6J-08

医用画像処理における LDDMM の GPU 高速化

杉浦 拓未† 大島 聡史‡ 中島 大地†3 片桐 孝洋‡ 横田 達也†4 本谷 秀堅†4 永井 亨‡
名古屋大学工学部電気電子・情報工学科† 名古屋大学 情報基盤センター‡
名古屋大学大学院情報学研究科†3 名古屋工業大学大学院工学研究科†4

1. はじめに

医用画像処理において，臓器の統計形状モデルは学習用 3 次元画像中の臓器領域データ群より構築される．臓器の統計形状モデルを求める手法のうち最も広く知られている手法は Active Shape Model(ASM)であるが，ASM によって構築されたモデルは表現性能が十分でなく，臓器として適切ではない形状が含まれる可能性がある．妥当な形状のみを表現するモデルの構築方法としては，臓器領域間の非線形な微分同相写像(1 対 1 かつ順写像も逆写像も滑らかな写像)によるモデル化が知られている．

2 つの臓器間の微分同相写像を求める手法の中でも Large Deformation Diffeomorphic Metric Mapping(LDDMM)[1]が広く採用されている．しかし，LDDMM は多くの計算を必要とする手法である．それにも関わらず統計形状モデルを構築するためにはこの手法を多数の学習データに対して多数回適用しなければならない．そのため LDDMM の高速化への要求は大きい．

中島ら[2]は LDDMM コードをマルチコア CPU 向けにハイブリッド MPI/OpenMP 並列化し高速化したが，その実行時間のうちの多くは浮動小数点演算で占められていることが分かっている．したがって，高い浮動小数点演算性能を持つ GPU を活用することでさらなる高速化が期待される．以上から本研究ではハイブリッド MPI/OpenMP 並列化された LDDMM コードをベースとして，複数の GPU を用いた LDDMM コードを実装し性能評価を行う．

2. LDDMM

LDDMM は 2 枚の入力画像I_0, I_1に対して，I_0からI_1への微分同相写像の生成を行う． $I_{0.25}, I_{0.5}, I_{0.75}$を生成画像とした例を図 1 に示す．

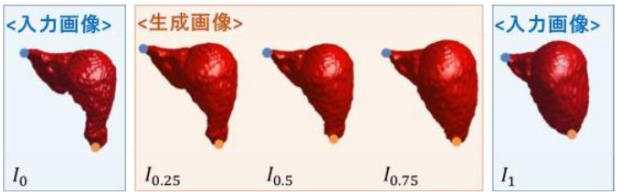

図 1 肝臓平均形状から肝臓症例への LDDMM

本研究で用いる LDDMM コードでは再急降下法を用いて微分同相写像を求める．写像を求めるための計算部分は以下の 4 ステップを繰り返す構成となっている．

Step1. ヤコビアンの演算
Step2. Backward Integration
Step3. ベクトル場vの更新
Step4. 写像 φ と位置情報の更新

ベースとなるハイブリッド MPI/OpenMP 並列化コードでは，Step3 において 2 回の MPI による AllGather 通信を含んでいるが，それ以外はプロセスごとに独立して計算を行っている．

3. GPU 並列化手法

本研究では，GPU を用いた並列化に OpenACC[3]を用いる．OpenACC は既存の C,C++/Fortran ソースコードにディレクティブを挿入することで処理の一部を GPU へオフロードさせて実行できる．今回は Step1 から Step4 までの計算部分をすべて GPU 上で処理する．

マルチ GPU での実行時は GPU の枚数と同数のプロセスを立ち上げ 1 プロセスに 1GPU を対応させ，それぞれのプロセスで OpenACC による GPU へのオフロードを行う．Step3 に含まれる AllGather 通信については複数 GPU 間で通信を行う必要がある．

複数 GPU 間で高速に通信を行う方法としては，CUDA-Aware MPI が広く用いられてきた．特に GPUDirect を利用可能な環境においては GPU 側のメモリのデータをメインメモリにコピーすることなく GPU 間で通信できるため，高速な通信が可能である．本研究では OpenMPI[4]を用いる．

NCCL(NVIDIA Collective Communications Library)[5]は NVIDIA により開発された GPU 向けの集団通信ライブラリである．NCCL は NVLink を用いた高速な GPU 間集団通信関数を提供しており，MPI プログラムに対して少しの修正で利用するこ

GPU acceleration of LDDMM code in medical image processing
† Takumi Sugiura, Electrical and Electronic Engineering, School of engineering, Nagoya University
‡ Satoshi Ohshima, Takahiro Katagiri, Toru Nagai, Information Technology Center, Nagoya University
†3 Daichi Nakajima, Graduate School of Informatics, Nagoya University
†4 Hidekata Hontani, Tatsuya Yokota, Graduate School of Engineering, Nagoya Institute of Technology

とができる．そのため集団通信を頻繁に行うディープラーニングの分野において広く利用されている．LDDMM コードでは Step3 の中でデータ量の大きい GPU 間の AllGather 通信を行っているため，NCCL を用いて通信の高速化を試みた．

4. 性能評価

本章では LDDMM コードの CPU 逐次実行，CUDA-Aware MPI/OpenACC 並列実行，NCCL/OpenACC 並列実行の 3 つの実行時間を比較する．実行時間の測定は計算部分(Step1 から Step4 の反復の合計時間)のみであり，その前後のメモリ確保/解放や初期化などの時間は含まない．

4.1 評価環境

名古屋大学情報基盤センター設置の GPU サーバ sx40 を利用する．表 1 にスペックを示す．

表 1　評価環境

CPU	CPU Intel Xeon Gold 5122 (Skylake-SP, 3.6GHz, 4 コア 8 スレッド) × 2 ソケット
GPU	NVIDIA Tesla V100 SXM2 × 4 枚
メインメモリ	DDR4-2666 384 GB
コンパイラ	PGI C++コンパイラ (ver.19.4-0)

4.2 問題設定

入力画像データは 100×100×100 の 3 次元画像，Step1 から Step4 の処理の反復回数を 1500 回に固定して性能評価を行った．

4.3　評価結果

LDDMM コードの CPU 逐次実行，CUDA-Aware MPI/OpenACC を用いた 4GPU 実行，NCCL/OpenACC を用いた 4GPU 実行の 3 つの実行それぞれに対し，ステップごとの実行時間，合計実行時間，CPU 逐次実行に対する速度向上率を表 2 に示す．

GPU 間通信を含む Step3 について，NCCL を用いた場合は CUDA-Aware MPI を用いた場合に比べて約 1.75 倍の速度となっており，NCCL による効率的な集団通信によって速度が大きく向上していることが分かる．実行時間全体については，NCCL/OpenACC を用いたコードが最も早く，逐次の LDDMM コードに対して速度が約 290 倍に向上しており GPU 並列化の効果が大きいことが分かる．

表 2 実行結果

	逐次実行 (CPU)	CUDA-Aware MPI(4GPU)	NCCL(4GPU)
Step1(秒)	0.894	0.068	0.068
Step2(秒)	0.022	0.151	0.149
Step3(秒)	2601.905	12.189	6.947
Step4(秒)	512.506	3.570	3.567
合計(秒)	3115.327	15.978	10.731
速度向上率		194.977	290.323

5. まとめ

OpenACC，NCCL を用いた複数 GPU による実行により逐次のプログラムに対して約 290 倍という大幅な高速化を達成することができた．また，CUDA-Aware MPI による GPU 間通信よりも NCCL を用いた場合のほうが高速であるという結果であった．

しかし，実装した GPU 並列化コードでは GPU 間通信中に GPU コアが待機状態となっており，ここにさらなる高速化の余地がある．OpenACC の async 節を用いた通信と演算のオーバーラップなどによるさらなる性能向上は今後の課題である．

また，今回は 1 ノードで実験を行ったが，複数ノードを用いた大規模な環境においての実行と評価も今後の課題である．

謝辞

本研究は JSPS 科研費 JP18K19782，および，JP18H03262 の助成を受けたものです

参考文献

[1]Beg, M.F. et al., “Computing Large Deformation Metric Mappings via Geodesic Flows of Diffeomorphisms”, International Journal of Computer Vision, vol.61, issue 2, pp.139–157, 2005

[2]中島 大地，田中 友揮，物部　峻太郎，本谷　秀堅，横田　達也，片桐　孝洋，永井 亨，「医用画像処理における LDDMM コードのハイブリッド MPI/OpenMP 実行の評価」，情報処理学会論文誌，2019

[3]OpenACC, https://www.openacc.org/

[4]OpenMPI, https://www.open-mpi.org/

[5]NVIDIA Collective Communications Library(NCCL), https://developer.nvidia.com/nccl