Vol.2014-ARC-213 No.31
Vol.2014-HPC-147 No.31
2014/12/10

RDMA 評価のための大規模インターコネクトシミュレータ「NSIM-ACE」

薄田竜太郎[†1]　森江善之[†1]　南里豪志[†1]　柴村英智[†2]

我々は大規模インターコネクトにおける RDMA の性能評価を可能とするネットワークシミュレータ「NSIM-ACE」を開発した．本講演では NSIM-ACE のシミュレーションモデル，実装技術及び preliminary なシミュレーション精度の評価結果に関して報告する．

NSIM-ACE: A Simulator for Evaluating RDMA on Large-Scale Interconnection Networks

RYUTARO SUSUKITA[†1]　YOSHIYUKI MORIE[†1]　TAKESHI NANRI[†1]
HIDETOMO SHIBAMURA[†2]

We developed NSIM-ACE, a simulator enabling us to evaluate RDMA on large-scale interconnection networks. In this paper, we report simulation models and implementation techniques used in NSIM-ACE. Also we report preliminary evaluation results on the simulation accuracy.

1. はじめに

現在最大級の並列コンピュータシステムはインターコネクトで接続された数 10 万のノードから構成されている．このようなシステム上で稼働するアプリケーションはノード間でデータを通信しながら並列動作するため，高性能なインターコネクト及びノード間通信を行う通信ライブラリを設計し，また実機における内部動作を解析することはアプリケーションを効率よく動作させる上で極めて重要である．しかし，数 10 万ノードを接続するインターコネクトや通信ライブラリの性能をシステム設計段階で予測することや実機における内部動作を解析することは容易ではない．特に多数のノード間で同時に大量のデータを通信し，インターコネクト上で頻繁に通信衝突が発生するような通信パターンでは，衝突によって通信性能が理論値よりも大幅に低下する．そのためノード間の最小レイテンシやバンド幅などをパラメータとする数式モデルを構築して性能予測や内部動作を解析することは困難である．このような問題を解決する試みとして，従来より大規模インターコネクトを対象とするシミュレータが開発されてきた．

NSIM[1]は超大規模インターコネクトの性能評価を行うために開発された汎用シミュレータであり，対象とするインターコネクトを詳細に設定できる．NSIM はシミュレーション速度を重視し，精度を犠牲にしない範囲で簡略化されたシミュレーションモデルを採用している．他のシミュレータにない特長として NSIM は入力として Message Passing Interface (MPI)と互換性のあるプログラムを受け付け，そのプログラムを疑似的に実行しながらインターコネクトをシミュレートする．従ってシミュレータユーザに利用しやすい形で 1 対 1 通信や集団通信をシミュレーションする環境を提供する．NSIM は分散メモリ上の並列離散事象シミュレーションとして実装されており，高並列実行が可能である．

Adiga[2]らは BlueGene/L の 3 次元トーラスインターコネクトの性能を予測するために専用のシミュレータを開発した．彼らは IBM 製のトレーサを拡張し，疑似的なアプリケーションコードからトレースを生成してそれをシミュレータの入力としている．このシミュレータは共有メモリ上の並列プログラムとして実装されている．最大 64K ノードのインターコネクトのシミュレーションが報告されている．

BigNetSim[3]は汎用的なインターコネクトシミュレータである．トポロジー，ネットワークサイズ，レイテンシを柔軟に設定できる．BigNetSim は 2 種類の実行モードをもつ．1 つは内部で人工的に生成した通信パターンをシミュレートするモードであり，他の 1 つは BigSim[4]という超大規模並列システムのシミュレータによって生成されたトレースを入力としてシミュレーションを行うモードである．BigSim は Charm++[5]アプリケーションを簡易的なインターコネクトモデルに基づいてシミュレートし，トレースを出力する．BigNetSim はそのトレースを入力として，詳細なインターコネクトシミュレーションを行う．BigNetSim は分散メモリ上の並列離散事象シミュレータとして実装されている．

FSIN は INSEE[6]と呼ばれるシミュレーションフレームワークに含まれるインターコネクトシミュレータである．

†1 九州大学
Kyushu University
†2 (公財)九州先端科学技術研究所
Institute of Systems, Information Technologies and Nanotechnologies (ISIT)

Vol.2014-ARC-213 No.31
Vol.2014-HPC-147 No.31
2014/12/10

FSIN は多様なルータモデルとトポロジーをサポートしているが，ネットワークイベントが同じ時間で処理されるという比較的単純なインターコネクトモデルを採用している．FSIN は入力として人工的な通信パターンおよび MPI アプリケーションから生成されたトレースを受け付ける．他のシミュレータに見られない特長としてシミュレーション結果を元の入力トレースにフィードバックすることができる．SICOSYS[7]もまた INSEE に基づくインターコネクトシミュレータであるが，さらに詳細なインターコネクトモデルを採用している．ただし，シミュレーション可能なインターコネクト規模は 100 ノード程度に限られる．これらのシミュレータは並列化されていない．

一方，最近の大規模インターコネクトに使用されている NIC は送信ノードのメモリから受信ノードのメモリへプロセッサを介さずに直接データを転送する Remote Direct Memory Access (RDMA)機能を備えている．RDMA には
(1) 送信ノードのメモリから受信ノードのメモリへ直接データを転送するため，通信レイテンシを短縮できる．
(2) プロセッサを介さないため，効率よくアプリケーションにおける通信以外の演算処理と並列に通信できる．複数の NIC が通信を並列に行う場合もプロセッサが逐次的に処理することがない．
(3) 通信の途中でバッファを必要としないために最小限のメモリ使用量ですむ．
などの利点がある．MPI ではアプリケーションから直接 RDMA を操作するプログラミングモデルが主流ではないが，RDMA には上記のような利点があるために将来はアプリケーションレベルにおいても直接 RDMA を操作するようになり，RDMA の重要性が増す可能性もある．RDMA を意識したプログラミングモデルを採用している通信ライブラリとしては ARMCI[8], GASNet[9]などがあげられる．これらの通信ライブラリは RDMA と直接結びついた put/get 型のインターフェースを備えている．MPI もまたメッセージパッシング以外に put/get 型の通信が可能である．最近の例では ACP ライブラリの基本層が RDMA を直接操作する機能をサポートしている[10][11][12]．ACP ライブラリは，エクサスケール時代に向けた低遅延，省メモリな低レベル通信ライブラリである．

RDMA では通信ライブラリが送信ノードと受信ノードの双方が事前に準備を必要とする通信と異なり，一方のノードのみで通信を開始できる．例えば RDMA による書き込みでは，通信ライブラリから送信ノードの NIC を操作するだけで通信を開始することが可能である．逆に受信ノードにおいて事前に RDMA によって書きこまれるメモリの範囲を OS に登録する必要があったり，通信ライブラリが RDMA による書き込みが行われたことを直接検出しない可能性もあり，何らかの追加的な処理が必要になる場合がある．したがって RDMA はインターコネクトレベルにおける動作のみならず，通信ライブラリレベルにおける動作も他の通信とは異なってくる．

従来の大規模インターコネクトシミュレータはインターコネクトや通信ライブラリの設計を行うシミュレータユーザに使いやすい形で RDMA を直接シミュレーションする機能をもっていない．送信ノードと受信ノードの双方が事前に通信の準備をすること前提にしたシミュレータで RDMA の動作をシミュレートしようとすると，リモートノードにおいて RDMA では本来必要のない動作がシミュレーションに含まれたり，ユーザがリモート側の動作を指定しなければならいないなど不都合が生じる．

そこで我々は前述の NSIM を拡張しユーザに使いやすい形で RDMA をサポートした大規模インターコネクトシミュレータ NSIM-ACE を実装した．本論文はこの NSIM-ACE について論述する．

以下本論文は次のように構成されている．2 節では NSIM について，シミュレーションの入出力，シミュレーションモデル，実装の概略を説明する．3 節で NSIM-ACE の拡張された RDMA のシミュレーション機能及びその実装について述べる．4 節では実機における RDMA の通信性能と NSIM-ACE によるシミュレーション結果を比較し，シミュレーション精度を評価する．5 節で本論文のまとめを行う．

2. NSIM の概略

前述のように NSIM は MPI と互換性をもったプログラムを入力として受け付け，ユーザが使いやすい高性能な大規模インターコネクトシミュレータであるが，直接 RDMA をサポートしていない．そこで我々は NSIM を拡張し，RDMA のシミュレーション機能をもった NSIM-ACE を実装した．本節では NSIM の概略を説明する．

NSIM は現在 6 次元までのメッシュ/トーラスネットワーク及びファットツリーネットワークをサポートしている．また特別な 6 次元のメッシュ/トーラスネットワークとして，Tofu ネットワーク[13]をサポートしている．ネットワークで接続される各ノードは 1 つのプロセッサと 1 つないし複数の Network Interface Card (NIC)から構成されているとする．ネットワークはルータ及び，ルータ―ルータ間または NIC―ルータ間のリンクから構成されるとモデル化される．ルータモデルとして，静的次元順ルーティング，バーチャルカットスルー，パイプラインルータを仮定している．リンク上のデータ転送は基本的にパケット単位でシミュレートされるが，フリットレベルでシミュレーションした場合と同等の精度を持つよう工夫している．メモリ使用量を抑制するため，データ転送においてはデータ量の情報のみ転送され，実際のデータの内容は転送されない．

NSIM の入力は MPI プログラムにおいて MPI 関数の接頭辞 MPI_を MGEN_に置き換えたものである．これを MGEN プログラムと呼ぶ．MPI 関数以外の計算部分に関しては実

際に計算を行うコードの代わりにMGEN_Comp(t) (tは計算時間の予測値)という形で表現される．これにより，プロセス間の通信開始時刻のずれや計算時間を考慮したシミュレーションを行うことができる．NSIMはこのMGENプログラムをシミュレートし，シミュレーション結果として，予測される全実行時間，リンクの実効バンド幅，リンクの使用効率やその他詳細な統計データを出力する．

NSIMは次の5つのモジュールから構成されている．

(1) MGEN (Message level event GENeration)
MGENプログラムを疑似的に実行してMessage Level Event (MLE)を生成する．MLEはMPIにおけるメッセージに対応する．

(2) PGEN (Packet level event GENeration)
MGENを呼び出し，生成されたMLEからPacket Level Event (PLE)を生成する．PLEはパケット転送を処理する離散事象シミュレーションのイベントである．

(3) SIM (SIMulation control)
PGENを呼び出し，生成されたPLEをネットワークシミュレーションを行う離散事象シミュレータのイベントキューに投入し，離散事象シミュレーションの時刻を進める．また受信NICに到達したパケットの終了処理を行う．

(4) DES (Discrete Event Simulation)
離散事象シミュレータ．

(5) EP (Event Processing)
パケット転送を処理する．PLEを処理する離散事象シミュレータのイベント処理ルーチン．

シミュレーションはSIMがPGENを呼び出すところから開始される．PGENは内部でMGENを呼び出し，PGENは生成されたMLEから，パケットを送信NICからネットワークに送出する処理に対応するPLEを生成する．SIMはそのPLEをDESのイベントキューに投入する．SIMはイベントの処理順序が時間的に逆転しないよう適切な時間ステップで離散事象シミュレーションの時刻を進める．ネットワークに投入されたパケットは離散事象シミュレーションによりEPでイベント処理されながらネットワーク上を転送され，受信NICに到達する．SIMはそのパケットを受信すれば新しくパケットの送信を開始するプロセスを検索する．そのようなプロセスが見つかれば新しいパケットに対応するPLEをDESのイベントキューへ投入するなどの処理を行う．

3. NSIM-ACE

本節ではNSIM-ACEのシミュレーションモデル，入出力，及び実装について，NSIMから拡張された部分を中心に説明する．

3.1 シミュレーションモデル

NSIMでは通信に関与する両方のプロセスにおいてかならずプロセッサが関与するが，RDMAでは一方のプロセッサしか関与しない．NSIM-ACEではput型とget型の2種類のRDMAについて以下のようにモデル化した．

(1) Put
Putは送信プロセスが，送信ノードのメモリ上におかれたデータを受信ノードのメモリ上へ転送するRDMAである．送信ノードのメモリ上におかれたデータはDMAにより送信NICへ転送される．送信NICはデータをパケットに分解し，ネットワークへ送出する．パケットはネットワーク上を転送されて受信NICへ到達する．ネットワーク上のパケット転送についてはNSIMと同じようにモデル化されている．受信NICはパケットからデータを再構成して，DMAにより受信ノードのメモリ上へ転送する．受信NICはその後コントロールパケットを送信ノードへ向け返送する．コントロールパケットが送信ノードのNICへ到達するとputが完了する．

(2) Get
Getは受信プロセスが，送信ノードのメモリ上におかれたデータを受信ノードのメモリ上へ転送するRDMAである．受信ノードのNICはコントロールパケットを送信ノードへ向け送出する．送信ノードにコントロールパケットが到達すると，コントロールパケットに格納された情報に応じて送信ノードのメモリ上からデータが送信NICへDMAにより転送され，パケットに分解されてネットワークへ送出される．ネットワーク上のパケット転送についてはやはりNSIMと同じようにモデル化されている．パケットが受信ノードのNICへ到達すると，NICはパケットからデータを再構成し，DMAにより受信ノードのメモリ上へ転送する．これでgetが完了する．

3.2 シミュレータの入出力

MGENプログラムにRDMAを記述するため，前述のACPライブラリ基本層においてRDMAを行う関数と類似の機能を持つ2つの関数を用意した．

(1) MGEN_acp_handle MGEN_acp_copy(int src_rank, int dest_rank, int data_size);
MGEN_acp_copyはランクsrc_rankのプロセスからランクdest_rankのプロセスへdata_sizeバイトのデータをRDMAにより転送する．Src_rankが本関数を発行したプロセスのランクである場合，putとして動作する．Dest_rankが本関数を発行したプロセスのランクである場合，getとして動作する．本関数はRDMAが完了しなくても終了し，起動したRDMAに対応するハンドルを返す．

(2) void MGEN_acp_complete(MGEN_acp_handle handle);
MGEN_acp_completeはhandleに対応するRDMAが完了するまで待つ関数である．

これらの関数名はACPライブラリ基本層における関数名に接頭辞MGEN_を付加している．この他，putの受信ノードにおいてデータがメモリ上へ転送されるのを待つポーリングを行うために LMGEN_で始まるいくつかの関数を追

加した．

RDMA を行う MGEN プログラムのサンプルコードを以下に示す．このサンプルでは 2 つの隣接プロセスのうち，一方からデータを get し，他方にデータを put する．Put におけるポーリングを行うために，MGEN_Recv の tag に put により転送されたデータであることを示す特別な tag ACP_PUT_TAG を指定している．MGEN_Recv はデータがメモリ上に転送されるまで待つ．LMGEN_Isend_acp_put_ctrl は put のコントロールパケットを MPI_Isend のように送信する関数，LMGEN_Wait_ctrl はその送信完了を待つ関数である．

```
#include “mgen.h”
#include <string.h>
int MGEN_Main(int argc, char **argv){
  int rank, size, ms=4, tag = ACP_PUT_TAG, r, l;
  MGEN_Request rq;
  MGEN_Status st
  MGEN_Datatype dt = MGEN_BYTE;
  MGEN_Comm com = MGEN_COMM_WORLD;
  t_NsimRoutingInfo info;
  MGEN_acp_handle handle;
  bzero(&info, sizeof(t_NsimRoutingInfo));
  MGEN_Comm_rank(com, &rank);
  MGEN_Comm_size(com, &rank);

  r = (size + 1) % size;
  l = (rank – 1 + size) % size;

  handle = MGEN_acp_copy(l, rank, ms);
  MGEN_acp_complete(handle);

  handle = MGEN_acp_copy(rank, r, ms);
  MGEN_Recv(NULL, ms, dt, l, tag, &st);
  LMGEN_Isend_acp_put_ctrl(l, com, &rq, 0, &info);
  MGEN_acp_complete(handle);
  LMGEN_Wait_ctrl(&rq, &st);
}
```

NSIM-ACE はこのような MGEN プログラムをシミュレートし，シミュレーション結果として，NSIM と同じく，予測される全実行時間やシミュレーションに関する豊富な統計情報を出力する．

3.3 実装

NSIM-ACE を実装するうえで NSIM と大きく異なる点は通信順序がシミュレーションの結果により確定することである．NSIM の場合は MGEN プログラムにより，発行される通信の順序がすべて確定している．しかし，NSIM-ACE において例えば 2 つのプロセスがほぼ同時に同じノードのメモリを get する場合を考えると，2 つの get のコントロールパケットのうちどちらが先にそのノードに到達し，get のデータ転送が行われるかはシミュレーション結果によって決定される．NSIM では MGEN や PGEN が MLE や PLE を生成するとき，一定数をまとめて生成して時刻順にキューに格納しておくが，get のデータ転送に対応する MLE や PLE はノードに get のコントロールパケットが到達するまで生成できず，NSIM の実装をそのまま使用することができない．そのため get のコントロールパケットが到達した時点でキューの途中に get に対応する MLE や PLE を挿入する実装に変更している．

通信順序がシミュレーション結果により確定することで，離散事象シミュレーションにおける時刻の進め方に関しても検討する必要がある．NSIM では SIM が PLE を DES のイベントキューに投入するという動作と離散事象シミュレーションの時刻を進め，EP モジュールがイベントキューのイベント処理を行うという動作を交互に繰り返す．NSIM では現在の時刻が t であるとき，時刻 $t+\Delta t$ までに受信ノードに到達するパケットの到達時刻が確定する．ここで Δt は 1 ホップ離れたノードからパケットが受信ノードに到達するために必要な時間である．パケットの到達時刻が確定すると，そのパケットを受信すれば新しくパケットの送信を開始するプロセスがある場合，SIM は新しいパケットに対応する PLE をイベントキューへ投入する．時刻 t で到達時刻が確定していないパケットは将来時刻 $t+\Delta t$ 以降に受信ノードに到達する．そのパケットを受信すればイベントキューに投入される PLE がある場合，そのイベント時刻はパケットの到達時刻よりも遅いため，イベント時刻の下限は $t+\Delta t$ である．したがって，$t+\Delta t$ まで時刻を進めても将来時刻 $t+\Delta t$ 以前のイベント時刻をもつ PLE がイベントキューに投入され，イベントの処理順序が時間的に逆転することはない． NSIM-ACE において get のコントロールパケットを受信すれば get のデータ転送を開始する状況は，上述のパケットを受信すれば開始される送信に対応する．したがって同様に $t+\Delta t$ まで時刻を進めてもイベントの処理順序が時間的に逆転することはない．

4. シミュレーション精度に関する評価実験

本節ではシミュレーション精度を評価するためにシミュレーションと実機上の測定結果を比較した実験について述べる．

シミュレーション精度を評価するため，ランダムリングトラフィックのバンド幅を実機における測定結果と比較した．ランダムリングトラフィックは HPC Challenge ベンチマークスイート[14]の 1 つである．ベンチマークプログラムのプロセスはランダムな順序でリングを構成する．各プロセスは 2MB のデータを左右の隣接プロセスへ並列に送

信し，左右の隣接プロセスから並列に受信する．そしてバンド幅を測定する．元のプログラムは MPI のメッセージ通信で書かれているが，我々は ACP ライブラリ基本層を用いて put または get による通信に書き直した．現在，ACP ライブラリ基本層は udp，Tofu ネットワーク，InfiniBand 上で動作する．本実験で利用したのは InfiniBand 上の実装である．InfiniBand 上の実装では起動時にプロセスあたり通信スレッドが 1 つ生成され，実行中状態を監視しながら動作しているため，コア数と同じ数のプロセスでベンチマークプログラムを実行すると通信スレッドの動作がバンド幅に影響を及ぼす．その影響を排除するため，半分のノードあたり 2 プロセスでプログラムを実行した．また元のプログラムと異なり，1 通りのプロセス順序でバンド幅を測定した．

実機には富士通 PRIMERGY RX200 S7 を用いた．ノード数は 16 で，各ノードに 4 コア Intel Xeon プロセッサ E5-2609 (2.40 GHz) が搭載されている．ノード間は InfiniBand QDR スイッチで接続されている．スイッチのスループットは片方向 4.0 GB/s，スイッチ内部におけるポート間レイテンシは 140 ナノ秒以下，ルーティングは destination based routing である．シミュレーションの設定パラメータを表 1 に記す．ノードの DMA 転送速度は実機で 1 ノード 2 プロセスのランダムリングトラフィックを実行し，その測定値から求めた．メモリバンド幅はネットワークのバンド幅よりもはるかに大きく，シミュレーション結果にほとんど影響しないため，∞に設定した．また通信ライブラリのレイテンシは 2 MB のデータの 1 ホップレイテンシに比べて非常に小さく結果にほとんど影響しないため 0 に設定した．それ以外のパラメータは実機の仕様に基づいて値を決定した．

表 1　NSIM-ACE の設定パラメータ

種別	パラメータ	値
ルータ	ネットワークの最大理論通信速度	4.0 GB/s
	スイッチスループット	4.0 GB/s
	ルーティング計算時間	4.0 ns
	仮想チャネル割り当て時間	4.0 ns
	スイッチ割り当て時間	4.0 ns
	スイッチレイテンシ	128 ns
	ケーブルレイテンシ	0.6 ns
ノード	DMA 転送速度	2.8 GB/s
	メモリバンド幅	∞
	通信ライブラリのオーバーヘッド	0 ns
	プロセス数	1 プロセス/ノード

シミュレーションと実機上の測定にはいくつか異なる点が存在する．1 つはプログラムである．MGEN プログラムは実機のプログラムから通信に関する部分を抽出して作成したが，NSIM-ACE は NSIM と同じくノードあたり 1 プロセスを仮定しているため，ノードあたり 2 プロセスを実行するかわりに 1 つのプロセスが 2 プロセス分の通信を並列に実行するように記述した．また，本シミュレータはノード内通信をシミュレートしないため，かわりにノードの DMA 転送速度で 2MB のデータを転送するのに必要な時刻だけ，MGEN_Comp でシミュレーション時刻を進めるように記述した．本実験で用いたリングのプロセス順序ではノード内通信とノード間通信の比率は図 1 の通りである．その他，調停やルーティングのアルゴリズムなどが異なっており，後述する誤差の原因になっていると考えられる．

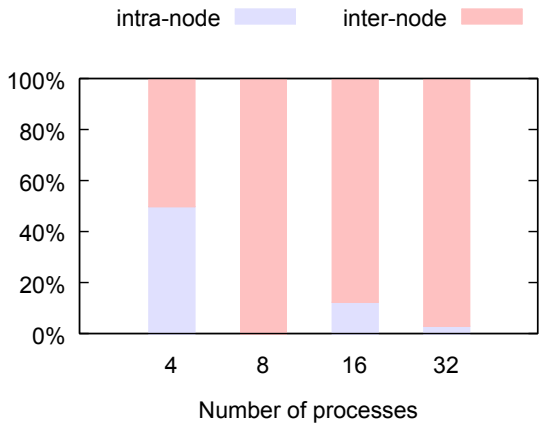

図 1　ノード内通信とノード間通信の比率

MGEN プログラムは実機のプログラムと同じプロセス順序のリングで実行し，バンド幅を求めた．

図 2，図 3 にシミュレーションと実機を比較した結果を示す．Put，get のいずれの場合も 8 プロセス以下ではシミュレーションと実機がほぼ一致し，put や get による RDMA を精度よくシミュレートしていることがわかる．しかし，16 プロセス以上では実機と比較してシミュレーションにおけるバンド幅が低下している．理由としてルーティング及び調停のアルゴリズムの違いが考えられる．本実験では 8 プロセス以下の場合は，プロセスが割り当てられたすべてのノードが 1 つのスイッチで接続されているためにルーティングや調停の影響がバンド幅に現われにくい．16 プロセス以上の場合はノードが複数のスイッチで接続され，ノード間を異なるスイッチを経由して通信するルートが存在する．そのため，ルーティングや調停のアルゴリズムの違いにより，シミュレーションではより輻輳が起こりやすく，バンド幅が低下している可能性がある．

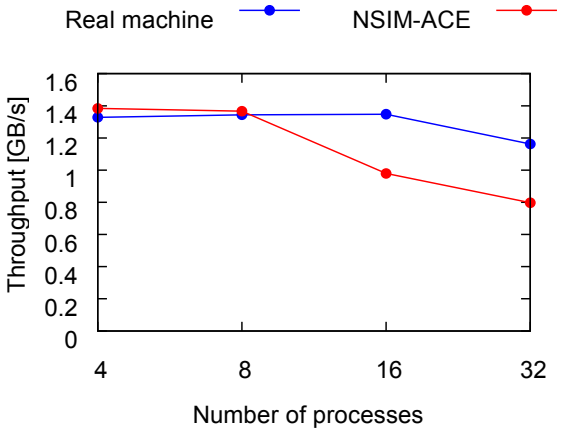

図 2　Put によるランダムリングのバンド幅

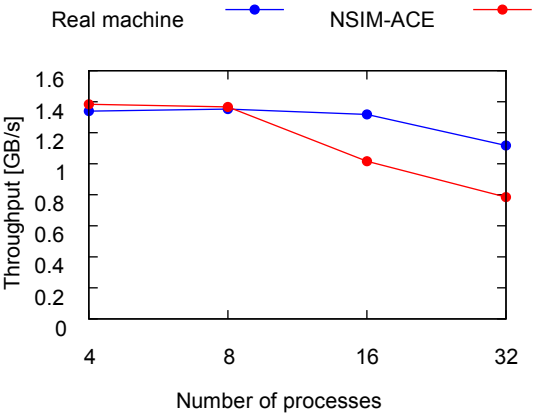

図 3　Get によるランダムリングのバンド幅

5. まとめ

本論文では大規模インターコネクト上で RDMA を評価するためのシミュレータ NSIM-ACE について述べた．NSIM-ACE は NSIM の機能を拡張して入力プログラムで RDMA 関数が記述可能であり，NSIM と同様，ユーザにわかりやすい形で RDMA による通信パターンを記述することができる．また離散事象シミュレーション部分は変更していないため，高並列で動作させることができる．Preliminary な評価実験の結果は NSIM-ACE によってインターコネクト上の RDMA をシミュレーションできることを示している．

謝辞　本研究は，科学技術振興機構 戦略的創造研究推進事業 (CREST)「ポストペタスケール高性能計算に資するシステムソフトウェア技術の創出」研究領域，「省メモリ 技術と動的最適化技術によるスケーラブル通信ライブラリの開発」の一部として実施された．

参考文献

1) Miwa, H. et al.: NSIM: An Interconnection Network Simulator for Extreme-Scale Parallel Computers, IEICE Trans. Inf. & Syst., Vol.94, No.12, pp.2298-2308, (2011).
2) Adiga, N. R. et al.: Blue Gene/L Torus Interconnection Network, IBM J. Res. Dev., Vol.49, pp.265-276, (2005).
3) Choudhury, N., Mehta, Y., Wilmarth, T. L., Bohm, E. J. and Kale, L.V.: Scaling an Optimistic Parallel Simulation of Large-scale Interconnection Networks, Proc. 37th Conference on Winter simulation, Conference, WSC '05, pp.591-600, (2005).
4) Zheng, G., Kakulapati, G. and Kalé, L.V.: BigSim: A Parallel Simulator for Performance Prediction of Extremely Large Parallel Machines, Parallel and Distributed Processing Symposium, International, Vol.1, p.78b, (2004).
5) Kale, L.V. and Krishnan, S.: Charm++: A Portable Concurrent Object Oriented System Based on C++, Proc. Eighth Annual Conference on Object-Oriented Programming Systems, Languages, and Applications, OOPSLA '93, pp.91-108, (1993).
6) Ridruejo, F. J. and Alonso, J. M.: INSEE: An Interconnection Network Simulation and Evaluation Environment, Proc. 11th Euro-Par Parallel Processing Conference 2005, Euro-Par'05, pp.1014-1023, (2005).
7) Puente, V., Gregorio, J. A. and Beivide, R.: SICOSYS: An Integrated Framework for Studying Interconnection Network Performance in Multiprocessor Systems, Proc. 10th Euromicro Conference on Parallel, Distributed and Network-Based Processing, EUROMICRO-PDP'02, pp.15-22, (2002).
8) Nieplocha, J. and Carpenter, B.: ARMCI: A Portable Remote Memory Copy Library for Distributed Array Libraries and Compiler Run-time Systems. Proc. RTSPP of IPPS/SDP'99. (1999).
9) Bonachea, D.: GASNet Specification, v1.1, U.C. Berkeley Tech Report (UCB/CSD-02-1207), (2002).
10) 住元真司, 安島雄一郎, 佐賀一繁, 野瀬貴史, 三浦健一, 南里豪志: エクサスケール通信向け ACP スタックの設計思想, 情報処理学会研究報告, Vol.2014-HPC-143 No.8, (2014).
11) 安島雄一郎, 佐賀一繁, 野瀬貴史, 三浦健一, 住元真司: ACP 基本層の設計思想とインタフェース, 情報処理学会研究報告, Vol.2014-HPC-143 No.9 (2014).
12) 佐賀一繁, 安島雄一郎, 野瀬貴史, 三浦健一, 住元真司: ACP 基本層の実装と初期評価, 情報処理学会研究報告, Vol.2014-HPC-143 No.10 (2014)
13) Ajima, Y., Sumimoto, S. and Shimizu. T.: Tofu: A 6D Mesh/torus Interconnect for Exascale Computers, Computer, Vol.42, pp.36-40, (2009).
14) Luszczek, P. R., Bailey, D. H., Dongarra, J. J., Kepner, J., Lucas, R. F., Rabenseifner, R. and Takahashi, D.: The HPC Challenge (HPCC) Benchmark Suite, Proc. 2006 ACM/IEEE Conference on Supercomputing, SC '06, (2006).