計算機アーキテクチャ 137−21
ハイパフォーマンス 80−21
コンピューティング
（2000. 3. 3）

Beowulf クラスクラスタ ERATO-1 のチューニングと評価

奥乃 博[†1,†2] 京田耕司[†1,†3]
中臺一博[†1] 北野宏明[†1,†4]

Beowulf クラスクラスタは，PC クラスタの論理構成法であり，コモディティハードウェアやソフトウェアによりPC クラスタが容易に構築できる．しかし，それらの組合せによってはうまく動かなかったり，あるいは，性能が全く出ないということがある．本稿では，Beowulf クラスクラスタのチューニングを (1) ネットワーク，(2) プロセッサ間通信ライブラリ (MPI や PVM)，(3) 応用プログラム，という 3 つのレベルで分けて，行うことを提案する．具体的には，NetPIPE というネットワーク測定用ツールを用いて，(1) と (2) をチューニングする．次に，線形代数でよく使われる LINPACK の一つ ScaLAPACK を応用プログラムとして利用し，(1) と (2) から得られたネットワーク特性を用いて，ScaLAPACK のチューニングを行う．とくに，小さな行列に分割することと，最適化された線形代数パッケージを使用することが，ScaLAPACK の性能向上に不可欠であることが判明した．これらの知見を利用することにより，Pentium-II 450 MHz, 256 MByte メモリのノード 32 台で構成される ERATO-1 に本手法を適用した結果，ハードウェアの不具合いが発見でき，また，LINPACK ベンチマークで 6.76GFlops の性能が得られた．

Tuning and Evaluation of the Beowuf-Class Cluster ERATO-1

Hiroshi G. Okuno,[†1,†2] Koji M. Kyoda,[†1,†3] Kazuhiro Nakadai[†1]
and Hiroaki Kitano[†1,†4]

Beowulf-Class cluster is a logical organization of PC clusters composed of mass-market off-the-shelf hardware and software. The user may have problems that their implementation won't work well in hardware level or their implementation provides quite a poor performance. In this paper, we present a new method to tune and evaluation of the Beowulf-Class cluter by focusing on three levels: (1) network level, (2) message passing system level (e.g., MPI, PVM), and (3) application level. The first two performance is measured by NetPIPE developed by Ames Lab. ScaLAPACK (parallel version of LINPACK) is used as benchmarks for application programs, because it is one of the most common linear algebra subprograms and its evaluation is beneficial for numerical computation users. ScaLAPACK is tuned by using parameters determined by NetPIPE. ERATO-1 Beowulf-class cluster, 32 nodes of Pentium-II 450HHz processor with 256MByte of memory, is tuned by the proposed method. First, a network interface card installed in each ERATO-1 node is indentified as the cause of poor performance and finally ERATO-1 attained 6.76 GFlops with LINPACK benchmark.

1. はじめに

最近，パーソナルコンピュータ (PC) を用いて PC クラスタを構成し，分散並列処理を行うことが，盛んになってきている．とくに，米国 NASA で始まった Beowulf クラスクラスタ (以降，Beowulf クラスタと呼ぶ) は，Linux OS を用い，ネットワークインターフェースカード (NIC) に対するドライバが充実し，さらに，必要なソフトウェアが無料で手に入るので注目されている．実際，世界中の多くの研究室で Beowulf クラスタの構築され，小さな研究グループでスーパーコンピューティングの実験することが可能となっている．

Beowulf クラスタは，ネットワークで接続された UNIX PC 群に対する『論理的な』クラスタ構成法である．Beowulf クラスタは，汎用品を用いてスーパーコンピュータが構成できる一方，個々のクラスタシステムの性能がその設計時には予測が極めて難しいことや，ハードウェアやソフトウェアの組合せによってはクラスタシステムが正しく動かなかったり，性能が出ないことなどが挙げられる．

本稿では，Beowulf クラスタの調整と性能のチューニングを次の 3 つのレベルで行う方法を提案する：

(1) ネットワークレベル

†1 科学技術振興事業団 ERATO 北野共生システムプロジェクト
Kitano Symbiotic Systems Project, ERATO, JST
†2 東京理科大学 理工学部 情報科学科
Science University of Tokyo
†3 慶應義塾大学大学院 理工学研究科 計算機科学専攻
Department of Computer Science, Keio University
†4 ソニーコンピュータサイエンス研究所株式会社
Sony Computer Science Laboratories, Inc.

(2) プロセッサ間通信ライブラリレベル
(3) 応用プログラム (数値計算)

前者2つのレベルの性能測定は，米国 Ames 研究所 Scalable Computing 研究室で開発された NetPIPE[8] を用いる．というのは，NetPIPE では TCP レベル，および，標準的なプロセッサ間通信ライブラリである MPI (Message Passing Interface)[5] と PVM (Parallel Virtual Machine)[3] レベルでの測定が統一的にできるからである．これにより，うまく動かない場合の挙動とうまく動く場合の挙動が容易に切り分けられる．

応用レベルでは，数値計算の標準的なベンチマークの一つである行列計算の ScaLAPACK (LINPACK) を取り上げる．というのは，Beowulf クラスタを計算科学 (Computing Science) に適用する場合，線形計算の標準的なライブラリとして LAPACK がよく使われるので，その性能評価や最適化ができれば，Beowulf クラスタの開発者やユーザへの指針となるからである．

2. Beowulf クラスタ ERATO-1 の構成

2.1 ハードウェアシステム構成

本稿で報告する Beowulf クラスタ "ERATO-1" は，32台のノードと1台のサーバから構成されている[6]．各ノードの構成は，下記の通りである．

- CPU: Intel Pentium-II 450MHz, 16KByte の L1 命令キャッシュと 512KByte L2 キャッシュ
- Memory: SDRAM 256MByte
- HDD: 6.4 GB IDE
- NIC: Accton Cheetah EN-2701TX
- Video Card: 2MByte, CD-ROM: 40 倍速

サーバは，Memory が 384MByte SDRAM であり，HDD が 12.8GB，NIC が2枚装備していることを除いて，ノードと同じ構成である．サーバは，すべてのノードに対するファイルサーバと，研究所内のネットワークとのゲートウェイの役割を果たす．もちろん，ノードとして使用することもできる．

NIC (Network Interface Card) は，OS のバージョンやドライバーのバージョンに最も影響を受ける．Beowulf メーリングリストでは，DEC 社製の Tulip チップ (型番 211xx) が推奨されている．しかし，Tulip チップはマイナーなバージョンが多数存在しており，対応するドライバがなかったり，未対応だったりすることが，動作しなかったり，性能があがらなかったりする原因であることが我々の実験で判明した．

2.2 ネットワークシステム構成

ネットワークは Fast Ethernet (100 Base-TX) であり，スイッチングハブには Cisco 製の Catalyst 5505 (48 ポート，バックプレーン間のスイッチング速度は 3.6Gbps) を使用している．

2.3 ソフトウェアシステム構成

OS には Linux 2.2.5-15 kernel に基づいた Red Hat Linux release 6.0 (Hedwig) を，ノード間の通信ライブラリには MPI と PVM を使用した．MPI は，MPI のフリーソフトウェアである米国アルゴンヌ国立研究所で開発された mpich version 1.2.0 を，PVM は version 3.4 である．Fortran コンパイラは，市販の Absoft 社製 Pro Fortran 5.0 と富士通九州システムエンジニアリング製 Fortran V1.0 を使用した．

ScaLAPACK の行列は，スタック領域に割り当てられる．Linux では，ユーザの使用できるスタック領域は制限があるので，`tcsh` の `unlimit` コマンドで無制限にして，大規模な行列に対する測定を行った．

3. NetPIPE による通信性能測定

NetPIPE は，クラスタの1対1のノード間でピンポン型の通信によって，TCP, MPI, PVM レベルでの性能を測定する．まず，最適な NIC を選択するために次の組合せに対して，NetPIPE で測定した．

- プロトコル — TCP, MPI, PVM
- 接続形態 — クロスケーブルによる直接接続，スイッチングハブ (Cisco 製 Catalyst 5505，48 ポート)，スイッチングハブ (Allied Telesis 製の CentroCOM FS708XL，8 ポート) の3種類．
- NIC — Accton NIC 311 (DEC Tulip 21143 PC 使用)，Accton Cheetah EN1207C-TX (DEC Tulip 21143 PD 使用)，Corega FastEther PCI-TX (DEC Tulip 21140 AF 使用) の3種類．

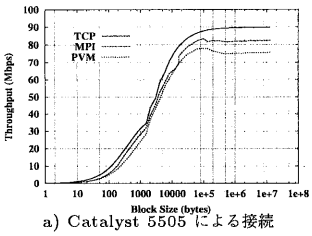
測定データは下記のような整理を行った．

(1) バンド幅 (bandwidth) — x 軸に転送ブロック長，y 軸にスループットを取った "スループットグラフ" で，飽和時のスループットを "バンド幅" と定義する．

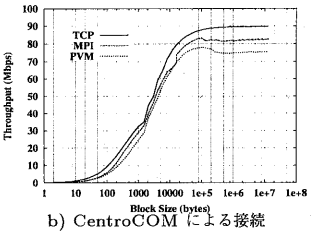
(2) 通信遅延時間 (latency) — x 軸に転送時間，y 軸に転送ブロック長を取ったネットワークシグニチャグラフにおいて，0バイトのブロックを転送した時の転送時間を "通信遅延時間" と定義する．

(3) 飽和点 (saturation point) — x 軸に転送時間，y 軸に転送ブロック長を取った "飽和グラフ" で，ブロック長と転送時間の関係が一定となる転送ブロック長を "飽和点" と定義する．それより小さいブロックの転送時間の変動が大きいするので，注意が必要である．

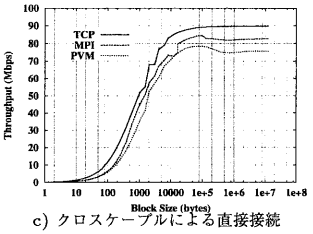
図1に，NIC に EN2701C-TX (ドライバ ver. 0.91 を使用) を使用し，3種類の接続形態で，3つのプロトコルで測定したスループットグラフを示す．また，図



a) Catalyst 5505 による接続



b) CentroCOM による接続



c) クロスケーブルによる直接接続

図1 3種類の接続形態によるスループット

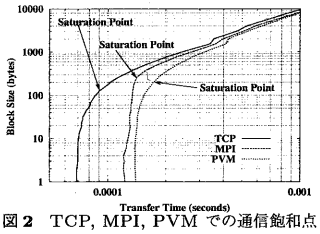


図2 TCP, MPI, PVM での通信飽和点

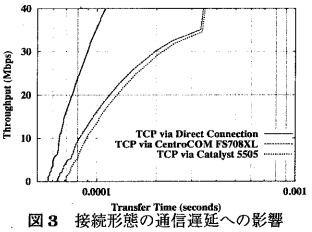


図3 接続形態の通信遅延への影響

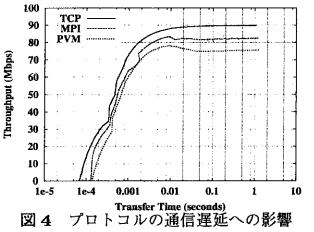


図4 プロトコルの通信遅延への影響

5~7に3つの接続形態での通信特性を示す．スイッチングバブによる接続は，直接接続よりスループットの立ち上がりが悪いが，バンド幅はほとんど変わらない．

この測定は，1対のノード間で NetPIPE を実行したものである．4対のノード間 (8ノード) で NetPIPE を同時に実行しても，スループットの変化はほとんどない．Catalyst の方が少し安定している．

図2に，ERATO-1 での3つの接続形態に対する TCP での通信遅延を示す．各スループット曲線の x 軸切片が通信遅延である．直接接続，CentroCOM，Catalyst の順に通信遅延が大きくなる．また，Catalyst を用いた場合の3つのプロトコルでの通信遅延を図4に示す．MPI と PVM との通信遅延は，TCP よりは大きいものの両者はほとんど差がない．

プロトコル層による通信オーバヘッドでスループットが低下する．MPI はこのシグニチャーグラフから，スループットおよび通信遅延が TCP よりも低下している．MPI と PVM のバンド幅の低下は，約8%，16%であり，各々の通信遅延は，1.8倍，2.0倍である．

図4に，3つのプロトコルに対する飽和グラフを示す．各プロトコルに対する飽和点は図中の矢印で示す．TCP の飽和点が一番小さく，MPI の飽和点が一番大きい．つまり，MPI では，転送ブロック長を TCP や PVM よりは大きくしないとネットワークの通信の性能が出ない．一方，ERATO-1 では，NIC が1枚しか実装されていないので，バンド幅のスループットが出るブロック長で転送を行うと，プロセス起動のサーバであるノードでの NIC へのアクセス競合が生じ，そこで逐次的に処理されることになり，性能が劣化するので，これを避けるためにブロック長を極力小さくすることが要求される．最適な転送ブロック長を見つける上で，飽和グラフは有用である．実際，ScaLAPACK での行列の分割サイズ決定に重要な役割を果たした．

3.1 NetPIPE による不良箇所の発見

NIC を NIC311 に変更してみると，スループットは図8 a) に示すように，TCP は 20KByte 辺りから，MPI は 4KByte 辺りからスループットが急激に悪くなり，200KByte 辺りから飽和状態に復帰する．このようなスループットの激減は，10Base-T の Ethernet card でも報告されている[8]．この原因の一つとして，NIC の実装が OS の TCP バッファのサイズ (4096Byte) や MTU (Maximum Transmission Unit, 1500Byte) の影響を受けるようになっていることが推察される．ただし，PVM の場合には，UDP を使用しているためか，そのような影響は見られない．

しかし，この NIC を使用したクラスタでは，4ノードまではノード数が増えるとともに実行時間が減少するが，4ノード以上では，プログラムが通信待ちでなかなか終了しない．この性能低下は，MPI (TCP を使用) でも PVM (UDP を使用) でも観測された．この時に NetPIPE で1対間のスループットを測定すると，図8 b) に示したようにスループットが急に激減している．

シグニチャーグラフ暴れの様子は，図9 a) から図9 b) への変化から見てとれる．スループットグラフとこのグラフから，MPI では転送ブロックサイズが TCP バッファのサイズを超えた途端に急激にネットワークの性能が低下する．このような現象は，NFS でも生じることが Beowulf メーリングリストでも報告されている．ま

図5 Catalyst 5505 を使用した場合の通信特性

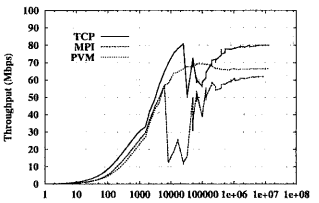
	バンド幅	通信遅延	飽和点
TCP	89.75Mbps	0.069ms	139B
MPI	82.67Mbps	0.123ms	262B
PVM	75.67Mbps	0.140ms	200B

図6 CentroCOM を使用した場合の通信特性

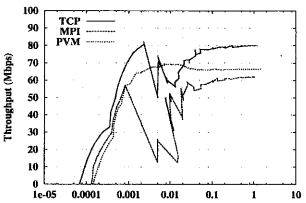
	バンド幅	通信遅延	飽和点
TCP	89.75Mbps	0.064ms	140B
MPI	82.76Mbps	0.118ms	260B
PVM	75.39Mbps	0.134ms	200B

図7 直接接続での通信特性

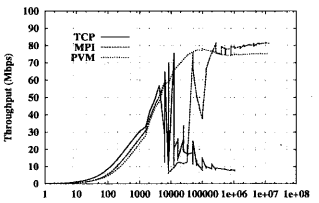
	バンド幅	通信遅延	飽和点
TCP	89.76Mbps	0.057ms	140B
MPI	82.66Mbps	0.108ms	280B
PVM	75.66Mbps	0.123ms	200B



a) 良好時のスループット

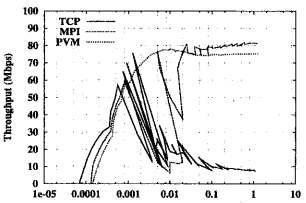


a) 良好時のシグニチャーグラフ



b) トラッフィク増後の変化

図8 相性の悪い NIC のスループット



b) トラッフィク増後の変化

図9 相性の悪い NIC の性能の暴れ

た，同じ現象は，EN2701C-TX で自作の Category 5 のネットワークケーブルを使用した時にも観測された．NetPIPE の性能測定でこのような現象を確認した場合には，速やかに NIC やケーブルを交換することが重要である．なお，このような性能の低下は，Corega 製の NIC では観測されなかった．(同製品は生産中止．)

4. ScaLAPACK による性能測定

4.1 応用プログラム (数値計算) のチューニング

ScaLAPACK (Scalable Linear Algebra software PACKage for high performance MIMD distributed-memory parallel computers) は，Oak Ridge 国立研究所，ライス大学，UCB，UCLA，イリノイ大学，テネシー大学などの複数の機関で共同で開発された密な帯行列用のソフトウェア群である[2]．ScaLAPACK は，LINPACK の後継のソフトウェア LAPACK を分散メモリ環境に移植したものである．ScaLAPACK は，次のようなサブプログラムから構成されている．

(1) BLAS (Basic Linear Algebra Subprograms) — 線形代数計算の根幹となるプログラム．

(2) LAPACK (Linear Algebra PACKage) — BLAS を使って書かれている．

(3) BLACS (Basic Linear Algebra Communication Subprograms) — MPI や PVM といった分散計算環境と PBLAS とのインターフェースをとる．

(4) PBLAS (Parallel BLAS) — BLAS と BLACS を統合するプログラムである．

ScaLAPACK で最も性能に影響を与えるのは，BLAS コードとノード間通信ライブラリ MPI や PVM である．BLAS 最適ライブラリには，アセンブラコードや Pentium-II の L1, L2 キャッシュ使用のものがある．

次の 2 つの ScaLAPACK プログラムを使用した．

(1) GESV (行列の LU 分解)

(2) GEMM (Matrix-to-Matrix) — GEMM は，3 つの行列 A，B，C に対して，α，β をスカラとすると，$C := \alpha\, A\, B + \beta\, C$ を計算する．ただし，各要素は以下の式で計算される：

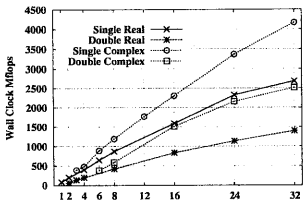
$$c_{M,N} := \alpha \times a_{M,K} \times b_{K,N} + \beta \times c_{M,N}$$

ScaLAPACK は Fortran と C で実現されている．一般に Fortran コンパイラは，C とのリンクを取るためにコンパイル時のオプションとして変数や関数名の変換モードを複数提供している．たとえば，大文字にする，小文字にする，先頭だけ大文字にする，名前の最後に下線を 1 つ付加，下線を 2 つ付加，など．使用する Fortran システム (システムライブラリ) がどのモードでコンパイルされているかを確かめて，それに対応するように，システムを作成する必要がある．

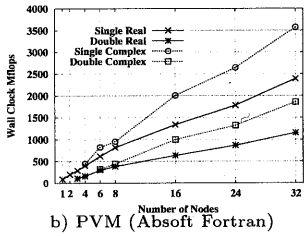
本稿では次のような組合せをチェックした．C コンパイラは，gcc (egcs-2.91.66) である．

(1) Absoft Pro Fortran 5.0 — 「大文字」でコンパイル．ベンダー提供の precompiled BLAS ライブラリは，最適コードではない．(*Absoft* と略記)

(2) gnu G77 Fortran — f2c (Fortran to C) システムに従うようにコンパイル．www.beowulf-underground.org 提供の韓国の Yoon Ho 氏が precompile した 3 種類のシステムを使用．BLAS ライブラリとして，netlib.org 提供のもの (*G77* と略記)，ATLAS (Automatically Tuned Linear Algebra Software) から作成したもの (*G77-Atlas* と略記)，Greg Henry 氏作成のもの (*G77-LC* と略記) である．

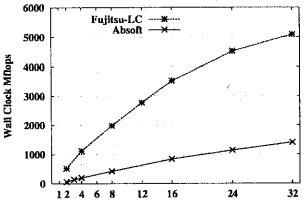


a) MPI (Absoft Fortran)

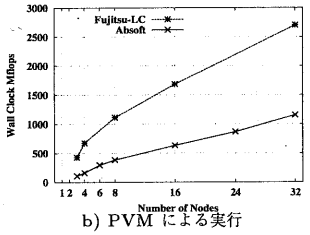


b) PVM (Absoft Fortran)

図10 データ型による実行時間の違い



a) MPI による実行



b) PVM による実行

図11 行列サイズによる実行時間の違い

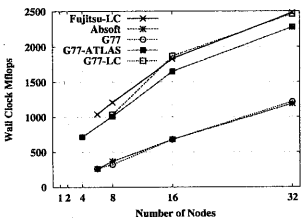


図12 5つのシステムのLU分解による実行速度の比較

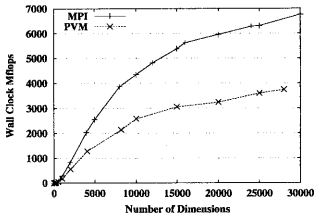


図13 32ノードでのMPI通信オーバヘッド

(3) Fujitsu Fortran V1.0 — 下線を1つ付加するモードでコンパイル．BLAS ライブラリは，Greg Henry 氏作成のものを使用した (*Fujitsu-LC* と略記)．

測定は下記の2種類の方法で行った．

(1) Wall クロック — **gettimeofday** を使用．

(2) CPU 時間 — **getrusage** を使用．

前者にはタスク分割と通信のためのオーバヘッドが含まれているので，クラスタの性能の測定に適しており，本稿で示す測定結果はすべて Wall クロック時間とした．

4.2 各システムの比較

Fortran の数値データ型は，単精度浮動小数点 (*Single Real* と略記)，倍精度浮動小数点 (*Double Real* と略記)，単精度複素数 (*Complex* と略記)，倍精度複素数 (*Double Complex* と略記) ある．これら4つのデータ型に対する GEMM を Absoft Fortran で実行した時の計算時間を図10に示す．MPI (図10 a)) も PVM (図10 b)) も共に同じ傾向を示しており，MPI の方が PVM より十数％程度速い．

実行時間は，Single Real，Double Real，Single Complex，Double Complex の順にがかかるが，Flops 値は，Single Complex が一番大きく，Single Real，Double Complex，Double Real の順に小さくなっている．というのは，複素数の乗算には乗算と加減算が入ってくるので，行列演算中に実行される総浮動小数点演算中の乗算の占める割合が，実数の場合よりも少なく，その分相対的に Flops 値が上昇するためである．Pentium Linux では，単精度複素数と倍精度浮動小数点は共に8バイトで表現されるが，図10に示したように Flops 値で表した性能は大きく違っている．性能と台数との関係は，データ型には依存しないので，Flops が最小の倍精度浮動小数点だけを取り上げる．

タスクを分割する時のデータのサイズ (粒度) は，前章より飽和グラフで，飽和点からバンド幅のスループットの間にあるブロック長にするのがよい．ERATO-1 での MPI の好ましいブロック長は 262 ～約3万バイトである．倍精度浮動小数点が8バイトで表現されるので，6×6 から 111×111 の正方行列で，行列を分解するのがよいと予想される．実際 GEMM や LU 分解では，32×32 か 64×64 で最もよい性能が得られることが多い．SMP では，逆にバンド幅あたりのブロック長になるように大きなデータを割り当てることが多い．

図12に次のようなシステムを LINPACK の1つである LU 分解 (4800 × 4800) を MPI 上で行った結果を示す．最適化された BLAS ライブラリを使うことは必須であり，現在，利用可能な最適 BLAS ライブラリはほとんど性能が変わらない．

4.3 クラスタ計算のオーバーヘッド

クラスタ計算のオーバーヘッドを調べるために，ノード数を32に固定し，行列のサイズを変化させて LU 分解を行い，得られた Flops 値の変化を求めた．その結果を図13に示す．行列のサイズが大きくなるとほぼリニアな性能 (Flops 値) となる．この傾向を小さな行列に外挿して得られる理想的な性能値から，実際の性能は大きく悪くなっている．このような性能の外挿値からのずれが，クラスタ化によるオーバーヘッドと考えられる．

4.4 ERATO-1 の最大性能

ScaLAPACK の性能評価には，Fujitsu-LC を用いた．また，図11から Flops 値をより大きくするには，より大きなサイズの行列演算を用いればよい．ERATO-1 はメモリが256MByte なので，それで処理可能な最大

表1 ERATO-1 の倍精度浮動小数点の最大性能

GEMM	15000 × 15000	4800 × 4800
Wall Clock Time	7.68 GFlops	5.04 GFlops
LU 分解 (LINPACK)	30000 × 30000	4800 × 4800
Wall Clock Time	6.76 GFlops	2.43 GFlops

の行列サイズ (前述したように行列は2次記憶ではなく，実メモリ上に置かれる) は，GEMM の場合には 15000 × 15000 であり，LU 分解の場合には 30000 × 30000 である．この時の性能を表1にまとめる．

ATLAS 構築時の自動最適化のログによると，乗算と加減算とを別々に3段のパイプラインにした時の CPU のピーク性能は，ERATO-1 では 446.54MFlops であり，倍精度浮動小数点のピーク性能はその 70.50% である 332.66MFlops となっている．従って，得られた最大性能である 6.76 GFlops は，理想的なピーク性能である 10.6GFlops (= 332.66MFlops × 32) の 64% である．GEMM での最大性能である 7.68GFlops はピーク性能の 72% となる．

5. 考　察

本稿で得られた知見と今後の研究課題をまとめる．

(1) コモディティハードウェアやソフトウェアを用いて構築した Beowulf クラスタで所望の性能を引き出すためには，NetPIPE に代表されるネットワーク性能測定ツールなどを用いて，異なるプロトコルレベルでの性能評価を総合的に行うことが不可欠である．

(2) CPU チップの性能向上に見合った性能向上を達成するためには，通信周りの性能向上が不可欠である．例えば，Gigabit Ethernet，Myrinet か，TCP ではなく VIA (Virtual Interface Architecture) か，複数の NIC をチャンネルボンディングによってバンド幅を向上させるのか等，いろいろな選択肢が考えられる．

(3) FreeBSD のネットワーク制御は、Linux より堅牢であり，性能がよいと言われている．FreeBSD 上で Beowulf クラスタの構築を ERATO-1 で試みたが，DEC Tulip チップを使用した NIC に対する適切なドライバが提供されていなかったので，失敗に終わった．

(4) ScaLAPACK は計算重視型のプログラムであり，通信重視型の応用では本稿で得られた性能よりは劣化する．ERATO-1 の主たる応用は，遺伝アルゴリズムの高速化であり，各遺伝子集団をほぼ独立に分散処理できるので，計算重視型とみなせる．実際，遺伝アルゴリズムの並列化による高速化は，ほぼノード数に比例した線形の速度向上を達成している[6]．

(5) 並列化による速度向上に加えて，大規模な行列が計算できるという側面も応用研究者には重要である．

6. おわりに

本稿では，Beowulf クラスタのチューニングと性能評価の手法を提案し，実際に ERATO-1 に適用し，その有効性を検証した．コモディティハードウェアやソフトウェアを使用して「お手軽に」クラスタが構築できるようにするためには，チューニングや性能測定が重要である．従来の研究発表では，個別の性能評価ツールや手法が中心であり，総合的な性能評価手法やチューニングが行われて来なかった．お手軽クラスタなら，ScaLAPACK あるいは LAPACK がお手軽に使えないと，ユーザの立場からは決してお手軽とはならないのではないであろうか．本稿で提案した手法が，コモディティチューニングの最初の一歩となれば幸いである．

謝辞 Beowulf クラスタのベンチマーク方法をご教示いただいたベルギー Mons-Hainaut 大学の Scalmani Giovanni 博士，ScaLAPACK の G77 ベンチマークプログラムを提供いただいた韓国の Yoon Ho 氏にする．ERATO-1 について日頃から議論をいただいた科学技術振興事業団北野共生システムプロジェクトの中川友紀子氏，諸橋峰雄氏，山口あづさ氏に感謝する．

参考文献

1) Becker, D.J., *et al.*: BEOWULF: A parallel workstation for scientific computation, *Proc. of Int. Conf. on Parallel Processing*, 1995.
2) Blackford, L.S., *et al.*: Installation Guide for ScaLAPACK, V1.5, LAPACK Working Note 93, Univ. of Tennessee, Knowxville, '97.
3) Geist, A., *et al.*: *"PVM: Parallel Virtual Machine. "*. MIT Press, 1994.
4) 森, 富田: 並列計算機アーキテクチャからみた計算機クラスタ．「情報処理」, 39(11),'98, 1073–1077.
5) M.P.I. Forum: MPI: A message passing interface standard, *Int. J. of Supercomputer Appl. and HPC*, vol.8, 3–4, Special Issue on MPI, '94.
6) Okuno, H.G., *et al.*: Initial Assessment of ERATO-1 Beowulf-Class Cluster. Ito, T. and Yuasa, T. (eds.) *Proc. of Parallel and Distributed Computing for Symbolic and Irregular Applications*, World Scientific Pub., *in print.*
7) Reschke, C., *et al.*: A Design Study of Alternative Network Topologies for the Beowulf Parallel Workstation, *Proc. of HPDC*, '96.
8) Snell, Q. O., *et al.*: NetPIPE: A Network Protocol Independent Performance Evaluator.
9) Sterling, T., Salmon, Becker, D.J., and Savarese, D.: *"How to Build a Beowulf: A Guide to the Implementation and Application of PC Clusters"*. The MIT Press, 1999.