フジタ技術研究報告
第 48 号　2012 年

CUDA を用いた並列数値解析手法の一考察

仲 沢 武 志

概　　要

従来、大規模な問題や複雑な形状が計算対象の場合では、スーパーコンピュータのような高価な計算リソースを使用する必要があった。これに対し、コンピュータのハード的な進化によって、現在では市販のパーソナルコンピュータ(PC)でも計算機能は目覚しく向上し、並列演算すらも比較的容易に可能となってきている。

マルチコアな PC においては、OpenMP や MPI などの CPU による並列化が可能である。また、最近では、Graphic で使用される計算機能を数値解析に適用する GP-GPU（General Purpose on Graphic Processor Unit）という手法を適用することも実施され適用が拡大されつつある。

本文では、この GP-GPU におけるフレームワークの中で Nvidia が提唱している CUDA を採用し、いくつかの例題を試行してその適用性を検討する。

Evaluation of parallel numerical calculation method using CUDA system

Abstract

Previously large size memory problems and some complex shape problems required use of a supercomputer for analysis. However recently, personal computers have achieved sufficiently high performance to undertake these activities, and some of these personal computers can perform parallel calculations in numerical analysis.

Personal computers with multi core CPU(s) can perform as a system for open MPI or MPI to parallel calculation. At the same time, other components within the system such as high-performance graphics cards can be used for a parallel calculation procedure called General Purpose on Graphic Processor Unit (GP-GPU).

We used Nvidia's CUDA framework for GP-GPU systems, and applied it to some problems to check performance of these systems.

キーワード：有限要素法　高速化
並列計算　GP-GPU　CUDA

§1. はじめに

有限要素法などの数値解析は、結局のところ微分方程式を離散化し連立1次方程式に帰着させて、所要の境界条件の下に解を求める計算方法であり、使われ始めてからかなりの年月が経過している。この間、計算機の性能などの計算リソースの開発と関連しながら数々の計算アルゴリズムが発展して計算実効の効率化が追及されてきた。

しかしながら、最近におけるこの分野の技術は、CPU 単体の性能を飛躍的に進歩させる技術が頭打ちとなり、また計算アルゴリズムも一定の手法に落ち着くなど、停滞しつつあった。

ところが、ここ最近、計算リソースの方向性は、いわゆるPCと言われる個人で使うコンピュータでもマルチコアは当たり前になってきている。このような状況から計算手法も複数ある CPU を合理的に使用した並列アルゴリズムを試行して、計算効率の向上を望める状況が復活してきた。さらに、GP-GPU(General Purpose on Graphic Processor Unit)という概念で、グラフィックエンジンを数値解析などの一般的用途にも応用しようという動きも活発になっている。GP-GPU によって並列化を実装するフレームワークには OpenCL および Nvidia の CUDA や AMD の ATIStream などがある。

本文では、比較的簡単に並列化が実現できる GP-GPU のうち Nvidia が提唱している CUDA を使った試行に主眼を置き、いくつかの例題でその適用性を検討する　なお、ここでの議論では　CPU　による並列化は対象外としている。

§2. PC のハード構成

本文では、計算効率の向上にグラフィック機能で並列演算を実施する GP-GPU を考えることとしている。

PC の標準的なハード構成と GP-GPU における拡張例を図1に示す。通常の PC の構成を右側の破線内に示している。PC には、通常描画用のグラフィックボードが搭載されている。このグラフィックボード内の複数のプロセッサーを並列に使って計算を効率化するシステムが GP-GPU である。1枚のグラフィックカードを描画と同時に演算に使用することも可能であるが、システムに対応したグラフィックカードを複数搭載して、両者の役割を分担した方が効率は良いものと予想される。

そこで本文では、図 1 の左側に示したように、グラフィックボードを通常の描画用のものとは別にさらに1枚実装したハード環境としている。

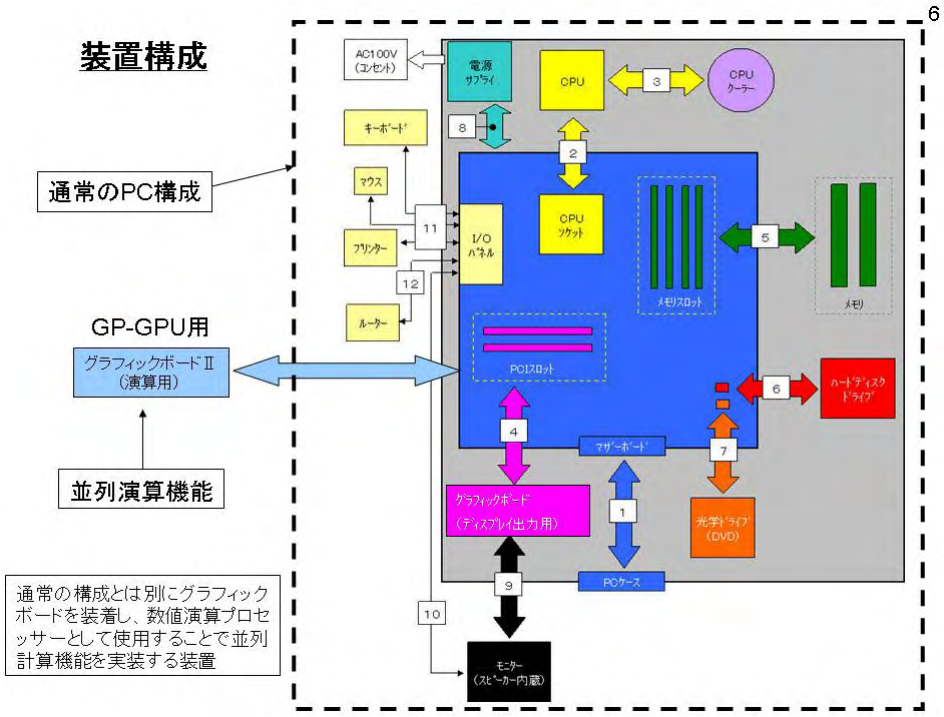

図1　PC の標準構成と CUDA での拡張例

以下にグラフィック以外の全体的なハード構成も示しておく。

CPU：Intel Core i-7-2600K
GPU-1：Nvidia Ge-Force GTX580
（ディスプレイ表示用）
GPU-2：Nvidia Tesla C 2070
（演算用）
MB：ASUS SABERTOOTH P67
チップセット：Intel P67 Express
実装メモリー総計：16GByte
OS：Windows7 Professional

§3. GP-GPU

3.1 グラフィックカードの構成

グラフィックボードの内部を模式的に示したものが図2である。演算最小単位はストリーミングプロセッサ（SM）であり、8 個の SM で構成されるセグメントがマルチストリーミングプロセッサ（MSP）と言われている。MSP はハードのグレードにより異なるがグラフィックボードに 256 個あるいは 512 個搭載されている。これらが、CPU やメインメモリと通信しながら並列演算を実施することになる。

3.2 プログラミング技法

プログラミングにおいては、基本的に並列演算させたい部分をサブルーチンで記述する。C言語での記述を例示すると、所要のメモリーを確保した上で、

(sub_name)<<gd,BSZ>>(argument)

などとして呼び出す。
またサブルーチン側は

global void sub_name(argument)

などと宣言して受ける。
文法の詳細は参考文献 2)～4)を参照。

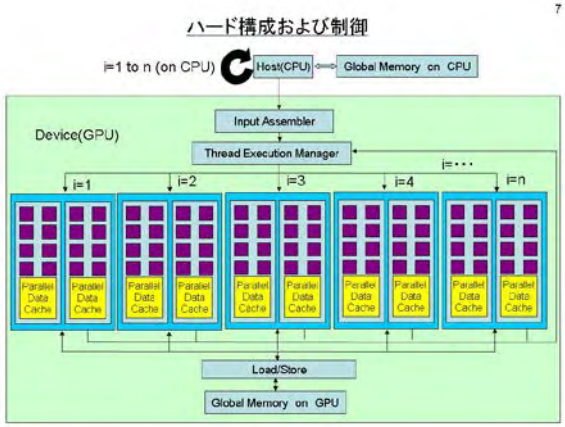

図2 グラフィックカード内の演算機能 [1)]

§4. 適用例

時間発展する物理的化学的現象の予測は、一般に多大な計算時間を必要とすることが多い。そこで、CUDA による並列演算が可能となれば、計算時間が短縮され、作業が効率化する。

有限要素法のような数値解析手法は、結局微分方程式を連立1次方程式に帰着させるわけであるが、連立1次方程式の演算時間は、その次数（自由度）の増加に連れて長くなることは周知である。並列演算が最大限に活用できた場合の効用は、グラフィックカード上の MSP の数にもよるが、図3の模式図が示すように、連立1次方程式の次数によらずほぼ一定の時間で計算を実施できることにある。

実際には、演算チップ数の制約から、このような効用を得ることができる最大の自由度が存在するが、本文ではそれに関しては議論の対象とせず、2000 自由度程度での並列演算の適用性を調べる。

時間発展問題の例として、まず地下水汚染などの状況を予測する移行拡散方程式に CUDA による並列演算を適用する。拡散方程式への適用は比較的事例情報も多く、手始めの検討に適切な例と考えたためである。

その他に、建設分野で当然遭遇する変形解析および、Laplace 方程式への適用を試みる。この2つの例への適用は、そもそも間接手法による連立1次方程式における係数行列の性質の違いを調べるためである。

4.1 移行拡散解析への適用

移行拡散方程式は次式のように与えられている。

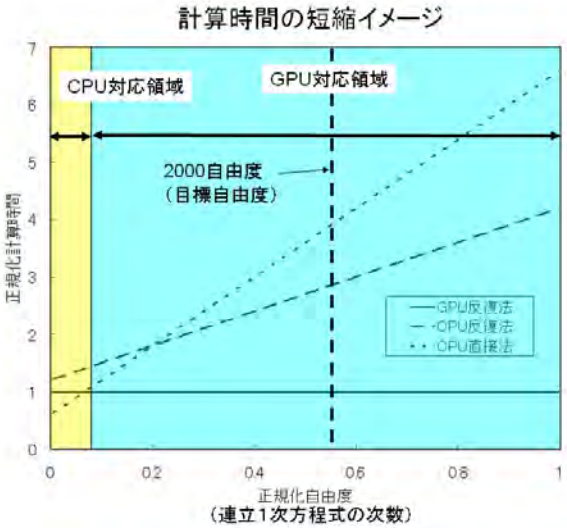

図3 並列化による計算時間短縮のイメージ

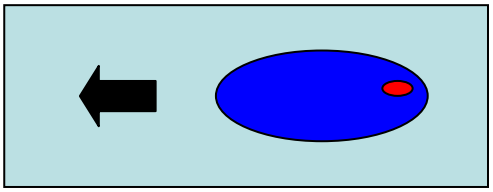

図4　計算対象(移行拡散解析)

$$\frac{\partial C}{\partial t}+v_k\frac{\partial C}{\partial x_k}=D_{ij}\frac{\partial^2 C}{\partial x_i \partial x_j} \qquad (1)$$

これを 3.2 に示した技法を織り交ぜて有限要素法に基づきプログラミングする。離散化した行列演算に対して、CUDA での並列演算と CPU での単一演算との計算時間を比較する。計算対象は図4のような単純な濃度拡散問題を設定する。

得られた計算時間の比較を図5に示す。

ここでは、離散化について詳しく記述はしないが、式(1)を空間的には有限要素法、時間に関しては前進差分を採用している。また、時間変化に対応する左辺第1項に影響する質量マトリックスに関しては対角化(lumping)を施している。これにより、各行が独立するので完全な並列演算を実施できることになる。

結果は図 5 に示すように、CPU での演算時間(20ms)に比べて 1/500 の結果(0.04ms)となった。また、計算時間の節点数依存性は予想した成果を示す図4そのものとなっていて、節点数に依存しない一定値であり、CUDA による並列化の効用は十分に発揮されていると思われる。

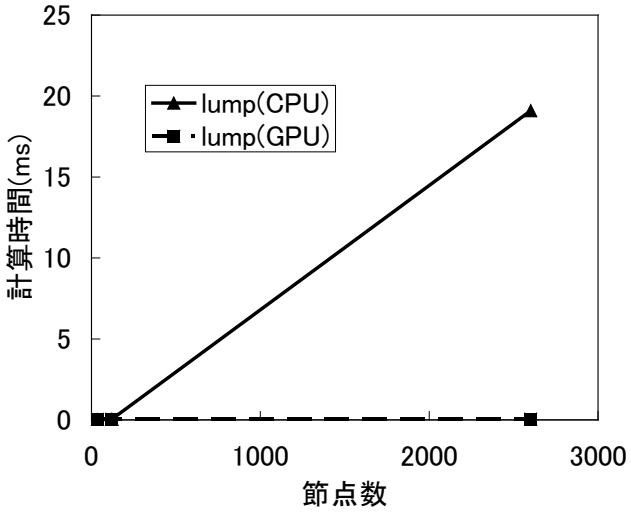

図5　計算時間の比較(移行拡散解析)

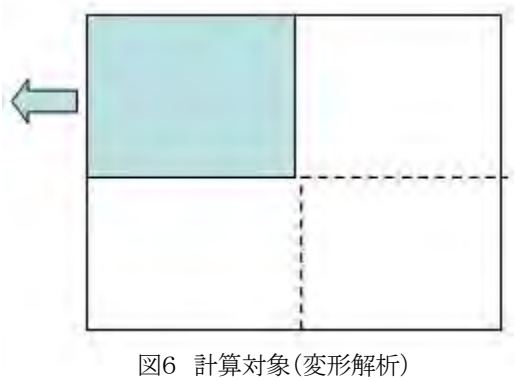

図6　計算対象(変形解析)

ここでの計算は時間ステップ1回分を示しているが、物質移行拡散解析は解析メッシュの最小幅や時間刻みに対して、クーラン数あるいはペクレ数で表される指標の制約を受ける。

例えば実際に物質移行拡散解析を地下水の汚染拡散の予測に適用する場合などでは、空間メッシュは工場が立地されるような規模の土地面積であり、時間的には戦後から 60 年オーダを対象とするようなこともある。このような場合には、単一ステップの計算時間は短くても、時系列な計算が不可能ではないにしても事実上相当な作業負荷を発生する可能性もある。これを、ここで示した計算時間のスピードアップによって、作業負荷が飛躍的に軽減することとなる。これにより、対策工法も含めた十分な検討資料を用意できるので、並列演算の効用は極めて意義が深いものと考えられる。

4.2　変形解析への適用

変形解析の支配方程式は下式で与えられている。

$$\frac{\partial}{\partial x_j}\left(C_{ijkl}\frac{\partial u_k}{\partial x_l}\right)=0 \qquad (2)$$

有限要素法で離散化した連立1次方程式を種々の解法で計算した[5)6)]。変形解析における計算対象は図6に示す単純引張問題とした。

得られた計算時間の比較を図7に示す。この問題は対角優位性 $(k_{ii}\geq\sum_{i\neq j}k_{ij})$ がないため、収束性が悪い。これは CUDA を適用したことによる結果ではなく、そもそも解法が持つ性質であることに注意する。つまり、一般に言われているように、対角優位性をもたない係数行列の場合、並列化の恩恵はあまり受けないと認識するのが正しい理解と思われる。なお、ここでの掃き出し法では、一旦逆行列を計算し、その後の行列積に並列化を施している。

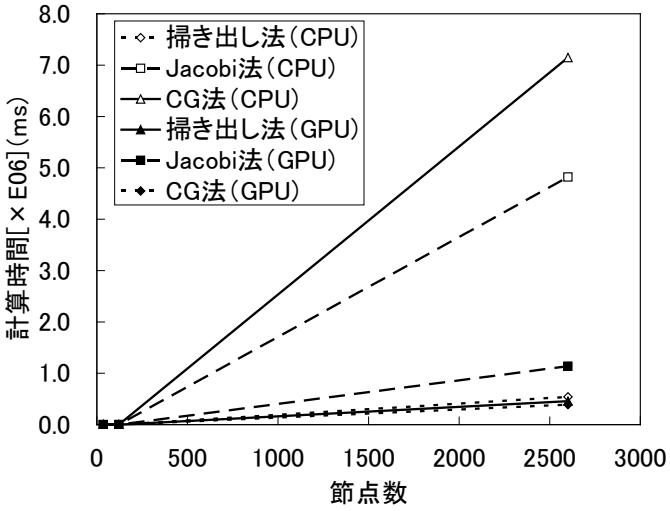

図7　計算時間の比較(変形解析)

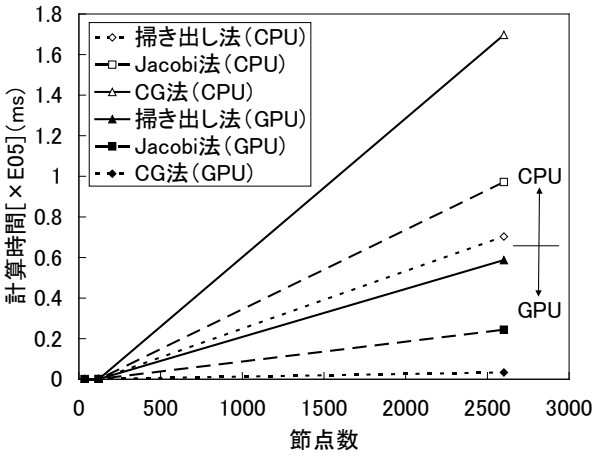

図9　計算時間の比較(ラプラス方程式)

4.3 ラプラス方程式への適用

一般に言われるように、対角優位性を持った係数行列の場合、間接解法の収束性は良好であるとすれば CUDA による並列計算の恩恵を得ることができる。このことを確認するために Laplace 方程式への適用を試みた。

Laplace 方程式は次式である。

$$\frac{\partial}{\partial x_i}\left(\frac{\partial \phi}{\partial x_i}\right)=0 \qquad (3)$$

これを有限要素法で離散化し、得られた係数行列を種々の解法で計算した。計算対象を図8に示す。

なお、蛇足ながら、図8における Direcret BC とはディリクレ境界条件であって、目的変数 ϕ 自体の値を境界条件として与えるものである。

これらから得られた計算時間は図９のようになった。成果の予想イメージ図3のような傾向を示している。

これより、移行拡散方程式ほどではないにしろ対角優位性をもった係数行列の場合には CUDA の恩恵を得ることが可能であることが確認できた。なお、ここでも掃き出し法は、一旦逆行列を計算し、その後の行列積に並列化を施しているのは変形解析と同様である。

図8　計算対象(ラプラス方程式)

§5. まとめ

本文での知見を以下にまとめる。

1)　自明ではあるが、同じアルゴリズムであれば CUDA によって並列化を実装した方が計算時間は短い。

2)　並列演算の効用が大きく期待できる問題は、例えば係数行列を対角化して各行を独立に扱えるような問題である。

この条件に適合する計算対象は、時間発展の問題を前進差分によって時間積分した離散化である。これは、移行拡散解析に CUDA を適用したことで得られた知見である。

3)　対角項以外にも非ゼロ数が要素として入る一般の係数行列を間接手法で計算する場合には、対角優位性のように、扱う係数行列の性質によって収束性の良否が場合分けされる。

対角優位性とは、係数行列の対角項以外の要素の総計より対角項の要素が大きいという性質である。本文に示したように、ラプラス方程式などの対角優位性を持つ問題では、計算手続きも簡単で収束性は良好であるため、CUDA の適用が効果的である。しかしながら、変形解析のように、対角優位性を持たない係数行列の場合には、収束性が悪い。このような場合には、バンドマトリックス法やスカイライン法などによる直接法の方が効率的である可能性がある。

参考文献

1)　青木尊之、額田彰：はじめての CUDA プログラミング、工学社
2)　小山田耕二、岡田賢治：CUDA 高速 GPU プログラミング、秀和システム
3)　株式会社クイープ訳：CUDA BY EXAMPLE、インプレスジャパン
4)　David B. Kirk, Wen-mei W. Hwu:CUDA：CUDA プログラミング実践講座、株式会社ボーンデジタル
5)　戸川隼人：マトリクスの数値計算、オーム社
6)　戸川隼人：共役勾配法、教育出版

ひとこと

仲沢武志

グラフィック機能を使って数値演算を並列化する GP-GPU は、比較的安価にハードウェアを装備することが可能であるし、その効果も問題によっては大きい。今後は、プリ処理も含めた効率化を進めることで、作業の合理化をシステムとして進展させることができれば良いと考えている。