Vol.2014-HPC-146 No.22
2014/10/3

有限要素法係数行列生成プロセスのマルチコア・メニィコア環境における最適化

中島研吾[†1 †2] 大島聡史[†1 †2] 塙 敏博[†1]

有限要素法は偏微分方程式の数値解法として様々な科学技術シミュレーションに使用されている．有限要素法において係数行列生成部は疎行列計算と並んで計算時間を要するプロセスである．本研究では，OpenMP によってマルチスレッド並列化された係数行列生成部に関する 3 種類の実装によるプログラムを様々なマルチコア，メニィコア環境で評価した事例を紹介し，自動チューニングに向けた方策について検討する．

Optimization of matrix assembling process in FEM applications on multicore/manycore architectures

KENGO NAKAJIMA[†1 †2] SATOSHI OHSHIMA[†1 †2] TOSHIHIRO HANAWA[†1]

Finite-element method (FEM) is one of the most well-known numerical methods for solving partial differential equations (PDE), and applied to various kinds of scientific simulations. Matrix assembling and sparse matrix solver are the most expensive processes in finite-element procedures. In the present work, the matrix assembling process is parallelized using OpenMP, and three types of implementations are evaluated on various types of multicore/manycore architectures. Results and analyses of computations and strategies towards automatic tuning will be described in the presentation.

1. はじめに

有限要素法に代表される偏微分方程式の数値解法において，最も計算時間を要するプロセスは大規模な疎行列を係数行列とする連立一次方程式の求解であり，その最適化に向けて様々な試みがなされてきた（例えば〔1〕）．有限要素法では，各要素における積分方程式から密な要素行列を計算し，これを重ね合わせることによって疎な全体係数行列を導出する．このような係数行列生成部（Matrix Assembling）は連立一次方程式求解部と比較してアプリケーションに依存する部分も多く，計算プロセスの最適化に関する研究は，これまであまり行われて来なかったのが現状である〔2〕．一般に，係数行列生成のコストは連立一次方程式求解よりは少ないものの，例えば非線形計算の場合には係数行列を反復のたびに計算し直す必要があり，できるだけ効率を高める工夫が必要である．

著者等は科学技術振興機構戦略的創造研究推進事業（CREST）「ポストペタスケール高性能計算に資するシステムソフトウェア技術の創出」の 1 プロジェクトとして実施されている「ppOpen-HPC : 自動チューニング機構を有するアプリケーション開発・実行環境」〔3,4〕において有限要素法に代表される様々な科学技術計算手法の各計算プロセスのマルチコア, メニィコアアーキテクチャ向け最適化, ライブラリ化と自動チューニング手法の適用に関する研究開発を実施している．有限要素法の係数行列生成部もその対象の一つであり，最適化と自動チューニング手法の検討が進められている．本研究は，ppOpen-HPC における有限要素法アプリケーション開発用フレームワークである ppOpen-APPL/FEM のフィージビリティスタディとして実施したものである．

本論文では，以下，係数行列生成部の処理の概要とその最適化，計算環境の概要，計算結果とその分析について紹介し，最後に自動チューニングに向けての方策についてまとめる．ppOpen-HPC はメッセージパッシング（MPI）とプロセス内スレッド並列（OpenMP）を組み合わせたハイブリッド並列プログラミングモデルを基本としているが，本研究では特に各計算ノード上でのスレッド並列化に着目し，MPI プロセス数を 1 として計算を実施した．

2. 係数行列生成部

2.1 対象アプリケーション

本研究で対象としているのは，GeoFEM プロジェクト〔5,6,7〕で開発された並列有限要素法アプリケーションを元に整備した性能評価のためのベンチマークプログラム「GeoFEM／Cube」である．本ベンチマークは，三次元弾性静解析問題（Cube 型モデル（図 1））に関する並列前処理付き反復法による疎行列ソルバーの実行時性能（GFLOPS 値）を様々な条件下で計測するものである．要素タイプは三次元一次六面体要素（tri-linear）であり，各要素 8 つの節点を有している．プログラムは全て OpenMP ディレクティヴを含む FORTRAN90 および MPI で記述されている． GeoFEM で採用されている局所分散データ構造

†1 東京大学情報基盤センター
Information Technology Center, The University of Tokyo
†2 科学技術振興機構 CREST
CREST, Japan Science and Technology Agency

〔5〕を使用しており，マルチカラー法等に基づくリオーダリング手法によりマルチコアプロセッサにおいて高い性能が発揮できるように最適化されている．また，MPI, OpenMP, Hybrid（OpenMP＋MPI）の全ての環境で稼動する．

三次元弾性静解析問題では係数行列が対称正定な疎行列となることから，前処理を施した共役勾配法（Conjugate Gradient，CG）法によって連立一次方程式を解いている．

本来の GeoFEM／Cube ベンチマークでは前処理手法として Symmetric Gauss Seidel（SGS）を使用しているが，本研究では Block Diagonal Scaling 法〔5,6〕を使用しており，OpenMP 並列化した場合の前処理プロセスにおけるデータ依存性を考慮する必要がないため，節点のリオーダリングは実施していない．また，三次元弾性問題では 1 節点あたり 3 つの自由度があるため，これらを 1 つのブロックとして取り扱っている．係数行列はこのブロック型の特性を利用したブロック CRS 形式（Compressed Row Storage）によって格納されている．

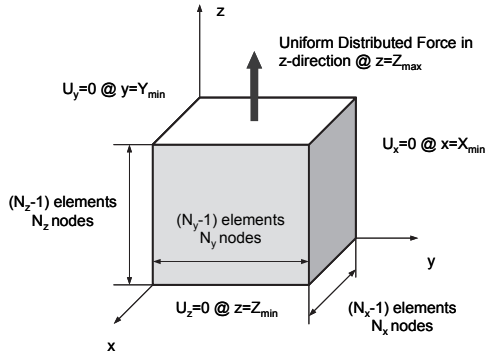


図 1　GeoFEM／Cube の解析対象（Cube モデル）

2.2 係数行列生成部

有限要素法では，要素毎に得られる積分方程式から導かれる密な要素行列を重ね合わせて疎な全体行列を生成する．図 2 に示すような二次元一次四角形要素（bi-linear, 双一次）では各要素の節点数が 4 であるので各節点の自由度数が 1 であれば，要素行列は 4×4 の密行列となる．

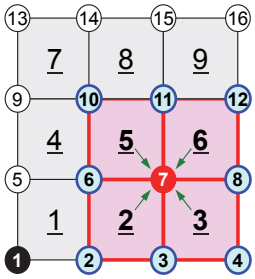


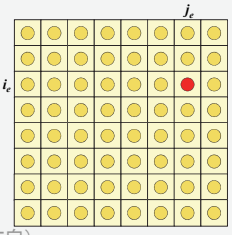
図 2　要素行列の重ね合わせによる全体行列の生成

図 2 の 7 番の節点は周囲の 4 要素（2，3，5，6 番）からの寄与がある．したがって，係数行列生成のプロセスを OpenMP 等でスレッド並列化した場合，ある節点に複数の要素から同時にデータの書き込みが発生する場合がある．要素行列の重ね合わせを実施する際にはマルチカラーオーダリング等を使用してこのような同時書き込みの発生を回避する方法が広く使用されている〔2〕．図 3 は本研究における行列生成部の処理の概要を示すものである．三次元一次六面体要素を使用しているため，要素あたりの節点数は 8 であり，8×8 の密な要素行列が生成される（実際は各節点に 3 つの自由度があるため，要素行列は 24×24 となる）．ループの構成としては一番外側が各要素に関するループ（do icel= 1, ICELTOT，ICELTOT：全要素数）である．その内側の二重ループ（do ie=1, 8，do je= 1, 8）は要素行列を生成するためのループであり，各要素の節点が 8 個あることに対応している．更にその内側にはガウスの積分公式に対応する三重ループ（do ipn/jpn/kpn =1, 8）がある．

```
do icel= 1, ICELTOT      要素ループ
  ＜8節点の座標から，ガウス積分点における，
    形状関数の「全体座標系」における微分，
    およびヤコビアンを算出（JACOBI）＞

  do ie= 1, 8          局所節点番号
    do je= 1, 8        局所節点番号
      ＜全体節点番号：ip，jp＞
      ＜A_ip,jpのitemにおけるアドレス：kk＞

      do kpn= 1, 2     ガウス積分点番号（ζ方向）
        do jpn= 1, 2   ガウス積分点番号（η方向）
          do ipn= 1, 2 ガウス積分点番号（ξ方向）
            ＜要素積分⇒要素行列成分計算＞
          enddo
        enddo
      enddo
    enddo
  enddo
  ＜要素行列成分の全体行列への加算＞
enddo
```



$[k_{i_e,j_e}]$　$(i_e, j_e = 1 \sim 8)$

図 3　GeoFEM／Cube における係数行列生成の処理

2.3 係数行列生成部の実装手法

図 3 に示す処理をまとめると以下の 4 つとなる：

① 各積分点におけるヤコビアン，形状関数導関数計算
② 要素行列成分の全体行列（疎行列）におけるアドレス探索
③ ガウス数値積分，要素行列成分計算
④ 要素行列成分の全体行列への加算

```
   do color= 1, COLORtot
!$OMP PARALLEL DO
   do icel= col_index(color-1)+1, col_index(color)
     ＜①各積分点におけるヤコビアン，形状関数導関数計算＞
     do ie= 1, 8; do je= 1, 8
       ＜②要素行列成分の全体行列（疎行列）におけるアドレス探索＞
       ＜③ガウス数値積分，要素行列成分計算＞
       ＜④要素行列成分の全体行列への加算＞
     enddo; enddo
   enddo
   enddo
```

図 4　GeoFEM／Cube オリジナル実装（Original）の概要（COLORtot：要素色数（=8），col_index(color)：各色に含まれる要素数）

図 4 は，図 3 に示した処理内容を，上記①～④を考慮して簡略化し，OpenMP によるスレッド並列化が適用されていると仮定したものである．COLORtot はマルチカラーオー

ダリングの色数であり，本ケースのような規則正しい形状の場合には 8 である．配列 col_index(color)は各色に含まれる要素数である．図 4 に示すようにオリジナル実装では，これらの処理を要素毎に実施しており，特に②～④については要素行列の各成分について個別に実施している．各ループの中で，探索，ガウス積分，全体行列への加算などの複雑な処理が繰り返し実施されるため計算効率が低くなっている可能性がある．

有限要素法における疎な係数行列は要素毎に得られる積分方程式から導出される要素行列に基づくものであり，アプリケーションへの依存性が強い．ppOpen-HPC 開発の見地からはアプリケーション開発者の負担をできるだけ軽減することが重要であり，疎行列計算に関わる上記②，④の処理に関わる機能はできるだけ ppOpen-HPC で提供することが望ましい．①もライブラリとして提供が可能な機能であるが，③は最もアプリケーションに最も依存する部分であり，アプリケーション開発者自ら記述する必要がある．

③の部分を他と切り離す場合には，要素行列用配列（1 要素あたり 4.61KByte（=24×24×8÷1,000））のため記憶容量が必要である．また，②の部分を分離する場合には要素行列各成分の疎行列におけるアドレスを記憶するための配列に要素あたり 256Byte が必要である．これらの配列はスレッド並列化を実施する場合には，各スレッドにおいて別途必要となる．したがって，これらの配列を全要素について記憶することは非現実的であり，100 個以下の要素によるブロックを形成し，ブロック毎に計算を実施するのが適切である．

以上を考慮して，図 5，図 6 に示すような実装（Type-A，Type-B）を試みる．ここで BLKSIZ は各ブロックに含まれる要素数，NBLK は要素ブロックの総数である．

Type-A（図 5）

- 図 3 に示した①～④の処理のうち，②，①＋③，④を分離して，3 つのループとする．
- 疎行列アドレス記憶用配列，要素行列用配列のための追加の記憶容量が必要である．

Type-B（図 6）

- 図 3 に示した①～④の処理のうち，②，①＋③＋④を分離して，2 つのループとする．
- 疎行列アドレス記憶用配列のための追加の記憶要領が必要である．要素行列用配列の記憶は不要である．

上記のうち，アプリケーション開発者の負担の少ないのは Type-A である．図 5 に示す②と④の計算を実施しているループは分離してライブラリ化することが可能であり，第 2 のループの計算のうち，①もライブラリ化が可能である．

```
    do color= 1, COLORtot
!$OMP PARALLEL DO
    do ib= 1, NBLK
      do blk= 1, BLKSIZE
        icel: calculated by (col_index, ib, blk)
        do je= 1, 8; do je= 1, 8
          <②要素行列成分の全体行列（疎行列）におけるアドレス探索＋格納>
        enddo; enddo
      enddo
      do blk= 1, BLKSIZ
        icel: calculated by (col_index, ib, blk)
        <①各積分点におけるヤコビアン，形状関数導関数計算>
        do je= 1, 8; do je= 1, 8
          <③ガウス数値積分，要素行列成分計算＋格納>
        enddo; enddo
      enddo
      do blk= 1, BLKSIZ
        icel: calculated by (col_index, ib, blk)
        do je= 1, 8; do je= 1, 8
          <④要素行列成分の全体行列への加算>
        enddo; enddo
      enddo
    enddo
    enddo
```

図 5　Type-A 実装の概要（COLORtot：要素色数（=8），col_index(color)：各色に含まれる要素数，NBLK：要素ブロック総数，BLKSIZ：要素ブロックサイズ，icel：要素番号）

```
    do color= 1, COLORtot
!$OMP PARALLEL DO
    do ib= 1, NBLK
      do blk= 1, BLKSIZE
        icel: calculated by (col_index, ib, blk)
        do je= 1, 8; do je= 1, 8
          <②要素行列成分の全体行列（疎行列）におけるアドレス探索＋格納>
        enddo; enddo
      enddo
      do blk= 1, BLKSIZ
        <①各積分点におけるヤコビアン，形状関数導関数計算>
        icel: calculated by (col_index, ib, blk)
        do je= 1, 8; do je= 1, 8
          <③ガウス数値積分，要素行列成分計算>
          <④要素行列成分の全体行列への加算>
        enddo; enddo
      enddo
    enddo
    enddo
```

図 6　Type-B 実装の概要（COLORtot：要素色数（=8），col_index(color)：各色に含まれる要素数，NBLK：要素ブロック総数，BLKSIZ：要素ブロックサイズ，icel：要素番号）

3. 計算機環境

本研究では以下の 3 種類の計算機環境を使用した：

- FX10：Fujitsu PRIMEHPC FX10 に基づく東京大学情報基盤センターの Oakleaf-FX システム〔8〕
- MIC：Intel Xeon Phi（Knights Corner）
- IvyB：Intel Xeon E5（IvyBridge-EP）

プログラムは Fortran90 で記述してあり，FX10 では富士通製コンパイラ，MIC，IvyB では Intel Comliler（Ver.15）／Intel Parallel Studio XE 2015 を使用した．表 1 に計算機環境の概要を示す．

本研究では，各環境において表 1 に示す 1 ソケットを用いて計算を実施した．1.でも述べたように，MPI プロセス数を 1 とし，ソケット内を OpenMP によりスレッド並列化したプログラムを実行している．表 1 に示すように実際に使用したスレッド数は，FX10：16，MIC：240，IvyB：10 である．IvyB の有効スレッド数は 20 であるが，今回は 10 として実施している．

表 1　各計算環境（1 ソケット）の概要

略　称	FX10	MIC	IvyB
名　称	Fujitsu SPARC64 IX fx	Intel Xeon Phi 5110P (Knights Corner)	Intel Xeon E5-2680 v2 (Ivy-Bridge-EP)
動作周波数（GHz）	1.848	1.053	2.80
コア数（有効スレッド数）	16 (16)	60 (240)	10 (20)
使用スレッド数	16	240	10
メモリ種別	DDR3	GDDR5	DDR3
理論演算性能（GFLOPS）	236.5	1,010.9	224.0
主記憶容量（GB）	32	8	64
理論メモリ性能（GB/sec.）	85.1	320	59.7
キャッシュ構成	L1:32KB/core L2:12MB/socket	L1:32KB/core L2:512KB/core	L1:32KB/core L2:256KB/core L3:25MB/socket
コンパイルオプション	-Kfast, openmp	-O3 -openmp -mmic -align array64byte	-O3 -openmp -ipo -xAVX -align array32byte

4. 計算結果

4.1 計算条件

計算対象としては図 1 に示す Cube モデルで N_x=N_y=N_z=128 とした場合について検討した．したがって，要素数＝2,048,383（=127^3），節点数＝2,097,152（=128^3）である．実装としては，図 4～図 6 に示す GeoFEM／Cube オリジナル，Type-A，Type-B の 3 種類を考慮し，Type-A，Type-B の要素ブロックサイズ（図 5，図 6 の BLKSIZ）は 1～100 の範囲で変化させた．表 2 に実施ケースを示す．

表 2　実施ケース

	実装	要素ブロックサイズ（BLKSIZ）
Org.	オリジナル（図 4）	―
A-001	Type-A（図 5）	1
A-002		2
A-005		5
A-010		10
A-020		20
A-050		50
A-100		100
B-001	Type-B（図 6）	1
B-002		2
B-005		5
B-010		10
B-020		20
B-050		50
B-100		100

4.2 計算結果

図 7 に示す計算結果は各計算機環境における各ケース（実装方法，要素ブロックサイズ）の行列生成部の計算性能を記載したものであり，横軸は Type-A，Type-B 実装における要素ブロックサイズである．オリジナル実装の場合は要素ブロックサイズというパラメータが存在しないため，各計算機環境における結果（●，▲，◆）は BLKSIZ=1 の軸に記載されている．記載されているデータは FX10／Org.（オリジナル実装）の場合の行列生成部計算時間を各ケースにおける行列生成部計算時間で割った値であり，1 より大きい場合は，FX10／Org.よりも速いということになる．

各計算機環境において計算結果を比較すると，まず FX10 においては，B-010，B-020，B-050，B-100 の値が 1 をわずかながら上回っているものの，Type-B による結果はほぼ 1 に近くオリジナル実装と変わらない．Type-A の場合は，Type-B と比較して 10%～15%程度性能が低い．Type-A で最も性能が良い A-010，A-020 の場合はオリジナル実装，Type-B と比較して性能は 92%程度である．

それに対して，MIC，IvyB では Type-A による性能は Type-B と比較して，MIC において 35%，IvyB では 60%程度高くなっている．各ケースとも要素ブロックサイズによる性能の変化は少ないが，Type-A ではブロックサイズが大きくなると性能の低下傾向が見られ，特に MIC／Type-A（△），IvyB／Type-A（◇）ではブロックサイズが 2 を超えると性能が低下していく．Type-A は要素行列を記憶するため，他の手法と比較して要素ブロックサイズの影響を受けやすい．ブロックサイズ＝10 の場合のスレッド当たりの要素行列に要する記憶容量は 46.1KByte である．IvyB では Type-B の性能が他のケースと比較して非常に低い．

FX10 と IvyB は理論演算性能，理論メモリ性能は似通っているが，性能最大値を比較すると IvyB の性能が 60%程度高い.また，MIC は FX10 の約 2 倍の性能である．

図 8 は図 7 の中から，各計算機環境において最大性能を

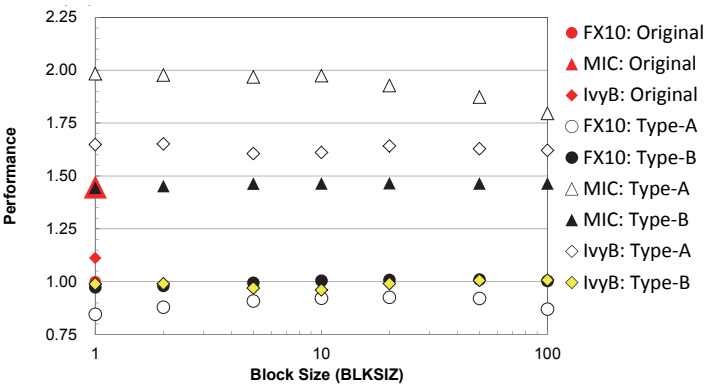


図 7　GeoFEM／Cube 計算結果，係数行列生成部計算性能（FX10，Org.の計算時間で無次元化）（2,048,383 要素，2,097,152 節点），計算機環境，実装方法，要素ブロックサイズの効果，GeoFEM／Cube の結果は BLKSIZ=1 の軸に記載（●，▲，◆）

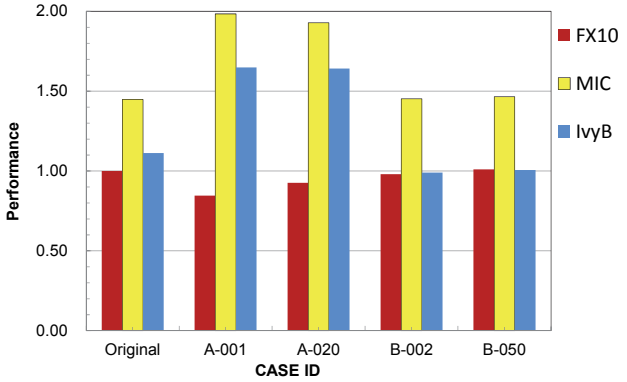


図 8　GeoFEM／Cube 計算結果，係数行列生成部計算性能（FX10，Org.の計算時間で無次元化）（2,048,383 要素，2,097,152 節点），計算機環境，実装方法，要素ブロックサイズの効果

表 3　行列生成部におけるメモリスループット（GB/sec）

	FX10	IvyB
Org.	18.93	23.49
A-001	15.24	31.28
A-020	16.76	30.57
B-002	17.76	21.34
B-050	18.26	22.64
Org.（疎行列ソルバー）（参考）	55.86	53.29

表 3 は図 8 に示す各ケースの行列生成部におけるメモリスループット（GB/sec）である．FX10 は Fujistu PRIMEHPC FX10 専用詳細プロファイラ〔8〕，IvyB は Intel VTune〔9〕によって計測を実施した．参考のために疎行列ソルバー（前処理付き共役勾配法）における値を併記してある．図 9，図 10 は Fujistu PRIMEHPC FX10 専用詳細プロファイラによって算出した，FX10／Org.の場合の行列生成部，疎行列ソルバーの命令数（Instructions），計算時間の内訳である．行列生成部は疎行列ソルバーと比較すると，演算命令数，演算時間の比率が高く，逆に疎行列ソルバーはメモリ，キャッシュアクセスに関する命令や待ち時間の比率が高い．図 11 は Intel VTune を使用して求めた IvyB／Org.の場合の計算時間の内訳である．

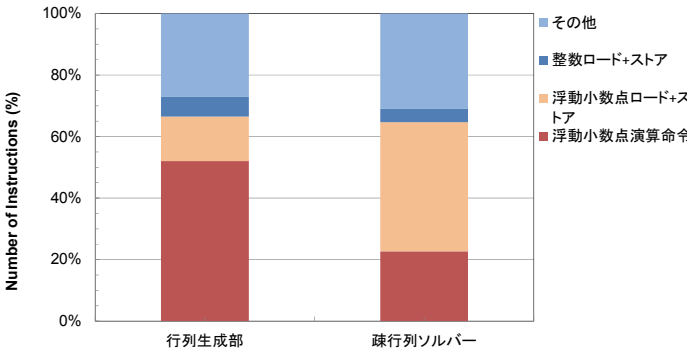


図 9　GeoFEM／Cube 計算結果，FX10／Org.における行列生成部，疎行列ソルバーの命令数（Instructions）の比率（2,048,383 要素，2,097,152 節点）

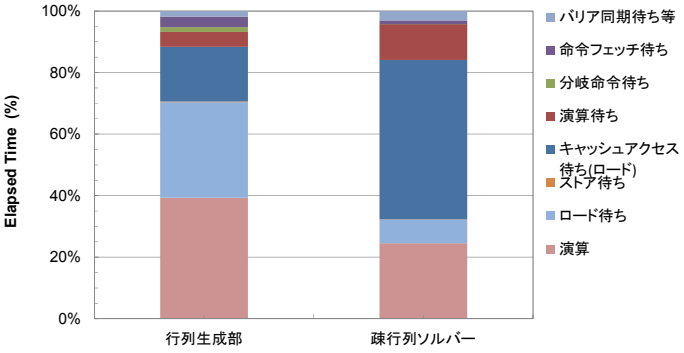


図 10　GeoFEM／Cube 計算結果，FX10／Org.における行列生成部，疎行列ソルバーの計算時間の比率（2,048,383 要素，2,097,152 節点）

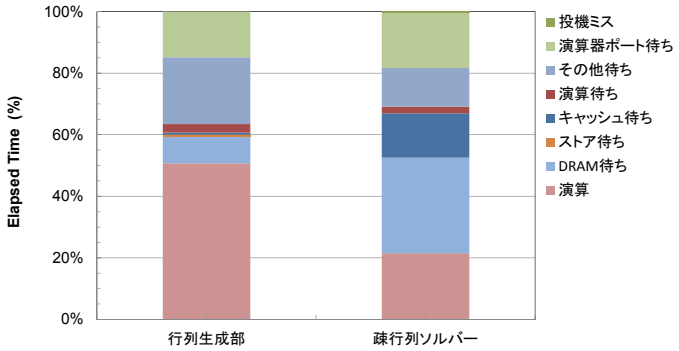


図 11　GeoFEM／Cube 計算結果，IvyB／Org.における行列生成部，疎行列ソルバーの計算時間の比率（2,048,383 要素，2,097,152 節点）

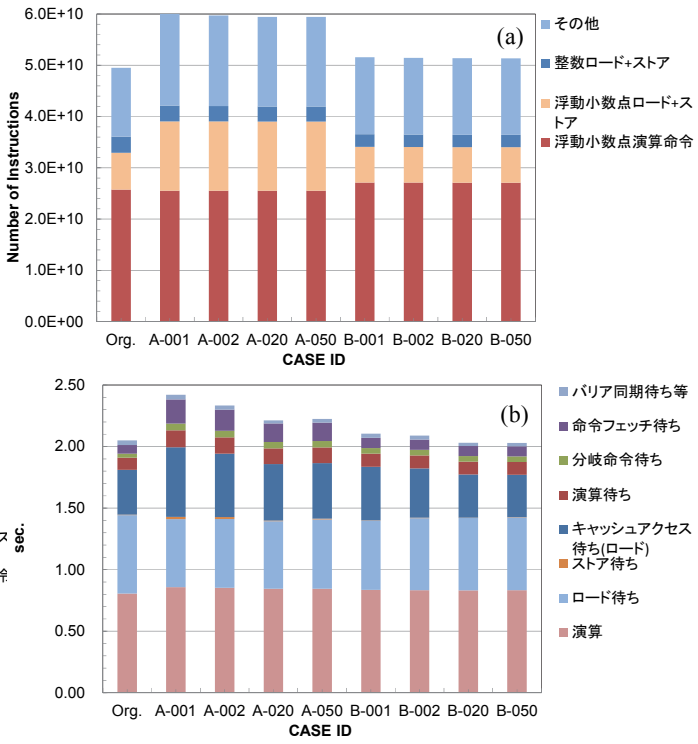


図 12　GeoFEM／Cube 計算結果，FX10 における行列生成部の専用詳細プロファイラによる解析結果（2,048,383 要素，2,097,152 節点）（a）命令数，（b）計算時間

疎行列ソルバーは memory-bound なプロセスとして知られているが，これらの結果より，行列生成部はそれと比較すると compute-bound であるということができる．専用詳細プロファイラの解析結果によると，FX10／Org.における対ピーク性能比は行列生成部が 13.3%であるのに対して，疎行列ソルバーは 4.83%である．表 3 によると IvyB の方が FX10 と比較してメモリバンド幅を効率的に使用できていることがわかる．このことが図 8 に示す計算性能にも影響していると考えられる．また，IvyB は Type-A が Org.，Type-B と比較して 40%程度高いメモリスループットを達成している．これは図 7，図 8 において，IvyB／Type-A，MIC／Type-A が高い性能を示していることと関連していると考えられる．

図 12 (a, b) は FX10 で実施した各ケースにおける行列生成部の専用詳細プロファイラによる解析結果である．Type-A では命令数がオリジナル実装，Type-B と比較して命令数が多く，計算時間も長く，また表 3 に示したメモリスループットの低さとも関連しているものと考えられる。既に述べたように，Type-A の計算性能は比較的低いものの，最適なケース (A-020) ではオリジナル実装と比較して 92%程度の性能は達成していることは図 12 (b) からも明らかである．FX10／A-020 では命令数も他の Type-A のケースと比較してわずかながら低下している（図 12 (a)）．

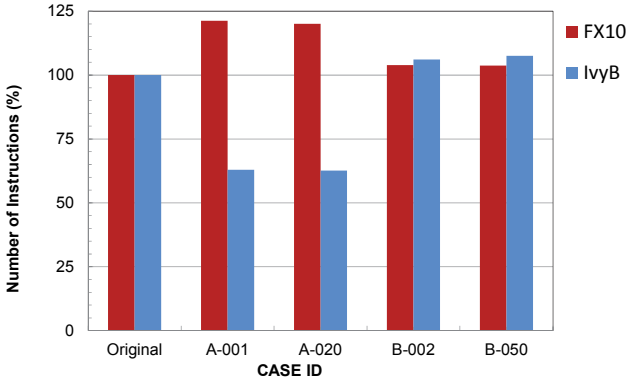


図 13 GeoFEM／Cube 計算結果，プロファイラ（Fujitsu PRIMEHPC FX10 専用詳細プロファイラ，Intel VTune）により求めた係数行列生成部総命令数（FX10，IvyB 共にオリジナル実装の場合 (Org.) の命令数で無次元化）(2,048,383 要素，2,097,152 節点），計算機環境，実装方法，要素ブロックサイズの効果

図 13 は FX10 と IvyB においてプロファイラ（Fujitsu PRIMEHPC FX10 専用詳細プロファイラ，Intel VTune）を使用して算出行列生成部の総命令数である．それぞれオリジナル実装（Org.）における命令数で無次元化してある．IvyB／Type-A では命令数がオリジナル実装の 60%に減少していることがわかる．一般に，命令数が少ない方が計算時間は短く，計算性能が高いと考えることができ，図 13 の結果は図 8 に示す各ケースの計算性能，表 3 に示すメモリスループットに対応しているものと考えられる．FX10 では Type-A の計算性能が悪化するのに対して，IvyB，MIC では性能が大幅に向上していることとも関連していると考えられる．

5. まとめ

本研究では，並列有限要素法による三次元弾性静解析アプリケーションに基づく性能評価用ベンチマーク GeoFEM／Cube において，OpenMP によってマルチスレッド並列化された係数行列生成部に関する 3 種類の実装によるプログラムを Fujitsu SPARC64 IXfx（FX10），Intel Xeon Phi 5110P（MIC），Intel Xeon E5-2680 v2 (IvyB)の 3 種類の環境を使って評価し，プロファイラを使用して性能の解析を実施した．

オリジナル実装は複雑な処理を 1 つのループで処理していたが，これを 3 つに分割した Type-A の実装は，要素行列生成以外のループをライブラリ化することが可能であり，アプリケーション開発者にとって最も負担が少ないアプローチである．

Type-A は MIC，IvyB で高い性能を示し，オリジナル実装と比較してそれぞれ 35%，60%程度の性能向上が得られた．IvyB の実行結果を Intel VTune によって分析した結果，総命令数がオリジナル実装の 60%程度に低下していることと関連することがわかった．逆に FX10 では総命令数が 20%程度増加し，計算時間も 10%～15%程度増加している．しかしながら要素ブロックサイズを 20 とした場合には計算時間増加を 8%程度に抑制することができ，命令数もわずかながら減少している．

これら，Type-A におけるアーキテクチャによる総命令数，性能の傾向の差異の原因については今後更に詳細に検討をしていく必要がある。MIC，IvyB において Type-A による性能の向上が顕著であった点については，複雑な処理を多数含むループをやや単純な構造を持つ 3 つの小ループに分割した効果もあると考えられる．

MIC，IvyB では A-001，A-002 等，ブロックサイズが少ない場合に最適値を達成し，ブロックサイズの増加によって性能が低下する．FX10 の場合もブロックサイズが 50 を超えた場合の性能低下は顕著であるが，最適値が A-020 で得られているように，MIC，IvyB とは挙動が異なっている．

Type-A は，疎行列の処理に関して適切なライブラリ化が実施されれば，アプリケーション開発者の負担が最も少ない実装である．本研究において，少なくとも MIC，IvyB については Type-A が適していることが示された．FX10 については，他と比較してやや性能が劣るものの，要素ブロックサイズが適切な値となるように制御すれば，性能向上が期待できる．現状の結果に基づくと，今後の ppOpen-HPC におけるアプローチとしては，基本的に Type-A に基づい

て実装し，状況に応じて自動チューニングによって要素ブロックサイズの最適値を決定するのが適切と考えられる．

今回は，問題サイズも一種類で，対象とするアーキテクチャも限定されていたため，今後はより広範囲の問題に対して適用することにより，ppOpen-HPC への適用について検討を継続する予定である．

参考文献

1) 中島研吾, 前処理付きマルチスレッド並列疎行列ソルバー, 情報処理学会研究報告(HPC-139-6)（2013）.
2) 大島聡史, 林雅江, 片桐孝洋, 中島研吾, 三次元有限要素法アプリケーションにおける行列生成処理の CUDA 向け実装, 情報処理学会 研究報告（HPC-130-11）（2011）.
3) ppOpen-HPC：科学技術振興機構戦略的創造研究推進事業（CREST)「ポストペタスケール高性能計算に資するシステムソフトウェア技術の創出：自動チューニング機構を有するアプリケーション開発・実行環境」, http://ppopenhpc.cc.u-tokyo.ac.jp/
4) 中島研吾，佐藤正樹，古村孝志，奥田洋司，岩下武史，阪口秀，自動チューニング機構を有するアプリケーション開発・実行環境 ppOpen-HPC，情報処理学会研究報告（HPC-130-44）（2011）.
5) GeoFEM：並列有限要素法による固体地球シミュレーションプラットフォーム，http://geofem.tokyo.rist.or.jp
6) 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)
7) 中島研吾，片桐孝洋，マルチコアプロセッサにおけるリオーダリング付き非構造格子向け前処理付反復法の性能，情報処理学会研究報告（HPC-120-6）（2009）.
8) 東京大学情報基盤センター（スーパーコンピューティング部門），http://www.cc.u-tokyo.ac.jp/
9) Intel VTune, https://software.intel.com/en-us/intel-vtune-amplifier-xe