Bulletin of Networking, Computing, Systems, and Software – www.bncss.org, ISSN 2186–5140
Volume 3, Number 1, pages 29–33, January 2014

# Topology Aware Performance Prediction of Collective Communication Algorithms on Multi-Dimensional Mesh/Torus

Hironobu Sugiyama
Faculty of Engineering
Kyushu University
6-10-1 Higashi-ku, Fukuoka, 812-8581 Japan
Email: sugiyama.hironobu.559@m.kyushu-u.ac.jp

Yoshiyuki Morie
R.I.I.T of Kyushu University
JST, CREST
Japan
Email: morie.yoshiyuki.404@m.kyushu-u.ac.jp

Takeshi Nanri
R.I.I.T of Kyushu University
JST, CREST
Japan
Email: nanri@cc.kyushu-u.ac.jp

***Abstract*—As the scale of computers becomes large, technologies for appropriate selection of collective communication algorithm have become important. Especially, on Multi Dimensional Mesh/Torus topology, which is popularly used on large-scale supercomputers, this selection is one of the most important issues for achieving sufficient scalability. Currently, the authors are developing a method that combines performance prediction models and runtime measurement to enable efficient algorithm selection. This paper introduces a performance prediction model of algorithms on Multi-Dimensional Mesh/Torus topology. This model considers effects of collisions on links to increase the preciseness of the prediction. Some experiments have been done to show the advantages of the proposed model.**

## I. Introduction

計算機の大規模化により集団通信の性能が重要になっている．集団通信の実装には様々なアルゴリズムが提案されているが，あらゆる状況で最適な性能を出せるアルゴリズムというのは存在しない．また，最も性能の良いアルゴリズムと性能の劣るアルゴリズムとの間で実行時間に 10 倍以上の差がでる場合もあるため，適切なアルゴリズムを選択する事が重要である．アルゴリズムの選択には，アルゴリズムの性能予測が重要な役割を担っている．例えば，性能予測の結果を用いる事で，アプリケーション実行時に性能を計測されるアルゴリズムの候補から，極端に性能の劣るものを予め切り捨てる事が出来る．また，通信ライブラリは予測結果をもとに，使用するアルゴリズムを直接選択する事が出来る．このどちらの場合においても，予測の精度が効率的で正確なアルゴリズム選択の鍵となる．

本稿では，数万ノード規模の大規模計算機で用いられる事の多い，メッシュ／トーラストポロジ上での集団通信アルゴリズムの性能予測手法を提案する．このトポロジでは，通信スピードがリンク上での通信衝突の発生に依存するため，提案手法では通信衝突の影響を見積もる．また，提案手法はアルゴリズムを通信パートとメモリコピーパートの二つに分け，それぞれ個別に見積もりを行う．本稿では，Allgather 通信の 2 つのアルゴリズムに関して性能モデルを作成し，さらにその精度の検証を行う．

## II. Related Work

Tu ら [8] は，マルチコアのノードによるクラスタを対象とし，ノードの階層構造を考慮して垂直方向，水平方向それぞれのデータ移動をモデル化して集団通信の性能を予測している．さらに 4 コアの CPU を 2 基搭載したノード 8 台を Ethernet で接続したクラスタで、提案手法による予測値を実測値と比較し、精度を確認している．しかし，この手法はノード内のプロセス間通信や，不連続なデータ転送に伴うメモリアクセスに関する所要時間を正確に見積もる手法であり，通信衝突による影響については考慮されていない．Kielmann ら [11] は，通信性能が非線形であり、さらに性能の異なるネットワークを組み合わせた階層的な並列計算環境の複雑な通信性能を予測するため、オーバヘッドとメッセージ間ギャップについて、それぞれメッセージサイズをパラメータとした関数で表す P-LogP モデルを提案し，アルゴリズム選択という目的に対して，提案手法による予測精度が十分であることを確認した．しかしながら，通信衝突の影響は考慮に入れていない．Pjesivac ら [3] は，Hockney, LogP/LogGP, P-LogP のそれぞれの一対一通信モデルをベースに、集団通信アルゴリズムの性能予測モデルを提案し，その精度を確認している．さらに、Broadcast について、メッセージサイズとプロセス数毎に最速だったアルゴリズムを、実測値と予測値それぞれについて示し、アルゴリズム選択への予測モデルの適用可能性を議論している。しかし，アルゴリズム毎の細かい挙動や，ランク配置による通信衝突の影響に関しては考慮していない．

一方，通信衝突を考慮した性能予測に関する研究も行われている．Girona ら [12] は，集団通信をアルゴリズムのステップ数と転送されるデータサイズで抽象化し、さらに各ステップの所要時間を同時発行通信数と使用可能なバスの本数から見積もることで、衝突の影響を考慮した集団通信の所要時間を算出している。通信のタイミングを考慮するという点では著者らの研究に近いが、通信衝突の見積もり方法としては、使用可能なバスの本数と同時通信数だけに基づいており、特にシステム規模が大きくなり、衝突の影響が大きくなった場合の予測精度に問題があると考えられる。また，Steffenel ら [13] は，衝突を考慮しない場合の予測時間に対する、実環境での実測時間の比 “contention ratio” を適用することで、衝突の影響を含んだ集団通信の性能予測を可能としている．しかし，ランク配置によって衝突の影響が変化するため、全てのランク配置パターンに対して “contention ratio” の計測が必要で有り、多次元メッシュ/トーラスでは実用的で無い。森江ら [2] は，多次元トーラスを対象として，出来るだけ通信衝突の発生を抑えるためのタスク配置最適化技術を提案しており，その中で通信衝突回数を見積もる手法も合わせて提案している．本研究では，この手法を通信衝突の影響を考慮した性能見積もりに応用する．

また，これらの研究で提案されている性能モデルはメモリコピーを考慮に入れておらず，メモリコピー操作を含むアルゴリズムに対しては十分な精度が得られないと考えられる．さらに，多くの研究どが線形モデルを用いている．この中で，P-LogP は高精度な性能予測が出来ると期待できる．我々は P-LogP とは異なる非線形のモデルを用いて予測を行う．そこで本稿では，メモリコピーと通信衝突を考慮にいれた，新たな非線形モデルを提案する．

## III. Collective Communication

### A. Importance of Collective Communication

集団通信とは，複数プロセスで構成されたグループによる 1 対全，もしくは全対全の定型パターン通信である．プロセス数が増えるに従って，プログラムの実行時間に占める集団通信の割合は増加していくため，集団通信を多く用いる科学技術計算等において，その性能は重要である．集団通信の例として，Allgather, Allreduce, Alltoall, Barrier などがある．それぞれの集団通信の実装には複数のアルゴリズムが用意されており，集団通信が呼び出されると，メッセージサイズやプロセス数などの状況に応じて用意されているアルゴリズムの中から一つが選ばれ実行される．例えば，Allgather と呼ばれる集団通信は，それぞれのプロセスが他の全プロセスからデータを受け取り，受け取ったデータをソースプロセスの順番に配置するものであり，行列の内積計算などの際に多く用いられる．

### B. Algorithms of Allgather

集団通信は基本的に一対一通信の組み合わせで構成されており，その組み合わせ方をアルゴリズムと呼ぶ．Allgather の実装アルゴリズムとしては，Open MPI では Bruck, Ring, Recursive Doubling, Neighbor Exchange が用意されている．この中から特に，Ring アルゴリズムと Bruck アルゴリズムの疑似コードを Fig. 1, 2 にそれぞれ示す．これらの図中で，P はプロセス数，M はメッセージサイズ，rank はプロセス番号を表している．

Ring アルゴリズムは，各プロセスがデータをリング状に隣へ隣へと渡していく事で，P-1 回の通信操作により Allgather を実現している．一方で Bruck アルゴリズムは，各プロセスがもとから持っていたデータに加え，他のプロセスから受け取ったデータをまとめて次の通信相手に渡す事で，通信回数を $\lceil log_2 P \rceil$ 回に削減する事が出来る．しかし，送受信するデータ量が通信を重ねるごとに増加するほか，データをランク順に並べるためのメモリコピー操作が必要になる．

```
for i = 0 to P - 2
  sendrecv(M)
```

Fig. 1. Pseudo-code of Ring Algorithm

```
for i = 0 to ⌈log2(P)⌉ -1
  sendrecv(M * 2^i)
memcopy((P - rank)*M)
memcopy(rank*M)
memcopy((P - rank)*M)
```

Fig. 2. Pseudo-code of Bruck Algorithm

## IV. Performance Models of Collective Communication Algorithms

### A. Policy

提案手法ではアルゴリズムを通信パートとメモリコピーパートの 2 つにわけ，それらを個別に見積もりその総和を取る事でアルゴリズムの所要時間とする．通信パートに関して，今回は以下を前提としている．

1) 通信パートは，1 つ以上のステップに分かれている
2) 各ステップは全プロセスで同時に開始する
3) 各ステップの所要時間は，そのステップで最も時間を要したプロセスの所要時間である
4) 全ステップの所要時間の和が，通信パートの所要時間である

通信とメモリコピーの性能見積もりは，簡単なベンチマークプログラムを用いて対象とするシステム上でそれぞれの基本操作の性能を計測し，その結果をもとに行う．

通信のベンチマークは，隣接する 2 ノード間の Send と Receive による Ping-Pong 通信プログラムを用いる．複数のメッセージサイズに関して，データの送受信を十分な回数繰り返し，その所要時間を繰り返し回数で割る事で，1 回当たりのデータ転送に要する時間を求める事が出来る．

一方，メモリコピーのベンチマークには，あるメモリ領域 A の先頭から M バイトを別のメモリ領域 B にコピーするための関数である memcpy を単独で行うプログラムを用いる．通信のベンチマークと同様に，複数のメッセージサイズに関して memcpy を十分な回数繰り返し，その所要時間の平均を取る事で基本性能を求める．

メモリコピーの計測の際問題となるのがキャッシュの影響である．メッセージサイズがキャッシュサイズよりも小さい範囲においては，2 回目以降の繰り返しの際にキャッシュの影響により，集団通信アルゴリズム内部で実際に行われるメモリコピーよりも高速になる可能性がある．そのため，メッセージサイズがキャッシュよりも小さい範囲では，繰り返し毎にコピー元とコピー先のメモリアドレスを変えながら計測をする必要がある．

### B. Calculation of the time for the message size

あるメッセージサイズにおける所要時間を求める最も簡単な方法は，前節で述べたベンチマークプログラムによる計測結果に対して，最小二乗法で線形式に近似する事である．

しかし，メッセージサイズに対する所要時間の変化が線形でない場合最小二乗法では誤差が大きいため，我々の提案するモデルでは，目的のメッセージサイズに近い 2 つの計測値から目的のメッセージサイズにおける所要時間を計算する．例えば，ベンチマークプログラムによりメッセージサイズ 100, 200, 300 と飛び飛びにサンプルポイントを取っているとする．この時，メッセージサイズ 150 における所要時間を求めたい場合はその前後のサンプルポイント 100 と 200 の計測値を線形に近似して予測通信時間を算出する事で求める事が出来る．

メッセージサイズ M において線形近似を行った場合の切片を $L_M$，傾きを $\alpha_M$ とすると，所要時間は式 (1) によって与えられる．この式を，通信とメモリコピーそれぞれに適用し，その和を取る事で集団通信アルゴリズムの所要時間を求める事が出来る．

$$t_M = L_M + \alpha_M M \quad (1)$$

### C. Base Performance Models for Algorithms of Allgather without collision

式 1 を用いて構築した，Bruck, Ring それぞれの性能モデルを式 (2)，(3) に示す．

Bruck は Fig. 2 から分かる通り，通信とメモリコピーから構成されている．通信は，$\lceil log_2 P \rceil$ ステップで構成されており，ステップごとにメッセージサイズが増加していく．また，メモリコピーは計 3 回実行され，ランクによってコピーされるデータ量が異なる．提案モデルでは，最もデータ量の大きくなるランク 1 の値を用いる．この時，Bruck の性能モデルを式 (2) に示す．

$$\begin{aligned} T_{Bruck,M} &= t_{mem,M} + t_{comm,M} \\ &= (L_{mem,M} + \alpha_{mem,M}) + 2(L_{mem,M} + \alpha_{mem,(P-1)M}) \\ &\quad + \sum_{step=0}^{\lceil log_2 P \rceil} (L_{comm,M} + \alpha_{comm,M} M_{step}) \end{aligned} \quad (2)$$

Ring アルゴリズムは，Fig.3 の示す通り，P-1 回の通信によって構成されている．そこで，Ring の性能モデルを式 (3) に示す．

$$\begin{aligned} T_{Ring,M} &= t_{comm,M} \\ &= \sum_{step=0}^{P-2} (L_{comm,M} + \alpha_{comm,M} M_{step}) \end{aligned} \quad (3)$$

## V. Estimation of the Effect of Collisions on Communication Performance

IV-C 章で述べた性能モデルでは，通信衝突の影響を考慮していないため，衝突の影響を考慮する必要がある．例えば，アルゴリズム x を 4 プロセスで実行する場合，y 番目のステップでランク a とランク b，ランク c とランク d の通信が同時に発生すると想定される．ここで，プロセスの配置が，Fig. 3 左の通りである場合，ランク a とランク b，ランク c とランク d の 2 つの通信が衝突する．これにより，このステップの通信時間は IV-C 章での予測より長くなる．一方，プロセスの配置が Fig. 3 右の通りである場合，これらの 2 つの通信による衝突が発生しないため，IV-C 章の予測結果に対する影響はない．このように，集団通信アルゴリズムの性能予測には，プロセス配置情報に基づいた通信衝突の予測結果を加味する必要がある．

提案手法では，各ステップで行われる通信をシミュレートすることで通信衝突の影響を見積もり，各ステップの時間を計算する．各ステップを通した合計時間をそのアルゴリズムの通信時間と見なす．各ステップでの通信衝突の見積もりは，トポロジやルーティングポリシーなどのシステム情報とともに，通信パターンやプロセス配置情報を用いる事で行う．まずプロセス配置とルーティングポリシー，通信パターンから，各通信の使用する経路がわかるため，全通信の使用する通信経路を探索し，同一方向に同一リンクを用いる通信を見つけ，それを衝突と見なす．複数箇所で衝突が発生した場合，メッセージの転送スピードは衝突数最大のリンクによって制限される．そのため提案手法では，最大衝突数 C，各ステップで転送されるメッセージサイズ M に対し，そのステップで転送されるメッセージサイズを C×M バイトと見なして通信時間を見積もる事とする．


(a) Rank Allocation 1　(b) Rank Allocation 2

Fig. 3. Relation Between Rank Allocation and Collision

## VI. Experiments

予測精度の調査のため，提案手法による予測時間を実測時間と比較をした．実験環境は以下の通りで,「京」コンピュータの互換機である九州大学の Fujitsu PRIMEHPC FX10 を使用している.「京」コンピュータとの主な違いは，クロック数，コア数，ノード当たりのメモリサイズである．

TABLE I. Experimental Environment

| | |
|---|---|
| System | Fujitsu PRIMEHPC FX10 (compatible with K computer system) |
| CPU | Fujitsu SPARC64 IXfx (1.848Ghz, 16core), 1CPU/node |
| Memory | 32GB/node |
| Network | Tofu interconnect (6D mesh/torus) |
| Nodes | 768 |
| Link BW | 5GB/sec |
| OS | Proprietary OS (Linux-based) |
| Compiler, MPI | Fujitsu Technical Computing Suite v1.0 |

今回実験に使用した FX10 のネットワークは物理的には 6 次元である．しかしジョブスケジューラはそれを 2 次元ずつにまとめて仮想的に 3 次元で構成しており，利用者は使用したいノードを 3 次元で指定する事が可能である．それゆえ，本実験中のノード形状は 3 次元で表現する．

実験では，2 つの通信アルゴリズムについて複数のメッセージサイズとプロセス配置で実行時間を計測する．複数の実験結果から，特に形状 1x6x4 の Bruck と形状 2x3x4 の Ring の予測時間と実測時間の比を Fig. 4 と Fig. 5 に示す．

グラフ中の no consideration は予測に通信衝突を考慮していない場合，in consideration は通信衝突を考慮した場合の結果を表している．また，横軸はメッセージサイズ，縦軸は予測時間を実測時間で割った値であり，各メッセージサイズで縦軸の値が 1 に近いほど予測精度が高いと考える．ほとんどの場合で，衝突を考慮することで予測と実測の比は 1 に近くなっている．この結果は，衝突を考慮する事で予測の精度が向上している事を示す．また，提案手法の効果はメッセージサイズが大きい部分で顕著である．なぜならサイズが大きいほど通信スピードに与える衝突の影響が大きいからである．

## VII. Discussion

### A. Accuracy of Prediction

Fig. 4，Fig. 5 を見ると，メッセージサイズが小さい所では縦軸の値が 1 より大幅に低くなっている事が分かる．これは，予測所要時間を実際よりも短く見積もりすぎている事を示しており，要因として予測の際に考慮していなかった 2 つの要素が考えられる．

1 点目は，ホップ数による影響である．ホップ数とは，通信時に経由するスイッチの数である．今回，通信時間の基礎データ測定には 1 ホップでの Ping-Pong 通信を行い，全ての通信を 1 ホップと仮定して予測をしているが，実際にはホップ数に応じたレイテンシーが加わる．

2 点目は，ロードインバランスによる影響である．IV-A 節で述べた通り，通信パートは複数のステップに分かれて

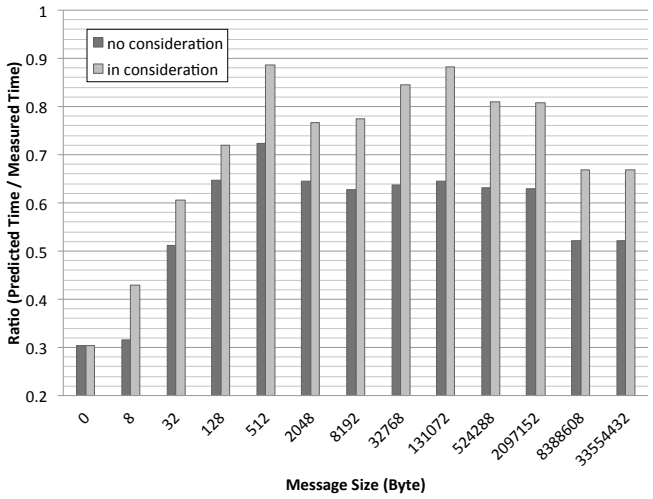


Fig. 4. Ratio (Predicted Time / Measured Time) of Bruck Algorithm with Shape 1x6x4

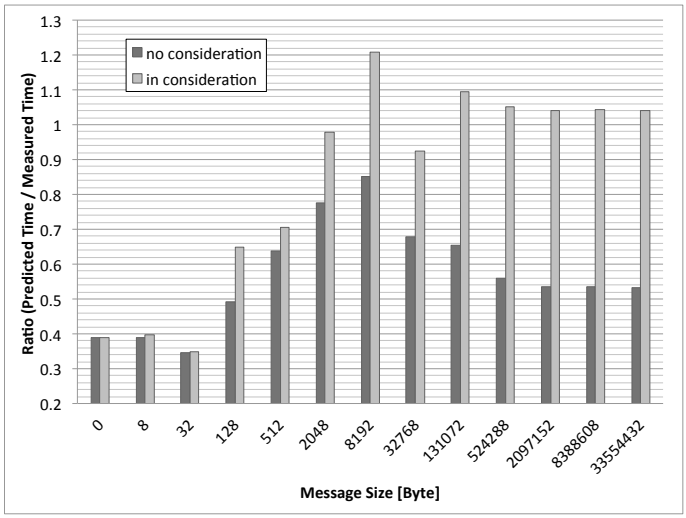


Fig. 5. Ratio (Predicted Time / Measured Time) of Ring Algorithm with Shape 2x3x4

おり，全プロセスが各ステップで同時に通信を開始する事を前提とし，各ステップの通信時間をそれぞれ個別に見積もっている．ロードインバランスにより各プロセスで開始時間にばらつきが生じると，その後の通信ステップでの受信待ちにつながり，性能が低下する可能性が高くなる．

メッセージサイズが小さい範囲では通信そのものの時間が短いため，これらの影響が顕著に現れているのではないかと考えられる．それゆえ，より精度を向上させるためにはこれらの要素について検討する必要がある．

### *B. Possibility of Algorithm Selection Using the Performance Models*

提案手法のアルゴリズム選択への影響を調査するために，予測精度とは別に予測時間と実測時間を比較した．Fig. 6, Fig. 7 は横軸にメッセージサイズ，縦軸に所要時間をとったものであり，両グラフとも実線は実測時間，破線は予測時間を示している．Fig. 6 で予測時間の線は通信衝突を考慮していない状態のもので，Fig. 7 で予測時間の線は提案手法による予測結果である．これらのグラフそれぞれで，実線同士，破線同士が交わる点は Bruck アルゴリズムと Ring アルゴリズムの性能が入れ代わるメッセージサイズを示しており，Fig. 6 では実測値と予測値でこのメッセージサイズが大きくずれている．一方，Fig. 7 では実測値と予測値それぞれの線が交わる点のメッセージサイズがほぼ一致している事が分かる．つまり，通信衝突の影響を考慮に入れる事で，より早いアルゴリズムにかわるメッセージサイズを正確に予測する事ができるようになるという事である．それゆえ，提案手法は適切なアルゴリズム選択に対し有用であると考えられる．

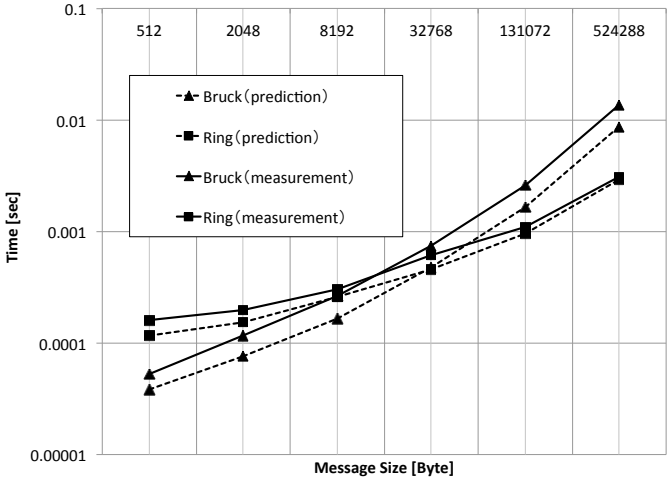


Fig. 6. Measured Time and Predicted Time (without Consideration of Collisions) of Ring and Bruck Algorithms

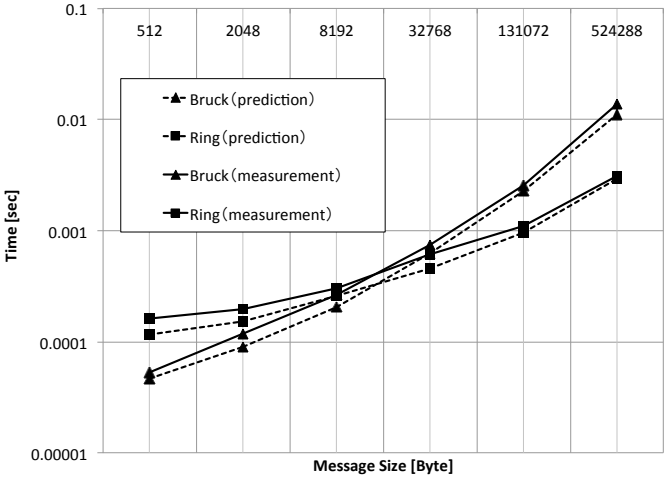


Fig. 7. Measured Time and Predicted Time (with Consideration of Collisions) of Ring and Bruck Algorithms

## VIII. Conclusions and Future Works

本稿では，多次元メッシュ/トーラストポロジにおける集団通信アルゴリズムの性能予測手法の提案と，その予測精度の評価実験，及びアルゴリズム選択に対する提案モデル

の有用性について検証した．実験結果より，リンク上での通信衝突の影響を考慮する事により提案手法での精度向上を確認した．さらに，提案手法を用いる事により，アルゴリズムの切り替わる閾値をより正確に予測する事が出来る事を示した．

今後の課題としては，小メッセージサイズ域での予測精度の向上や，提案手法の他のアルゴリズムや集団通信への適用，集団通信アルゴリズムの動的選択への統合がある．

## REFERENCES


[1] Lastovetsky, A. and O'Flynn, M., "A Performance Model of Many-to-One Collective Communications for Parallel Computing", *In Proceedings of IEEE International Parallel and Distributed Processing Symposium*, pp. 1–8, 2007.

[2] Morie, Y. and Nanri, T., "Task Allocation Technique for Avoiding Contentions on Multi-Dimensional Mesh/Torus," IPSJ Transactions on Advanced Computing System, Vol. 6, No. 3, pp.12—21, 2013.

[3] Pjesivac-Grbovic', J., Angskun, T., Bosilca, T., Fagg, G.E., Gabriel, E. and Dongarra, J.J., "Performance Analysis of MPI Collective Operations", Cluster Computing, vol 10, issue 2, pp. 127–143, 2007.

[4] Hamid, A. and Coddington, P., "Analysis of Algorithm Selection for Optimizing Collective Communication with MPICH for Ethernet and Myrinet Networks", *8th International Conference on Parallel and Distributed Computing*, Applications and Technologies, pp. 133–140, 2007.

[5] Sivac-Grbovi'c, J. P., Fagg, G. E., Angskun, T., Bosilca, G. and Dongarra, J., "MPI Collective Algorithm Selection and Quadtree Encoding", *In Proceedings of the 13th European PVM/MPI User's Group Meeting*, pp. 40–48, 2006.

[6] Nishtala, R., "Automatically Tuning Collective Communication for One-Sided Programming Models", PhDThesis, UC Berkeley, Berkeley, 2009.

[7] Nanri, T. and Kurokawa, M., "Effect of Dynamic Algorithm Selection of Alltoall Communication on Environments with Unstable Network Speed", *International Conference on High Performance Computing and Simulation (HPCS), 2011*, pp. 693–698, 2011.

[8] Tu, B., Fan, J., Zhan, J. and Zhao, X., "Performance analysis and optimization of MPI collective operations on multi-core clusters", The Journal of Supercomputing, vol 60, issue 1, pp. 141–162, 2012.

[9] Hoefler, T., Schneider, T. and Lumsdaine, A., "LogGP in Theory and Practice- An In-depth Analysis of Modern Interconnection Networks and Benchmarking Methods for Collective Operations", Elsevier Journal of Simulation Modelling Practice and Theory (SIMPAT). vol 17, Nr. 9, pp. 1511–1521, 2009.

[10] Pjeivac-Grbovi, J., Angskun, T., Bosilca, G., Fagg, G. E., Gabriel, E. and Dongara, J. J., "Performance analysis of MPI collective operations", Cluster Computing, vol 10, issue 2, pp. 127–143, 2007.

[11] Kielmann, T., Bal, H., Gorlatch, S., Verstoep, K. and Hofman, R., "Network performance-aware collective communication for clustered wide-area systems", Clusters and computational grids for scientific computing, vol 27, issue 11, pp. 1431–1456, 2001.

[12] Girona, S., Labarta, J. and Badia, R. M., "Validation of Dimemas Communication Model for MPI Collective Operations", Recent Advances in Parallel Virtual Machine and Message Passing Interface, Lecture Notes in Computer Science vol 1908, pp. 39–46, 2000.

[13] Steffenel, L. A., "Modeling Network Contention Effects on All-to-All Operations", Cluster Computing, 2006 IEEE International Conference on, pp. 1–10, 2006.

[14] Steffenel, L. A., Martinasso, M. and Trystram, D.: Assessing Contention Effects on MPI alltoall Communications, *In proceedings of GPC 2007*, pp. 424–435, 2007

[15] Faraj, A., Yuan, X. and Lowenthal, D.: STAR-MPI: Self Tuned Adaptive Routines for MPI Collective Communications, *In proceedings of International Conference on Supercomputing*, pp. 199–208, 2006.

[16] Nanri, T. and Kurokawa, M.: Efficient Runtime Algorithm Selection of Collective Communication with Topology-Based Performance Models, *In proceedings of the 2012 International Conference on Parallel and Distributed Processing Techniques and Applications*, 2012.

[17] Pjesivac-Grbovic, J., Angskun, T., Bosilca, G., Fagg, G. E., Gabriel, E. and Dongarra, J. J.: Performance analysis of MPI collective operations, Published in Cluster Computing, vol 10, issue 2, pp. 127–143, June. 2007.

[18] Barker, J., Davis, K. and Kerbyson, J. J.: Performance Modeling in Action : Performance Prediction of a Cray XT4 System during Upgrade, *In proceedings of IEEE International Parallel & Distributed Processing Symposium*, 2009.