

超並列マルチコア環境での自動チューニング機能の有効性： T2K オープンスパコン上の固有値ソルバを例にして

片 桐 孝 洋[†]

本稿では、超並列マルチコア環境での自動チューニングの有効性を評価するため、自動チューニング機能付き固有値ソルバ ABCLib_DRSSD を T2K オープンスパコン（東大）上で性能評価する。評価結果から、(1)ハイブリッド MPI 実行がノード数増加時に高速となること；(2)ピュア MPI とハイブリッド MPI では入力行列サイズに依存し最適な実装が異なること；が明らかになった。これはマルチコア環境では、自動チューニング機能が、従来環境よりも必須の機能となることを意味している。

Effectiveness of An Auto-tuning Facility on Massively Parallel Multicore Environment: An Example of Eigensolver on The T2K Open Supercomputer Takahiro Katagiri[†]

In this report, we evaluate performance of ABCLib_DRSSD, which is an eigensolver with auto-tuning facility, on the T2K Open Supercomputer (Todai) to show the effectiveness of auto-tuning on massively parallel multicore environment. The performance evaluation indicated that: (1) Hybrid MPI execution was faster when the number of nodes increased; (2) The optimal implantations among pure MPI and hybrid MPI were different according to input matrix sizes. This shows that the auto-tuning facility will be crucial function on multicore environment with compared to conventional environment.

1. はじめに

現在、超並列型の並列計算機が普及してきている。ハイエンドなスーパーコンピュータでは、10 万プロセッサ以上を有する。また、1 チップの上に複数のプロセッサを配置する**マルチコア型**のアーキテクチャが主流となり、各プロセッサはローカルなキャッシュと共有されたキャッシュを有する。このような、マルチコア型かつ階層化されたメモリを持つ計算機環境では、キャッシュレベルの局所性と、マルチコアレベルの並列性を同時に満たす計算アルゴリズムが性能の観点から要求される。この観点からアルゴリズムや実装方式を開発するという、新しい技術が求められている。

一方、数値計算ライブラリにおいては専門知識を必要とするアルゴリズムや実装上の性能パラメタが多数存在し、チューニング作業の人的・時間的コストが高いことが問題となっている。そこで、性能パラメタをユーザが利用する計算機環境に自動的に調整する（以降、**自動チューニング**、**Auto-Tuning (AT)**）とよぶ機能が研究され、一部の数値計算ライブラリの新機能として研究開発がなされてきた[1]-[2]。

本稿の目的は以下の通りである。従来から開発してきた AT 機能付き数値計算ライブラリを最新のマルチコア型かつ階層化されたメモリを持つ並列計算機で性能評価し問題点を吟味

する。対象の並列計算機（マルチコア型）として、2008 年 6 月から東京大学情報基盤センターで稼働している T2K オープンスパコン（HITACHI HA8000 クラスタシステム）を利用する。

本稿の構成は以下のとおりである。2 節で、AT 機能付き固有値ソルバ ABCLib_DRSSD の概略を説明する。3 節では、T2K オープンスパコン（東大）を利用した ABCLib_DRSSD の性能評価を行う。最後に、本稿で得られた知見を述べる。

2. AT 機能付き固有値ソルバ ABCLib_DRSSD

2.1 概要

ABCLib_DRSSD[2]は、対称実数固有値問題の任意の個数の固有値・固有ベクトルを計算できる機能をもつ並列数値計算ライブラリである。現在公開されている ABCLib_DRSSD ver. 1.04 [3]は、MPI-1 と Fortran90 で実装されている。ライブラリが提供する最適化方針と性能パラメタの定義範囲（実装方式）内で性能が自動的に最適化される機能を有し、これを自動チューニング機能とよぶ。性能のモデリングとして、任意次数の多項式近似による実行時間予測機能、および、サンプリング点の指定機能が実装されている。

ライブラリ上のアルゴリズムとして、対称固有値ソルバに利用される密対称行列用 Householder 三重対角化、三重対角対称行列用の固有値計算のための二分法、三重対角対称行列用の固有ベクトル計算のための逆反復法、および密対称行列の固有ベ

[†] 東京大学情報基盤センタースーパーコンピュータモデリング研究部門
Supercomputing Division, Information Technology Center,
The University of Tokyo

クトル計算のための Householder 逆変換の各ルーチンが利用できる。さらに、密行列用 QR 分解ルーチンも提供している。ABCLib_DRSED は、ベクトル化（もしくは、疑似ベクトル化）機能を有するプロセッサをもつノードにおいて、超並列環境（2000 年頃において 1000 並列以上）で高速に動作する方式が採用されている[4]¹。すなわち、1 プロセッサ当たりのメモリ量を固定しプロセッサ台数が増えるごとに全体の問題サイズが大きくなる状況での台数効果である weak scaling の効率を求めるのではなく、全体の問題サイズを固定しプロセッサ台数を増加させる状況での台数効果である strong scaling において、従来方式よりも効率が良いアルゴリズムを採用している。現在主流となっている階層キャッシュを有するプロセッサからなる超並列計算環境においては、行列サイズが大きくなるとキャッシュミスを生じて性能低下が起こる。単体性能の観点では、効率的なアルゴリズムが採用されていない。ただし台数効果では、十分な性能向上が期待できる。

2.2 ベンチマークとしての特徴

ベンチマークの観点からの ABCLib_DRSED の各処理について、以下にまとめる。

- 対称実数固有値ソルバ
 - Householder 三重対角化部（以降、TRD）
 - ◇ BLAS 2 演算性能評価
 - 行列-ベクトル積演算性能
 - 行列更新演算性能
 - ◇ マルチキャスト通信性能評価
 - プロセッサのグリッド ($p \times q$) における、同時放送処理 p 個（従事プロセッサ数 q 個）の性能
 - Householder 逆変換部（以降、HIT）
 - ◇ BLAS 1 演算性能評価
 - 必要な固有ベクトル数 (k) に応じて BLAS 1- k 更新となる行列更新演算性能。全固有ベクトルが必要なときは、BLAS 2 演算となる。
 - ◇ 集団通信性能評価
 - Gather 演算性能
- QR 分解

- 修正 Gram-Schmidt 演算部（以降、MGS AO）
 - ◇ BLAS3 演算性能評価
 - 行列更新演算性能。ただし、最外ループの刻み幅はブロック幅となる。
 - ◇ 1 対 1 通信性能評価
 - 通信粒度がブロック幅の逆数で変化する。つまり通信回数が、上記 BLAS3 のブロック幅の逆数となる実装における評価。

対称実数固有値ソルバではブロック化が実装されていないので、階層メモリを有するプロセッサで、かつキャッシュミスヒット時の性能劣化が大きなプロセッサでは、大規模問題実行時に性能劣化が起こることが予想される。一方、QR 分解ではブロック化アルゴリズムの採用により、大規模問題実行時の性能劣化を食い止めることが期待できる。ブロッキングある／なしでの性能差も、プロセッサ性能の評価指標の 1 つである。

2.3 性能パラメタの説明

AT 性能パラメタの観点では、ABCLib_DRSED の性能パラメタは以下に分類される[2]。

- 対称実数固有値ソルバ 性能パラメタ
 - Householder 三重対角化用
 - ◇ 通信実装方式切り替え *ictr*
 - ◇ 行列-ベクトル積アンローリング段数制御 *inv*
 - ◇ 行列更新演算アンローリング段数制御 *iud*
 - Householder 逆変換用
 - ◇ 通信実装方式切り替え *ichit*
 - ◇ 行列更新演算アンローリング段数制御 *ihit*
- QR 分解 性能パラメタ
 - ブロック幅調整 *ibl*
 - 枢軸ブロック演算用
 - ◇ 最外ループアンローリング段数制御 *iop*
 - ◇ 第 2 ループアンローリング段数制御 *isp*
 - 主演算用
 - ◇ 最外ループアンローリング段数制御 *ioo*
 - ◇ 第 2 ループアンローリング段数制御 *iso*

2.4 性能パラメタの定義域：現在の実装における制約

ABCLib_DRSED 1.04 では、自動チューニングパラメタについて、以下の実装がなされている。

- 対称実数固有値ソルバ 性能パラメタ定義域

¹ HITACHI SR2201 において、512PE 実行で 20,000 次元の三重対角化が、従来方式の ScaLAPACK の三重対角化ルーチンに比べ 2.7 倍高速であった[4]。

- Householder 三重対角化用
 - ◇ 通信実装方式切り替え
 $ictr = \{1 \text{ 対 } 1 \text{ 通信関数を用いた実装 集団通信関数を用いた実装}\}$
 - ◇ 行列ベクトル積アンローリング段数制御
 $imv = \{1 \text{ 段 } 2 \text{ 段 } \dots, 16 \text{ 段}\} : \text{最外ループ}$
 - ◇ 行列更新演算アンローリング段数制御
 $iud = \{1 \text{ 段 } 2 \text{ 段 } \dots, 16 \text{ 段}\} : \text{最外ループ}$
- Householder 逆変換用
 - ◇ 通信実装方式切り替え
 $ichit = \{\text{放送関数を用いた実装 } 1 \text{ 対 } 1 \text{ ブロッキング通信関数を用いた実装 } 1 \text{ 対 } 1 \text{ ノンブロッキング関数を用いた実装}\}$
 - ◇ 行列更新演算アンローリング段数制御
 $ihit = \{1 \text{ 段 } 2 \text{ 段 } \dots, 16 \text{ 段}\} : \text{最外ループ}$
- QR 分解 性能パラメタ定義域
 - ブロック幅調整
 $ibl = \{1, 2, 3, 4, 8, 16\}$
 - 枢軸ブロック演算用
 - ◇ 最外ループアンローリング段数制御
 $iop = \{1 \text{ 段 } 2 \text{ 段 } 3 \text{ 段 } 4 \text{ 段}\}$
 - ◇ 第2ループアンローリング段数制御
 $isp = \{1 \text{ 段 } 2 \text{ 段 } 3 \text{ 段 } 4 \text{ 段 } 8 \text{ 段 } 16 \text{ 段}\}$
 - 主算演用
 - ◇ 最外ループアンローリング段数制御
 $ioo = \{1 \text{ 段 } 2 \text{ 段 } 3 \text{ 段 } 4 \text{ 段}\}$
 - ◇ 第2ループアンローリング段数制御
 $iso = \{1 \text{ 段 } 2 \text{ 段 } 3 \text{ 段 } 4 \text{ 段 } 8 \text{ 段 } 16 \text{ 段}\}$

3. 性能評価

マルチコア型の計算機では、MPI とスレッドによるハイブリッドMPI (ハイブリッドプログラミング) の有効性が指摘されている[5]。そこで、本性能評価では、ハイブリッドMPI の性能評価をすることも目的とする。

3.1 実験環境

T2K オープンスパコン (東大) (HITACHI HA8000 クラスタシステム) のノードは、AMD Opteron 8356 (2.3GHz, 4コア) を4台 (4ソケット) 搭載しており、メモリは32GBである。理論最大演算性能は、ノードあたり147.2GFLOPSである。キャッシュサイズは、L1 命令キャッシュ、L1 データキャッシュともに64Kbytes であり、2 Way Associativity (ライトバック, 3 サ

イクル) である。また、L2 キャッシュは512Kbytes である。L1 キャッシュとL2 キャッシュはコアごとに独立している。L3 キャッシュ2048Kbytes であり、ソケット内で共有されている。各ソケットには、ローカルメモリ8GBがある。通信性能は運用クラスター群で異なる。ここではMiri-10G が4本実装されており、最大で5GB/secの双方向性能を有するA群を利用している。

コンパイラは、日立最適化Fortran90 V01-00-/A で、コンパイラオプションは、**ビュアMPI** (MPI のみによる実行) では、`-Oss -noproallel`、**ハイブリッドMPI** (MPI と自動並列化によるスレッド並列化の混合実行) では `-Oss -parallel` である。実験期間は、2008年7月1日～7月9日である。

以降の性能評価では、各問題サイズにおいて性能パラメタを全て自動チューニングして最高速となるパラメタが自動設定されている。なお、ABCLib_DRSSD 1.04 におけるデフォルトパラメタは、以下のとおりである。

- $ictr = \{1 \text{ 対 } 1 \text{ 通信関数を用いた実装}\}$
- $imv = \{8 \text{ 段}\}$
- $iud = \{6 \text{ 段}\}$
- $ichit = \{\text{放送関数を用いた実装}\}$
- $ihit = \{1 \text{ 段}\}$
- $ibl = \{4\}$
- $iop = \{4 \text{ 段}\}; isp = \{8 \text{ 段}\};$
- $ioo = \{4 \text{ 段}\}; iso = \{8 \text{ 段}\};$

以上のデフォルトパラメタに対するATによる速度向上について評価する。アンローリング段数を1段に固定し、コンパイラ最適化による効果のみと比べたものではない²。このようなコンパイラ最適化のみより、本デフォルトパラメタの方が経験的に高速となる。

なお、全固有値、全固有ベクトルを計算する場合について、性能評価を行う。

3.2 単体性能

ビュアMPIにおいて、Linux では、プロセスを任意のCPUとメモリに割り付ける `numactl` というコマンドが提供されている。T2K オープンスパコン (東大) でも `numactl` が利用できる。ビュアMPIにおいて、`numactl` を使用した実行時間について、各処理のGFLOPS値を計算した。図1にそれを示す。

² これをベースラインとしてATの効果を評価する論文も多い。このベースラインを用いた評価に比べ、本性能評価の基準では経験的にATの効果が低めに算出される。

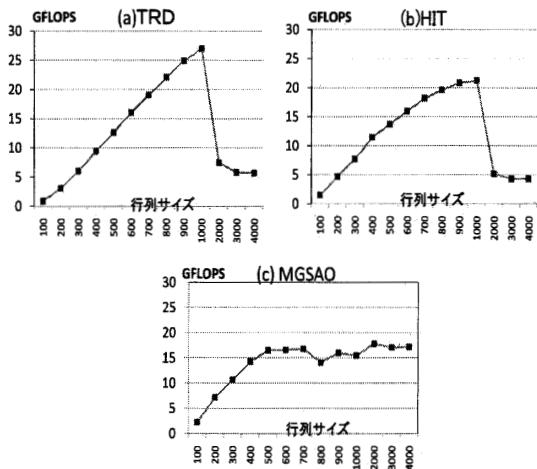


図 1 各処理の GFLOPS 値 (1 ノード, ピュア MPI)

図 1 では、TRD と HIT において行列サイズ 2,000 を超えてから演算性能が 1/4~1/5 に低下する。一方、MGS AO では、このような性能低下がない。MGS AO はブロック化アルゴリズムを採用していることから、ブロック化の効果がきわめて大きいことを意味している。また、行列サイズ 1000 の時のワーキングスペースは、TRD と HIT では概算で $4 \times N$ (ここで、 N は行列サイズ) となるため $4 \times 1000 \times 8 \approx 32K$ である。したがって L1 キャッシュにデータが乗る上限となることが予想されることから、結果は妥当である。

一方、ピーク性能 (147.2 GFLOPS) に対する効率率は 12%~17% であり、極めて十分でない。この理由は ABCLib_DRSSD 1.04 の実装においては、(1) TRD と HIT はブロック化が採用されていないこと；および、(2) MGS AO ではブロック化が採用されているがループアンローリングやタイリングなどの自動チューニング項目が最内側ループに対して行われていない (アンローリングについては TRD、HIT も同様)；という、AT ライブラリ自体の最適化戦略から生じたものと推測される。ABCLib_DRSSD 開発時に想定した計算機では、最内側ループ長が長くとも性能劣化が起こりにくいアーキテクチャを想定していたためであり、このことから T2K オープンスパコンでは技術的な最適化戦略の再構築が必要である。

3.3 ハイブリッド MPI 実行の効果

次に、MPI のみによる実行 (ピュア MPI) と、MPI とスレッドの混合環境による実行 (ハイブリッド MPI) の性能を比較する。図 2~図 4 に、各処理における行列サイズ 4000 の時の、ピュ

ア MPI による実行とハイブリッド MPI による実行の実行時間を載せる。ここで図中の表記“pure”とはピュア MPI (16 プロセス)、“p4 * th4”はハイブリッド MPI (4 プロセス、4 スレッド)、“p1 * th16”はハイブリッド MPI (1 プロセス、16 スレッド) である。これらはいずれも、ノード内における実行形態であり、ノード間は MPI による実行となる。

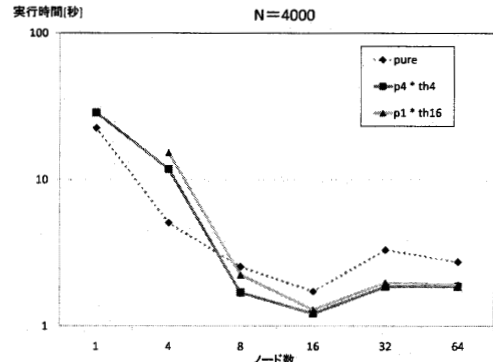


図 2 ハイブリッド MPI の効果 (TRD)

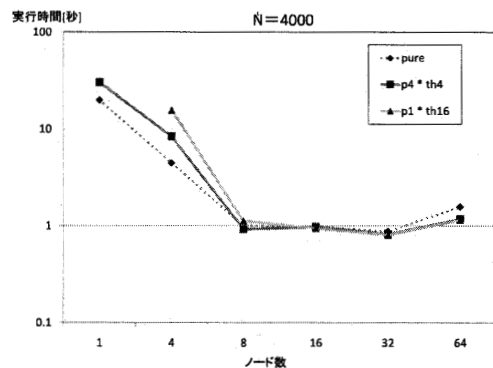


図 3 ハイブリッド MPI の効果 (HIT)

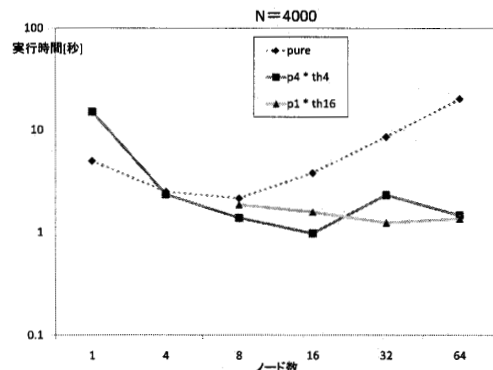


図 4 ハイブリッド MPI の効果 (MGS AO)

図2～図4では、1ノードのときにはピュアMPIによる実行が高速であるが、4ノード、もしくは8ノードを超えたあたりから逆転が生じ、最も高速となるのは4プロセス、4スレッドのハイブリッドMPIであることがわかる。したがって、ノード数が増加するにつれ、ハイブリッドMPIが高速となる場合があることが判明した。

一方、台数効果をみるため、図5にTRDの場合の台数効果(ピュアMPI (16プロセス)、ハイブリッドMPI (4プロセス、4スレッド))の1ノード実行を基準とする台数効果を乗せる。

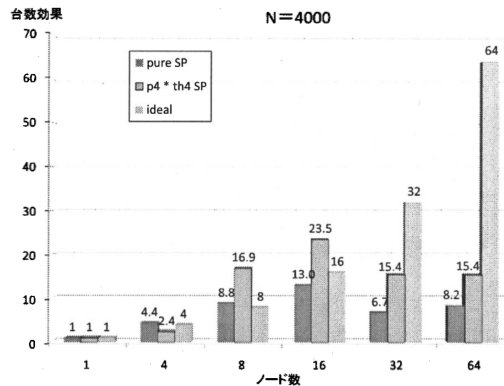


図5 台数効果 (TRD)

図5から、4ノード～16ノードにおいて、理想的な台数効果を超えるスーパーリニアスピードアップが観測された。この理由は、図1からわかるように、4ノードを超えると行列サイズ4,000ではL1キャッシュに乗るサイズになることから、キャッシュの効果だと推定される。また、ピュアMPIよりハイブリッドMPIの方が台数効果が高い。これは、ccNUMAアーキテクチャを考慮したメモリ配置によりメモリアクセス時間が短縮されることと、MPIプロセス数がハイブリッドMPIによるものはピュアMPIより1/4になることから通信性能の劣化が小さいことが原因と推察される。いずれにせよ、ハイブリッドMPIの長所が現れたものと思われる。

3.4 自動チューニングの効果

図6、図7にATによる速度向上を示す。図6、図7から、TRD、MGS AO では1ノードより64ノードのほうが概ねATの効果が高い。この理由は、それぞれ集団通信演算の実装選択と通信粒度調整がAT機能として入っているので、通信最適化の効果がTRD、MGS AO のアルゴリズムでは出やすいからと推察される。一方HITでは64ノード時に1ノードよりもAT効果が低い。これは、HITでは頻りにGather処理をする通信が必要であるが、この処理を最適化するだけの実装選択のバリエーションが少

なかったことによると思われる。なお、オリジナルのABCLib_DRSSSED 1.04ではHITにおいてノンブロッキング通信による実装選択機能が提供されているが、本環境ではMPIバッファのオーバーフローとなり実行できなかったことから、この通信実装選択はOFFにした。また、全般的にハイブリッドMPIはAT効果が低い。この理由は、デフォルトパラメタのアンローリング段数では4スレッド実行で性能を出すだけの並列性がなく、ATによりアンローリング段数を大きくすることで並列性を向上できた(高速化できた)ことが理由と考えられる。

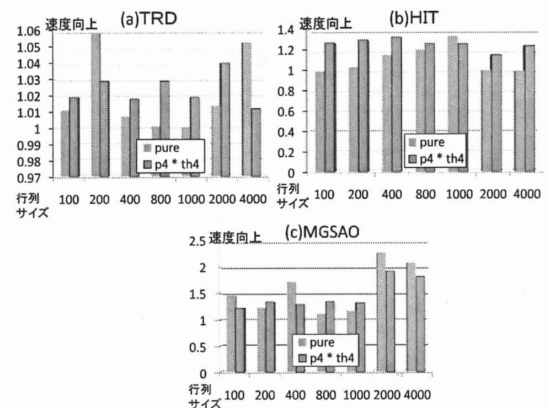


図6 ATの効果 (1ノード)

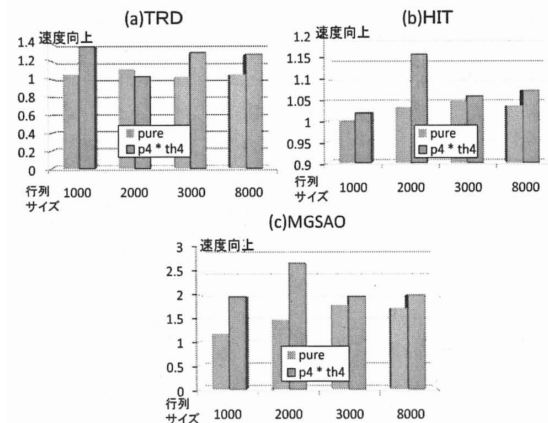


図7 ATの効果 (64ノード)

ブロック化に伴う演算カーネル実装の変化について、最適なMGS AO のループアンローリング長を見ることで推察してみる。MGS AO では、BLAS3 演算部分に最外ループと第2ループについてのアンローリング段数をATしている。そこで、(最外ループアンローリング段数) × (第2ループアンローリング段数)を実装指標とした場合の評価結果を図8に示す。

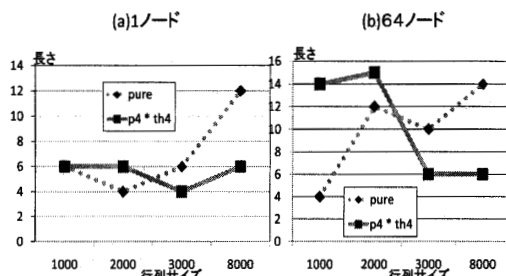


図 8 最適ループ長の指標

(MGS AO, 最外ループアンローリング段数×
第2ループアンローリング段数)

図 8 から面白いことに、ピュア MPI では行列サイズが小さいとき指標が小さく、行列サイズが大きくなるにつれ指標も大きくなる傾向がある。一方、ハイブリッド MPI ではピュア MPI と逆の傾向が伺われる。この理由は、ピュア MPI では行列サイズを大きくすると各コアの演算器を効率的に動かすために多数の命令レベルの並列性（プログラム上に“陽”に現れるもの）が必要となることから指標を大きくする必要があること、一方で、ハイブリッド MPI では行列サイズが大きくなると各スレッドからのデータ読み書きによるメモリ衝突がオーバーヘッドとなるため命令レベル並列性（プログラム上に“陽”に現れるもの）を抑える必要があることが理由として考えられる。

いずれにせよここでの主張は、「ピュア MPI とハイブリッド MPI では、高性能を達成するために、行列サイズに影響される全く異なった実装が必要となる」ということである。このことは高性能を達成するために、ユーザに従来よりもさらなる負担（ピュア MPI とハイブリッド MPI の 2 種それぞれについて、行列サイズを考慮した最適実装）が必要となることを意味している。今後、コンパイラ自動最適化やライブラリ上の AT 機能など、何らかの自動性能チューニング技術の強化が、マルチコア型の並列計算機環境では必須となるに違いない。

4. おわりに

マルチコア型の超並列環境において、ノード数増加に伴いハイブリッド MPI 実行が高速となる例が確認された。今後、利用するノード数により、ピュア MPI 実行かハイブリッド MPI 実行かの切り替えをする機能が、自動チューニング機能として必要になる。また、入力される行列サイズはライブラリレベルでは実行時になるまで不明である。今回の実験結果のように、ピュア MPI とハイブリッド MPI 実行で最適な実装方式が入力サイズ

に依存して変化する場合、手動でチューニングすることがさらに困難となる。これは、マルチコア環境では自動チューニング機能が必須の機能になることを示唆するものである。

一方速度面では、T2K オープンスパコン（東大）の 64 ノードの三重対角化は、行列サイズ 10,000 のとき約 4.7 秒、行列サイズ 80,000 においても約 50 分であった。従来採用してきたアルゴリズムと自動チューニング戦略は、単体性能の観点から十分ではなかった。これは十分なキャッシュ最適化が施されていないことに起因すると考えられるが、同時に、CPU における演算速度向上に対するメモリアクセス性能の向上がハードウェアとして不十分であることも意味している。このハードウェア上の演算速度とメモリアクセス速度の「ギャップ」を解決するには、さらなるアルゴリズム上の工夫が必要である。今後の課題として、ブロック化アルゴリズムの実装、マルチコア型のアーキテクチャを意識した新アルゴリズムの開発が必要である。加えて、マルチコア型アーキテクチャむきの自動チューニング方式（たとえば、ハイブリッド MPI 実行が自動選択できるもの）の開発が重要である。

謝辞 本研究の一部は、特定領域研究（情報爆発）「情報爆発時代のロバストな自動チューニングシステムに向けた数理的基盤技術の研究」（19024018）の支援による。

参考文献

- [1] James Demmel, Jack Dongarra, Victor Eijkhout, Erika Fuentes, Antoine Petit, Richard Vuduc, R. Clint Whaley, and Katherine Yelick: Self-Adapting Linear Algebra Algorithms and Software, Proceedings of the IEEE, Special Issue on Program Generation, Optimization, and Adaptation, 93(2) (2005)
- [2] Takahiro Katagiri, Kenji Kise, Hiroki Honda, and Toshitsugu Yuba: ABCLib_DRSSD: A Parallel Eigensolver with an Auto-tuning Facility, Parallel Computing, Vol. 32, Issue 3, pp. 231-250 (2006)
- [3] ABCLib_DRSSD Home Page: <http://www.abc-lib.org/>
- [4] Takahiro Katagiri and Yasumasa Kanada: An Efficient Implementation of Parallel Eigenvalue Computation for Massively Parallel Processing, Parallel Computing, Vol. 27, No. 14, pp. 1831-1845 (2001)
- [5] 中島 研吾：階層型領域分割によるマルチステージ並列前処理手法へのハイブリッド並列プログラミングモデルの適用, 情報処理学会研究報告 2007-HPC-110, Vol. 2007, No. 59 (20070608) pp. 25-30 (2007)