Vol.2012-HPC-133 No.5
2012/3/26

大規模SMP並列スーパーコンピューター (HITACHI SR16000 モデル M1) の性能評価

大島聡史[†1] 實本英之[†1] 鴨志田良和[†1]
片桐孝洋[†1] 田浦健次朗[†1,†2] 中島研吾[†1]

本稿では東京大学情報基盤センターにおいて 2011 年 10 月に稼働を開始したスーパーコンピューターシステム HITACHI SR16000 モデル M1（愛称 Yayoi）の性能について報告する．本システムは計算ノードに Power7 プロセッサを搭載した最新のスーパーコンピューターシステムである．いくつかのベンチマークを用いて性能評価を行った結果，性能の特性や重要な実行時環境変数の設定などが明らかとなった．

Performance Evaluation of HITACHI SR16000 model M1 Supercomputer System

Satoshi OHSHIMA ,[†1] Hideyuki JITSUMOTO ,[†1]
Yoshikazu KAMOSHIDA ,[†1] Takahiro KATAGIRI ,[†1]
Kenjiro TAURA [†1,†2] and Kengo NAKAJIMA[†1]

We report the performance of HITACHI SR16000 model M1 supercomputer system (named Yayoi) which has started in October 2011 at Information Technology Center, The University of Tokyo. This is a latest supercomputer system which mounts Power7 CPU on the computation node. We executed several benchmarks on the system and unveiled characteristic features of performance and imporant parameters.

†1 東京大学 情報基盤センター
Information Technology Center, The University of Tokyo
†2 東京大学 大学院 情報理工学系研究科
Graduate School of Information Science and Technology, The University of Tokyo

1. はじめに

東京大学情報基盤センター（以下，当センター）では 2011 年に既設のスーパーコンピュータシステム SR11000/J2 システム（以下 SR11000/J2）の使用期限を迎えた．後継のシステムとしては，従来の SR11000/J2 システム利用者の継続性を重視した「大規模 SMP 並列スーパーコンピューターシステム」と大規模超並列計算向けの「大規模超並列スーパーコンピューターシステム」の 2 システムを導入することとなった．前者については日立製作所社製の SR16000 モデル M1 システム（愛称 Yayoi，以下 SR16000/M1）に決定し，2011 年 10 月に運転を開始，11 月より正式サービスを開始している[1)]．また後者については富士通社製の PRIMEHPC FX10 システム（愛称 Oakleaf-FX，以下 FX10）に決定し，2012 年 4 月の運用開始に向けて準備が進められている[2)]．これら 2 システムはそれぞれ異なる最新のアーキテクチャを採用した計算機システムである．

本稿では既に正式サービスを開始している SR16000/M1 の性能について，実運用環境でのベンチマーク結果を用いて報告する．今回用いたベンチマークプログラムは以下の 6 種類である．

(1) STREAM ベンチマーク
(2) HPL ベンチマーク
(3) MPIFFT ベンチマーク
(4) GeoFEM ベンチマーク
(5) MDTEST ベンチマーク
(6) IOR ベンチマーク

本稿の構成は以下の通りである．2 章では SR16000/M1 のハードウェア構成について述べる．3 章では SR16000/M1 上で様々なベンチマークプログラムを実行した結果を実行時に意味のあったコンパイラオプションなどの情報とともに報告する．4 章はまとめの章とする．

2. ハードウェア構成

2.1 全体構成

本章では SR16000/M1 の主なハードウェア構成について述べる．

はじめに SR16000/M1 の全体構成について述べる．表 1 に SR16000/M1 の性能諸元を旧システムである SR11000/J2 とあわせて示す．SR16000/M1 は 56 の計算ノード，556TB

のストレージ，高速な内部ネットワーク，そしてログインノードや各種管理用ノードから構成される計算機システムである（図 1）．特に計算ノードとログインノードは高密度に搭載されており，計算ノード 56 ノードとログインノード 2 ノードが水冷 2 ラックに収められている．

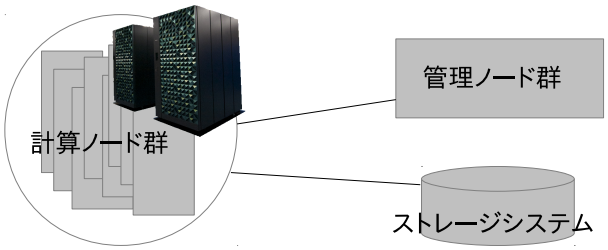


図 1　SR16000/M1 の全体構成

表 1　SR16000/M1 と SR11000/J2 の性能諸元

	SR16000/M1	SR11000/J2（旧システム）
CPU	Power7 3.83 GHz	Power5+ 2.30 GHz
ノード数	56	128
コア数/計算ノード	32	16
理論演算性能/コア	30.64 GFLOPS	9.2 GFLOPS
理論演算性能/計算ノード	980.48 GFLOPS	147.2 GFLOPS
理論演算性能/全計算ノード	54906.88 GFLOPS	18841.6 GFLOPS
主記憶容量/計算ノード	200 GByte (170GByte 使用可能)	128 GByte (112GByte 使用可能)
主記憶容量/全計算ノード	11200 GByte	16384 GByte
Byte/FLOPS 値	0.52	1.39
SMT 機能	最大 4 スレッド/コア （最大 2 スレッド/コアにて運用）	非対応
計算ノード間 ネットワーク構成	階層型完全結合	3 段クロスバー
計算ノード間転送性能	96GByte/sec（単方向）× 双方向	12GByte/sec（単方向）× 双方向
ストレージ容量	556 TByte	94.2 TByte
LINPACK 性能値	0.8075 TFLOPS （1 計算ノード） 6.46 TFLOPS （8 計算ノード）	15.81 TFLOPS （全計算ノード）

2.2　CPU

SR16000/M1 は計算ノードの CPU として Power7 を搭載している．

Power7 と計算ノード群の構成を図 2 に示す．Power7 は 8 つのコアによって構成されているマルチコアプロセッサであり，1 ノードには Power7 CPU が 4 基（4 ソケット）搭載されている．ノードを構成する 4CPU は Multi Chip Module(MCM) とも呼ばれる．ラックへの搭載については，8 ノードからなるドロワ（一畳ほどの大きさがある）を 1 単位として行われている．

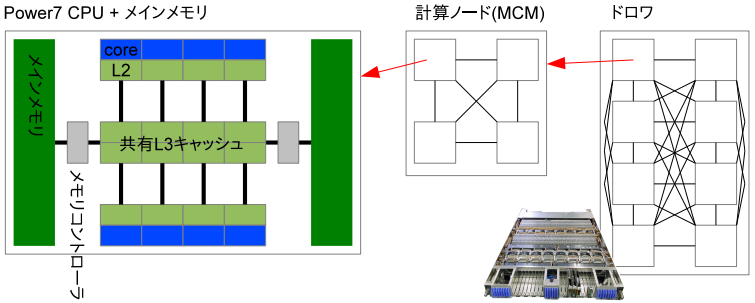


図 2　Power7 CPU と計算ノード群の構成

SR16000/M1 に搭載されている Power7 の動作周波数は 3.83GHz であり，理論倍精度浮動小数点演算性能は以下の通りである：

- 1 コアあたり (乗算 2FLOP+加算 2FLOP)×2 演算器 ×3.83GHz=30.64GFLOPS
- 1CPU あたり 30.64GFLOPS×8 コア=245.12GFLOPS
- 1 ノードあたり 245.12GFLOPS×4CPU=980.48GFLOPS
- 計算ノード群全体 980.48GFLOPS×56 ノード=54.90688TFLOPS

またキャッシュについては，L1 キャッシュをデータと命令それぞれにコアごとに 32KB，L2 キャッシュをコアごとに 256KB，L3 キャッシュを 1CPU ごとに 32MB 搭載しており，L1 キャッシュはパリティ，L2 および L3 キャッシュは ECC によって保護されている．

さらに Power7 は SMT(Simultaneous Multi Threading) に対応しているため，状況に応じて 1 コアあたり最大 4 スレッドを同時に処理することが可能である．ただし SMT は同時実行スレッド数を増やすほど高い性能が得られるとは限らないため，本システムでは最大

同時実行スレッド数を 2 に設定して運用している．

旧機種 (SR11000/J2) も Power5+を搭載していたため基本アーキテクチャは同一であるが，Power5+から Power7 にかけて様々な機能･性能の追加や向上が行われている．SR16000/M1 の演算性能を SR11000/J2 と比較すると，ノードあたりでは約 6.7 倍，計算ノード群全体では約 2.9 倍に向上している．

2.3 メ モ リ

Power7 CPU とメインメモリ（主記憶）の接続については、図 2 に示したように各 CPU に搭載されたメモリコントローラを介して接続されている．そのため CPU コアがメモリアクセスを行う際に，対象のメモリがどこに存在するか（ローカルな CPU に接続されたメモリなのか他の CPU に接続されたメモリなのか）によってメモリアクセス性能に差が生じる．いわゆる NUMA(Non-Uniform Memory Access) アーキテクチャである．NUMA 環境下におけるメモリアクセスの最適化は性能に大きな影響を及ぼすため，SR16000/M1 におけるプログラム最適化・性能チューニングの際には注意が必要がある．

SR16000/M1 は計算ノード 1 ノードあたり 200GByte，計算ノード群全体では 11200GByte のメインメモリを搭載している．ただしメモリの一部をシステムが占有するため，実際に利用者が使えるメインメモリ容量は 1 ノードあたり 170GByte となる．またメインメモリの種別については，DDR3 SDRAM メモリが搭載されている．計算ノード上の各 CPU コアとメインメモリ間の物理転送性能の合計値は 512GByte/秒である．1 ノードあたりの演算 FLOPS 値あたりメモリ性能 Byte/s 値（B/F 値）については，0.52（SMT について考慮しない場合）である．

SR16000/M1 のメモリ性能値を SR11000/J2 と比較すると，メモリ容量については計算ノード 1 ノードあたりでは増加している一方で計算ノード群全体では減少している．転送性能（計算ノード 1 ノードあたりの CPU-メモリ間の物理転送性能の合計値）については大きく向上しているものの，B/F 値は減少していることから，SR16000/M1 は SR11000/J2 と比べると計算インテンシブなアプリケーションに適したシステムであると考えることができる．なお，SR16000/M1 の B/F 値は SR11000/J2 と比べると確かに低い値であるが，HA8000 クラスタシステム（T2K 東大，主要な計算ノードの B/F 値が 0.28 である）[3] などと比べると高い値である．

2.4 計算ノード間ネットワーク

SR16000/M1 のネットワーク構成は，1 ドロワ内の 8 ノードが完全結合であり，さらにドロワ単位でも完全結合である階層型の完全結合である．そのため計算ノード群から任意の複数ノードを抽出して通信を行うと，ノードの組み合わせによっては完全結合ではない組み合わせとなり，理論上は通信性能が低下する可能性がある．

計算ノード間ネットワークの性能は 96GByte/秒 (単方向)× 双方向である．

2.5 共有ストレージ

SR16000/M1 では，全ての計算ノードおよびログインノードからアクセスしてファイルを共有可能なストレージとして General Parallel File System for AIX(以下，GPFS) による共有ファイルシステムを提供している．I/O サーバは 4 台で構成されており，ファイル入出力の際はデータを一定のブロックに分割して各ノードから並列にディスクアクセスすることで高い I/O スループット性能を実現する．全てのサーバが全てのディスクにアクセス可能であり，独立したメタ・データサーバが不要であるため，一台の GPFS サーバがダウンした場合も他の GPFS サーバによりサービスを引き継ぐことが可能である．ディスクアレイ装置は Hitachi AMS2500 16 台 (32 コントローラ) を使用している．各コントローラは 2G バイトのキャッシュを搭載しており，8Gbps FC ケーブル 4 本を接続している．ディスクには 600G バイト，15krpm の 3.5 インチ SAS ディスクを使用しており，139 個の 7D+2P の RAID6 グループから構成されている．フォーマット後のユーザーが利用可能な容量として 500T バイトの容量を提供している．

3. ベンチマークによる性能評価

3.1 STREAM ベンチマーク

STREAM ベンチマーク[4] を用いて計算ノードのメモリ性能を測定した．STREAM ベンチマークは配列に対して “特定の処理” を繰り返し実行した際の実行時間からメモリ性能（MB/s）を算出する．“特定の処理” としては，

- 配列のコピーを行う Copy（c[j] = a[j]）
- 配列とスカラーとの乗算を行う Scale（b[j] = scalar*c[j]）
- 二つの配列を加算する Add（c[j] = a[j]+b[j]）
- スカラーとの乗算と配列加算を組み合わせた Triad（a[j] = b[j]+scalar*c[j]）

が用意されている．計算内容によってメモリのロード回数とストア回数に違いがあるため，メモリアーキテクチャとの相性により性能差が生じる．

今回は使用するノードを計算ノード 1 ノードのみとして，使用するスレッド数や環境変数を変更して性能の測定を行った．プログラムの作成には xlc(IBM XL C/C++ Enterprise Edition for AIX V11.1) を使用し，並列化については OpenMP を用いた．コンパイルオプ

ションは “-q64 -O3 -qsmp=omp -qreport” を指定した．問題サイズ (N) は 300,000,000，計算繰り返し回数 (NTIMES) は初期値 (10) とした．実行結果を図 3 に示す．

はじめに環境変数として使用するスレッド数のみを設定してベンチマークを実行したところ，図 3(a) の結果が得られた．使用スレッド数が増加するとそれにともない性能も向上するが，物理スレッド数を超えて SMT 実行となる 64 スレッドでは性能向上が得られなかった．また Copy と Scale に比べて Add と Triad の方が高い性能が得られた．

つづいて，各スレッドがそれぞれ最も近い位置にあるメモリを利用できるようにメモリの割り当て (AFFINITY) の設定を行った場合の性能を測定した．既に述べたように Power7 は NUMA アーキテクチャであるため CPU コアとメモリとの性能が一定ではなく，何も設定を行わない状態では CPU コアが一番近いメモリ以外のメモリを参照し性能が低下する可能性がある．そこで，環境変数 MEMORY_AFFINITY に MCM を設定してプログラムを実行すると，各 CPU コアが常に一番近いメモリを参照するようになるためプログラムの性能が向上する可能性がある．本設定を行った場合の実行結果が図 3(b) である．(a) と比較して最大性能値が向上していることがわかる．

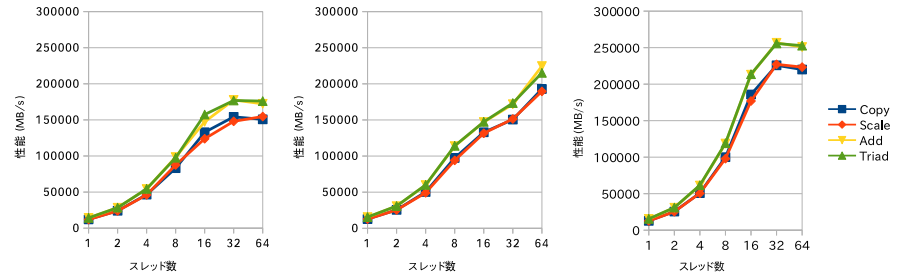
さらに (b) に加えて環境変数 XLSMPOPTS に startproc=0 および stride=(64÷ スレッド数) を設定した場合の実行結果を図 3(c) に示す．本設定によりいずれの処理についても最大性能が向上し，それぞれ 32 スレッド実行時に以下の最大性能が得られた．

- Copy 224825.3361
- Scale 226349.5329
- Add 256364.6680
- Triad 255192.6583

startproc はスレッドを割り当てる CPU コア番号の先頭番号を指示する値であり，stride は CPU コア割り当て時の間隔を指示する値である．今回の設定では CPU コアが計算ノード全体に分散するような配置になり，特定の CPU コア-メモリ間に負荷が集中するのを回避できるため良い性能が得られている．

なお，Copy や Scale に比べて Add や Triad の性能が高いことおよび理論上の性能である 512GByte/sec に比べて全ての値が低い（およそ 50%以下）ことについては以下の影響が考えられる．

Power7 のメモリ構成と性能について詳しく見てみると，Power7 には 1CPU（8 コア）あたり 2 つのメモリコントローラ（MC）が搭載されており，メモリコントローラあたり Load8Byte+Store4Byte の 4 チャンネル構成となっている．メモリの動作周波数は 1333MHz であるため，1CPU あたり最大で (Load8Byte+Store4Byte)×4channel×2MC×1.333= 128GByte/sec となる．1 計算ノードあたりでは 4CPU（4 ソケット）搭載されているため，128×4=512GByte/sec の性能となる．以上から，Load8Byte と Store4Byte の比に対応したアクセス，すなわちロード 2 に対してストア 1 の割合でメモリアクセスを行った場合に最大性能が得られることとなる．STREAM ベンチマークで行っている各処理はロードとストアの比が Power7 にとって最適ではない比となっているために低めの性能が出ていると考えられる．そこで試験的に STREAM のプログラムを改編しベクトルの積和計算（Daxpy 相当の計算，a[j] = a[j]+scalar*b[j]）を行わせてみたところ，338758.7643 という性能値が得られた．理論上の最大性能には及ばないものの，メモリの構成に適した処理を行うことで他の処理より高い性能（最大性能比約 66%）を得ることができた．



(a) 使用スレッド数の設定のみ行った場合（OMP_NUM_THREAD）
(b) メモリ割り付けの設定を加えた場合 (+ MEMORY_AFFINITY)
(c) さらに詳細なスレッド割り当ての指定を加えた場合 (+ XLSMPOPTS)

図 3 STREAM ベンチマークの結果

3.2 HPL ベンチマーク

HPCC ベンチマーク (HPCC 1.4.0)[5] に含まれる HPL の性能を測定した．このベンチマークは LU 分解による連立一次方程式の求解を行うものであり，特に行列-行列積計算（BLAS3 DGEMM）の性能がベンチマークスコアに大きな影響を与えるベンチマークである．

プログラムの作成には xlc(IBM XL C/C++ Enterprise Edition for AIX V11.1) を使用し，BLAS ライブラリは ESSL ライブラリに含まれるものを使用した．主なコンパイルオプションとして “-O5 -qarch=pwr7 -qtune=pwr7 -qmaxmem=-1 -qreport” を指定した．実行環境としては，SR16000/M1 は全系でのプログラム実行を想定していないことから，計

算ノード 1 ノードおよび 8 ノードを用いた．いずれも 1 ノードあたり 32CPU コアを使用し，各計算ノードにおける MPI プロセス数を 32，プロセスあたりスレッド数を 1（フラット MPI）として実行した．使用した全プロセス数は 32 プロセス ×1 ノード=32 プロセスおよび 32 プロセス ×8 ノード=256 プロセスとなる．実行時の主な設定としては，各 CPU コアが近くのメモリを利用するように MEMORY_AFFINITY 環境変数に MCM を設定した．また主な問題設定（hpccinf.txt に指定する値）としては以下の値を用いた．

- 1 ノード実行
  - Ns = 107520, NBs = 160, Ps = 4, Qs = 8
- 8 ノード実行
  - Ns = 302080, NBs = 160, Ps = 8, Qs = 32

実測性能および理論演算性能に対する性能割合は以下の通りとなった（小数点第 3 位以下切り捨て）:

- 1 ノード 0.83 TFLOPS, 84.65%
- 8 ノード 6.38 TFLOPS, 81.33%

3.3 MPIFFT ベンチマーク

HPCC ベンチマーク (HPCC 1.4.0) に含まれる FFT の性能を測定した．このベンチマークは一次元の高速フーリエ変換を行うものであり，全対全通信（MPI_Alltoall）の性能がベンチマークスコアに大きな影響を与えるベンチマークである．

プログラムの作成には xlc(IBM XL C/C++ Enterprise Edition for AIX V11.1) を使用した．HPL と同様に主なコンパイルオプションとして “-O5 -qarch=pwr7 -qtune=pwr7 -qmaxmem=-1 -qreport” を指定した．実行環境としては計算ノード 8 ノードを使用し，計算ノード 1 ノードあたり MPI プロセス数を 32，プロセスあたりスレッド数を 1（フラット MPI）として実行した．HPL の 8 ノード実行と同様，合計プロセス数は 256 となる．主な問題設定（hpccinf.txt の設定値）としては以下の値を用いた．

- Ns = 320000, NBs = 80, Ps = 16, Qs = 16

実測性能としては

- 151.121 GFLOPS

の性能値が得られた．この際，各コアが近くのメモリを利用するように MEMORY_AFFINITY 環境変数に MCM を設定した．また MP_S_IGNORE_COMMON_TASKS 環境変数を yes に設定することが性能の向上をもたらした．この環境変数はタスク配置が MPI コレクティブ通信に適しているかどうかを調査可能とするという役割を持ち，Yes に設定しておくことで MPI コレクティブ通信の性能が向上する場合があるとされている．

3.4 GeoFEM ベンチマーク

3.4.1 概　　要

GeoFEM プロジェクト[6] 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[7] による評価を実施した．オリジナルの GeoFEM ベンチマーク[8] 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．

著者らは参考文献 8) において，3 種類の GeoFEM ベンチマークのうち図 4 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について cc-NUMA アーキテクチャを有する HA8000 に対して様々な最適化を試みた．この成果を性能評価用のベンチマークプログラムとして整備したものが GeoFEM-Cube である．

GeoFEM-Cube では，係数行列が対称正定な疎行列となることから，SGS(Symmetric Gauss-Seidel)[8] 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）．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．

連立一次方程式の係数マトリクスの格納法としてオリジナルの GeoFEM ベンチマークでは

(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，GeoFEM-Cube ではスカラープロセッサ向けの CRS 法を使用している．

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

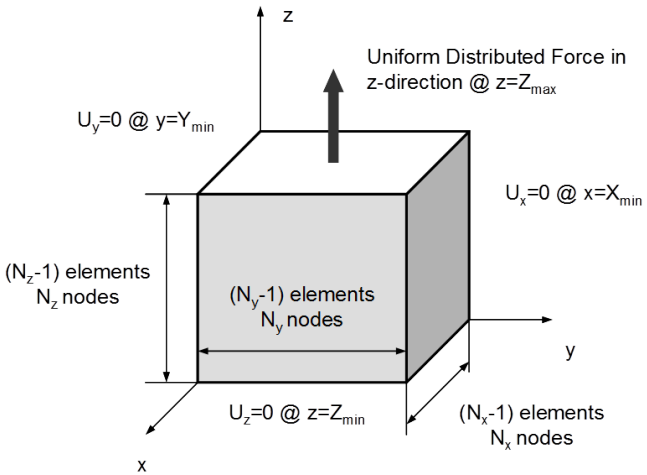


図 4　Cube 型ベンチマークの境界条件

施しないが，前処理における前進後退代入はグローバルなデータ依存性を有するプロセスのため，並列性を抽出するためのリオーダリングが必要である[8]．GeoFEM ベンチマークでは，マルチカラー法（Multicoloring，MC）法，Reverse Cuthill-McKee（RCM）法，更に RCM 法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）の 3 種類が利用可能となっている．

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扱うことができる．Hybrid については「Hybrid a×b（HB a×b）」（a：MPI プロセス当りの OpenMP スレッド数，b：ノード内 MPI プロセス数）という形で，ノード構成に応じてスレッド数，MPI プロセス数を自由に決められるようになっている．

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．SR16000/M1 は，HA8000 と同様に NUMA（Non Uniform Memory Access）アーキテクチャを有しており，実行時制御コマンド（NUMA control）を使用して，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広く知られている[8]．HA8000 では NUMA control を陽に指定する必要があったが，SR16000/M1 では

- ローカルメモリにデータを確保するための環境変数（MEMORY_AFFINITY=MCM）
- CPU を固定的に割り当てるための実行時コマンド（mpibind）

が準備されている．

GeoFEM-Cube では，Hybrid 並列プログラミングモデルにおける性能改善のため，

（1）First Touch Data Placement の適用[9]
（2）連続データアクセス，キャッシュヒット率向上のためのデータ再配置（Sequential リオーダリング，図 5）

を適用している．各最適化手法の詳細については参考文献 8) を参照されたい．

本稿では以下の 3 ケースについて評価を実施した：

- CASE-1：MC，RCM，CM-RCM によるリオーダリングを適用
- CASE-2：更に First Touch Data Placement を適用（Flat MPI は除く）
- CASE-3：更に図 5 に示すデータ再配置を適用（Flat MPI は除く）

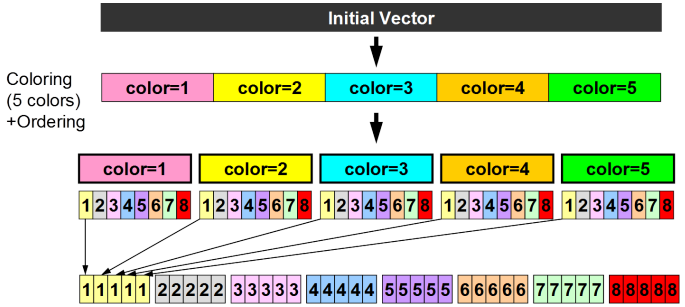


図 5　連続データアクセスのためのデータ再配置（Sequential Reordering）（5 色，8 スレッドの場合）

3.4.2　性能評価結果

図 6 に 2,097,152 節点（6,291,456 自由度），CM-RCM（色数 10），最適化 CASE-3 における計算結果を示す．1 ノード（4 ソケット，32 コア）を使用，スレッド数 × コア数=32 であり SMT は適用していない．Flat MPI の他，HB 2×16，4×8，8×4，16×2，32×1（ノード内 OpenMP のみ）を実施した．図 6 で示したのは SGS/CG 法部分の計算性能（GFLOPS 値）である．収束までの反復回数は各並列プログラミングモデルによって若干異なるが数%以内であり，この計算性能が実際の計算時間に直接反映される．

Flat MPI の性能が最も良く，以下，OpenMP のスレッド数を増加するに従って性能が低下し，HB 32×1 では約 50%にまで低下している．図 7 は，同じ問題に対する SR11000/J2,

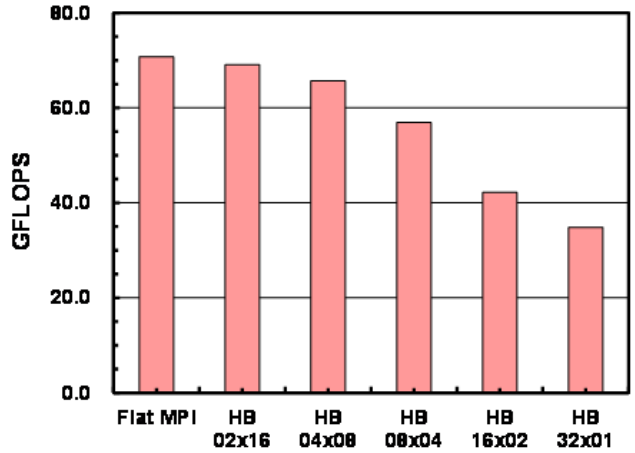


図 6　GeoFEM-Cube 性能評価結果（SGS/CG 法の性能（GFLOPS 値）），SR16000/M1 1 ノード（4 ソケット，32 コア），2,097,152 節点（$=128^3$，6,291,456 自由度），CM-RCM（色数 10），最適化：CASE-3

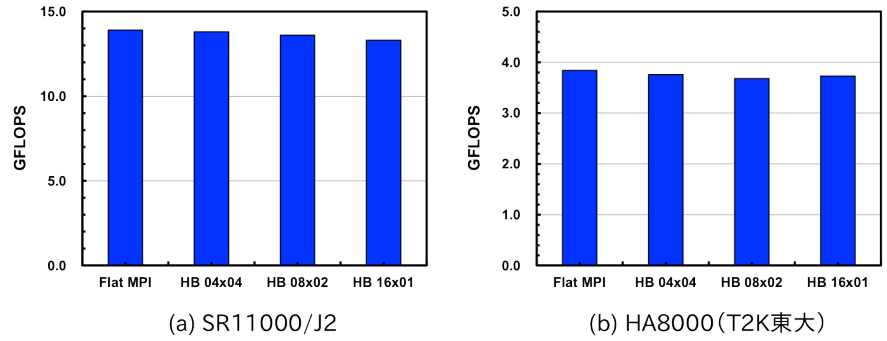


図 7　GeoFEM-Cube 性能評価結果（SGS/CG 法の性能（GFLOPS 値）），2,097,152 節点（$=128^3$，6,291,456 自由度），CM-RCM（色数 10），最適化：CASE-3,（a）SR11000/J2,（b）HA8000（T2K 東大）

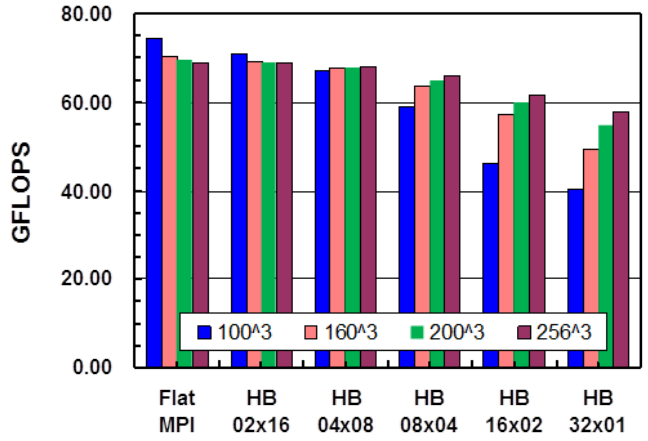


図 8　GeoFEM-Cube 性能評価結果（SGS/CG 法の性能（GFLOPS 値）），SR16000/M1 1 ノード（4 ソケット，32 コア），節点数：100^3（1,000,000 節点，3,000,000 自由度）-256^3（16,777,216 節点，50,331,648 自由度），RCM，最適化：CASE-3

HA8000 における CASE-3 の例である[8]．いずれの場合も Flat MPI と他の手法はほぼ同じ性能を示しており，ノード数を増加させると全般的に Hybrid 並列プログラミングモデルの方が全体的に性能が良いことも既に示されている[10]．

図 8 は SR16000/M1 において問題サイズ（節点数）を 100^3（1,000,000 節点，3,000,000 自由度）から 256^3（16,777,216 節点，50,331,648 自由度）まで変化させた場合の性能である．リオーダリングは RCM 法によっている．問題サイズを大きくすることによって，全体的な性能は低下するものの，OpenMP のオーバーヘッドが隠蔽されるため Flat MPI と Hybrid 並列プログラミングモデルの差は少なくなり，節点数 256^3 では，Flat MPI に対して HB 32×1 の性能は約 84%である．

図 9 は前項で示した最適化手法（CASE-1〜CASE-3）の効果を各並列プログラミングモデルについて示したものである．特に MPI プロセス当りのスレッド数が多い HB 16×2，32×1 においては特に CASE-3 による最適化の効果が顕著であることがわかる．

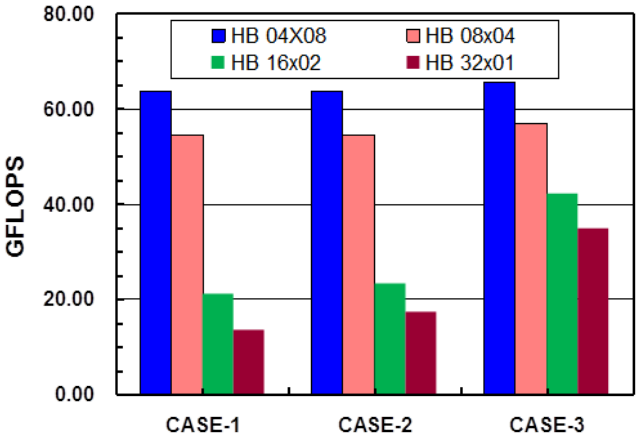


図 9　GeoFEM-Cube 性能評価結果（SGS/CG 法の性能（GFLOPS 値）），最適化手法（CASE-1〜3）の効果，SR16000/M1 1 ノード（4 ソケット，32 コア），2,097,152 節点（$= 128^3$，6,291,456 自由度），CM-RCM（色数 10）

3.5　MDTEST ベンチマーク

共有ファイルシステム上で MDTEST ベンチマークを実行し，性能評価を行った．MDTEST ベンチマークは Lawrence Livermore National Laboratory の Livermore Computing Center が公開している I/O ベンチマーク[11] であり，メタデータアクセス性能を計測するものである．

MDTEST では多数のプロセスが一斉に共有ファイルシステムにアクセスし，一定の処理を行う時間から共有ファイルシステムのメタデータアクセス性能を測定する．今回の性能評価では以下の条件で計測を行った．

- ファイル・ディレクトリの作成・削除の速度を測定
- プロセスごとに上記の操作を 3000 回ずつ実行
- プロセスごとに個別の作業ディレクトリを作成して処理を実行
- 5 回の測定を行い，平均値からアクセス速度を計算

図 10 はプロセス数を 8 に固定して MDTEST を実行した結果である．横軸には使用したノード数とノードあたりプロセス数の組み合わせを，縦軸にはアクセス速度 (Operations per second) を，ファイル作成等の各操作の実行回数を全プロセスで合計した数を 1 秒あたりの値に正規化して示している．

8 プロセスの実行の場合，すべての操作について 5000 回/秒以上の速度であり，操作によって結果のばらつきはあるが，ディレクトリ作成以外の操作についてはノード数が多いほど高い性能となることが分かった．

一方，図 11 はノード数を 1 に固定してノード内で起動するプロセス数を変えて MDTEST を実行した結果である．こちらはプロセス数に関わらず似たような速度となった．64 プロセスの場合は他の場合の平均と比較して 10%から 30% 低い性能となっているが，これはノードの物理コア数 32 より多いプロセスを起動していることが影響していると推測される．

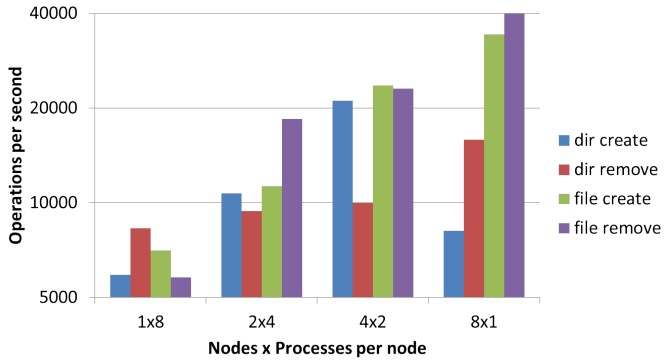


図 10　mdtest の実行結果（8 プロセス）

3.6　IOR ベンチマーク

共有ファイルシステム上で IOR ベンチマークを実行し，性能評価を行った．IOR ベンチマークは MDTEST と同様に Lawrence Livermore National Laboratory の Livermore Computing Center が公開している I/O ベンチマークであり，ブロック入出力のスループットを計測するものである．

IOR では多数のプロセスが一斉に共有ファイルシステム上のファイルを読み書きし，データ転送性能を測定する．使用するファイルのプロセスへの割り当てについては，プロセスごとに別のファイルを割り当てるか，単一ファイル内でプロセスごとに別々の領域に割り当て

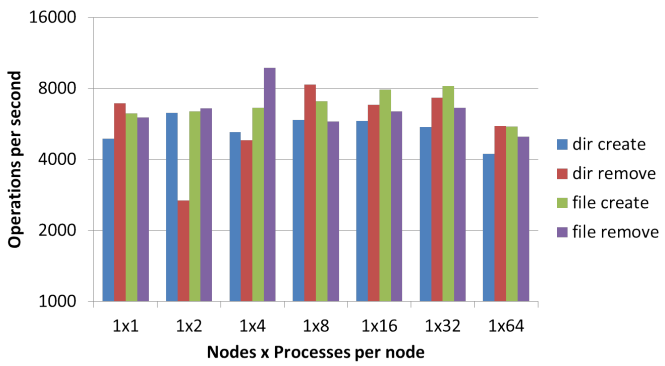


図 11 mdtest の実行結果（1 ノード）

るかを選択することが可能である．以下では前者を ior-multi，後者を ior-single と呼ぶことにする．今回の性能評価では，両者について以下の条件で計測を行った．

- POSIX I/O を使用
- ファイルの書き込みの性能を測定
- プロセスごとに 16GB のファイルを出力

実行結果を図 12 から図 14 までのグラフに示す．各グラフの縦軸にはデータ転送性能 (Write (MB/sec)) として，全プロセスが生成するファイルサイズの合計を，全プロセスがファイル書き込みを完了するまでにかかる時間で割ったものを示している．

図 12 と図 13 は，ior-multi，ior-single それぞれについて，書き込みのブロックサイズを 1M バイトとしてノード数とノードあたりのプロセス数を変化させて計測した結果である．メタデータアクセスの競合などがあるため，ior-single の性能は ior-multi の性能と比較して若干低くなる傾向にある．また，ノード数を 4 ノードから 8 ノードに増加させた場合の性能向上率は低いものの，ノード数を増加させると性能は向上する傾向にある．一方，同じノード数の場合の性能を比較すると，1 ノード，2 ノード，ior-multi の 4 ノードの実験では，ノードあたりのプロセス数を変化させても性能に大きな差が出なかった．これは，GPFS がノードごとに I/O 帯域幅の制御を行っているためであると考えられる．ノード数およびプロセス数を増加させた場合に性能にばらつきがあるのは，他のジョブ等の影響があると思われる．

図 14 は 1 ノードでの ior-multi 実行を，書き込みのブロックサイズを変化させて計測した結果である．ノード内のプロセス数は 8 または 16 の場合に最も高い性能となり，それ以上増やしても性能はよくならないことが分かる．また，ブロックサイズを 4K バイトから 64K バイトに変化させるとデータ転送性能の向上が見られるが，64K バイト以上に増加させても性能はほぼ横ばいになっている．

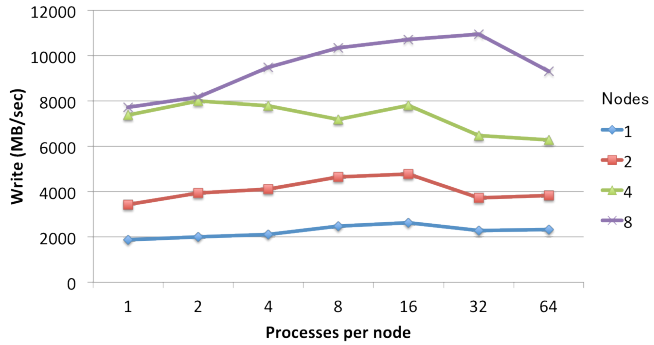


図 12 ior-multi の実行結果

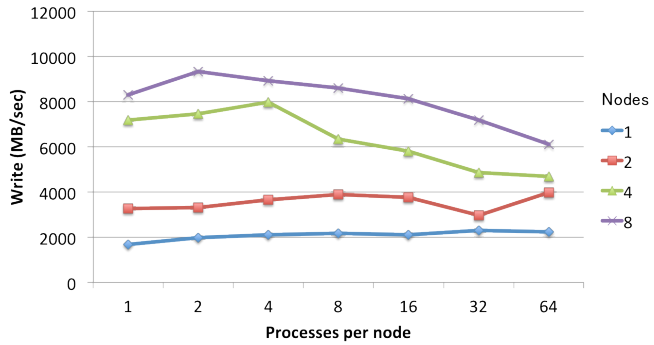


図 13 ior-single の実行結果

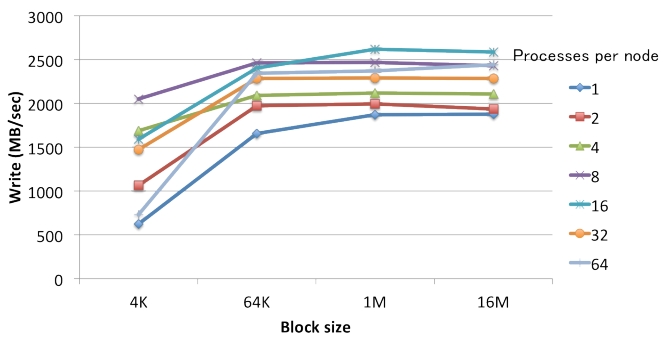


図 14　ior-multi の実行結果（1 ノード）

4. おわりに

本稿では SR16000/M1 の性能について実運用環境におけるベンチマーク測定結果を用いて評価した．ベンチマークを通じてシステムの性能のみならず，Hybrid 並列化における最適化の効果や，性能に影響を与える環境変数の設定などが明らかとなった．

当センターでは SR16000/M1 に引き続き FX10 の導入も控えている．FX10 導入後は SR16000/M1 と FX10 と HA8000 を用いて性能や最適化手法の比較を行う予定である．またその結果を基に実行環境の差を吸収・隠蔽するソフトウェアの開発や実行環境の構築等にも取り組む予定である．

謝辞　システムの導入・実験にあたっては株式会社 日立製作所，東京大学情報基盤センターの皆様にご協力いただきました．

参　考　文　献

1) SR16000 システム（SMP）（Yayoi），東京大学情報基盤センター http://www.cc.u-tokyo.ac.jp/system/smp/.
2) FX10 スーパーコンピュータシステム（Oakleaf-FX），東京大学情報基盤センター http://www.cc.u-tokyo.ac.jp/system/fx10/.
3) HA8000 クラスタシステム（T2K 東大），東京大学情報基盤センター http://www.cc.u-tokyo.ac.jp/system/ha8000/.
4) STREAM BENCHMARK http://www.cs.virginia.edu/stream/.
5) HPC Challenge Benchmark http://icl.cs.utk.edu/hpcc/.
6) GeoFEM http://geofem.tokyo.rist.or.jp/.
7) UT-HPC benchmark http://www.cspp.cc.u-tokyo.ac.jp/ut-hpc-benchmark/.
8) 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）.
9) Mattson, T.G., Sanders, B.A., Massingill, B.L.: Patterns for Parallel Programming, Software Patterns Series (SPS), Addison-Wesley (2005).
10) Nakajima, K.: New Strategy for Coarse Grid Solvers in Parallel Multigrid Methods using OpenMP/MPI Hybrid Programming Models, ACM Proceedings of PPoPP/PMAM 2012, New Orleans, LA, USA (2012).
11) Scalable I/O Benchmark Downloads, Lawrence Livermore National Laboratory https://computing.llnl.gov/?set=code&page=sio_downloads.