社団法人 情報処理学会 研究報告
IPSJ SIG Technical Report

2009－ARC－182 (18)
2009－HPC－119 (18)
2009／2／27

マルチコアクラスタにおける有限要素法アプリケーションのための階層型領域間境界分割に基づく並列前処理手法

中島 研吾 †,††

本研究では，不均質場における静的弾性問題を対象とした並列有限要素法アプリケーションに対して，階層型領域間境界分割に基づく前処理付反復法ソルバー及び OpenMP と MPI の Hybrid 並列プログラミングモデルを適用し，T2K オープンスパコン（東大）512 コアにおいて Flat MPI との性能比較，最適化を実施した。First Touch Data Placement，データ再配置，「--localalloc」を含む NUMA control の組み合わせにより，Hybrid 並列プログラミングモデルが Flat MPI と同等かそれを上回る性能を得られることがわかった。

Parallel Preconditioners based on a Hierarchical Interface Decomposition for Finite-Element Applications on Multicore Clusters

KENGO NAKAJIMA †,††

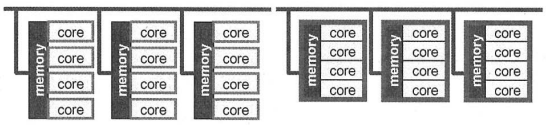
In this study, "HID (Parallel Hierarchical Interface Decomposition)" and a hybrid parallel programming model have been implemented to a finite-element application for linear-elastic simulations with heterogeneous material property using parallel preconditioned iterative solvers. Developed code has been tested and optimized on T2K Open Supercomputers (Tokyo) with up to 512 cores. Combination of "First-Touch Data Placement", "reordering of data", and NUMA control with "--localalloc" improved performance of hybrid parallel programming models, and final performance of hybrid is competitive with or rather better than that of Flat MPI programming model.

1. はじめに

近年，マルチコアプロセッサの普及，大規模システムにおけるコア数の増加を背景として，ハイブリッド（Hybrid）並列プログラミングモデルが脚光を浴びるようになり，Flat MPI（または Pure MPI）との優劣に関する議論が盛んとなっている（Fig.1 参照）。Hybrid 並列プログラミングモデルはメッセージパッシングによる「coarse-grain parallelism」と，ディレクティブによる「fine-grain parallelism」の融合であり，一般的には MPI と OpenMP を組み合わせたスタイルである。

両者の優劣は，さまざまなハードウェア性能諸元（コアのピーク性能，通信バンド幅，メモリバンド幅等）とそのバランス，アプリケーションの特性，問題サイズに依存することはよく知られている。

著者らは，「地球シミュレータ」を中心とした SMP クラスタを対象として，並列有限要素法向けに最適化された前処理付き反復法による線形ソルバーを，Hybrid 並列プログラミングモデルを使用して開発した〔1〕。〔2〕では，階層型領域間境界分割（Hierarchical Interface Decomposition，HID）〔3〕に基づく並列前処理手法を使用した並列有限要素法アプリケーションに同様の最適化手法を適用し，T2K オープンスパコン（東大）（以下 T2K（東大））〔4〕上で Hybrid と Flat MPI の性能比較を実施している。SMP クラスタ上では両者の性能はほぼ同じであるが，ノード数が増加すると Hybrid が優位となる傾向がある。



(a) Flat MPI (b) Hybrid

Fig.1 Parallel programming models on multicore clusters [1,2]

NUMA（Non Uniform Memory Access）アーキテクチュアに基づくマルチコアクラスタ（T2K（東大））上では，その特性を利用するために，実行時制御コマンド（NUMA control）使用して，コア（またはソケット）とメモリの関係を明示的に指定することによって，特に Hybrid の性能が向上することが明らかとなった。

本研究では，〔2〕で扱った，不均質場における静的弾性問題を対象とした並列有限要素法アプリケーションについて，特に Hybrid 並列プログラミングモデルの更なる最適化を，T2K（東大）上で試みた。

以下の各章では，アプリケーション・計算手法，実行環境，計算結果，最適化について紹介する。

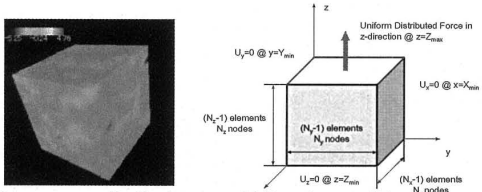
2. アプリケーション・計算手法の概要

(1) 不均質場における三次元弾性問題

本研究では，Fig.2 に示すような，不均質な物性値分布を有する立方体形状における三次元静的弾性問題を並列有限要素法（Finite-Element Method，FEM）によって解く。一次補間関数に基づく，六面体アイソパラメトリック要素を使用している。各要素は辺の長さ＝1 の立方体である。ポアソン比は全要素で 0.25 である。ヤング率は，地質統計学の分野で使用されている sequential Gaussian アルゴリズム〔5〕に基づき，発生

† 東京大学情報基盤センター
Information Technology Center, The University of Tokyo.
†† 科学技術振興機構 戦略的創造研究推進事業（CREST）
CREST, Japan Science and Technology Agency (JST)

させた値を使用した。ヤング率は位置座標の関数として求められ，メッシュごとに異なった値が与えられる。ヤング率の最大値と最小値はそれぞれ，10^3 及び 10^{-3} である（平均値を 1.0 とする）。〔2〕では GPBi-CG 法を採用したが，係数行列が対称正定となることから，SGS（Symmetric Gauss-Seidel）〔1,2〕を前処理手法とし共役勾配法（Conjugate Gradient，CG）法によって連立一次方程式を解いている（以下　SGS/CG 法と呼ぶ）。SGS 前処理では，係数行列 A そのものが前処理行列として利用されるため ILU 分解は実施しない。



(a) Heterogeneous distribution of material property　　(b) Boundary conditions

Fig.2 Simple cube geometries with heterogeneity for 3D linear-elastic problems

プログラムは GeoFEM〔6〕の並列 FEM の枠組みに基づいており，GeoFEM の局所データ構造を利用している。GeoFEM の局所データ構造は節点ベース，オーバーラップ要素付きの領域分割に基づいているが，本研究で使用する HID の階層的なデータ構造に適用可能なように修正されている〔2〕。

（2）階層型領域間境界分割（HID）

階層型領域間境界分割（HID）のプロセスは，節点（vertices）と辺（edges）から構成されるグラフ（graph）を互いに共通部分を持たない節点の集合（コネクタ，connector）に分割することから始まる。

この節点の集合のことを，〔3〕では「レベル 1 の connector（C^1）」と呼んでいる。それぞれのレベル 1 の connector が並列計算における各領域に対応するため，C^1 に属する各節点群を「sub-domain」と呼ぶこともある。残りの各節点に対してレベル（多重格子法のレベルと区別するため「HID レベル」と呼ぶ）を設定する。レベル k の connector（C^k）（k>1）は k 個の sub-domain と共有節点を持つ節点の集合である。また，それぞれの C^k は他のレベル k の connector とは共有節点を持たない。Fig.3 は二次元 9 点差分格子を 4 領域に分割する場合の例である。このケースでは 4 つのレベル 1 の connector（C^1）が存在し，各 sub-domain に対応している。更に，4 つのレベル 2 の connector（C^2），1 つのレベル 4 の connector（C^4）から構成される。同じ HID レベルにある異なった connector は直接に結合されることは無く，より HID レベルの高い connector によって隔てられている。このような connector の特性と並び替え（reordering）によって，全体マトリクス [A] は各 connector に対応したブロック的な構造を有するようになる。各節点が HID レベルの順番に番号付けされた場合，並び替えられた全体マトリクスのブロック構造は Fig.4 に示すようなものになる。同じレベルに属するコネクタ群同士は独立であるため，このようなブロック構造によって ILU／IC 前処理やガウス・ザイデル等の計算プロセスの並列化が容易に実現可能である。各レベルの計算終了後に通信を実施することにより，ブロックヤコビ型の局所並列前処理と比較して，より完全な前進後退代入を実現できる〔2,3〕。Fig.5 は T2K（東大）における，HID とブロックヤコビ型局所前処理の比較である〔2〕。両者はスケーラビリティの点では共に優れているが，Fig.5（a）は HID における線形ソルバーの相対性能を示す。値は 1 より大きく，概して HID の方が性能が良いことがわかる。

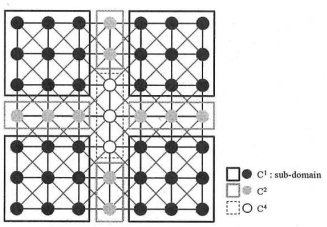
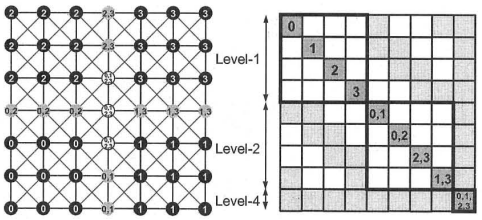


Fig.3 Partitioning of a 9-point grid into 4 sub-domains



(a) Domain decomposition (connectors and keys)　　(b) matrix and non-empty blocks

Fig.4 Domain/block decomposition of the matrix according to the HID reordering

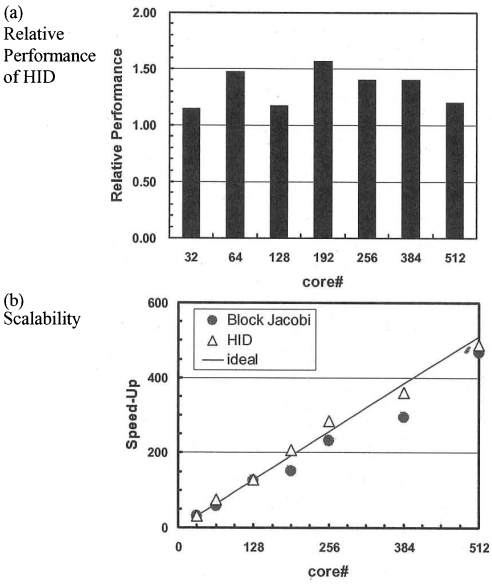


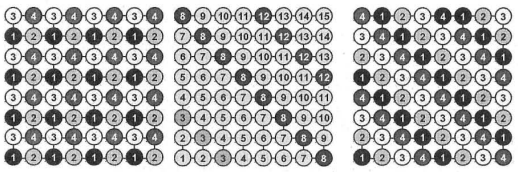
Fig.5 Performance of SGS/GPBi-CG solver on T2K (Tokyo) for 3D linear-elastic model with heterogeneous distribution of Young's modulus ($E=10^{-3}$~10^{+3}), 2,097,152 elements, 6,440,067 DOF, Flat MPI parallel programming model [2]

(3) 並び替え手法

Hybrid 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にを 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。SGS 前処理に基づく反復法では，節点の並び替えにより節点間のグローバルな依存性を排除することによって，並列化が実現される。著者らは主としてマルチカラー法（MC）を使用して並び替えを実施してきた（Fig.6(a)，数字は色番号）〔1〕。MC 法は高い並列性能とスレッド間の負荷分散を容易に達成可能であるが，悪条件問題で収束が悪化することが知られている。また，色数を増やすことによって収束を改善できるが，OpenMP のオーバーヘッドにより性能が低下する場合がある〔1〕。高い並列化効率を得るためには，できるだけ各色内の節点数が多い方が都合が良い。

レベルセットによる並べ替え法（level set reordering method）である Reverse Cuthill-McKee（RCM）法（Fig.6(b)，数字はレベル番号）は，悪条件問題に対する収束性は良いが，各レベルセットに含まれる節点数は不均一であり，並列性能は MC 法と比べると低い。

この問題を解決する手法として RCM 法によって並び替えを施された節点に対して，更にサイクリックに再番号付けする Cyclic マルチカラー法（cyclic multicoloring，CM）を適用する手法（CM-RCM）が考案されている〔2〕。Fig.6(c)は CM-RCM 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，たとえば，RCM の第 1，第 5，第 9，第 13 組の節点群が CM-RCM 法の第 1 色に分類されている。各色には 16 の節点が含まれている。CM-RCM 法における色数は，各色内の節点が依存性を持たない程度に充分大きい必要がある。



(a) MC: 4 colors　(b) RCM　(c) CM-RCM: 4 colors

Fig. 6 Example of MC, RCM, and CM-RCM coloring for reordering on 5-point grid, numbers indicate color/level ID [1,2]

```
      do lev= 1, LEVELtot
        do ic= 1, COLORtot(lev)
!$omp parallel do private(ip,i,SW1,SW2,SW3,isL,ieL,j,k,X1,X2,X3)
          do ip= 1, PEsmpTOT
          do i = STACKmc(ip-1,ic,lev)+1, STACKmc(ip,ic,lev)
            SW1= WW(3*i-2,R); SW2= WW(3*i-1,R); SW3= WW(3*i  ,R)
            isL= INL(i-1)+1; ieL= INL(i)
            do j= isL, ieL
              k= IAL(j)
              X1= WW(3*k-2,R); X2= WW(3*k-1,R); X3= WW(3*k  ,R)
              SW1= SW1 - AL(9*j-8)*X1 - AL(9*j-7)*X2 - AL(9*j-6)*X3
              SW2= SW2 - AL(9*j-5)*X1 - AL(9*j-4)*X2 - AL(9*j-3)*X3
              SW3= SW3 - AL(9*j-2)*X1 - AL(9*j-1)*X2 - AL(9*j  )*X3
            enddo
            X1= SW1; X2= SW2; X3= SW3
            X2= X2 - ALU(9*i-5)*X1
            X3= X3 - ALU(9*i-2)*X1 - ALU(9*i-1)*X2
            X3= ALU(9*i  )*  X3
            X2= ALU(9*i-4)*( X2 - ALU(9*i-3)*X3 )
            X1= ALU(9*i-8)*( X1 - ALU(9*i-6)*X3 - ALU(9*i-7)*X2)
            WW(3*i-2,R)= X1; WW(3*i-1,R)= X2; WW(3*i  ,R)= X3
          enddo
          enddo
!$omp end parallel do
        enddo

        call SOLVER_SEND_RECV_3_LEV(lev,…):   Communications using
                                              Hierarchical Comm. Tables.
      enddo
```

Fig.7 Forward substitution process of preconditioning written in FORTRAN and MPI with OpenMP directives. Global communications using hierarchical communication tables occur in the end of the computation at each level.

Fig.7 は前処理プロセスにおける前進代入処理を FORTRAN，MPI，OpenMP によって記述したものである。階層的通信テーブルを使用したグローバル通信は各レベルの計算の最後の部分で実施される

3. 実行環境の概要

T2K（東大）は筑波大，東大，京大の 3 大学で定められた「T2K オープンスパコン仕様」〔7〕に基づき日立製作所が製作した 952 ノード，約 15,000 コア，ピーク性能 140TFLOPS のクラスタ型コンピュータシステムである〔4〕。各ノードは AMD quad-core Opteron（2.3GHz）4 ソケット，合計 16 コアから構成されており（Fig.8），ノードあたりの記憶容量は 32GB（一部 128GB）である。

T2K（東大）は Fig.9 に示すように，内部でクラスタ群に分かれている。各クラスタ群内のノード間は Myrinet-10G（1 リンクあたり 1.25GB/sec×双方向）で接続されている。ノード A 群は各ノード 4 本（5.00GB/sec × 双方向），ノード B 群は 2 本（2.50GB/sec×双方向）である。

本研究ではノード A 群のうちの 32 ノード（合計 512 コア）を使用した。コンパイラは日立製専用コンパイラ（FORTRAN90）を使用した。各ノード 16 コアを全て使用した。

T2K（東大）は NUMA（Non-Uniform Memory Access）アーキテクチュアによっており，この特性を考慮したプログラミング，データ配置が必要となる。

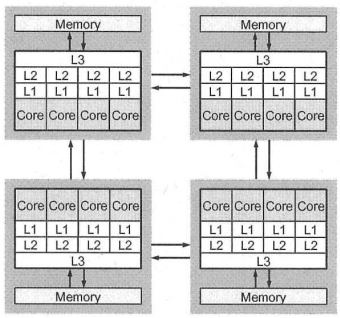


Fig. 8 Overview of a "node" of T2K/Tokyo, each node consists of four sockets of AMD Quad-core Opteron processors (2.3GHz)

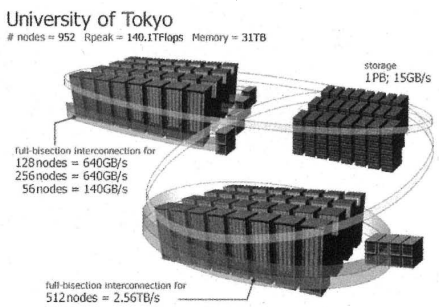


Fig. 9 Overview of "T2K Open Super Computer (Todai Combined Cluster) (T2K/Tokyo)" [7]

4. 予備的計算結果

(1) 概 要

T2K（東大）の 32 ノード（512）コアを使用して予備的な評価を実施した。問題サイズは 2,097,152 要素，6,440,067 自由度（Degrees of Freedom，DOF）である。Flat MPI と Hybrid 並列プログラミングモデルの比較を実施した。Hybrid については以下の 3 種類のプログラミングモデルを適用した。

- **Hybrid 4x4（HB 4x4）**：Fig.8 の各ソケットに OpenMP スレッド×4，ノード当たり 4 つの MPI プロセス
- **Hybrid 8x2（HB 8x2）**：2 ソケットに OpenMP スレッド×8，1 ノード当たり 2 つの MPI プロセス
- **Hybrid 16x1（HB 16x1）**：1 ノード全体に 16 の OpenMP スレッド，1 ノード当たりの MPI プロセスは 1 つ

(2) NUMA Control について

GeoFEM の局所分散データ構造に基づき，局所的なデータは各ローカルメモリに格納されているが，NUMA（Non Uniform Memory Access）アーキテクチュアの特性を利用するための実行時制御コマンド（NUMA control）使用して，コア（またはソケット）とメモリの関係を明示的に指定することによって，性能が向上することは〔2〕でも既に明らかとなっている。本研究でも，TABLE 1 に示す 6 種類の制御コマンド群（NUMA policy）を適用した。

TABLE 1 Summary of NUMA Policies

Policy ID	Command line switches
0	no command line switches
1	`--cpunodebind=$SOCKET --interleave=all`
2	`--cpunodebind=$SOCKET --interleave=$SOCKET`
3	`--cpunodebind=$SOCKET --membind=$SOCKET`
4	`--cpunodebind=$SOCKET --localalloc`
5	`--localalloc`

(3) 計算結果

Fig.10，11 は，線形ソルバー部分の収束（収束判定値＝10^{-8} 以下同様）に要した時間の比較である。反復回数は，Flat MPI：1,264 回，HB 4×4：1,261 回，HB 8×8：1,216 回，HB 16×1：1,244 回とほぼ一様である。

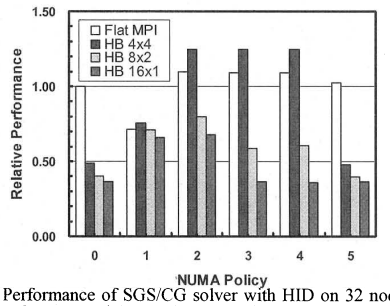


Fig.10 Performance of SGS/CG solver with HID on 32 nodes (512 cores) of T2K (Tokyo) for 3D linear-elastic model with heterogeneous distribution of Young's modulus ($E=10^{-3}$~10^{+3}), 2,097,152 elements, 6,440,067 DOF, Effect of NUMA control, Performance of Flat MPI (policy 0) is set to 1.0

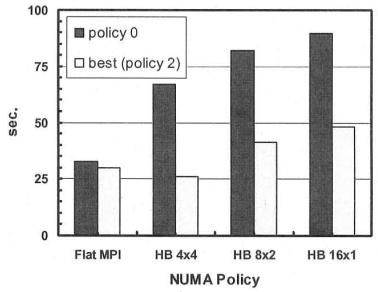


Fig.11 Performance of SGS/CG solver with HID on 32 nodes (512 cores) of T2K (Tokyo) for 3D linear-elastic model with heterogeneous distribution of Young's modulus ($E=10^{-3}$~10^{+3}), 2,097,152 elements, 6,440,067 DOF, Effect of NUMA control

Fig.10 は Flat MPI において，NUMA control を適用しない場合（policy 0）の性能を 1 とした相対性能で，1.0 より大きければ Flat MPI（policy 0）より速いことになる。Fig.11 は各並列プログラミングモデルについて，policy 0 の場合と，最速のケース（いずれの場合も policy 2）の計算時間が示してある。〔2〕と同様に，Hybrid においては NUMA control の効果は顕著であり，特に HB 4×4 では 3 倍近い改善が得られ，Flat MPI よりもやや良好な結果が得られている。HB 8×2 や HB 16×1 は，これと比較して 1.5 倍から 2.0 倍程度遅い。

5. 最適化

(1) 概 要

4. で示したように，NUMA control の適用によって，本研究においても，特に HB 4×4 では，大幅な性能向上が得られることがわかったが，HB 8×2，HB 16×1 では不十分である。将来，コア数が増加した場合のアプリケーションの信頼性を高めるためには，MPI プロセスの数を可能な限り減らすこと，すなわち，Hybrid 並列プログラミングモデルにおいて，MPI プロセス内の OpenMP スレッド数を増加させることが 1 つの方策である。従って，HB 8×2，HB 16×1 における性能向上の検討が必要である。

NUMA（Non-Uniform Memory Access）アーキテクチュアでは，各コアができるだけローカルなメモリ上（Fig.8 における各ソケットのメモリ上）にあるデータをアクセスできるように，データ配置等に配慮することが望ましい。一方，Flat MPI，Hybrid 4×4 では各コアの扱うデータがソケット上のメモリにあるように指定することは NUMA control によって可能である。

本研究では，Hybrid 並列プログラミングモデル（HB 4×4，8×2，16×1）以下の 2 項目について検討する：

- First Touch Data Placement の適用
- 連続データアクセスのためのデータ再配置

以下，ケース番号を以下のように設定する：

- ケース 1： 4. で紹介した予備的評価ケース
- ケース 2： First Touch Data Placement の適用
- ケース 3： ケース 2 に加えてデータの再配置を実施した場合

(2) First Touch Data Placement

NUMA アーキテクチュアでは，プログラムにおいて変数や配列を宣言した時点では，物理的メモリ上に記憶領域は確保されず，ある変数を最初にアクセスしたコア（の属するソケット）のローカルメモリ上に，その変数の記憶領域が確保される。これを First Touch Data Placement〔8〕と呼び，配列の初期化手順により大幅な性能の向上が達成できる場合もある。具体的には，Fig.12 に示すように，Fig.7 の実際の計算の手順にしたがって配列を初期化することによって実現できる。

Fig.13 は First Touch Data Placement の効果である。ケース 1 の Flat MPI（policy 0）の場合（Fig.10 参照）の性能を 1 として無次元化してある。ケース 1 からの改善は HB 4×4：1.25⇒1.25 と変わらないが，HB 8×2：0.80（policy 2）⇒0.89（policy 4），HB 16×1：0.68（policy 2）⇒0.70（policy 5） のようにわずかではあるが，性能が向上している。Fig.10 と Fig.13 を比較すると，HB 8×2，HB 16×1 では，policy 4，policy 5 における性能改善が顕著であることがわかる。また policy 0 の性能が全体として向上している。

```
      do lev= 1, LEVELtot
        do ic= 1, COLORtot(lev)
!$omp parallel do private(ip,i,j,isL,ieL,isU,ieU)
          do ip= 1, PEsmpTOT
          do i = STACKmc(ip,ic-1,lev)+1, STACKmc(ip,ic,lev)
            RHS(i)= 0.d0
            X  (i)= 0.d0
            D  (i)= 0.d0

            isL= indexL(i-1)+1
            ieL= indexL(i)
            do j= isL, ieL
              itemL(j)= 0; AL(j)= 0.d0
            enddo

            isU= indexU(i-1)+1
            ieU= indexU(i)
            do j= isU, ieU
              itemU(j)= 0; AU(j)= 0.d0
            enddo
          enddo
          enddo
        enddo
      enddo
```

Fig.12 Example of array initialization for efficient local mapping

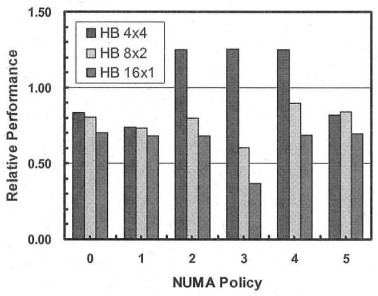


Fig.13 Performance of SGS/CG solver with HID on 32 nodes (512 cores) of T2K (Tokyo) for 3D linear-elastic model with heterogeneous distribution of Young's modulus ($E=10^{-3}$~10^{+3}), 2,097,152 elements, 6,440,067 DOF, Effect of First Touch Data Placement (CASE-2), Normalized by the result of Flat MPI (policy 0) of CASE-1 (Fig.10)

(3) データ再配置

First Touch Data Placement（ケース 2）ではわずかながら性能向上が見られた。現在の CM-RCM 法による並べ替えでは，Fig.14 に示すように：

- 同一の色（またはレベル）に属する要素は独立であり，並列に計算可能
- 「色」の順番に番号付け
- 色内の要素を各スレッドに振り分ける

という方式を採用しているが，同じスレッド（すなわち同じコア）に属する要素は連続の番号では無いため，効率が低下している可能性がある。そこでケース 3 では，同じスレッドで処理するデータをなるべく連続に配置するように更に並び替え，効率の向上を図ることとした。番号の付け替えによって要素番号の大小関係は変わる可能性があるが，上三角，下三角の関係は変えず，元の計算と反復回数が変わらないようにする。従って自分より要素番号が大きいのに下三角成分に含まれているような場合もありうる。Fig.15 がこのようなデータ再配置の概要である。

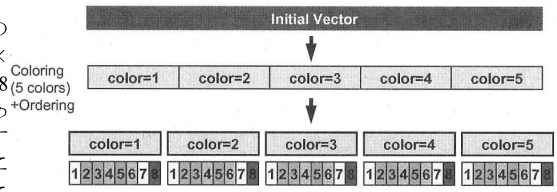


Fig.14 Original array configuration by CM-RCM, Because all arrays are numbered according to "color", discontinuous memory access may happen on each thread

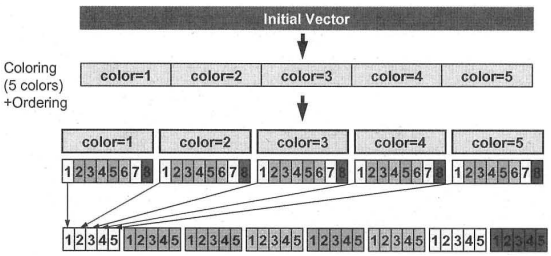


Fig.15 Array configuration in CASE-3, Continuous memory access is possible on each thread by further reordering

Fig.16 はケース 3 の計算結果（相対性能）である。ケース 2 からの改善は，HB 4×4：1.25（policy 2）⇒1.30（policy 3），HB 8×2：0.89（policy 4）⇒1.18（policy 4），HB 16×1：0.70（policy 5 ）⇒1.00（policy 5）となっておりそれぞれ性能の向上が得られている。Flat MPI の最適値（Fig.10，policy 2）が 1.10 であるので，HB 8×2 はこれを上回っており，HB 16×1 もこれに匹敵する性能の改善が得られていることがわかる。Fig.17 は各ケースにおいて最高性能を得た場合の結果をまとめたものである。

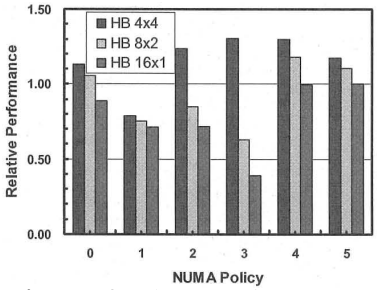


Fig.16 Performance of SGS/CG solver with HID on 32 nodes (512 cores) of T2K (Tokyo) for 3D linear-elastic model with heterogeneous distribution of Young's modulus ($E=10^{-3}$~10^{+3}), 2,097,152 elements, 6,440,067 DOF, Effect of Data Reconfiguration (CASE-3), Normalized by the result of Flat MPI (policy 0) of CASE-1 (Fig.10)

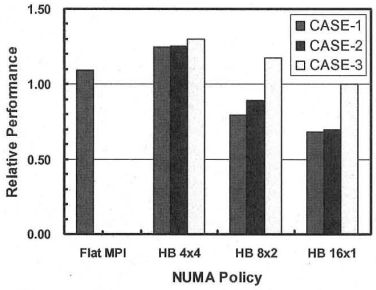


Fig.17 Performance of SGS/CG solver with HID on 32 nodes (512 cores) of T2K (Tokyo) for 3D linear-elastic model with heterogeneous distribution of Young's modulus ($E=10^{-3}$~10^{+3}), 2,097,152 elements, 6,440,067 DOF, Best results of CASE-1~CASE-3, Normalized by the result of Flat MPI (policy 0) of CASE-1 (Fig.10)

ケース 1 では，policy 2 が最適であるが，ケース 2，ケース 3 では policy 4, 5 など「`--localalloc`」を含む場合の性能が良い。

Fig.18 はケース 3 において，First Touch Data Placement を省略した場合で，HB 8×2，HB 16×1 の性能はケース 2 とほぼ同様で，ケース 3 で適用したデータ再配置（Fig.15）は First Touch Data Placement と組み合わせないと効果が得られないことがわかる。

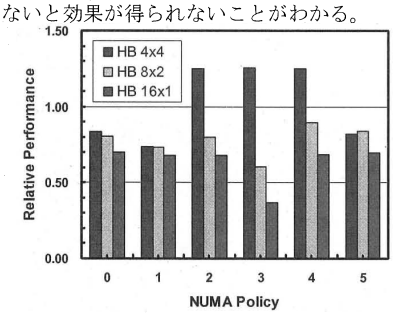


Fig.18 Performance of SGS/CG solver with HID on 32 nodes (512 cores) of T2K (Tokyo) for 3D linear-elastic model with heterogeneous distribution of Young's modulus ($E=10^{-3}$~10^{+3}), 2,097,152 elements, 6,440,067 DOF, CASE-3 without First Touch Data Placement, Normalized by the result of Flat MPI (policy 0) of CASE-1 (Fig.10)

6. ノード内並列化への HID の適用

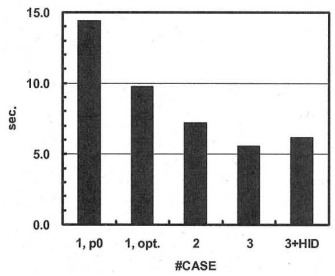


Fig.19 Performance of ICCG solver on a single node of T2K (Tokyo) for 3D homogeneous Poisson equations [9], 1,000,000 DOF, OpenMP with 16 threads, Comparison of Red-Black Coloring (CASE 1-3) and HID (CASE-3)

本研究では，ノード内の並列化に CM-RCM 法を適用しているが，高い Fill-in レベルに基づく前処理等を使用する場合，ノード内の並列化にも HID を適用することが考えられる。Fig.19 は〔9〕に示した，規則的な差分格子におけるポアソン方程式アプリケーションについて，本研究と同様のケース 1～3（Red-Black Multicoloring（2 色））を適用した場合と，ケース 3（First Touch Data Placement＋データ再配置）に HID を適用した場合の線形ソルバー（ICCG 法）の計算時間である。T2K（東大），1 ノードを使用し，16 スレッドの OpenMP を適用している。ケース 3＋HID は反復回数は，333 回から 259 回に減少しているものの，スレッド間の負荷が不均衡となるため，計算時間としては却って増加している。

7. まとめ

並列有限要素法アプリケーションに対して，階層型領域間境界分割に基づく SGS/CG 法および OpenMP と MPI の Hybrid 並列プログラミングモデルを適用し，T2K オープンスパコン（東大）の 512 コアにおいて Flat MPI との性能比較，最適化を実施した。First Touch Data Placement，データ再配置，「`--localalloc`」を含む NUMA control の組み合わせにより，Hybrid 並列プログラミングモデルが Flat MPI と同等かそれを上回る性能を得られることがわかった。

今後は，Fill-in レベルの高い前処理に対して適用できるように改良していくとともに，1 コアあたりの性能を高めていく必要がある。現状では対ピーク性能比は 3%程度であり，キャッシュの階層性を考慮した最適化，それに基づくライブラリ整備が急務である。

参 考 文 献

[1] Nakajima, K. (2007), The Impact of Parallel Programming Models on the Linear Algebra Performance for Finite Element Simulations, Lecture Notes in Computer Science 4395, 334-348

[2] Nakajima, K. (2008), Parallel Multistage Preconditioners by Hierarchical Interface Decomposition on "T2K Open Super Computer (Todai Combined Cluster)" with Hybrid Parallel Programming Models, Proceedings of the 2008 IEEE International Conference on Cluster Computing (Cluster 2008), 298-303

[3] Henon, P. and Saad, Y. (2007), A Parallel Multistage ILU Factorization based on a Hierarchical Graph Decomposition, SIAM Journal for Scientific Computing, 28-6, 2266-2293

[4] T2K オープンスパコン（東大）：http://www.cc.u-tokyo.ac.jp

[5] Deutsch, C.V. and Journel, A.G. (1998), GSLIB Geostatistical Software Library and User's Guide, Second Edition, Oxford University Press.

[6] GeoFEM: http://geofem.tokyo.rist.or.jp/

[7] The T2K Open Supercomputer Alliance: http://www.open-supercomputer.org/

[8] Mattson, T.G., Sanders, B.A. and Massingill, B.L. (2005), Patterns for Parallel Programming, Addison Wesley

[9] 中島研吾（2009）T2K オープンスパコン（東大）チューニング連載講座（その 5），OpenMP による並列化のテクニック：Hybrid 並列化に向けて，スーパーコンピューティングニュース（東京大学情報基盤センター）11-1