Vol.2010-HPC-126 No.14
2010/8/3

パケットペーシングを用いた最適全対全通信アルゴリズムのシミュレーション評価

柴村英智[†1] 三輪英樹[†2] 薄田竜太郎[†1]
平尾智也[†1] 安島雄一郎[†2] 三吉郁夫[†2]
清水俊幸[†2] 石畑宏明[†3] 井上弘士[†4]

本稿では，複数のメッセージを同時送受信可能な 2 次元トーラス網上で、ネットワークのバンド幅を有効に活用する最適全対全通信アルゴリズム（A2AT）の性能評価について述べる．まず，従来の全対全通信アルゴリズムとの性能比較を行った後，パケットペーシングを適用した最適全対全通信の性能について，NSIM と呼ぶインターコネクトシミュレータを用いた定量的な評価を行った．その結果，A2AT は他の全対全通信アルゴリズムよりも良い通信性能を実現することがわかった．また，A2AT にパケットペーシングを適用することにより，さらに高いリンクバンド幅を維持した高速な実行を達成でき，バンド幅を積極的に利用する A2AT のようなアプリケーションにペーシングが効果的であることを確認した．さらに，詳細なシミュレーション解析に基づき，アルゴリズムに最適化したペーシングを行った結果，最大で理想実行時間の 107.5%で実行できることが明らかになった．

Simulation Evaluation of an Optimal All-to-all Communication Algorithm using Packet Pacing

Hidetomo Shibamura ,[†1] Hideki Miwa ,[†2]
Ryutaro Susukita ,[†1] Tomoya Hirao ,[†1]
Yuichiro Ajima ,[†2] Ikuo Miyoshi ,[†2]
Toshiyuki Shimizu ,[†2] Hiroaki Ishihata [†3]
and Koji Inoue [†4]

This paper presents quantitative performance evaluation of A2AT, an optimal all-to-all communication algorithm. The algorithm exploits the network bandwidth of two-dimensional torus network which nodes can transmit and receive multiple messages simultaneously. This algorithm is compared with conventional all-to-all algorithms and we describe improvement of execution performance with explicit packet pacing by using NSIM, an interconnection network simulator. The result shows that A2AT achieves good performance than the other algorithms and holds higher link bandwidth with packet pacing. Then, the pacing optimized for the algorithm based on detailed simulation analysis gains 107.5% of the ideal all-to-all execution time.

1. はじめに

並列処理における全対全通信は，行列演算や FFT をはじめとする科学技術計算における重要な通信の一つである．個々の処理ノードが持つデータを全てのノードへ分配するため，ノード間通信を担うインターコネクト（相互結合網）への負荷が非常に高い．そこで，全対全通信はインターコネクトの評価項目の一つとして利用されるとともに，通信効率の良いアルゴリズムがこれまでに数多く考案されている．

近年では，ノードに複数の通信エンジンを搭載し，複数のリンクから同時にメッセージを送出可能なインターコネクトが提案されている[1)]．また，2 次元のメッシュ網やトーラス網を対象に，このような同時送信機構を活用する全対全通信アルゴリズム（A2AT）が提案されている[2)]．このアルゴリズムは，全てのリンクにおいて常に同数のメッセージが通過するようスケジューリングされており，リンクバンド幅を公平に共有することで全体の通信効率を高めている．しかし，積極的にリンクを利用するため，送信メッセージの送出時期といった通信タイミングに敏感になると考えられる．すなわち，実際のシステムでは，わずかなタイミングのずれによって通信混雑を招き，全体の通信性能を低下させると考えられる．

一方，インターネット環境を基盤とするマルチメディア配信やグリッドコンピューティングの分野では，複数のバースト的なトラフィックによってネットワークの通信帯域を過剰に圧迫することによる性能の低下が問題となっている．これを解決するために，パケットの送

†1 (財) 九州先端科学技術研究所
Institute of Systems, Information Technologies and Nanotechnologies
†2 富士通株式会社
Fujitsu Ltd.
†3 東京工科大学
Tokyo University of Technology
†4 九州大学
Kyushu University

Vol.2010-HPC-126 No.14
2010/8/3

出間隔を制御しトラフィックを平滑化する，ペーシングに関する研究や評価実験が盛んに行われている[3),4)]．このようなペーシング技術を利用したコモディティネットワークでは非常に高いネットワーク利用効率を達成しており，高スループットも重要視されるインターコネクトへの応用が期待できる．しかし，並列計算機におけるインターコネクトへの応用はほとんどなく，BG/L ではバリア同期のペーシング性を利用した全対全通信にとどまっている[5)]．

本研究では，全対全通信のような高トラフィックとなるアプリケーションにおいて，メッセージの送信時にアルゴリズムに適したパケットペーシングを明示的に行うことで通信混雑や輻輳を抑制し，全体の実行性能の向上を図ることを目的とする．

本稿では，通常のメッセージ通信と比較して大量のデータを授受する全対全通信について焦点を絞る．まず，A2AT と既存の全対全通信アルゴリズムの基本的な性能についてシミュレーションによる比較評価を行う．次に，パケットペーシングを施した A2AT の性能向上について定量的な評価を行う．そして，詳細なシミュレーション解析によって，アルゴリズムに最適化されたペーシング方針を導くことで，A2AT の実行性能の向上を図る．

以下，第 2 章では，本研究で評価対象とする最適全対全通信アルゴリズムについて述べる．第 3 章では，NSIM と呼ぶインターコネクトシミュレータについて概説する．第 4 章では，全対全通信アルゴリズムの比較実験やパケットペーシングによる評価実験について述べた後に考察する．そして，第 5 章でまとめと今後の課題について述べる．

2. 最適全対全通信アルゴリズム

本研究では，文献 2) で提案されている全対全通信アルゴリズム（A2AT）に焦点を絞る．

A2AT は，ノード間を双方向リンクで接続した一般的な 2 次元のメッシュ網やトーラス網を対象とする．また，各ノードは，複数リンクを用いて複数メッセージを同時送信，ならびに同時受信する機構を持つことを前提とする．

本アルゴリズムでは，常に全てのリンクを利用し，ある時刻において各リンクは全体を通して同じメッセージ数で共有するように送受信がスケジューリングされている．具体的には，2 次元格子において，自ノードから 4 つの座標軸方向と 4 つの象限への送信に区別される．自ノード座標を $(0,0)$ とし，送信先ノード座標を (x,y) と表す場合，A2AT では，$|x|+|y|$ = ホップ数 となるノードに同時送信を行う．ここで，座標軸方向への送信は，**図 1** のように $(i,0)$, $(-i,0)$, $(0,i)$, $(0,-i)$ となり，象限への送信は，**図 2** のように (i,j), $(-i,-j)$, (j,i), $(-j,-i)$ となる．そして，アルゴリズム全体では，自ノードを中心に送信先ノードが放射状に広がるように i,j を増加させる．

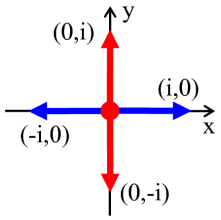

図 1 座標軸方向への送信

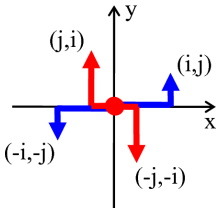

図 2 象限方向への送信

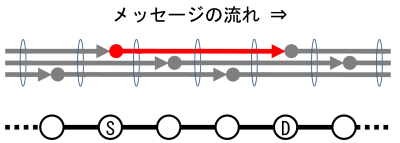

図 3 $+X$ 方向におけるリンクの共有

全ノードが一斉に同様の送信を行った場合，座標軸方向への送信時は，$+X$ 方向へのリンクは，i 個（= ホップ数）のメッセージで共有される．例えば，3 ホップの送信では，**図 3** のように，各リンクは 3 つのメッセージでリンクバンド幅を分かち合う．$-X$, $+Y$, $-Y$ 方向についても同様で，これらは他方向の通信を妨げることはない．一方，象限方向への送信時は，4 方向の各リンクを，$(i+j)$ 個（= ホップ数）のメッセージで共有する．

ここまではノード数が奇数の場合であるが，偶数時についても基本的な考え方は同じである．まず，包含する奇数ノードの処理を行い，その後余った行と列の処理を行う．アルゴリズムについては省略するが，この処理において，一般的な次元順ルーティングでは各リンクを公平に共有する方向へ送信できず，リンクを共有するメッセージ数が偏ってしまう．

以上のように，A2AT では，ホップ数と同数のメッセージが各リンクを通過するようスケジューリングし，リンクバンド幅を公平に共有することで全体の通信効率を高めている．

3. NSIM：インターコネクトシミュレータ

現在，我々は NSIM と呼ぶインターコネクトシミュレータを開発している[6)]．NSIM は，数万ノードを接続する次世代インターコネクトの性能評価を目的とし，設計・開発現場での実践的な利用を念頭に，現実的な時間内でシミュレーションを完了することを目指している．

これまでに開発されてきたシミュレータの多くは，具体的なアプリケーションの通信パターンを入力とするため，実際の並列計算機から取得した通信ログ，もしくは人工的に生成した通信パターンを利用することが多い．この手法は，実際に発生した通信事象に基づいた忠実なシミュレーションができる．一方，インターコネクトのトポロジや通信性能の差異によってプログラムの挙動が動的に変化する場合は，正確なシミュレーションを行うことが難しい．また，大規模インターコネクトのシミュレーション時には，相当する通信ログの取得限界問題が障壁となる[7),8)]．

そこで我々は，アプリケーションの振る舞いに忠実に従ったシミュレーションを行うために，NSIM では「MGEN プログラム」と呼ぶ MPI 互換のインタフェース（MGEN API）で記述されたプログラムを入力とし，実際にそのプログラムを駆動させることによってオンデマンドで通信パターンを生成している．一般的な MPI 通信ライブラリと同様に，利用者は MGEN ライブラリを用いてプログラムを記述し実行することで，所望のアプリケーションとインターコネクトのシミュレーションを行うことができる．また，インターコネクトに関する詳細仕様を設定ファイルで与えるため，実機のみならず新規のインターコネクトの性能予測ツールとしても利用できる．

NSIM は，並列離散事象シミュレーション（PDES）モデルに基づき，MPI で実装しているため，NSIM によるシミュレーションは多くのプラットフォームで実行可能である．また，従来のシミュレータ[9] と比較して，シミュレーションイベントの大幅な集約やシミュレーションエンジンの刷新によって飛躍的な高速化を達成している．

現在，MGEN ライブラリは，基本的な Send/Recv 通信プリミティブをはじめ，非同期通信や基本的な集団通信を提供しており，ランデブー通信，ゼロコピー通信といったオプションもサポートしている．また，将来のインターコネクト技術研究に向けた NSIM 専用の高レベル API を用意しており，複数 NIC による同時送受信や，パケットペーシングのためのパケット間ギャップの設定ができる．さらに，実システムに即した評価を行うために，OS ジッタやロードインバランスを踏まえたシミュレーションが可能である．なお，シミュレーションでは通信による実際のデータの授受は行われない．NSIM はインターコネクトの性能評価を目的としており，MGEN ライブラリは NSIM への通信イベントを生成するための枠組みゆえである．

NSIM は，シミュレーションの終了後，MGEN プログラムの予測実行時間（MGEN 実行時間），リンクバンド幅，リンクビジー率，バッファ利用率，総転送データ量といった数多くの統計情報を出力し，輻輳解析に役立つ支援機構も備えている．また，データ解析のための処理スクリプトやツールを取り揃えている．その一つに，NSIM が出力した通信ログを，MPICH2[10] の MPE ライブラリに含まれる Jumpshot の入力ファイル形式に変換するツールがあり，MGEN プログラム実行時の通信状況を可視化できる．

NSIM の具体的なメカニズムや実装などの詳細については，文献 6) を参照されたい．

表 1 評価対象システムの仕様一覧

パラメータ	設定値	パラメータ	設定値
トポロジ	2 次元トーラス網	MTU	2KiB
ルーティング方式	次元順+dateline	パケット長	32B〜2KiB(MTU)
パケット転送方式	VCT	パケットヘッダ長	32B
ノード間リンクバンド幅	4GB/s（単方向）	仮想チャネル数	2
ルーティング計算時間 (RC)	4ns	仮想チャネルバッファ	8KiB (MTU×4)
仮想チャネル設定時間 (VA)	4ns	フリット長	16B
スイッチ設定時間 (SA)	4ns	NIC 数	4（同時送受信可能）
フリット転送時間 (ST)	4ns	DMA 転送レート	16GB/s
スイッチ遅延時間	78ns	メモリバンド幅	16GB/s
ケーブル遅延時間	10ns	MPI 関数処理オーバヘッド *	200ns

* スタートアップレイテンシ／後処理遅延

4. 評 価 実 験

4.1 評価対象システム

本研究で扱うプログラムが動作するシステムは，近年のスーパーコンピュータ相当の機能・性能を有するものを想定する．以降，NSIM がシミュレーションのモデルとする並列計算機を**評価対象システム**と呼ぶ．同様に，NSIM によって駆動され通信パターンを生成するプログラムを**評価対象アプリケーション**と呼ぶ．

本評価実験で仮定する評価対象システムのインターコネクトの仕様を**表 1** にまとめる．なお，これらの仕様設計では，1) 現在の主流となる技術・性能を反映し，かつ，2) 評価対象アプリケーションの評価を容易にできる値とした．したがって，表中のインターコネクトやルータの仕様を持つシステムは実在しないことに注意されたい．例えば，リンクバンド幅は，現在の主流である InfiniBand 4x QDR（10Gbps×4）を想定している．また，実機における 1 ホップあたりのレイテンシは，BG/L では 89ns，一般的な InfiniBand スイッチでは 100〜140ns である．そこで，本実験では，1 ホップあたりのレイテンシを 100ns と設定し，スイッチ遅延は，ルーティング時間などを差し引いた 78ns とした．パケット長は 32B〜2KiB に設定し，その 4 倍の値を仮想チャネル 1 つあたりのバッファ容量とした．

4.2 実験 1：全対全通信アルゴリズムの性能比較

本実験では，A2AT と従来の全対全通信アルゴリズムの基本的な性能を比較する．まず，本実験における 2 次元トーラス網上での全対全通信の理論実行時間 (T_{ideal}) を算出する．2 次元トーラス網での理論的な実行時間は次式で表される．

表 2 実験 1 の評価パラメータ

パラメータ	設定値
評価対象アプリケーション	a2at，pwx，ring
NIC 数	4: a2at, 1: pwx, ring
ノード数	64(8 × 8)，256(16 × 16)，1,024(32 × 32)，4,096(64 × 64)
メッセージサイズ	64MiB 領域 ÷ ノード数

$$T_{ideal} = \frac{1}{2} \times \left\lfloor \frac{n}{2} \right\rfloor \times \left\lceil \frac{n}{2} \right\rceil \times n \times \frac{\text{メッセージサイズ}}{\text{リンクバンド幅}} = \frac{1}{2} \times n^2 \times \frac{\text{総データ転送量}}{\text{バイセクションバンド幅}}$$

この式から，2 次元トーラス網上で 1MiB メッセージの全対全通信を行った場合の理想実行時間は，パケットヘッダの転送オーバヘッドも考慮すると，8 × 8 では 21.3ms，同様に，9 × 9 では 24.0ms となる．

比較する評価対象アプリケーションには，pairwise exchange (以下，pwx) と ring を用いる．**表 2** に本実験に用いたパラメータを示す．なお，各ノードに分配するデータサイズは，問題サイズを固定するために，一定サイズ（64MiB）をノード数で割ったものとした．

4.2.1 実験 1：結果と考察

ノード数に応じた各アルゴリズムの実行時間を**図 4** に示す．この結果から，A2AT が pwx や ring よりも実行性能が良いことがわかる．また，**図 5** に示すリンク帯域の利用効率も A2AT の方が高い．これは，A2AT は，4 つの NIC を利用した同時送受信を行っており，他のアルゴリズムよりもリンクを利用する機会が多いためである．なお，64(8 × 8) ノード時の A2AT の実行時間は 71.3ms であり，理想実行時間の約 3.3 倍になっている．

次に，プログラム実行中の各次元方向 $(+X, -X, +Y, -Y)$ のリンクバンド幅，およびそれらの平均（図中赤線）を，時系列で表したものを**図 6** から**図 9** に示す．図 6 は，A2AT(8×8) のリンクバンド幅の推移である．実行開始時は，すべてのリンクを利用するため一時的に最大スループットとなるが，直ちに複数のメッセージでリンクを共有するため，通信混雑によって効率が低下する．また，実行の前半は，各リンクとも同数のメッセージで共有されるため各次元におけるバンド幅はほぼ同じである．しかし，後半は各次元のバンド幅が偏り，全体的に平均バンド幅が低下している．これは，ノード数が偶数のためである．第 2 節で述べたように，A2AT では，偶数ノード数時にプログラム後半部で最遠部（自ノードを中心とした場合に最も遠くなる辺部分）への送信を行う．その際，次元順ルーティングではリンク

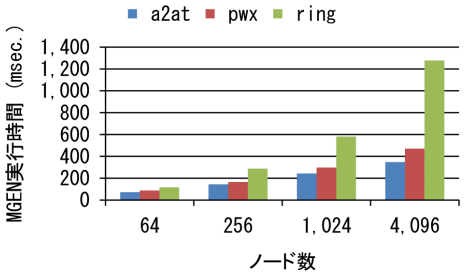

図 4 各アルゴリズムの実行時間

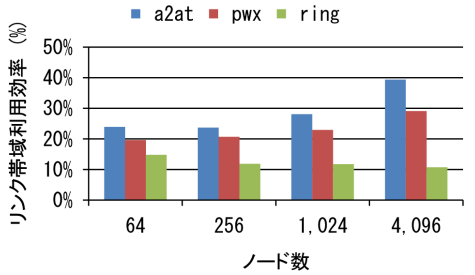

図 5 各アルゴリズムのリンク帯域利用効率

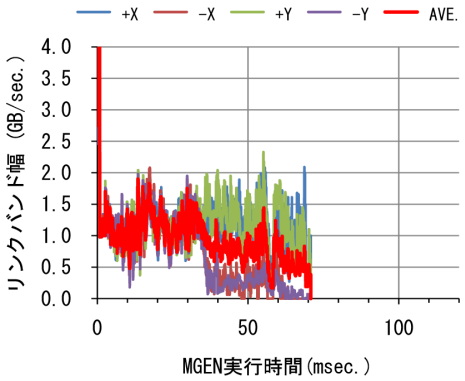

図 6 a2at(8x8) のリンクバンド幅の推移
（MGEN 実行時間：71.3ms）

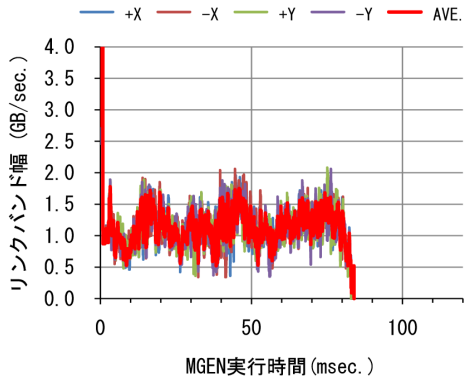

図 7 a2at(9x9) のリンクバンド幅の推移
（MGEN 実行時間：84.0ms）

を公平に共有する方向へ送信できず，リンクを共有するメッセージ数が偏るためである．

図 7 は，A2AT(9 × 9) のリンクバンド幅の推移を示している．奇数ノード数時は，リンクを共有するメッセージ数の偏りが起こらないため，各次元のバンド幅は一貫して同じになる．なお，この A2AT は他のノードサイズとの比較のために，メッセージサイズを 1MiB に設定している．したがって，総データ転送量は他よりも大きい．

同様に，**図 8** と図 9 は，それぞれ pwx と ring の推移である．A2AT と比較して，各次元の帯域効率は実行とともに大きく変動している．これは，単一 NIC であるため，4 リンクのうち一時期には 1 つのリンクからしかメッセージを送出することができないためである．

以上の結果から，A2AT では，全体でリンクのバンド幅を公平に使うスケジューリング

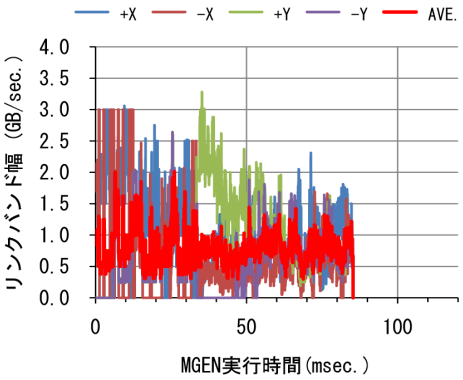

図 8 pwx(8x8) のリンクバンド幅の推移
（MGEN 実行時間：87.0ms）

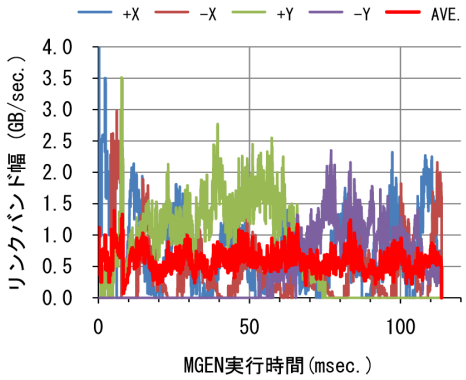

図 9 ring(8x8) のリンクバンド幅の推移
（MGEN 実行時間：163.0ms）

が成されており，pwx や ring と比較して実行性能が高くなっているといえる．また，奇数ノード数時は，常に一つのリンクを同数のメッセージで共有するため，実行中のリンクバンド幅にばらつきが少ない．9 × 9 の場合は，8 × 8 の pwx や ring よりもノード数が大きく，総データ転送量も多いにも関わらず実行時間は速くなっている．一方で，平均的なリンクバンド幅は 1GB/s と低く，時節で述べるパケットペーシングによる性能向上が期待できる．

4.3 実験 2：パケットペーシングによる性能改善

A2AT では，リンクを共有するメッセージ数は高々ホップ数と同じである．よって，理論的には，パケットの送出時に（（ホップ数 −1）× パケット長）の転送に要する時間以上の非送出期間（以下，**パケット間ギャップ**）を設けることで，他のパケットと干渉することなく交互にリンクを利用することが可能である．本実験では，A2AT に対して，全体を最適化するユニークなパケット間ギャップ値を用いてペーシング行い，その効果を明らかにする．

プロセスのロードインバランスが通信タイミングになんらかの影響を及ぼす可能性が考えらるため，理論的に最適なパケット間ギャップ値である（ホップ数 −1）に，様々な**パケット間ギャップバイアス**（以下，ギャップバイアス）を加えた実行を行う．すなわち，ギャップバイアスを変化させ，理論的に最適なパケット間ギャップ値を中心とした評価実験を行う．**表 3** に実験に用いた評価パラメータを示す．

パケット間ギャップ値は，MGEN ライブラリの一つである高レベル API 関数で設定した．なお，現在の NSIM では，パケット間ギャップの刻み幅を 1/8 に設定している．すなわち，最小でパケット長の 1/8 の長さのパケット間ギャップを挿入することができる．

表 3 実験 2 の評価パラメータ

パラメータ	設定値
評価対象アプリケーション	a2at
ノード数	64(8 × 8)，81(9 × 9)，· · ·，256(16 × 16)
メッセージサイズ	1MiB
パケット間ギャップ値	（ホップ数 −1）＋ パケット間ギャップバイアス
パケット間ギャップバイアス †	−2.0 ～ +16.0 (ステップ：1.0)
	0 ～ +6.0 (ステップ：0.125)

† ギャップバイアスは各実行で一様．

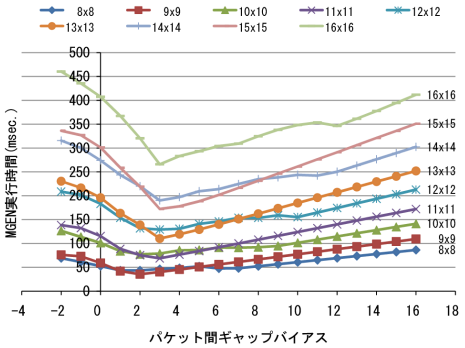

図 10 各ギャップバイアスにおける実行時間

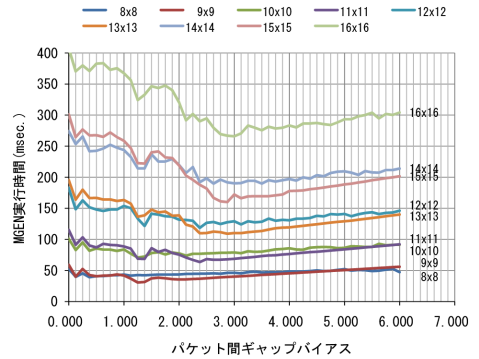

図 11 各ギャップバイアスにおける実行時間（詳細）

4.3.1 実験 2：結果と考察

図 10 に，各ノードサイズにおいて，−2 から +16 までギャップバイアスを変化させた場合の A2AT の実行時間を示す．そして，ペーシングの効果が顕著に表れているギャップバイアスが 0 から +6 について詳細に調査した結果を**図 11** に示す．これらのグラフから，A2AT の実行時間（MGEN 実行時間）を最小化する最適なギャップバイアスは一意には定まらず，概ね +1 から +3 周辺に分布していることがわかる．また，ノード数が偶数と奇数の違いに注目すると，例えばギャップバイアスが 3 の場合に，偶数ノード数の 14 × 14 よりもノード数が多い奇数ノード数の 15 × 15 の方が，実行時間が小さくなっている．これは，実験 1 で述べたように，奇数ノード数ではリンクを共有するメッセージ数の偏りが発生せず，ペーシングが効果的に作用しためと考えられる．

ギャップバイアスに対するリンクバンド幅，ならびにパケット待ち時間を，それぞれ，**図 12**

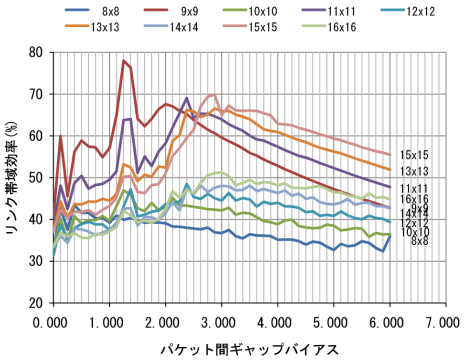

図 12 リンク帯域効率

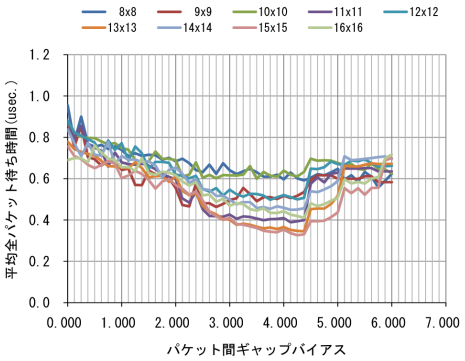

図 13 パケット待ち時間

と**図 13** に示す．ギャップバイアスが 0 の付近に着目すると，リンク帯域効率が低くパケット待ち時間が大きいことから輻輳が発生していると考えられる．ギャップバイアスが大きくなるとパケットの送出間隔が広がるため，パケットの流量が増加するとともに，リンク帯域効率の増加，ならびにパケット待ち時間が減少している．そして，パケット間ギャップの値がホップ数を上回るとリンク帯域効率が急激に増加し，1.250 から 1.375 付近で非常に高くなっている．なお，このように 1.250 付近で性能が良くなっている点については，4.4.1 節で考察する．さらにギャップバイアスが大きくなるとパケット間ギャップが広がりすぎることでリンクを流れるパケットが疎らになり，リンク帯域効率とパケット待ち時間が徐々に低下している．以上の解析によって，各ノード数における最適なパケット間ギャップバイアスが明らかになった．

次に，ノード数が 8×8 と 9×9 の場合に注目し，それぞれの通信性能について言及する．**図 14**，および**図 15** に，それぞれのノード数におけるリンクバンド幅の推移を示す．これらは，パケット間ギャップバイアスが 1.250 で，実行時間が最も小さくなる場合である．

それぞれの図において，不定間隔でスパイク状にリンクバンド幅が低下している箇所がある．これは，送信先に応じたフェーズの実行において，フェーズ間でメッセージの受信終了処理と送信開始処理を行っており，通信がないためである．なお，9×9 の場合では，20 回のフェーズ実行を行っており，リンクバンド幅の出力は 20 個のシーケンスに分類できる．

これらの図から，両方とも実行前半部は同様のリンクバンド幅の変化を示しており，後半部からはノード数の奇偶数よるリンクバンド幅の出力の差異が明らかである．また，ペーシングを行わない場合（図 6，図 7 ）と比較してリンクバンド幅が約 3 倍と大幅に向上して

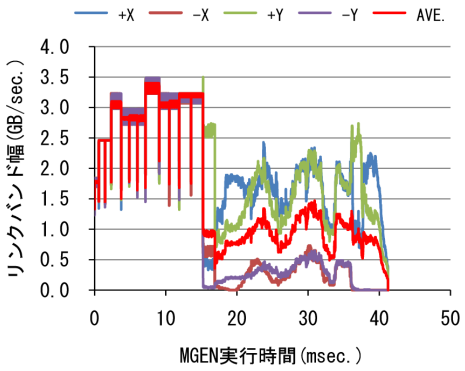

図 14 a2at(8×8) にペーシング（バイアス値：1.250）適用した場合のリンクバンド幅の推移（MGEN 実行時間：42.7ms）

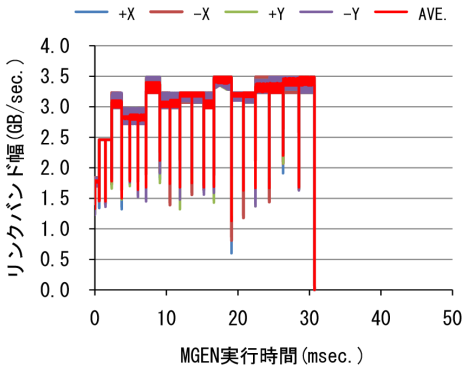

図 15 a2at(9×9) にペーシング（バイアス値：1.250）適用した場合のリンクバンド幅の推移（MGEN 実行時間：30.7ms）

いる．そして，リンク幅の変動も少なく安定した通信が行われていることがわかる．その結果，9×9 の場合については，ペーシングによって実行時間は 84.0ms から 30.7ms へと 2 倍以上高速化しており，理想実行時間の約 127.9%になっている．

以上の結果から，本実験では，A2AT にパケットペーシングを適用し最適なパケット間ギャップバイアス値で実行することによって，大幅な高速化を達成できることを確認した．一方，全体で一様のパケット間ギャップバイアス値を与えているため，フェーズによってはパケット間ギャップが大きすぎるといった，最適な実行ができていない可能性もある．

4.4 実験 3：シーケンス毎のチューニング

本実験では，A2AT を実行フェーズ毎に細分化し，各フェーズに最適なパケット間ギャップ値を与えて実行することで全体性能の向上を図る．

まず，A2AT が実行する一連のフェーズに対してシーケンス番号を与える．なお，ノード数が偶数の場合は 6 つのフェーズを実行し，奇数の場合は 4 つを実行する．ノード規模が大きいと一つのフェーズに複数のシーケンス番号が付与される場合もある．

次に，ギャップバイアス値を様々に変化させながら，単一シーケンスのみを NSIM でシミュレーションする．これによって，各シーケンスを最適に実行するパケット間ギャップバイアスが得られる．**表 4** に本実験に用いたパラメータを示す．

そして，最適なギャップバイアスによる各シーケンスを連続して実行することで，A2AT 全体の最適な実行とする．

表 4 実験 3 の評価パラメータ

パラメータ	設定値
評価対象アプリケーション	a2at
ノード数	81(9x9)
メッセージサイズ	1MiB
パケット間ギャップ値	（ホップ数 −1）＋ パケット間ギャップバイアス
ギャップバイアス	−3.0〜+3.0（ステップ：0.125）

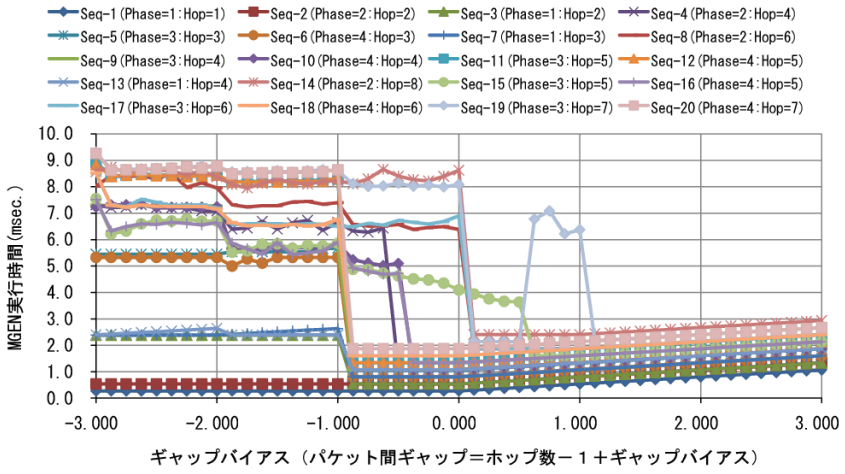

図 16 ギャップバイアスに対する各シーケンスの実行時間

4.4.1 実験 3：結果と考察

図 16 に，ギャップバイアスに対する各シーケンスの実行時間を示す．この図から，シーケンスを最適に実行するギャップバイアス値は，シーケンス毎に異なることがわかる．また，同じフェーズの実行であっても，シーケンス番号が異なる．すなわち，ホップ数が異なるとギャップバイアスも違う結果となった．

ギャップバイアスが小さい場合は，パケット間の送信間隔が狭いために通信混雑が発生し，シーケンスの実行は遅い．そして，ギャップバイアスが大きくなるにつれて送信タイミング遅くなるものの，リンクでの競合が減るためにパケットが順調に流れるため，それぞれのシーケンスにおける実行時間が速くなる．しかし，さらにギャップバイアスを大きくするとパケット間隔が開きすぎて逆に遅くなっていることがわかる．

一方，実験 2 では，9×9 の実行を最適化するギャップバイアス値は 1.250 であった．これは，1.250 における全てのシーケンスの実行時間の総和が最も小さくなることに結論できる．

次に，フェーズ毎に最適なパケット間ギャップ値を用いて A2AT をシミュレーションした．その際のリンクバンド幅の推移を**図 17** に示す．このシミュレーションから，A2AT の実行時間は 25.8ms であり，理想実行時間の 24ms に対して 107.5%と非常に高い実行性能を示している．また，この図における各シーケンスの実行では，5 つのシーケンス（第 8, 14，15，17，19 シーケンス）以外はリンクバンド幅を 100%利用しており，A2AT アルゴリズムの性能の高さを示している．この 5 つのシーケンスについて単体性能を確認したところ，それぞれの最高到達リンクバンド幅はすべて 3.5GB/s 前後となっており，第 15 シーケンス以外の 4 つの通信性能は順当である事がわかった．そこで，第 15 シーケンスについて通信ログや Jumpshot を用いた解析を行ったところ，第 14 シーケンス終了時のロードインバランス（以下，インバランス）が大きいことが原因であることがわかった．具体的には，第 14 シーケンスは他のシーケンスと比べてホップ数が 8 と最も大きく，各プロセスにおけるシーケンスの終了時刻に大きな差があった．そこで，すでに全ての受信を完了したプロセスは次の第 15 シーケンスを実行し，そのフェーズが発するパケットによって他のプロセスの第 14 シーケンスの通信を阻害していることを確認した．したがって，図中 16ms あたりでの 15 番目のシーケンス実行は，第 14 シーケンスと第 15 シーケンスが一部オーバラップしており，その結果バンド幅は 3GB/s 付近まで低下している．

以上の結果から，A2AT は，各シーケンスの実行に最適なパケット間ギャップを与えることで，高い性能を発揮できることを確認した．一方で，シーケンス完了時のインバランスが次のシーケンスの実行に影響を与えることが明らかになった．この問題は，ノード数が大きくなると顕著に表れると考えられる．

4.5 実験 4：バリア同期によるロードインバランスの抑制

実験 3 では，第 14 シーケンス終了時のロードインバランスによって，次のシーケンスの実行を妨げることがわかった．そこで，本実験では，各シーケンスの終了後にバリア同期を挿入し，インバランスを抑制することで実行性能の向上を図る．

表 5 に評価パラメータを示す．また，各シーケンスのギャップバイアスは実験 3 で取得したものを用いた．バリア同期には，MGEN ライブラリによってソフトウェア実装した dissemination アルゴリズムを使用した．このバリア同期のコストを事前に調査した結果，無負荷状態では 11.5us を要した．すなわち，A2AT(9×9) での 20 シーケンス実行時には，

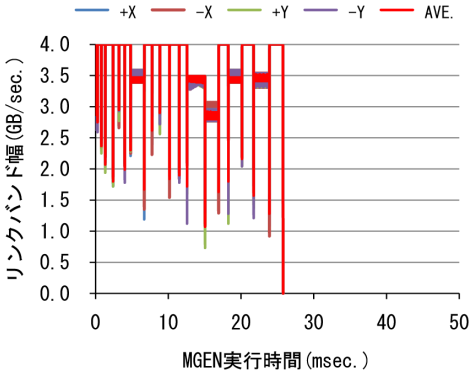

図 17 各シーケンスの最適ギャップ実行の統合
（MGEN 実行時間：25.8ms）

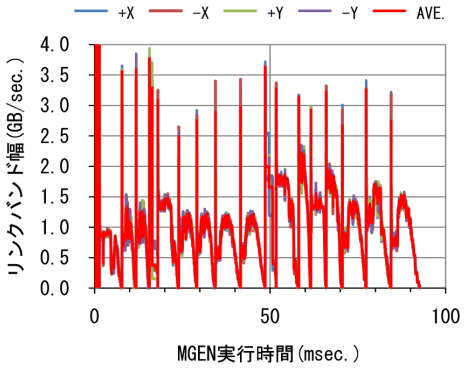

図 18 最適パケットペーシングとバリア同期の統合
（MGEN 実行時間：92.6ms）

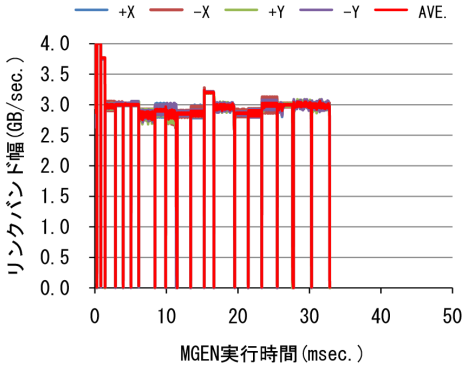

図 19 最適パケットペーシングとバリア同期の統合
（MGEN 実行時間：32.8ms）

表 5 実験 4 の評価パラメータ

パラメータ	設定値
評価対象アプリケーション	a2at
ノード数	81(9x9)
メッセージサイズ	1MiB
パケット間ギャップ値	（ホップ数 −1）＋ パケット間ギャップバイアス
ギャップバイアス	各シーケンスにおける最適値
バリア同期コスト ‡	11.5us(バリア後のインバランス：3us 以下)

‡ dissemination

218.5us のオーバヘッドとなるが全体の実行時間と比較すると無視できる．一方，本バリア同期によるインバランスも発生する．そこで，様々なインバランス環境下でバリア同期を実行し，その後のインバランス量を調査した結果，全て 3us 未満に収まることがわかった．

4.5.1 実験 4：結果と考察

最適なシーケンス実行にバリア同期を併せた実験結果を**図 18** に示す．この実験では，実行時間は実験 3 における 25.8ms から 92.6ms へ大幅に悪くなった．また，図 18 から，バリア同期による効果は見られず，A2AT の実行を阻害していることがわかる．最初の 3 つのシーケンスについては，ホップ数が 1 か 2 であるため，バリア同期によるインバランスの影響を受けていない．一方，残りの 17 個のシーケンスは 3 ホップ以上あり，すべてリンクバンド幅が不安定になっている．すなわち，バリア同期後の 3us のインバランスによって各フェーズの実行が崩れ，性能を十分に発揮できていないと言える．そこで，インバランスへの耐性を高めるために，ロードインバランス環境下における各シーケンスの最適なギャップバイアス値を調査した．具体的には，バリア同期後のインバランス量が 3us の状態を作り，その後ギャップバイアスを測定した．その結果，全シーケンスでのギャップバイアスは概ね増加し，最大値は第 14 シーケンスの 2.5 であった．

この調査結果に基づき，再度，A2AT を実行した結果を**図 19** に示す．この図から，バリア同期がもたらすインバランスの影響を抑制していることがわかる．実行時間，ならびにリンクバンド幅の平均は，実験 3 の場合よりも低下しているが，実行時間は 32.8ms と比較的良い性能を出しているといえる．一方，ノード数が大きくなり，ホップ数が増えることによって，フェーズ実行後のロードインバランスが顕著になると考えられる．したがって，大規模システムを想定した場合，実験 3 のような最適化手法を採ることが難しく，バリア同期によるインバランスの抑制が効果的であると考える．

5. ま と め

本稿では，NSIM と呼ぶインターコネクトシミュレータを用いて，2 次元トーラス網での最適全対全通信アルゴリズムの一つである A2AT の基礎的な性能評価を行った．また，A2AT に対して積極的な（明示的な）パケットペーシングを適用し，その性能評価を行うとともに A2AT の性能の高さ，ならびにペーシングの有効性を示した．そして，A2AT の各フェーズの実行にあわせたパケット間ギャップを設定することで，リンクバンド幅を十分

に活用し，理論実行時間の 107.5%で実行できることを確認した．以上の結果から，A2AT での通信はロードインバランスに敏感であることがわかった．したがって，実システムで利用するには OS ジッタの影響を考慮した評価も必要となる．これについては重要な課題とし，今後は，本稿で述べなかった他の全対全通信や集団通信アルゴリズムなど，様々なアプリケーションについても評価を行う予定である．

一方，問題の性質が同じものとして，1 チップに複数コアを搭載しコア間を接続する NoC(Network-on-Chip) が挙げられる．NoC でのパケット転送方式は，ハードウェアの制約からバッファ容量をフリット分しか持たない Wormhole が多くを占める．特にメモリ転送時に高トラフィックとなるため，通信混雑や輻輳の発生頻度は VCT と比べて高くなり，コア間の転送遅延も小さいためネットワークの飽和が速くなるといえる．また，コア間で同期操作を伴うアプリケーションを実行する際には，本稿で議論したようなロードインバランスの問題が同様に存在すると考えられる．そこで，本稿で述べた性能解析手法やペーシングによる性能向上が期待できる．

なお，本実験を通して，NSIM による効果的なシミュレーション評価ができたと考える．今後，一層の精度向上や機能充実を計るとともに，これからのスーパーコンピュータの研究開発へ貢献するために NSIM の公開を予定している．

謝辞 本研究を進めるにあたりご協力頂いた九州大学 村上和彰教授，青柳睦教授，南里豪志准教授に感謝する．また，本研究の実験結果の一部は，九州大学情報基盤研究開発センターの研究用計算機システムを用いて取得したことを付記する．

参 考 文 献

1) 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).
2) 高上治之，矢崎俊志，安島雄一郎，清水俊幸，石畑宏明：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，2010 年ハイパフォーマンスコンピューティングと計算科学シンポジウム (HPCS2010) 論文集，pp.83–90 (2010).
3) Aggarwal, A., Savage, A. and Anderson, T.: Understanding the Performance of TCP Pacing, *IEEE INFOCOM2000 Conference on Computer Communications*, pp.1157–1165 (2000).
4) 高野了成，工藤知宏，児玉祐悦，松田元彦，石川裕，岡崎史裕：ギャップパケットを用いたソフトウェアによる精密ペーシング方式，情報処理学会論文誌コンピューティングシステム， Vol.47, No.SIG7, pp.194–206 (2006).
5) Kumar, S., Sabharwal, Y., Garg, R. and Heidelberger, P.: Optimization of All-to-All Communication on the Blue Gene/L Supercomputer, *37th Int'l Conf. on Parallel Processing 2008, ICPP'08*, pp.320–329 (2008).
6) 三輪英樹，薄田竜太郎，柴村英智，平尾智也，眞木淳，稲富雄一，井上弘士，安島雄一郎，三吉郁夫，清水俊幸，安藤壽茂：NSIM:将来の大規模相互結合網を対象とした通信シミュレータの開発，情処研報， Vol.2010-HPC-125, No.5, pp.1–9 (2010).
7) 井上弘士，薄田竜太郎，安藤壽茂，石附茂，小松秀実，稲富雄一，本田宏明，山村周史，柴村英智，于雲青，青柳睦，木村康則，村上和彰：大規模システム評価環境 PSI-SIM:数千個のマルチコア・プロセッサを搭載したペタスケールコンピュータの性能予測，情処研報， Vol.2008, No.39, pp.51–56 (2008).
8) Susukita, R., Ando, H., Aoyagi, M., Honda, H., Inadomi, Y., Inoue, K., Ishizuki, S., Kimura, Y., Komatsu, H., Kurokawa, M., Murakami, K.J., Shibamura, H., Yamamura, S. and Yu, Y.: Performance Prediction of Large-scale Parallel System and Application using Macro-level Simulation, *Proc. Intl. Conf. Supercomputing (SC2008)* (2008).
9) 柴村英智，薄田竜太郎，本田宏明，稲富雄一，于雲青，井上弘士，青柳睦：PSI-NSIM:大規模並列システムの性能解析に向けた並列相 互結合網シミュレータ，信学技報 CPSY2007-32， Vol.107, No.276, pp.45–50 (2007).
10) MPICH2: http://www.mcs.anl.gov/research/projects/mpich2/.