1M-8

ブロック圧縮行格納形式への高速データ構造変換方式の開発

佐藤真之介† 藤井昭宏‡ 田中輝雄‡
工学院大学大学院情報学専攻† 工学院大学情報学部‡

1. はじめに

大規模なシミュレーションでは，問題を差分化することにより現れる疎行列ベクトル積(SpMV)がよく用いられている．SpMV は，形状によって性能が大きく異なる．そのため自動的にチューニングするライブラリの作成が必要である．

一般的に SpMV では，メモリを節約し演算量を削減するために，疎行列を非零要素のみに圧縮して扱う．疎行列のデータ格納形式の一つとして圧縮行格納形式(CRS)[1]が用いられている．疎行列の形状によっては，CRS 形式のデータ構造をブロック化して保持するブロック圧縮行格納形式(BCRS)に変換して扱うことにより，SpMV を高速化することができる．通常，疎行列は CRS 形式で確保されている．BCRS 形式を扱うには変換処理が必要となる．この変換処理は，SpMV の処理時間と比較して多くの時間がかかる．

本研究では，BCRS 形式への変換処理の時間を含めた SpMV の処理時間について論じる．

2. 疎行列格納形式

疎行列は行列内の零要素の割合が多いため，零要素を省いて格納することで圧縮している．アルゴリズムの違いによって様々な格納形式が提案されている．本研究では，CRS 形式と BCRS 形式を用いている．CRS 形式と BCRS 形式の構造を図 1 に示す．

まず，CRS 形式は疎行列の非零要素を行方向に格納していく形式である．データを保持するために，非零要素の値を格納する配列 val，要素の列番号を格納する col_ind, 要素上での行番号の開始位置を格納する row_ptr といった 3 つのベクトルを用いる．

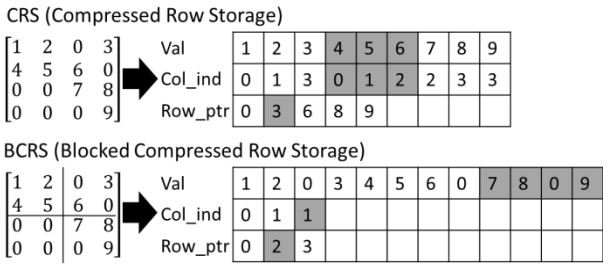


図 1 CRS 形式と BCRS 形式の構造

次に，BCRS 形式は疎行列を零要素を含むブロックに分割し，ブロック毎にデータを格納していく形式である．ブロック単位で格納するため，もしブロック内に零要素がある場合はその零要素も格納する．BCRS 形式のベクトルは，val には格納された非零要素に加えブロック内の零要素，col_ind にはブロック単位でみた列番号，row_ptr には col_ind 上でのブロック幅ごとの行番号の開始位置を格納する．

3. BCRS 形式への変換

大規模な CRS 形式の疎行列データを BCRS 形式に変換する場合，CRS 形式の要素を何度も探索すると，変換処理の時間が大きくなってしまう．そのため一度の探索で変換処理を行う方式とした．

まず，CRS 形式の要素をブロック幅の各行の先頭の非零要素を登録する．次に，各行の先頭の非零要素のブロック幅における列の値を比較し，その値が小さい非零要素から格納する．格納する際に，登録された先頭の非零要素のブロック幅内に別の非零要素が存在するかを確認し，先頭の非零要素とブロック幅内の非零要素を同時に格納する．この工程をブロック幅ごとに繰り返すことで，一度の探索での変換処理を実装した．なお，CRS 形式の疎行列データはソーティングされていると仮定する．

Development of high-speed data structures conversion system to Block compressed row storage format
Shinnosuke Sato†, Akihiro Fujii‡ and Teruo Tanaka‡
† Major in Informatics, Kogakuin University
‡ Faculty of Infomatics, Kogakuin University

Copyright ©2013 Information Processing Society of Japan.
All Rights Reserved.

4. 実験

実験には，Intel Xeon E5570 2.93GHz ，コンパイラは gcc 4.1.2，オプションは-O3 を使用した．

実験に用いた疎行列は，The University of Florida Sparse Matrix Collection[2] の 10 種類の疎行列データである．この疎行列データは，BCRS 形式に変換することで CRS 形式より効果が得られた疎行列データである．使用した疎行列を表 1 に示す．size は行数，nnz は行列内の非零要素の数である．また，SpMV には，疎行列計算ライブラリである OSKI(Optimized Sparse Kernel Interface)[3]を用いた．

まず，CRS 形式の SpMV の処理時間と BCRS 形式の SpMV と変換処理の合計の処理時間を比較した結果を図 2 に示す．今回用いた疎行列データでは，BCRS 形式の SpMV の処理時間のほうが，CRS 形式の SpMV の処理時間より速い．これに変換処理を考慮すると，BCRS 形式の SpMV の処理時間のほうが 3~7 倍の時間がかかっている．

表 1　実験に使用した疎行列データ

データ名	size	nnz
Kuu	7,102	340,200
olafu	16,146	1,015,156
raefsky3	21,200	1,488,768
bcsstk36	23,052	1,143,140
bcsstk16	4,884	290,378
s2rmq4m1	5,489	263,351
bcsstk15	3,948	117,816
msc01440	1,440	44,998
Na5	5,832	305,630
nasa4704	4,704	104,756

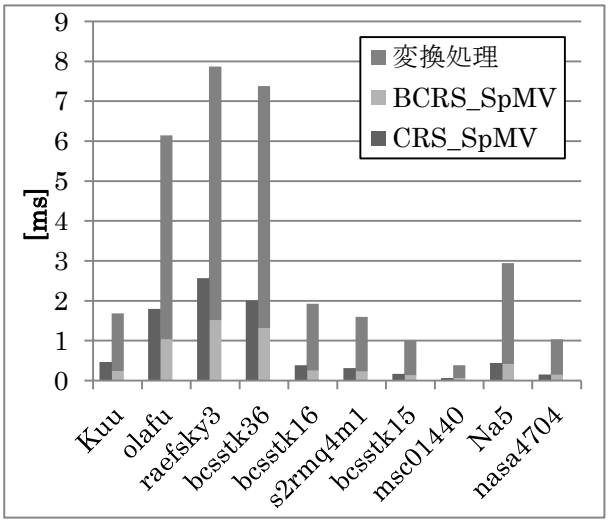


図 2　変換処理含めた SpMV の比較

表 2　変換コストと反復回数

データ名	変換コスト	反復回数
Kuu	6.0	7
olafu	4.9	7
raefsky3	4.2	7
bcsstk36	4.6	9
bcsstk16	6.7	13
s2rmq4m1	6.1	16
bcsstk15	6.7	23
msc01440	6.1	29
Na5	6.1	106
nasa4704	6.2	122

次に，変換コストと反復回数を表 2 に示す．変換コストとは，BCRS 形式の SpMV と変換処理の時間の比である．反復回数とは，反復して行われる SpMV において変換処理を含めた BCRS 形式の SpMV が、CRS 形式の SpMV より処理時間が速くなる反復回数である．

問題によって，BCRS 形式の SpMV が効果の得られる反復回数には，最小が 7 回，最大で 122 回と大きな差があった．また，変換コストが小さい疎行列の”olafu”，”raefsky3”，”bcsstk36”では，7~9 回以上の反復回数で効果が得られることが確認できた．問題ごとの効果が得られる反復回数の指針を示した．

5. まとめ

本研究では，BCRS 形式への変換処理の時間を含めた SpMV の処理時間について論じた．

実験の結果，SpMV と比較した変換処理にかかる時間と，変換処理を含む BCRS 形式の SpMV が CRS 形式の SpMV より実用性がある反復回数を示した．

今後の課題は，変換処理の時間を考慮し，実用性の高いライブラリを構築することである．

参考文献

[1] Barrett, R., et al.: Templates for the Solution of Linear Systems:Building Blocks for Iterative Methods, SIAM pp.57-65(1994)

[2] The University of Florida Sparse Matrix Collection, http://www.cise.ufl.edu/research/sparse/matrices/

[3] Richard Vuduc，James W.Demmel and Katherine A. Yelick The Optimized Sparse Kernel Interface (OSKI) Library_User’s Guide for Version 1.0.1h June 25, 2007

Copyright ©2013 Information Processing Society of Japan.
All Rights Reserved.