GPGPU を用いた格子ボルツマン法と粗視化粒子法による ハイブリッドシミュレーションの高速化

増田知記†‡、田中良夫†、小林亮*、井上洋平*、尾形修司*、後藤俊幸*

1. はじめに

我々は流体スケールと粗視化スケールのマルチスケールシミュレーション手法である、格子ボルツマン法（Lattice Boltzmann Method, LBM 法）による流体シミュレーションと粗視化粒子法（Coarse Grained Particle Method, CGP 法）による固体シミュレーションのハイブリッドシミュレーションの開発を進めている。本稿においては、GPU を用いた CGP-LBM ハイブリッドシミュレーションの高速化について報告する。

CGP-LBM 法では複数の粒子に対して計算を行うため並列性は高く GPU を使う事による性能の向上が期待できる。今回は NVIDIA の Tesla C1060(以降 Tesla と呼ぶ)と最新の Fermi アーキテクチャを採用した Tesla C2050(以降 Fermi と呼ぶ)を利用して高速化と評価を行った。

Fermi は Tesla からいくつもの改良が行われており、浮動小数点演算性能は単精度で 933GFlops から 1030GFlops へ、倍精度で 78GFlops から 515GFlops へ、メモリ帯域は 102GB/sec から 144GB/sec へと性能が向上しており、また L1 と L2 キャシュが追加された事で性能を発揮しやすくなっている。他にも ECC の追加や float 型の atomic 加算等の様々な機能拡張も行われている。

2. プログラムの概要

今回の CGP-LBM 法のプログラムは Fortran で組まれており MPI にも対応していたが GPU への移植後には MPI は利用しないこととした。プログラムの中では二次元に配置された各粒子について CG force を計算する箇所が処理時間の大部分を占めており、この部分の高速化が重要となる。演算の精度は一部で倍精度浮動小数演算の利用があるが、大部分が単精度浮動小数で行われ、3×3 等の比較的小さな行列の演算が多用されている。

3. GPU への移植

移植にあたってプロファイルを行った結果、最も負荷の高い CG force を求める関数が全体の処理時間の 92.7%を占めていたので、最初にこの関数の移植を行った。その結果この関数の全体に占める割合が低下したので、元々の全体に占める時間が 5.2%と 1.7%の 2 つの関数についても移植を行った。関数の移植にあたり Fortran で書かれていたものを CUDA と親和性の高い C 言語に書き直し同時にループの展開、計算結果の再利用などの最適化を行った。次に CUDA を使って GPU への移植を行ったが、並列化を妨げる粒子間の依存関係を解決する必要があった。Tesla では計算の途中結果をテンポラリの領域へ保存し最後にリダクションを行う方法で、Fermi では新しく追加された float の atomic 加算を利用する方法で対処を行った。他には GPU 上でのアクセスが効率良く行える形へのデータの並び替え、PC と GPU 間のデータ転送量の削減、GPU 上の thread 数の調整等の最適化も行った。

4. 評価の結果とまとめ

今回の評価では Intel Xeon X5550 2.66GHz を 2 機搭載した HP の Z800 Workstation に Tesla と Fermi の両方のボードを挿して行った。GPU を利用しない移植前のプログラムの評価もこの上で行った。評価では粒子の配列の大きさとして GPU の計算資源と比べて並列度の低い 20×20 と、十分な並列度を持つ 20×640 を利用した。

修正前の Fortran のプログラムで 20×20 の粒子の処理に掛かる時間は 35.25 秒だったが、C 言語への移植後で 11.14 秒に、GPU へ負荷の最も高い関数を移植した時点で Tesla では 4.14 秒に Fermi では 4.08 秒に、さらに 2 つの関数の移植後には Tesla では 4.78 秒、Fermi では 3.80 秒となった。20×640 の粒子では、Fortran のプログラムで 1120.36 秒、C 言語への移植後で 338.74 秒、GPU へ負荷の最も高い関数を移植後に Tesla で 56.41 秒、Fermi で 53.00 秒、残り 2 つの関数の移植後には Tesla で 40.90 秒、Fermi で 30.71 秒となった。全体の処理時間でみると 20×20 の粒子数では Fortran でのプログラムから C 言語に移植する事で 3.16 倍、Tesla の利用で 8.51 倍、Fermi の利用で 9.28 倍の高速化が行えた。20×640 の粒子では、C 言語への移植で 3.31 倍、Tesla の利用で 27.39 倍、Fermi の利用で 36.48 倍の高速化が行えた。また最も負荷の高かった関数にのみ注目すれば、20×640 の粒子の時に Fortran のプログラムと比べて Tesla で 78.39 倍、Fermi では 108.32 倍の高速化を達成した。

このように CGP-MD の計算を NVIDIA の GPU 上で動かすことで並列化を阻害する要因を排除する必要はあったが、大きな性能向上を実現することができた

今後は燃料電池の電解質膜から拡散層流路に至る統合シミュレーターなどの応用展開を目指して、より大規模なシミュレーションの評価を進めていく予定である。

参考文献

[1] Ryo Kobayashi, Takahide Nakamura and Shuji Ogata: Development and Implementation of Recursive Coarse-Grained Particle Method for Meso-Scale Simulation, Materials Transactions, Vol. 49, No.11, pp. 2541-2549, 2008.

† 産業技術総合研究所情報技術研究部門
‡ 株式会社フィックスターズ
＊名古屋工業大学大学院創成シミュレーション工学専攻