前処理付き反復法と疎行列格納手法

中 島 研 吾

東京大学情報基盤センター

1. はじめに

本稿は 2015 年 2 月 26〜27 日に実施された「大規模 HPC チャレンジ」の概要について述べたものである．

本研究では，有限体積法によるポアソン方程式ソルバー〔1〕から導かれる対称正定な疎行列を係数とする連立一次方程式を不完全コレスキー分解前処理付き共役勾配法（Preconditioned Conjugate Gradient Method by Incomplete Cholesky Factorization，ICCG 法）によって解く場合を想定して，係数行列格納手法，特に Ellpack-Itpack（ELL）形式とその拡張に着目した検討を実施した〔1,2〕．Fujitsu PRIMEHPC FX10（Oakleaf-FX）において以下に示す 3 種類の OpenMP/MPI ハイブリッド並列プログラミングモデルを使用した評価を実施した．

- **Hybrid 4×4（HB 4×4）**：各ノードにスレッド数 4 の MPI プロセスを 4 つ起動する
- **Hybrid 8×2（HB 8×2）**：各ノードにスレッド数 8 の MPI プロセスを 2 つ起動する
- **Hybrid 16×1（HB 16×1）**：各ノードにスレッド数 16 の MPI プロセスを 1 つ起動する

2. 対象アプリケーション

本稿で対象とするアプリケーションは図 1 に示す差分格子によってメッシュ分割された三次元領域において，以下のポアソン方程式を解くものである〔1〕：

$$\Delta\phi = \frac{\partial^2\phi}{\partial x^2} + \frac{\partial^2\phi}{\partial y^2} + \frac{\partial^2\phi}{\partial z^2} = f \tag{1}$$

$$\phi = 0 @ z = z_{\max} \tag{2}$$

形状は規則正しい差分格子であるが，プログラムの中では，一般性を持たせるために，有限体積法に基づき，非構造格子型のデータとして考慮する．

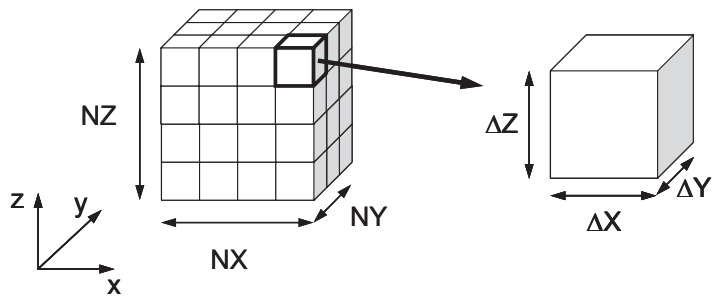

図 1　三次元ポアソン方程式ソルバーの解析対象，差分格子の各メッシュは直方体（辺長さはΔX，ΔY，ΔZ），X，Y，Z 各方向のメッシュ数は NX，NY，NZ

図 1 における任意のメッシュ i の各面（6 面）を通過するフラックスについて，式（1）により以下に示す式（3）が得られる：

$$\left[\sum_{k=1}^{6}\frac{S_{ik}}{d_{ik}}\right]\phi_i-\left[\sum_{k=1}^{6}\frac{S_{ik}}{d_{ik}}\phi_k\right]=+V_i f_i \qquad (3)$$

ここで，S_{ik}：メッシュ i と隣接メッシュ k 間の表面積，d_{ik}：メッシュ i-k 重心間の距離，V_i：メッシュ i の体積，f_i：メッシュ i の体積あたりフラックスである．これは各メッシュ i について成立する式であり，全メッシュ数を N とすると，N 個の方程式を連立させて，境界条件を適用し，連立一次方程式 $[A]\{\phi\}=\{b\}$ を解くことで解を得る．式（3）の左辺第一項は$[A]$の対角項，第二項は非対角項，右辺は$\{b\}$に対応する．各メッシュ i に対応する非対角成分数は最大 6 個であるので，係数行列$[A]$は疎（sparse）な行列となる．

係数行列$[A]$は対称かつ正定（Symmetric Positive Definite，SPD）であるため，前処理付き共役勾配法（Preconditioned Conjugate Gradient Method）を適用する．前処理手法としては，対称行列向けに広く使用されている不完全コレスキー分解（Incomplete Cholesky Factorization，IC）を使用する〔1,2〕．本研究では，係数行列は対称であるが，プログラム内では上下三角成分を別々に記憶している〔1〕．本研究では， fill-in を考慮しない IC(0)を使用している．

不完全コレスキー分解を前処理手法とする共役勾配法を ICCG 法と呼ぶ．ICCG 法では，不完全コレスキー分解生成時，前進代入，後退代入でメモリへの書き込みと参照が同時に生じ，データ依存性が発生する可能があるため，リオーダリングが必要である〔1〕．

本研究では OpenMP/MPI ハイブリッド並列プログラミングモデルの適用を前提としており，全領域を各 MPI プロセスにおいて均等な直方体に分割し，各 MPI プロセス内において OpenMP による並列化を実施している（後述）．

各 MPI プロセスに対しては筆者による先行研究〔3,4〕に基づき，オーバーラップを有する局所分散データ構造を適用している（図 2）．IC(0)前処理は各 MPI プロセスに独立に適用するブロック Jacobi 型局所的手法〔4〕を適用しており，局所前処理の安定化のために，Additive Schwartz Domain Decomposition〔3,4,5〕を各反復に一回適用している．

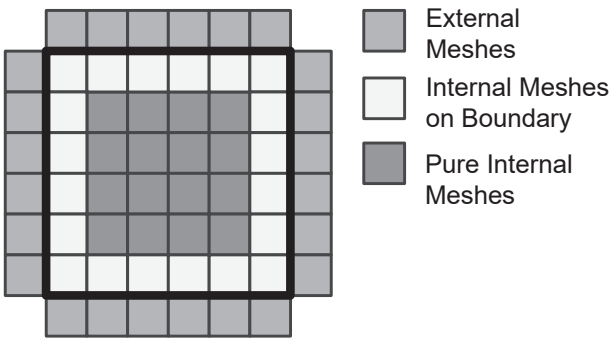

図 2　オーバーラップを有する局所並列データ構造〔3,4〕

3. OpenMP によるノード内並列化

ハイブリッド並列プログラミングモデルでは，各ノード（ソケット）に対応した局所データを OpenMP などのマルチスレッド的な手法によって並列化に処理する．ICCG 法では不完全コレスキー分解，前進代入，後退代入のプロセスでメモリへの書き込みと参照が同時に生じ，データ依存性が発生する可能がある．これを回避するための方法として色づけ（coloring）によるリオーダリング（reordering）が広く使用されている〔1,3,4〕．お互いに依存性を持たない要素群を同じ色に色づけすることによって，色内での並列処理が可能となる．本研究では，並列性に優れたマルチカラー法（Multicoloring, MC）とより安定した収束を示す Reverse Cuthill-McKee（RCM）法を組み合わせ，RCM 法に Cyclic マルチカラー法（Cyclic Multicoloring, CM）を適用した CM-RCM(k)法を使用した〔1,4〕．図 3 は CM-RCM(k)法による並び替え例である．ここでは，4 色に色分けされており（CM-RCM(4)），たとえば，RCM の第 1，第 5，第 9，第 13 組の要素群が CM-RCM(4)法の第 1 色に分類される．各色には 16 の要素が含まれる．CM-RCM(k)法における色数（=k）は，各色内の要素が依存性を持たない程度に大きい必要がある．

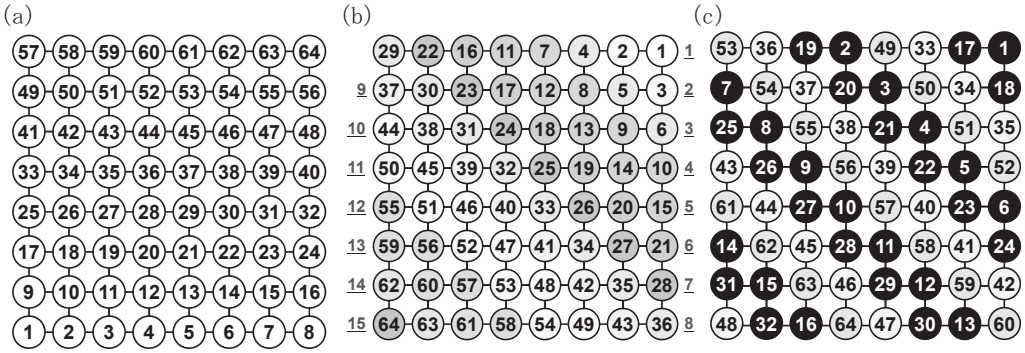

図 3　CM-RCM(k)法による色づけとリオーダリング，（a）元のグラフ，（b）RCM 法によるリオーダリング（赤字はレベルセット番号），（c）CM-RCM(k)法による再リオーダリング（4 色：CM-RCM(4)），各色内の要素数は 16 でバランス

更に，図 4 に示すように：

- 同一の色に属する要素は独立であり，並列に計算可能
- 「色」の順番に各要素を番号付けする
- 色内の要素を各スレッドに振り分ける

という Coalesced Numbering〔1〕を採用している．

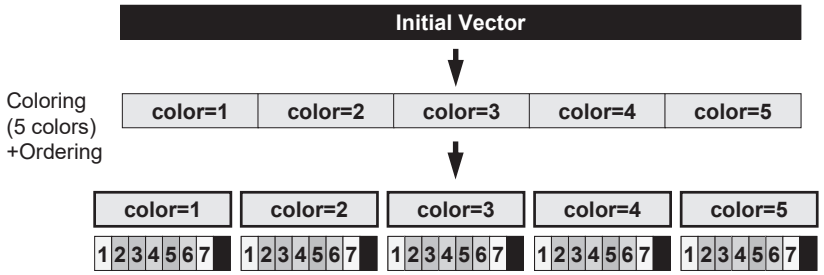

図 4　CM-RCM(k)法による番号付け（Coalesced Numbering）（5 色，8 スレッド）

4. 疎行列格納方法

疎行列計算は間接参照を含むため memory-bound なプロセスである．従って疎行列演算において，演算性能と比較してメモリ転送性能の低い昨今の計算機の性能を引き出すことは困難である．係数行列の格納形式が性能に影響することは広く知られており，様々な手法が提案されている．

Compressed Row Storage（CRS）形式は，図 5（a）に示すように疎行列の非零成分のみを記憶する方法である．Ellpack-Itpack（ELL）形式は各行における非零非対角成分数を最大非零非対角成分数に固定する方法であり（図 5（b）），実際に非零非対角成分が存在しない部分は係数=0として計算する．CRS と比較して高いメモリアクセス効率が得られることが知られているが，計算量，必要記憶容量ともに増加する．

これまで，行列格納形式に関する研究は行列ベクトル積に関するものが主であったが，著者等は IC 法，ILU 法（Incomplete LU Factorization，非対称行列向けの前処理手法）等の前処理のようなデータ依存性を有するプロセスについて検討を実施している〔1,3〕．差分法に見られるような規則正しいメッシュでは，各行における非零非対角成分数がほぼ固定されているため，その性質を適用することが可能である．本研究で対象としている図 1 に示すような形状では，辞書的な初期番号付けにおいては，上三角成分（自分より番号の大きい隣接要素），下三角成分（自分より番号の小さい隣接要素）の最大数は各要素において最大 3 であり，容易に ELL 形式を適用できる．スレッド並列化のためのリオーダリングに RCM 法を適用した場合もこの関係は変わらない〔1〕．また，CM-RCM(k)法を適用した場合は，図 6 に示すように，総色数を NC とすると以下のようになることがわかっている〔1〕：

- 第 1 色：下三角成分数：0，上三角成分数：最大 6
- 第 2 色～第（NC-1）色：上下三角成分ともに最大 3
- 第 NC 色：下三角成分数：最大 6，上三角成分数：0

著者等の先行研究〔1,3〕では ELL 形式を適用する場合に外側ループを行方向，内側ループを列方向とする Row-wise な手法を適用してきた（図 7）．図 6 に示すようなやや不規則行列に適用する場合，無駄な計算を避けるためには，非零非対角成分の数の順番に並び替え，ループ長を変化させる手法が考えられる．図 8 に示す例では，非零非対角成分が 4 以上の要素（5 行目まで）と 3 以下の要素（6 行目以降）に分類する．図 7 の例に基づけば，5 行目までは「k=1,6」，6 行目以降は「k=1,3」とすることができる．

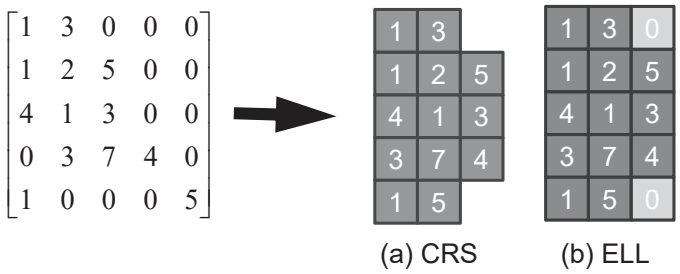

図 5　疎行列の格納形式（a）CRS，（b）ELL

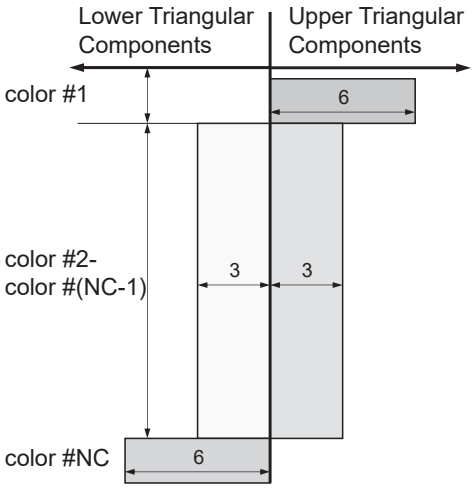

図 6　CM-RCM(k) 法における上下三角成分数（NC：総色数）

```
!$omp parallel
    do icol= 1, NCOLORtot
!$omp do
    do ip  = 1, PEsmpTOT
      do i= Index(ip-1,icol)+1, Index(ip,icol)
        do k= 1, 6
          Z(i)= Z(i) - AML(k,i)*Z(IAML(k,i))
        enddo
        Z(i)= Z(i) / DD(i)
      enddo
    enddo
    enddo
!omp end parallel
```

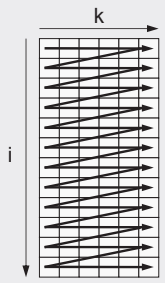

図 7　ELL 形式の前進代入への適用例（Row-wise），非零非対角成分の最大数=6．NCOLORtot：総色数，PEsmpTOT：総スレッド数，Index(ip,icol)：各色，スレッドに属する要素総数，AML(k,i)：非零非対角成分，IAML(k,i)：非零非対角成分（列番号），DD(i)：対角成分．

ただしこのような手法は，やや非効率的であり，図 8 の 6 行目以降を計算する場合に AUnew(4,i)～AUnew(6,i)が例えキャッシュに載っていたとしても棄却されてしまう．そこで，ELL 形式を拡張し，複数の配列を使用して，より効率的な計算を実施する手法として，Sliced-ELL 形式〔1,2,3〕が提案されている（図 9）．

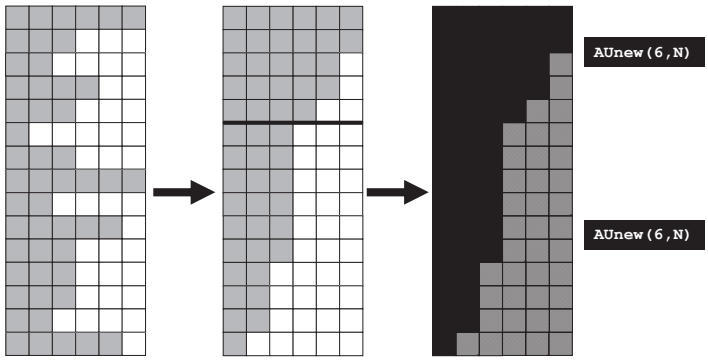

図 8　ELL 形式の不規則行列への適用例

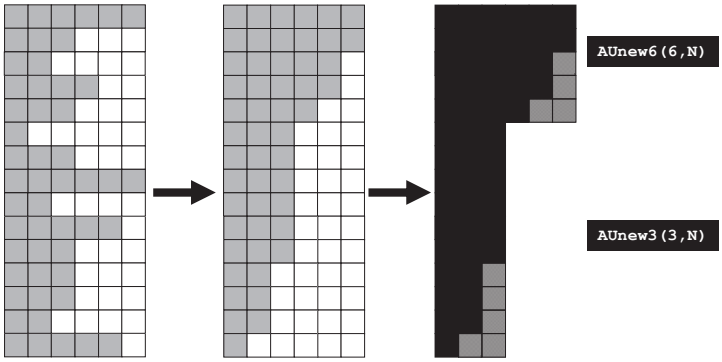

図 9　Sliced-ELL 形式の例〔2〕

5. 計算結果

まず，小規模な問題について予備的評価を実施し，Oakleaf-FX のプロファイラによる性能評価を実施した．ここでは：

- 並列プログラミングモデル：HB 8×2
- ノード数：4，MPI プロセス数：8
- 問題規模：64^3（=262,144）要素／コア，16,777,216 要素（全体）
- CM-RCM(k)色数：75

とし，収束までの ICCG 法部分の性能を測定した．表 1 に結果を示す：

表 1　ICCG 法の性能評価（4 ノード実行，ノード当り）

行列格納方法	計算時間（sec.）	メモリースループット（GB/sec）	命令数
CRS	27.8	41.3	9.30×10^{11}
ELL（図 8）	23.0	64.8	3.26×10^{11}
Sliced ELL（図 9）	16.8	62.3	3.06×10^{11}

ELL，Sliced ELL は CRS と比較してメモリースループットが高く，命令数も少ないことがわかる．CRS と比較した速度向上率は ELL：21%，Sliced ELL：65%であり，Sliced ELL によって大幅な速度向上が得られている．ELL，Sliced ELL のメモリースループットは STREAM Triad のノード当り性能（64.7 GB/sec）に匹敵する〔6〕．

続いて，コア当りの問題サイズを同じにして，4,800 ノードにおける計算を実施した．計算結果（CG 法の計算時間，反復回数）を表 2 に示す．未知数の総数は約 210 億である．3 つのハイブリッド並列プログラミングモデルの中では，HB 4×4 の性能が全般的に高く，CRS-RCM⇒Sliced ELL-RCM で約 69%の性能向上が得られる．HB 4×4 の場合 RCM のレベル数は 318 であるが，CM-RCM(75)（色数：75）を適用すると，収束までの反復回数は 4,782⇒4,785 とわずかに増加するものの，OpenMP の同期オーバーヘッドが減少するため，更に約 10%の性能向上が得られている．一般に，RCM⇒CM-RCM(k)によって反復回数はやや増えるが OpenMP の同期オーバーヘッド減少によって計算時間が減少する場合もある．HB 8×2 の場合は 4,760⇒4,839

と反復回数が増加しており，計算時間も 243.3 秒⇒316.9 秒と増加している．計算時間増加の割合（約 30%）は反復回数増加（約 1.66%）と比較して大きいが，この原因については不明である．HB 16×1 では RCM⇒CM-RCM(150)によって反復回数は約 4.5%，計算時間は約 16.55

表 2　ICCG 法の性能評価（計算時間（秒）（反復回数）），4,800 ノード，総自由度数：20,132,659,200

	HB 4×4 RCM レベル数：318 CM-RCM(75)	**HB 8×2** RCM レベル数：382 CM-RCM(75)	**HB 16×1** RCM レベル数：510 CM-RCM(150)
CRS-RCM	409.7 (4,782)	417.8 (4,760)	567.3 (4,749)
ELL-RCM	323.5 (4,782)	326.8 (4,760)	343.0 (4,749)
Sliced ELL-RCM	242.5 (4,782)	243.3 (4,760)	272.8 (4,749)
Sliced ELL-CM-RCM(k)	228.9 (4,785)	316.9 (4,839)	229.8 (4,545)

6. まとめ

疎行列格納法の前処理付き反復法の性能への影響を評価した．ELL，Sliced ELL によって，Oakleaf-FX 4,800 ノードにおいても大幅な性能向上が達成された．ノード内 OpenMP 化に適用する CM-RCM(k)の色数は収束に影響する．色数 k の増加によって概して反復回数は減るが，OpenMP の同期オーバーヘッドの増加により，一反復あたりの計算時間は増加する．MPI による並列化を実施した場合，色数と反復解法，計算時間の関係は一様ではないため，最適色数の探索法も含めた検討が必要である．

参考文献

[1] 中島研吾，拡張型 Sliced-ELL 行列格納手法に基づくメニィコア向け疎行列ソルバー，情報処理学会研究報告（HPC-147-3）（2014）

[2] Monakov, A., A. Lokhmotov, and A. Avetisyan, Automatically tuning sparse matrix-vector multiplication for GPU architectures, Lecture Notes in Computer Science 5952 （2010）112-125

[3] Nakajima, K., Optimization of Serial and Parallel Communications for Parallel Geometric Multigrid Method, Proceedings of IEEE ICPADS 2014（2014）25-32

[4] Nakajima, K., Parallel Iterative Solvers of GeoFEM with Selective Blocking Preconditioning for Nonlinear Contact Problems on the Earth Simulator, ACM/IEEE Proceedings of SC2003, (2003)

[5] Smith, B., P. Bjφrstad, and W. Gropp, *Domain Decomposition, Parallel Multilevel Methods for Elliptic Partial Differential Equations*, Cambridge Press (1996)

[6] STREAM: https://www.cs.virginia.edu/stream/