社団法人 情報処理学会 研究報告
IPSJ SIG Technical Report

2007－ARC－172 (29)
2007－HPC－109 (29)
2007／3／2

コモディティネットワークによる5GB/s通信の可能性

松葉浩也[†] 石川 裕[†,††]

東京大学、筑波大学、京都大学の計算機センターは共同で次世代のスーパーコンピュータである「オープンスーパーコンピュータ」の仕様を策定している。この仕様はコモディティーハードウェアを用いてスーパーコンピュータを構成することを求めており、ノード間通信の性能として5GB/s以上の物理性能、4GB/s以上のMPI性能を求めている。本稿ではこの仕様を満たすために、10Gbpsクラスのネットワークをトランキングするライブラリの設計と実装について述べる。また、このライブラリを用いてネットワークバンド幅の測定を行い、NUMAアーキテクチャの特性を考慮すれば、上記の性能が達成可能であることを示す。

Feasibility of 5GB/s Communication Using Commodity Network

Hiroya Matsuba[†] and Yutaka Ishikawa[†,††]

The University of Tokyo, University of Tsukuba, and Kyoto University are making the specification of "Open Supercomputer", that is the common specification of next generation supercomputers installed in computer centers of these universities. This specification requires that the supercomputer is composed of commodity hardware and that the bandwidth of internode communication must be 5 GB/s or more in physical bandwidth and 4 GB/s or more in MPI level bandwidth. This paper shows the design and implementation of an MPI library to aggregate multiple network links in order to meet the requirement. The performance evaluation of this MPI library shows that it meets the bandwidth requirement if it takes the characteristic of NUMA architecture into the consideration.

1. はじめに

東京大学情報基盤センターでは筑波大学、京都大学と共同で「オープンスーパーコンピュータ」と呼ばれる次世代スーパーコンピュータの共通仕様を策定している。この次世代スーパーコンピュータは、PCクラスタから移行するユーザーの利便性のため広く使われている IA32(64bit 拡張) アーキテクチャの計算機上でオープンソースベースのシステムソフトウェアを用いる。一方で、10 ペタフロップス・スーパーコンピュータへの橋渡しとして、十分な性能を確保するため、本仕様はノード間接続に5GB/s以上の物理性能、4GB/s以上のMPI性能を求めている。その他の仕様を含めたオープンスーパーコンピュータ全体の資料招請時の仕様を次ページの**表1**に示す。

本稿ではオープンスーパーコンピュータの仕様の中で特徴的である、物理 5GB/s 以上、MPI レベルで4GB/s以上のバンド幅について、コモディティーハードウェアにおける実現可能性を検討する。このバンド幅は Gigabit Ethernet の 40 倍であり、単一のネットワークリンクでこの性能を満たすネットワークカードは現時点では存在しない。そこで、我々は 10Gbps 以上のネットワークを複数リンク束ねることでこの性能を実現することを検討している。本稿では MPI レベルで複数リンクのトランキングを行う手法を採用し、実際にトランキングに対応した MPI ライブラリを設計、実装する。

一方、今日では複数の Opteron または Power プロセッサを搭載したコンピュータに見られるように、プロセッサから見たときのメモリの性能が均一でない NUMA 構成の計算機が広く使われるようになってきており、この構成は今後の主流になることが予想される。本稿の性能評価実験においては、NUMA 構成のノードにおいて、ネットワークカードとメモリとの距離が性能に及ぼす影響を明らかにする。そして、NUMA 構成を考慮し、ネットワークカードとメモリの配置を最適化した場合には、オープンスーパーコンピュータの要求要件である「MPI レベルにおける 4GB/s 以上」が実現可能であることを示す。

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

表 1　オープンスーパーコンピュータの仕様案

CPU アーキテクチャ	IA32 64bit 拡張
ノード性能	160GFlops 以上
メモリ	32GB/node
メモリバンド幅	40GB/s/node
ローカルディスク	250GB/node 相当以上
ノード間バンド幅	物理 5GB/s,MPI で 4GB/s
ノード間遅延	MPI で往復 6μs 以下
OS	UNIX(Linux が望ましい)

2. MPI によるトランキング

本章では MPI におけるトランキングの方法について述べる。使用する MPI ライブラリは YAMPI[8] であり、10Gbps クラスのネットワークとして 20Gbps の Infiniband DDR を使用する。Infiniband のソフトウェアスタックには OpenIB[6] を用いる。

YAMPI のソフトウェアアーキテクチャを**図 1** に示す。図 1 に示される通り、YAMPI は複数のデバイスをサポートできるよう設計されており、デバイスに依存する部分を"P2P Layer"と呼んでいる。今回はまず、これまで未実装であった OpenIB のための P2P Layer を作成した。Infiniband を MPI から使用する際の問題とその解決方法に関しては文献 3) で議論されており、YAMPI に関しても特別の工夫を要する事項は存在しない。従って本稿では詳細を省略し、バンド幅の議論において重要となる大きなサイズのメッセージの通信におけるランデブープロトコルと RDMA の使用に関してのみ概略を述べる。

2.1 ランデブー通信と RDMA

MPI における通信では、プロセッサによるデータのコピーを避けるため、受信側でバッファが確保されていることを保証してから送信を行う「ランデブープロトコル[1]」が用いられることがある。Infiniband のようにリモートメモリアクセス (RDMA) をサポートするデバイスでは、ランデブープロトコルと RDMA 機能を組み合わせることによって、CPU によるメモリコピーを完全に排除したゼロコピー通信が可能となる。YAMPI においてゼロコピー通信を行う場合のプロトコルの概要は以下のようになる。

(1) メッセージの送信要求が出されると、Request Layer が受信側に対しメッセージのサイズとマッチング情報 (MPI における tag, communicator, 送信者のランク) を送信する
(2) P2P Layer は上記のメッセージを送信した後、送信バッファの仮想アドレスと物理アドレスを固定する操作をカーネルに要求し、ネットワークカード内にページテーブルを準備する。(メモリレジストレーションと呼ばれる)。
(3) 受信側の Request Layer はマッチする受信命令が発行され次第、送信側に準備完了を伝える。

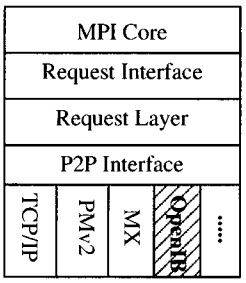

図 1　YAMPI のソフトウェアアーキテクチャ

(4) 受信側の P2P Layer は上記メッセージを送る前に受信バッファをレジストレーションし、そのキーを準備完了メッセージに追加する
(5) 送信側の Request Layer がデータ本体の送信を P2P Layer に指示する。P2P Layer は RDMA によってデータ送信を行い、さらに RDMA 完了メッセージを送信する。
(6) 受信側の P2P Layer が RDMA 完了メッセージを受信し次第、すべての操作の終了を上位層に報告する。
(7) メモリレジストレーションを解除する

上記の手順中、メモリレジストレーションはシステムコールとデバイスへのアクセスを伴うコストの大きい作業である。そこで実装上は通信毎にメモリレジストレーションを解除することは行わず、後に同じ領域で通信が行われることを期待してキャッシュしておくことが多い。このレジストレーションキャッシュは多くの MPI で実装されており、YAMPI にも実装した。

2.2 トランキング

先に述べたランデブー通信においては、送受信手続きにわずかな変更を加えるのみで、複数リンクのトランキングが可能となる。具体的な変更は以下の 3 点である。

- すべてのネットワークカードにおいてメモリレジストレーションを行うようにする
- データ本体の通信時には通信領域をネットワークカードの数で等分し、それぞれのカードに送信要求を出す
- RDMA 完了メッセージはそれぞれのネットワークカードで送信する。受信側はすべてのカードにおける RDMA 完了メッセージの受信をもって通信作業の終了とみなす。

この手法を YAMPI の OpenIB レイヤに実装した。次々章で性能測定を行う。

3. NUMA

NUMA 構成のコンピュータをネットワークに接続する際には、ネットワークカードから見たときのメモ

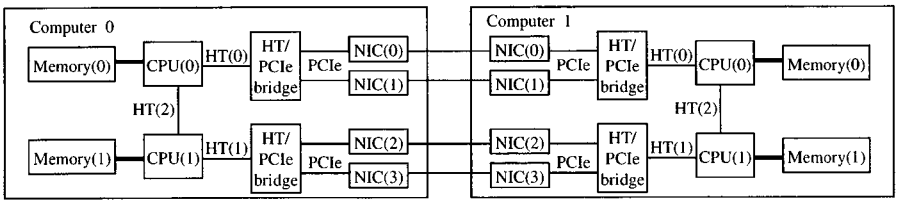

図 2 NUMA 構成におけるネットワーク接続

リ性能も不均一となるため、通信を行う際もこの不均一性を考慮に入れて最適化を行う必要がある。ここではオープンスーパーコンピュータの仕様を満たし、現在入手可能な NUMA アーキテクチャの計算機として Opteron プロセッサを搭載した計算機を例に、その構成を**図 2** に示す。この図では CPU を 2 個搭載したコンピュータ 2 台が 4 本のネットワークによって接続されている。NIC(0) または NIC(1) から見た場合、Memory(0) は最短距離にあるが、Memory(1) には HT(2)(Hyper Transport) を経由してアクセスする必要がある。このため、Memory(0) にあるデータを通信する場合と比較して Memory(1) にあるデータを通信する場合は性能が低下することが考えられる。

NUMA を考慮した最適化を行うために、OS は様々な機能を提供している。例えば Linux カーネルにおいては、sched_setaffinity システムコールによりプロセスが動作する CPU を固定することができる。また、set_mempolicy システムコールにより、プロセスが使用する物理メモリの位置を固定することができる。次章の性能測定においては、上で紹介した Linux による NUMA 向け最適化機能を使用し、NUMA 構成を考慮に入れた性能測定を行う。

今後の議論では NIC(0) または NIC(1) から見た Memory(0) を「近いメモリ」、Mmory(1) を「遠いメモリ」と呼ぶことにする。NIC(2) または NIC(3) から見た場合 Memory(0) は「遠いメモリ」Memory(1) が「近いメモリ」となる。

4. 性能測定

本章では第 2 章で作成したトランキング機能を有する YAMPI/OpenIB を用いて性能測定を行い、4GB/s の実現可能性を検証する。

4.1 測定環境

性能測定には 2 個のデュアルコア Opteron プロセッサを搭載したコンピュータ 2 台を用いる。測定に用いるコンピューターの詳細な仕様を**表 2** に示す。使用するネットワークは 1 台あたり 4 本の Infiniband DDR であり、4 本それぞれをスイッチなしで直結している。

ここで使用したマザーボードは、前章で示した図 2 のように、2 個のプロセッサと 2 個の HyperTransport/PCI-Express ブリッジ（チップセット）がそれぞれ直結されている。同じ、nForce Professional 3000 シリーズを搭載したコンピュータでも、市場に出回っているほとんどのサーバ機は、シングルプロセッサでも使用できるよう片方のプロセッサから nForce Professional 3600 と 3050 の両方を接続しているので注意が必要である。また、適切にチューニングされた BIOS を使用することは非常に重要である。後に示すように BIOS のバージョンの違いにより 1GB/s もの性能差が生まれることがある。

表 2 測定用コンピュータの仕様

CPU	Opteron 2212 (Dual Core, 2.0GHz) x 2
Chipset	nVIDIA nForce Pro 3600 + 3050
Main Board	Tyan Thunder n6650W (S2915)
	BIOS version: 1.01D beta
Memory	DDR2-667 Dual Channel (2GB)
PCI	PCI-Express x8 + x8 from 3600
	PCI-Express x8 + x8 from 3050
Network	Mellanox InfiniHost III Lx (x4 DDR)
	Memfree(NIC にメモリを持たない)
OS	Fedora Core 6 (Linux 2.6.18)

4.2 測定プログラム

MPI の通信性能測定のためには Intel MPI Benchmark[2] を使用するのが一般的であるが、このベンチマークは NUMA を考慮していないため、実行毎に測定条件が異なってしまう。これを避けるため、本測定では NUMA を考慮した測定プログラムを作成した。測定プログラムの概要は**図 3** のようになっている。このプログラムは、通信に先立って測定プロセスが動作するプロセッサおよび通信用のメモリの物理的配置を固定している。なお、ここでは前章で紹介した NUMA 用のシステムコールを直接使うのではなく、libnuma のライブラリ関数を用いている。

バンド幅の測定のためには、メッセージを送信開始から受信側からの ACK を受け取るまでの時間を測定し、あらかじめ測定した小さなメッセージの片道通信時間（図中の C）を減じる。この減じる値は、メッセージサイズ 4byte で同じ図 3 のプログラムを走らせ、この補正を行わない状態 ($C = 0$) での結果を 2 で割った値である。以下では特に言及しないが、この C の測定はすべての測定に先立って行っている。

4.3 理論性能の検討

結果を示す前にこのシステムにおける理論性能を検討する。検討対象は、1) メモリバンド幅、2)Hyper-

```
benchmark
  (cpunode, memnode[], size, nr_nic){
 /* プロセスを特定の CPU に固定 */
 numa_run_on_node_mask(cpunode);
 buf = malloc(size);
 /* NIC 一枚当たりのサイズ */
 chunk = size / nr_nic;
 /* 物理アドレスが割り付けられる前に
    メモリの場所を指定する */
 for(i = 0; i < nr_nic; i++){
   numa_tonodemask_memory
    (buf+chunk*i, chunk, memnode[i]);
 }
 /* 物理メモリを割り付ける */
 memset(0, buf, size);
 /* 測定開始 */
 st = MPI_Wtime();
 if (node==0){
   for(i = 0; i < niter; i++) {
     MPI_Send(buf, size, ...);
     MPI_Recv(buf, 4, ...);
   }
 } else {
   for(i = 0; i < niter; i++) {
     MPI_Recv(buf, size, ...);
     MPI_Send(buf, 4, ...);
   }
 }
 ed = MPI_Wtime();
 /* C は前もって測定した片道遅延 */
 time = (ed - st)/ niter - C;
 bandwidth = size / time;
}
```

図 3 測定プログラムの概要

Transport のバンド幅、3)PCI-Express のバンド幅、4)Infiniband DDR のバンド幅である。1) のメモリは DDR2 667MHz のメモリを Dual Channel で使用しているため理論性能は 10.6GB/s である。2) の HyperTransport の理論性能は 6.4GB/s である。3) の PCI-Express については、実験に使用したネットワークカードが 8 レーンの PCI-Express 接続であるため、理論性能は 2Gbps×8 = 16Gbps = 2GB/s である。これは 8b/10b エンコーディングによるロスを考慮した値である。4) の Infiniband DDR の理論性能は 16Gbps である。これも 8b/10b エンコーディングのオーバヘッドを考慮した理論性能である。これらを総合すると 1 リンク当たりの理論性能は 2GB/s となる。ただし PCI-Express に関しては、実装によってはパケットヘッダのオーバヘッドが無視できないほどに大きくなる。今回実験に使用したチップセットの場合、PCI-Express の最大ペイロードサイズは 128byte でありヘッダは最大 24byte なので、実際に得られる性能は 2GB/s × 128/(128+24) = 1.68GB/s となる。今回の測定環境では、この性能が全体のボトルネックであり、これを基に計算すると 2 リンク、4 リンクの最大バンド幅はそれぞれ 3.37GB/s、6.74GB/s となる。

4.4 測定結果

4.4.1 1 リンクの基本性能

基本性能測定のため、4 本ある Infiniband のうちの 1 本のみを使用して性能を測定する。測定は送受信共にネットワークカードから近いメモリを使用する場合と遠いメモリを使用する場合について行った。近いメモリを使用する通信とは、図 2 において、Memory(0) → HT(0) → bridge → NIC(0) と送信され、受信側ではこの逆の経路で受信されることを意味する。遠いメモリを使った通信とは同じ図において、Memory(1) → HT(2) → HT(0) → bridge → NIC(0) と送信され、受信側ではこの逆の経路で受信される場合に相当する。

また、今回作成した YAMPI の OpenIB 層に性能バグが存在しないことを確認するため、結果を広く使われている MPI 実装のひとつである OpenMPI[7] における測定結果と比較する。**図 4** に結果を示す。

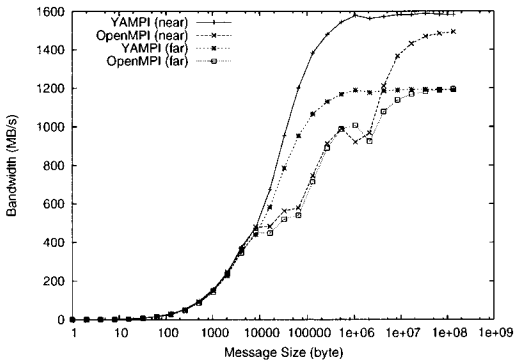

図 4 1 リンク使用時のバンド幅

遠いメモリ、近いメモリ、それぞれにおいて YAMPI は OpenIB よりも高速であり、YAMPI の実装に問題がないことが確認された。また、近いメモリを使用した場合 YAMPI は最大 1.6GB/s を達成しており、前節で検討した理論性能の 95%の性能が得られている。

注目すべき点は近いメモリと遠いメモリの性能差である。HyperTransport の理論バンド幅に問題がないにもかかわらず、遠いメモリを使用した場合には大きな性能低下が見られる。原因を調査するため、送受信の一方のみを遠いメモリにして性能を測定した結果を

図 5 に示す。送受信とも近いメモリまたは遠いメモリの結果も併せてプロットされているが、これは先に示した図 4 の YAMPI の結果の再掲である。

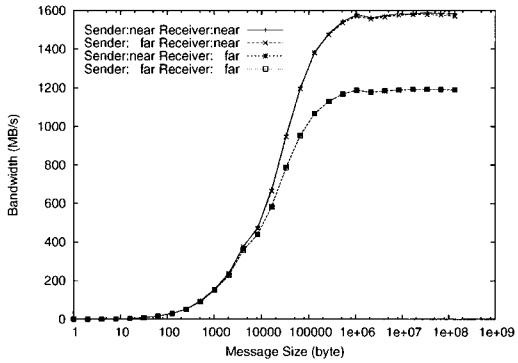

図 5　メモリとネットワークの距離と性能

図 5 の結果から、受信側におけるメモリとネットワークカードの距離が性能に影響しており、送信側の距離はほとんど問題にならないことがわかる。受信側においてのみ、メモリとネットワークカードとの距離が問題になる理由は現時点では不明である。

4.5　2 リンク使用時の性能

本節では 4 本のうちの 2 本を使用した場合の性能を測定する。まずは、高い性能が期待できる近いメモリを使用した場合について性能を測定する。2 本のリンクの選び方として、以下の 3 通りがある。

(1)　nForceProfessional（以下、NFP）3600 に接続された 2 本のリンクを使う
(2)　NFP3050 に接続された 2 本のリンクを使う
(3)　NFP3600 から 1 本、NFP3050 から 1 本を使う

これらについて測定結果を**図 6** に示す。比較のため、NFP3050 から 2 本を使用する場合については、現時点での最新 BIOS であるバージョン 1.01Dβ を使用した場合と、その前バージョンになるバージョン 1.01 を使用した場合の二通りについて測定を行う。なお、先に述べたようにトランキングが有効となるのはランデブー通信の時であり、本測定ではメッセージサイズが 32KB 以上のときのみである。

結果からわかるように、いずれの場合もほぼ同じ性能を示しており、ピークで 3.0GB/s である。これは先に検討した理論性能の 89%である。また、NFP3600 と NFP3050 の性能差はない。

ただし、この結果が得られたのは BIOS のバージョンが 1.01Dβ の時のみである。グラフ中の一番下の線に示されているように、バージョン 1.01 の BIOS では NFP3050 の性能が十分でなく、2 枚のネットワークを接続するのに十分な性能は得られていなかった。その時の性能は 2.1GB/s なので、BIOS のバージョンアップにより 42%も性能が改善したことになる。バージョン 1.01 の BIOS はチップセットを適切に設定していなかったものと推測される。

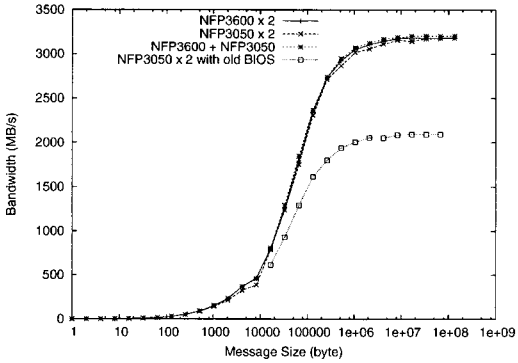

図 6　2 リンク使用時のバンド幅

次に、2 リンクを使用する場合において、遠いメモリを使用する代表的な例として以下の 3 パターンの性能を測定する。

(1)　NFP3600 に接続された 2 本のリンクを使う。そのうち 1 リンクは受信側で遠いメモリを使う。
(2)　NFP3600 から 1 本、NFP3050 から 1 本を使う。NFP3050 を使うリンクの受信側で遠いメモリを使う。
(3)　NFP3600 から 1 本、NFP3050 から 1 本を使う。両リンクとも送受信双方で遠いメモリを使う。

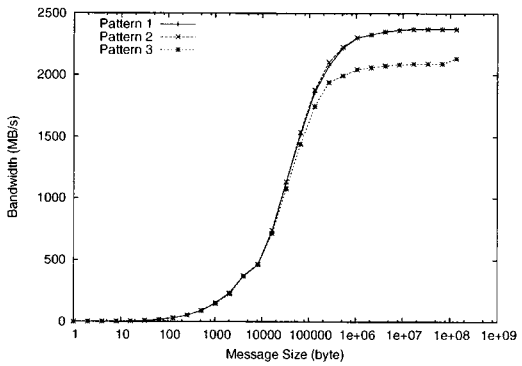

図 7　2 リンク使用時のバンド幅（遠いメモリを含む）

これらの結果を**図 7** に示す。MPI は送信データは均等量に分割し、すべてのリンクでの送信の終了を待っている。従って一番遅いリンクが律速となる。そのため、図 7 の結果は前節で示した 1 リンクで遠いメモリを含む場合の性能の約 2 倍となっている。3 番目のパターンは 2 リンクを使用する場合においては最悪の条

件であり、性能は 2.1GB/s にとどまっている。

4.6 4リンクの性能

最後に 4 リンクすべてを使用した場合の性能を**図 8** に示す。"Near"はすべてが近いメモリになるように配置した場合、"Far" は NFP3050 を使用する 2 リンクのうち 1 リンクの受信側のみを遠いメモリにした場合である。

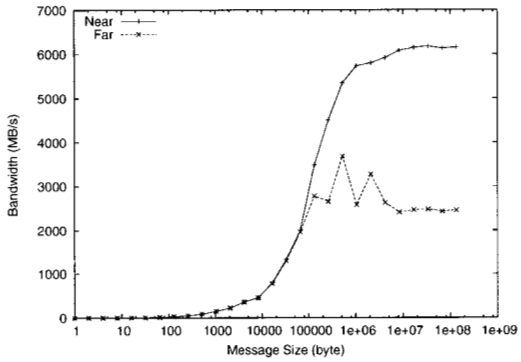

図 8 4 リンク使用時のバンド幅

すべて近いメモリを使用した場合のバンド幅は最大で 6.17GB/s である。これは理論性能の 91.5%であり、ほぼ期待される性能である。また、オープンスーパーコンピュータ要求要件である 4GB/s の 1.54 倍であり、余裕を持って要件を満たしていると言える。一方、1 リンクの受信側のみで遠いメモリを使用した場合、性能は安定せず最大でも 3.7GB/s にとどまっている。この結果から、最大限に性能を発揮させるためには NUMA を考慮し、近いメモリのみを使用して通信を行うことが必要不可欠であることがわかる。

5. 関連実装

米国オハイオ州立大学のグループによって開発されている MVAPICH は Infiniband を使用する MPI であり、トランキングをサポートしている [4]。また、OpenMPI[7] は、デバイスに依存しない層で複数ネットワークにメッセージを分割する機能を備えている。ただ、MVAPICH, OpenMPI 共に、手元の環境では複数のカードを使って高いバンド幅を得ることはできなかった。Infiniband を販売する Voltaire 社による Voltaire MPI もトランキングをサポートしているが、現時点ではプログラムを入手しておらず、動作を確かめることはできなかった。

6. おわりに

筑波大学、東京大学、京都大学が共同で策定中である次世代の「オープンスーパーコンピュータ」の要求要件である「物理 5GB/s、MPI レベルで 4GB/s 以上」の実現可能性の検討のため、本稿では Infiniband DDR のトランキングが可能な MPI ライブラリを作成し、その性能を測定した。その結果 4 本の Infiniband DDR を使用して 6.17GB/s の性能を達成し、上記の性能は達成可能であることを示した。またメモリ配置を考慮した通信実験により、4GB/s 以上の性能を達成するためには NUMA アーキテクチャを考慮した上でメモリの物理的配置を最適化する必要があることを示した。

今後の課題は 10Gbps クラスのネットワークとして Myrinet 10G[5] を使用して、MPI レベルでの 4GB/s 以上の実現可能性を検討することである。また、Infiniband DDR、Myrinet 10G 双方において通信遅延の評価を行う必要がある。また、NUMA アーキテクチャにおいてはアプリケーションに応じたメモリ配置を行うことも重要であり、アプリケーションが高速に動作するメモリ配置が通信に最適な配置と一致するとは限らない。従ってアプリケーションが行う計算と通信の双方を考慮し、ユーザーアプリケーションの実行速度を最大化する手法を検討することが重要となると考えられる。これらを検討した上でアプリケーションベンチマークの性能を評価することも今後の課題である。

参考文献

1) Gropp, W., Lusk, E., Doss, N. and Skjellum, A.: A high-performance, portable implementation of the MPI message passing interface standard, *Parallel Computing*, Vol. 22, No. 6, pp. 789–828 (1996).
2) Intel MPI Benchmark: . http://www.intel.com/cd/software/products/asmo-na/eng/219848.htm.
3) Liu, J., Wu, J., Kini, S., Wyckoff, P. and Panda, D.: High Performance RDMA-Based MPI Implementation over InfiniBand, *In proceedings of the ACM 17th annual international conference on Supercomputing* (2003).
4) Liu, J., Vishnu, A. and Panda, D.K.: Building Multirail InfiniBand Clusters: MPI-Level Designs and Performance Evaluation, *In proceedings of the ACM/IEEE SC2004 Conference*.
5) Myrinet: . http://www.myri.com.
6) OpenFabrics Alliance: . http://www.openib.org.
7) OpenMPI: . http://www.open-mpi.org.
8) 石川裕: YAMPII もう一つの MPI 実装, 情報処理学会研究報告 SWoPP'04 (2004).