情報処理学会論文誌　コンピューティングシステム　Vol. 4　No. 3　47–55 (May 2011)

メッセージフローに基づくネットワークシミュレータ MFS の評価

矢崎 俊志[†1]　石畑 宏明[†2]

本論文では，筆者らが通信アルゴリズムの評価を目的として提案したフローベースシミュレータ Message Flow Simulator（MFS）の，より汎用的な利用可能性を示すため，既存のパケットベースシミュレータ Booksim を用いて様々なネットワークトポロジと通信パターンで MFS を比較評価した結果を述べる．筆者らはこれまで，MFS がパケットベースシミュレータ BigSimulator より短時間で全対全通信アルゴリズムを評価可能であることを示した．また，メッセージが相互結合網を通過する時間のみを評価可能な Booksim を用いて，Fattree ネットワーク上のランダム通信シミュレーションによる比較評価を行ってきた．本論文では新たに MFS と Booksim のシミュレーション結果の差がスイッチで行われるアービトレーションの影響により生じることを示した．このことから，通信の平均ホップ数が少ないトポロジのネットワーク評価や，近距離のノード通信を頻繁に行う並列プログラムの通信シミュレーションに MFS が利用できる可能性を示した．1 万ノード以上の大規模なネットワークについて，全ノードが 10 パケットをランダムな宛先に送るシミュレーションを実行した．このとき，MFS は Booksim の 1〜2%の実行時間とメモリ使用量でシミュレーションを実行した．

An Evaluation of Message-flow-based Network Simulator

Syunji Yazaki[†1] and Hiroaki Ishihata[†2]

This paper describes evaluation results of Message Flow Simulator (MFS) to show capabilities of application of MFS. MFS is a flow-based network simulator for large-scale parallel computer. We previously showed that MFS performed simulation of all-to-all communication algorithms faster than BigSimulator which is a packet-based network simulator. We also compared evaluation results of communication time estimated by MFS and Booksim which is a packet-based network simulator. In the paper, we show that MFS gives different result with Booksim due to effect of the arbitration in the router. From this result, we find that MFS provides better results when many messages are communicated in low hop count or average hop count in networks is low. MFS performs simulation with less run-time and memory usage when the number of nodes is over 10,000. Run-time and memory usage of MFS were from 1% to 2% by those of Booksim.

1. はじめに

1.1 背　景

多数の計算ノードを相互結合網で接続した超並列計算機上で効率の良い並列計算を実現するためには，ノード間通信の効率化が重要な課題である．これは，並列計算における通信時間の増大が並列化効率を低下させる要因となるためである．

ノード間で高い通信効率を実現する相互結合網の実装コストは，一般に，接続するノード数と最大通信性能を決定づけるバイセクションバンド幅に依存して増大する．実際の並列計算機においては，通信効率とコストのトレードオフにより，様々な構成の相互結合網が用いられている．比較的小規模の並列計算機においては，クロスバで相互結合網を構成することができた[1]．数千ノードの接続には，よりコストの低い Fattree トポロジの相互結合網を採用している例がある．数十万ノード規模の並列計算機では，多次元の Mesh や Torus，バンド幅の小さい Fattree，またはこれらを組み合わせた不均一なトポロジを選択せざるをえない[2)–4)]．このようなトポロジで構成された相互結合網は，バイセクションバンド幅が狭く，通信経路の競合が起きやすい．また，通信の偏りによってホットスポットも発生しやすく，通信路をまんべんなく効率的に利用することが難しい．

並列プログラムの開発者は限られた通信路を効率的に使うため，メッセージ送受信のタイミングや順番を工夫した様々な通信アルゴリズムを用いる．通信競合が起きやすいトポロジを持つ近年の大規模並列計算機向けには，特定のトポロジや通信パターンに合わせた最適な通信アルゴリズムが考案されている．これらの通信アルゴリズムの多くは複雑であり，その評価を解析的に行うことが難しい．

従来，大規模ネットワークを対象とした通信アルゴリズムの評価は，長い実行時間を必要とする通信シミュレーションの結果に基づいて行われてきた．並列計算機の通信シミュレー

†1 電気通信大学
The University of Electro-Communications
†2 東京工科大学
Tokyo University of Technology

ションに使われるネットワークシミュレータは，本来，相互結合網の評価や通信時間の精密な予測を目的として開発されたものが多い．そのため，通信をパケットやフリット単位で詳細にモデル化したパケットベースシミュレータを用いるのが主流である．パケットベースシミュレータの実行には通信を行うノード数や通信されるパケット数などに比例した時間がかかる．

より効率良く通信アルゴリズムを評価することを目指して，筆者らは Message Flow Simulator（MFS）を開発した[5),6)]．MFS は，並列プログラムの開発者が考える，抽象度の高い通信モデルを実装したものである．MFS は，通信をパケットやフリットのように粒の動きとしてとらえるのではなく，流体の流れ（フロー）として抽象化し，その競合度合いを算出することで通信アルゴリズムを評価するフローベースシミュレータである．MFS の大きな特徴は，より大規模な通信シミュレーションに対応した高い拡張性と，様々な通信パターンのシミュレーションを短時間で実行できる高速性である．一方で，実機に近いという意味での精度はパケットベースシミュレータと比較して低い．

筆者らは，文献 5) で，MFS が既存のパケットベースシミュレータ BigSimulator[7),8)] より短時間で通信アルゴリズムを評価することができることを示した．この結果は，2 次元 Torus トポロジを持つ相互結合網における全対全通信アルゴリズムを対象としたものであり，その他のトポロジや通信パターンに対する評価は行われていない．また，BigSimulator は，MFS と異なり，メッセージの流れだけでなくノードで行われる通信の初期化や終了に関わる処理もシミュレーション結果に含める．このようなノードでの処理時間を除外するため，BigSimulator による測定においては，通信されるメッセージのサイズを大きくした場合と小さくした場合の差を通信時間とした．しかしこの方法では，ノードでの処理のうち，メッセージサイズに比例して大きくなる処理時間の影響を完全に取り除くことが難しかった．

筆者らは文献 6) において，メッセージが相互結合網を流れる時間のみを評価することが可能なパケットベースシミュレータ Booksim を用いて，Fattree トポロジで構成される相互結合網上でのランダム通信におけるシミュレーション結果の比較を行った．この比較では，Booksim のシミュレーションにおいてパケットを構成するフリット数を大きくすることで，結果が MFS に近くなることを示した．これは，1 パケットあたりのフリット数の増加が，より流体モデルに近い粒度の細かな通信を行った結果に近くなるためであると考える．

1.2 目　　的

本論文では，通信アルゴリズムの評価だけでなく，相互結合網の評価など，MFS のより汎用的な利用法における有用性を示すことを目的として，既存のパケットベースシミュレータと MFS を比較評価し，MFS の利用可能性を議論する．本論文では，メッセージが相互結合網を流れる時間のみを評価対象とするため，文献 5) で用いた BigSimulator ではなく，文献 6) で用いた Booksim を比較に用いる．また，文献 6) では行われなかった，Mesh や Torus トポロジで構成された相互結合網やランダム以外の通信パターンも評価に用いる．

本論文では，2 章で関連研究を引用し，並列計算機向けネットワークシミュレータについて述べる．3 章で Booksim による統計的な相互結合網評価手法をもとに，MFS と Booksim を比較評価する．4 章で MFS と Booksim による大規模ネットワークのシミュレーション結果を比較する．最後に 5 章でまとめる．

2. 並列計算機向け通信シミュレータ

本章では，まず，並列計算機を対象とした通信シミュレータについて，パケットベースおよびフローベース方式についてシミュレーションモデルを説明する．続いて，3 章以降の評価に用いるシミュレータについて述べる．

2.1 シミュレーションモデル

パケットベースおよびフローベースのシミュレーションモデルを**図 1** に示す．パケットベースシミュレーションは，図 1 の左側に示すように，通信を構成するパケットやフリットを粒としてとらえ，この単位でシミュレーションを行う．このモデルは，実際に行われる通信をハードウェア側からの視点で抽象化したものであるといえる．これらの多くは，並列計算機の相互結合網上で発生する輻輳とその影響をハードウェアに近いレベルで精密に調べるために用いられる場合が多い[7)–12)]．並列計算機上で実行される並列プログラムの実行時間予測に用いられた例もある[13),14)]．

フローベースシミュレーションは，図 1 の右側に示すように，通信を連続体の流れ（フ

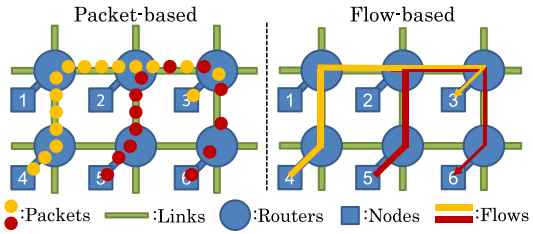

図 1　パケットベースおよびフローベースのシミュレーションモデル
Fig. 1　Simulation models of Packet-based and Flow-based methods.

ロー）として扱い，その流量が通信経路の競合により制限されるというモデルに基づきシミュレーションを行う．このモデルは，通信をソフトウェア側からの視点で抽象化したものであるといえる．この方式は，パケットやフリットの振舞いを個々に再現しないため，シミュレーションを高速に実行することができる．フローベース方式は通信経路競合の度合いを効率良く評価することができるため，通信アルゴリズムの評価などに適している．

2.2 Booksim

パケットベースシミュレータの実装として Booksim がある．Booksim は，Stanford 大学の Dally らによって開発されたシミュレータである[15]．このシミュレータでは，通信はフリット単位で再現される．Booksim は，サイクル単位でシミュレーションが進行するサイクルベースシミュレーションである．通信を詳細に表現するため，通信に関わるスイッチ（ルータ）のバッファや Virtual Channel（VC），アービトレーション機構などもモデル化されている．シミュレーションを実装する際には，相互結合網のトポロジ，サイズ，スイッチのバッファサイズ，VC の数，アービトレーションアルゴリズム，1 パケットを構成するフリット数，通信パターンなどを指定することができる．

Booksim は，フリットが相互結合網を通過する平均的な割合や，平均的なレイテンシを統計的に見積もることができる．これにより，相互結合網の通信性能を評価する．フリットの平均通過率は相互結合網に投入されたフリットのうち，平均的に何割のフリットが相互結合網を通過できたかを表す値であり，これは，最大スループットに対する実効スループットの割合（Average Throughput Rate，ATR）に相当する．

Booksim では，相互結合網に投入されるフリットの多さを Injection Rate として設定することができる．相互結合網の最大通信能力を評価するためには，相互結合網がフリットで飽和した状態で評価を行う必要がある．ただし，通信がデッドロックしている状態は除外する．Booksim による相互結合網の統計的手法による通信性能評価において，シミュレーションの開始直後および終了直前の通信は測定から除外される．開始直後および終了直前は，相互結合網内にフリットがあまりない．この状態での測定はスループットやレイテンシを過度に高く見積もる恐れがある．

2.3 Message Flow Simulator（MFS）

MFS は，並列計算機上で行われる通信を最適化するための通信アルゴリズムを評価する目的で開発された．MFS はフローの重なりによって通信の流量が制限されるモデルに基づいて通信に必要な時間を計算する．計算の手順を次に示す．より詳細なアルゴリズムは文献 5) で述べられている．

(1)　与えられたトポロジと通信パターンで，全ノードペア間のフローと，その経路のリストを作る．
(2)　相互結合網中の全通信路について，各通信路を通過するフローの重なりの数を数える．
(3)　フローの重なりの数から，各通信路を通るフロー 1 つあたりが利用できるバンド幅を算出する．このとき，バンド幅は通信路を共有する全フローで等分される．重なるフローの数が多ければ，1 つのフローが利用できるバンド幅は小さくなる．
(4)　算出した各フローのバンド幅と各メッセージサイズから，各メッセージの通信に必要な時間を求める．
(5)　求めた時間の最小値を次に進むシミュレーション時間とし，その分だけシミュレーションを進める．
(6)　上記の処理を，与えられた通信パターンに含まれる全通信が完了するまで繰り返す．

MFS は手順 (5) で決定される時間単位でシミュレーション内の時間を進める．したがって，通信の重なりが一様である場合，大規模なネットワークであってもシミュレーションは短時間で終了する．一方，通信の重なりが一様でない場合，最も混雑している通信路を通るフローが利用できるバンド幅に合わせてシミュレーションが進行するため実行時間は長くなる．

3. 統計的手法による結果比較

3.1 方　　法

ここでは，2 章で述べた MFS および Booksim で測定した Average Throughput Rate（ATR）に基づいて，両シミュレータの実行結果を比較する．Booksim は統計的な手法に基づいて測定した ATR と平均レイテンシを用いて相互結合網を評価することができる．一方，MFS の通信モデルはフローに基づくものであり，あるノードペアにおいてメッセージの送信時刻と受信時刻は同じである．よって，レイテンシを得ることが難しい．MFS においては，各ノードごとに最大スループットに対する測定した実効スループットの割合を計算し，それら平均したものを評価に用いる．これは，Booksim が見積もる ATR に近い値である．

Booksim による測定では，Injection Rate を 1 とし，フリットを絶え間なく相互結合網に投入することで，相互結合網をフリットで飽和させた状態で ATR の測定を行う．MFS による測定もこの条件に合わせる．MFS による測定では，シミュレーション終了時刻前に 1 つ以上のノードが通信を完了した時刻までを測定時間とした．MFS のシミュレーションモデルでは，通信の送信時刻と受信時刻は同じである．よって，相互結合網はシミュレー

ション開始直後に飽和状態になる．一方，シミュレーション終了時刻の少し前では，通信を完了したノードが徐々に増える．このとき，相互結合網は飽和状態とはいえない．

トポロジには 2 段の Fattree および，2 次元の Mesh と Torus を用いる．それぞれネットワークの大きさを k-ary 2-tree または k-ary 2-cube で表現する．Fattree については，フルバイセクションバンド幅を持つ構成とする．

Fattree トポロジにおけるルーティングは静的に行う．Mesh と Torus トポロジについては，Dimension order による X-Y routing を用いる．Booksim では同じ距離の経路が複数ある場合，どの経路を選ぶかはパケットごとにランダムで決定される．MFS ではこの場合でも静的に定義された決まりに従って経路を選択する．

Booksim において通信経路が競合した場合，アービトレーションはラウンドロビンで行う．MFS は，通信路の物理バンド幅をその通信路を共有する通信の数で公平に分割することで実効バンド幅を求めている．これは相互結合網中の全通信を考慮した公平なアービトレーションにより全通信の公平性が保たれている状態に相当する．

比較にあたり，Booksim にあらかじめ実装されている通信パターンの中から Uniform，Transpose，Tornado を用いた．各通信パターンの詳細は文献 15) で述べられているが，ここでも簡単に説明する．

Uniform はメッセージの宛先ノード番号をランダムに決める通信パターンであり，相互結合網の様々な統計的評価に用いられる．

Transpose は b-bit で表現された受信ノード番号 d の i-bit 目を，同じく b-bit で表現された送信ノード番号 s の i-bit 目から $d_i = s_{i+b/2 \bmod b}$ で求める通信パターンである．ただし，$0 \leq i < b$ である．このパターンは 1 本の対角線を中心として鏡面対称位置のノードをペアとし，通信を行う．よって，対角線から遠いノードほど，ペアとなるノードへの通信距離（ホップ数）が長くなる．この通信パターンは，行列の変換や並べ替えを行う際に現れる．ただし，Booksim の実装においては，この通信パターンは全ノード数が 2 のべきである場合にのみ利用可能である．

Tornado は k-ary n-cube のように，k-ary n-digit で表現されるトポロジにおいて，受信ノード番号 d の p 桁目の値を，送信ノード番号 s の p 桁目の値から $d_p = s_p + (\lceil k/2 \rceil - 1) \bmod k$ で求める通信パターンである．2 次元の Mesh または Torus トポロジ（2-ary n-cube）においては，第 1 次元（x 次元）目は 1 桁目，2 次元目（y 次元）は 2 桁目として表現される．このパターンは，すべてのノードペアが比較的長い距離の（ホップ数の多い）通信を行うため，Mesh や Torus トポロジの相互結合網に高い負荷をかける．

実際の相互結合網の構成には様々な選択がありうる．ここでは次のような構成を用いる．VC の数は通信に必要な最小数とする．Fattree，Mesh トポロジにおいては，VC の数を 1 とする．Torus トポロジについては，デッドロックを回避するため，VC の数を 2 とする．VC ごとのバッファは 2 フリット分の容量を持つものとする．また，1 パケットを構成するフリット数（Flit per Packet，FPP）を変えた場合の変化をみるため，FPP を 10，40，80，100 とした場合についても測定を行う．シミュレーションの実行時間は，両シミュレータともに測定時間が十分に長くなるように設定した．

3.2　Uniform 通信による比較

図 2，**図 3**，**図 4** に Fattree，Mesh，Torus トポロジにおける Uniform 通信のシミュレーション結果を示す．図中のグラフは横軸がネットワークの大きさ k，縦軸が最大スループットに対する実効スループットの割合の平均（ATR）を示している．図中 ATR_{MFS} と ATR_{BS} はそれぞれ MFS と Booksim の ATR を意味する．FPP を 10，40，80，100 とした場合のグラフはそれぞれ FPP10，FPP40，FPP80，FPP100 として示されている．

図 2，図 3，図 4 から，Uniform 通信においては，すべてのトポロジにおいて，MFS と Booksim の結果には差が生じている．この原因について考察する．

まず，Fattree に関する図 2 をみると，図中の MFS と Booksim のグラフは，k の増加に従って緩やかに下がるという同じ傾向を示している．MFS および Booksim の ATR に差がある原因として，アービトレーションの影響が考えられる．

MFS は，相互結合網中の全通信を公平に行うアービトレーションをモデル化している．

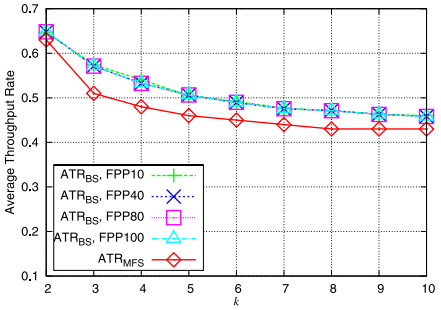

図 2　ATR の比較（Uniform，Fattree）
Fig. 2　Comparison of ATR (Uniform, Fattree).

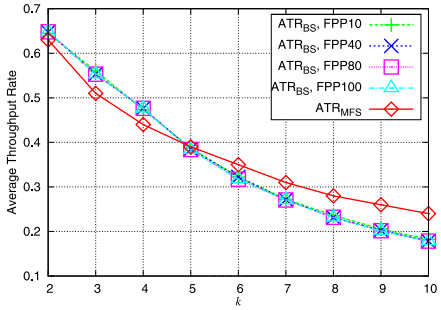

図 3　ATR の比較（Uniform，Mesh）
Fig. 3　Comparison of ATR (Uniform, Mesh).

一方，Booksim では，パケットが経路上のスイッチごとにラウンドロビンによって調停される．通信経路上に多数のスイッチがある場合，その通信はそれだけ多くの調停を受けることになる．調停を受ける回数は通信によって様々である．多くの調停を受けた通信とそうでない通信が，あるスイッチで合流し調停を受ける場合を考える．各通信に対する調停はスイッチ内では公平に行われる．しかし，このとき，各通信がこれまで受けてきた調停は考慮されないため，結果的に各ノードペア間の通信量は公平ではなくなる．Booksim のシミュレーションにおいては，スイッチをホップするたびにこの調停が行われる．そのため，通信ごとのホップ数にばらつきがある場合に，このような差が生じると考える．また，今回の実験ではバッファの容量を 2 フリット分としたが，これも Booksim と MFS の差に影響を与えていると考える．

図 2 と図 3 に示す Fattree と Mesh トポロジの場合については，全体を通して FPP10 から FPP100 のグラフにほとんど差はない．

図 4 に示す Torus トポロジの場合に着目する．$k = 2$ においては，FPP10 と FPP100 の点が FPP40 と FPP80 の点より若干離れた位置にある．また，グラフ全体を見ると FPP10 のグラフはやや他のグラフから離れている．これをふまえて，以降の実験では FPP40 の値を平均に近い代表値として比較に用いる．FPP40 を比較に用いるもう 1 つの理由として評価の効率化があげられる．Booksim によるシミュレーション実行時間はフリット数の増加に従って長くなる．先の実験では FPP40 と FPP80 の結果はほぼ同じである．FPP80 を代表値として用いることも可能であるが，今回はシミュレーション実行時間がより少ない FPP40 を用いた．

3.3　Transpose 通信による比較

図 5，**図 6**，**図 7** に Fattree，Mesh，Torus トポロジにおける Transpose 通信のシミュレーション結果を示す．図 5 と図 6 に示す Fattree および Mesh トポロジに関しては，MFS と Booksim の ATR はほぼ一致している．Uniform 通信と異なり，Transpose 通信では特定のノードペアで規則的な通信を繰り返す．また，Fattree は全ノードペアで平均的にホップ数が少なく，なおかつそのばらつきも少ない．

Mesh トポロジにおいては，対角線を中心として鏡面対称の位置あるノードどうしでペアが作られる．対角線から離れた位置にあるノードの数は，近い位置にあるノードよりも少ない．よって，鏡面対称位置のノードをペアにして通信を行うと，全体の平均ホップ数は小さくなる．この場合，MFS と Booksim の差は近くなる．

図 7 に示す Torus トポロジに関するグラフを見ると，両グラフに若干の差がみられる．Torus トポロジは，Mesh トポロジと同じ理由からノードペアの平均ホップ数は小さい．しかし，3.1 節で述べたように，Booksim は Dimension order の X-Y routing において，同距離の経路が複数あると，その中からランダムで 1 つの経路を選び通信を行う．Torus トポロジにおいて k が偶数である場合は右回りまたは左回り，上まわりまたは下回りの組合せで 4 通りの経路選択がある．Booksim はこの経路選択をパケットごとに行うため，同じノードペアどうしの通信であっても経路がつねに同じとは限らない．一方で MFS はこの経路選択を静的な決まりに従って決定するため，このような場合でもつねに同じ経路で通信

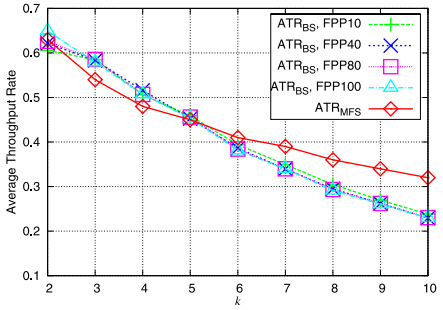

図 4　ATR の比較（Uniform，Torus）
Fig. 4　Comparison of ATR (Uniform, Torus).

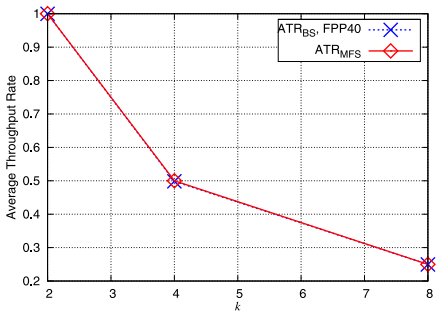

図 5　ATR の比較（Transpose，Fattree）
Fig. 5　Comparison of ATR (Transpose, Fattree).

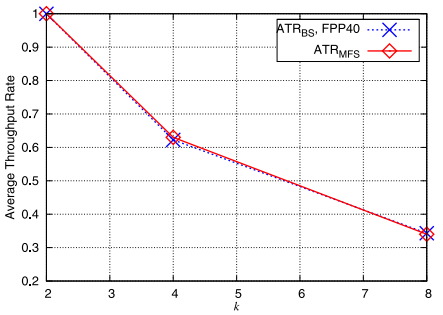

図 6　ATR の比較（Transpose，Mesh）
Fig. 6　Comparison of ATR (Transpose, Mesh).

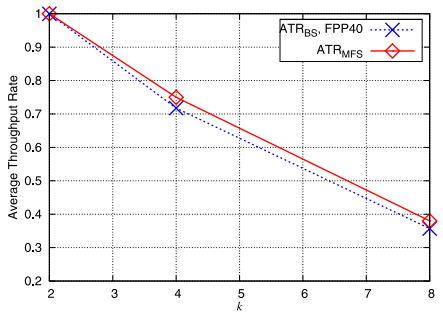

図 7　ATR の比較（Transpose，Torus）
Fig. 7　Comparison of ATR (Transpose, Torus).

を行う．この違いが値の差に影響を与えていると考える．このとき，$k = 4$ における MFS と Booksim の ATR の差は 0.037 であった．よって，この影響は，MFS の ATR に対して 4.93%（= 0.037/0.750）程度であり，小さいものであるといえる．ただし，この差にも VC のバッファによる影響は含まれていると考える．

3.4　Tornado 通信による比較

図 8，**図 9** に Mesh および Torus トポロジにおける Tornado 通信のシミュレーション結果を示す．図中に × で示されたグラフはこれまでと同じように，VC の数を 1 および 2 とした場合の測定結果を示す．＋で示されたグラフは，比較のため Virtual Output Queue（VOQ）方式を再現するように Booksim のパラメータを設定した場合の測定結果を表す．VOQ 方式は，各スイッチの VC をノード数分用意し，宛先ごとに専用の VC を用いる方法である．これにより，相互結合網全体で通信が公平に調停される状態に近くなる．VC のバッファサイズはこれまでと同じように 2 フリット分とした．FPP も同様に 40 とした．

Fattree ネットワーク上の Tornado 通信は k がどのような値でも通信の競合が起きない．シミュレーションでも同様の結果を得たため，今回の比較からは除く．

図 8 に示す VC1（Booksim）のグラフと図 9 に示す VC2（Booksim）のグラフに着目する．Tornado 通信では $k \leq 5$ の場合，通信の競合は起こらない．そのため，MFS と Booksim ともに ATR は 1 となっている．

$k = 6$，7 ではどの経路でも 2 個の通信が重なる．MFS のモデルでは，このときの ATR は 0.5 になる．図 8 中の VC1（Booksim）のグラフは，MFS のグラフとほぼ一致している．

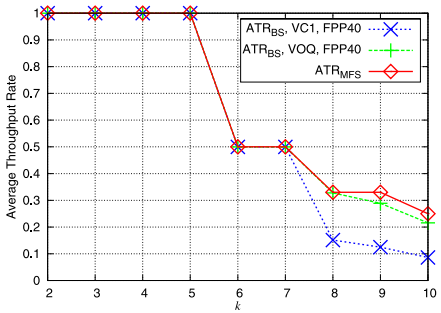

図 8　ATR の比較（Tornado，Mesh）
Fig. 8　Comparison of ATR (Tornado, Mesh).

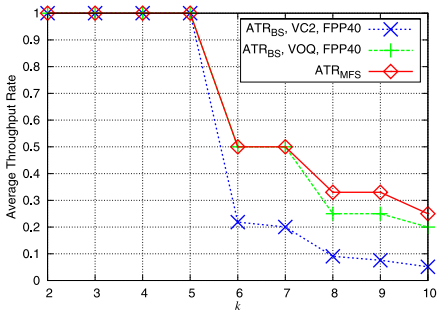

図 9　ATR の比較（Tornado，Torus）
Fig. 9　Comparison of ATR (Tornado, Torus).

一方，図 9 中の VC2（Booksim）のグラフでは，この区間で MFS の ATR に対して約 0.3 の差がある．$k \geq 8$ では，Mesh，Torus トポロジの場合，両方とも MFS と Booksim のグラフに差がある．これらの差の原因として，Uniform 通信の場合と同様にアービトレーションの影響が考えられる．Tornado 通信は，すべてのノードペアが長い距離の通信を行うため，全ノードペアの平均ホップ数が大きい．よって，Mesh や Torus トポロジにおいては通信経路の競合が多く発生し，アービトレーションによる差がさらに拡大すると考える．

図 8，図 9 中に＋で示す VOQ（Booksim）のグラフは，Mesh，Torus トポロジそれぞれに示す VC1（Booksim）および VC2（Booksim）のグラフと比較して，より MFS のグラフに近い．これは VOQ により相互結合網全体で通信が比較的公平に調停されるようになったためである．なお，Mesh トポロジにおいては $k \geq 9$，Torus トポロジにおいては $k \geq 8$ でまだ MFS と VOQ（Booksim）のグラフに差がみられる．この差は，3.3 節で議論した，ルーティングの違いやバッファの影響によるものであると考える．

これまでに示した MFS と Booksim の比較から，MFS の利用範囲について考察する．MFS と Booksim のシミュレーション結果の差は，実験結果よりスイッチで行われるアービトレーションの影響により生じると考える．よって，MFS はアービトレーションがあまり必要ない通信のシミュレーションに利用できると考える．具体的には，通信競合があまり起きない通信シミュレーションや，競合は頻繁に起きるが通信の平均ホップ数が少ない通信シミュレーションが考えられる．利用例としては，Fattree や多次元 Mesh・Torus 系トポロジのように，通信の平均ホップ数が少ないネットワークの評価，または差分法演算などのように近距離のノードと頻繁に通信を行う並列プログラムの通信シミュレーションなどが考えられる．

4.　大規模ネットワークの通信シミュレーション

4.1　方　　法

ここでは，MFS および Booksim で大規模ネットワークの通信シミュレーションを行い，そのシミュレーション結果と実行時間およびメモリ使用量を比較する．

シミュレーションに用いる計算機の CPU は一般のデスクトップ PC で使用されている Intel Core i7 X980，3.33 GHz である．また，メモリの容量は 12 GB である．トポロジには Fattree を用いる．Booksim のシミュレーションにおいては，VC の数は 2，各 VC のバッファサイズを 10 フリット分とした．FPP は 3.2 節での議論をふまえて 40 とする．MFS のシミュレーションにおいては，1 メッセージが Booksim の 1 パケット分になるようにパラ

表 1　k-ary 3-tree における MFS と Booksim のシミュレーション結果比較
Table 1　Comparison of simulation results performed by MFS and Booksim on k-ary 3-tree networks.

		MFS			Booksim			MFS/Booksim		
k	# of nodes	VCT [μs]	Run-time [s]	Memory [MB]	VCT [μs]	Run-time [s]	Memory [MB]	VCT	Run-time	Memory
16	4,096	32.82	2.9	27	35.10	142.4	1300	0.935	0.020	0.021
18	5,832	33.68	4.3	39	34.00	257.2	2300	0.991	0.017	0.017
20	8,000	35.18	6.1	54	31.65	473.2	4200	1.112	0.013	0.013
22	10,648	34.51	8.7	71	34.50	855.0	7200	1.000	0.010	0.010
26	17,576	35.38	21.5	118	-	-	-	-	-	-
30	27,000	34.75	42.0	182	-	-	-	-	-	-
34	46,656	34.84	67.0	263	-	-	-	-	-	-

メータを与えた．通信パターンは各ノードがランダムに決定する宛先に対して 10 個のメッセージを送るものとする．

シミュレーション結果から得られる通信時間の比較においては，Booksim の 1 パケットを MFS の 1 メッセージとし，この大きさを 1 Kbit とおいた．また，MFS，Booksim ともに，すべての通信路の物理バンド幅を 1 Gbps とした．MFS の時間単位 UT_{MFS} はメッセージサイズと物理バンド幅で $UT_{MFS} = 1\,\mu$s（$= 1 \times 10^3/1 \times 10^9 s$）と定義される．Booksim は，1 フリットが 1 つの通信経路を伝わる時間を 1 サイクルと定義しているため，1 サイクルの時間 UT_{BS} は 1 パケットあたりのフリット数を FPP とすると，物理バンド幅から $UT_{BS} = 1/FPP\,[\mu$s] で求めることができる．いま，$FPP = 40$ なので，$UT_{BS} = 0.025\,\mu$s である．

メモリ使用量の比較には Linux のメモリフットプリントを top コマンドにより観測した値を用いる．Booksim はプログラム中で動的なメモリ確保と開放を多数繰り返す実装となっているため，正確なメモリ使用量を測定することは難しい．よって，今回はこの手法により，おおよその値を測定した．

4.2 結　　果

MFS と Booksim のシミュレーション結果を**表 1** にまとめる．表中の VCT は仮想通信時間（Virtual Communication Time）の略であり，シミュレーションにより見積もられた通信時間を示す．Run-time はシミュレーションの実行時間，Memory はメモリ量をそれぞれ示す．表 1 中の MFS/Booksim の列には，MFS と Booksim の値の比をまとめた．

表から，MFS の VCT は Booksim の 0.935〜1.112 倍であり，両シミュレーションの結果は近くなった．また，MFS は Booksim の 0.010〜0.020 倍程度の時間でシミュレーションを完了している．Memory についても，MFS は Booksim の 0.010〜0.021 倍程度である．MFS については，$k = 24$ 以上のより大規模なネットワークについてもシミュレーションを実行したが，いずれの場合もシミュレーション実行時間とメモリ使用量は小さく抑えられている．以上より，シミュレーションの実行速度および使用リソースの点から，MFS のスケーラビリティが高いことが分かる．今回の実験では，数万ノードの通信シミュレーションを，一般のデスクトップ PC 上で実行することができた．

5. おわりに

本論文では，筆者らが提案した Message Flow Simulator（MFS）を評価した結果について述べた．MFS は，通信を流体の流れ（フロー）として抽象化し，その流量に基づいて通信シミュレーションを行うモデルを採用している．これは，通信をソフトウェア側の視点から抽象化したものである．本論文では，通信アルゴリズムの評価だけでなく，相互結合網の評価など，MFS のより汎用的な用途への利用可能性を示すことを目的として，MFS を既存のパケットベースシミュレータの Booksim と比較した．

比較においては，両シミュレータで Uniform，Transpose，Tornado 通信と Fattree，Mesh，Torus トポロジの組合せでシミュレーションを行い，結果から得られた Average Throughput Rate（ATR）をもとに，Booksim と MFS の違いを比較した．比較結果から，MFS と Booksim のシミュレーション結果の差は，複数の通信がスイッチを同時に通過する際に行われるアービトレーションの影響により生じる可能性を示した．このことから，Fattree や多次元 Mesh・Torus 系トポロジのように，通信の平均ホップ数が少ない相互結合網の評価，または差分法演算などのように近距離のノードと頻繁に通信を行う並列プログラムの通信

シミュレーションなどにおいては，MFS が利用可能であるという知見を得た．

すべてのノードが 10 メッセージ（パケット）をランダムな宛先に送るシミュレーションを実行し，その結果を比較した．測定には Intel Core i7 X980（3.33 GHz）と 12 GB のメモリを搭載した計算機を用いた．MFS が見積もった通信時間 VCT が Booksim のものに近いことを確認したうえで，シミュレーション実行時間とシミュレーション中のメモリ使用量を比較した．比較から，MFS は Booksim の 0.010〜0.021 倍程度の実行時間とメモリ使用量でシミュレーションを実行可能であることが分かった．MFS については，1 万ノード以上の大規模なネットワークについてもシミュレーションを実行した．いずれの場合も MFS のシミュレーション実行時間とメモリ使用量は小さく抑えられていた．このことから，MFS のスケーラビリティが高いことを示した．

今後は，Booksim とのシミュレーション結果の差を生む原因となったアービトレーションを再現できるよう，MFS を機能拡張することで，利用範囲をさらに広げることが課題である．また，今回は VOQ を用いない場合のシミュレーションにおいて，VC のバッファを最小限の大きさとした．バッファサイズを変化させたときの影響についても，さらなる検討が必要であると考える．

謝辞　本研究を進めるにあたりご協力いただいた富士通株式会社次世代テクニカルコンピューティング開発本部の追永勇次氏，清水俊幸氏に深謝します．本研究は，九州大学情報基盤研究センターの研究用計算機システム，電気通信大学情報基盤センター教育用計算システムを利用して行われました．本研究の一部は科研費（22500052）の助成を受けたものです．

参　考　文　献

1) http://www.jamstec.go.jp/esc/index.en.html
2) Hoisie, A., Johnson, G., Kerbyson, D.J., Lang, M. and Pakin, S.: A Performance Comparison Through Benchmarking and Modeling of Three Leading Supercomputers: Blue Gene/L, Red Storm, and Purple, *Proc. SC '06*, p.3 (2006).
3) Alam, S.R., Kuehn, J.A., Barrett, R.F., Larkin, J.M., Fahey, M.R., Sankaran, R. and Worley, P.H.: Cray XT4: an early evaluation for petascale scientific simulation, *Proc. SC '07*, New York, NY, USA, pp.1–12, ACM (2007).
4) Barker, K.J., Davis, K., Hoisie, A., Kerbyson, D.J., Lang, M., Pakin, S. and Sancho, J.C.: Entering the petaflop era: the architecture and performance of Roadrunner, *Proc. SC '08*, Piscataway, NJ, USA, pp.1–11, IEEE Press (2008).
5) 矢崎俊志，石畑宏明：通信アルゴリズム評価用メッセージフローシミュレータの開発，情報処理学会論文誌　コンピューティングシステム（ACS），Vol.3, No.2, pp.88–98 (2010).
6) Yazaki, S. and Ishihata, H.: Message Flow Simulator for Evaluating Communication Algorithms, *Proc. PDCN '10*, pp.291–298 (2010).
7) http://charm.cs.uiuc.edu/research/bignetsim/
8) Choudhury, N., Mehta, Y., Wilmarth, T.L., Bohm, E.J. and Kalé, L.V.: Scaling an optimistic parallel simulation of large-scale interconnection networks, *Proc. WSC '05*, pp.591–600, Winter Simulation Conference (2005).
9) Ang, B.S., Chiou, D., Rudolph, L. and Arvind: Micro-architectures of high performance, multi-user system areanetwork interface cards, *Proc. IPDPS 2000*, pp.13–20 (2000).
10) 若林正樹，天野英晴：並列計算機シミュレータの構築支援環境，電子情報通信学会論文誌 D-I，Vol.J84-D-I, pp.247–256 (2001).
11) Boku, T., Harada, T., Sone, T., Nakamura, H. and Nakazawa, K.: INSPIRE: A generalpurpose network simulator generating system for massively parallel processors, *Proc. PER-MEAN95*, pp.24–33 (1999).
12) Wilmarth, T.L., Zheng, G., Bohm, E.J., Mehta, Y., Choudhury, N., Jagadishprasad, P. and Kale, L.V.: Performance Prediction Using Simulation of Large-Scale Interconnection Networks in POSE, *Proc. PADS '05*, Washington, DC, USA, pp.109–118, IEEE Computer Society (2005).
13) 久保田和人，板倉憲一，佐藤三久，朴　泰裕：大規模データ並列プログラムの性能予測手法と NPB 2.3 の性能評価，情報処理学会論文誌，Vol.40, pp.2293–2303 (1999).
14) Susukita, R., Ando, H., Aoyagi, M., Honda, H., Inadomi, Y., Inoue, K., Ishizuki, S., Kimura, Y., Komatsu, H., Kurokawa, M., Murakami, K., Shibamura, H., Yamamura, S. and Yu, Y.: Performance Prediction of Large-scale Parallel System and Application using Macro-level Simulation, *Proc. SC '08* (2008).
15) Dally, W.J. and Towles, B.: *Principles and Practices of Interconnection Networks*, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (2003).

(平成 22 年 10 月 1 日受付)
(平成 23 年 1 月 13 日採録)

矢崎　俊志（正会員）

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）．パルテノン研究会，電子情報通信学会，IEEE 各会員．

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．画像処理システム，並列コンピュータの開発に従事．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，1993 年電子情報通信学会論文賞，博士（工学）．電子情報通信学会，IEEE 各会員．