき，性能を大きく向上できる．

b) メモリー参照の局所性

関連性の強いデータをメモリーの中で互いに近くに配置すると，GPU のキャッシュを活用でき，メモリーへのアクセスを回避し，性能を向上できる．

(2) AoS と SoA へのメモリー参照

前述の coalesced メモリー参照と局所性を考えると，AoS と SoA へのメモリー参照は表-2 に示した特徴がある．

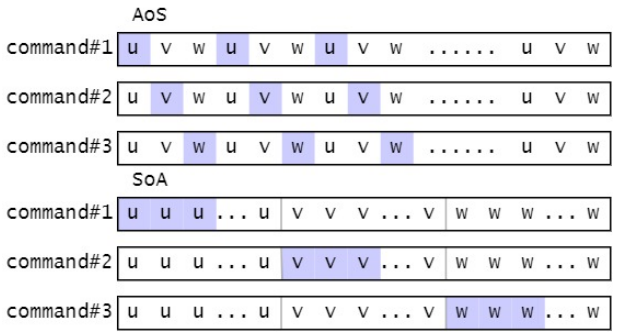
ここで，下記の命令を例としてそれらの特徴を説明する．1 つのスレッドは，1 つの計算格子の速度各成分を読み，それに基づき計算を行う．

```
GPU kernel start
    i = thread_id
    read u@grid_i /* command#1 */
    read v@grid_i /* command#2 */
    read w@grid_i /* command#3 */
    /* do some calculation */
GPU kernel end
```

図-2 には，それら命令のメモリー参照のパターンを示す．ここで，便利のために，Warp のスレッド数を 3 とし，青色の部分はスレッド 1~3 が参照するメモリーである．

表–2 AoS と SoA へのメモリー参照

data structure	coalescing	locality
AoS	not coalesced	local
SoA	coalesced	not local



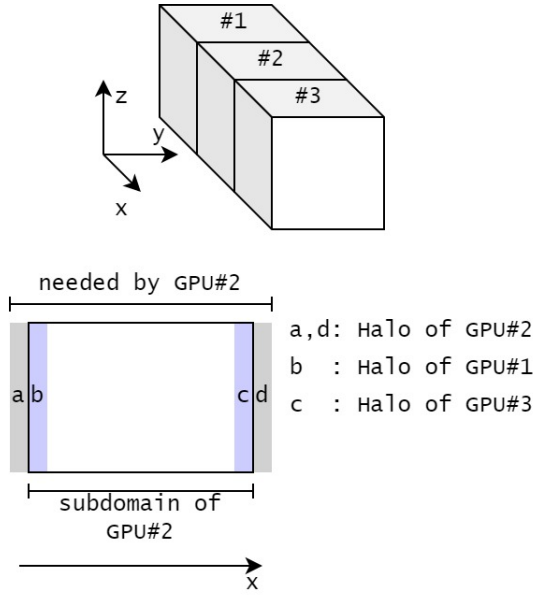
図–2 メモリー参照のパターン

AoS の場合，成分 u のメモリーは連続でないため，その Warp の命令ごとのメモリー参照は coalesced ではない．その一方，格子 1~3 のデータは連続に配置されているので，その 3 つの命令は比較的に小さい範囲に参照している．

SoA の場合，各成分はそれぞれ連続のメモリーに配置されているため，その Warp の各メモリー参照命令は coalescing できる．しかし，同じ格子の各成分はメモリーの中に散在しているため，キャッシュのメリットを引き出すことが難しい．

(3) GPU 間通信のメモリー参照

GPU 1 基のメモリーはおおむね十数 GB であり，大規模シミュレーションの場合ではメモリー不足の恐れがある．計算負荷を複数の GPU に分散するために，本



図–3 領域分割と Halo

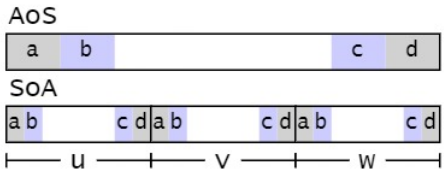
研究は計算領域を x 軸に沿って分割し，1 つのサブ領域を 1 基の GPU に割り当てる．

ステンシル計算において，ある格子点の計算には隣接する格子点の情報が必要であるため，分割されたサブ領域の両端に Halo と呼ばれる部分がある．Halo 部のデータは，本 GPU の計算に必要であるが，Halo の計算は隣の GPU が担当している．したがって，GPU の間では，Halo 部分のデータを交換する必要がある．図-3 には，GPU#2 のサブ領域と Halo を示している．

Halo	SEND	RECEIVE
a	GPU#1	GPU#2
b	GPU#2	GPU#1
c	GPU#2	GPU#3
d	GPU#3	GPU#2

表–3 Halo の送信と受信

前述のデータ構造により，GPU 間通信のためのバッファーが異なる点には注意が必要となる．速度場を例としてバッファーのメモリーレイアウトを説明する．図-4 には，各 Halo に対応しているバッファーのレイアウトを示している．AoS の場合，バッファーは Halo と同じく，速度場を保持するメモリーの両端に位置する．それに対し，SoA の場合，速度の各成分はそれぞれ各自のバッファーを持ち，同じ Halo に対応しているバッファーが，3 セグメントに分けられ，メモリーの中に散在する．したがって，AoS の場合では，2 回の SEND と 2 回の RECEIVE が必要であり，SoA の場合では SEND と RECEIVE 各 6 回が必要となる．その一方，Halo のサイズはデータ構造にかかわらず，一定であるため，通信量には差がない．



図–4 バッファーのレイアウト

5. 実験による性能評価

(1) 実験環境と問題設定

性能評価の数値実験として，風車 2 基の風況解析を行う [2]．全計算格子数は約 9 千万であり，無次元時刻 $T = 10$ まで時間積分を計算する．そのシミュレーションは，九州大学スーパーコンピューター ITO のサブシステム B の 1 ノードで行う．システムの諸元は表-4 に示す．プログラムは，NVidia HPC SDK（NVHPC）が提供した OpenMPI 環境を用いてコンパイルし，4 つのプロセス（GPU 4 基）で実行する．

表–4 ITO サブシステム B の 1 ノード

CPU	Intel Xeon Gold 6140 18 core ×2
GPU	NVIDIA Tesla P100 ×4
performance	CPU: 2649.6 GFLOPS/node GPU: 5.3 TFLOPS/GPU
memory	DDR4: 384 GB/node HBM2: 16 GB/GPU
CPU-GPU connect	PCIe Gen.3 ×16 (16 GB/s)
GPU interconnect	NVLink (20 GB/s ×1or2)

(2) 実行時間の内訳

表-5 には，実行時間の中でアルゴリズム 1 に示した各ステップに対応するカーネルの割合を示す．

表–5 実行時間の内訳

	AoS		SoA	
	time(s)	ratio	time(s)	ratio
total GPU time	2115		1519	
calculate $\boldsymbol{u}^*$	574	27.1%	335	22.1%
interpolate $\boldsymbol{u}^*$	367	17.4%	131	8.6%
poisson eq.	341	16.2%	346	22.8%
project p at grid	94	4.4%	97	6.4%
project p at face	203	9.6%	102	6.7%

SoA は AoS と比べて実行時間の約 1/4 を削減し，擬似速度の計算，内挿，および格子面での projection では大きな性能改善を示している．その一方，ポアソン方程式の計算と格子点での projection では，大きな変化がないことも明らかとなった．

6. 議論

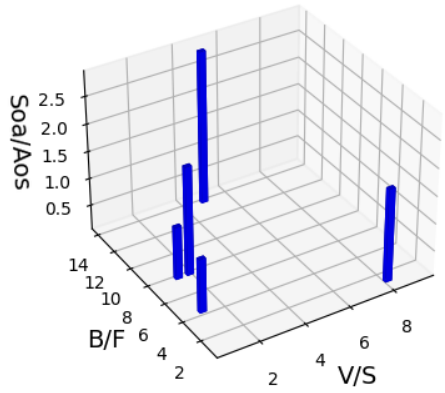
前述のように，SoA はメモリー参照を coalescing でき，シミュレーション性能の大きな改善を示している一方，ポアソン方程式の計算など，性能に変化のないカーネルも存在している．ここで，その原因について議論する．

SoA と AoS は，スカラー場のデータ構造に影響を及ぼさないため，スカラー場へのメモリー参照が比較的多いカーネルでは，SoA の効果を引き出しにくいと考えられる．また，メモリー参照と比べて，計算のほうが比較的に多いカーネルでは，メモリー参照の性能が改善されても，全体の性能には大きな違いがないことも推察される．

したがって，ここでは「V/S」と「B/F」という 2 つの値を定義する．V/S はカーネルの中で，ベクトル場に対するメモリー参照とスカラー場に対するメモリー参照の比であり，B/F はカーネルのメモリー参照の byte 数と浮動小数点演算数の比である．

表–6 V/S，B/F，および性能の比

	V/S	B/F	perf. SoA/AoS
calculate $\boldsymbol{u}^*$	8.25	1.43	1.71
interpolate $\boldsymbol{u}^*$	5	14	2.82
poisson eq.	1.2	4.51	0.99
project p at grid	1.5	8	0.97
project p at face	2	8	1.99

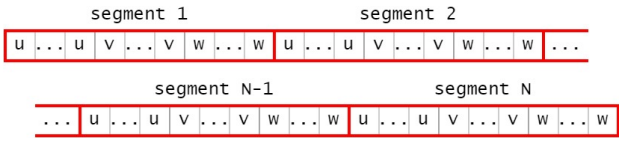


図–5 V/S，B/F，および性能の比

表-6 と図-5 には，表-5 に示した各カーネルの V/S, B/F，および SoA と AoS の性能の比を示す．図-5 に，青色の棒は各カーネルに対応し，棒の長さが SoA と AoS の性能の比を表し，棒の位置が V/S と B/F の値に対応する．表-6 と図-5 に，V/S と B/F の小さいカーネルは SoA と AoS の性能がほぼ同じであり，V/S または B/F の大きいカーネルは SoA が AoS よりいい性能を引き出せる傾向を示している．

7. 結論

本研究はマルチ GPU を用いた非圧縮性流体シミュレーションのコードに基づき，ベクトル場の AoS と SoA 表現を実装し，その性能評価を行った．



図–6 AoSoA での速度場のメモリーレイアウト

SoA と AoS はそれぞれ異なるメモリー参照のパターンを持ち，どちらも GPU メモリー参照の coalescing と局所性という 2 つの重要な点を同時に確保するのが困難であるが，シミュレーション全体としての性能は，SoA が AoS を大きく上回り，coalesced メモリー参照が優先であることを示した．

その一方，各カーネルの特徴により，SoA のカーネルごとの性能が異なる．V/S または B/F の高いカーネルでは SoA が大きな改善を示し，V/S と B/F の低いカーネルではその効果を引き出しにくい傾向がある．

8. 展望

本研究では，V/S と B/F などを計算するとき，WRITE（書き戻し）と READ（読み出し）を区別せず，どちらも 1 回のメモリー参照とする．しかし，実際には，WRITE の性能が READ を大きく下回る可能性も低くない．したがって，カーネルのメモリー参照特徴をさらに詳しく研究するには，WRITE と READ の性能の違いを考慮する必要がある．

また，メモリー参照の coalescing と局所性は，GPU におけるメモリー参照の性能に非常に重要な点であるが，本研究が対象としている SoA と AoS がその両方を同時に確保するのは難しい．それに対し，Array of Structures of Arrays (AoSoA)[4][5] が提案された．図-6 に示したように，ベクトル場を等長のセグメントに分け，1 つのセグメントを 1 つの SoA に保存し，複数の SoA が連続に並び，AoSoA になる．AoSoA は coalescing と局所性両方を活用するポテンシャルを持つと考えられ，AoSoA によるベクトル場の表現を実装し，その性能を評価することが本研究の将来の課題となっている．

参考文献

[1] 梶島岳夫: 乱流の数値シミュレーショ, 養賢堂, 1999.
[2] 内田孝紀, 小野謙二, 飯田明由, 吉村忍, 加藤千幸, 山出吉伸, 今村博, 植田祐子: スパコン版リアムコンパクトによる風車ウエイクの相互干渉に関する大規模数値シミュレーション, 風力エネルギー学会論文集, Vol.45(4), pp.71-82, 2021.
[3] Cheng, J., Grossman, M., McKercher, T.: Professional CUDA C Programming, Wrox, 2014.
[4] Fei, Y., Huang, Y.H., Gao, M.: Principles towards Real-Time Simulation of Material Point Method on Modern GPUs, arXiv:2111.00699 [cs.GR], 2021.
[5] Wang, X.L., Qiu, Y.X., Slattery, S.R., Fang, Y., Li, M.C., Zhu, S.C., Zhu, Y.X., Tang, M., Manocha, D., Jiang C.F.F.: A massively parallel and scalable multi-GPU material point method, *ACM Transactions on Graphics*, Vol.39(4), pp.1-15, 2020.

計算工学講演会論文集　Vol.28 (2023年5月)　　計算工学会

数値計算ライブラリの自動チューニングにおけるXAI適用の試み

An Adaptation of XAI to Auto-tuning for Numerical Calculation Library

青木将太[1)]，片桐孝洋[2)]，大島聡史[3)]，永井亨[4)]，星野哲也[5)]
Takahiro Katagiri, Shota Aoki, Satoshi Ohshima, Toru Nagai
1)　学(情報) 名古屋大学大学院 情報学研究科（〒464-8601 愛知県名古屋市千種区不老町 情報学研究科, E-mail: aoki@hpc.itc.nagoya-u.ac.jp)
2)　博(理) 名古屋大学情報基盤センター 教授（〒464-8601 愛知県名古屋市千種区不老町 名古屋大学情報基盤センター, E-mail: katagiri@cc.nagoya-u.ac.jp）
3) 博(工) 九州大学情報基盤研究開発センター 准教授（〒819-0395 福岡市西区元岡744, E-mail: ohshima@cc.kyushu-u.ac.jp）
4) 博(理) 名古屋大学情報基盤センター 助教（〒464-8601 愛知県名古屋市千種区不老町 名古屋大学情報基盤センター, E-mail: nagai@cc.nagoya-u.ac.jp）
4) 博(理) 名古屋大学情報基盤センター 准教授（〒464-8601 愛知県名古屋市千種区不老町 名古屋大学情報基盤センター, E-mail: hoshino@cc.nagoya-u.ac.jp）

We explain an adaptation of Explainable AI (XAI) to auto-tuning problem for parameter tuning on a PICCG solver, which is one of preconditioned sparse iterative solvers. Result from SHAP, which is an XAI tool, indicates that predicted execution time for the solver from AI output can be well-explained in our experiment.

Key Words : *Explainable AI, Auto-tuning, PICCG Solver*

1.　はじめに

人工知能（AI）が出力する答えを検証が不十分なまま利用することで，社会的な問題を引き起こすことが懸念されている．またAI出力結果については，WEB等の公知で入手可能な文書から生成されたものがあるため，自動生成された文章等は，真実が出力される保証はない．

以上のAIの現状では，AIの出力に対して，人間がなんらかの検証を事前に行わなくてはならない．しかしAIモデルからの莫大な出力を，すべて人間が確認することは現実的ではない．つまるところ，このAI出力の検証工程の自動化や，検証工数の削減手法が必須となる．そのため，AIモデルを構築する際のコストを減らす研究開発がされている．

一方，我々は数値計算分野での性能チューニング工数の削減を目的として，ソフトウェア自動チューニング（Software Auto-tuning, AT）技術[1]へのAI適用を行ってきた．ここでのAT技術は，単にモデル上のパラメタ調整をする技術ではない．AT技術は，高性能コード生成も行う性能チューニングの総合的技術であり，コンパイラ最適化やアルゴリズム選択などの上位の最適化対象も取り扱う．ここでは，性能パラメタ調整の観点にのみ着目する．

本研究では，疎行列反復解法上に現れる性能パラメタチューニング事例を例題として，AIを適用したAT機能を実現した場合における，AIが出力する予測時間に関する説明性について検証を行う．

2.　XAIツール

(1) 概要

AI出力の説明性を高めることで「信頼されるAI」を実現するためのツール（Explainable AI (XAI)ツール）がいくつか提案されている．さらにこのXAIツールは，いくつかはオープンソース化がなされており，容易に利用可能である．以下に代表的なXAIツールを説明する．

(2) LIME

幅広い学習済みモデルに適用できるXAIツールの1つとして，LIME (Local Interpretable Model-agnostic explanations)[2] が知られている．

LIMEはLocal surrogate model であり，ブラックボックスモデルの個々の予測を説明するために用いられる解釈可能なモデルである．分類器が特定の予測を行った理由を，人間が理解できるように提示する機能を持っている．それぞれの特徴がどの程度，分類に貢献しているか調べることにより，分類器の予測結果を説明することができる．分類器の予測結果を用いるため，任意の分類器に適用できるのが特徴である．

LIMEは個別の事例を説明するのに使われる「局所説明ツール」である．

(3) SHAP

LIMEに加えて良く利用されるXAIツールとして，SHAP (SHapley Additive exPlanations) [3] が知られている．SHAPは，協力ゲーム理論のシャープレイ値（Shapley Value）を機械学習に応用したものである．そのため，採用されている評価基準に，理論的な妥当性がある．シャープレイ値の計算は，厳密にすると計算量が高い．そのため，近似的にシャープレイ値を算出する手法が研究されており，SHAPで利用されている．

SHAPは全体的な傾向を説明するのに使われる，大域説明ツールである．

3. 不完全コレスキー分解前処理付CG法（PICCG法）

(1) 概要

不完全コレスキー分解（Incomplete Cholesky Decomposition, IC）は，疎な対称行列Aを係数とする連立一次方程式 $Ax = b\,(A \in \mathbb{R}^{n\times n},\ x, b \in \mathbb{R}^n)$ のような連立1次方程式を解く反復法の1つである**共役勾配法（Conjugate Gradient (CG) Method）**の前処理として活用される．IC分解前処理付きのCG法は，**PICCG法**と呼ばれる．

ICを用いて行列A を，以下の式(1)の分解をすることを考える．

$$A = U^t D\, U + R\ (U, D \in \mathbb{R}^{n\times n}) \qquad \cdots(1)$$

ここで，U:上三角行列，D:対角行列，R: AとIC分解後の$U^t D\, U$との差の行列，とする．このとき，行列の要素値が0であったAの要素が，分解行列Uにおいて，非ゼロ要素となる場合がある．これを，**fill-in**と呼ぶ．

基本的なIC分解は，fill-inをすべて棄却する実装になっている．つまり，全て要素を0要素として扱う．このことで，分解後の行列の非ゼロ要素を少なく保ち，メモリ量と計算量を減らすことを狙う．しかし反面，Aと$U^t D\, U$の差が大きく異なる場合は，うまく機能しないかもしれない．すなわち，前処理行列として機能しなくなることがある．

(2) 閾値付きIC分解前処理とアルゴリズム

ここでは，閾値付きIC分解について説明する．

式(1)の行列分解過程で，分解行列Uの非ゼロ要素について，**fill-inレベル**を持つようにする．Fill-inレベルとは，非零要素をどこまで許容するかのレベルである．通常，行当たりや対角要素当たりを対象として，行列や扱う問題対象の特性を考慮し，このレベルを定義することが多い．

本研究では，東京大学情報基盤センターの河合による実装[4]を利用する．この実装では，**最大fill-inレベル(m)**，および**閾値(t)**を設定する．

最大fill-inレベルでは，指定以下のレベルのfill-inに対して，閾値より小さいfill-in対象の要素値を0とみなして，閾値以上のfill-inを許容する．このことで，IC分解前処理と同程度以上の収束性と，より少ない非ゼロ要素数の行列に分解することで，メモリ量を低く抑えつつ高速化を狙うものである．つまり，最大fill-inレベルを増やせば，一般的に，反復回数は減る．しかし，行列が密に近づき演算量が増えるため，反復回数の削減と演算量の増加のトレードオフがある．つまりは，これらは性能チューニングパラメタになる．著者らが扱っている問題では経験的に，最大fill-inレベル2が最適であるが，当然，扱う問題の性質に依存する．

以上のアルゴリズムの概略を，図-1に記載する．ここで，$a_{i,j}$:Aのi,j要素，$d_{i,i}$:Dのi,i要素，$u_{i,j}$:Uのi,j要素，$f_{i,j}$:$u_{i,j}$のfill-inレベル，t:0とみなす閾値，m:最大fill-inレベルである．

$$d_{i,j} = a_{i,j} - \sum_{k=1}^{i-1} u_{i,k} d_{i,i} u_{k,j}$$

$$f_{i,j} = \begin{cases} 0, & a_{i,j} \neq 0 \\ f_{i,k} + f_{k,i} + 1, & else \end{cases}$$

$$u_{i,j} = \begin{cases} d_{i,j}^{-1}\left(a_{i,j} - \sum_{k=1}^{i-1} u_{i,k} d_{i,i} u_{k,j}\right), & f_{i,j} \leq m \wedge |u_{i,j}| \geq t \\ 0, & else \end{cases}$$

図-1　閾値付き IC 前処理の概要

4. 適用事例と分析

(1) 計算機環境

AIの教師データを取得するため，名古屋大学情報基盤センター設置のスーパーコンピュータ「不老」Type Ⅰサブシステム[5]を利用した．また，機械学習は，「不老」Type Ⅱサブシステム[5]のGPUを利用した．

Pythonはver. 3.6.13, Tensorflowは ver. 2.4.1, SHAPは ver. 0.39.0を利用している．

(2) PICCG法の実行時間を予想するモデル

Tensorflowを使用し，以下の入力と出力を設定する．

- 入力：
  - I. 係数行列の特徴画像（山田ら[6]の手法により作成）
  - II. 最大 fill-in レベル，閾値
- 出力：閾値付きICCG法の計算時間

畳み込み層、プーリング層:2層、全結合層:3層のCNNを作成した．モデルの概要を図-2に示す．

学習設定として，エポック数は200，バッチサイズは256，活性化関数はReLU，最適化はAdam法，損失関数は平均二乗誤差とした．

(3) データセットの作成

差分格子によってメッシュ分割された三次元領域において定常熱伝導方程式を解くアプリケーション（P3D）[6]

を対象にし，有限体積法に基づき非構造格子型のデータとして考慮したデータを利用する[7]．ここで，熱伝導率λの分布（$\lambda_2 \leqq \lambda_1$）の問題空間であり，熱伝導率が$\lambda_1$の層の中に，伝導率$\lambda_2$の層を入れ込んだものである．この$\lambda_2$の値が小さくなると，生成される係数行列の条件数が大きくなっていく．つまり悪性問題に近づき，反復解法では解きにくい問題になるので，問題の性質をλ_2で制御できる特徴を持つ．

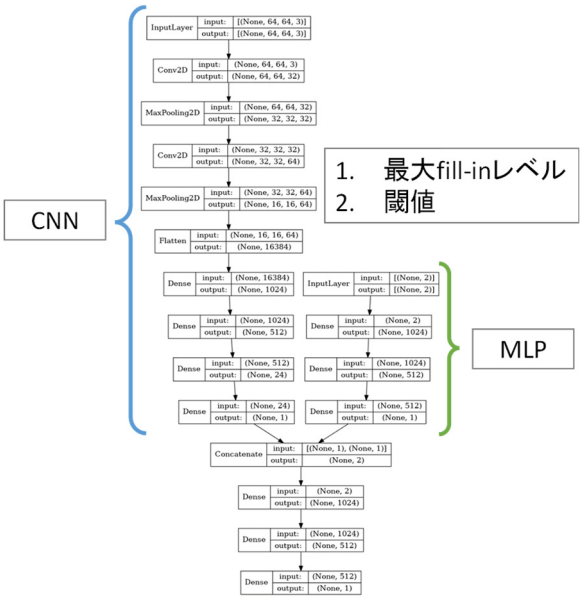


図-2 モデルの概要

教師データは，「不老」TypeIサブシステムで以下の条件で計算時間を計測して取得する．

- 係数行列
  - サイズ：4096×4096，32768×32768，262144×262144 【3種類】
  - 各サイズ条件数を変えた（伝導率λ_2を変化）【90種類】
- 最大fill-inレベル：0，1，2 【3種類】
- 閾値：0.001～0.02(0.001間隔) 【199種類】

以上から教師データ総数は，各サイズで【41,073個】となる．また，テストデータ総数，は各サイズで【10,269個】となる．

各サイズ，に閾値付きICCG法の実行時間を予測する回帰モデルを作成する．

(4) SHAPによる説明結果

まず，図-3によるテストデータ（実測実行時間）と，AIモデルによる予測結果（予測実行時間）の平均絶対誤差は0.000526であり，5%以下の誤差であったため，モデル生成は妥当である．かつ，本テストデータの範囲において，AIによるATが，誤差5%で実現できたことも意味している．

次に，この生成したAIモデルが，妥当な回答をしているか，SHAPの説明により検証する．図-3に，SHAPによる問題サイズ32768×32768の説明結果を示す．

図-3の説明から，以下の解釈ができる．

- 閾値が一定以下になると，閾値が小さいほど実行時間が短くなる傾向がある
- Fill-inレベル2の場合，閾値が0.075以下において，実行時間が短くなっていく傾向がある

以上のSHAPにより説明された傾向は，対象アルゴリズムの知見を使ったものではないことに注意する．しかしこの傾向は，閾値付きIC前処理のICCG法のアルゴリズム上，および，今回のテストデータの実測実行時間の分布傾向から妥当な解釈であった．したがって，生成したAIモデルが妥当であることを示している．

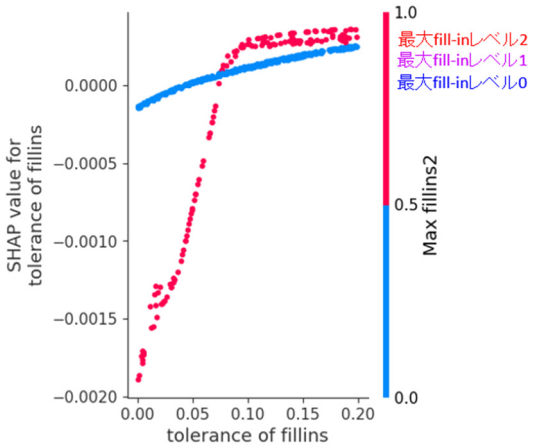


図-3 問題サイズ 32768×32768 の説明結果

5. おわりに

本研究では，説明可能なAI（XAI）のツールとして，SHAPを利用して，疎行列反復解法に現れる性能パラメタチューニングにAIを活用する自動チューニング（AT）機能を開発する場合の説明可能性について，事例をもとに結果を検証した．

検証結果から，SHAPを利用した本事例の説明については，妥当に説明されていることを確認した．また，本稿では説明していないが，SHAPにより妥当な解説がされていないことを発見し，原因を解析した．その結果，説明変数を定式化する際に，あるテクニックを適用したところ，解の精度と，SHAPによる妥当な説明が可能となる事例も発見した．本件については，当日報告する予定である．

今後の課題として，さらに多くの事例解析を行い，XAIツールが数値計算処理におけるAT機能にも有効であることを示していく必要がある．加えて，XAIツールからの説明に基づき，有効となる説明変数の追加や削減を自動で行い，AT機能の性能向上に寄与する仕組みの研究開発も必要である．

謝辞：本研究は，科学技術研究費補助金，基盤研究(S)，「(計算＋データ＋学習)融合によるエクサスケール時代

の革新的シミュレーション手法」(課題番号:19H05662), および, 学際大規模情報基盤共同利用・共同研究拠点、および、革新的ハイパフォーマンス・コンピューティング・インフラ (課題番号: jh220022) の支援による. また, PICCGソルバーの知識提供について, 東京大学情報基盤センター河合直聡 特任助教に感謝の意を表する.

参考文献

[1] T. Katagiri, and D. Takahashi: Japanese Auto-tuning Research: Auto-tuning Languages and FFT, *Proc. of the IEEE*, 106 (11), pp.2056-2067, 2018.

[2] M. T. Ribeiro, S. Singh, and C. Guestrin: Why should I trust you?: Explaining the predictions of any classifier, *Proc. of 22nd ACM SIGKDD*, pp.1135-1144, 2016.

[3] S. M. Lundberg, and S. Lee: A unified approach to interpreting model predictions, P*roc. of the 31st International Conference on Neural Information Processing Systems (NIPS' 17)*, pp. 4768-4777, 2017.

[4] 河合直聡, 中島研吾, 前処理付きクリロフ部分空間法への低/任意精度の適用, 京都大学数理解析研究所研究集会, 2022

[5] 名古屋大学情報基盤センター スーパーコンピュータ「不老」ホームページ
https://icts.nagoya-u.ac.jp/ja/sc/overview.html#type2
[2023年3月21日閲覧]

[6] K. Yamada, T. Katagiri, H. Takizawa, K. Minami, M. Yokokawa, T. Nagai, and M. Ogino: Preconditioner Auto-Tuning Using Deep Learning for Sparse Iterative Algorithms, *Proc. of 2018 Sixth International Symposium on Comput. and Networking Workshops (CANDARW)*, pp. 257-262, 2018.

[7] 中島研吾, 大島聡史, 塙敏博, 星野哲也, 伊田明弘, ICCG法ソルバーのIntel Xeon Phi向け最適化, 情報処理学会研究報告 (2016-HPC-157-16) , 2016.

計算工学講演会論文集　Vol.28 (2023年5月)　　計算工学会

「不老」 Type II上でcuQuantum量子シミュレータを用いた相対論的量子化学計算の事例

Relativistic quantum chemical calculations by using cuQuantum simulator on Flow Type II

杉崎研司[1,2,3], Prasannaa V. S[4], 大島聡史[5], 片桐孝洋[6], 森野慎也[7], 望月祐志[8,9], Sahoo B. K.[10], Das B. P.[11,12]

Kenji Sugisaki, V. S. Prasannaa, Satoshi Ohshima, Takahiro Katagiri, Shinya Morino, Yuji Mochizuki, B. K. Sahoo, and B. P. Das

1) 博(理) 慶應義塾大学大学院理工学研究科 特任准教授（〒212-0032 神奈川県川崎市幸区新川崎7-7 KBIC本館, E-mail (KS): ksugisaki@keio.jp）
2) 博(理) 慶應義塾大学量子コンピューティングセンター 研究員（〒223-8522 神奈川県横浜市港北区日吉3-14-1）
3) 博(理) TCG Crest量子工学研究教育センター 客員准教授（Sector V, Salt Lake, Kolkata 700091, India）
4) 博(理) TCG Crest 量子工学研究教育センター 助教（Sector V, Salt Lake, Kolkata 700091, India）
5) 博(工) 九州大学情報基盤研究開発センター 准教授 （〒819-0395 福岡県福岡市西区元岡 744）
6) 博(理) 名古屋大学情報基盤センター 教授 （〒464-8601 愛知県名古屋市千種区不老町）
7) 博 (工) エヌビディア合同会社 主 ML エンジニア（〒107-0052 東京都港区赤坂 2-11-7 ATT 新館）
8) 理博 立教大学理学部 教授 （〒171-8501 東京都豊島区西池袋 3-34-1, E-mail: fullmoon@rikkyo.ac.jp）
9) 理博 東京大学生産技術研究所 リサーチフェロー （〒153-8503 東京都目黒区駒場 4-6-1）
10) 理博 物理学研究所 教授 （Navrangpura, Ahmedabad 380009, India）
11) 理博 TCG Crest 量子工学研究教育センター センター長（Sector V, Salt Lake, Kolkata 700091, India）
12) 理博 東京工業大学理学部 名誉教授 （〒152-8550 東京都目黒区大岡山 2-12-1）

Recently, GPU-based quantum simulators have attracted considerable interest in the development of quantum chemical methods. NVIDIA's cuQuantum is a representative of such simulators. In this paper, we present the fully relativistic calculations for the spin-orbit splitting of $2p_{1/2}$ - $2p_{3/2}$ in boron-isoelectric systems using the Bayesian Phase Difference Estimation (BPDE) method with cuQuantum on the Flow type II supercomputer at Nagoya University.

Key Words: *Quantum computer, cuQuantum, Relativistic quantum chemical calculation, Spin-orbit splitting, Bayesian estimation, Flow supercomputer*

1. はじめに

量子コンピュータ[1]は，量子力学の原理を利用して高速な計算を可能にする新しい計算機で，ゲート型とアニーラ型に大別できますが，多分野の問題を扱える汎用性の面では前者の方が高い潜在力を持っています[2]．とりわけ，化学反応や物質の性質を理論的に予測する量子化学計算はゲート型量子コンピュータの重要な応用分野として期待されており，世界中で研究開発が活発に行われています[3-5]．しかし，現在のNISQ(Noisy Intermediate-Scale Quantum)段階の実機は，ノイズの影響が非常に大きいため，開発のプラットフォームとしては実機ではなく，量子ビット群の動作を模した量子シミュレータが使われることが多いです．また，多くの場合，量子化学計算で最終的に議論される値は全エネルギーではなく，エネルギー差であることから，エネルギー差を直接算定するアプローチが有効と考えられ，杉崎らが提案したベイズ推定を援用して2状態間のエネルギー差を直接算定する位相差推定(BPDE)[6]は有望な手法の一つです．一方，近年，計算機科学の高性能計算(HPC)の立場からも膨大なテンソル演算を処理する量子シミュレータに関心が持たれ，特にGPUによる加速を得るNVIDIAのcuQuantum SDK[7]が注目を集めています．いずれにせよ，「量子コンピュータ＋HPC」が新しい学際領域として認知され，両分野の研究者のコラボレーションの重要性も高まっています．

こうした中，量子化学分野とHPC分野の研究者の混成チームである私たちは，4成分の相対論的量子化学計算にBPDEを適用し，B原子ならびに等電子系の原子イオン(Lr^{98+}まで)の$2p_{1/2}$－$2p_{3/2}$のスピン軌道相互作用による分裂エネルギーを高い精度で系統的に評価することに成功しました．実際の計算は，NVIDIAのV100を積む名古屋大学のスーパーコンピュータ「不老」 Type II上でcuQuantumを用いて実行しました．計算スキームと計算結果の詳細はarXivで既に公開[8]されていますので，本稿ではポイントを要約しつつ，HPC的な側面を補完する形でまとめたいと思います．

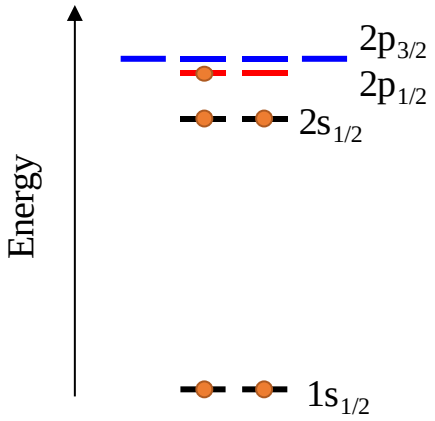


図-1. 5 電子系原子のスピン軌道分裂.

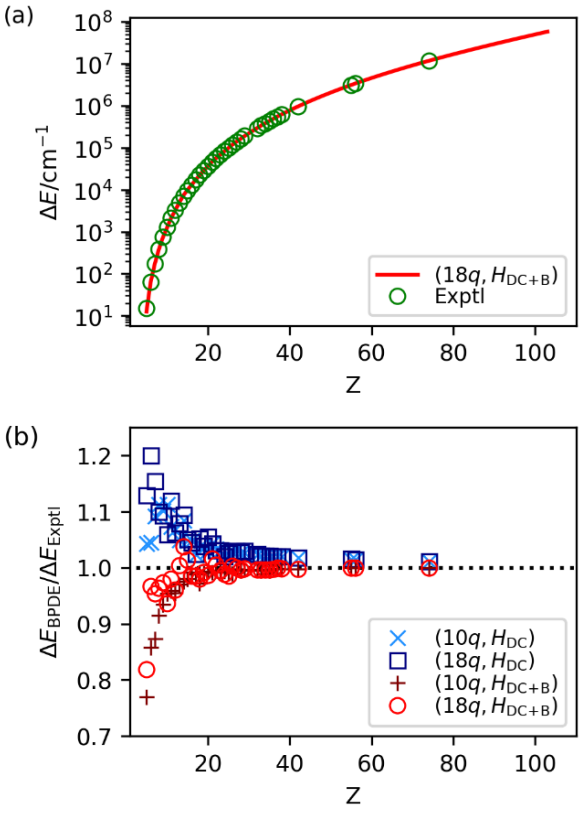


図-2. (a) 計算されたスピン軌道分裂値.
(b) 量子ビット数と Breit 項の影響評価.

2. 計算手法

今回の計算では周期表[9]を貫き，Z＝5のB原子からZ=103のLr^{98+}イオンまで同じ基準で扱うことを意図するため，Dirac方程式の4成分ハミルトニアンではCoulomb項だけでなく，伝達遅延に関するBreit補正も含めています．大成分にはユニバーサル基底($\alpha_0 = 0.01$ and $\beta = 1.80$)を用いて，動力学的バランスによって小成分の基底を生成しました．2p軌道のj=1/2とj=3/2のスピン軌道分裂の模式図を図-1に示します．従って，BPDEで対象とする2状態は

$$^2P_{1/2} = |(1s_{1/2})^2(2s_{1/2})^2(2p_{1/2})^1\rangle \quad (1)$$

$$^2P_{3/2} = |(1s_{1/2})^2(2s_{1/2})^2(2p_{3/2})^1\rangle \quad (2)$$

になります．Dirac-Hartree-Fockは，2p軌道の1電子を除いた状態で解いて空の2p軌道を静電的に原子価領域に引き込むと共に，対称性も保持しています．

量子コンピュータによる相対論的な計算は，文献[10]が先駆的でした．全エネルギーを直接位相差推定するタイプのアルゴリズムでSbH分子のスピン軌道分裂を評価していましたが，重原子が変わる場合には難点がありそうです．今回のBPDEではエネルギー差を直接求めるため，Zが5から103まで大きく変化しても精度を一貫して保てる点でより優れています．

OpenFermion[11]とCirq[12]を使った活性軌道空間のJordan-Wigner変換のマッピングで，10量子ビット{1s, 2s, 2p}，または18量子ビット{1s, 2s, 2p, 3s, 3p}としました．後者では，動径方向の電子相関が考慮されます．本来であれば，d型基底関数を導入して軌道をさらに増やし，角度方向の電子相関も取り入れるべきですが，j=1/2とj=3/2の状態のエネルギー差の評価では「影響の相殺」が期待されます(特にZが大きく高価数のイオン).

NVIDIA cuQuantum SDK[7]は，GPUを用いた量子回路シミュレータ向けのソフトウエア開発キットです[13]. GPUは数値演算性能が高く，メモリアクセスも高速であることから，ステートベクトルシミュレーションを大きく高速化します．今回は，Cirqのシミュレーターバックエンドとして用いられるqsim[14]にて，cuStateVecライブラリを使用しています．GPU利用により，現在の量子コンピュータの実機で実現できるよりも深く(ゲート数が多い)かつ広い(量子ビット数が多い)量子回路のシミュレーションを，現実的な時間で行うことが可能です．状態ベクトルだけでなく，テンソルネットワークにも対応しています．今回は行っていませんが，複数GPUを用いた分散処理もサポートしており，将来のより広い軌道空間での計算でも威力を発揮するものと期待されます．

今回の計算の実行環境は，「不老」 Type IIスーパーコンピュータ(Intel Xeon Gold-6230 & NVIDIA A100)に加え，杉崎が管理するGPU無しのワークステーション(Intel Xeon Gold-6134)も使いました(Pythonスクリプトの調製の都合)．なお，Xeonに関しては後出しますが，両者でほぼ同等の性能になります．

3. 結果と考察

図-2の(a)に，18量子ビットでBreit補正まで含めて計算したスピン軌道分裂値をプロットしました．緑丸は実験値ですが，全域に渡ってきわめて良好な対応を示していることが見てとれます．個別の値では，B原子の計算値12.5287±0.9813 cm^{-1}に対して実験値は15.287 cm^{-1}で，今回の活性軌道空間の制約の中ではリーズナブルな差と思われます．Z=29のCu^{24+}では，計算値が190942±246 cm^{-1}に対して実験値[15]が191280 cm^{-1}で，相対的誤差は有意に減少します．さらに，Z=74のW^{69+}では各々11800183±44 cm^{-1}と11802000 cm^{-1}となり，対応はきわめて良いです．Zが大

表-1. 計算時間の比較(単位は秒). 加速は WS と GPU 使用の場合の比で算出した.

#Qubits	WS	Flow		
	CPU	CPU	& GPU	Acc.
8	628	731	177	3.55
10	2197	2267	588	3.74
16	73452		4830	15.2
18	387328		9081	42.7

きくなると, 相対論効果が支配的になるとも言えます. 全域での実験値との平均二乗誤差は605.3 cm^{-1}となりました.

図-2の(b)は, 量子ビット数・軌道空間の設定とBreit項の影響を比の形で図示したものです. Breit項を含めた方は過小評価傾向ですが, 赤丸で示す18量子ビット計算で誤差が明らかに小さくなり, 今回の計算スキームの妥当性を示しています. Zが大きくなると, この傾向はより明確になります.

最後に計算時間についてです. 表-1に, 杉崎のワークステーションと「不老」 Type IIでの計算時間をまとめました. 加速はワークステーションと「不老」のCPU+GPUの場合との比で示しています(Xeon間でそれほど差がないため). 量子ビット数が増えるほどGPU加速は顕著になり, 18量子ビットでの加速は42.7倍に達しました. Z=5からZ=103までを尽くすには, ワークステーションでは1年程度はかかるところでしたが, GPUスーパーコンピュータを使うことで, 約1週間で済みました. これは, インパクトのある事実で, 量子シミュレータを使った研究の実施判断にも影響を与えるものです.

4. まとめと今後

本稿では, cuQuantumを使った相対論的なBPDE計算によってB原子と等電子系イオンのスピン軌道分裂のエネルギーをBPDEによって高精度で系統的に評価した成果を紹介しました. CPUに比してのGPUによる加速は最大(18量子ビットの場合)で42.7倍となり, 高いパフォーマンスを示しました.

将来のFTQC(Fault-Tolerant Quantum Computer)を睨んだ今後の応用展開の方向としては, 計算の報告例が未だほとんど無い遷移金属原子を含む小型分子の扱いが挙げられます. これらの系では, スピンの異なる状態がエネルギー的に近接するためにBPDEを使うメリットが活きると思われます.

5. 謝辞

本研究は, JSTさきがけ「量子化学計算の高効率量子アルゴリズムの開発」 (助成番号 : JPMJPR1914), および日本学術振興会の科研費(助成番号 : 21K03407)の支援を受けました. 「不老」 Type IIでの計算は, JHPCN共同研究プロジェクト(課題番号 : jh220010)の枠で実施しました. また, 立教SFRからの支援を得ました. 最後に, NVIDIAのcuQuantum SDKを利用するきっかけを作っていただいたblueqat株式会社の湊雄一郎CEOにも深謝します.

参考文献

[1] R. P. Feynman, "Simulating physics with computers", *Intern. J. Theor. Phys.*, **21** (1982) 467-488.
[2] S. S. Gill et al., "Quantum computing: A taxonomy, systematic review and future directions", *Softw. Pract. Exper.*, **52** (2022) 66-114.
[3] Y. Cao et al., "Quantum Chemistry in the Age of Quantum Computing", *Chem. Rev.*, **119** (2019) 10856-10915.
[4] B. Bauer et al., "Quantum Algorithms for Quantum Chemistry and Quantum Materials Science", *Chem. Rev.*, **120** (2020) 12685-12717.
[5] 杉崎研司 他, "量子アプリケーション", *映像情報メディア学会誌*, **77** (2023) 43-48.
[6] K. Sugisaki et al., "Bayesian phase difference estimation: a general quantum algorithm for the direct calculation of energy gaps", *Phys. Chem. Chem. Phys.*, **23** (2021) 20152-20162.
[7] https://developer.nvidia.com/cuquantum-sdk
[8] K. Sugisaki et al., "Bayesian phase difference estimation algorithm for direct calculation of fine structure splitting: accelerated simulation of relativistic and quantum many-body effects", arXiv:2212.02058 (2023).
[9] P. Pyokkö, "A suggested periodic table up to Z ≤ 172, based on Dirac-Fock calculations on atoms and ions", *Phys. Chem. Chem. Phys.*, **13** (2011) 161-168.
[10] L. Veis et al., "Relativistic quantum chemistry on quantum computers", *Phys. Rev. A*, **85** (2012) 030304-1-5.
[11] https://github.com/quantumlib/OpenFermion
[12] https://quantumai.google/cirq
[13] https://developer.nvidia.com/cuquantum-sdk
[14] https://github.com/quantumlib/qsim
[15] https://physics.nist.gov/PhysRefData/ASD/levels_form.html. Retrieved: 2022-08-22.

3:15 PM - 3:30 PM (Thu. Jun 1, 2023 2:30 PM - 3:45 PM Room D)

[D-09-04] 量子アニーリングを用いた固体構造解析手法の開発

*本田 理央[1]、遠藤 克浩[2]、鈴木 雄大[1]、松田 佳希[3]、田中 宗[1]、村松 眞由[1] （1. 慶應義塾大学、2. 産業技術総合研究所、3. Fixstars Amplify）

3:30 PM - 3:45 PM (Thu. Jun 1, 2023 2:30 PM - 3:45 PM Room D)

[D-09-05] Factorization Machineを用いた量子アニーリングによる Phase-fieldモデルのハミルトニアン補正項構築手法の開発

*青木 汐里[1]、遠藤 克浩[2]、松田 佳希[3]、関 優也[1]、田中 宗[1]、村松 眞由[1] （1. 慶應義塾大学、2. 産業技術総合研究所、3. Fixstars Amplify）