jh240081

次世代演算加速装置とそのファイル IO に関する研究

塙 敏博（東京大学）

概要

GPU クラスタにおいて，実アプリケーションを効率よく実行するためには GPU に対するデータの入出力の考慮が必要であり，データ転送と演算のオーバラップ，転送レイテンシの短縮を工夫する必要がある．そこで本研究では，GPU 上データの直接ファイル IO である GPUDirect Storage (GDS)，あるいは計算とファイル IO のオーバラップの両者を容易に取り扱い可能にする手法を確立し，様々なファイル入出力特性を持つ実アプリケーションにおいて GPU-ファイル IO 間の処理を効率化することを目的とする．今年度は，主に GH200 システムにおける性能について調査した．また，より先進的な GPU 直接 IO の試みとして提案されている BaM (Big accelerator Memory) や GIDS, SCADA についても調査を行った．

1 共同研究に関する情報

1.1 共同研究を実施した拠点名

- 東北大学 サイバーサイエンスセンター
- 東京大学 情報基盤センター
- 東京科学大学 情報基盤センター
- 名古屋大学 情報基盤センター
- mdx

1.2 課題分野

- 大規模計算科学課題分野

1.3 参加研究者の役割分担

塙 敏博 (東大・代表): 全体取りまとめ，GPU 通信，機械学習
建部 修見 (筑波大・副代表): ストレージ技術，機械学習
三木 洋平 (東大): 宇宙物理コード
佐藤 拓人 (筑波大 ⇒ JAEA): City-LES
星野 哲也 (名古屋大): GPU プログラミング
河合 直聡 (名古屋大 ⇒ 東北大): 数値アルゴリズム
中島 研吾 (東大): 計算科学アプリケーション
住元 真司 (東大): ストレージ技術
山崎 一哉 (東大): 計算科学アプリケーション
小林 諒平 (筑波大 ⇒ 東京科学大): GPU 通信，IO
藤田 典久 (筑波大): GPU 通信，IO
朴 泰祐 (筑波大): GPU 通信，IO
平賀 弘平 (筑波大): ストレージ技術、機械学習

2 研究の目的と意義

HPC システムにおいて, アクセラレータとして GPU が広く利用されているが，JHPCN 構成機関においても，ほぼ全てで GPU を採用したシステムが稼働中である．GPU クラスタにおいて, 実アプリケーション実行の際には GPU に対するデータの入出力の考慮が必要であり，データ転送と演算のオーバラップ，転送レイテンシを削減する必要がある．近年 NVIDIA

によって GPUDirect Storage (GDS) が提供されており IO オーバヘッドの短縮が期待される．本研究は，GPU 上データの直接ファイル IO，あるいは計算とファイル IO のオーバラップの両者を容易に取り扱い可能にする手法を確立し，様々なファイル入出力特性を持つ実アプリケーションにおいて GPU-ファイル IO 間の処理を効率化することを目的とする．さらに，JCAHPC の Miyabi システムにおいても GH200 におけるファイル IO 最適化を実現する．

3 当拠点公募型研究として実施した意義

本研究では，計算科学の実アプリにおけるファイル IO の観点で，宇宙物理コードの GOTHIC，気象コードの City-LES を対象にし，各分野の専門家を交えて研究を実施した．さらに PyTorch などの機械学習フレームワークも対象にしている．

本研究では，AMD EPYC Milan CPU + NVIDIA A100 80GB PCIe GPU を搭載する実験用 GPU サーバ，NVIDIA GH200 Grace-Hopper Superchip を搭載する実験用サーバ，Wisteria-Mercury と呼ばれる H100 搭載実験クラスタ，および筑波大学計算科学研究センターに導入された，Intel Sapphire Rapids CPU, NVIDIA H100 80GB PCIe GPU を搭載する Pegasus システムも使用して実施している．

これらの知見を，今後の JHPCN 各計算資源にフィードバックすることも一つの目的である．

4 前年度までに得られた研究成果の概要

前年度は，N 体シミュレーションに基づくベンチマーク `h5gds` を作成した．また HDF5 の既存の GDS プラグイン vfd-gds を拡張し，IO の性質に基づいて GDS に限らず最適なファイル転送を実現した．その結果，既存手法と比較してローカルファイルシステムにおいては最大 4.33 倍，リモートファイルシステムにおいては最大 6.09 倍のバンド幅向上が得られた．

5 今年度の研究成果の詳細

5.1 N 体シミュレーション

2025 年 1 月から運用開始した Miyabi の演算加速ノード Miyabi-G に搭載される NVIDIA GH200 において GDS の利用に向けた準備を行った．まず，Miyabi-G に向けた検証として，NVIDIA GH200 を搭載したテストノードにおいて GDS の動作検証を行った．用いたコードは前年度に実装したベンチマークプログラム `h5gds` であり，HDF5 から GDS を用いたファイル入出力性能を測定する．本テストプログラムでは N 体シミュレーションにおける典型的なデータ構造を想定し，GPU メモリ上に確保された粒子位置と質量を格納する `float4` 型の配列，粒子速度の x 成分と y 成分を格納する `float2` 型の配列，粒子速度の z 成分を格納する `float` 型の配列，粒子 ID を格納する `uint64_t` 型の配列のデータ入出力を `H5Dread_multi()` 及び `H5Dwrite_multi()` で行った際の実行時間を測定する．データ入出力方式については asis と hyperslab の 2 つのモードがある．Asis モードにおいては，N 体計算時に用いる `float4` 型や `float2` 型の配列データをそのまま入出力する．Hyperslab モードは，実際のシミュレーション実行及びポスト

プロセスでの解析処理における利便性を考慮し，HDF5 の hyperslab 機能（MPI における派生データ型に類似の機能）を用いて，粒子位置と質量を格納する `float4` 型の配列を位置データを格納する $N \times 3$ 成分の配列と粒子質量の配列に分離して出力する実装である．Hyperslab モードにおいては，粒子位置を格納する $N \times 3$ 要素の `float` 型の配列，粒子速度を格納する $N \times 3$ 要素の `float` 型の配列，粒子質量を格納する N 要素の `float` 型の配列，粒子 ID を格納する N 要素の `uint64_t` 型の配列が出力ファイルに書き込まれ，またこのファイルを GPU から読み出すこととなる．

図 1 に，NVIDIA GH200 上での性能測定結果を示す．今回の測定結果においては，GDS を用いるよりも互換モードで動作させた方が高性能であった．ただし，7 月末時点までは GH200 上で GDS を動作させることすらできなかったこともあり，また GH200 においては，NVIDIA GPU ドライバのバージョンによって，特定のアプリケーション性能が極端に落ちるケースも見受けられることから，今後のシステムソフトウェアの改良や GH200 に適したパラメータセットの探索などによって性能が向上していくと期待している．

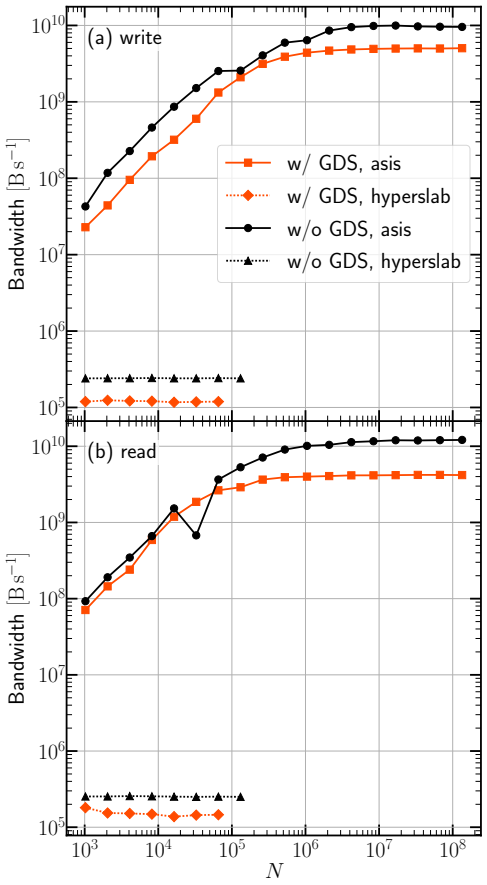

図 1　GDS 使用の有無によるデータ入出力性能の比較．上段: データ書き出し時のバンド幅，下段: データ読み込み時のバンド幅，x 軸: 粒子数 N

5.2　都市街区気象シミュレーション

これまで主に担当してきた佐藤 (筑波大) が JAEA に転出したため，今年度は本テーマの継続が困難になった．City-LES については，GDS，あるいはより先進的な GPU-ファイル IO の利用に向けた最適化について，別途改めて検討を行うこととした．

その一方で，本プロジェクトでは，今後取り扱う気候・気象シミュレーションのコードについて，Scale-RM を対象にすることにした．例えば以下の参考文献に示すような熱帯低気圧のシミュレーションにおいては，大規模現象における高精度データの詳細な時間発展について観察をする必要があり、IO オーバーヘッド削減の方法について検討を行う．

- J. Ito, T. Oizumi, and H. Niino, “Near-surface coherent structures explored by large eddy simulation of entire tropical cyclones,” Scientific Reports, Vol. 7, No. 3798, 2017.

5.3　機械学習

機械学習フレームワークにおいて, GH200 と GDS を組み合わせた最適化について検討を実

施した．GH200 において，PyTorch による機械学習の training における性能を測定した [9]．これによると，H100 に比べて GH200 が GNN training において 1.4 倍，3D U-Net training において 1.6 倍の性能向上を得られている．CPU と GPU 間のリンクである NVLink-C2C の転送バンド幅が，GH200 では H100 に比べて 6.1 倍高く，これが性能向上に大きく寄与している．DALI において GDS を有効に組み合わせることで，GH200 では全体の処理時間を低減することができると考えられる．

一方，より先進的な GPU 直接 IO の試みとして BaM (Big accelerator Memory) が提案されており [1]，別途検証を進めている．通常，GPU メモリに収めるためのデータの分割やページフォールトの処理等に CPU 側のソフトウェアが関わっており、GDS 等では依然としてこれらに起因するオーバーヘッドが生じており，特に小データサイズの際に顕著である．そこで Qureshi らは、GPU スレッドがオンデマンドかつ細粒度に直接 NVMe-SSD にアクセスすることを可能にした Big accelerator Memory (BaM) という新たなシステムアーキテクチャを開発し、直接 I/O の更なる高速化を試みている．

さらに Park らはグラフニューラルネットワーク (GNN) の学習に用いるデータローダーに BaM を組み込み，大規模 GNN の学習高速化を図る GPU Initiated Direct Storage Access dataloader (GIDS) を提案した [2]．大規模な GNN の学習においてはミニバッチ学習が広く用いられるが，ミニバッチングから特徴量集約までの前処理を CPU で行い, 学習を GPU で行っていた．しかし，CPU の処理速度がボトルネックとなり，GPU が CPU の処理完了を待つことになってしまう．そこで GIDS では，データの前処理・学習を共に GPU で行うこととし，GPU で処理するデータをストレージから直接 GPU メモリにフェッチすることで CPU のソフトウェアによるオーバーヘッドを削減するため，BaM を用いる．また，GIDS では BaM に加え，より高速な学習のため以下の機能も持つ．

- Constant CPU Buffer: 頻繁にアクセスするノードを格納する CPU バッファ
- Dynamic Storage Access Accmulator: BaM によりピークの SSD バンド幅に到達するために必要なストレージアクセス数を動的に測定
- Window Buffer: ノードのアクセスパターンに応じ、退避させるキャッシュラインを最適化

また, Chang らは, CPU メモリをキャッシュとして使う拡張として GMT を提案している [3]．BaM では GPU メモリから退避するページは全て SSD に格納されたが、退避されるページが近く再利用される可能性が高い場合、CPU メモリに格納しておくため，高い性能が得られることが期待できる..

今年度は BaM を使ったシステムを構築し，動作検証を実施した．GDS との比較では，NVMe-SSD のブロックサイズである 4KB 単位の IO において，図 1 にも示す通り GDS では顕著な性能低下が見られる一方，BaM ではほぼ理想的なバンド幅を達成することを確認した．この際，NVMe-SSD に対して，ファイルシステム等を用いず，ブロックデバイスに直接アクセスをするため，BaM 専用のデバイスが必要であり，BaM のままでは実現のハードルが高いこともわかった．

参考文献

[1] Z. Qureshi et al., GPU-Initiated On-

Demand High-Throughput Storage Access in the BaM System Architecture, ASPLOS 2023, pp. 325–339, 2023.

[2] Jeongmin Brian Park et al., Accelerating Sampling and Aggregation Operations in GNN Frameworks with GPU Initiated Direct Storage Accesses, Proceedings of the VLDB Endowment, Vol. 17, No. 6, 2024.

[3] Chia-Hao Chang et al., GMT: GPU orchestrated memory tiering for the big data era, In Proceedings of ASPLOS, pp. 464-–478, 2024.

6 今年度の進捗状況と今後の展望

今年度まで 3 年間にわたって JHPCN のプロジェクトを継続してきたが，来年度はメンバーの入れ替えや，方針の変更もあることから，新規のプロジェクトとして実施する予定である．

6.1 N 体シミュレーション

今年度は，2025 年 1 月から運用を開始した Miyabi-G における GDS の活用に向けて注力してきた．様々な問題点を洗い出しながら，今後も．NVIDIA 社の協力を得ながら関連するソフトウェアの改良に取り組んでいく予定である．現在は GH200 上で GDS の動作について確認を進めている段階であり，今後 GH200 上での動作に適したパラメータセットの探索を進めていく．また，hyperslab モード使用時に IO 性能が極端に低下する問題は NVIDIA 社とも共有済みであり，今後も連携しながら機能改善に取り組んでいく．

6.2 気象シミュレーション

前述のように，今年度は City-LES を用いた研究の継続ができなくなったため，Scale-RM を対象にしていくことを検討した．

Scale-RM は GPU 化がなされており，GPU 上の必要な解析データを読み出す部分がボトルネックになる可能性がある．一方，GH200 においては，GPU メモリに対して CPU 側から比較的高速にアクセスすることができ，ファイル IO 性能を大きく改善できる可能性がある．

6.3 機械学習

今年度は，GH200 における実証に注力した．機械学習自体の性能改善にも NVLink-C2C の効果が実証された．

また，前述のように，BaM の検証も実施した．BaM は，ファイルシステムに起因するオーバヘッドを劇的に減らし，GPU のスループットを最大限に活かせるアーキテクチャであるが，あくまでも実験用のプロトタイプの位置付けである．NVIDIA は，SCADA (Scaled accelerated data access) を提案し，以下の講演でも紹介している．十分に性能をスケールさせるには，ファイルシステムの構造をバイパスする必要があり，これまでの GDS および cuFile から，S3 over RDMA および cuObject を用いることとなる，今後，構成を検討し，基本性能を確認する必要がある．

- C.J. Newburn et al., How to Get Data Between Storage and the GPU at the Speed of Light, S73012, GPU Technology Conference 25, Mar. 2025.

6.4 HDF5 向け GDS プラグインの拡張

これまで，HDF5 向けの GDS プラグインを拡張するとともに，NVMe-SSD と Lustre 並列ファイルシステム上で、HDF5 に対して，CPU メモリコピーによるファイル IO と GPUDirect Storage（GDS）を比較してきた．ユーザが提供するヒントを使用することで，GPU アプリケーションにおける HDF5 形式の書き込

み性能が向上することが示された [5,6]．しかし，この方法には 2 つの大きな問題点がある．まず，ユーザーが提供するヒントに依拠しており、その信頼性は保証されない．また、GDS と CPU メモリの選択は、転送サイズとスレッド数のハードコードされた値に依存する．

そこで，来年度以降は，HDF5 に加え，NetCDF、VisIt、h5py、ParaView、Binary などの追加のファイル形式を考慮し、GPU アプリケーション向けの最適なデータパスを選択するフレームワークを拡張する予定である．また、ユーザが提供するヒント、転送サイズ、スレッド数以外のファクタを特定し、フレームワークに組み込むことで、汎用性と精度を向上させる．また，来年度は，様々な CPU アーキテクチャを持つ GPU クラスタを利用し，フレームワークにおけるパラメータの調整について検討する．