T2K オープンスパコン（東大）チューニング連載講座番外編

Hybrid 並列プログラミングモデルの評価（I）

中 島 研 吾

東京大学情報基盤センター

1. はじめに

本「スーパーコンピューティングニュース」では，2008 年 5 月号から 2009 年 3 月号まで 6 巻，1 年間にわたって T2K オープンスパコン（東大）チューニング講座[1]を連載し，各方面から好評をいただいた．本稿はその番外編として，特にノード（またはソケット）内に OpenMP，ノード間に MPI を適用したいわゆる「Hybrid」並列プログラミングについて，

「有限要素法アプリケーションから得られる疎行列を，前処理付反復法で解く」

事例を中心に解説する．

ノード内に OpenMP を適用した事例については，2008 年 12 月 3 日，4 日に開催した日本応用数理学会「2008 年秋の学校：科学技術計算のためのマルチコアプログラミング入門[2]」（共催：東京大学情報基盤センター）等で扱い，本誌でも既に紹介済である〔1〕．「秋の学校」の場でも多くの受講者から「Hybrid 並列についても教えてほしい」という要望が多かった．

「Hybrid」並列プログラミングについては，まだまだ，色々な例を試しながら知見を得ている段階であり，講義や講習会の教材としてまとめるには多少完成度が不足しているが，最近の経験も踏まえて，今回と次回（もしかしたらもう一回くらい）で解説する．なお，これらの事例については筆者による論文や解説記事〔1～4〕と重複する部分もあり，より詳細な情報についてはこれらの参考文献を参照されたい．

本解説では，T2K オープンスパコン（東大）を中心とするが，参考のため，下記のシステムについての事例も紹介する．

- Hitachi SR11000/J2（東京大学情報基盤センター）[3]
- Cray XT4（アメリカ国立ローレンスバークレイ研究所）[4]

また，今回は主として 1 ノード（16 コア）について，次回以降は複数ノードのケースについて紹介する．

2. 背 景

（1）Hybrid 並列プログラミングモデル

近年プロセッサのマルチコア化が進み，並列計算におけるプログラミングモデルとして，複

[1] http://www.cc.u-tokyo.ac.jp/publication/news/
[2] http://nkl.cc.u-tokyo.ac.jp/seminars/0812-JSIAM/
[3] http://www.cc.u-tokyo.ac.jp/service/intro/
[4] http://www.nersc.gov/nusers/systems/franklin/

数のコアを有するノード（またはソケット）内に OpenMP，ノード間に MPI を適用する「Hybrid」並列プログラミングモデルが再び脚光を浴びている．

ノード上のメモリを複数の CPU で共有する SMP（Symmetric Multiprocessors）をネットワークで結合した SMP クラスタは 1990 年代半ばから「テラスケール」スーパーコンピュータの主流となった．代表的なものが米国エネルギー省の ASCI 計画（現 ASC（Advanced Simulation and Computing））で開発された，IBM SP3，IBM System p5 シリーズに基づくハードウェア群，日本の「地球シミュレータ」である．当時，ノード内の CPU を全て独立に扱い，全てに MPI を適用する Flat MPI（または Pure MPI，図 1（a））と Hybrid（図 1（b））の優劣については盛んに論じられた．Flat MPI では，CPU 数分だけのプロセスが発生する．Hybrid では，ノード数分だけプロセスが発生し，各ノード内には CPU 数（図 1 の場合は 4）に対応したスレッドが発生する．Hybrid では Flat MPI と比較して，MPI プロセス数が少なくて済む（図 1 の場合は 4 分の 1）．

参考文献〔5〕によると Flat MPI と Hybrid の優劣は：

① 対象とするアプリケーションの性質，問題サイズ
② ハードウェア諸元（CPU 速度，メモリ性能，ネットワーク性能，それらのバランス）

によって決まり一意に決めることは難しい．今世紀初頭を中心に様々な研究が実施されたが，Hybrid は余り流行らなかった．最大の理由は，プログラミングの困難さと比べて，得られる性能の向上が少なく，アプリケーションによっては却って低下する場合もあることである．

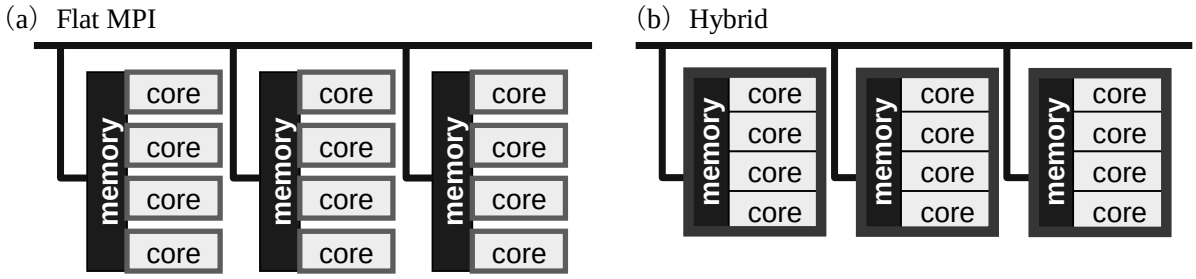


図 1　Flat MPI，Hybrid 並列プログラミングモデル

(2) 有限要素法，反復法と並列プログラミングモデル

筆者は「GeoFEM[5]」という地球シミュレータ向けの並列有限要素法のためのフレームワークを開発するプロジェクトを通して Hybrid 並列プログラミングモデルと関わることになった〔6〕．有限要素法は，図 2 に示すように対象領域を要素分割することによって偏微分方程式を数値的に解く手法であり，様々な実用的問題に使用されている．有限要素法は最終的には個々の要素において成立する線形化された積分方程式を重ね合わせて得られる大規模で「疎な」（sparse，0 成分が大部分を

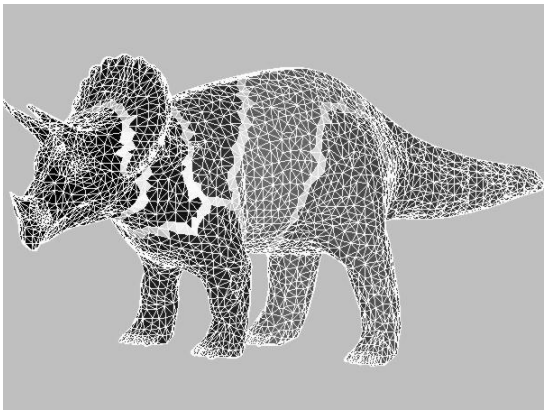


図 2　有限要素による要素分割・領域分割例，白い帯状の部分は並列計算向けに領域分割した場合の領域間境界である

[5] http://geofem.tokyo.rist.or.jp/

占める）係数行列を持つ連立一次方程式を解くことに帰着させられる．対象とする問題にもよるが，有限要素法で最も時間を要するプロセスはこの連立一次方程式を解く部分である．方程式の解法としては，逆行列を陽に計算する直接法と，反復的に計算する反復法があるが，大規模な疎行列を並列計算機上で解く場合，共役勾配法（Conjugate Gradient，CG 法）などの反復法が広く使用されている．従って，並列反復法の高速化が，並列計算機上での大規模有限要素法アプリケーションの計算効率の鍵を握っているのである．

有限要素法は，計算プロセスの点からは以下のような特徴がある：

① 要素内で成立するローカルな方程式に基づくため，得られる係数行列は 0 成分が多い疎行列である．係数行列が密であれば，CG 法などの反復法によく現れる{Y}=[A]{X}という行列ベクトル積を計算する場合に，以下に示すように計算する：

```
for (i=0; i<N; i++) {
  for (j=0; j<N; j++{
    Y[i]= Y[i] + A[i,j]*X[j];
  }
}
```

疎行列の場合には，非ゼロ成分だけ記憶しておけばよいため，下記のようになる：

```
for (i=0; i<N; i++) {
  for (k=Index(i-1); k<Index(i); k++{
    Y[i]= Y[i] + A [k]*X[Item[k]];
  }
}
```

ここで Index は各行における非ゼロ成分の数，Item は対応する列番号である．密行列の場合と比較すると，間接参照が多くなり，メモリへの負担が大きくなるため，メモリ性能がアプリケーション全体の性能を決定する，すなわち memory bound なプロセスとなる．

② 並列計算においては，全体領域を MPI の各プロセスに割り当てて計算するが，元々要素ごとのローカルな方程式を基本としているため，通信は領域境界を中心に，隣接している領域においてのみ発生する．従って，通信バンド幅よりはレイテンシがよりクリティカルである．疎行列を対象とした反復法を並列化する場合についても同じことが言える．

図 3 は「地球シミュレータ（先代）」160 ノード（1,280（=160×8）PE（Processing Element），地球シミュレータ（先代）では各ノードに 8 PE が搭載されている）を使用して，三次元弾性体における静的な荷重のつりあいの問題（静的弾性問題）を有限要素法で解く場合に得られる疎行列を ICCG 法（CG 法に不完全コレスキー法（Incomplete Cholesky Facorization，IC）による前処理を施したもの）で解いた場合の性能比較である〔6〕．横軸が PE 数，縦軸が TFLOPS 値である．ノードあたりの

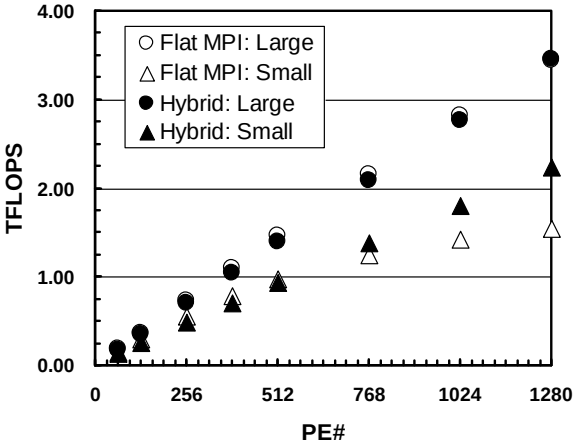


図 3　「地球シミュレータ（先代）」上での三次元静的弾性問題向け ICCG ソルバーの性能比較（Weak Scaling）(Large: 12,582,912 DOF/node, Small: 786,432 DOF/node), (DOF：自由度（Degrees of Freedom），PE: Processing Element)〔6〕

問題規模を固定したいわゆる「Weak Scaling」の計算結果である（これに対して「Strong Scaling」では，全体の問題規模を固定して，ノード数を変化させる）．本来，性能はノード数，PE 数に比例して増加するはずであるが，通信のオーバーヘッドがあるためノード数が増加すると理想値よりは若干低めとなる．

160 ノードを使用する場合，Flat MPI では全体領域を PE 数（1,280＝160×8）に分割する．Hybrid では全体を 160 に分割し，各領域に 8 個の OpenMP スレッドを発生させる．

ノードあたりの問題規模が大きいケース（○●）は，Flat MPI と Hybrid の差は無くほぼ理想値に近い効率であるが，ノード数が増加すると，若干 Hybrid が優位になる．ノードあたりの問題規模が小さい場合（△▲）は最内ループ長が短く，「地球シミュレータ」のようなベクトル型並列計算機では絶対的な性能が全体的に低いが，ノード数が増加すると Hybrid（▲）が優位である．「地球シミュレータ（先代）」は，CPU・メモリ性能，通信バンド幅と比較して，通信レイテンシが比較的大きい〔6〕．ノードあたり問題規模が小さい場合は，レイテンシによるオーバーヘッドの効果が顕著となり，ノード数増加によって更に増幅され，MPI プロセス数の少ない Hybrid の方が優位となっている．このような現象の可能性ついては，地球シミュレータ（先代）が本格的に稼動する前から米国ロスアラモス国立研究所の性能評価モデル〔7〕によって予測されていた．しかし，このような Hybrid の優位性は通信レイテンシ値が相対的に大きい「地球シミュレータ（先代）」特有の現象であり 2003 年頃は，他の超並列計算機では観察されていなかった〔6〕．

（3）歴史はまた繰り返す：何故また Hybrid か？

James Sexton（IBM Research）による最近の講演[6]によると：

- コモディティプロセッサのコアあたり性能は今後も 2～4GHz 程度に留まり，ペタスケール（Peta=10^{15},「テラ」の 1,000 倍）のシステムのコア数は数十万，エクサスケール（エクサ：Exa=10^{18},「ペタ」の更に 1,000 倍）では数億の規模になる
- メモリの性能は将来それほど向上せず, むしろ消費電力を下げることが研究開発の中心となるであろう

ということである．

既に述べたように，並列計算機による有限要素法アプリケーションでは，大規模な疎行列を反復法で解く部分が最も計算時間がかかる. また, 疎行列を対象とした並列反復法においては，レイテンシがクリティカルである．この影響は MPI プロセスが増加するほど深刻となるため，ペタ／エクサスケールのシステムでは Hybrid 並列プログラミングモデルの導入によって，MPI プロセス数の爆発的な増加を少しでも抑制することが重要である．

また疎行列を対象とした並列反復法は memory bound なプロセスである．従って，メモリに負担をかけないように，できるだけ各コアあたりの問題規模を小さく抑えて，多くのコアを使って計算することが得策である．図 3 の説明でも示したように，このような場合も，Hybrid 並列プログラミングモデルが有利となる可能性がある．

---

[6] Sexton, J.（2009）Computational Science Challenges from Petascale and Exascale Computing, SIAM Conference on Computational and Engineering (CSE09), Miami, FL, USA

このような背景もあり，マルチコアの時代を迎えて，Hybrid 並列プログラミングモデルは再び脚光を浴びつつある．2008 年初頭から SIAM[7]や SC-XY Conference Series[8]でも関連した発表やチュートリアルが目立つようになってきた．今世紀初頭のブームの時と違い，現在は T2K オープンスパコンに代表されるマルチコア，マルチソケットの cc-NUMA（Cache Coherent Non-Uniform Memory Access）アーキテクチュアが登場している．

T2K オープンスパコンは図 4 に示すように，各ノード上に 4 コアを有する AMD Opteron（2.3GHz）(Barcelona)を 4 ソケット搭載している（合計 16 コア）．SMP では全てのプロセッサからメモリに平等にアクセスすることが可能であった．T2K オープンスパコンの各ノード内では他ソケットのメモリ上のデータをアクセスすることは可能であるが，ローカルなメモリと比べてアクセスに時間がかかる（Non-Uniform Memory Access）．ここで，cc-NUMA の「Cache-Coherent」とは「キャッシュが整合している」すなわち，メモリ上と各ソケット上のキャッシュ上のデータの整合性が保たれる，ということである．従って，計算効率を上げるためにはできるだけ各ソケット上のローカルなメモリ上にデータを格納するような工夫が必要となる．そのために，①実行時制御コマンド（NUMA control），②First Touch Data Placement，③データのメモリ上での連続アクセスが重要であることは，既に知られている〔1，2〕．

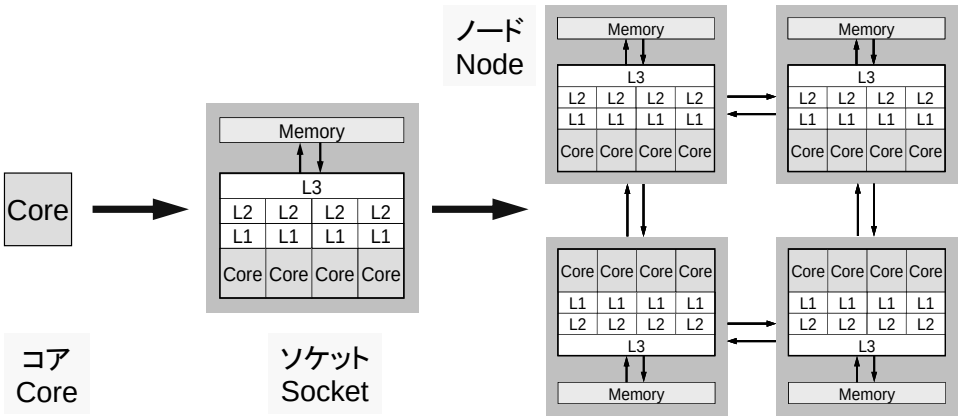


図4　T2Kオープンスパコン（東大）の各ノードの構成

3. アプリケーションの概要

本稿ではGeoFEMプロジェクト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群〔6〕を使用した GeoFEM ベンチマークは，

① 三次元弾性静解析問題（Cube 型モデル，PGA モデル）
② 三次元接触問題
③ 二重球殻間領域三次元ポアソン方程式

に関する並列前処理付き反復法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．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

[7] http://www.siam.org/　Society for Industrial and Applied Mathematics（米国応用数理学会）
[8] http://www.sc-conference.org/　毎年 11 月にアメリカで開催されている IEEE 主催による国際会議 The International Conference for High Performance Computing Networking, Storage, and Analysis のこと

記述されている．各ベンチマークプログラムでは，GeoFEM で採用されている局所分散データ構造〔6〕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ベクトルプロセッサ, SMP, 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．連立一次方程式の係数マトリクスの格納法として（a）CRS（Compressed Row Storage），（b）DJDS（Descending order Jagged Diagonal Storage）の 2 種類の方法が準備されているが，本稿ではスカラープロセッサ向けの CRS 法を使用した．

本稿では，3 種類のベンチマークのうち図 5 に示すような一様な物性を有する単純形状（Cube 型）を対象とした三次元弾性静解析問題を扱った．係数行列が対称正定な疎行列となることから，SGS（Symmetric Gauss-Seidel）〔6〕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．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 つのブロックして取り扱っている．

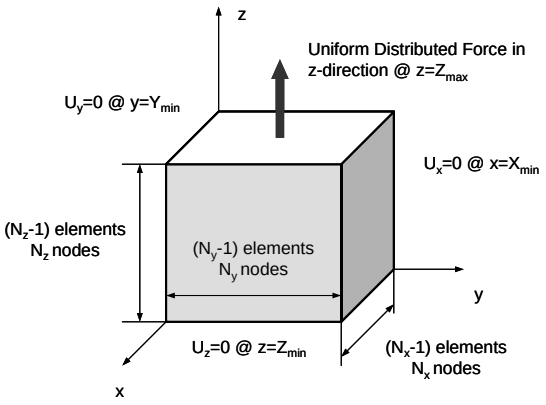


図 5　Cube 型ベンチマークの境界条件

4. リオーダリング手法

不完全 LU／コレスキー分解，SGS 等前処理等に基づく反復法を OpenMP を使用してマルチコアプロセッサ上で並列化しようとすると，内積，疎行列ベクトル積，DAXPY などのプロセスではディレクティヴを挿入するだけでよいが，行列の分解プロセス，前進後退代入プロセスでは「データ依存性」が生じるため, この依存性を排除するためにデータの並び替え（reordering, リオーダリング）が必要となる．基本的な考え方は，グラフを構成する節点（node，vertex）を互いに依存性を持たないグループ同士で色分けし，同じ「色」に属する節点が互いに独立であることを利用して並列計算を実施する，というものである〔8，9〕．図 6 は全体を 5 つの色に分類し，その色の順番に節点の番号を並び替え，各色内の節点を 8 つのスレッドで並列に計算を行なう例である．

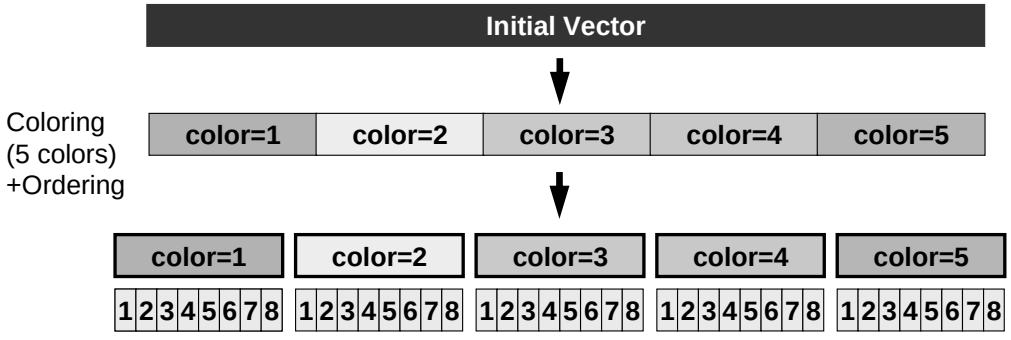
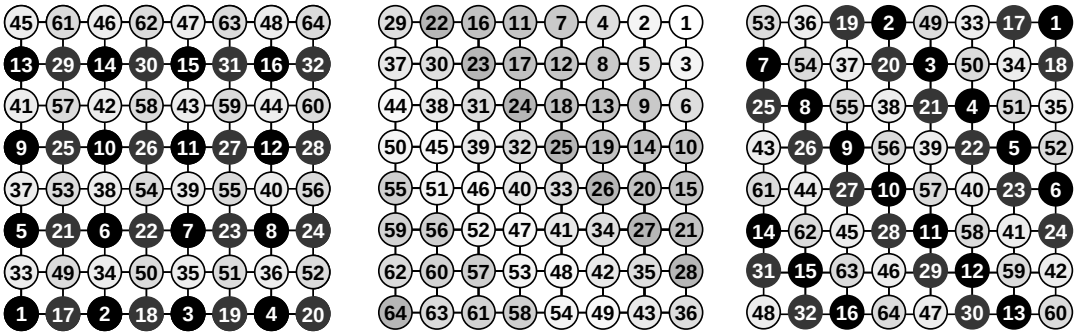


図 6　リオーダリングによるデータ依存性の回避（5 色，8 スレッドの場合）

このようなリオーダリング手法として最もよく使用されているのが，マルチカラー法（Multicoloring，MC）法である．色数が 2 色の特別の場合は，特に Red-Black 法と呼ばれ，規則正しい差分格子に適用される．図 7（a）は MC 法（4 色）の例である．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，特に規則正しい形状の場合，もとの自然な番号付け（辞書的番号付け）と比較して反復回数が増加する〔8，9〕．一般的には，色数を増やすことによって収束を改善できるが，図 8 に示すように OpenMP の同期オーバーヘッドが増加するため，性能が低下する場合がある〔6〕．また，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（図 7（b））は，MC 法と比較して収束性は良いが，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 法と比べて低い．これ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〔8〕．図 7（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．本稿では，MC 法，CM-RCM 法，RCM 法の比較も実施した．

MC 法，RCM 法等の詳細については〔9〕を参考にされたい．



（a）MC 法（4 色）　　（b）RCM 法　　（c）CM-RCM 法（4 色）

図 7　リオーダリングの例

```
    do ic= 1, COLORtot
!$omp parallel do private(ip,i,SW,isL,ieL,j,k,Xk)
      do ip= 1, PEsmpTOT
        do i = STACKmc(ip-1,ic)+1, STACKmc(ip,ic)
          SW= WW(i-2,R)
          isL= INL(i-1)+1; ieL= INL(i)
          do j= isL, ieL
              k= IAL(j)
             Xk= WW(k,Z)
             SW= SW - AL(j)*Xk
          enddo
          WW(i,Z)= Xk/D(i)
        enddo
      enddo
!$omp end parallel do
    enddo
```

OpenMP 並列化

図 8　SGS 前処理における前進代入プロセスの OpenMP による並列化例
（図 6 に示すようなリオーダリングを適用してデータ依存性が排除されている）

5. 計算環境

本稿では，Hitachi SR11000/J2（以降 SR11K），T2K オープンスパコン（東大）（以降 T2K（東大））（東京大学情報基盤センター）の 1 ノード 16 コアを使用した．

SR11K は，2 つの POWER5+コア（2.3GHz，ピーク性能 9.2GFLOPS）によって POWER5+チップが構成される．4 つのチップ，すなわち 8 つのコアから構成されるモジュール（Multi Core Module，MCM）2 つから成る 16-way のユニット（図 9）が 1 ノードを構成している．各コアは 32KB の L1 キャッシュを持ち，L2・L3 キャッシュは各チップ内で 2 つのコアに共有されており，サイズは各々1.875MB，36MB である．コンパイラとしては日立製最適化コンパイラ（オプション：-Oss）を使用した．

T2K（東大）の各ノードは AMD quad-core Opteron（2.3GHz）4 ソケット，合計 16 コアから構成される（図 4）．各コアは L1 キャッシュ（64KB），L2 キャッシュ（512KB）を持ち，L3 キャッシュ（2.048MB）は各ソケットで 4 つのコアに共有される．コンパイラとしては日立製最適化コンパイラ（オプション：-Oss）を使用した．

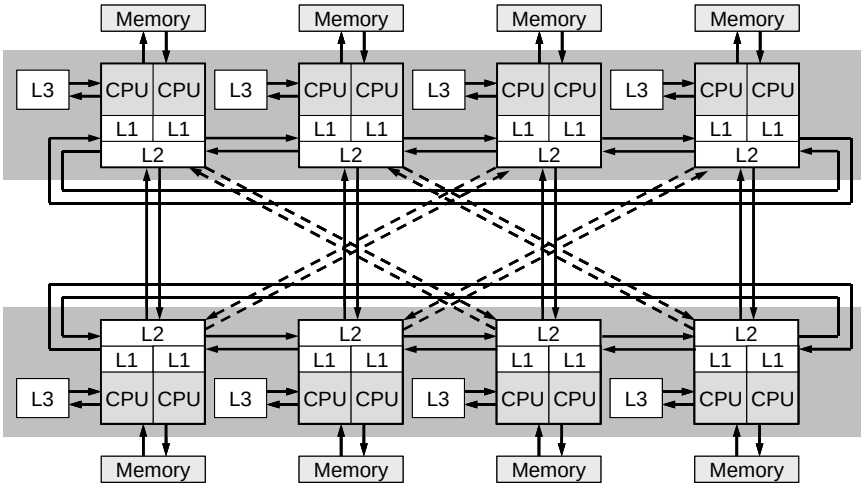


図9　Hitachi SR11000/J2のノード（プロセッサブック）のアーキテクチュア（〔10〕により作成）
点線はHitachi SR11000/J2特有のマルチコアモジュール（Multi Core Module，MCM）間の結線を示す

表 1　T2K（東大），Hitachi SR11000/J2 のノード諸元比較

	T2K（東大）	Hitachi SR11000/J2
L1（命令）キャッシュ	64 KB/core	32 KB/core
L1（データ）キャッシュ	64 KB/core	64 KB/core
L2 キャッシュ	512 KB/core	1,875 KB/chip（2 cores）
L3 キャッシュ	2,048 KB/socket（4 cores）	36,000 KB/chip（2 cores）
ピーク性能	147.2 GFLOPS/node	
実測メモリバンド幅[9]	19.6 GB/sec/node	102.4 GB/sec/node

表 1 は T2K（東大），SR11K のノード諸元を比較したものである．両者は NUMA アーキテクチャによっているが, SR11K はメモリのレイテンシが小さいためこの影響は比較的少なく, T2K（東大）では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．1 コア当りピーク性能は共に 9.2GFLOPS であり，ノード当りピーク性能は等しい（147.2GFLOPS）．1 ノードあたりのメモリバンド幅は SR11K：約 100GB/s，T2K（東大）：約 20GB/s（表 1）と大きく異

[9] STREAM ベンチマークによる実測値（Triad）（http://www.streambench.org/）

なり，本稿で対象とする疎行列ソルバーのように memory bound なアプリケーションではこの差が大きく影響すると考えられる〔6〕．

また，比較のため，Cray XT4（以降 XT4）（アメリカ国立ローレンスバークレイ研究所）4 ノード 16 コアによる計算も実施した．XT4 の各ノードは AMD quad-core Opteron（2.3GHz）1 ソケット 4 コアから構成され，T2K（東大）の 1 ソケットと全く同じである．

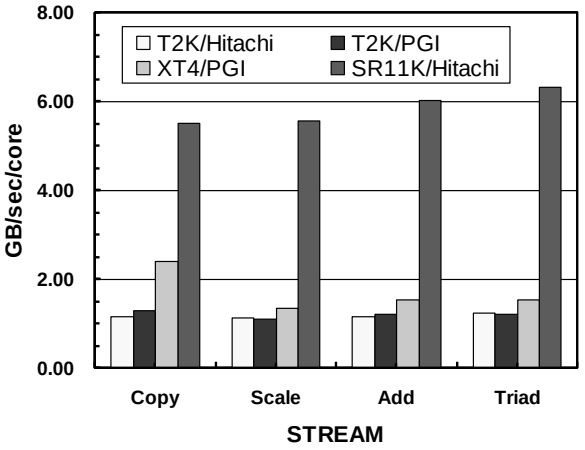


図 10　コア当たりメモリバンド幅，STREAM ベンチマーク結果，Flat MPI 16 コア

コンパイラとしては，PGI コンパイラ（オプション：-O3）を使用した．

図 10 はメモリバンド幅測定のための STREAM ベンチマークを 16 コア，Flat MPI で実施した場合の結果をコアあたりのメモリバンド幅に換算したものである．T2K については日立，PGI の 2 種類のコンパイラを適用した結果を比較したが，差異は認められなかった．XT4 はソケット間の coherency を考慮しない分，T2K（東大）より性能が高く，Copy で約 2 倍，その他のベンチマークでは 10%～20%高かった．

6. 評価結果

（1）並列プログラミングモデル

並列プログラミングモデルとしては各コアを独立に扱う Flat　MPI と Hybrid 並列プログラミングモデルの両者を実施した．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 4×4（HB 4×4）**：　スレッド数 4 の MPI プロセスを 4 つ起動する，T2K（東大）（図 3）の場合，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 8×2（HB 8×2）**：スレッド数 8 の MPI プロセスを 2 つ起動する，T2K（東大）（図 3）の場合，2 ソケットに OpenMP スレッド×8，ノード当たり 2 つの MPI プロセス（XT4 については実施せず）
- **Hybrid 16×1（HB 16×1）**：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（XT4 については実施せず）

（2）評価ケース・データ配置

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，　T2K（東大）では，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既に明らかとなっている〔1，2〕．本稿では，様々な実行時制御コマンドの組み合わせの中で最適のものを選択して適用した．

この他，Hybrid 並列プログラミングモデルを使用する場合，①First Touch Data Placement の適用，②連続データアクセスのためのデータ再配置によって性能が改善することも明らかとなっ

ている〔1，2〕.

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〔11〕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，図 8 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．

MC，RCM，CM-RCM 法による並べ替えでは，図 6 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．図 11 に示すように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，更に First Touch Data Placement を適用することによって性能が向上することは〔1，2〕でも既に明らかとなっている．

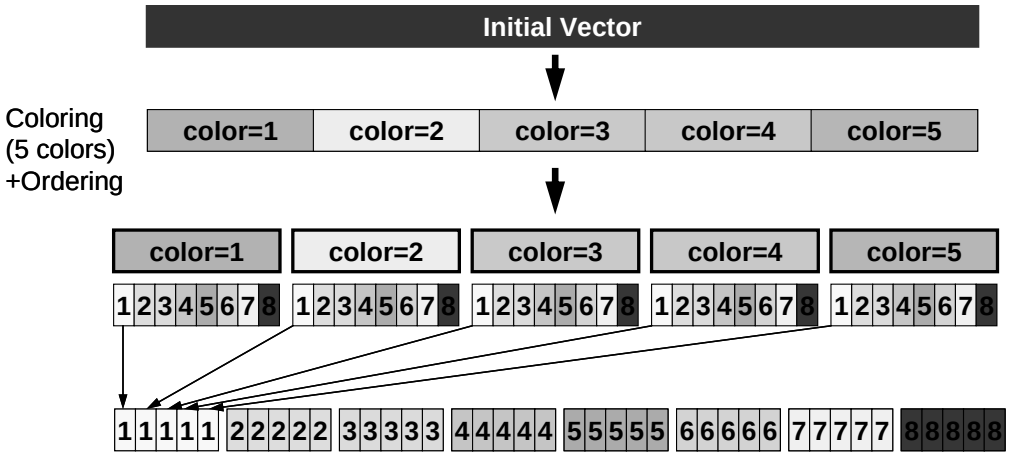


図 11　連続データアクセスのためのデータ再配置（5 色，8 スレッドの場合）

本稿では以下の 3 ケースについて評価を実施した：

- CASE-1：図 6 に示すリオーダリングを適用した状態
- CASE-2：更に First Touch Data Placement の適用（Flat MPI は除く）
- CASE-3：更に図 11 に示すデータ再配置を適用（Flat MPI は除く）

（3）様々な問題サイズにおける評価

図 5 に示した Cube 型ベンチマークにおいて，問題サイズを 4,096 節点（12,288 自由度）～2,097,152 節点（6,291,456 自由度）まで変化させた場合の反復法ソルバー（SGS／CG 法）の計算性能（GFLOPS）を図 12，13 に示す．CM-RCM（色数 10）を適用した．SR11K と T2K（東京）を比較すると，4. で示したようにメモリバンド幅の影響が大きく，T2K（東京）の性能は SR11K の 25%～30%程度である．両者ともスカラープロセッサであるため，問題サイズが大き

くなると性能が低下する傾向があるが, 表 1 で示したように SR11K はキャッシュサイズが大きく，より大きい問題サイズで性能が低下する．

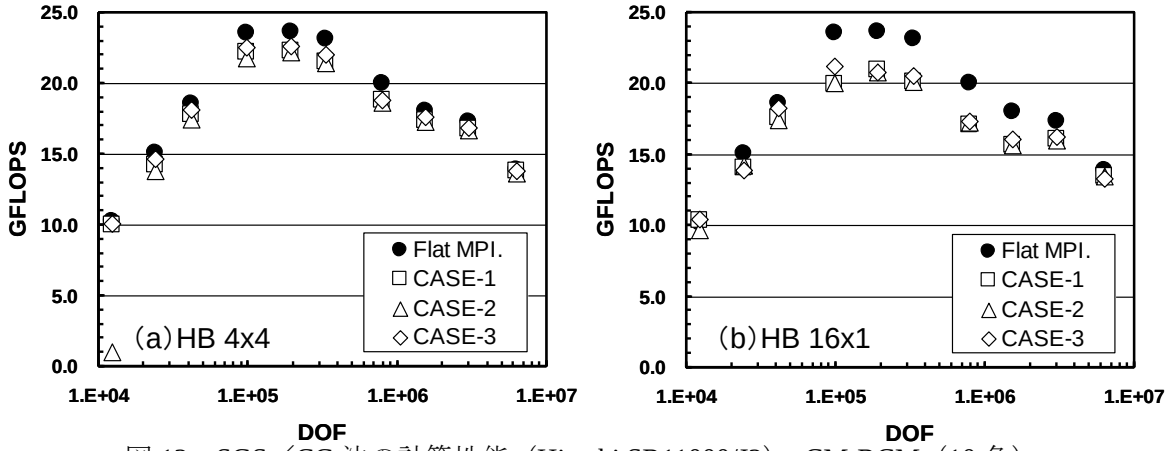


図 12　SGS／CG 法の計算性能（Hitachi SR11000/J2），CM-RCM（10 色）

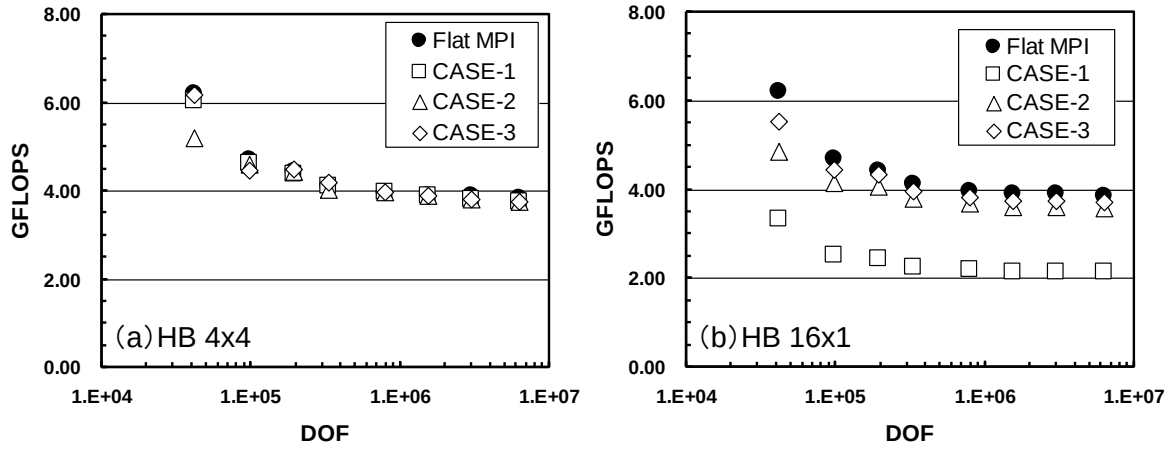


図 13　SGS／CG 法の計算性能（T2K オープンスパコン（東大)），CM-RCM（10 色）

いずれの場合も，HB 4×4 の場合は Flat MPI との差異はほとんど無く，CASE-1～CASE-3 の差も無い．HB 16×1 については，SR11K では全体的な性能が若干低下するものの傾向は同じであるが，T2K（東大）では，CASE-1 と CASE-2，-3 の差異が明らかで，First Touch Data Placement が重要であることがわかる．これは〔1, 2〕に示した例と同じ傾向である．しかし，CASE-2，CASE-3 の差異は小さく，図 11 に示すデータ配置の影響は小さい．HB 8×2 の結果は省略したが HB 16×1 と同様の傾向である．図 14 は CASE-3 の結果を T2K（東大）と XT4 で比較したものである．各ソケットは同じであるが，図 10 に示すような実効メモリバンド幅の違いもあり，1.50 倍から 2.00 倍程度 XT4 の方が性能が良い．

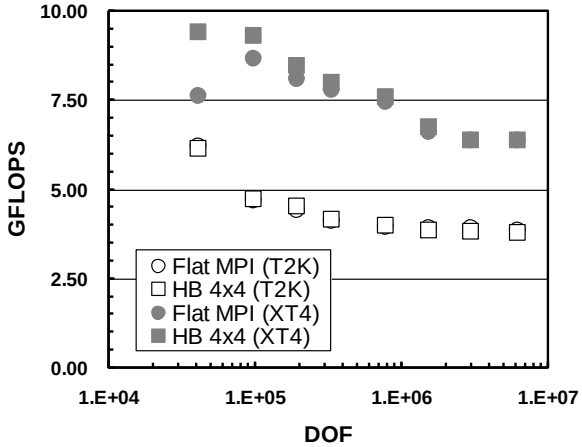


図 14　T2K（東大）と XT4 の比較（CASE-3）

T2K（東大）では，表 2 に示すような NUMA Policy を各ケースにおいて適用した．図 15 は

最大問題サイズ（2,097,152 節点（6,291,456 自由度））における NUMA Policy の影響であり，1 ノード・16 コアの性能である．Flat MPI については，全て CASE-1 の結果が表示してある．各プログラミングモデルにおいて適用する NUMA Policy によって性能は大きく左右される. HB 8×2，16×1 において First Touch，データ再配置（図 11）の影響が大きいことがわかる．特に policy4 ，policy5（表 2）の場合の性能増加が顕著である．図 15（d）に示したように，First Touch をしないとデータ再配置の効果は全く無い．図 13，14 の結果は最適な NUMA Policy を適用したケースの結果である。

表 2　適用した NUMA Policy

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`

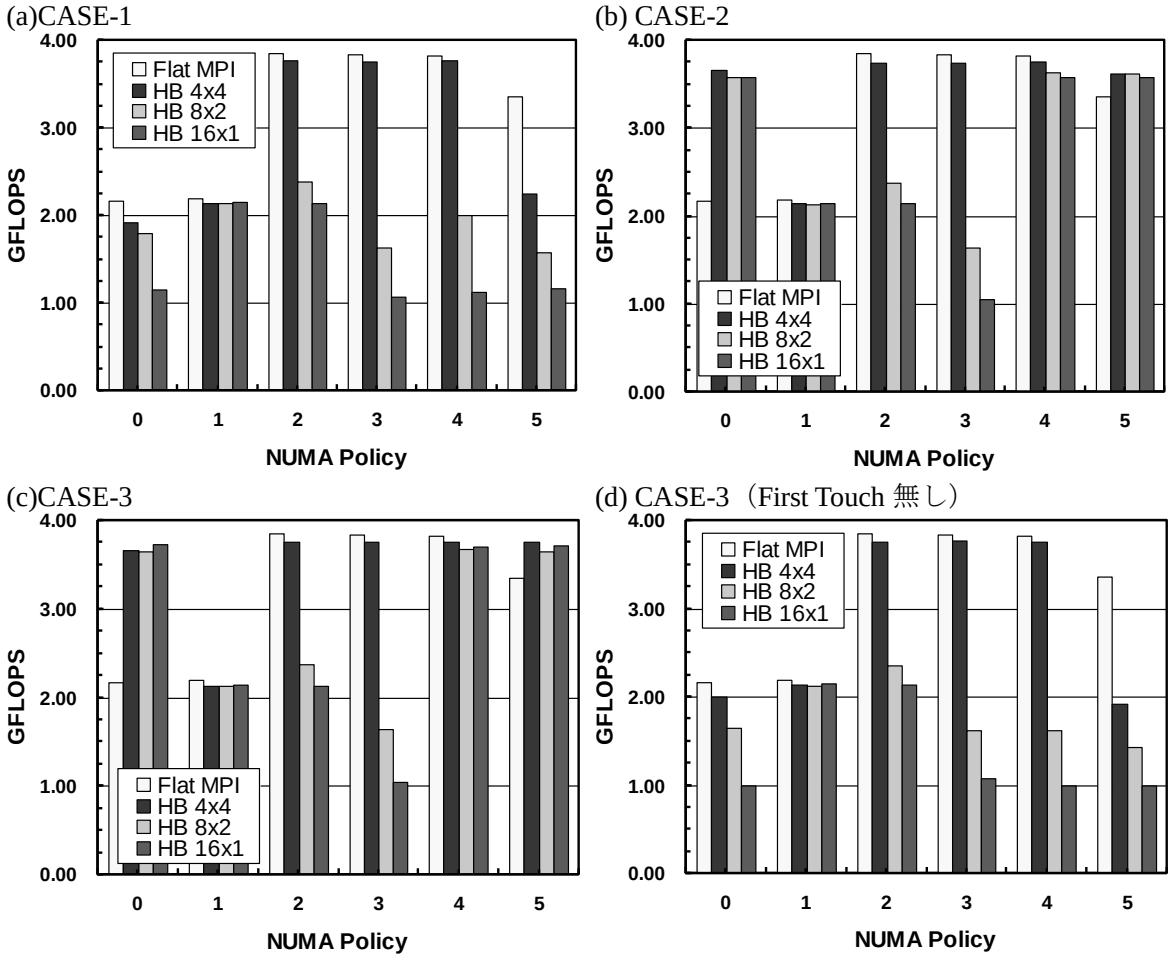


図 15　最大問題サイズ（2,097,152 節点（6,291,456 自由度））における NUMA Policy の影響（T2K（東大），1 ノード 16 コアあたりの性能）（Flat MPI は各図で CASE-1 の結果が表示されている）.

（4）リオーダリング手法，色数の影響

続いて，問題規模を 1,000,000 節点（3,000,000 自由度）に固定して，リオーダリング手法, 色数の影響について評価した．Hybrid については CASE-2, CASE-3 のみ実施した．図 7 に示す MC 法，RCM 法，CM-RCM 法について検討した．図 16 は各並列プログラミングモデルにおける，収束（残差ノルム＝10^{-8}）までの反復回数である．ここで RCM 法は CM-RCM 法の色数最大のケースに相当する（Flat MPI：319 色，HB 4×4：544 色，HB 8×2：644 色，HB 16×1：694 色）．

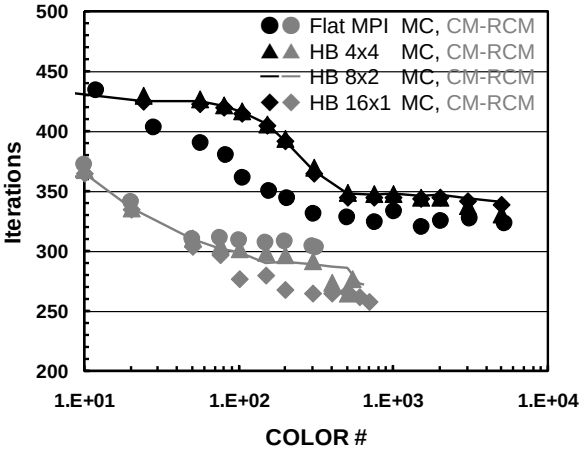


図 16　SGS／CG 法の収束までの反復回数
（1,000,000 節点，3,000,000 自由度）

色数が増加するとともに収束までの反復回数は減少していうことがわかる．

MC法等の色数とSGS／CG法，ICCG法等の前処理付反復法の収束性への効果については，これまで様々な研究によって説明が試みられているが，〔8〕においては土肥らによって「Incompatible Nodes（以下ICN）」の概念に基づいて説明されている．

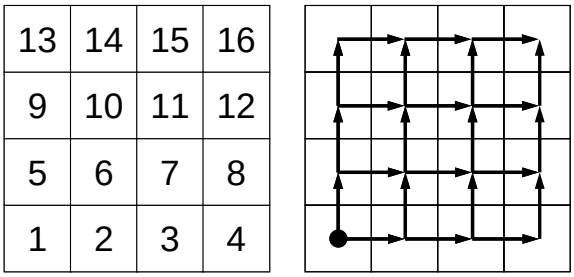


図17　初期状態（●：Incompatible Nodes）

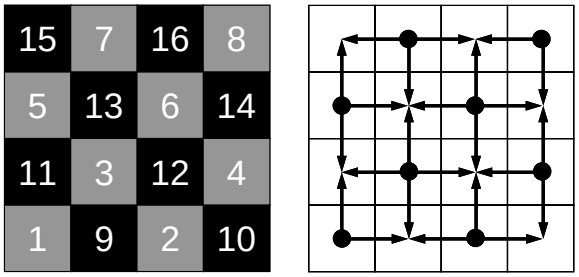


図18　MC（色数:2）（●：Incompatible Nodes）

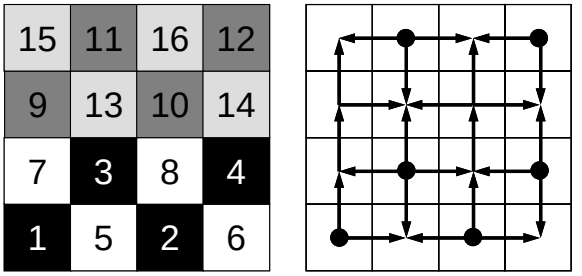


図19　MC（色数:4）（●：Incompatible Nodes）

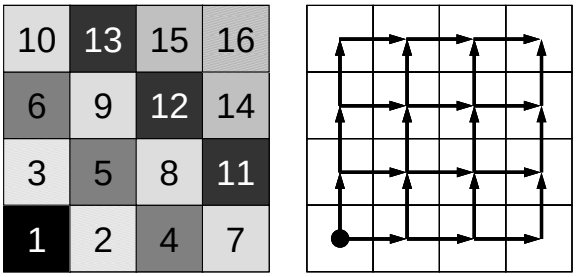


図20　CM（レベル数:7）（●：Incompatible Nodes）

図17に示した16要素の二次元体系において，要素番号順に前進代入あるいはGauss-Seidelのような操作を施した場合，要素1以外の節点は全て，番号の若い要素の影響を受ける．ICNとは，この図における要素1，すなわち，他の要素から影響を受けない要素のことである（図17）．一般にICNの数が少ないほど，他の要素の計算結果の効果を考慮しながら計算が実施されていることから，収束が良い．

2色に塗り分けるred-black ordering の場合，多くのICNを持つ（図18）．また，色数を増やして，4色にすると，図19に示すように，ICNの数は減少する．基本的に色数を増加させるとICNの数は減少する（非常に複雑な形状の場合などで例外はあるが）．

また，CM法（Cuthill-McKee）の場合は，図20に示すようにICNの数は1である．MC法では，各色における要素の独立性のみが考慮されているのに対して，CM法，RCM法では，各レベル（色）における要素の独立性とともに，各レベル（色）間の依存性についても考慮されており，前進後退代入における計算順序に適合した並び替えとなっている．

図21は，8,000要素（NX=NY=NZ=20）においてポアソン方程式をICCG法で解いた場合の，色数と収束までの反復回数，ICNの数の関係を示したものである〔9〕．ICCG（図17に対応した辞書式番号付けに基づくICCG法）とICCG/CM（CM法に基づくICCG法），ICCG/RCM（RCM法に基づくICCG法）の反復回数が，ほぼ同じでICCG/MC（MC法に基づくICCG法）と比較して早く収束しているのは，図17～図20で示したICNの数と反復回数の関係に対応している．また，色数の増加とともにICNの数が減少していることもわかる． ICCG/MCでは若干の例外はあるものの，全体的に色数の増加に従って，収束は改善されている．

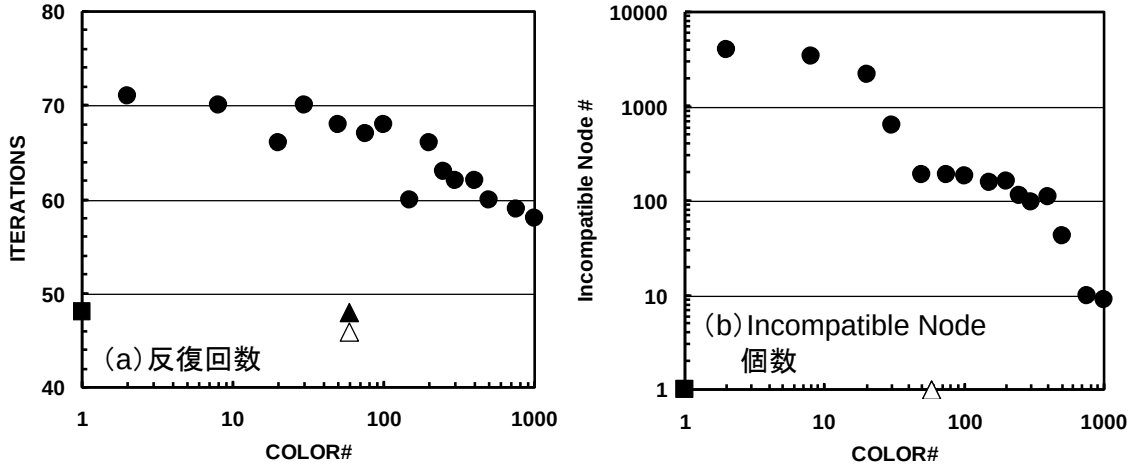


図21　三次元ポアソン方程式におけるICCG法の収束，Incompatible Nodeの数と色数の関係
（20^3=8,000要素，収束=10^{-8}）
（■：ICCG（辞書式番号付けに基づくICCG法），●：ICCG/MC，▲：ICCG/CM，△：ICCG/RCM）

CM-RCM 法も節点間の依存性を考慮した並べ替えになっているため，MC 法と比較すると収束性に優れている．GeoFEM における Block Jacobi 型局所前処理を採用しているため，並列プログラミングモデルによって反復回数に多少の違いはあるが，顕著では無い．色数を増やすと，図 7 に示すように，Hybrid 並列プログラミングモデルの場合には，OpenMP の同期オーバーヘッドが増すため性能が低下し反復回数が減少しても計算時間が増加する可能性がある．

図 22 は Flat MPI の場合の性能である．「図 22（a）Solver」は，SGS/CG 法の相対性能を収束までの計算時間で表したもので，SR11K，T2K（東大），XT4 ともに CM-RCM（10 色）のときの計算時間で無次元化してある（相対性能が 1.0 より大きいと CM-RCM（10 色）よりも性能が良いことを示す）．同様に「図 22（b）FLOPS」は，一回の反復計算あたりの性能を SR11K，T2K（東大），XT4 ともに CM-RCM（10 色）のときの計算時間で無次元化した値である．

色数が増加するとFLOPS値も増加するため，図16に示した反復回数の減少との相乗効果で，CM-RCM 法の場合は 30%以上 Solver 性能が上昇している（SR11K：1.31，T2K（東大）：1.34，XT4：1.42（表 3 参照））．色数が少ないと，1 つの色内の節点数が多いため，隣接点との節点番号との差が大きく，キャッシュからはずれやすいが，色数が増加するとこれがある程度解消され，色数が増加すると FLOPS 値が増加すると考えられる．SR11K，T2K（東大）ではこの増加が 10%程度であるが，XT4 の場合は 17%にも達している（表 3）．

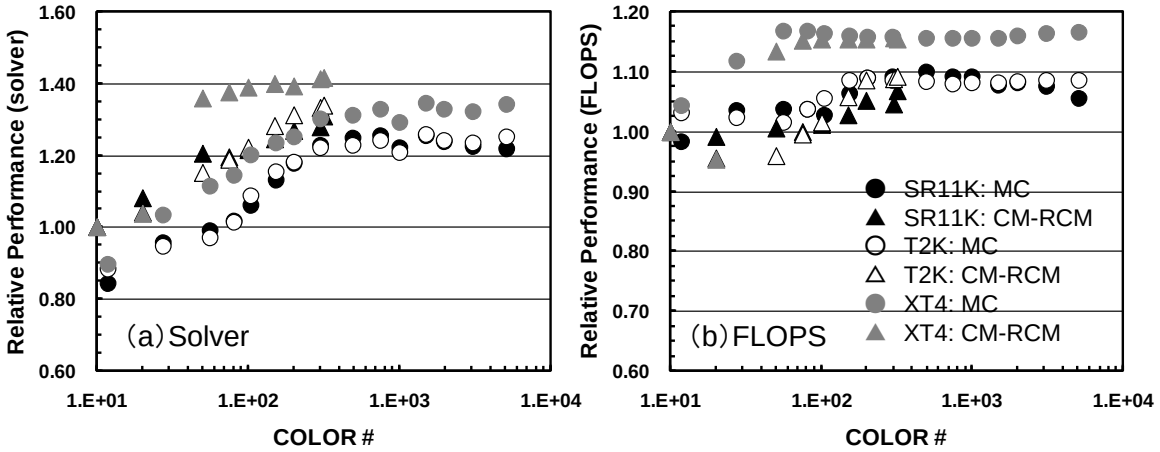


図 22　SGS／CG 法の相対計算性能（Flat MPI），CM-RCM（10 色）性能で無次元化（SR11K ●：MC，▲：CM-RCM），（T2K（東大）　○：MC，△：CM-RCM），（XT4 ●：MC，▲：CM-RCM）

表 3　各ケースにおける性能（CM-RCM，RCM における最適値）

		Hitachi SR11000/J2		T2K オープンスパコン（東大）		Cray XT4	
		CASE-1		CASE-1		CASE-1	
Flat MPI	Solver	1.31		1.34		1.42	
	FLOPS	1.07		1.09		1.17	
		CASE-2	CASE-3	CASE-2	CASE-3	CASE-2	CASE-3
HB 4×4	Solver	1.38	1.46	1.50	1.52	1.57	1.58
	FLOPS	.984	1.04	1.07	1.08	1.14	1.15
HB 8×2	Solver	1.33	1.36	1.36	1.46	-	-
	FLOPS	.974	.994	.999	1.07	-	-
HB 16×1	Solver	1.37	1.39	1.39	1.49	-	-
	FLOPS	.973	.990	.993	1.06	-	-

図 23 は HB 4×4 の相対計算性能である．図 22 と同様に Flat MPI／CM-RCM（10 色）で無次元してあるので，図 21 と直接比較することができる．特に MC 法では色数が 1,000 程度になると急速に FLOPS 値が低下し，反復回数が減少しているにもかかわらず計算時間が増加する．この傾向は SR11K の場合に顕著である．

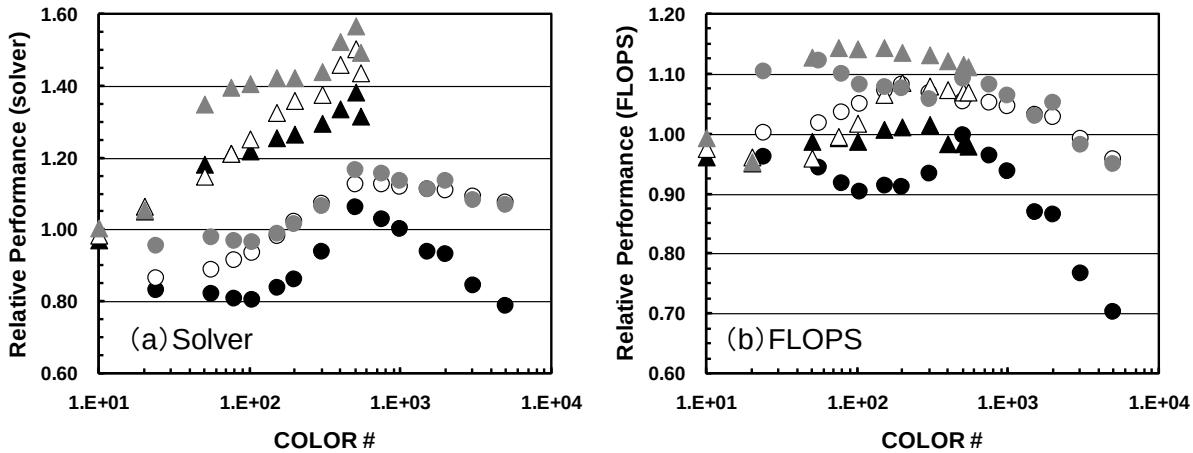


図 23　SGS／CG 法の相対計算性能（HB 4×4，CASE-2），Flat-MPI／CM-RCM（10 色）の性能で無次元化（SR11K ●：MC，▲：CM-RCM），（T2K（東大）　○：MC，△：CM-RCM），（XT4 ●：MC，▲：CM-RCM）

図 24 は，図 11 に示したデータ再配置を適用した場合（CASE-3）である．SR11K，T2K（東大），XT4 ともに CASE-2 より全体的に性能が上昇し，特に MC 法で色数が 1,000 を超えた場合の性能低下が抑制されている．T2K（東大）では色数を増加させても FLOPS 値がほぼ一様に留まっている. XT4 ではデータ再配置の効果はそれほど顕著ではない. CM-RCM 法における Solver 最高性能は，表 3 に示すように，SR11K : 1.38⇒1.46，T2K（東大）: 1.50⇒1.52，XT4 : 1.57⇒1.58 となり全体的に Flat MPI より良い．対応する FLOPS 性能については，SR11K：0.984⇒1.05，T2K（東大）: 1.07⇒1.08，XT4 : 1.14⇒1.15 となり，Flat MPI（SR11K : 1.07，T2K（東大）: 1.09，XT4：1.17）より若干低い値である（表 3 参照）.

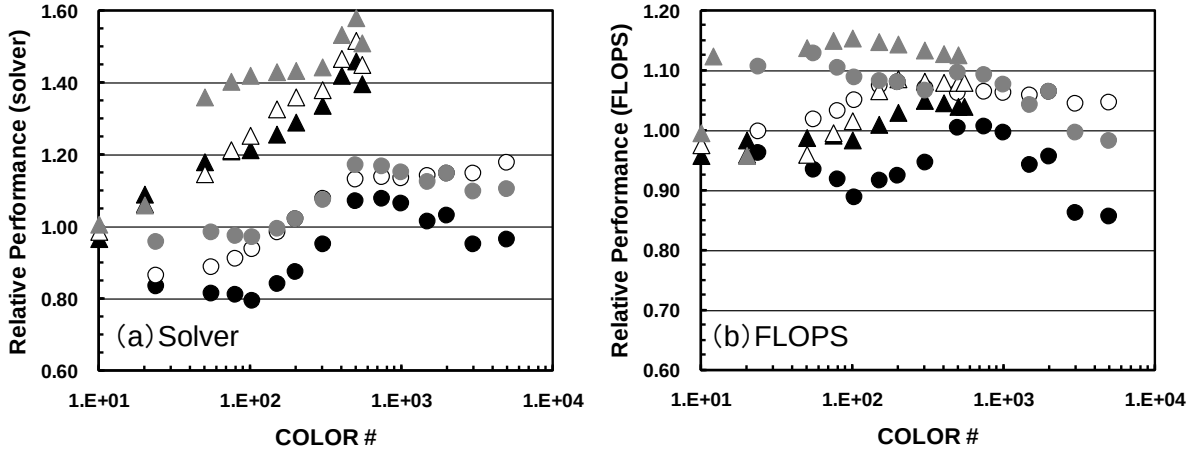


図 24　SGS／CG 法の相対計算性能（HB 4×4，CASE-3），Flat-MPI／CM-RCM（10 色）の性能で無次元化（SR11K ●：MC，▲：CM-RCM），（T2K（東大） ○：MC，△：CM-RCM），（XT4 ●：MC，▲：CM-RCM）

図 25，図 26 は HB 8×2，HB 16×1 について Solver 性能を比較したものである．MC 法で 1,000 色を超えた場合の CASE-3 の効果は，HB 4×4 のときほど顕著では無いものの，若干の改善が見られる．T2K（東大）では CASE-3 における性能上昇が 1,000 色以下の場合に特に顕著であり，CM-RCM，RCM については顕著な性能上昇が見られる．FLOPS 性能については図を省略したが，HB 4×4 の場合とほぼ同等である（表 3 参照）.

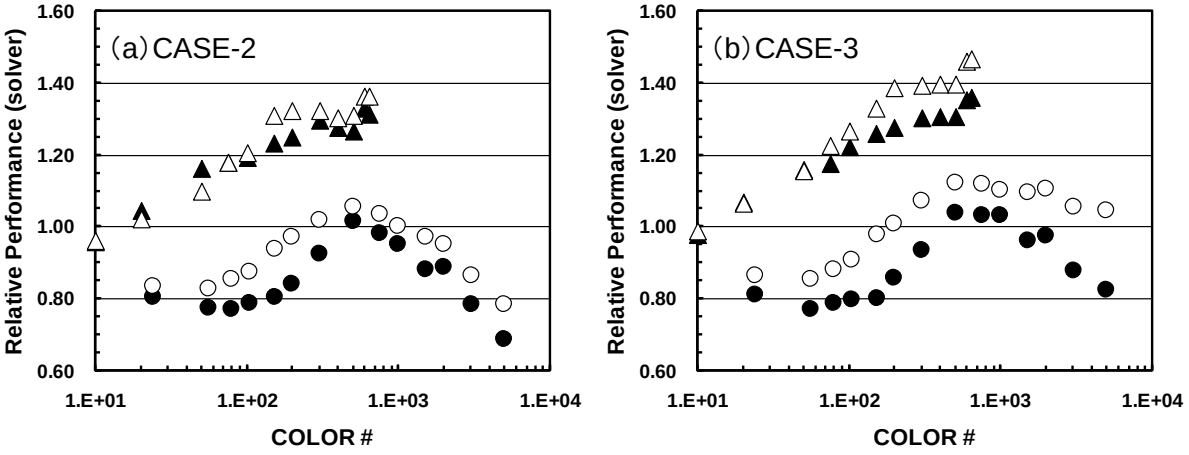


図 25　SGS／CG 法の相対計算性能（HB 8×2，CASE-2・CASE-3），Flat-MPI／CM-RCM（10 色）の性能で無次元化：（SR11K ●：MC，▲：CM-RCM），（T2K（東大） ○：MC，△：CM-RCM）

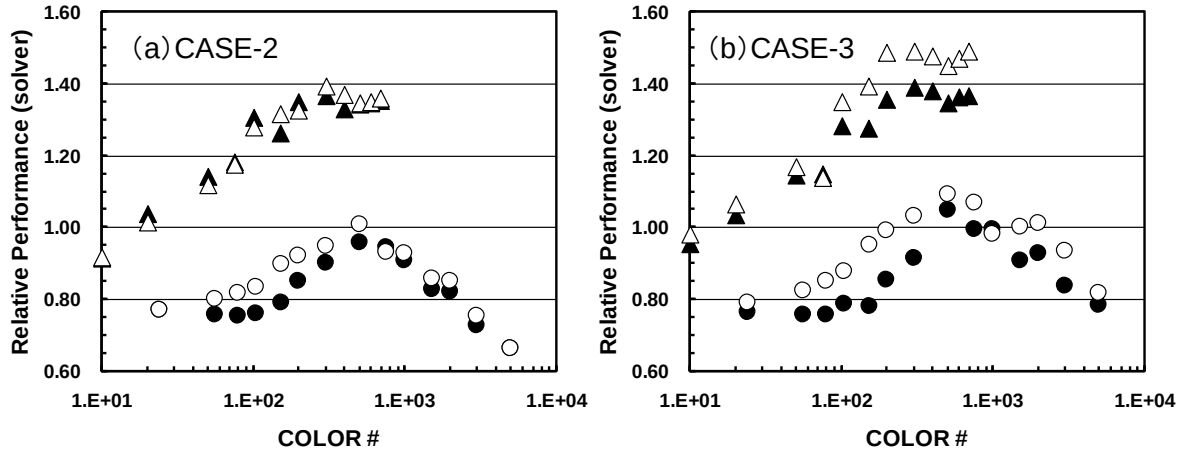


図 26　SGS／CG 法の相対計算性能（HB 16×1，CASE-2・CASE-3），Flat-MPI／CM-RCM（10 色）の性能で無次元化：（SR11K ●：MC，▲：CM-RCM），（T2K（東大） ○：MC，△：CM-RCM）

7. まとめ

不完全 LU/コレスキー分解等による疎行列向け前処理付反復法を OpenMP 等によりマルチコアソケット上で並列化するための手法として，MC 法，RCM 法，CM-RCM 法等による並び替えが広く使用されている．色数を増やすことで通常反復回数は減少するが，同期のオーバーヘッドによって性能が低下する．

本稿では，三次元弾性静解析問題を有限要素法で離散化して得られる対称正定な疎行列を SGS 前処理付 CG 法で解く場合の性能に対するリオーダリングの影響について，Hitachi SR11000/J2，T2K オープンスパコン（東大），Cray XT4 上で，Flat MPI，Hybrid 並列プログラミングモデルに対して評価した．First Touch Data Placement とスレッド上で番号が連続となるようなデータ再配置を組み合わせることによって，Hybrid 並列プログラミングモデルにおいて，全体的な性能は改善され，Flat MPI とほぼ同等であることが示された．データ再配置の影響は色数が多い場合に特に顕著に現れる．データ再配置は Hitachi SR11000/J2 においても有効である．リオーダリング法としては色数の多い CM-RCM 法，または RCM 法が有効である．RCM 法はマルチスレッドにおける負荷バランスの悪化が懸念されたが，本稿の範囲では性能に対する影響はほとんど無い．

今後はより悪条件の実用的問題についても検討を実施する予定であるが，そのような場合に色数の多い CM-RCM 法，RCM 法は効果的であると考えられる〔12〕．

次号では，ノード数を増加させた場合の性能評価について紹介する．

参 考 文 献

[1] 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
http://www.cc.u-tokyo.ac.jp/publication/news/VOL11/No1/200901tuning.pdf

[2] 中島研吾（2009）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，情報処理学会研究報告（HPC-119-18）103-108

[3] 中島研吾，片桐孝洋（2009）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）

[4] 中島研吾（2009）並列反復法と自動チューニング　ーマルチコア時代の並列プログラミングモデルー，特集：科学技術計算におけるソフトウェア自動チューニング「ソフトウェア自動チューニング技術の応用」，情報処理 50-6，517-522，情報処理学会

[5] Rabenseifner, R.（2002）Communication Bandwidth of Parallel Programming Models on Hybrid Architectures, Lecture Notes in Computer Science 2327, 437-448

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

[7] Kerbyson, D.J., Hoisie, A. and Wasserman, H.（2002）A Comparison between the Earth Simulator and Alpha Server Systems using Predictive Application Performance Models. LA-UR-02-5222, LANL

[8] Doi, S. and Washio, T.（1999）Using Multicolor Ordering with Many Colors to Strike a Better Balance between Parallelism and Convergence, RIKEN Symposium on Linear Algebra and its Applications, 19-26

[9] 中島研吾（2007）OpenMPによるプログラミング入門（II），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） 9-6
http://www.cc.u-tokyo.ac.jp/publication/news/VOL9/No6/200711OpenMP.pdf

[10] 青木秀貴，中村友洋，助川直伸，齋藤拡二，深川正一，中川八穂子，五百木伸洋（2005）スーパーテクニカルサーバーSR11000 モデルJ1のノードアーキテクチュアと性能評価, 情報処理学会論文誌：コンピューティングシステム 45-SIG12（ACS11），27-36

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

[12] Nakajima, K.（2007）Parallel Multistage Preconditioners based on a Hierarchical Graph Decomposition for SMP Cluster Architectures with a Hybrid Parallel Programming Model, Lecture Notes in Computer Science 4782, 384-395