Vol.2018-HPC-163 No.28
2018/3/2

有限要素法における係数行列生成部のマルチコア・メニィコア向け最適化

中島研吾[†1†2] 星野哲也[†1†2] 成瀬 彰[†3] 塙 敏博[†1] 三木洋平[†1]

有限要素法は偏微分方程式の数値解法として広く計算科学・工学分野で使用されている．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生成し，それを重ね合わせて得られる疎な全体行列に境界条件を適用し，全体行列を係数行列とする連立一次方程式を解いて解を得る．要素行列・全体行列を生成する係数行列生成部は連立一次方程式求解と並んで時間を要するプロセスである．本研究では，Intel Xeon（Broadwell），Intel Xeon Phi（Knights Landing）および NVIDIA Tesla P100（Pascal）及び V100（Volta）を対象としてそれぞれの特性を生かした最適化を実施した．本稿では最適化の詳細と性能評価結果について述べる．

Optimization of generation process for sparse coefficient matrices in FEM on multicore/manycore architectures

Kengo Nakajima[†1†2] Tetsuya Hoshino[†1†2] Akira Naruse[†3]
Toshihiro Hanawa[†1] Yohei Miki[†1]

Finite Element Method (FEM) is widely used for solving Partial Differential Equations (PDE) in various types of applications of computational science and engineering. In FEM, dense element matrix is introduced based on integral equations for each element, and sparse global matrix is assembled from element matrices. Boundary conditions are applied to this global matrix, and derived linear equations are solved. This process for generation of element and global matrices and the sparse matrix solver are the most expensive procedures in FEM procedures. In the present work, the matrix assembly process is optimized on Intel Xeon Phi (Broadwell), Intel Xeon Phi (Knights Landing) and NVIDIA Tesla P100 (Pascal) and V100 (Volta) based on features of each architecture. The paper describes details of optimization and results of performance evaluation.

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 Assembly）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．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．

近年，係数行列生成部の重要性が注目されつつあり，著者等も GPU，マルチコアプロセッサにおける有限要素法の行列生成プロセスの最適化に関する研究を実施している〔2,3〕．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」〔4,5〕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．本研究及び先行研究〔2,3〕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/FVM〔6〕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．

本研究では，著者等の先行研究〔3〕に基づき，最新のマルチコア，メニィコアアーキテクチャである下記環境における性能評価を実施した：①Intel Xeon Phi（Knights Landing）（KNL）〔7〕，②Intel Xeon（Broadwell-EP）（BDW）〔7〕，③NVIDIA Tesla P100（Pascal）（P100）〔8〕，④NVIDIA Tesla V100（Volta）（V100）〔8〕．

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する．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

†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
†3 エヌビディア
NVIDIA Corporation

ⓒ2018 Information Processing Society of Japan

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. 係数行列生成部の概要

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〔9,10〕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．本研究では各六面体要素を 6 個の三次元一次四面体要素に分割した場合の計算も実施した．プログラムは全て OpenMP ディレクティヴを含む Fortran 90 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〔10〕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．

表 1　問題サイズ

	N_x, N_y, N_z	節点数	六面体	四面体
M	128, 128, 128	2,097,152	2,048,383	8,193,532
L	200, 200, 128	5,120,000	5,029,327	20,117,308
S	80, 80, 100	640,000	617,859	2,471,436

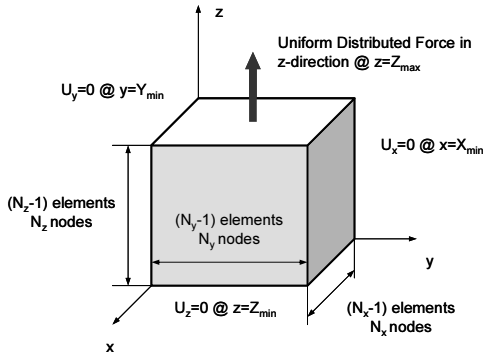

図 1　GeoFEM／Cube の解析対象（Cube モデル）

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〔2,3〕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．本研究では，表 1 に示す各問題サイズについて計算を実施した．

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．各節点における自由度数が 1 であれば，各要素の節点数に応じて，六面体要素であれば 8×8，四面体要素であれば 4×4 の密行列となる．本研究では 1 節点あたり 3 自由度のため，それぞれ 24×24，12×12 となる．

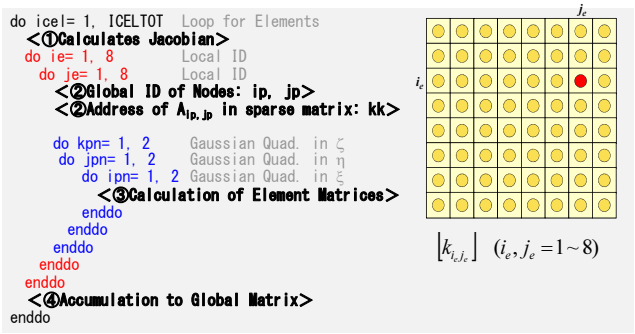

図 2　GeoFEM／Cube における係数行列生成の処理（六面体要素）

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リング（multicoloring）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〔2,3〕．図 2 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．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．四面体要素の場合は，節点数は 4 であるため，図 2 に示す二重ループは（do ie=1, 4, do je= 1, 4）となる．また，要素内の積分は解析的に実施できるため，要素当りの計算量は六面体要素と比較して少ない．

2.3 係数行列生成部の実装手法

図 2 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

ⓒ2018 Information Processing Society of Japan

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

```
!$omp parallel (…)
    do color= 1, COLORtot
!$omp do
    do icel= col_index(icol-1)+1, col_index(icol)
        icel: calculated by (col_index, ib, blk)
      <①Jacobian & Grad. Shape Fn' s>
!$omp simd
      do ie= 1, 8; do je= 1, 8
        <② Address in Sparse Matrix >
        <③ Element Matrices>
        <④ Global Matrix>
      enddo; enddo
    enddo
    enddo
!$omp end parallel
```

図 3　GeoFEM／Cube オリジナル実装（Original）の概要（六面体要素）（COLORtot：要素色数（=8），col_index(color)：各色に含まれる要素数）

図 3 は，図 2 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場合の六面体要素の場合の実装例の概略である．COLORtot はマルチカラーリングの色数であり，六面体要素の場合には 8，四面体要素を適用した場合には 33～34 程度となる．1 つの節点を共有する要素数は六面体要素の場合，ほとんどの節点で 8 となるが，四面体要素の場合は 24 となる．配列 col_index(icol)は各色に含まれる要素数である．図 3 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．

係数行列生成部は連立一次方程式を解く線形ソルバ部と同様に memory bound なプロセスであるが，要素単位のローカルな計算が中心であり計算性能がでやすい傾向がある〔1,2,3〕．

3. 計算機環境

本研究では以下の 4 種類の計算機環境を使用した：

- KNL: Intel Xeon Phi（Knights Landing）〔7〕
- BDW: Intel Xeon（Broadwell-EP）〔7〕
- P100: NVIDIA Tesla P100（Pascal）〔8〕
- V100: NVIDIA Tesla V100（Volta）〔8〕

表 2 に計算機環境，使用ソフトウェアの概要を示す．KNL は最先端共同 HPC 基盤施設（JCAHPC）の Oakforest-PACS（OFP），BDW，P100 は東京大学基盤センターの Reedbush-U，Reedbush-H〔12〕，V100 は IBM Power9〔13〕をホストとするサーバーを使用した．KNL は 1 ノード（68 コア，272 スレッド），BDW は 1 ノード（2 ソケット，36 コア）を使用した．P100，V100 はそれぞれ各計算ノードに 2GPU，4GPU を搭載しているがそのうち 1 基を使用した．

KNL では従来の DDR4 に加えて，Multi-Channel DRAM（MCDRAM）と呼ばれる 3D 積層による高帯域のオンパッケージ・メモリーが搭載されている．本研究では Flat モード〔7,11〕を採用し，主としてこの MCDRAM のみ使用時の性能を評価した．参考のため DDR4 のみ使用時の計算も実施している．

表 2　各計算環境の概要

略　称	KNL	BDW	P100	V100
システム名	Oakforest-PACS〔11〕	Reedbush-U〔12〕	Reedbush-H〔12〕	
アーキテクチャ	Intel Xeon Phi 7250 (Knights Landing)	Intel Xeon E5-2695 v4 (Broadwell-EP)	NVIDIA Tesla P100 (Pascal)	NVIDIA Tesla V100 (Volta)
動作周波数（GHz）	1.40	2.10	1.328	1.455
コア数	68	36（18×2）	3,584	5,120
使用スレッド数	272	36	-	
理論演算性能（GFLOPS）	3,046	1,210	5,300	7,500
主記憶容量（GB）	MCDRAM: 16 DDR4: 96	256	16	16
STREAM Triad 性能（GB/sec.）〔14〕	MCDRAM: 490 DDR4: 80.1	131	530	795
コンパイラ	Intel Parallel Studio XE 2018		PGI 17.10, CUDA 8.0.44	PGI 17.10, CUDA 9.1

4. KNL 及び BDW における最適化

4.1 最適化の概要

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

ⓒ2018 Information Processing Society of Japan

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 4.61 KByte（=24×24×8））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．

```
!$omp parallel (…)
    do color= 1, COLORtot
!$omp do
    do ip= 1, THREAD_total
      NBLK: calculated by (col_index, color, thread#)
    do ib= 1, NBLK
      do blk= 1, BLKSIZ
        icel: calculated by (col_index, ib, blk)
!$omp simd
        do je= 1, 8; do je= 1, 8
          <② Address in Sparse Matrix>
        enddo; enddo
      enddo
      do blk= 1, BLKSIZ
        icel: calculated by (col_index, ib, blk)
        <① Jacobian & Grad. Shape Fn' s>
!$omp simd
        do je= 1, 8; do je= 1, 8
          <③ Element Matrices >
        enddo; enddo
      enddo
      do blk= 1, BLKSIZ
        icel: calculated by (col_index, ib, blk)
!$omp simd
        do je= 1, 8; do je= 1, 8
          <④ Global Matrices>
        enddo; enddo
      enddo
    enddo
    enddo
    enddo
!$omp end parallel
```

図 4　Type-A 実装の概要（六面体要素）（COLORtot：要素色数（=8），THREAD_NUM：スレッド数（=240），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）

```
!$omp parallel (…)
    do color= 1, COLORtot
!$omp do
    do ip= 1, THREAD_total
      NBLK: calculated by (col_index, color, thread#)
    do ib= 1, NBLK
      do blk= 1, BLKSIZ
        icel: calculated by (col_index, ib, blk)
!$omp simd
        do je= 1, 8; do je= 1, 8
          <② Address in Sparse Matrix>
        enddo; enddo
      enddo
      do blk= 1, BLKSIZ
        <① Jacobian & Grad. Shape Fn' s>
        icel: calculated by (col_index, ib, blk)
!$omp simd
        do je= 1, 8; do je= 1, 8
          <③ Element Matrices>
          <④ Global Matrices>
        enddo; enddo
      enddo
    enddo
    enddo
    enddo
!$omp end parallel
```

図 5　Type-B 実装の概要（六面体要素）（COLORtot：要素色数（=8），col_index(color)：各色に含まれる要素数，THREAD_NUM：スレッド数（=240），NBLK：要素ブロック総数，BLKSIZ：要素ブロックサイズ，icel：要素番号）

以上を考慮して，図 4，図 5 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 は各色，各スレッド内の要素ブロックの総数である．

Type-A（図 4）

- 図 2 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 5）

- 図 2 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容量が必要である．要素行列用配列の記憶は不要である．

両者のうち，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可能である．各節点に関するループは!$omp simd ディレクティヴ〔15〕を使って明示的にベクトル化，SIMD 化を実施している．BDW，KNL は SIMD 幅がそれぞれ 256bit，512bit であるため 64bit の倍精度実数におけるベクトル長はそれぞれ 4,8 となる〔7〕．

4.2 計算ケース

表 3 に計算ケースを示す．ここでは，①要素（六面体，四面体），②実装タイプ（オリジナル，Type-A，Type-B），③!$omp simd の有無，を考慮して計算ケースを設定している．Hex/Tet-A1, A2, B1, B2 では図 4～5 に示す要素ブロックサイズをパラメータとして計算を実施した．

表 3　計算ケース

ケース名	要素タイプ	実装タイプ	SIMD 指示行
Hex-O1	六面体	オリジナル（図 3）	無し
Hex-O2			有り
Hex-A1		Type-A（図 4）	無し
Hex-A2			有り
Hex-B1		Type-B（図 5）	無し
Hex-B2			有り
Tet-O1	四面体	オリジナル（図 3）	無し
Tet-O2			有り
Tet-A1		Type-A（図 4）	無し
Tet-A2			有り
Tet-B1		Type-B（図 5）	無し
Tet-B2			有り

4.3 計算結果

図 6～図 8 は問題サイズ M（表 1）について，六面体要素，四面体要素について，要素ブロックサイズをパラメータとした場合の計算結果（計算時間）である．KNL では MCDRAM のみ使用している．図 8 は，オリジナル実装

ⓒ2018 Information Processing Society of Japan

（Hex-O1，Tet-O1）に対する速度向上率を示す．Type-A，Type-B については最短時間を達成したブロックサイズにおける値を代表値とする．六面体要素については，先行研究〔2,3〕でも示した通り，`!$omp simd` を挿入しない場合は，オリジナル実装（Hex-O1）と Type-A（Hex-A1），Type-B（Hex-B1）の性能は KNL，BDW ともにほぼ同じである．Type-A（Hex-A1）についてはブロックサイズ増加とともに性能が低下するが，Type-B（Hex-B1）についてはほぼ一定である．`!$omp simd` 挿入により KNL ではオリジナル実装（Hex-O1）と比較して最大 2 倍程度，BDW では最大 20% 程度速度向上が見られる（図 6，図 8）．KNL では Type-B（Hex-B2），BDW では Type-A（Hex-A2）が良い性能を示している．ブロックサイズの性能に対する影響は Hex-A1，Hex-B1 の場合とほぼ同じである．BDW の速度向上率が低いのはメモリバンド幅の差に起因するものと考えられる．四面体については，各実装法の差は少なく，`!$omp simd` 挿入の効果も 10%程度である．四面体要素はループ長が 4 であり，特に KNL の場合に性能向上に十分なベクトル長が得られていないためと考えられる．但し，この傾向は長さ 4 の二重ループを統合してループ長 16 とした場合も同様であった．またオリジナル実装（Tet-O1）と比較して，Tet-O2 は KNL，BDW いずれも大幅に性能が低下している．KNL と BDW の性能比は六面体，四面体ともに 2.30:1.00 程度で，メモリバンド幅よりは演算性能比に近い（表 2）．

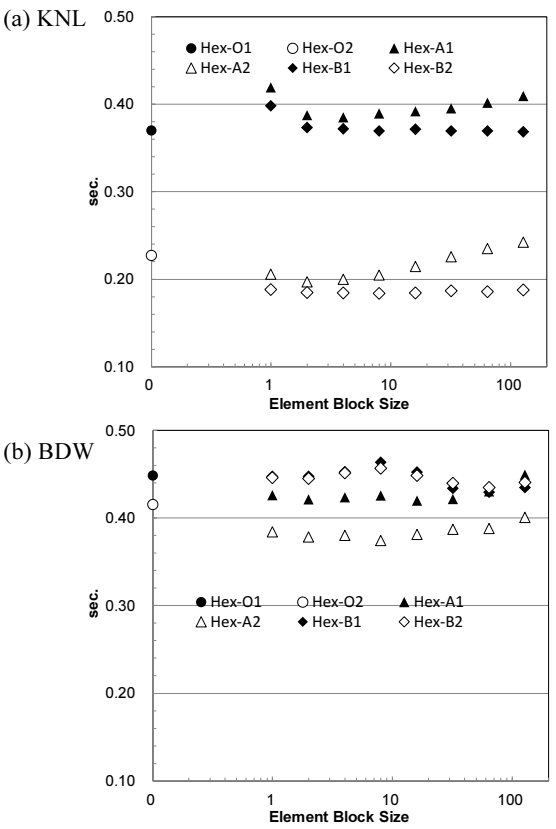

図 6　GeoFEM／Cube 計算結果，係数行列生成部計算時間（六面体要素）問題サイズ M，KNL：MCDRAM のみ使用

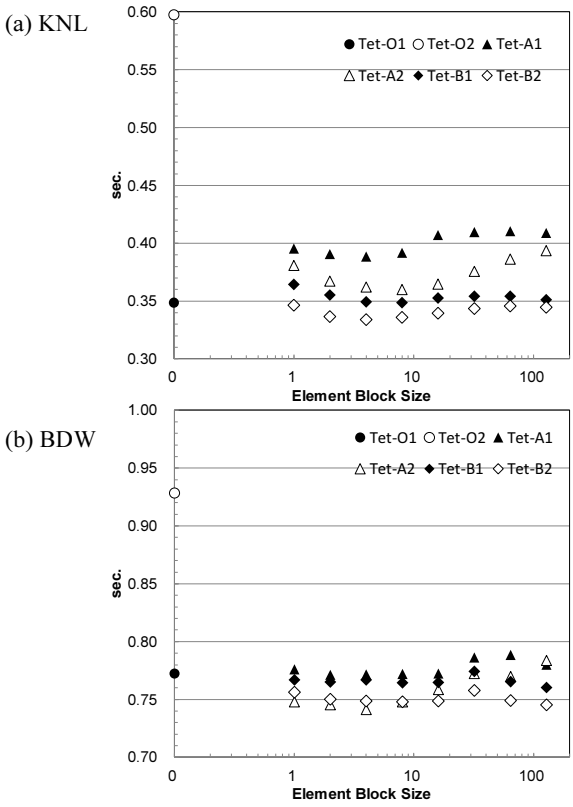

図 7　GeoFEM／Cube 計算結果，係数行列生成部計算時間（四面体要素）問題サイズ M，　KNL：MCDRAM のみ使用

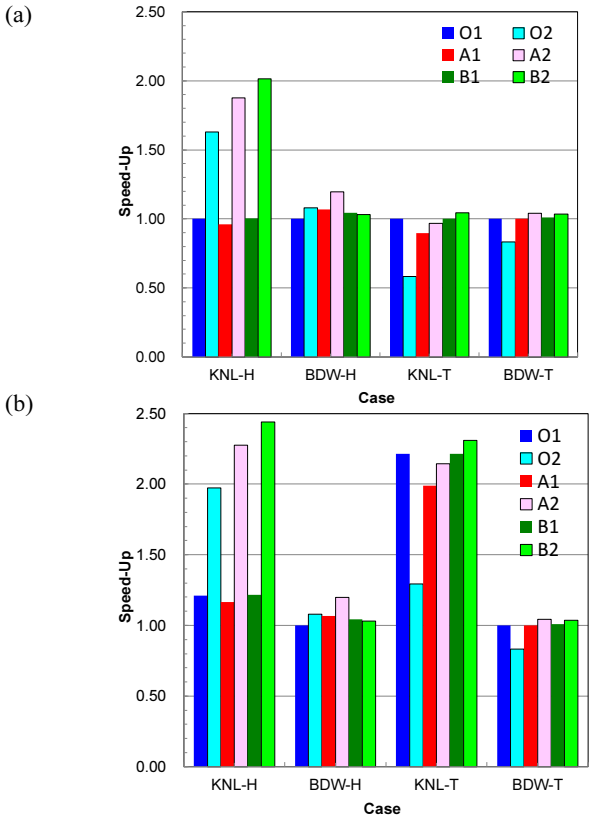

図 8　GeoFEM／Cube 計算結果，係数行列生成部計算時間，問題サイズ M，オリジナル実装（Hex-O1，Tet-O1）に対する速度向上率，-H：六面体，-T：四面体，（a）O1 の計算時間を基準，(b) BDW における Hex/Tet-O1 の時間を基準，KNL：MCDRAM のみ使用

ⓒ2018 Information Processing Society of Japan

図 9 は，問題サイズ L,S の場合の速度向上率であるが，基本的な傾向は問題サイズ M の場合と変わらず，ブロックサイズの影響についても同様である．問題サイズ S の場合は，KNL の速度向上率が問題サイズ L，M の場合と比較してやや低い．

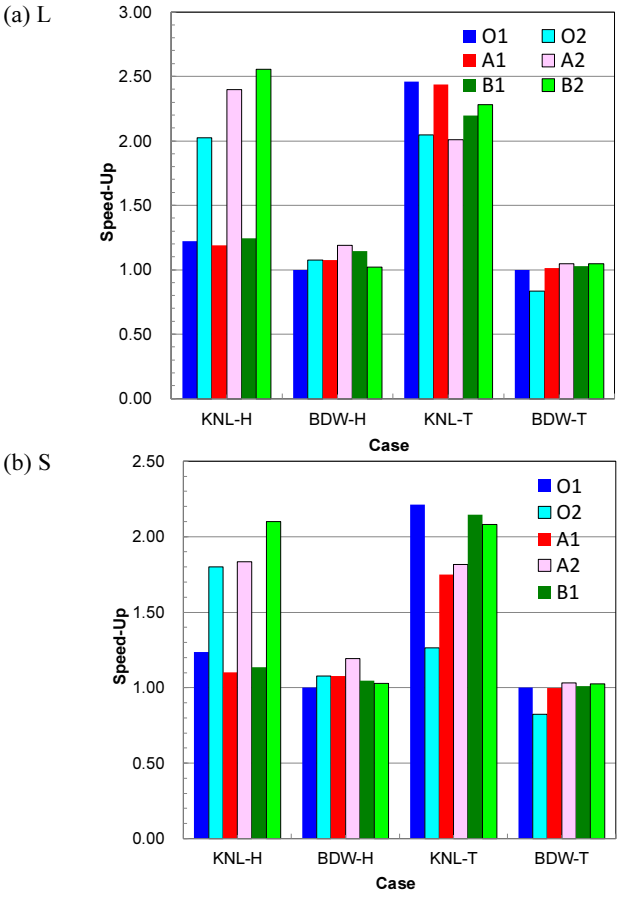

図 9　GeoFEM／Cube 計算結果，係数行列生成部計算時間，問題サイズ L・S，オリジナル実装（Hex-O1，Tet-O1）に対する速度向上率，-H：六面体，-T：四面体，BDW における Hex/Tet-O1 の時間を基準，KNL：MCDRAM のみ使用

図 10（a,b）は，KNL において，DDR4 を使用した場合の結果である．`!$omp simd` 挿入の効果は六面体の場合で約 10%の性能向上であり，四面体の場合にはほとんど無い．これは BDW と同様の傾向で，メモリバンド幅の制約によるものと考えられる．また六面体，四面体ともに Type-A でブロックサイズを増加させた場合の性能低下が著しい．図 10（c）は MCDRAM 利用時のオリジナル実装（Hex/Tet-O1）の計算性能を基準とした性能比である．MCDRAM 利用時と DDR4 利用時の性能比は，それぞれの最適値を基準とすると六面体，四面体ともに 3:1 程度である．

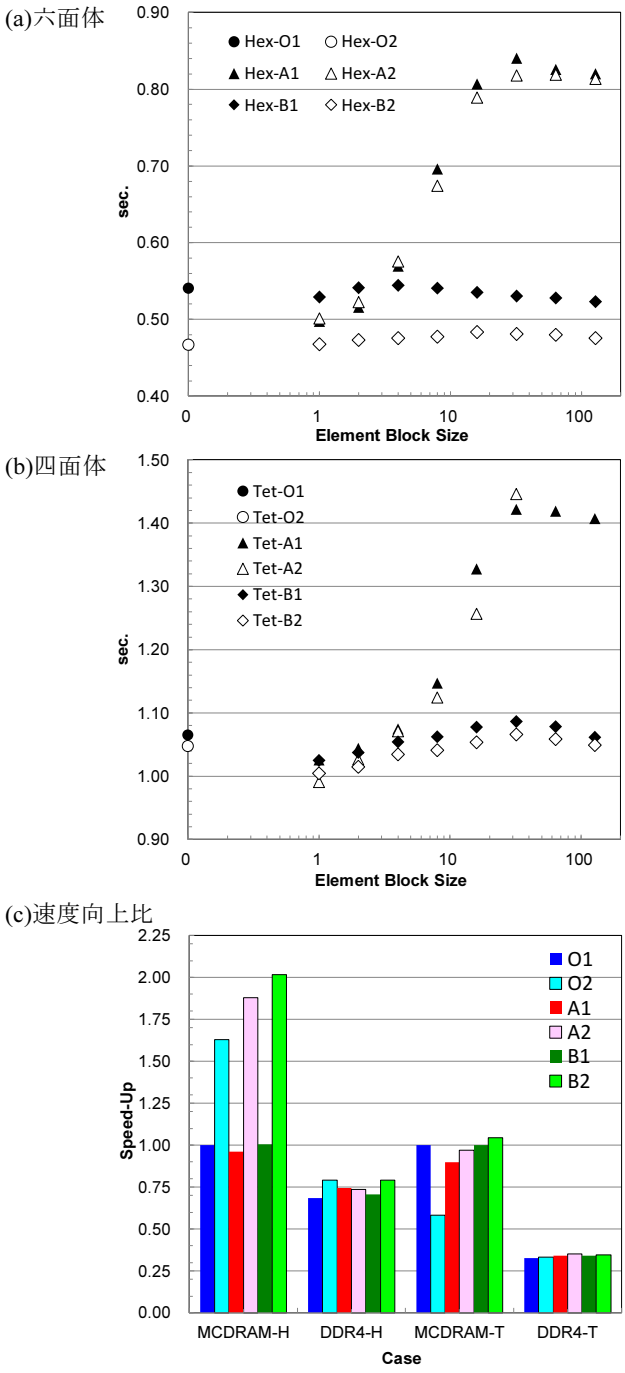

図 10　KNL による GeoFEM／Cube 計算結果，係数行列生成部計算時間，DDR4 使用時，問題サイズ M（a）六面体，（b）四面体，（c）オリジナル実装（Hex-O1，Tet-O1）に対する速度向上率，-H：六面体，-T：四面体，MCDRAM 利用時 Hex/Tet-O1 の時間を基準

5. P100，V100 における性能評価

5.1 概 要

P100，V100 についても，著者等による先行研究〔3〕で使用したのと同じ NVIDIA Tesla K40 向けに最適化されたプログラムを使用した．OpenACC 実装例（六面体）を図 11 に示す．ブロックサイズ（`BLKSIZ`）は 32,000 としている．図 11 は図 3 のオリジナル実装によっている．実装手順

ⓒ2018 Information Processing Society of Japan

を以下に示す：

1. （オリジナル実装から）マルチカラーリングの取り外し，
2. 処理①～④の分離と一時配列の追加，
3. ブロッキングの適用
4. OpenACC ディレクティヴの追加

単純にマルチカラーリングを外すと正しい結果が得られないため，全体行列の加算部分には Atomic 操作を使って同時書き込みを回避している．OpenACC では atomic クローズを使って Atomic 操作が必要な箇所を簡単に指定することができる．GPU でもマルチカラーリングを使って全体行列への同時書き込みを回避する方法も有効であるため，オリジナル実装同様にマルチカラーリングを適用した手法（カラーリング版（coloring））と比較する．

CUDA に関する最適化については著者等による先行研究〔3〕を参考にされたい．

```
   do ib= 1, NBLK
!$acc parallel
!$acc loop gang vector
      do blk= 1, BLKSIZ
         icel = blk + BLKSIZ * (ib-1)
         ＜①各積分点におけるヤコビアン，形状関数導関数計算＞
      enddo
!$acc end parallel
!$acc parallel
!$acc loop gang
      do blk= 1, BLKSIZ
         icel = blk + BLKSIZ * (ib-1)
!$acc loop collapse(2) vector
         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＋格納＞
         enddo; enddo
      enddo
!$acc end parallel
!$acc parallel
!$acc loop gang
      do blk= 1, BLKSIZ
         icel = blk + BLKSIZ * (ib-1)
!$acc loop collapse(2) vector
         do ie= 1, 8; do je= 1, 8
            ＜③ガウス数値積分，要素行列成分計算＞
         enddo; enddo
      enddo
!$acc end parallel
!$acc parallel
!$acc loop gang
      do blk= 1, BLKSIZ
         icel = blk + BLKSIZ * (ib-1)
!$acc loop collapse(3) vector
         do ie= 1, 8; do je= 1, 8 ; do k= 1, 9
            ＜④要素行列成分の全体行列への加算＞
         enddo; enddo; enddo
      enddo
!$acc end parallel
   enddo
```

図 11　GPU 実装の概要（六面体要素，OpenACC）（NBLK：要素ブロック総数，BLKSIZ：要素ブロックサイズ）〔2〕

5.2 性能測定と考察

表 4 は，六面体，問題サイズ M について各アーキテクチャで計算性能を比較したものである．上段は計算時間，下段は，先行研究〔2〕での性能からの推定値である．

- KNL・BDW：!$omp simd 挿入時／非挿入時における最適値，KNL は MCDRAM のみ使用
- P100・V100：Coloring（OpenACC），Atomic（OpenACC），Atomic（CUDA）の結果

である．対ピーク性能比では：

- KNL：11.63%
- BDW：14.60%
- P100：7.52%-11.70%
- V100：9.04%-16.87%

である．カラーリング版では，KNL と P100 は，ほぼ同じ計算性能である．先行研究〔3〕と同様に，P100，V100 では Atomic を適用したバージョンの方が全般的に高性能である．

表 4　GeoFEM／Cube 計算結果（上段：係数行列生成部計算時間（sec），下段：GFLOPS 値），六面体　問題サイズ M

略　称		Coloring OpenMP OpenACC	Atomic OpenACC	Atomic CUDA
KNL (MCDRAM)	w/o SIMD	0.3683 176.6		
	w SIMD	0.1836 354.2		
BDW	w/o SIMD	0.4195 155.0		
	w SIMD	0.3683 176.6		
P100	-	0.1632 398.5	0.1342 484.6	0.1049 619.9
V100	-	0.09596 677.7	0.07260 895.75	0.05139 1265.4

先行研究〔3〕によれば，カラーリング版の総メモリアクセス量は Atomic 版の 1.4 倍である．カラーリング版では，全体行列への加算が同じ箇所に行われることがないため，キャッシュは効かない．Atomic 版では全体行列への加算が同じ箇所に行われることがあり，キャッシュが効いてメモリアクセス量を削減できる可能性がある（図 12）．

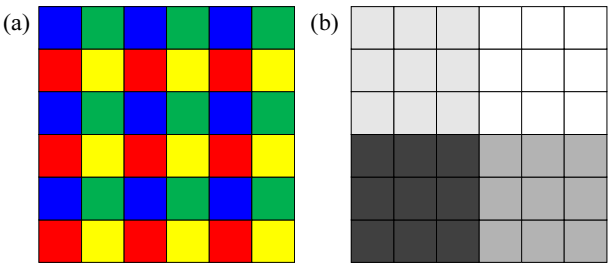

図 12　カラーリング版・Atomic 版の比較，(a) カラーリング版（4 色の例）：同じ色の要素を同時処理，キャッシュが効かない：同じ色の処理中には 1 節点に一回しか触らない（b）Atomic 版（4 スレッドの例）：辞書式番号付けでキャッシュが効きやすい

ⓒ2018 Information Processing Society of Japan

6. 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を元に，著者等による先行研究〔3〕において開発されたプログラムを KNL，BDW，P100，V100 等の最新のアーキテクチャに適用し，性能を評価した．

得られた結果の傾向は先行研究〔3〕と同様であった。Intel Xeon Phi（Knights Landing，KNL）においては`!$omp simd`挿入の効果が大きく，オリジナル実装の 2.50 倍程度の速度向上が得られたが，Intel Xeon（Broadwell-EP，BDW）ではメモリバンド幅の制約のため，最大 20%程度に留まった．また，最適な実装は KNL（Type-B），BDW（Type-A）で異なっている．P100，V100 では Atomic の効果が顕著であった．

KNL では高速，小容量の MCDRAM，低速，大容量の DDR4 の 2 種類のメモリを利用可能であるが，DDR4 のみ使用時の性能は MCDRAM のみの場合より性能が低く，`!$omp simd` 挿入によるベクトル化の効果も小さい．問題規模が大きく，MCDRAM のみではメモリ容量が足りない場合の，MCDRAM の有効な活用が今後の重要な技術的課題である．

また，P100，V100 については，先行研究〔3〕において NVIDIA Tesla K40 に対して最適化されたプログラムを適用したため，P100，V100 向けの最適化を適用する必要がある．

謝 辞

本研究の実施にあたり，NVIDIA Tesla V100 環境の提供，データ測定にご協力いただいた笠毛知徳氏，平尾義郎氏，亀ノ上剛氏（日本 IBM）に感謝いたします．

参考文献

1) 中島研吾，大島聡史，塙敏博，星野哲也，伊田明弘，ICCG 法ソルバの Intel Xeon Phi 向け最適化，情報処理学会研究報告（2016-HPC-157-16）（2016）
2) 中島研吾，大島聡史，塙敏博，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，情報処理学会研究報告（HPC-146-22）（2014）
3) 中島研吾，成瀬彰，大島聡史，塙敏博，片桐孝洋，田浦健次朗，有限要素法係数行列生成プロセスのメニィコア環境における最適化，情報処理学会研究報告（HPC-152-12）（2015）
4) 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/
5) Nakajima, K., Satoh, M., Furumura, T., Okuda, H., Iwashita, T., Sakaguchi, H., Katagiri, T., Matsumoto, M., Ohshima, M., Jitsumoto, H., Arakawa, T., Mori, F., Kitayama, T., Ida, A., and Matsuo, M.Y., ppOpen-HPC: Open Source Infrastructure for Development and Execution of Large-Scale Scientific Applications on Post-Peta-Scale Supercomputers with Automatic Tuning (AT), Optimization in the Real World –Towards Solving Real-Worlds Optimization Problems, Mathematics for Industry 13, 15-35, Springer (2015)
6) 中島研吾，片桐孝洋，大島聡史，塙敏博，ppOpen-APPL/FVM を使用した並列有限要素法アプリケーションの開発，情報処理学会研究報告（HPC-151-24）（2015）
7) Intel, http://www.intel.com/
8) NVIDIA, http://www.nvidia.com/
9) 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
10) 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)
11) 最先端共同 HPC 基盤施設，http://jcahpc.jp/
12) 東京大学情報基盤センター, http://www.cc.u-tokyo.ac.jp/
13) IBM, https://www.ibm.com/
14) STREAM, https://www.cs.virginia.edu/stream/
15) OpenMP, http://openmp.org/

ⓒ2018 Information Processing Society of Japan