

AlphaFold2 最適化の取り組み

曾我 隆¹⁾, 寺前 勇希²⁾, 山口 健太³⁾, 川崎 隆矢³⁾, 伊達 進^{1),4)}

1) 大阪大学 D3 センター

2) 大阪大学 情報推進部

3) NEC ソリューションイノベータ

4) 大阪大学 大学院情報科学研究科

`takashi.soga.cmc@osaka-u.ac.jp`

Efforts of AlphaFold2 Optimization

Takashi Soga¹⁾, Yuki Teramae²⁾, Kenta Yamaguchi³⁾, Ryuya Kawasaki³⁾,
Susumu Date^{1),4)}

1) D3 Center, Osaka University.

2) Department of Information and Communications Technology Services, Osaka University

3) NEC Solution Innovators, Ltd.

4) Graduate School of Information Science and Technology, Osaka University.

概要

大阪大学 D3 センターは、全国の研究者に対して大規模計算機システムの計算資源を提供するだけでなく、同システムの利用者が計算資源を有効に利用いただくための様々な支援を実施している。本報告では利用者支援の一つである「GPU 再チャレンジ支援プログラム」で実施した、深層学習を用いてタンパク質の立体構造を予測するプログラムである AlphaFold2 の最適化の取り組み事例を紹介する。

1 はじめに

大阪大学 D3 センター（旧サイバーメディアセンター）は全国の大学の研究者が学術研究・教育に伴う計算及び情報処理を行う全国共同利用施設として、大規模計算機システムを提供している。現在稼働している大規模計算機システムとして、SQUID (Supercomputer for Quest to Unsolved Interdisciplinary Datascience) がある [1]。SQUID は汎用 CPU ノード群、GPU ノード群、ベクトルノード群、大容量ストレージから構成されており、利用者は自身の利用用途に適した計算資源を選択して利用することが可能である。

D3 センターは大規模計算機システムの利用拡大と利用者の利便性向上のための施策として、以下の利用者支援の取り組みを実施している。

- 講習会・セミナーの実施
- 対面相談
- 公募型利用制度
- 性能チューニングプログラム

このうち「性能チューニングプログラム」は利用者が実際に使用するプログラムをセンターでお預かりして、大規模計算機向けの最適化（並列化を含むプログラムの高速化）を実施する [2]。本取り組みの一環として、2022 年度に「GPU 再チャレンジ支援プログラム」を実施した。これは GPU を利用しようとして過去に断念された利用者、性能が出なくて困っている利用者を対象として公募を行い、採択された利用者のプログラムコードに対して GPU ノード群向けのプログラムチューニング（Open ACC 指示行挿入による GPU への移植を含む）作業を行い、利用者が GPU ノード群で高い演算性能を達成するための支援を目的とした。

AlphaFold2 は DeepMind 社が開発したタンパク質の立体構造を深層学習を用いて予測するアプリケーションであり、オープンソースとしてプログラムコードが公開されている。AlphaFold2 を利用することにより、従来の NNR といった実験的な手法や分子軌道計算を行うアプリケーションの利用と比較して短時間でタンパク質の立体構造を予測することができる。しかし、現状の AlphaFold2 は、GPU カード 1 基（1 枚）を搭載した GPU ノードを用いた実行を前

提とし開発・配布されている。そのため、SQUID の GPU ノード群のようなノード内に複数の GPU カード (SQUID の場合は 8 基) を搭載する環境に対して、AlphaFold2 が計算機資源を十分に活かしていない課題が生じ得る。

そのような背景もあり、2022 年度実施した本支援プログラムに対して、大阪大学大学院生命機能研究科の利用者から AlphaFold2 に対するチューニングを希望する応募が得られた。本報告は、高性能計算・データ分析融合計算技術を研究する教員と大規模計算機システムを運用する職員、同システムを保守するシステムエンジニアとが連携して、SQUID システム向けに AlphaFold2 の最適化に取り組んだ事例を紹介する。

2 AlphaFold2

DeepMind 社が開発した深層学習を用いてタンパク質の立体構造を高い精度で予測するアプリケーション AlphaFold2 は、ソースコードが GitHub 上で無償公開されており、誰でも自由に利用することが可能である [3]。本学においても、多くの研究者が AlphaFold2 を利用してタンパク質の複合体 (複数のタンパク質が結合) の立体構造を予測している [4]。コンピュータ上でタンパク質複合体のスクリーニング (タンパク質の結合状態を計算により明らかにする) を実施した上で、実験による検証を行う。その結果、実験の過程で毒性の高いタンパク質の取り扱いを減らすことや実験に要する工数の削減を図ることに成功している。しかし、今回応募のあった利用者の研究室のサーバを用いても 1 日にスクリーニングできる複合体は 2 ペア程度で、研究を加速させるためには AlphaFold2 の処理時間の短縮が不可欠である。このような理由から、SQUID システムを利用する上で GPU 再チャレンジプログラムに AlphaFold2 の最適化 (高速化) を希望する応募が「GPU 再チャレンジ支援プログラム」にあった。

図 1 に AlphaFold2 の主な処理のフローを示す。準備ブロックでは、まずインスタンスを作成するインスタンス作成ステップを実行する。次に、データベース検索ステップにおいて、入力されたタンパク質のアミノ酸配列から AlphaFold DB (AlphaFold Protein Structure Database) [5] を検索して MSA (Multiple Sequence Alignment) を作成する。予測ブロックでは MSA に基づくアミノ酸残基間コンタクト予測と、予測対象タンパク質との類似性を有すると考えられる「鋳型」構造情報を深層学習により統合した手法を用いて立体構造予測を行う [6]。最後に、予測したタンパク

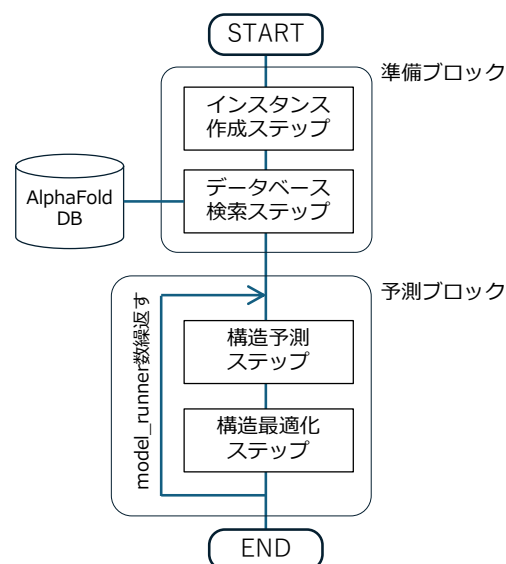


図 1 AlphaFold2 の処理フロー

質の立体構造に MD 計算によって修正を加える構造最適化ステップを実行する。この構造予測ステップと構造最適化ステップは model_number 数分繰返すが、利用者の研究で常に 5 つのモデルを評価しているため、本報告の評価では model_number の値は「5」である。

AlphaFold2 は Python を用いて記述されている。AlphaFold2 の各ステップの処理の多くは、OpenMM や HHSuite といった一般に公開されている汎用ライブラリに含まれる関数によって実行されている。そのため、個々の関数は既に多くの研究者によってチューニングが試みられており、ソースコードレベルのチューニング余地は少ないと考える。

また、データベース検索ステップにおいて検索の対象となる AlphaFold DB は 2023 年時点で総計 2 億を超える予測構造モデルを擁するデータベースに拡大しており、データベース検索ステップの処理時間の占める割合が大きくなっている。図 2 は SQUID の GPU ノード群を用いた実行結果である。AlphaFold DB はアクセス時間を短縮するため SSD (Solid State Drive: Lustre NVMe) に置いている。vas.tej と spn-E.tej の 2 つの入力データの結果を見ても、準備ブロックの処理時間が全体の半分以上を占めており、特に vas.tej では 80% 近くの時間を占めていることが分かった。

準備ブロックの実行はすべて CPU 上で行われるため、利用者は準備ブロックと予測ブロックの実行を分離して、準備ブロックは汎用 CPU ノード群を、予測ブロックは GPU ノード群を利用して実行している。

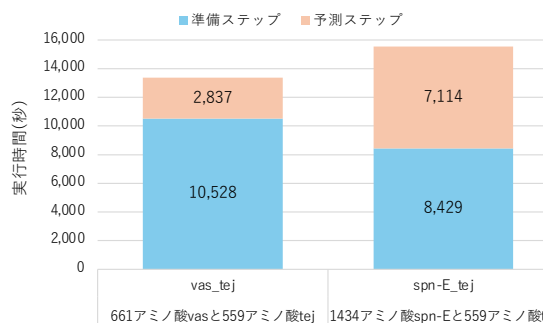


図2 準備ブロックと予測ブロックの処理時間

これは汎用 CPU ノード群の時間当たりの利用料が GPU ノード群と比較して安価であることと、ジョブの開始待ち時間が短いためである。GPU 再チャレンジ支援プログラムでも準備ブロックと予測ブロックを分離した形態のまま最適化を検討した。

3 最適化

AlphaFold2 は現在のバージョンは v2.3.0 である。しかし、本報告では利用者が公募時に使用していた v2.2 を対象としている。

まず、準備ブロックの最適化を検討した。準備ブロックの処理の内容を分析したところ、入力データ zuc_zuc (253 アミノ酸 zuc 2 分子) において、約 89% の時間を HHblits (HH-suite ツール [7]) と Jackhmmer (HMMER ツール [8]) が占めていることが分かった。この HHblits と Jackhmmer はデータベース AlphaFold DB に対して配列の検索を行うものである。AlphaFold2 の実行時に設定するパラメータの中に、HHblits と Jackhmmer の実行スレッド数を指定する項目がある。それぞれのデフォルト値は HHblits は「4」、Jackhmmer は「8」であるが、このデフォルト値が最適値であるかどうか確認する必要があった。SQUID の汎用 CPU ノード群のノード当たりのコア数は 76 であることから、パラメータに設定する HHblits と Jackhmmer のスレッド数を 4 から 76 まで変更して評価した。

予測ブロックは主に構造予測ステップと構想最適化ステップで構成される。構造予測ステップは数値計算ライブラリ JAX [9] を使用することで、ほとんどの部分を GPU で実行する。また、構造最適化ステップは OpenMM ライブラリ [10] による分子動力学計算を用いて、予測した立体構造の最適化 (修正) を行う。実行時間を比較すると、構造予測ステップが予測ブロック

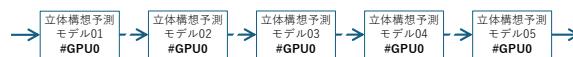


図3 最適化前の予測ブロックの処理フロー

の約 60% から 70% を占めており、構造最適化ステップが 15% から 25% を占める。構造予測ステップと構造最適化ステップは異なる 5 つのモデルを実行時オプション「-l」で指定した回数実行されるが、今回の利用者は「-l=1」を設定している。

図3に最適化前の予測ブロックの処理フローを示す。オリジナルの AlphaFold2 は異なる 5 つのモデルの立体構造予測 (構造予測ステップ、構造最適化ステップ) の実行を、1 基の GPU カードを使用して逐次的に処理を実行する。これは第1章で述べた通り AlphaFold2 が GPU カード 1 基実装したサーバでの運用を想定しているためと考えられる。また、この 5 つのモデル相互に依存関係 (他のモデルの結果を参照する必要がある等) がないことも確認できた。そのため、SQUID の GPU ノード群のような複数 GPU カードを搭載するスーパーコンピュータ向けに、モデル毎に異なる GPU カードを割り当て、5 つのモデルを並列に実行することとした。しかし、今回の最適化作業では構造予測ステップのみを並列に実行した。これは構造最適化ステップやその他の処理の実行時間に占める割合が小さく、CPU と GPU を頻繁に切り替えて使用したため、並列化によるオーバーヘッドが大きくなる可能性があるためである。

図4に最適化後の予測ブロックの処理フローを示す。構造予測ステップに対して、モデル毎に GPU を割り当てるように設定し、かつ、5 つのモデルを同時に処理する修正を行なった。オリジナルの AlphaFold2 は各モデルを逐次的に実行する過程で、あるモデルの構造最適化ステップのエラーチェック処理でエラーを検出した場合、例外処理関数を呼び出してプログラム自体を終了して、継続するモデルの実行は打ち切られる構造になっていた。そこで、5 つのモデルを並列に正しく実行するために、エラーを検出したモデルが存在した場合でもプログラムを終了 (打ち切り) させないように修正した。

4 評価

AlphaFold2 の最適化の評価を大阪大学の SQUID システムの汎用 CPU ノード群と GPU ノード群を用いて実施した。

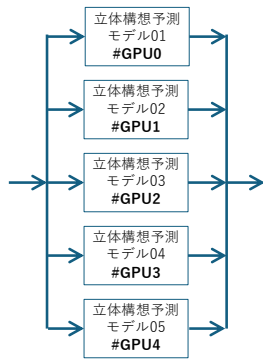


図4 最適化後の予測ブロックの処理フロー

表1 評価システムの諸元.

汎用 CPU ノード群	Intel Xeon Platinum 8368 (Icelake) (2.40GHz 38 コア) 2 基 主記憶: 256GB
GPU ノード群	Intel Xeon Platinum 8368 (Icelake) (2.40GHz 38 コア) 2 基 主記憶: 512GB GPU : NVIDIA A100 8 基

4.1 評価システム

汎用 CPU ノード群と GPU ノード群の諸元を表1に示す。どちらのノード群も CPU は同じ Intel Xeon Platinum 8368 を2基搭載している。ノード当たりの主記憶は GPU ノード群が2倍の512GB搭載する。また、GPU ノード群に搭載している8基の NVIDIA A100 は、Delta Baseboard 上の NVSwitch により高速に相互接続している。ストレージは AlphaFold DB を高速にアクセスする必要から、DDN ExaScaler (Lustre) の SSD を使用した。

4.2 準備ブロックの評価

準備ブロックではデータベース (AlphaFold DB) に対して配列の検索を行う HHblits と Jackhmmer のスレッド数を指定する実行パラメータを変更し、最適なスレッド数を評価した。入力データは zuc.zuc (253 アミノ酸 zuc 2 分子) を使用した。図5にスレッド数を4から76までの HHblits 部分の実行時間を示す。デフォルト値4スレッドに対して、SQUID の汎用 CPU ノード群では38スレッドを設定した場合の実行時間が一番短くなることが分かった。また、図6にスレッド数を4から76までの Jackhmmer の実行時間を示す。こちらはデフォルト値の8スレッドの実行時間が一番短いことが分かった。

この結果、デフォルトのスレッド数を指定した実行時間1,311秒に対して、HHblits のスレッド数を38に変更することで1,005秒と約300秒の実行時間の削減を得た。ただし、今回の結果は SQUID システムの汎

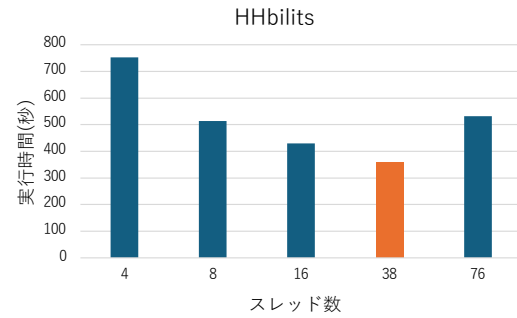


図5 HHblits のスレッド数による実行時間

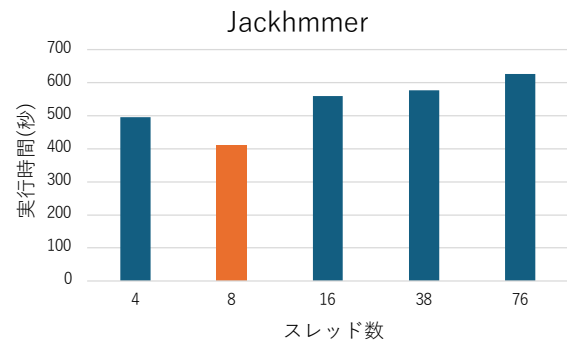


図6 Jackhmmer のスレッド数による実行時間

用 CPU ノード群（搭載するコア数は76）における入力データ zuc.zuc の結果であり、使用するシステムのコア数や入力データが異なる場合は最適なスレッド数も変わる可能性があるが、デフォルトのスレッド数は最適な値でないことを示した。

4.3 予測ブロックの評価

予測ブロックでは構造予測ステップの5つのモデルの実行に関して依存関係がないことに着目し、各モデル (JAX ライブラリ) の実行をそれぞれ異なる GPU に割り当てて、並列に実行する最適化を実施した。なお、今回の最適化は SQUID システムの GPU ノード群 (GPU を8基搭載) を前提にしており、ノード内の GPU 数が5基未満の環境では評価しない。

各入力データとも5つのモデルを5基の GPU を用いて並列に実行した結果、zuc.zuc は約1.5倍、vas.tej は約2.8倍、spn-E.tej は約3.6倍の並列効果を得た。しかし、最も高い並列効果が得られたケースでも3.6倍程度の結果であった。これは構造予測ステップの処理中で、JAX ライブラリの jit 関数を用いて JIT コンパイルを行っている部分は CPU 側で実行されている影響や、Python のスレッド並列が現状の AlphaFold2 では CPU 側で逐次に処理されるためと考えられる。

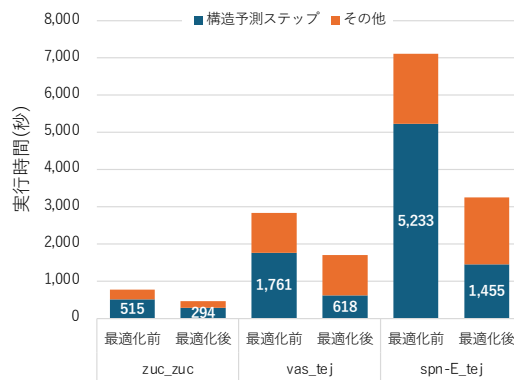


図 7 構造予測ステップの並列化結果

5 まとめ

大阪大学 D3 センターでは大規模計算機システムの利用者支援として「性能チューニングプログラム」を実施している。この取り組みの一環として「GPU 再チャレンジ支援プログラム」を実施した。SQUID システムを利用する大阪大学大学院生命機能研究科の利用者から申請された、深層学習用いてタンパク質の立体構造予測を行う AlphaFold2 の最適化を実施した。

大学の教員と職員、システムエンジニアとが連携して最適化を実施した結果、汎用 CPU ノード群上で実行する「準備ブロック」は、HHblits のスレッド数を最適化することにより、入力データ zuc_zuc において約 300 秒の実行時間の削減することを明らかにし、入力データや使用する計算機システムに応じたスレッド数の変更が必要であることを示した。また「予測ブロック」は、5 つのモデルの実行に 5 基の GPU を使用して並列に実行することにより、入力データ zuc_zuc は約 1.5 倍、vas_tej は約 2.8 倍、spn-E_tej は約 3.6 倍の並列効果を得ることで実行時間の削減を図った。

最適化の結果、これまで利用者が研究室にあるサーバ利用時では 1 日に 1 ないし 2 ペアのスクリーニングしか行えなかったものが、最適化を実施した AlphaFold2 を SQUID システムで実行することにより、1 日に 100 から 150 ペアのスクリーニングを可能にした。

本報告のとおり、シミュレーションプログラムの最適化を実施することで研究に要する時間（計算資源）の削減を達成した。大阪大学 D3 センターでは今後も利用者のシミュレーションプログラムを高速に実行する計算機環境を提供していくと同時に、「性能チュー

ニングプログラム」を継続して、大規模計算機システムの利用者の研究を支援していく所存である。

謝辞

本報告の成果は、大阪大学 D3 センターの SQUID を利用して得られたものである。

参考文献

- [1] 大阪大学 D3 センター SQUID, <http://www.hpc.cmc.osaka-u.ac.jp/squid/>.
- [2] 2023 年度性能チューニングプログラム, https://www.hpc.cmc.osaka-u.ac.jp/lec_ws/20230927-2/.
- [3] google-deepmind/alphafold/ <https://github.com/google-deepmind/alphafold/>.
- [4] Shinichi Kawaguchi, Xin Xu, Takashi Soga, Kenta Yamaguchi, Ryuuya Kawasaki, Ryota Shimouchi, Susumu Date, Toshie Kai, "In silico screening by AlphaFold2 program revealed the potential binding partners of nuage-localizing proteins and piRNA-related proteins," Cold Spring Harbor Laboratory, 2024, doi=10.1101/2024.07.28.605023.
- [5] AlphaFold Protein Structure Database, <https://alphafold.ebi.ac.uk/>.
- [6] 徳井健太郎, AlphaFold がもたらすタンパク質構造生物学の新時代, 実験医学 Vol.41, No.16, P2564, 2023 年.
- [7] soedinglab/hh-suite, <https://github.com/soedinglab/hh-suite/>.
- [8] HMMER, <http://hmmer.org/>.
- [9] JAX documentation, <https://jax.readthedocs.io/en/latest/quickstart.html/>.
- [10] OpenMM, <https://openmm.org/>.