情報処理学会論文誌　コンピューティングシステム　Vol. 3　No. 2　88–98 (June 2010)

2次元Meshネットワーク・Torusネットワーク上での最適全対全通信アルゴリズム

高上治之[†1]　矢崎俊志[†2]　安島雄一郎[†3]
清水俊幸[†3]　石畑宏明[†1]

本論文では，各ノードは同時に複数のメッセージを送受信可能なモデルを前提とした，2次元Meshネットワーク・Torusネットワーク上での全対全通信アルゴリズムを提案する．提案するアルゴリズムでは，各ノードは，複数のメッセージを同時に送信する方式をとっており，Meshネットワークでは，同時に2つのメッセージを送信することにより，Torusネットワークでは，同時に4つのメッセージを送信することによりネットワークのバイセクションバンド幅を最大限に引き出すようにスケジューリングしている．本方式での通信時間は，2次元Meshネットワーク・Torusネットワーク上での理論的下限を達成していることを示す．

Optimal All-to-All Communication Algorithm on 2-dimensional Mesh Network and Torus Network

Haruyuki Takaue,[†1] Syunji Yazaki,[†2]
Yuichiro Ajima,[†3] Toshiyuki Shimizu[†3]
and Hiroaki Ishihata[†1]

In this paper, we present an optimal all-to-all communication algorithm for a 2D mesh/torus network. The proposed algorithm ensures full utilization of the network link bisection bandwidth without the need for split-phase operation, which are used in previously proposed algorithm, provided each node can transfer several messages concurrently. We show the proposed alogrithms acheives the theoretical lower bound time of all-to-all communicaiton in both a 2D mesh with two concurrent message transfers and a 2D torus with four concurrent message transfers.

1. はじめに

ネットワークへの通信負荷の高い通信パターンとして全対全通信がある．全対全通信とは，すべてのノードが他のすべてのノードに対して，それぞれ異なった内容のメッセージを送信する通信パターンである．この通信パターンは，行列の転置，FFTなど多くのアプリケーションで頻繁に使用される．近年の大規模並列計算機では，ノード数の増加にともない，Mesh・Torusなどのネットワークトポロジが用いられることが多くなってきた．このようなネットワークを持つシステムでは，ネットワークのバンド幅を最大限に引き出す通信アルゴリズムの実現が重要となる．

特定のトポロジのネットワークで，最適な全対全通信アルゴリズムが提案されている．Scott[1]は，Hypercubeおよび Mesh上での最適な全対全通信アルゴリズムを提案している．堀江ら[2]，Yu-Cheeら[3]はそれをn次元Torusに拡張したものを提案している．これらのアルゴリズムは，ノード間の1対1通信をグループ化し，グループごとに通信するフェーズと休むフェーズとを組み合わせることにより，通信ネットワークのバイセクションバンド幅を100%使用するように通信の順序をスケジューリングし，理論的な下限の通信時間で全対全通信を行えることが示されている．

これらの手法の問題点として，個々のノードがメッセージの送信タイミングを同期しなければならないという点があげられる．このアルゴリズムを実際に利用しようとすると，全ノード間でフェーズを合わせることが必須となり，これが大きなオーバヘッドとなる．

BlueGene/Lでは，専用のアルゴリズムが用いられている．全対全通信において，Adaptive Routingと宛先のランダム化を組み合わせることにより，100%に近い通信性能を実現している[4]．このアルゴリズムを使用するためには，Adaptive Routingの実装が必要となる．また，各次元のサイズの異なるTorusでの性能を改善する方式も提案されている[5]．

Mesh・Torusのように，ノードから複数のリンクが出ている場合には，複数の通信コントローラを並行動作させ，リンクを同時に使用することにより通信性能を向上させることができる．最近の並列コンピュータは，1つのノードに複数の通信コントローラを持ち，複数

†1 東京工科大学
Tokyo University of Technology
†2 電気通信大学情報基盤センター
Information Technology Center, The University of Electro-Communications
†3 富士通株式会社
FUJITSU, LIMITED.

のメッセージを同時に送信できるものが増えている．Bruck ら[6]，Tipparaju ら[7]，Ajima ら[8] は，そのようなシステムについて報告している．

本論文では，2 次元 Mesh ネットワーク，2 次元 Torus ネットワークと複数のメッセージを同時送信可能な通信コントローラを持つシステム向けに，全対全通信を最適に実行するアルゴリズム A2AT を提案する．2 次元 Mesh ネットワークでは，同時に 2 つの，2 次元 Torus ネットワークでは，同時に 4 つの通信コントローラを並行動作させる．つねにすべてのリンクを使い，送信を行うようスケジューリングするものである．本方式は，従来の方法のように各ノードが通信フェーズごとに同期をとる必要はない．個々のノードは複数のメッセージを独立に並行して送信するだけで，理論的な下限の通信時間で全対全通信を行うことができる．

本論文では，2 章で対象とするネットワークと全対全通信時間について述べ，3 章で提案するアルゴリズムについて述べる．4 章では Torus ネットワークへの拡張について述べ，5 章で提案するアルゴリズムと既存のアルゴリズムとの全対全通信時間を比較し，6 章でまとめる．

2. 全対全通信

2.1 対象とするネットワーク

本論文では，2 次元格子状に接続された各ルータにノードを 1 つ接続する構成の 2 次元 Mesh・Torus ネットワークを扱う．ルータ間は双方向のリンクで接続されている．Mesh・Torus を構成する全リンクは同一のバンド幅を持ち，各リンクは同時に双方向の通信を行うことが可能なものとする．

メッセージの中継は，バーチャルカットスルーまたは，ワームホールを想定している．通信時間はメッセージ長が中継段数に対して十分に長い場合，通信距離の影響を無視でき，メッセージサイズに比例する．ルーティングは Dimension-order ルーティングとする．この方式は，XY 平面上にある 2 次元 Mesh・Torus ネットワークの場合は，まずメッセージを送信元ノードから X 軸方向に送り，次に Y 軸方向に送って，宛先ノードに到達させる方式である．

ネットワーク上でのコンテンションの影響を考慮するために，ネットワーク中の全リンクについて，メッセージはフェアにリンクのバンド幅を共有して流れるものとする．最近のハードウェアでは，ホップ数に応じて，アービトレーションの優先度を変えてグローバルフェアネスとなるような実装がある[9]．

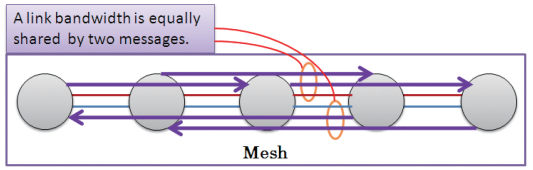

図 1　バンド幅をメッセージ数でフェアに共有
Fig. 1　Bandwidth is equally shared by the number of messages.

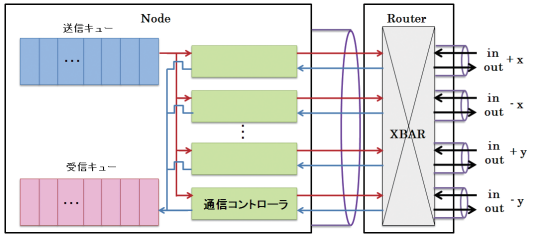

図 2　Node 内，Router 内の構成
Fig. 2　Composition of node and router.

図 1 のように，各ノードが 2 つ先のノードに対して，メッセージを送信した場合，各リンクに流れるメッセージ数は 2 つとなる．各メッセージは，パケットに分割されリンクを共有して流れる．バンド幅はフェアに等分されるため，各ノードはバンド幅を 1/2 ずつ享受して送信を行う．すべてのリンクを使い，通信をする際に，ネックとなるバイセクションバンド幅を最大限に引き出し送信を行う．各ノードは，**図 2** のように，送信メッセージが格納された 1 つの送信キューに対し，複数の通信コントローラを持つ．ノードとルータ間は双方向の通信路で接続され，ルータ間リンクのバンド幅に対して十分に大きいバンド幅を持つものとする．各ノードは，全対全通信に必要な $(N-1)$ 回の送信を FIFO 順に並行動作する通信コントローラに割り当てていく．

2.2 既存の全対全通信アルゴリズム

途中でメッセージの蓄積・結合を行わない，直接法による全対全通信アルゴリズムとして，MPICH[10] に使用されている SimpleSpread 法（以降，A2A）がある．直接法はバンド幅がボトルネックとなるような，メッセージサイズが大きい場合に用いられる．このため，バンド幅を最大限に引き出す通信アルゴリズムの実現がより重要となる．

A2A は，以下のように宛先を 1 個ずつ順に変えて送信する．

```
for i = 1    to    N − 1 do
  send((myid + i) mod N);
end for
```

ここで，N は全ノード数，$myid$ は，自分の Rank，$send(t)$ は，ノード t へのメッセージ送信を示す．

通信ネットワークが Mesh や Torus である場合は，1 次元で順に送受信位置を移動させていく A2A のほかに，d 次元座標上の相対位置を順次移動させる方法もありうる．この方法を以降，A2AND と呼ぶ．x 方向のサイズが N_x，y 方向のサイズが N_y の 2 次元 Mesh・Torus では，送信先のノードの相対位置を 2 次元座標 (x, y) で示す．各ノードは $0 \leq x \leq N_x - 1$，$0 \leq y \leq N_y - 1$ の範囲に以下のように規則的に送る．

```
for x = 0 to N_x − 1, y = 0 to N_y − 1 do
  if x ≠ 0    and    y ≠ 0 then
    send((myidx + x) mod N_x,
         (myidy + y) mod N_y);
  end if
end for
```

2.3　Mesh・Torus での全対全通信時間の下限値

$N_x \times N_y$（$N_x \geq N_y$）ノードの Mesh ネットワークにおける全対全通信時間の下限値 T_{lb} は，バイセクションバンド幅が N_y であり，バンド幅で正規化すると，

$$T_{lb} = \lfloor \frac{N_x}{2} \rfloor \lceil \frac{N_x}{2} \rceil N_y \tag{1}$$

であることが知られている．Torus ネットワークでは，分割線上を通る通信の数は Mesh の半分となるため，下限値は

$$T_{lbt} = \frac{1}{2} \lfloor \frac{N_x}{2} \rfloor \lceil \frac{N_x}{2} \rceil N_y \tag{2}$$

となる．

3. 提案するアルゴリズム

提案する A2AT では，ノードに接続されたリンクを有効活用させることを考え，同一方向への複数の通信を重ねないよう，つねに全方向のリンクが使用されるように通信をスケジューリングする．MPICH で使用されている既存のアルゴリズムでは，どれも同一方向への送信が連続し，その方向のリンクのバンド幅がネックとなって全体のバンド幅を有効に利用できない．通信コントローラを複数用いるメリットを活かすため，全対全通信に必要な $(N-1)$ 回の送信を FIFO 順に並行動作する通信コントローラに割り当てていく．並行動作する通信コントローラの数を，以降，NCT とする．

1 辺のサイズ N が奇数である 2 次元 Mesh において，各ノードは自分から見て，x 方向に i，y 方向に j の位置にあるノードを相対位置 (i, j) で表す．各ノードは，自分の位置をネットワークの中心として送信先を考えているため，送信範囲は，$-(N-1)/2 \leq i$，$j \leq +(N-1)/2$ となる．送信元となる自ノードの位置と宛先ノードの位置を絶対位置で (x, y) で表す（$0 \leq x$，$y \leq N-1$）．$+x$ 方向の $(i, 0)$ に対し送信を行う場合を考えた場合，送信範囲が $-(N-1)/2 \leq i$，$j \leq +(N-1)/2$ であるため，宛先ノードの位置が N を超えるノードが存在する．このような場合，宛先ノードは $((x+i) \bmod N, y)$ の位置となる．送信を行う際の経路は，Mesh ネットワークでは，図 1 のように，$-x$ 方向のリンクを使用して送信を行う．Torus ネットワークでは，Mesh の端と端を接続するラップアラウンドパスが存在するため，各ノードは $+x$ 方向のリンクを使用して送信を行う．

Mesh ネットワークでは，図 1 のように，各ノードが $+x$ 方向の 2 つ先のノードに対し，メッセージを送信する場合，$-x$ 方向のリンクも同時に使用する．$+x$ 方向，$-x$ 方向の各メッセージの経路上に流れる最大のメッセージ数は 2 となるため，各ノードが享受できるバンド幅は 1/2 となる．y 方向への通信を行う場合についても同様である．NCT を 2 とした場合，x 方向へのメッセージ送信と y 方向へのメッセージ送信を組み合わせれば，すべてのリンクを使用できる．

このように，すべてのノードが自分の位置から (i, j) の位置にあるノードへメッセージを送信したとき，x 方向の各リンクに流れる最大のメッセージ数は i 個となり，y 方向の各リンクに流れる最大のメッセージ数は j 個となる．各リンクのバンド幅は，各リンクに同時に流れるメッセージ数により，フェアに等分されるため，x 方向のバンド幅は $1/i$，y 方向のバンド幅は $1/j$ となる．各ノードが享受できるバンド幅は経路の最少のバンド幅で律速されるので，各ノードは，$1/max(i, j)$ のバンド幅でメッセージを送信することができる．

表 1　各ネットワーク構成に必要な処理 Step の一覧
Table 1　A list of processing steps that are required by each network configuration.

正方形	奇数	Step 1，Step 2
	偶数	Step 1，Step 2，Step 3
長方形	奇数 × 奇数	Step 1，Step 2，Step 3_Odd
	偶数 × 奇数	Step 1，Step 2，Step 3_Odd，Step 4_Odd
	奇数 × 偶数	Step 1，Step 2，Step 3_Even，Step 4_Even，Step 5
	偶数 × 偶数	Step 1，Step 2，Step 3_Even，Step 4_Even，Step 5

$i \neq j$ の場合，x 方向，y 方向のリンクに流れるメッセージの数が異なるため，リンクのバンド幅を効率的に使用できていない部分が生じる．x 方向，y 方向のリンクに流れるメッセージの数を等しくするために，(i,j) のノードへのメッセージ送信と (j,i) のノードへのメッセージ送信を組み合わせて同時に送るようにする．この 2 つのメッセージを組み合わせることにより，x 方向，y 方向の負荷バランスがとれ，バンド幅の利用率が向上する．各ノードはバンド幅を $1/(i+j)$ として，送信することができる．

3.1　奇数サイズの正方形 2 次元 Mesh

A2AT では，最大 5 step で全対全通信を行う．**表 1** に，各ネットワーク構成に必要な処理 Step の一覧を示す．$N_x \times N_y$（$N_x \geq N_y$）の長方形状の構成の場合，横に並ぶノード数 N_x が奇数，縦に並ぶノード数 N_y が奇数であるネットワークの構成を，以降,「奇数 × 奇数」と表記する．

まず，奇数サイズの正方形 2 次元 Mesh の場合について述べ，偶数サイズ，長方形，2 次元 Torus の順に拡張していく．**図 3** に，Step 1，Step 2 の処理を示す．Step 3〜5 に関しては，3.2 節以降で述べる．

Step 1

for $i = 1$　to　$\frac{N-1}{2}$ **do**
　$send(i,0); send(0,i);$ {(a)(c)}
　$send(-i,0); send(0,-i);$ {(b)(d)}
end for

Step 1 では，送信元ノードは x 方向および，y 方向の軸上にあるノードに対して，図 3 (a)〜(d) のように，x 軸上にあるノードへのメッセージの送信と y 軸上にあるノードへのメッセージの送信を組み合わせて行う．全ノードが同時にこの操作を行うことにより，すべてのリンクを使用し，送信を行うことができる．

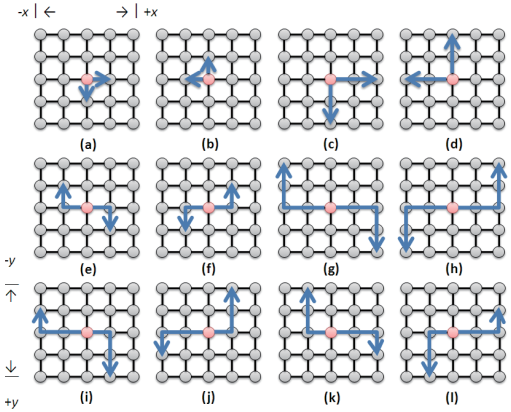

図 3　奇数サイズ正方形 2 次元 Mesh
Fig. 3　Odd size square 2D mesh.

Step 2

for $i = 1$　to　$\frac{N-1}{2}$ **do**
　for $j = 1$　to　$\frac{N-1}{2}$ **do**
　　$send(i,j); send(-j,-i);$ {(e)(g)(i)(k)}
　　$send(i,-j); send(-j,i);$ {(f)(h)(j)(l)}
　end for
end for

Step 2 では，Step 1 で送信が完了した以外の位置にあるノードに対して，図 3 (e)〜(l) のように，x 方向，y 方向の各リンクには，ともに $(i+j)$ 個のメッセージが流れるように組み合わせて送信を行う．この場合も，すべてのリンクを使用し，送信を行うことができる．

3.2　偶数サイズの正方形 2 次元 Mesh

1 辺のサイズ N が偶数である 2 次元 Mesh では，このネットワーク内にある最大の奇数サイズのネットワークについて，Step 1，Step 2 と順に処理し，その後，**図 4** に示す，Step 3 で余った各行の処理を行う．

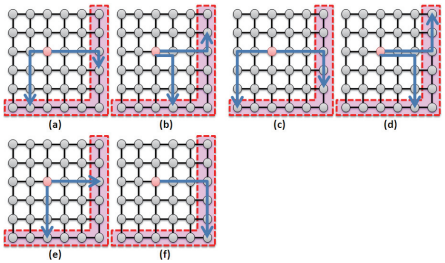

図 4　偶数サイズ正方形 2 次元 Mesh（Step 3）
Fig. 4　Even size square 2D mesh.

Step 3

for $i = 1$　to　$\frac{N}{2} - 1$ **do**
　$send(\frac{N}{2}, i); send(-i, \frac{N}{2});$ {(a)(c)}
　$send(\frac{N}{2}, -i); send(i, \frac{N}{2});$ {(b)(d)}
end for
$send(\frac{N}{2}, 0); send(0, \frac{N}{2});$ {(e)}
$send(\frac{N}{2}, \frac{N}{2});$ {(f)}

奇数サイズのときと同様の考えで，図 4 (a)〜(d) のように，x 方向，y 方向の各リンクには，ともに $(N/2+i)$ 個のメッセージが流れるように組み合わせて送信を行う．次に，図 4 (e) に示すように，水平垂直軸上にあるノードの処理を行い，最後に，図 4 (f) に示すように，対角線にあるノードの処理を行う．対角線にあるノードの処理は，NCT が 1 となるが，x 方向，y 方向のすべてのリンクを使用し，送信を行うことができる．Step 3 においても，つねにすべてのリンクを使用して送信を行っている．

3.3　長方形 2 次元 Mesh

$N_x \times N_y$（$N_x \geq N_y$）の長方形状の構成の場合についても同様に考えることができる．

ネットワークの構成が，奇数 × 奇数の場合，3 step で，偶数 × 奇数の場合は，4 step で，奇数 × 偶数，偶数 × 偶数の場合は，5 step で処理を行う．

まず，長方形に含まれる最大の正方形奇数サイズのネットワークについて，Step 1，Step 2 を順に処理する．次に，これに含まれない残りのノードに対する処理を順に行う．奇数 × 奇数および，偶数 × 奇数の場合，長方形に含まれる最大の正方形のネットワークは，奇数

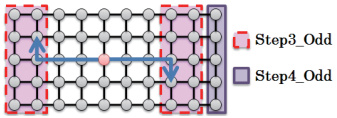

図 5　Step 3_Odd および Step 4_Odd での処理
Fig. 5　Processes for Step 3_Odd, Step 4_Odd.

サイズである．このとき，残りのノードに対しては，x 方向と y 方向の各リンクに，ともに $(i+j)$ 個のメッセージが流れるような組合せがない．そのため，**図 5** に示す Step 3_Odd のように，各ノードが享受できるバンド幅は，それぞれ $1/2i$ となる組合せでの処理を行う．

Step 3_Odd

for $i = \frac{N_y-1}{2} + 1$　to　$\lfloor \frac{N_x-1}{2} \rfloor$ **do**
　for $j = 1$　to　$(N_y - 1)/2$ **do**
　　$send(i, j); send(-i, -j);$
　　$send(i, -j); send(-i, j);$
　end for
　$send(i, 0); send(-i, 0);$
end for

偶数 × 奇数の場合，さらに残りのノードを Step 4_Odd で処理する．

Step 4_Odd

for $i = 1$　to　$\frac{N_y-1}{2}$ **do**
　$send(\frac{N_x}{2}, i); send(\frac{N_x}{2}, -i);$
end for
$send(\frac{N_x}{2}, 0);$

奇数 × 偶数の場合，偶数 × 偶数の場合，長方形に含まれる最大の正方形サイズのネットワークは，偶数サイズである．このときは，Step 3_Even のように処理を行う．Step 3_Even は，図 4 (f) に示す，偶数サイズの正方形 2 次元 Mesh のアルゴリズムの Step 3 の最終 step の処理を，$send(N_y/2, N_y/2); send(-N_y/2, N_y/2);$ として処理を行う．

Step 3_Even

for $i = 1$　to　$\frac{N_y}{2} - 1$ **do**

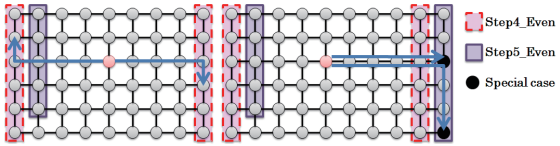

図 6　Step 4_Even および Step 5 での処理
Fig. 6　Processes for Step 4_Even, Step 5.

$send(N_y/2, i); send(-i, N_y/2);$
$send(N_y/2, -i); send(i, -N_y/2);$
end for
$send(N_y/2, 0); send(0, N_y/2);$
$send(N_y/2, N_y/2); send(-N_y/2, N_y/2);$

Step 4_Even，Step 5 では，同様の考えで，**図 6** に示すように，各ノードが享受できるバンド幅をそれぞれ $1/2i$ とし，残りのノードの処理を行う．

Step 4_Even

for $i = \frac{N_y}{2} + 1$　to　$\lfloor \frac{N_x}{2} - 1 \rfloor$ **do**
　for $j = 1$　to　$\frac{N_y}{2} - 1$ **do**
　　$send(i, j); send(-i, -j);$
　　$send(i, -j); send(-i, j);$
　end for
　$send(i, 0); send(-i, 0);$
　$send(i, N_y/2); send(-i, N_y/2);$
end for

Step 5

if N_x**mod**2 == 0 **then**
　for $i = 1$　to　$\frac{N_y}{2} - 1$ **do**
　　$send(\frac{N_x}{2}, i); send(\frac{N_x}{2}, -i);$
　end for
　$send(\frac{N_x}{2}, 0); send(\frac{N_x}{2}, \frac{N_y}{2});$ {(Special case)}
end if
for $i = 1$　to　$\frac{N_y}{2} - 1$ **do**
　$send(-\frac{N_y}{2}, i); send(-\frac{N_y}{2}, -i);$
end for
$send(-\frac{N_y}{2}, 0);$

3.4　2 次元 Torus への拡張

2 次元 Torus ネットワークについても 2 次元 Mesh ネットワークと同様に考えることができる．つねに全方向のリンクを使用するために，通信コントローラを 4 個使い，$+x$ 方向，$-x$ 方向，$+y$ 方向，$-y$ 方向の 4 方向を組み合わせて同時に送信を行う．

1 辺のサイズ N が奇数である正方形 2 次元 Torus ネットワークでは，奇数サイズの正方形 2 次元 Mesh ネットワークのアルゴリズムを，NCT を 4 として処理することにより，つねに 4 方向のリンクを使用することができる．これにより，各リンクに流れるメッセージ数が均一となる．

1 辺のサイズ N が偶数である正方形 2 次元 Torus ネットワークでは，ネットワーク内にある最大の奇数サイズのネットワークについて Step 1，Step 2 と順に処理する．残りのノードについては，Step 3_T で処理を行う．Torus ネットワークにおいて，偶数サイズでは，ちょうど中間の距離にあるノードへの送信には，2 通りの経路がある．このときは送信先に応じて，逆方向のリンクも使用し，$+x$ 方向，$-x$ 方向，$+y$ 方向，$-y$ 方向の各リンクに流れるメッセージの数が $(N/2 + i)$ となるように，送信を行う．

Step 3_T

for $i = 0$　to　$\frac{N}{2} - 1$ **do**
　$send(\frac{N}{2}, i); send(-i, -\frac{N}{2});$
　$send(-\frac{N}{2}, -i); send(i, \frac{N}{2});$
end for
$send(\frac{N}{2}, 0); send(0, \frac{N}{2});$
$send(-\frac{N}{2}, -\frac{N}{2});$

長方形の奇数 × 奇数 Torus ネットワークでは，長方形の奇数 × 奇数 Mesh ネットワークのアルゴリズムを，NCT を 4 として処理することにより，つねに 4 方向のリンクを使用することができる．これにより，各リンクに流れるメッセージ数は均一となる．

4. アルゴリズムの性能

本方式は，各ノードが同一サイズのメッセージを同時に送信を行うことを前提としている．また，本方式により各ステップで全メッセージが利用できるバンド幅は同一となる．よって，各 step における全ノードの送信時間は同じである．このため，各 step 間で同期をとらなくてもすべて同時に終了する．したがって，全対全通信時間は，各 step での通信時間の総和となる．

本方式による通信時間の理論値を求めるにあたり，リンクのバンド幅を 1 に正規化，さらにメッセージサイズを 1 と正規化する．A2AT では，距離 i にメッセージを送信する場合を考えると，メッセージの重なりは i となり，バンド幅は $1/i$ となるため，通信時間は i となる．通信時間の評価にあたっては，メッセージサイズが大きい場合を想定し，通信の立ち上がり時間を無視して考えるものとする．

4.1 奇数サイズ正方形 2 次元 Mesh・Torus

奇数サイズの正方形 2 次元 Mesh の場合，Step 1 では，$+x$ 方向と $+y$ 方向，$-x$ 方向と $-y$ 方向を組み合わせて処理を行う．これを距離 1 への送信から距離 $\lfloor (N-1)/2 \rfloor$ への送信まで繰り返す．以降，$S = \lfloor (N-1)/2 \rfloor$ とする．このため，Step 1 における通信時間は，

$$T_{Step1} = \sum_{i=1}^{S} i \times 2 \tag{3}$$

となる．

同様にして Step 2 では，$(+x, +y)$ 方向の (i, j) へのメッセージ送信と，$(-x, -y)$ 方向の (j, i) へのメッセージ送信を，組み合わせて処理を行う．同様に，$(+x, -y)$ 方向，$(-x, +y)$ 方向も処理を行う．x 方向，y 方向ともに，メッセージの重なりは $(i + j)$ なので，Step 2 における通信時間は，

$$T_{Step2} = \sum_{i=1}^{S} \sum_{j=1}^{S} (i + j) \times 2 \tag{4}$$

となる．

式 (3)，(4) の合計が，本方式の奇数サイズの正方形 2 次元 Mesh の通信時間の理論値となる．

$$T_{odd} = T_{Step1} + T_{Step2} \tag{5}$$

$$= S(S+1)(2S+1) \tag{6}$$

$$= \frac{N(N+1)(N-1)}{4} \tag{7}$$

この値は，式 (1) に示す Mesh ネットワークにおける理論的下限と等しくなっている．

奇数 2 次元 Torus において，各 Step の通信時間は 1/2 となるため，奇数サイズの正方形 2 次元 Torus の通信時間は，$N(N+1)(N-1)/2$ となる．この値は，式 (2) に示す Torus ネットワークにおいても理論的下限と等しくなっている．

4.2 偶数サイズ正方形 2 次元 Mesh・Torus

偶数サイズの正方形 2 次元 Mesh の場合，Step 3 では，$(+x, +y)$ 方向に $(N/2, i)$ へのメッセージ送信と，$(-x, +y)$ 方向に $(i, N/2)$ へのメッセージ送信を組み合わせて処理を行う．同様に，$(+x, -y)$ 方向，$(+x, +y)$ 方向も x 方向，y 方向ともに，メッセージの重なりは $(N/2 + i)$ として処理を行う．次に，$+x$ 方向の距離 $N/2$ へのメッセージ送信と，$+y$ 方向の距離 $N/2$ へのメッセージ送信を組み合わせて処理を行うため，$N/2$ の通信時間がかかる．最後に，$(+x, +y)$ 方向に $(N/2, N/2)$ へのメッセージ送信を行う．よって，Step 3 における通信時間は，

$$T_{Step3} = \sum_{i=1}^{S} (\frac{N}{2} + i) \times 2 + \frac{N}{2} + \frac{N}{2} \tag{8}$$

となる．ただし，$S = (N/2) - 1$ とする．

Step 1，Step 2 の通信時間である，式 (6) に，Step 3 の通信時間 $(S + 1)(S + N)$ を加算すれば，偶数サイズの正方形 2 次元 Mesh の通信時間の理論値

$$T_{even} = \frac{N^3}{4} \tag{9}$$

となる．この値は，式 (1) に示す Mesh ネットワークにおける理論的下限と等しくなっている．

偶数 2 次元 Torus において，Step 3 の通信時間は 1/2 となるため，偶数サイズの正方形 2 次元 Torus の通信時間は，$N^3/2$ となる．この値は，式 (2) に示す Torus ネットワークにおいても，理論的下限と等しくなっている．

4.3 長方形 Mesh

ネットワークの構成を $N_x \times N_y$（$N_x \geq N_y$）とする．まず，奇数 × 奇数の場合，偶数 × 奇数の場合について考える．長方形に含まれる最大の正方形奇数サイズである $N_y \times N_y$ のネットワークについて処理するため，式 (6) に $S = (N_y-1)/2$ を代入すると，$N_y(N_y+1)(N_y-1)/4$ となる．以降，$S_1 = \lfloor (N_y-1)/2 \rfloor, S_2 = \lfloor (N_x-1)/2 \rfloor$ とする．Step 3_Odd における通信時間を求める．各ノードは，距離 (i,j) へのメッセージを同時に 2 つ送るため，重なりは $2i$ となる．ただし，$i > j$ とする．水平垂直軸上のノードの処理に関しても，重なりは $2i$ として処理を行うため，

$$T_{S3odd} = \sum_{i=S_1+1}^{S_2} \sum_{j=1}^{S_1} 2i \times 2 + \sum_{i=S_1+1}^{S_2} 2i \tag{10}$$

$$= -(S_1 - S_2)(S_1 + S_2 + 1)(2S_1 + 1) \tag{11}$$

となる．$S_1 = (N_y-1)/2$，$S_2 = (N_x-1)/2$ より，$-N_y(N_y - N_x)(N_y + N_x)/4$ となる．この値に，Step 1，Step 2 の通信時間である $N_y(N_y+1)(N_y-1)/4$ を加算すると，

$$T_{oo} = \frac{(N_x-1)(N_x+1)N_y}{4} \tag{12}$$

となり，Mesh ネットワークの理論的下限の通信時間と等しい．

偶数 × 奇数の場合は，Step 4_Odd において残りの 1 行の処理を行う．各ノードは，距離 $(N_x/2, i)$ へのメッセージを同時に 2 つ送るため，重なりを N_x として処理を行う．最後に，距離 $N_x/2$ へのメッセージを 1 つ送るため，

$$T_{S4odd} = \sum_{i=1}^{S_1} N_x + N_x/2 \tag{13}$$

$$= N_x(2S_1 + 1)/2 \tag{14}$$

となる．$S_1 = (N_y-1)/2$，$S_2 = N_x/2-1$ より，$T_{S3odd} + T_{S4odd} = -N_y(N_y^2 - N_x^2 - 1)/4$ となる．この値に，Step 1，Step 2 の通信時間である $N_y(N_y+1)(N_y-1)/4$ を加算すると，

$$T_{eo} = \frac{N_x^2 N_y}{4} \tag{15}$$

となり，Mesh ネットワークの理論的下限の通信時間と等しい．

次に，奇数 × 偶数の場合，偶数 × 偶数の場合について考える．長方形に含まれる最大の正方形奇数サイズである $(N_y-1) \times (N_y-1)$ のネットワークについて処理する．このときの通信時間，式 (6) は $S = N_y/2 - 1$ より，$N_y(N_y-2)(N_y-1)/4$ となる．Step 3_Even では，x 方向，y 方向の各 1 行に対し処理を行う．重なりを $(N_y/2+i)$ として送信を行う．水平垂直軸上の処理において，x 方向，y 方向ともに，重なりは $N_y/2$，対角線上のノードに対する処理では，x 方向，y 方向ともに，重なりは N_y となるため，

$$T_{S3even} = \sum_{i=1}^{S_1} (N_y/2 + i) \times 2 + N_y/2 + N_y \tag{16}$$

$$= 2S_2^2 + 2N_yS_2 + 2S_2 + 3N_y/2 \tag{17}$$

となる．

次に，Step 4_Even における通信時間を求める．各ノードは，距離 (i,j) へ，重なりを $2i$ として送信を行う．ただし，$i > j$ とする．水平垂直軸上，対角線上のノードの処理に関しても，重なりを $2i$ として処理を行うため，

$$T_{S4even} = \sum_{i=S_1+2}^{S_2} \sum_{j=1}^{S1} 2i \times 2 + \sum_{i=S_1+2}^{S_2} 2i + \sum_{i=S_1+2}^{S_2} 2i \tag{18}$$

$$= -2(S_1+1)(S_1 - S_2 + 1)(S_1 + S_2 + 2) \tag{19}$$

となる．

最後に，Step 5 で残りのノードの処理を行う．各ノードは，距離 $(N_y/2, i)$ へ，重なりを N_y として送信を行う．最後に，距離 $N_y/2$ へのメッセージを 1 つ送る．偶数 × 偶数の場合，各ノードは，Step 5 のはじめに，距離 $(N_x/2, i)$，$(N_x/2, N_y/2)$ の処理を重なりを N_x として行うため，

$$T_{Step5} = \sum_{i=1}^{S_1} N_x + N_x + \sum_{i=1}^{S_1} N_y + N_y/2 \tag{20}$$

$$= 2S_1N_y + N_y + 2S_1N_x + 2N_x/2 \tag{21}$$

となる．奇数 × 偶数の場合は，式 (20) の第 1 項と第 2 項は除かれるため，$(2S_1+1)N_y/2$ となる．

奇数 × 偶数の場合，Step 3_Even，Step 4_Even，Step 5 における通信時間は，$S_1 = N_y/2-1$，$S_2 = (N_x-1)/2$ より，それぞれ $T_{S3even} = 3N_y^2/4$，$T_{S4even} = -N_y(N_y - N_x + 1)(N_y + N_x + 1)/4$，$T_{Step5} = N_y(N_y-1)/2$ となる．これら値に，Step 1，Step 2 の通信時間である $N_y(N_y-2)(N_y-1)/4$ を加算すると，

$$T_{oe} = \frac{(N_x - 1)(N_x + 1)N_y}{4} \tag{22}$$

となり，Mesh ネットワークの理論的下限の通信時間と等しい．

偶数 × 偶数の場合，Step 5 における通信時間は，$T_{Step5} = N_y(N_x + N_y - 1)/2$ となる．Step 3_Even，Step 4_Even の通信時間は，奇数 × 偶数の場合と同様であり，これらの値に，Step 1，Step 2 の通信時間である $N_y(N_y - 2)(N_y - 1)/4$ を加算すると，

$$T_{ee} = \frac{N_x^2 N_y}{4} \tag{23}$$

となり，Mesh ネットワークの理論的下限の通信時間と等しい．

4.4　既存のアルゴリズムとの比較

A2AND において NCT を 1 とした場合，送信順は任意であるとして通信時間を考えることができる．1 辺のサイズ N が奇数である正方形のネットワークの場合，宛先ノードの位置は，送信元ノードの水平垂直軸上，対角線上，およびその他の位置の 3 グループに分けることができる．各グループでは，バンド幅をそれぞれ $1/i$，$1/2i$，$1/max(i,j)$ として送信を行う．その他の位置にあるノードについて，バンド幅を $1/i$ として送信を行う位置にある宛先ノードと，$1/j$ として送信を行う位置にある宛先ノードは同じ距離で同じ数だけ存在する．そのため，バンド幅を $1/j$ として送信を行う位置に存在するノードの通信時間を 2 倍すればよい．4 方向に対し送信を行うため，A2AND の通信時間 T_{a2and} は，

$$T_{a2and} = \sum_{i=1}^{S} i \times 4 + \sum_{i=1}^{S} i \times 4 + \sum_{i=1}^{S} \sum_{j=i+1}^{S} j \times 8 \tag{24}$$

$$= 4S(S+1)(2S+1)/3 \tag{25}$$

となる．$S = (N-1)/2$ より，通信時間は，$N(N+1)(N-1)/3$ となる．

また，A2AT において，NCT を 1 とした場合，送信順は任意である．そのため，通信時間は同様に $N(N+1)(N-1)/3$ である．このように，NCT を 1 としたとき，A2AT と A2AND は同等の性能となる．

次に，NCT を増やした場合について考える．A2AND では，NCT が 2 以上とした場合，各リンクに流れるメッセージ数の変化が異なり，解析的に通信時間を求めることが難しい．よって，シミュレーションにより比較する．正方形 2 次元 Torus32 × 32 での mfa [11] によるシミュレーション結果を**図 7** に示す．横軸は，NCT，縦軸は，全対全通信時間である．

A2AT では，NCT の増加に従ってリンクを効率的に使用しているため，通信時間が短縮されている．A2A でも，NCT を増加させると通信時間が短縮されている．しかし，A2AT のような向上は得られない．A2AND については若干の通信時間の短縮があるものの，ほとんど変化がない．A2A，A2AND では，NCT を 2 以上としても同一方向への送信が重なるため，NCT を増加させても，リンクを効率的に使用しておらず，その方向のバンド幅がネックとなるためである．

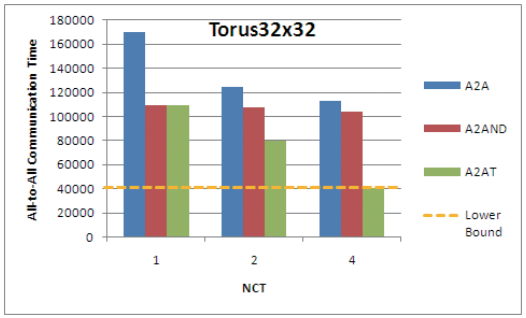

図 7　全対全通信時間（Torus32 × 32）
Fig. 7　All-to-All communication time (Torus32 × 32).

NCT を 2 としたとき，A2AT による通信時間は，Mesh ネットワーク上では，理論的下限となる．Torus ネットワークにおいては，つねに逆のリンクを使用する組合せで送信を行っているため，A2AT の Mesh ネットワーク上での性能よりも良くなる．Torus ネットワーク上では，NCT を 4 としたとき，理論的下限となる．A2AT では，Mesh ネットワークでは，x 方向，y 方向への送信を順に，Torus ネットワークでは，$+x$ 方向，$+y$ 方向，$-x$ 方向，$-y$ 方向への送信と順に行う．各方向のリンクを順に使用するようにスケジューリングをしているため，A2AT では NCT の増加による性能向上が得られる．A2AT は，NCT が理論的下限の性能を引き出す NCT に満たなかった場合でも，リンクを効率良く使用しており，同等以上の性能を実現できる．

5. ま と め

Mesh・Torus ネットワークと複数の通信コントローラを持つシステム向けに，全対全通信を最適に実行するアルゴリズムを提案した．提案したアルゴリズムでは，複数のメッセージを同時に送受信し，その送信方向を全経路の使用効率が一定になるように組み合わせることでリンクを有効活用させている．このアルゴリズムにより，正方形 2 次元 Mesh ネット

ワーク・Torus ネットワーク，長方形状の Mesh ネットワークでの全対全通信において通信時間の理論的下限の性能を引き出すことができることを示した．

提案アルゴリズムでは，既存のアルゴリズムとは異なり，同一方向への連続した送信を行わない．既存のアルゴリズムと比べ，リンクを効率良く使用しているため，NCT が少ない場合においても，同等以上の性能である．本方式は，従来の最適な全対全通信アルゴリズムのようにフェーズを分ける必要はない．複数のメッセージを順次並行して送信するだけでよいため，フェーズ間のバリア同期のためのオーバヘッドもなく，実装が容易である．ノード数 N に対し，各フェーズで同期には $\log N$ オーダの回数の通信が必要となる．メッセージ長が長い場合を想定しており，同期コストは，全体の処理にかかった時間に比べれば，小さいものである．しかし，従来アルゴリズムにあったフェーズごとに，$(N-1)$ 回の同期するための通信が不要となるメリットは大きい．

本論文では，2 次元に限定したアルゴリズムを提案したが，3 次元への拡張も可能である．また，3 次元 Mesh・Torus の構成をとるシステムにおいても，切り出された 2 次元の範囲のみで全対全通信を行うアプリケーションもあり，本方式は広い範囲のシステムに適用可能である．

今後の課題として，3 次元への拡張と今回示せなかった長方形の奇数 × 奇数以外の Torus に関して，研究をすすめていく．

参　考　文　献

1) Scott, D.S.: Efficient all-to-all communication patterns in hypercube and mesh topologies, *6th Distributed Memory Computing Conference*, pp.398–403 (1991).
2) 堀江健志，林　憲一：トーラスネットワークにおける最適全対全通信方式，情報処理学会論文誌，Vol.34, No.4, pp.628–637 (19930415).
3) Tseng, Y.-C. and Gupta, S.K.S.: All-to-all personalized communication in a wormhole-routed torus, *IEEE Trans. Parallel and Distributed Systems*, Vol.7, No.5, pp.498–505 (1996).
4) Almási, G., Heidelberger, P., Archer, C.J., Martorell, X., Erway, C.C., Moreira, J.E., Steinmacher-Burow, B. and Zheng, Y.: Optimization of MPI collective communication on BlueGene/L systems, *ICS '05: Proc. 19th annual international conference on Supercomputing*, New York, NY, USA, pp.253–262, ACM (2005).
5) Kumar, S., Sabharwal, Y., Garg, R. and Heidelberger, P.: Optimization of All-to-All Communication on the Blue Gene/L Supercomputer, *37th International Conference on Parallel Processing*, pp.320–329 (2008).
6) Bruck, J., Ho, C.-T., Kipnis, S. and Weathersby, D.: Efficient algorithms for all-to-all communications in multi-port message-passing systems, *SPAA '94: Proc. 6th annual ACM symposium on Parallel algorithms and architectures*, New York, NY, USA, pp.298–309, ACM (1994).
7) Tipparaju, V. and Nieplocha, J.: Optimizing All-to-All Collective Communication by Exploiting Concurrency in Modern Networks, *SC '05: Proc. 2005 ACM/IEEE conference on Supercomputing*, Washington, DC, USA, p.46, IEEE Computer Society (2005).
8) Ajima, Y., Sumimoto, S. and Shimizu, T.: Tofu: A 6D Mesh/Torus Interconnect for Exascale Computers, *Computer*, Vol.42, No.11, pp.36–40 (2009).
9) Abts, D. and Weisser, D.: Age-based packet arbitration in large-radix k-ary n-cubes, *SC '07: Proc. 2007 ACM/IEEE conference on Supercomputing*, New York, NY, USA, pp.1–11, ACM (2007).
10) MPICH. http://www.mcs.anl.gov/research/projects/mpi/
11) 石畑宏明：大規模並列コンピュータ用通信ネットワークのシミュレーション方式（ネットワーク，SWoPP 佐賀 2008-2008 年並列/分散/協調処理に関する『佐賀』サマー・ワークショップ），電子情報通信学会技術研究報告．CPSY，コンピュータシステム，Vol.108, No.180, pp.55–60 (20080729).

(平成 21 年 10 月 2 日受付)
(平成 22 年 2 月 25 日採録)

高上　治之

2009 年東京工科大学コンピュータサイエンス学部卒業．現在，同大学大学院バイオ・情報メディア研究科在学．

矢崎　俊志

2007 年電気通信大学大学院電気通信学研究科情報工学専攻博士後期課程修了．2009 年東京工科大学助教．2010 年電気通信大学情報基盤センター助教．並列計算機，生活支援システム，算術論理演算回路に関する研究に従事．博士（工学）．信学会，パルテノン研究会各会員．

安島雄一郎（正会員）

1997 年東京大学工学部電気工学科卒業．2002 年同大学大学院工学系研究科博士課程修了．博士（工学）．同年（株）富士通研究所入社．現在，富士通（株）次世代テクニカルコンピューティング開発本部に勤務．インターコネクトアーキテクチャの開発に従事．

清水　俊幸（正会員）

1986 年東京工業大学工学部電子物理学科卒業．1988 年同大学大学院理工学研究科修士課程修了．同年（株）富士通研究所入社．並列計算機アーキテクチャの研究に従事．現在，富士通（株）次世代テクニカルコンピューティング開発本部に勤務．HPC システム，インタコネクトアーキテクチャの開発に従事．信学会会員．

石畑　宏明（正会員）

1980 年早稲田大学理工学部卒業．同年（株）富士通研究所入社．2007 年東京工科大学教授．並列コンピュータアーキテクチャの研究に従事．1992 年元岡賞受賞，博士（工学）．信学会，IEEE 各会員．