1J-07

複数のGPU向けプログラミングモデルを用いた倍々精度疎行列ベクトル積の特性分析

寺田 洋人† 慈道 亮人† 大崎 健太† 菱沼 利彰‡ 藤井 昭宏† 田中 輝雄†
工学院大学† ブレイドテクノロジーズ株式会社‡

1. はじめに

倍々精度演算（DD 演算）は，ソフトウェアによる 2 つの倍精度変数を組み合わせた 4 倍精度相当の演算である [1]．DD 演算は倍精度演算と比べ 10 倍から 20 倍の計算量を要するため，計算時間が増加する．山浦らは，複数の GPU 向けプログラミングモデルにおける DD 演算で GPU による高速化が有効であることを示している [2].

GPU 向けのプログラミングモデルでは，並列化の粒度を設定する必要がある．並列化の粒度は粗粒度と細粒度という 2 つの値で決定する．粗粒度は処理を大まかに分割し，細粒度は粗粒度をさらに細かく分割する．CPU 向けのプログラムにディレクティブを挿入することで，GPU 向けプログラムを生成するプログラミングモデルがある．ディレクティブベースのプログラミングモデルは粗粒度と細粒度が自動で設定される，山浦らにより自動で設定される値は最適値でないことが報告されている [2].

本研究の目的は，GPU 向けのプログラミングモデルごとの細粒度，粗粒度の設定による性能の変化を評価し，条件を変化させることによって性能にどのような影響があるのかを分析することである．ディレクティブベースの GPU 向けのプログラミングモデルである，OpenMP の Offload 機能 (以下 OpenMP) [3] と OpenACC [4] の並列化の粒度を手動で設定し，倍精度疎行列と DD ベクトルの積 (DD-SpMV) を行った．加えて，実行時に必ず並列化の粒度を指定する必要がある CUDA の実行結果と比較した．

2. GPU向けのプログラミングモデル

2.1 CUDA

CUDA は NVIDIA 社製の GPU 向けのプログラミングモデルである．kernel と呼ばれる GPU 上で実行されるコードを CPU 上で実行されるコードと別に記述するため，CPU 向けのプログラムを GPU で動かすにはコードを大きく書き換える必要がある．本研究ではコンパイラに NVCC を使用した．

CPU と GPU 間のデータの受け渡し用の cudaMemcpy，cudaMemset や，GPU メモリを確保，開放するための関数として cudaMalloc，cudaFree などの関数が用意されている．CUDA において，粗粒度，細粒度の値は block，thread である．

```
1 #pragma omp target teams distribute parallel for \
2 num_threads(num_threads)
3 for (i=0;i<N; i++) {
4     y[i] = 0.0;
5     for (j=crs_row_ptr[i];j<crs_row_ptr[i+1];j++){
6         y[i] += crs_val[j] * x[crs_col_ind[j]];
7     }
8 }
```

図 1 OpenMP の Offload 機能を用いて実装した SpMV

表 1 計測した条件一覧

GPU (NVIDIA)	マシン名	場所
Tesla V100	不老 Type II	名古屋大学
RTX 3080	研究室サーバ	工学院大学
Tesla A100	Wisteria-A	東京大学

2.2 OpenMP の Offload 機能

OpenMP はディレクティブベースのプログラミングモデルで，並列化のためのディレクティブが実装されている．本研究では OpenMP に含まれる Offload 機能を用いる．コンパイラには NVC++ と Clang を使用した．

倍精度疎行列と倍精度ベクトルの積 (SpMV) の GPU 実行部分を図 1 に示す．並列化対象のコードに #pragma omp target teams distribute parallel for というディレクティブを挿入することで GPU で実行する．GPU へのデータコピーは enter data，exit data といったディレクティブを挿入する．OpenMP で粗粒度，細粒度は team，thread と呼ばれる．

2.3 OpenACC

OpenACC はディレクティブベースの並列コンピューティング用の規格のひとつである．本研究ではコンパイラとして NVC++ を使用した．

ディレクティブを挿入する場所は多くの場合 OpenMP と同じである．主なディレクティブとして，並列化部分を示すディレクティブである#pragma acc kernel や，データの転送を行う target enter data, target exit data などがある．OpenACC で粗粒度，細粒度は gang，vector と呼ばれる．

3. 実験

プログラミングモデルごとの DD-SpMV の性能を比較した．SpMV の性能との比較も行った．

プログラムの実行時に指定する粗粒度の値は，細粒

Analysis of Double-Double precision SpMV using three GPU programming models
Hiroto Terada† Ryoto Jido† Osaki kenta† Toshiaki Hishinuma‡ Akihiro Fujii† Teruo Tanaka†
† Kogakuin University ‡Braid Technologies K.K.

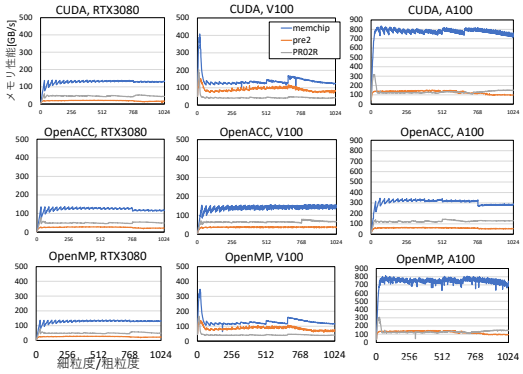

図 2　細粒度/粗粒度ごとのメモリ性能 (DD-SpMV)

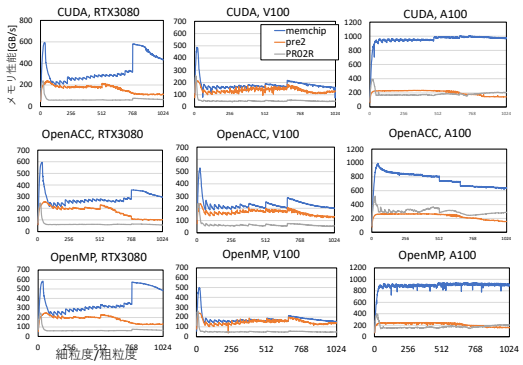

図 3　細粒度/粗粒度ごとのメモリ性能 (SpMV)

度を設定すると一意に定まる．実験では，粗粒度あたりの細粒度の数を 1 から 1024 まで変化させた．

疎行列の格納形式は CRS 形式を使用し，SpMV では行ごとの並列化を行った．計測範囲は演算部分のみとし，CPU と GPU 間のデータ転送の時間は計測の対象外とした．性能の指標は，DD と倍精度での演算回数が異なるため，1 秒あたりのアクセスしたメモリデータ量 [GB/s] とした．以下メモリ性能と呼ぶ．使用した行列はそれぞれ memchip（非ゼロ要素数:13,343,948/列数:2,707,524），PR02R（161,070/8,185,136），pre2（659,033/5,834,044）の三種類である．

使用した GPU を表 1 に示す．コンパイラは NVCC11.4，NVC++22.2 を用いた．RTX3080 環境のみ NVCC11.7，NVC++22.9 とした．性能の違いを確認するために，V100 では Clang14.0 も用いた．

3.1　GPU の違いによる性能変化の分析

DD-SpMV の性能を図 2，SpMV の性能を図 3 に示す．縦軸はメモリ性能，横軸は粗粒度あたりの細粒度での分割数である．

実行する GPU により並列化の粒度の影響は異なる．OpenMP と CUDA は似た性能の傾向を示した．最適な細粒度/粗粒度は GPU によって異なるが，多くは 36 以下である．DD-SpMV はすべての環境で SpMV のメモリ性能を下回った．

DD-SpMV は，一部の環境で性能が伸びず，徐々に上がり急に下がるギザギザの傾向を示した．性能が伸びない理由については判明していないが，粗粒度ごとの演算量の偏りが影響している可能性が考えられる．

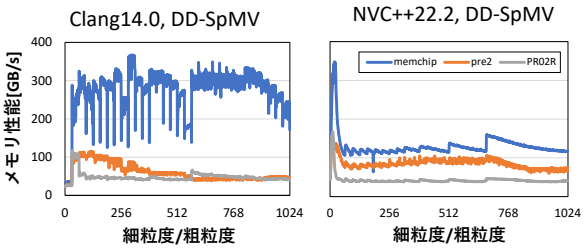

図 4　NVC++, Clang を用いた OpenMP の DD-SpMV (V100)

3.2　問題行列の変化による性能変化の分析

SpMV，DD-SpMV ともに，問題行列により性能の傾向が変化する．非ゼロ要素数の多い行列の方がメモリ性能が高くなる場合が多い．行列 memchip ではベクトルの型を問わず，性能が伸びた場合の最適な細粒度/粗粒度のメモリ性能が，平均値の約 2 倍の性能を示した．

3.3　コンパイラの違いによる性能変化の分析

NVIDIA V100 を用いた OpenMP の DD-SpMV を，Clang，NVC++ でそれぞれコンパイルして，実験を行った結果を図 4 に示す．コンパイラを変更した場合，性能の傾向は変化することが分かった．しかし，最適な細粒度/粗粒度の場合の性能差は，約 20GB/s とわずかに Clang が高い程度で，大きな差はなかった．

4.　おわりに

本研究では，DD-SpMV において，細粒度，粗粒度の設定による性能の変化を計測し，条件を変化させることによって性能にどのような影響があるのかを確認した．結果，計測に使用する GPU，問題行列，コンパイラ，プログラミングモデルによって性能の傾向が変化することが分かった．

今後の課題として，チューニングによって最適な粒度を自動的に選択する方法の模索などがあげられる．

謝辞

本研究は，JSPS 科研費 JP18K11340 の助成により実施した．

参考文献

[1] Bailey, D.H.: High-precision floating-point arithmetic in scientific computation, Computing in Science & Engineering, Vol.7, No.3, pp.54-61 (2005).
[2] 山浦 朴人，福永 晋司，菱沼 利彰，藤井 昭宏，田中 輝雄：ディレクティブベースの GPU プログラミングモデルを用いた倍々精度演算の性能評価，第 84 回情報処理学会全国大会，4J-04 (2022).
[3] OpenMP, https://www.openmp.org (2022-12-26).
[4] Application Programming Interface, OpenACC, https://www.openacc.org (2022-12-26).