1CD UNIX を用いた PC クラスタシステムの構築と性能改善に関する研究

山口広行*・角田亮**・川口潤**

Construction and Performance Improvement of PC Cluster System by using 1CD UNIX

Hiroyuki YAMAGUCHI*, Akira KAKUTA** and Jun KAWAGUCHI**

Abstract

By applying 1CD UNIX to client computers, we easily construct a PC cluster system (high performance computing system). The special feature of this system is not to influence existing PC environments. We evaluate the performance of this system by using the Himeno benchmark program. In order to improve the performance of this system, we tune up the TCP parameters and change switching hubs. When we use a switching hub corresponding to 1 Gbps, we show that the performance of this system is improved and achieves about 2GFLOPS by using 4 PCs.

Keywords: PC cluster system, 1CD UNIX, performance improvement

1. はじめに

現代の科学技術の発展に，コンピュータは必要不可欠な存在となっている。その中でも，科学技術計算（コンピュータ・シミュレーション）の分野はコンピュータと共に発展し，現在では理論・実験に次ぐ第三の研究手法としての位置を確立している。

大規模な科学技術計算に用いられる超高性能コンピュータは，スーパーコンピュータと呼ばれる。1990 年代までは，ベクトル・プロセッサと呼ばれる特殊な演算装置を複数用いて，並列処理を行う方式がスーパーコンピュータの主流であったが，パーソナルコンピュータ（以下 PC）などの小型コンピュータ向けマイクロ・プロセッサの飛躍的な性能向上と低価格化を受けて，近年では多数のマイクロ・プロセッサを接続して，並列処理を行う方式のスーパーコンピュータが増えている。世界中のスーパーコンピュータの性能ランキング（スパコン TOP500）は，インターネット上で公開されている[1]。

PC クラスタは，現在のスーパーコンピュータで使われている方式を PC に適用することで，手軽に高速・高性能な計算機を構築する方法として，注目を集めている。スーパーコンピュータは最先端の技術を結集して開発されるため，性能だけでなく価格も他のコンピュータとは比較にならないほど高価なのに対し，PC クラスタは市販製品のハードウェアと無料のソフトウェア（フリーウェア）を組み合わせて構築できるため，構築コストを安くできるのが大きな特徴としてあげられる。しかしながら，専用の PC クラスタシステムを構築・維持することは，科学技術計算を専門に行う研究者以外にはメリットが少ない。そこで，本研究では 1CD UNIX に着目した。1CD UNIX は，PC のハー

平成 17 年 12 月 16 日受理
* システム情報工学科・講師
** システム情報工学科・4年

ドディスクを使わずに CD から直接起動・動作するため，既存の PC 環境に全く影響を与えないのが特徴である。そのため，例えば Windows がインストールされた PC 環境を変更することなく，PC クラスタシステムとしても利用することが可能となる。そこで，1CD UNIX を用いて，手軽に PC クラスタシステムを構築する手法を確立することを，本研究の目的の 1 つとする。1CD UNIX としては，Debian と呼ばれる Linux ディストリビューションを元にした KNOPPIX[2] が有名であるが，本研究では，FreeBSD を元にした FreeSBIE[3] と呼ばれる 1CD UNIX を利用した。

スーパーコンピュータの場合は，通常，製造メーカからコンピュータの性能値が提供されるが，PC クラスタを自力で構築した場合は，その性能値も自力で把握する必要がある。さらに，十分な性能値が得られない場合は，性能の改善も行う必要がある。そこで，本研究では，構築した PC クラスタの性能評価，ならびに性能改善方法を研究することを，2 つめの目的とする。

2. 実験方法

ここでは，PC クラスタシステムの構築方法，ならびに性能評価・改善方法について述べる。

2.1 PC クラスタシステムの概要と構築方法

本研究で用いた，PC クラスタシステムの概念構成図を図 1 に示す。PC クラスタシステムは，並列計算の処理を司るサーバ PC と，サーバの指示に基づき計算処理を行う複数台のクライアント PC から構成される。そして，各 PC をスイッチに接続することで，1 つのシステムを形成する。

表 1 には，本研究で用いた PC の主要スペックとインストールしたソフトウェアを示す。本研究では，サーバのみ専用の PC を用意し，OS と並列計算用のライブラリをインストールした。一方，クライアント PC は 1CD UNIX を用いたので，ソフトウェアのインストールは行わなかった。

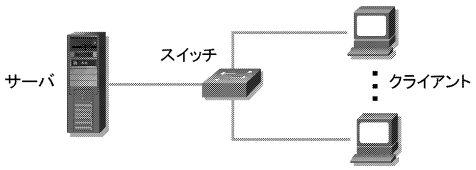

図 1　PC クラスタシステムの概念構成図

表 1　PC スペックとインストールしたソフトウェア

項目	サーバ	クライアント
CPU	Pentium4 3.0 GHz	Pentium4 2.8 GHz
メモリ	1 GB	512～768 MB
OS	FreeBSD5.3	なし（※ 1）
並列計算用ライブラリ	MPICH1.2.6	なし（※ 2）

（※ 1）オリジナル 1CD UNIX（FreeSBIE）を利用
（※ 2）NFS によりサーバ上のファイルを共有

【サーバの構築方法】

本研究では，専用のサーバ PC を用意し，FreeBSD と呼ばれる OS をサーバ PC にインストールした。FreeBSD では，対話形式でインストール作業を進めるインストーラが用意されているため，比較的容易にインストール作業を行うことが可能である。またインストール作業の中で，IP アドレス等のネットワーク設定や並列計算用のユーザ作成も行った。

次に，並列計算用のライブラリとして MPICH[4] をサーバ PC にインストールした。FreeBSD では，アプリケーションのインストール・アンインストール・管理を用意にする，ports と呼ばれる仕組みが存在する。MPICH も ports に登録されているため，インストール作業は ports を利用して行った。FreeBSD のインストールや ports の利用方法の詳細は，文献[5] 内のハンドブック等に記述されている。

ソフトウェアのインストール後に，並列計算

で必要となる rsh（Remote SHell）と NFS（Network File System）サーバの設定を行った。PC クラスタでは，PC 間の通信や命令実行のためにリモートのコンピュータを制御する必要があり，そのために rsh，または ssh（Secure SHell）サービスを利用する。rsh は，ssh よりも処理速度が速いのが長所であるが，セキュリティが考慮されていないのが欠点である。一方，ssh はデータを暗号化するためセキュリティ面では優れているが，暗号化処理を行うため処理速度が rsh よりも劣る。本研究では，PC クラスタとして利用する場合に，ネットワークの設定をインターネットと通信しないようにしたので，処理速度を優先して rsh を利用した。

NFS はファイル共有に関する標準的なシステムである。本研究では，並列計算用のライブラリ（MPICH）とプログラムの保管場所を，NFS によって共有する設定を行った。Linux での rsh や NFS の基本的な設定方法は，文献[6] に記述されている。

【クライアントの構築方法】

本研究では，クライアント PC で FreeSBIE と呼ばれる 1CD UNIX を利用した。FreeSBIE は，実際に動作している FreeBSD 環境をベースにして CD イメージを作成するため，本研究では FreeBSD をインストールした CD 作成用の PC を別途用意した。FreeSBIE は，カスタマイズを必要としない場合は，FreeBSD のインストーラと同様，メニュー形式で容易に 1CD UNIX を作成できる。しかしながら，本研究では並列計算を行うために，以下のカスタマイズを施した。

まず，FreeSBIE のデフォルト設定では，管理者権限を持つ root ユーザしか作成されない。一方，root ユーザでは前述の rsh が利用できないため，並列計算を行うには root 以外のユーザを作成する必要がある。そこで本研究では，root 以外のユーザを作成するために，カーネルの設定を変更した。さらに，CD イメージを作成する前に並列計算用のユーザを作成した。次に，FreeSBIE のデフォルト設定では，IP アドレスを自動的に取得する DHCP（Dynamic Host Configuration Protocol）や DNS（Domain Name System）を利用する設定になっている。本研究ではこれらの機能を利用しなかったため，無効にする設定を行った。

並列計算で必要となる rsh については，サーバと同様の設定を施した。そして，ファイル共有に関する NFS については，クライアント用の設定と起動時にサーバの共有ファイルを自動的にマウントする設定を施した。

1 枚目の CD 作成は，FreeSBIE 用のシステムやカーネルの構築を行うため多くの時間が必要となるが，2 枚目以降の作成はクライアント毎に異なる，ホスト名や IP アドレスの変更だけで済むので，短い時間で CD を作成することが可能である。

2.2 PC クラスタシステムの性能評価・改善方法

本研究では，構築した PC クラスタシステムを評価するために，ベンチマークプログラムを用いた。ベンチマークプログラムとは，計算機システムの性能を計測するためのプログラムで，並列計算機用としては，NASA が開発した NPB[7] や理化学研究所の姫野氏が開発した姫野ベンチマーク[8]，そしてスパコン TOP500 の性能評価でも利用されている LINPACK/ScaLAPACK[9] が有名である。本研究では，姫野ベンチマークを用いて性能評価を行った。姫野ベンチマークは，ポアッソン方程式をヤコビの反復法で解く場合の主要ループの処理速度を計測するプログラムである。簡単にコンパイルと実行ができ，即座に処理性能を FLOPS（1 秒間あたりの浮動小数演算数）の数値で得ることができる。

PC クラスタシステムの性能改善方法は，大きく次の 3 つに分類できる。

① 計算プログラムの変更
② 並列環境の設定変更

③　並列環境のハードウェア変更

ここで，① は性能改善の効果が最も表れやすい方法だが，今回はベンチマークプログラムを利用するため，改善方法の対象外とした。

② の方法としては，MPICH が PC 間の通信に TCP を利用することから，TCP 関連の設定変更が考えられる。また，NFS によるファイル共有を行うため，NFS 関連の設定変更も改善方法として考えられる。ただし NFS の設定変更による性能改善は，ターンアラウンドタイム（計算開始から終了までの時間）の短縮に対しては効果があると考えられるが，姫野ベンチマークで計測する処理性能に及ぼす影響は小さいと考えられる。また，NFS の設定変更を行うと，クライアント用 CD を再度作成する必要も生じる。このため本研究では，TCP 関連の設定変更が処理性能へ及ぼす影響のみを調べた。

③ の方法としては，CPU，メモリ，マザーボードなど PC を構成するハードウェア部品を変更する方法が考えられるが，本研究では PC クラスタ専用のシステムを構築することが目的ではないため，これらを改善方法の対象外とした。一方，各 PC を接続するスイッチやハブ，そして各 PC のネットワークカードは，通信速度が 10 Mbps の製品から 1 Gbps の製品まで数多く存在し，低価格・高性能化も著しい。そこで本研究では，PC を接続するスイッチやハブを変更し，処理性能へ及ぼす影響を調べた。

3.　実験結果と考察

3.1　PC クラスタシステムの性能評価

図 2 に，姫野ベンチマーク（S サイズ）を用いて，構築した PC クラスタシステムの性能を測定した結果を示す。ここで図 2 の破線は，PC1 台の処理性能を基準として，処理性能が PC の台数に比例すると仮定した場合の性能値（理想値）を表している。

図 2 より，PC の台数増加に伴い，PC クラスタシステムの処理性能が向上することが分かる。一方，測定した処理性能値は，PC の台数に比例するとした理想値に比べ大幅に低いことも分かる。

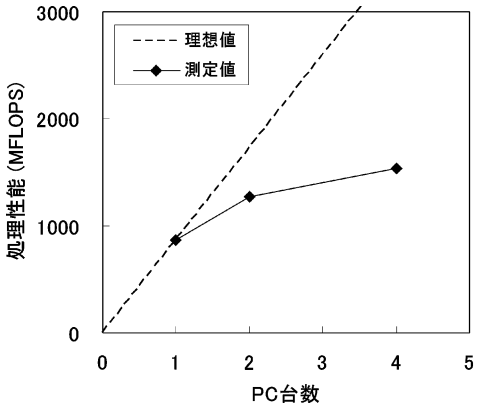

図 2　姫野ベンチマークによる性能測定結果

この測定結果は，アムダールの法則を用いて説明できる。アムダールの法則では，まず，プログラムが並列化可能な部分と，並列化不可能な（逐次処理）部分から構成されると考える。そして，このプログラムを N 台の PC を用いて処理すると，並列化可能な部分の実行時間は $1/N$ になるが，並列化不可能な部分が存在するため，全体の実行時間は $1/N$ にならないというものである。1 台の PC と N 台の PC での実行時間の比は速度向上率 S と呼ばれ，次式で表される。

$$S=\frac{1\text{台での実行時間}}{N\text{台での実行時間}}=\frac{1}{(1-P)+P/N} \quad (1)$$

ここで，P はプログラム全体に対する並列化可能な部分の割合を示す。(1)式において $N\to\infty$ とすると，$S\to 1/(1-P)$ となる。これは，たとえば $P=0.9$（プログラム全体の 90% が並列化可能）の場合，PC の台数をどんなに増やしたとしても，10 倍以上の速度向上は得られないことを示している。

図 2 の測定結果から，速度向上率 S と並列化可能な部分の割合 P を求めた結果を，表 2 に示す。姫野ベンチマークは計算時間が約 1 分とな

表 2　速度向上率 S と並列化可能な部分の割合 P

PC 台数	S	P
1	1.00	−
2	1.47	0.64
4	1.77	0.58

るように計算の繰返し回数を調節するため，ここでは実行時間を計算時間÷繰返し回数として，S と P の値を算出した。

表 2 の結果から，PC の台数増加に伴い速度向上率は増加するが，その値は PC の台数に満たないことが分かる。また，同じプログラムを利用しているのにも関わらず，並列化効率 P が 2 台よりも 4 台の場合の方が小さいことも分かる。

PC の台数によって並列化効率 P が異なるのは，並列化不可能な（逐次処理）部分に，PC 間の通信時間と同期待ち時間が含まれるためと考えられる。通信時間と同期待ち時間を減らすには，計算プログラムを改善するのが最も効果的な方法であるが，前述の通り本研究では計算プログラムの改善を対象としていない。そこで，本研究では並列環境に関する設定とハードウェアの変更によって，PC クラスタシステムの性能改善を目指した。

3.2　PC クラスタシステムの性能改善

ここでは，TCP 関連の設定変更と，各 PC を接続するスイッチ/ハブの変更が，処理性能へ及ぼす影響を調べた。

【TCP の設定変更による性能改善効果】

MPICH では，PC 間の通信に TCP を利用することから，TCP の設定を変更して処理性能を測定した。MPICH のドキュメントには，TCP の設定項目として以下の 7 項目が挙げられている。

- bufsize : ソケット通信のバッファサイズ
- winsize : ウィンドウサイズ

表 3　TCP のバッファサイズと処理性能

バッファサイズ（KB）	性能比	備考
4	0.92	
8	1.00	
16	−	default 値
32	1.00	
64	1.00	
128	1.00	

- netsendw : 出力処理の処理待ち機能
- netreadw : 入力処理の処理待ち機能
- writev : データと MPI の情報を 1 つのパケットで送る機能
- readb : ビジー状態でのブロッキング機能
- stat : 統計情報の出力機能

本研究では，winsize と stat を除く 5 項目について，設定変更の効果を検証した。なお，この実験は表 1 のスペックを持つ PC 2 台（サーバ 1 台，クライアント 1 台）で行った。

まず，TCP のバッファサイズ（KB）を変更して処理性能を測定し，デフォルトのバッファサイズ（16 KB）での処理性能に対する比（性能比）を求めた結果を，表 3 に示す。

表 3 の結果から，バッファサイズが 4 KB の場合に処理性能の低下が見られるが，それ以外では処理性能に変化がないことが分かる。

この結果は，MPICH で交換するデータ長から解釈できる。本研究では，MPICH において PC 間で交換されるデータを実際にキャプチャし，その内容を解析した。その結果，イーサネットの MTU サイズ（1,514 バイト）で通信を行う場合が多いことが分かった。MTU サイズではデータ長が 1,448 バイトとなるため，バッファサイズが 4 KB の場合は，数個のデータを連続して受信しただけで，バッファ溢れが発生する可能性が高い。バッファ溢れによりデータが廃棄されると，TCP の再送機能によりデータが再送されるため，通信時間が増大し，その結果と

表 4　TCP の各種設定と処理性能

	net sendw	net readw	writev	readb	性能比
de-fault	Y	Y	Y	N	−
1	N	Y	Y	N	1.00
2	Y	N	Y	N	1.00
3	Y	Y	N	N	1.00
4	Y	Y	Y	Y	1.00

表 5　利用したスイッチとハブ

	I	II	III
メーカ	CentreCom	Cisco	NetGear
機種	MR820TR	Catalyst 2950	JGS516JP
ポート速度	10 Mbps	100 Mbps	1 Gbps
通信方式	半二重	全二重	全二重

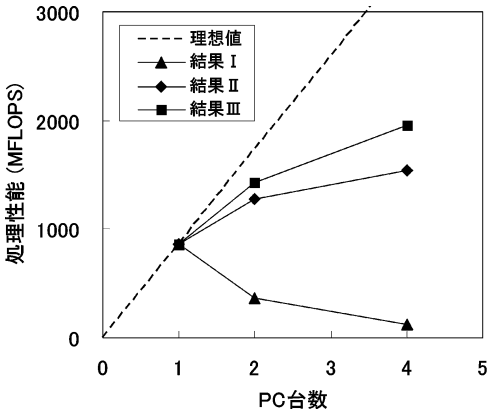

図 3　スイッチ/ハブの種類と処理性能

表 6　スイッチの通信性能測定結果

	II	III
ポート速度	100 Mbps	1 Gbps
通信性能（※）	43.84 Mbps	84.48 Mbps

（※）1,024 バイト近傍のランダム長データを転送

して処理性能が低下すると考えられる。

次に，TCP のバッファサイズ以外の 4 つの機能をそれぞれ変更して処理性能を測定し，デフォルト値に対する性能比を求めた結果を，表 4 に示す。ここで，Y/N は機能の有効/無効をそれぞれ表している。この結果から，各機能の有効/無効に関わらず，処理性能に変化が見られないことが分かる。

【スイッチ/ハブの変更による性能改善効果】

本実験で用いた，スイッチとハブの一覧を表 5 に示す。ここで半/全二重方式とは，データの送信と受信が同時にできない/できることを，それぞれ示している。

表 5 の機器を利用して，処理性能（MFLOPS）を測定した結果を図 3 に示す。この結果から，100 Mbps のスイッチ（結果 II）よりも，1 Gbps のスイッチを利用した方（結果 III）が処理性能は向上し，4 台の PC で 2GFLOPS 程度の性能を得ることができた。一方，10 Mbps のハブを利用した場合（結果 I）は，PC1 台の処理性能よりも低下することが分かる。この結果は，ネットワークの通信性能が PC クラスタシステムの性能に大きな影響を与えることを示している。

図 3 の結果 I は，通信速度だけでなく，半二重通信の影響も大きいと考えられる。半二重通信では，同一ネットワーク内で通信できる機器が 1 台に限定されるため，PC の台数が増えるほど通信時間が増大する。これが，処理性能を低下させる主な原因と考えられる。

図 3 の結果 III は，通信性能の高いネットワーク機器を利用することで，処理性能が向上することを示している。ただし，スイッチの通信性能は一般的にカタログスペックと異なるので，注意が必要である。本研究で用いたスイッチ（表 5 の II と III）について，通信性能を実際に測定した結果を表 6 に示す。この測定結果から，II と III のスイッチの通信性能の差は，ポート速度の差（10 倍）よりもはるかに小さい（2 倍

程度）ことが分かる。表 6 のように，不連続長のデータを交換する場合は，通信性能がカタログスペックと異なる場合が多いので，スイッチを選定する場合には注意が必要である。

それでは，Myrinet 等の通信性能のさらに高いネットワーク環境を利用すれば，PC の台数が増加しても通信時間は短くなるのだろうか。MPICH の通信形態を考慮すると，それには限界があると考えられる。まず MPICH では，1 台のサーバ PC が複数台のクライアント PC と通信するため，サーバ PC に通信データが集中する傾向がある。また，MPICH では 1 対多通信を行う関数が用意されているが，実際に通信データをキャプチャすると，1 対 1 通信（ユニキャスト通信）を順次実行しているに過ぎなかった。つまり，MPICH で実装されている 1 対多通信用の関数は，プログラマの記述量を削減するのが目的であり，データ通信量を削減することが目的ではないことが分かる。以上のことから，クライアント PC の台数が増加すると，通信性能の高いネットワーク環境であっても，通信時間は増大すると考えられる。

もし，MPICH でユニキャスト通信ではなく，データ通信量を削減できる 1 対多通信（マルチキャスト通信やブロードキャスト通信）を利用した関数を実装することができれば，PC クラスタの処理性能は飛躍的に向上すると考えられる。そこで，1 対多通信を利用した関数を開発することが，次のテーマと考えている。

4. ま と め

本研究により，次の結果が得られた。

1 つめは，1CD UNIX を用いることで，既存の PC 環境に影響を与えることなく，手軽に PC クラスタシステムを構築できることが分かった。

2 つめは，姫野ベンチマークを利用した場合，MPICH の TCP 設定はデフォルト設定でも最適な処理性能が得られること，また高速なスイッチを利用することで，処理性能が向上することが分かった。実際に，1 Gbps 対応のスイッチに 4 台の PC を接続して，2GFLOPS 程度の処理性能を得ることができた。

1 つのマザーボードに複数の CPU を搭載した PC も普及しつつあるため，並列処理は今後益々身近になると考えられる。本研究の発展が，PC クラスタだけでなく並列処理の発展につながればと考えている。

参 考 文 献

1) http://www.top500.org/
2) http://www.knopper.net/knoppix/
3) http://www.freesbie.org/
4) http://www-unix.mcs.anl.gov/mpi/
5) http://www.jp.freebsd.org/www.FreeBSD.org/ja/
6) 樫山和男，西村直志，牛島　省：並列計算法入門（丸善，2003）
7) http://www.nas.nasa.gov/Software/NPB/
8) http://accc.riken.jp/HPC/HimenoBMT/
9) http://www.netlib.org/scalapack/