GPGPUを利用した疎行列ソルバの高速化

五十嵐 公一†　藤井昭宏†　小柳 義夫†
工学院大学†

1. はじめに

近年画像処理装置であるGPU(Graphics Processing Unit)を汎用演算器として使用するGPGPUに関する研究が盛んに行われている．以前からGPUを演算器として使用することは行われていたが，NVIDIA社が自社のGPUで利用可能なCUDA(Compute Unified Device Architecture)をリリースしたことにより汎用性のあるプログラミングが容易に可能となり，広い分野でGPUによる高速化の研究がなされている．

本研究は，GPUを用いて疎行列線形ソルバを高速化することを目的としている．そこで，代表的な線形解法として共役勾配法を取り上げ，解法の内部で最も重い処理となっている疎行列ベクトル積について分析を行う．疎行列ベクトル積では，格納形式が性能に大きな影響を与えるため，過去に提案されているGPU用の格納形式から，一般によく知られている格納形式まで適用し，GPU上での性能を分析し報告する．

2. GPUのメモリアクセス

前述のようにGPUで疎行列の格納形式が重要になる理由として，GPU上のメモリ転送の構造があげられる．GPUではメモリアクセス命令は16スレッドが同時に実行するようになっている．そして32Byte・64Byte・128Byte単位にアライメントされたデータブロック転送される[1]．
そのため，数Byteのデータを読み込む際にもデータブロックごとの読み込みが必要となる．
さらにスレッドはそれぞれ不連続なアドレスにアクセスすることが多く，メモリの読み書きがかなりのボトルネックになりやすい．これに対する対応策としてCUDAではコアレスアクセスという手法が知られている．コアレスアクセスはcompute capability(CC)のバージョンにより違った挙動をする．CC1.2より前のGPUではメモリは16スレッドが連続したメモリ単位を読み込むときが最適とされており，その条件を満たした際にまとめて処理を行うことができる．CC1.2以降ではメモリアクセスをセグメントごとに分割することによりこの制限が緩和されているが，セグメント境界をまたぐアクセスをするとアクセス速度は低下する．つまり疎行列でどの格納形式がGPUにおいて有用か知ることはGPUの性能を引き出すうえにおいて重要である．

3. 疎行列における格納形式

本研究での格納形式の比較対象として，広く知られているCRS(Compressed Row Storage)形式(図1)とJDS (Jagged Diagonal Storage) 形式(図2)を取り扱う．CRS形式はゼロ要素を詰め，行列を横にアクセスする．行によってベクトル長が不定なのが欠点である．

JDS形式は配列を行の非ゼロ要素数が多い順に行を上から並び替え，行列を縦にアクセスする．最内ループのベクトル長が長くなるためベクトル計算機向きの格納形式といわれている．

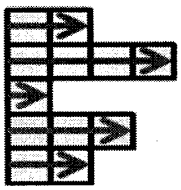

図1. CRS形式

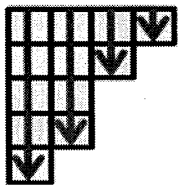

図2. JDS形式

さらに，メモリアクセス効率を上げるためにJDS形式をpaddingし行方向にアクセスすることで高速化する手法が発表されている[2]．この格納形式も本研究では取り扱う．

4. 実装手法

本章では上にあげられたCRS形式，JDS形式，paddingしたJDS(p-JDS)形式の3種類を比較し検討する．

4.1 CRS形式

CRS形式では行方向に並列性があるが，GPUではサイクリックに各スレッドに各行を割り当てることが多い．したがって作業量が多いスレ

Acceleration of Sparse Matrix linear solver
† Kouichi Igarashi
† Akihiro Fujii
† Yoshio Oyanagi
† Kogakuin University

ッドと少ないスレッドに分かれ非効率的になりやすい．また列の要素数が異なるためコアレスアクセスも難しい．以下図では○はスレッドがアクセスする先頭の要素を表し，矢印はそれぞれのスレッドがアクセスしていく方向方向を表す(図 3)．

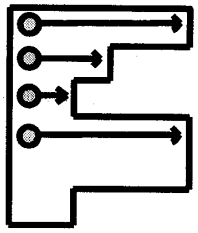

図 3. CRS 形式のアクセス

4.2 JDS 形式

JDS 形式では，並び替えによりベクトル長が長くなることを利用し，すべてのスレッドが同じ列をサイクリックに最終行まで計算し，列ごとに演算を終わらせて行く（図 4）．

4.3 p-JDS 形式

p-JDS 形式では一般的な JDS 形式とは違い JDS を行方向にアクセスし，コアレスアクセスになるように行列を padding する(図 5)．
この手法を取ることにより同じサイクル内の各スレッドは，すべて同じベクトルサイズで演算できるのでメモリが連続空間にあるためコアレスアクセスも可能になり，条件判定も不要になる．

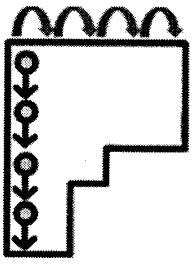

図 4. JDS 形式のアクセス

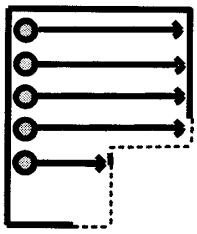

図 5. p-JDS 形式のアクセス

5. 数値実験と評価

共役勾配法の行列ベクトル積以外の内積などの部分は CUDA BLAS[3]ライブラリを使うことにより実装した．実行環境は NVIDIA Geforce GTX275 を使い，入力行列として行数 307,200 の 5 点差分の問題を与えた．

結果は以下の図 6．図 7 のようになった．

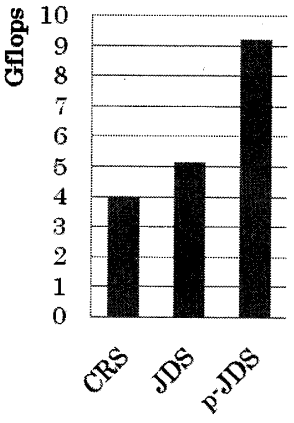

図 6

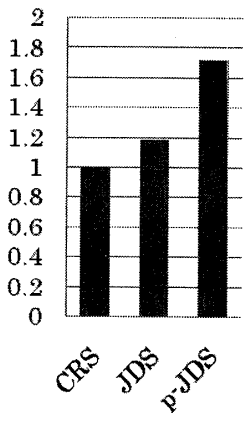

図 7

図 6. CG 法の行列ベクトル積部の各形式による演算性能
図 7. CRS 形式の全体の演算時間を 1 とした速度向上率

図 6．図 7 より p-JDS 形式が一番高速に演算できることが確認できる．これは padding したことにより条件文がなくなったことと，効率良くコアレスアクセスが行われる為である．今回の入力行列は各行の非ゼロ要素数がほぼ一定であり，CRS,JDS 形式でも想定以上の性能がでた．しかし CRS 形式では各行の非ゼロ要素数が不規則になるほど性能は著しく落ちるものと考えられる．JDS 形式では行数がスレッド数が割り切れない場合の為に，条件判定が入っており性能が落ちたものと考えられる．行数がスレッド数で割れるように行方向に padding することにより性能の向上が見られるかもしれない．

6. おわりに

本研究では，疎行列の格納形式により GPU の実効性能に差がでることを確認でき，また 5 点差分の疎行列に対して 9GFLOPS の疎行列ベクトル積が実現できた．今後の課題としては，さまざまな入力行列を与えることによりそれぞれの格納形式の評価を行う．また行列ベクトル積部分の配列の一部を global メモリより高速なメモリに載せることにより疎行列ソルバの更なる高速化も行う．

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(2009)
[2] ALI CEVAHIR, AKIRA NUKADA, SATOSHI MATSUOKA: An Efficient Conjugate Gradient Solver on Double Precision Multi-GPU Systems, SACSIS (2009)
[3] http://developer.download.nvidia.com/compute/cuda/2_1/toolkit/docs/CUBLAS_Library_2.1.pdf