「京」のための MPI 通信機構の設計

住元 真司[†] 川島 崇裕[†] 志田 直之[†]
岡本 高幸[†] 三浦 健一[†] 宇野 篤也[††]
黒川 原佳[††] 庄司 文由[††] 横川 三津夫[††]

本論文では 82,944 ノードの「京」上で使用メモリ量を極小化しながら MPI 通信性能を高める通信機構の設計について述べる。「京」が採用した Tofu インタコネクトは数十万ノードクラスのシステムで高い性能と耐故障性を実現するため直接網である 6 次元トーラス・メッシュ網を採用している。しかし、超大規模の直接網システムでは、通信ホップ数増加とネットワーク網でのメッセージ衝突による通信遅延の増加による通信性能の低下、ならびに、ノード数に比例して必要な使用メモリ量の増加が課題となる。この課題を解決するため、RDMA 通信を主体とし、通信バッファが必要な通信は隣接通信などよく利用される通信経路に絞る、遅延が大きな場合は省メモリ性を重視する通信方式やアルゴリズムを採用している。これらの設計により、「京」の MPI ライブラリにおいては、使用メモリを抑制しながら、MPI 通信遅延 1.27us, MPI バンド幅 4.7GB/s を達成している。集団通信においても 9,216 ノードの MPI_Bcast で 10.6GB/s と高い通信性能を実現している。

The Design of MPI Communication Facility for K computer

Shinji Sumimoto,[†] Takahiro Kawashima,[†] Naoyuki Shida,[†]
Takayuki Okamoto,[†] Kenichi Miura,[†] Atsuya Uno,[††]
Motoyoshi Kurokawa,[††] Fumiyoshi Shouji[††]
and Mitsuo Yokokawa [††]

This paper describes the design of high performance MPI communication which enables high performance communication with minimized memory usage on the 82,944 node K computer. The Tofu interconnect of K computer uses six dimension torus/mesh direct topology for realizing higher performance and availability on hundreds thousand node system. However, in such a ultra scale system, communication performance degradation by increasing hops and network congestion and much memory consumption with increasing number of nodes are still problems to solve. To solve the problems, MPI communication facility of the K computer uses RDMA based communication and buffer allocation policy that limits buffer based communication to neighbor communication and the other uses less-memory usage communication method. As a result of these designs, MPI communication facility of the K computer realizes 1.27us MPI communication latency and 4.7GB/s MPI communication bandwidth with less memory usage, and a collective communication of MPI_Bcast achieves 10.6GB/s on 9,216 node K computer.

1. はじめに

2008 年 6 月の TOP500 リスト[1] において Road Runner[2] システムが 1.0PFLOPS の Linpack 性能を達成した。その後も、HPC システムの性能向上は止まることなく、2011 年 11 月の TOP500 リストでは、文部科学省が推進する「革新的ハイパフォーマンス・コンピューティング・インフラ (HPCI) の構築」プログラムで開発中の「京」[3] が 10.5PFLOPS を達成し第 1 位となった。これらの HPC システムの性能向上の鍵は、計算ノード内コア数の増加とノード数増加であり、今後もこの傾向は続いていくと考えられている。

「京」は、82,944 ノードを高い性能と耐故障性を実現するために 6 次元トーラス・メッシュ網である Tofu インタコネクト[4)~6)] で結合している。6 次元網の採用により、平均ホップ数減少、バイセクションバンド幅向上、そして、迂回パスによる耐故障性を確保している。しかし、それでも超大規模の直接網システムでは、通信ホップ数増加とネットワーク網でのメッセージ衝突による通信遅延の増加による通信性能の低下、ならびに、ノード数に比例して必要な使用メモリ量の増加が課題となる。

本論文では、「京」上で使用メモリ量を極小化しながら通信性能を高める MPI 通信の設計について述べる。「京」の MPI 通信機構はオープンソースである Open MPI[7] を採用しているが、Open MPI は send-receive

† 富士通, FUJITSU
†† 理化学研究所, RIKEN

型通信と通信網を意識しない集団通信アルゴリズムを採用しているため、通信ノード数と性能に比例した通信バッファメモリを必要とするのが課題となる。

この課題を回避するため、「京」の MPI 通信機構では、RDMA 型通信を主体とした通信を採用し、通信バッファメモリが必要な通信を隣接通信などよく利用される通信先に限定し、使用メモリ量を抑制している。また、集団通信においても RDMA 通信を主体としながら、メッセージ衝突を避ける通信アルゴリズムと複数のネットワークインターフェイスの活用で性能向上を実現している。

これらの設計に基づいて「京」向けの MPI 通信機構を実装評価した結果、使用メモリを抑制しながら、MPI 通信遅延 1.27us, バンド幅 4.7GB/s を達成した。集団通信においても 9,216 ノードの MPI_Bcast で 10.6GB/s と高い性能を実現した。

本論文の構成は、第 2 章に「京」とそのインタコネクトである Tofu の概要について述べ、第 3 章で「京」の課題と設計指針、第 4 章で設計、第 5 章で実装、第 6 章で評価、第 7 章で関連研究について述べる。

2. Tofu インタコネクトとユーザビュー

本章では、「京」が採用している Tofu インタコネクトならびにそのユーザビューについて述べる。

Tofu インタコネクト（以下 Tofu）は超大規模システムにおける通信性能と可用性向上のため 6 次元トーラス・メッシュ網を採用している。以下に Tofu の特徴について述べる。

- 6 次元トーラス・メッシュ(座標: X,Y,Z,A,B,C) で最大 32x32x32x2x3x2(393,216 ノード) の内、「京」では 24x18x17x2x3x2(88,128 ノード) を採用。計算ノードはこのうち 24x18x16x2x3x2(82,944 ノード)。メッシュは Y,A,C 軸である。また、A,B,C 軸が小さいのはケーブルを減らすためボード内、ならびにバックプレーンで結合しているからである。
- 耐故障性と通信性能向上を想定した拡張ルーティングを持ち、ソフトウェア制御で同じ通信先に対して最大 12 経路を選択可能。
- 4 つの Tofu ネットワークインターフェイス (TNI) とハードウェアバリア機構を搭載
- STAG(メモリ識別子) による物理メモリ管理と RDMA 主体の通信を提供。STAG にメモリ登録すれば、RDMA 通信が可能。
  - 通常の RDMA の他、通信要求を記述する Descriptor 上にデータを格納する Piggy Back RDMA を提供。
  - レジスタ上の Descriptor を直接送信要求可能な Direct Descriptor 機構
  - Piggy Back RDMA と Direct Descriptor 機構を組み合わせることにより、送信時の DMA メモリアクセスなしにメッセージ送信可能。

Tofu インタコネクトの持つ、6 次元ネットワークトポロジをユーザにどのように見せるべきかについては、次の理由により、6 次元の軸を組み合わせ、ユーザに仮想的な 3 次元以下のトーラス網を提供することとした。

(1) システムの分割利用時においても高い性能を提供するため、ノード間のホップ数が半分になるトーラス網を提供可能。
(2) 既存アプリケーションは 3 次元以下を想定し書かれたものが多い。

3. 「京」上での MPI 通信の課題と設計指針

本章では、「京」上での通信機構の課題と通信機構の設計指針について述べる。

3.1 既存の MPI 実装

既存の一般的な MPI 通信においては、次の 2 つの通信方式が利用される。

Eager 方式: 通信遅延を抑えるため、通信相手との同期なしにメッセージを送信する方式である。受信時にメモリコピーが発生するため、通信遅延に比べメモリコピー時間が小さい場合に有効である。通信性能は、一般に受信側のバッファメモリ量に比例するため、通信性能をあげるためにはより多くのメモリが必要になる。

Rendezvous 方式: 使用するメモリ量を抑制するため、制御通信を利用し受信側のバッファと送信側のバッファの準備が完了後にデータ転送を行う方式である。転送要求を受信側で制御できる他、RDMA 通信と組み合わせることにより、プロセッサによるメモリコピーが不要な通信を提供することが可能である (Zero-Copy 通信)。

MPICH[8] や Open MPI では、これらの通信方式について、通信性能を高めるために通信のメッセージ長が比較的に短い場合には Eager 方式を採用し、それ以上のメッセージ長の場合には Rendezvous 方式による実装となっている。また、方式選択のポリシーは、相手先に関係なく同じポリシーが採用されている。

3.2 「京」向け MPI 通信の課題と目標

第 3.1 節で述べたように、現状の MPI 実装においては、通信遅延を考慮して Eager 方式と Rendezvous 方式を使い分けている。しかし、10 万ノードクラスの直接網を搭載した「京」では、次の理由により、性能とメモリ使用量で大きな課題がある。

(1) 最大で 66 ホップ (平均 33 ホップ) と相手先により通信遅延が大きく異なる。
(2) メッセージの衝突の有無とその程度で大きく性能が異なる。
(3) 「京」の MPI ベースである Open MPI をそのま

ま適用すると、10 万ノードクラスで 25.6GB☆が必要となり、「京」のメモリが 16GB ということから、省メモリ化が必須である。

以上のことより、「京」の MPI 通信においては、「システム関係のメモリ利用量を実用的な範囲に抑えながら高い通信性能を実現すること」を目標とする。

3.3 「京」向け MPI 通信の設計指針

本節では、第 3.1,3.2 節の議論より、次のような「京」向け MPI 通信の設計指針を設定する。

通信遅延を意識した通信方式： 第 3.1,3.2 節の議論より、直接網である Tofu では、相手先により通信遅延が大きく異なるため、相手先との通信遅延を意識した通信方式の選択が必須である。

メッセージ衝突を意識した通信アルゴリズム： 特に集団通信は採用するアルゴリズムによりメッセージ衝突の程度が大きく異なるため、メッセージ衝突の程度により最適なアルゴリズムを実装する。

Eager 方式利用の相手先選定方式： 相手先により通信遅延が大きく異なるため、一定の遅延を越えた宛先については、メモリ消費を抑えるため Eager 方式を採用しない。

省メモリ化の実現方式： ノード数が増えるにつれ MPI 通信機構のメモリ使用量が増加する。Eager 方式の通信では高い性能実現には多くのメモリが必要である。さらに、アプリケーションにより MPI 通信機構が使えるメモリ使用量は異なる。このことから、計算ノード毎に優先的に Eager 方式が利用可能な相手先数を制限することにより一定以下の使用メモリ量に抑えられるようにする。

4. 「京」上での MPI 通信機構の設計

「京」では、最新の MPI 機能の提供を早期に行うため SPARC で稼働実績のあるオープンソースの Open MPI を採用している。本章では、これまでの議論を元にした「京」上での MPI 通信機構の設計について述べる。

4.1 Open MPI の通信方式と実装

Open MPI では、送受信メッセージ長や配置によって複数の通信方式を提供している。例えば InfiniBand 向けの実装では次のような通信方式を提供している。

Eager RDMA: 短メッセージ向けの低遅延通信のために、RDMA を用いて Eager 方式でメッセージ転送を実現する。受信側では、ポーリングにより受信データを待ち合わせる。ポーリングは相手先毎に必要であるため、相手先数に比例したオーバヘッドが必要となる。

Eager Send/Receive: 一般の Eager 方式でのデータ転送である。InfiniBand は、Send/Receive 型の通信をハードウェアでサポートしているため、容易に実装可能である。しかし、一般的に最大メッセージサイズ (MTU) のバッファを複数準備する必要があるため、相手先が増えるとメモリ使用量が比例して大きくなる。InfiniBand では、共有 Receive Queue により相手先数に比例するまでは必要としないが、仕様上この Queue がオーバフローしないだけ十分に多く割り当てる必要がある。

RDMA Direct: 一般の Rendezvous 方式を利用した RDMA 通信であり、連続に置かれたデータ転送に用いる。

Send/Recieve Pipeline: 不連続データや集団通信のパイプライン転送に用いられる。

Tofu を用いた通信でも InfiniBand と同様の RDMA 通信を用いた通信方式を用いることが可能である。

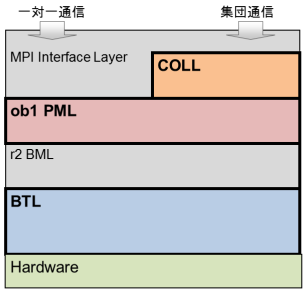

図 1 Open MPI のアーキテクチャ

次に Open MPI の実装構造を図 1 に示す。Open MPI は集団通信を担当する COLL 層と一対一通信を担当する PML,BML,BTL 層から構成される。PML 層でメッセージ通信向け API を提供している。モジュラ形式で役割分担が明確な半面、RDMA 通信の特性を活かした通信ができない、また、3 階層構造のソフトウェアオーバヘッドが問題となる。

4.2 「京」向け MPI 通信の設計

第 3.3 節で述べた設計方針に基づく「京」向け MPI 通信の設計について述べる。「京」向け MPI 通信においては、以下の 3 つの要件がある。

- 高い通信性能と省メモリのための機構
- 故障ノード回避と仮想 3 次元トーラス網提供
- 使用メモリ量設定、ネットワーク網指定などジョブシステムとの連携

これらの要件を満たすため、「京」向け MPI においては、MPI(高レベル) と Tofu 向けの低レベル通信機構との役割分担を次のように設定する。

- 低レベル通信機構では、3 次元トーラス網提供機能、低レベルの高性能通信を提供する。
- 高レベルで (MPI) では、ジョブシステム連携機

☆ Eager 方式用にそれぞれ 128KBx2 送受信用に割り当てた場合

能、通信方式の選択と資源割当て機能を提供する。次節以降それぞれの通信機構の設計について述べる。

4.3 低レベル通信機構の設計

低レベル通信機構では、3 次元トーラス網提供機能と高性能幅通信を提供する。

3 次元トーラス網提供: ユーザ指定の形状 (1,2,3 次元指定と大きさ) に合わせ 6 次元トーラス・メッシュ網にランクをマッピングする。同時にジョブシステムからの故障ノード情報を元にすべてのノード間で通信可能かを確認し、通信できない場合には第 2 章で述べた拡張ルーティングにより通信可能なパスを通信先毎に設定する。

高性能通信: 高性能通信を実現するために、可能な限り簡易でオーバヘッドの少ない実装とする。このため、ハードウェアの仮想化は行わずハードウェア制御関数として実装する。実現機能は RDMA 通信 (Read,Write) と集団通信 (Barrier, Reduce, Bcast, Allreduce) である。これを TNI の持つコマンドキューにより制御する。

4.4 MPI レベル通信機構の設計

本節では、MPI レベルの通信機構の設計について、まず、設計指針への対処法を延べた後、Tofu への Open MPI への実装として通信方式の選定、低遅延通信の実現手法について議論し、これを元にした MPI 実装アーキテクチャについて述べる。

設計指針に対しての対処法

第 3.3 節の設計指針に基づく設計について述べる。

通信遅延を意識した通信方式: Tofu では、1 ホップあたり 100ns 程度の通信遅延が増加するため、メッセージ長と同様に、ある一定以上の遅延増加で通信方式を変更することとする。

メッセージ衝突を意識した通信アルゴリズム: メッセージ衝突の確率はメッセージ長で大きく異なるためメッセージ長が長いところではメッセージ衝突のない経路を意識したアルゴリズムをパイプライン転送方式で実装し、短いところでは遅延が最小になるようなアルゴリズムを選定する。また、パイプライン転送方式の基本になる隣接通信については、遅延の極小化を徹底する。

Eager 方式利用の相手先選定方式: 一定ホップ数以下の相手先の中で一定回数を越えた中の相手先を Eager 方式利用とする。Eager 方式の選定は先着順登録を基本とする。それ以外の相手先については、メモリ量を抑えるため Rendezvous 方式を採用する。Eager RDMA 方式を採用する通信方式を高速型通信モード、採用しない通信方式を省メモリ通信モードと定義し、相手先毎に制御する。

省メモリ化の実現方式: メモリをより多く使用する高速型通信とメモリ使用量を極限に抑えた省メモリ型通信を定義し、各計算ノードで一定条件で優先的に高い性能が必要な通信先に高速型通信を一定数割り当てる。

通信方式の選定

第 4.1 節で述べたように InfiniBand は、RDMA 型通信と共有 Receive Queue により受信バッファを共有したメッセージ通信をハードウェアでサポートしているため、Eager RDMA, Eager Send/Receive, Direct RDMA 方式の 3 つの通信方式を自然に実装可能である。しかし、Tofu は RDMA 通信のみサポートしているため、Eager Send/Receive についてはソフトウェア実装となる。このため、最小限のメモリ量で実現するため固定量のメモリを割り当て、送信側で受信先のバッファメモリの更新先を制御することとする。ここで、Send/Receive Pipeline 方式をどう扱うかが課題となるが、Pipeline 転送は集団通信実装上必須であるため、別途 RDMA を用いた Pipeline 方式を実装する。

低遅延通信の実現手法

Tofu では、ハードウェアレベルで 0.91us[5] のノード間通信を実現しているため、MPI 通信においてもソフトウェア遅延を最小限にしたい。しかし、Open MPI の実装では、PML, BML, BTL と 3 つのソフトウェア階層から構成されているため、ソフトウェアオーバヘッドが無視できない。このため、一定条件を満たす通信については PML, BML, BTL の階層をバイパスすることで低遅延を実現する。

「京」向けの Open MPI アーキテクチャ

以上の議論より決定した「京」向けの Open MPI アーキテクチャを図 2 に示す。

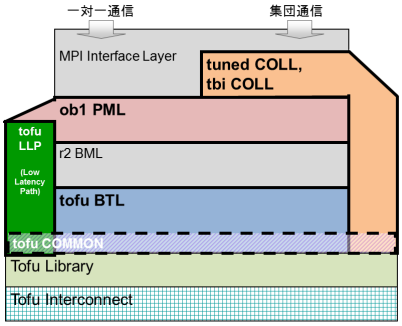

図 2 「京」向け MPI のアーキテクチャ

従来の Open MPI に対して低遅延通信部分 (Tofu LLP) と集団通信部分 (Tuned coll, TBI coll) を追加した構成としている。これは、低遅延通信については、PML, BML, BTL のオーバヘッドを排除するため、集団通信部分については、RDMA を主体とする複数の TNI を用いた Pipeline 転送とハードウェアによる集団通信を実装するためである。なお、Open MPI の構

造自体を根本から見直す手法もあるが、機能追加の頻繁な MPI 仕様を考慮すると機能追加対応のための実装コストが大きくなる。このため最新の仕様に早く対応できることを優先した。

5. 実　　装

低レベル通信機構の実装概要

低レベル通信機構は、第 4.3 節で述べたように、3 次元トーラス網提供機能と高性能幅通信を提供する。特に Tofu の特徴を活かすために、STAG による物理メモリ管理と RDMA 主体の通信を提供する他、次の通信を低ソフトウェアオーバヘッドで提供している。

- 通常の RDMA の他、Descriptor 上にデータを格納する Piggy Back RDMA による通信を提供
- 連続する 64bit レジスタを 8 本用いたレジスタ上の Descriptor を直接送信要求可能な Direct Descriptor 機構を提供

MPI レベル通信機構の実装概要

MPI レベル通信機構の実装では、特に低遅延化徹底のため連続データの場合は Tofu LLP を使用して通信を行う。これは、Open MPI の一対一通信を担当する PML 層の戦闘で振り分ける. また、一対一通信は次のようにメッセージサイズ毎に低遅延化を徹底している。

Eager RDMA の実装: Eager RDMA は 0-109 バイトまでの MPI メッセージ長で採用する。RDMA の転送単位 (128 バイト) でデータの順序性が保存されないためヘッダを除いて 109 バイトまでとしている。この中でも 0-16 バイトまでは、Piggy Back RDMA と Direct Descriptor 機構により、送信 DMA 時のメモリアクセスを削減している。17-109 バイトまでのデータについては、キャッシュライン（128 バイト）の部分アクセスを避けるために、メッセージ長に関係なく 128 バイトを転送することで無駄なメモリアクセスを減らしている。また、Eager RDMA の受信確認は、データ領域のポーリングにより実装している。高速型通信モード (第 4.4 節) は、この Eager RDMA を採用する。使用メモリは 32KB(128B x 256) である。

Eager Send の実装: 宛先毎に一定量の受信バッファを確保し、送信先より直接 RDMA でオフセットアドレスを変更して書き込むことにより実現している。受信確認は Tofu の持つ受信通知キューを用いて行う。高速型通信モードにおいては、ホップ数とメッセージ長による RDMA Direct の通信遅延差をなくすように最大メッセージ長を調節するが、。省メモリ通信モードでは一定値とする。割り当てるバッファは高速型通信では 1MB、省メモリ型では 2KB である。これらは変更可能である。

Direct RDMA の実装: Direct RDMA の実装では Rendezvous 方式により実装する。この実装には Open MPI の実装を利用するが、Tofu の持つ RDMA 先への通知機構により、RDMA 実行後の相手ノードへの通知メッセージを省略している.

集団通信の実装

集団通信の実装については tuned COLL, tbi COLL を使用するが、データが連続している場合に低レベル通信機構を用いた関数を使用し、直接 RDMA 通信と複数の TNI を用いて実装している. それ以外は既存の Open MPI の PML 層を使用する。

集団通信の実装例として、MPI_Bcast と MPI_Allreduce がある[9]。「京」上の MPI_Bcast と MPI_Allreduce の実装では、経路上での衝突が発生しない Trinaryx3 という通信アルゴリズムを用いている。これを、RDMA を用いてパイプライン方式で転送する。この際にパイプライン方式での転送に利用する中間バッファを集団通信実行の最初に 1 度だけ固定した後、連続的に RDMA を用いて転送し Rendezvous 通信を無くしている.

省メモリ化機構の実装

省メモリ化機構は以下のように動作する。

(1)　MPI_Init 時、バッファ割当て無し。
(2)　通信発生時に省メモリ型通信に移行。
(3)　一定回数以上の通信で高速型通信に移行。
(4)　一定数の高速型通信割当て後、割当停止。
(5)　高速型通信割当て数はユーザが指定可能。

図 3 に高速型通信と省メモリ型通信のプロトコル選択イメージを示す. このように「京」向けの MPI では用途別に細かく通信方式を設定して実現している.

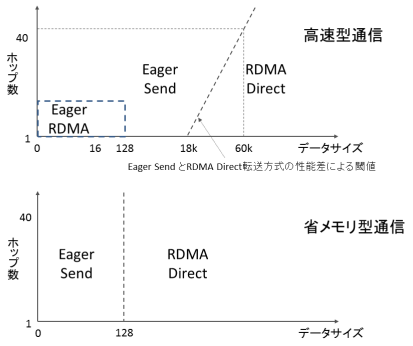

図 3　高速型通信と省メモリ型通信のプロトコル選択イメージ

図 3 上の図において、高速型通信においては、ホップ数により通信プロトコルを変更している. Eager RDMA については、相手先を絞りメモリ消費量をおさえるため 1 ホップで通信可能なノードについて選択し、2 ホップ以上は Eager Send を選択する。また、Eager Send と RDMA Direct の選択は実際の性能測の結果を用い、1 ホップ時 18KB、40 ホップ時に 60KB のラインに沿うよう通信プロトコルを選択する。

6. 評　　価

本章では、「京」向けに設計した通信機構について、MPI レベルならびに低レベル通信での通信性能を評価する。評価システムは「京」を用い、集団通信の評価には、9,216 ノード (3 次元トーラス 48x6x32) を用いた。評価環境を表 1 に示す。

表 1　評価環境

計算ノード	SPARC64 VIIIfx(8 Core,2GHz) + ICC
メモリ量	16GB
インタコネクト	Tofu 6D Torus/mesh(5GB/s(x2)x10link)
OS	Linux SPARC

なお、MPI レベルの評価には IMB(Intel MPI Benchmarks) を、低レベル通信機構の評価には、測定用のプログラムを開発し利用した。なお、プロセッサは 8Core であるが、すべての場合について 1 ノード 1 プロセス (1Core) での評価結果である。

6.1　通信遅延の評価

図 4、5 に低レベル通信機構と MPI 通信機構の片道通信遅延の評価結果を示す。低レベル通信は、Data Polling,Hard Queue Polling を用いた結果を示す。MPI 通信は制御ヘッダ (16 バイト) が実転送データに加えられる。

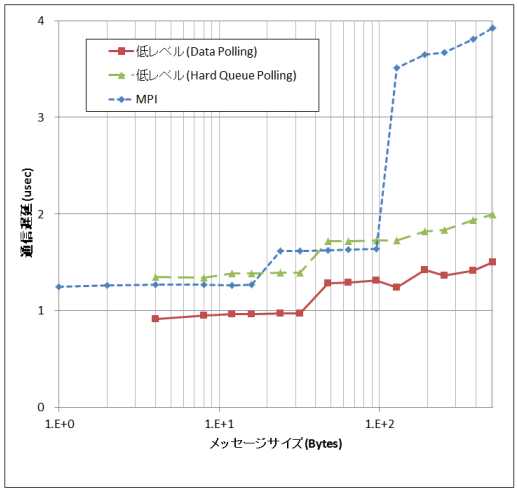

図 4　通信遅延 (ショートメッセージ)

図 4 において、低レベル通信の遅延は,4 バイトメッセージで 0.91us であった。また、32 バイトまでは Piggy Back RDMA による転送、64 バイト以上は通常の RDMA による転送である。一方、MPI 通信機構の結果では、MPI ヘッダが 16 バイト必要であるため、16 バイトまで Piggy Back RDMA による効果により、通信遅延は 1.27us であった。96 バイトまでは、RDMA による Data Polling を利用する 128 バイトキャッシュラインを一括して上書きするため遅延が小さくなっている。128 バイト以上は、TNI の通信完了キューによりデータ到着を確認する他、Eager Send の受信バッファ制御のため、遅延が大きくなっている。今回 Eager RDMA 採用を 109 バイトまでとしたが、これはハードウェア仕様が RDMA の転送単位が 128 バイト単位で、かつ、メモリに書かれる順序性保証がないため、実装簡易化のためである。しかし、性能差が大きいため、複数の RDMA 転送単位に対応した実装が考えられる．だが、Eager RDMA の転送サイズを増やすと消費メモリ量も比例して増加するため、対応する相手先を絞るなどの工夫が必要である．

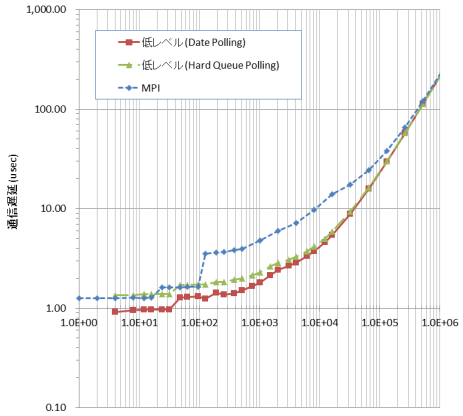

図 5　通信遅延

図 5 に長メッセージの通信遅延を示す。Eager Send と Direct RDMA の切り替えは 18KB 付近に設定した結果、滑らかにプロトコル切り替えができている。

6.2　通信バンド幅の評価

図 6 に低レベル通信機構と MPI 通信機構の通信バンド幅の評価結果を示す。MPI 通信機構では，処理オーバヘッドがあるため、立ち上がりに違いが見られるが、低レベル通信機構、MPI 通信機構ともに最大 4.7GB/s の通信バンド幅性能を実現している．

図 7 に低レベル通信機構で複数 TNI 利用時の通信バンド幅の評価結果を示す。評価は 2 ノード間で拡張ルーティングで経路を変更することで実現している。

図 7 の結果より、Tofu では複数 TNI 利用により、3 つまでは TNI 数分の転送性能が得られている。4 つ利用時に性能向上が小さいのは、CPU と ICC 間に 16GB/s で律速される部分があるからである。

6.3　MPI レベル集団通信の評価

本節では，MPI レベルの集団通信性能評価としてハードウェアとソフトウェアによる集団通信を評価する．

図 8 に、9,216 ノードでの MPI_Bcast の結果を示す。比較対照として Open MPI の Chain アルゴリズムの

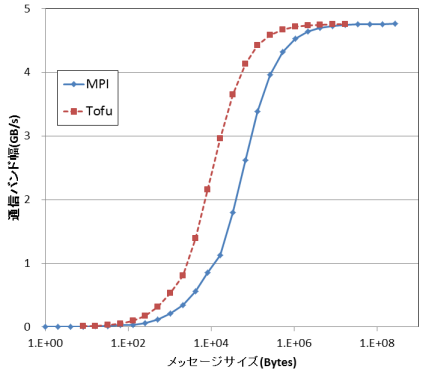

図 6　通信バンド幅

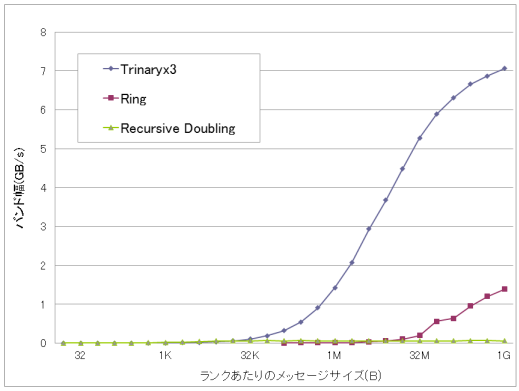

図 9　MPI_Allreduce の通信バンド幅

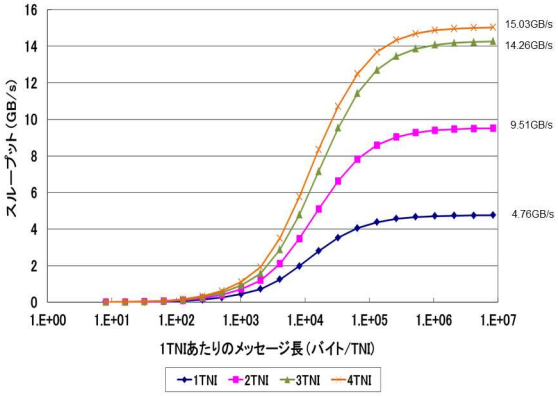

図 7　低レベル通信の TNI を増加させた場合のバンド幅

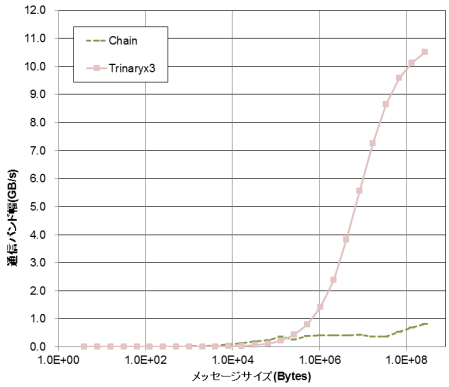

図 8　MPI_Bcast の通信バンド幅

結果も示す。Trinaryx3 アルゴリズムを用いた「京」向けの実装は 10.6sGB/s の転送性能に比べ，Open MPI の Chain アルゴリズムによる実装は 0.8GB/s と既存アルゴリズムに比べ 13.25 倍の性能を実現している．

図 9 に、9,216 ノードでの MPI_Allreduce の評価結果を示す[9]。比較対照として Open MPI の Ring と Recursive Doublling アルゴリズムの結果も示す.Trinaryx3 アルゴリズムを用いた「京」向けの実装は 7.1GB/s の転送性能に比べ，Open MPI の Ring アルゴリズムによる実装は 1.4GB/s と既存アルゴリズムに比べ 5.0 倍の性能を実現している．

6.4　省メモリ性の評価

本節では、「京」向けの MPI の省メモリ性を評価する。省メモリ性評価のため、高速型通信のみ、省メモリ通信のみ、一定数 (1,024) 高速型通信を利用した場合の 3 種につき、すべてのノードと通信する Simple Spread による AlltoAll プログラムを作成しノード数を変化させ MPI プログラムの使用メモリ量を評価した。

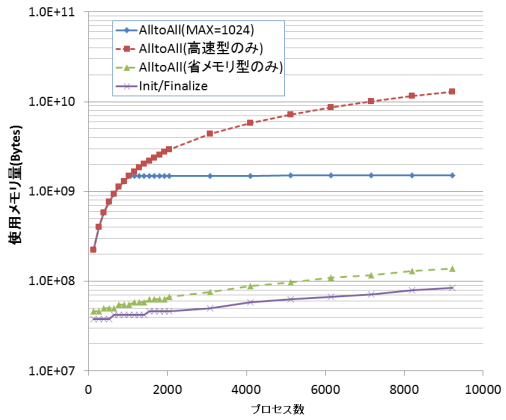

図 10　省メモリ性評価

図 10 に 9,216 ノードまでの評価結果を示す。参考までに MPI_Init/Finalize のみで実行した結果も示す。9,216 ノードの結果は、1.52GB(高速型 MAX=1024), 12.9GB(高速型のみ)、0.14GB(省メモリのみ)、0.08GB(MPI_Init/Finalize) であった。

省メモリ型では MPI_Init/Finalize に比べ 0.06GB の増加で Simple Spread 型の AlltoAll を実行可能である。性能には影響があるが、少ないメモリで実行できる

ことは意味がある。また、高速型のみの場合は 12.9GB と 16GB の物理メモリの 80%を占めてしまうが、使用数を 1024 に絞った場合は 1.52GB と全体の 10%まで使用メモリ量を抑えている。

7. 関 連 研 究

10,000 ノードを越える直接網システムとしては、CRAY XT-5 と IBM Blue Gene がある。

CRAY XT-5 システム： CRAY XT-5 システムの MPI 通信ライブラリは MPICH2 ベースの CRAY MPI の他、Open MPI が利用可能となっている。CRAY XT-6 ネットワークインターフェイス (NIC) が一つである他、NIC の転送バンド幅がネットワークの持つバンド幅に比べ、相対的に小さいため集団通信は通常の集団通信でもメッセージ衝突頻度が小さい。また、XT-5 はネットワークインターフェイスが一つであるため MPI 通信機構は複数のネットワークインターフェイスに対応していない。

IBM Blue Gene/L,P,Q システム： IBM Blue Gene/L,P,Q システムの MPI 通信ライブラリは MPICH である[10]。Tofu との 違いはリンク 1 本あたりのバンド幅が異なるため、トータルバイセクションバンド幅では Tofu が上回る。複数のネットワークインターフェイスを搭載しているため, 集団通信の実装は類似性があるが、ネットワークバンド幅に占める総ネットワークインターフェイスバンド幅の割合が Tofu に比べ大きいため、メッセージ衝突の発生する通信の場合、性能劣化が問題となる。Blue Gene Q については、リンクの本数が 10 本と多くトータルバンド幅を生かすためには 10 本のリンクを活用する必要がある.

これらのシステムで利用されている MPICH と Open MPI については、メッセージサイズ毎による通信プロトコルの切り替えは採用されているが、ホップ数と省メモリ性を考慮したプロトコルの切り替えは採用されていない点が異なる。

8. ま と め

本論文では 82,944 ノードの「京」上で使用メモリ量を極小化しながら MPI 通信性能を高める通信機構の設計について述べた。「京」が採用した Tofu インタコネクトは数十万ノードクラスのシステムで高い性能と耐故障性を実現するため直接網である 6 次元トーラス・メッシュ網を採用している。しかし、超大規模の直接網システムでは、通信ホップ数増加とネットワーク網でのメッセージ衝突による通信遅延の増加による通信性能の低下、ならびに、ノード数に比例して必要な使用メモリ量の増加が課題となる。この課題を解決するため、RDMA 通信を主体とし、通信バッファが必要な通信は隣接通信等の最小限に絞る、遅延の大きな場合は省メモリ性を重視する通信方式やアルゴリズムを採用している。これらの設計により、「京」の MPI ライブラリにおいては、使用メモリを抑制しながら、MPI 通信遅延 1.27us, バンド幅 4.7GB/s を達成した。集団通信においても 9216 ノードの MPI_Bcast で 10.6GB/s と高い通信性能を実現した。

今後の予定としては、アプリケーションを用いた性能評価を行い、更なる高性能化と安定性を目指す予定である。なお、今回は Open MPI のアーキテクチャを変更せず外側に Tofu 専用の 1 対 1 通信と集団通信を実現したが、より一般化した RDMA 主体のモジュールを開発し、コミュニティにフィードバックすることも検討したい。

参 考 文 献

1) Super Computer TOP500: http://www.top500.org/.
2) Los Alamos Lab Roadrunner: http://www.lanl.gov/roadrunner/.
3) スーパーコンピュータ「京」: http://www.kcomputer.jp/k/.
4) Yuichiro Ajima, Shinji Sumimoto, and Toshiyuki Shimizu. Tofu: A 6d mesh/torus interconnect for exascale computers. In *IEEE Computer*, pp. 36–40, Nov. 2009.
5) Yuichiro Ajima, Yuzo Takagi, Tomohiro Inoue, Shinya Hiramoto, and Toshiyuki Shimizu. The tofu interconnect. In *Hot Interconnects*, pp. 87–94, 2011.
6) 追永勇次. 次世代スパコン「京 (けい)」のコアテクノロジ. 応用物理 第 80 巻 第 7 号, pp. 590–593, 2011.
7) OpenMPI: http://www.open-mpi.org/.
8) MPICH2: http://www.mcs.anl.gov/research/projects/mpich2/.
9) 松本幸, 安達知也, 田中稔, 住元真司, 曽我武史, 南里豪志, 宇野篤也, 黒川原佳, 庄司文由, 横川三津夫. MPI_Allreduce の「京」上での実装と評価. 情報処理学会研究報告 2011-ARC-197. 情報処理学会, Nov 2011.
10) George Almási, Philip Heidelberger, Charles J. Archer, Xavier Martorell, C. Chris Erway, José E. Moreira, B. Steinmacher-Burow, and Yili Zheng. Optimization of mpi collective communication on bluegene/l systems. In *Proceedings of the 19th annual international conference on Supercomputing*, ICS '05, pp. 253–262, New York, NY, USA, 2005. ACM.