通信タイミングを考慮したランク配置最適化技術

森江, 善之
九州大学大学院システム情報科学府

末安, 直樹
富士通株式会社ソフトウェア事業本部ミドルウェアコンポーネント事業部

松本, 透
富士通株式会社ソフトウェア事業本部ミドルウェアコンポーネント事業部

南里, 豪志
九州大学情報基盤センター

他

出版情報：情報処理学会研究報告, 2007-HPC-109. 2007 (17), pp.145-150, 2007-03-02. Information Processing Society of Japan
バージョン：

通信タイミングを考慮したランク配置最適化技術

森江　善之 [†1]　末安　直樹 [†2]　松本　透 [†2]　南里　豪志 [†3]
石畑　宏明 [†3] [†4]　井上　弘士 [†5]　村上　和彰 [†5]

[†1] 九州大学大学院システム情報科学府
[†2] 富士通株式会社ソフトウェア事業本部ミドルウェアコンポーネント事業部
[†3] 九州大学情報基盤センター
[†4] 富士通株式会社サーバシステム事業本部次世代 HPC 開発企画室
[†5] 九州大学大学院システム情報科学研究院

概要　本稿では，通信性能の悪化の主要因である通信の衝突を避けるためのランク配置最適化技術の提案を行う．メッセージごとに通信のタイミングを考慮することで，衝突を回避する高精度な MPI ランク配置最適化の提案を行った．また，本手法では衝突を制御するための同期関数の挿入を行う必要がある．このオーバーヘッドを含めてどれほどの実行時間の削減効果があるか評価実験を行った．対象プログラムとしては recursive doubling の通信パターンや CG 法や umt2000 といった実アプリケーションの通信パターンを用いた．順配置に対して最大 45 %，従来研究の出力したランク配置に対して最大 24 %程度の通信時間の削減効果を示し，本手法の有効性を確認した．

Optimization of rank allocation considerin communication timing

Yoshiyuki Morie, [†1] Naoki Sueyasu, [†2] Toru Matsumoto, [†2] Takeshi Nanri, [†3]
Hiroaki Ishihata, [†3] [†4] Koji Inoue [†5] and Kazuaki Murakami [†5]

[†1]Graduate School of Information Science and Electrical Engineering, Kyushu University
[†2]Middleware Components Division, Software Unit, FUJITSU LIMITED
[†3]Computing and Communications Center, Kyushu University
[†4]Next Generation HPC Development Planning Department, Server Systems Unit, FUJITSU LIMITED
[†5]Graduate School of Information Science and Electrical Engineering, Kyushu University

Abstract　In this paper, it proposes the rank optimization of rank allocation technology of avoiding the communication contention that is the key factor of the deterioration of the communication performance. It proposes the method is possible to ward off a communication contention was allcated by considering the communication-timing of each message. Moreover, this method has a overhead that it has to add synchronous function. In the evaluation experiment, it check how does this method cut down communication time including that overhead. The communication pattern of the recursive doubling and the communication pattern of the real application such as CG and umt2000 are used in this evalucation experiments. The ratio of reduction in the communication time are 45 % or less for order rank allocation , 24 % or less for previous work rank allcation in the experiment.

1　はじめに

先端科学技術計算は，その性質上，並列計算機を用いて計算されることが多い．この先端科学技術計算においては，近年，性能要件がさらに厳しくなっている．このため，スーパーコンピュータなどの高速な並列計算機に対してはより高い性能向上を求められている．このような要求に対して，大量の計算ノードを搭載した分散メモリ型の計算機の導入が進められている．特にスーパーコンピュータの計算ノード数は非常に大きいものとなってきている．例えば，2006 年 11 月発表のスーパーコンピュータ性能ランキング TOP500[6] において 1 位となった IBM の BlueGnen/L では約 13 万個の計算ノード (2CPU/1node) を用いている．

しかし，計算ノード数が大きくなるとそのネットワークを多段のクロスバスイッチや Fat-Tree などの高速かつ均質なネットワークで構成することは価格

や電力などのコスト，設置面積などの問題で困難となる．そこで，結線数やスイッチ数を削減するため，Tree，3D-mesh，3D-Torus などのネットワークトポロジで大規模並列計算機のネットワークを構築することが増加している．しかしながら，これらの不均質なネットワークでは，通信の衝突が起こり易くなり，通信時間が増大する可能性が高い．

本研究は，このように不均質なネットワークトポロジを起因とした通信の衝突による通信時間の増大を削減することが目的である．

本研究で提案する手法は，通信の衝突を回避するようにランク (タスク) を計算ノードに割り付けることにより通信時間の削減を図る．このような最適化をランク配置最適化と呼ぶ．本研究では，メッセージごとのタイミングを調べ，同時に実行されるメッセージをフェーズとして区切ることで衝突を制御できるようにし，通信の衝突を回避するランク配置最適化を提案する．

2 章では，従来研究の紹介を行う．3 章では，通信タイミングを考慮したランク配置最適化の提案を行い，4 章では，提案手法の評価実験について述べる．ここでは，特にフェーズに区切ることで生じるオーバヘッドについての評価を行う．最後にまとめと今後の課題について述べる．

2 従来研究

通信時間の削減を目的とした従来のランク配置最適化では，ランクを節，ランク間の通信量を枝の重みに持つグラフを作り，並列計算機のネットワークトポロジにそのグラフを通信コストが最小になるように割り付ける．ランク配置の探索にかかる計算量は膨大なため，一般に，ネットワークトポロジや計算ノード数などをパラメータとする発見的な手法が用いられる．そのため，発見的手法のアルゴリズムや最適化の目的関数に関して広く研究されている．

T. Hatazaki[2] や J. L. Traff[3] の提案したランク配置最適化では，ツリー上のネットワークトポロジを対象としている．このネットワークトポロジでは上段になるほど低速になる階層型のネットワークを想定しており，通信のやり取りを多く行うランク同士が上位のネットワークを使わないように配置する．これにより上位の低速なネットワークを使われることが減り，通信時間の削減が行われる．F. Ercal ら [1] の提案では，任意のネットワークトポロジを対象としている．各計算ノード間でネットワークの通信性能が異なるため，通信量の多いランク同士は高速なネットワークで結ばれた計算ノードに割り付ける．これにより，通信時間の削減が行われる．しかし，これらの研究では，通信の衝突は考慮に入れられていない．

T. Agarwal ら [4] や G. Bhanot ら [5] の提案では，3D-Mesh や 3D-Torus，Tree などの不均質なネットワークトポロジを対象としている．計算ノードやスイッチ間の接続を行うネットワークの性能はすべて等しいと仮定する．3D-Mesh や 3D-Torus，Tree のようなネットワークトポロジでは，直接接続していない計算ノード同士が多数存在する．これらの計算ノードが通信を行う際は，複数の計算ノード，またはスイッチを通過して通信を行わなければならない．このことはレイテンシの増加や通信の衝突を発生を引き起こし，通信時間を増加させる．このため，これらの研究では，レイテンシや通信の衝突を削減するため，出来る限り通信が計算ノードやスイッチを通過しないようにランク配置を行うランク配置最適化手法を提案している．このランク配置最適化により，レイテンシや衝突が減少し，通信時間の削減が行われる．しかし，並列計算機に用いられる高性能ネットワークにおいては，ネットワークケーブルやスイッチを跨ぐことによるレイテンシの増加は，MPI 通信ライブラリやネットワークドライバなどのソフトウェアの実行にかかるレイテンシに比べて，小さい．これに対して通信の衝突による性能の低下は，プログラム性能に影響するほど大きい場合がある．従来手法においてもスイッチや計算ノードを超える通信が少なくなるので，通信が衝突する可能性が低くなるが，メッセージの通信のタイミングを考慮しないため，通信の衝突を回避を十分に行えない．

3 通信タイミングを考慮したランク最適化

本研究では，メッセージごとの通信タイミングを考慮し，ランク配置を決めることでメッセージごとの通信の衝突を回避し，より高速な通信を可能とするランク配置最適化の提案を行う．

3.1 通信タイミングを考慮したランク配置の例

はじめのステップでランク 1 とランク 2 の間で通信を実行し，次のステップでランク 4 とランク 5 の間で通信を実行する．さらに，次のステップでランク 1 とランク 3，ランク 4 とランク 6 の間で通信を実

行し，最後のステップでランク 1 とランク 6，ランク 2 とランク 4，ランク 3 とランク 5 の間で通信を実行する．この時，通信は各ステップにならなければ実行できないものとする．

このような通信パターンを持つランクを，ツリー型ネットワーク上の計算ノードに割り当てる．スイッチ A に接続する計算ノードにランク 1，2，3，スイッチ B に接続する計算ノードにランク 4，5，6 が割り当てられていたとする．このランク配置をランク配置 1 とする．

図 1 のように第 1，2，3 ステップではスイッチ AB を跨ぐ通信は行われない．第 4 ステップにおいて 3 つの通信が同時にスイッチ AB を跨ぐため，通信の衝突が起き，実行時間が増大する．

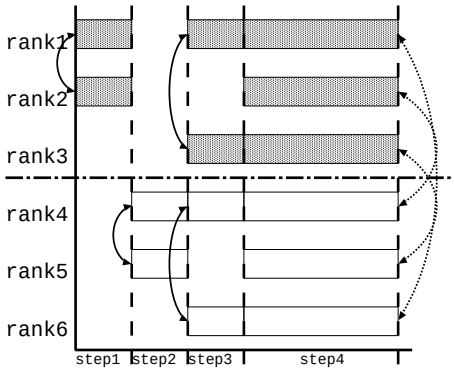

図 1: ランク配置 1 の時の通信の様子

次にメッセージごとの通信のタイミングを考慮に入れ，衝突が起きないようにランク配置した場合を考える．この配置では，スイッチ A に接続する計算ノードにランク 1，3，5，スイッチ B に接続する計算ノードにランク 2，4，6 が割り当てる．このランク配置をランク配置 2 とする．

このランク配置での通信の様子を図 2 に示す．第４ステップではランク 3 とランク 5 がスイッチ A に接続しているため，これらのランク間で行う通信ではスイッチ AB を跨ぐこと無い．また，ランク 2 とランク 4 はスイッチ B に接続しているため，ここでもスイッチ AB を跨ぐ通信は行わない．また，ランク 1 がスイッチ A，ランク 6 がスイッチ B に接続しているため，スイッチ AB を跨ぐ通信を行う．しかし，これ以外のスイッチ AB を跨ぐ通信は行われないため，通信の衝突は発生しない．このため，ランク配置 1 に比べ，ランク配置 2 では通信時間が削減される．

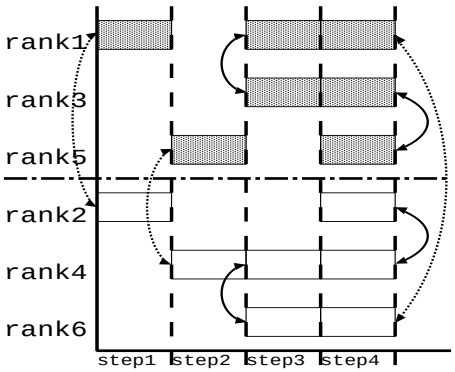

図 2: ランク配置 2 の時の通信の様子

このランク配置 1 とランク配置 2 は例となる参考文献で用いられている目的関数では同じ通信コストであると見なされる．これは，従来研究の目的関数がスイッチや計算ノードを跨ぐ通信量の総和を最小にするということを目指しているからである．このことから通信の衝突の回避を行うランク配置最適化ではメッセージごとのタイミングを考慮する必要があると考えられる．

3.2 メッセージの衝突を考慮した目的関数

前節のようなランク配置最適化を行うためには，メッセージごとに衝突が起こるかどうか考慮してランク配置最適化を行うことが必要である．このため，式 1 に示す目的関数の導入を行った．提案する目的関数では，通信による経過時間の予想値を求める事した．これは，メッセージの衝突がどれほどの通信時間の増加を招くかを正確に推定するためである．

$$F = \Sigma_t^N \max_i^n (L(\pi(i),\pi(j)) + (S(t,i,j) + Scoll(t,i,j))/B(\pi(i),\pi(j))) \quad (1)$$

t は t 番目のフェーズ，N はフェーズの総数である．ここでのフェーズは，同時刻に通信を開始するすべての MPI 通信が終了するまでの処理のひとかたまりの事をいう．このようにフェーズという概念を導入することで，通信の衝突の時間変化を検出することが出来る．通信の衝突の時間変化とは，各フェーズおける通信の衝突回数の時間的な変化のことをいう．全てのフェーズで通信の衝突回数が等しければ，通信の衝突の時間変換は無いという．i, j はランク番号，n はランク総数，$\pi(i)$ はランク i が割り付けられている計算ノードを返す関数である．$L(p,q)$ は計算ノード p，計算ノード q 間のレイテンシと通信帯域幅である．$S(t,i,j)$ はフェーズ t における i から

ランク j へのメッセージサイズであり，$Scoll(t,i,j)$ はフェーズ t における i からランク j へのメッセージと衝突を起こすメッセージサイズの総和である．この $Scoll(t,i,j)$ により，メッセージごとに通信の衝突が発生したときの通信時間の増大を見積もる．

このように目的関数を定義することでフェーズごとに通信の衝突がどれほど発生しているか知ることが可能となる．このため，この目的関数を最小にするランク配置を探索することは，全フェーズを通したメッセージの通信の衝突回数を最小とするランク配置を探索することと同義なり，全通信時間は削減することに繋がる．

3.3 同期関数の挿入

前節のような目的関数を実現するには，通信をフェーズごとに分解する必要がある．図 3 のようにフェーズとする通信関数の処理のひとかたまりの直前に同期関数 MPI_Barrier を挿入する．

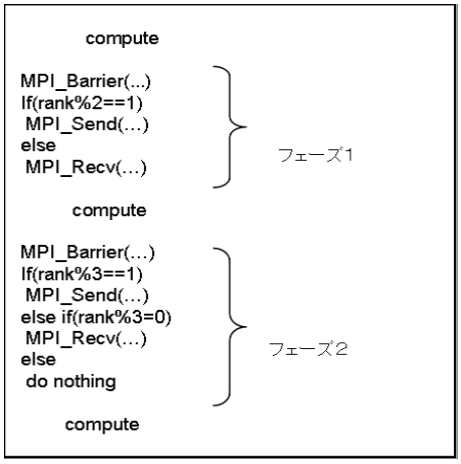

図 3: 同期関数 MPI_Barrier の挿入

これは，ランク配置の変更により各メッセージの通信時間に変化が生じ，通信のタイミングがずれ，想定していない衝突の発生を防ぐためである．従来手法では通信タイミングを考慮することは無く，同期関数を挿入しない．順配置も同様に同期関数の挿入は行わない．

4 実験

通信タイミングを考慮したランク配置最適化の効果を示すための評価実験を行う．

表 1: 実験環境

CPU	Intel Xeon 3.0GHz
L2 キャッシュ	2MB
RAM	7GB
計算ノード数	16(2CPU/nodes)
ネットワーク	Gigabit Ethernet
トポロジ	2 段のツリー
OS	RedHat Enterprise Linux AS (Linux kernel 2.4.21)
コンパイラ	gcc version 3.2.3
MPI ライブラリ	mpich-1.2.7p1

4.1 準備

実験環境としては表 1 に示す PC クラスタを用いる．ネットワークは，図 4 のように 5 つのスイッチを用いて，2 段のツリートポロジを構成する．

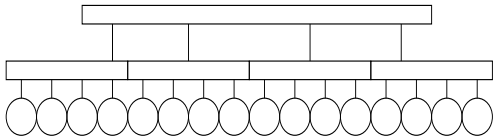

図 4: PC クラスタのネットワークトポロジ

また，対象プログラムとして，全体全の集団通信である allredecue などで用いられる recursive doubling の通信パターン及び実アプリケーションの通信パターンを用いた．実アプリケーションの通信パターンとしては，Nas Parallel Benchmark[8] の CG 法 (Conjugate Gradient method) の通信パターンと ASCI Purple Benchmark[7] の umt2000(Unstructured Mesh Transport 2000) の通信パターンを用いた．非同期通信などで同時に複数の通信がある場合はランク番号が小さい方を優先して通信を行うと仮定を行った．

4.2 比較対象

順配置，及び従来手法により出力されたランク配置についても通信時間の評価を行った．これらのランク配置と今回提案した目的関数を最小にしたランク配置の比較を行う．順配置とは，図 5 に示すようにスイッチごとに 1 から順にランクを計算ノードに割り当てるランク配置のことをいう．また，従来手法では，T. Agarwal らや G. Bhanot らが提案している目的関数を用いた．T. Agarwal らや G. Bhanot らの提案では，不均質なネットワークを対象としており，

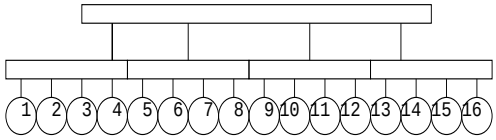

図 5: 順配置

本研究と対象が同じである．また，衝突をある程度考慮出来る点も本研究の比較評価実験を行う際に有用であると考えた．従来手法で提案している目的関数を式 2 に示す．

$$F' = \Sigma_i^n C(i,j) H(\pi(i), \pi(j)) \quad (2)$$

i, j はランク番号，n はランク総数とする．$\pi(i)$ はランク i が割り付けられている計算ノードを返す関数である．$C(i,j)$ はランク i からランク j への総通信量を表わす．また，$H(p,q)$ はネットワーク上における計算ノード p から計算ノード q への距離を示す．距離とは，スイッチや計算ノードを跨ぐ回数のことを言う．この目的関数を最小とするランク配置は，通信量の多いランク同士をなるべくスイッチや計算ノードを跨ぐ回数が少なくなるように計算ノード同士へ割り付けられることになる．通信パターンと対象ネットワークトポロジが一致しない場合は，複数のノードやスイッチを超えて通信を行うことになるため通信性能の向上が見込める．

4.3 実験方法

通信パターンは各プログラムのソースコードから抽出する．また，ネットワークトポロジ情報は図 4 のようになっており，これらを目的関数の入力として与える．従来手法の目的関数をランク配置最適化と本手法において提案した目的関数を用いたランク配置最適化を行う．今回は 2 段ツリートポロジを想定しているので，ルーティングは行われず，提案手法における目的関数では同時刻に同じスイッチをまたがる MPI 通信関数が複数あれば，通信の衝突が発生したとし，衝突したメッセージサイズに応じてその値を増加させた．また，評価実験で扱う対象並列計算機の計算ノードが 16 ノードであるため，提案手法，従来手法ともに全探索を行い，それぞれ最小となるランク配置の出力を行った．これにより，今回提案した目的関数によるそれぞれのランク配置に対する最大の通信時間の削減効果がわかる．各手法により出力されたランク配置に従い，プログラムの実行前，もしくは初期化の時などに計算ノードに対するランクの割り付けを行い，計算を実行，その通信時間の計測を行う．この時，提案手法のプログラムには，フェーズの最初に同期関数 MPI_Barrier を挿入する．

表 2: 通信時間の実測値

	recusive doubling	CG	umt2000
提案手法	43204	24303	60078
	(14.47)	(14.00)	(45.04)
従来手法	50515	28262	79131
	(0.00)	(0.00)	(27.61)
順配置	505150	28262	109326
	(0.00)	(0.00)	(0.00)

4.4 実験結果・考察

まず，表 2 にそれぞれのランク配置の時の通信時間の実測値，カッコ内に順配置に対する削減率を示す．本稿の提案手法により出力ランクされたランク配置での通信時間は recursive doubing で 43204usec, CG で 24303usec, umt2000 で 60078usec となり，順配置に対して，それぞれ約 14 %，14 %，45 %通信時間が削減された．従来手法により出力されたランク配置に対しても，それぞれ約 14 %，14 %，24 %の削減があった．従来手法では，umt2000 以外は順配置と同様の出力であった．recursive doubling や CG では，スイッチを跨ぐ通信の数を最小にするランク配置が順配置と一致するからである．一方，本手法では，recursive doubing や CG には通信の衝突に時間変化があるため，スイッチを跨ぐ通信の数が少々増えようともなるべく衝突をしないようなランク配置を選ぶことができるので，通信時間の削減につながったと考えられる．umt2000 では，ランクの通信先に偏りがあるため，その偏りに対応してお互いに通信のあるものは同一スイッチに接続する計算ノードに割り当てることで高速になる．これは，従来手法でも可能であるため，28 一方，umt2000 には通信の時間変化もあるため，本手法の場合は従来手法に対してさらに 24 %ほどの通信時間削減が行えている．

図 6 に, 各プログラムについて，従来手法を用いたランク配置による通信時間に対する提案手法を用いたランク配置による通信時間の比率 (previous work) 及び，順配置による通信時間に対する提案手法を

用いたランクは位置による通信時間の比率 (order rank) を示す．ただし，recursive doubling と CG は従来手法を用いたランク配置と順配置が一致したので，順配置との比率のみを示す．メッセージサイズ

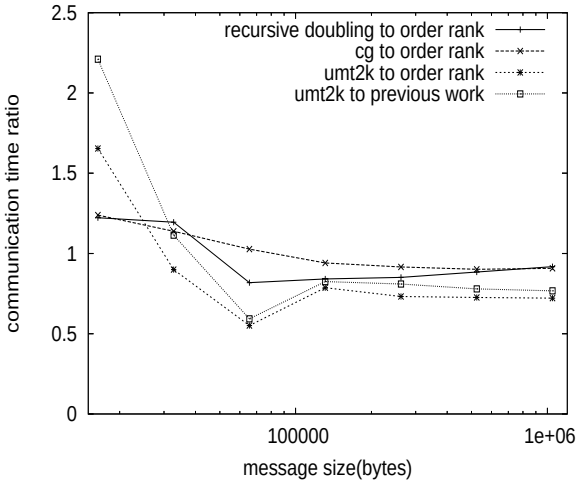

図 6: 通信時間の比率

が小さい時は同期関数の影響が大きいことまた，通信の衝突が起き難くなることから，提案手法を適用した方が通信時間が増加した．本実験の環境では，1 回の MPI_Barrier に要する時間は約 150usec である. umt2k は 16 フェーズあるので，同期通信だけで 2400usec の通信時間がかかる．このことからも本手法による効果が得られるのは，メッセージサイズが同期関数の所要時間に対して十分大きい場合である事がわかる. 本実験の環境では，メッセージサイズが 64kB を越えると同期関数による影響が無くなり，通信の衝突も起きやすくなる．このため，衝突の回避をするための提案手法を適用した方が通信時間が削減されたと考えれる．umt2000 において 64kB になると非同期通信を行う場合に約 25000usec ほどの通信時間が必要となってくるからである．これに対し提案手法では 14000usec ほどの通信時間で済む．

5 終わりに

通信タイミングを考慮したランク配置最適化の提案を行い，その有効性を示すための評価実験を行った．評価実験では本手法を適用することで順配置に対して最大 45 %程度，従来手法により出力されたランク配置に対して最大 24 %程度の通信時間の削減効果を示した．また，メッセージサイズが大きい場合は同期関数の挿入によるオーバヘッドも影響を上回る通信時間の削減効果があることが分かった．これにより，本手法の有用であることが分かった．

今後の課題は，提案手法の適用範囲がどの程度のものであるか調べるため，他の通信パターンや実アプリケーションでの評価実験を行う．次に大規模並列計算機での評価を行う．この時，現在はランク配置探索を全探索で行っているため，本手法に適した大規模並列計算機用のランク配置探索アルゴリズムを考える必要がある．

謝辞 本研究は,「ペタスケール・システムインターコネクト技術の開発」プロジェクト (文部科学省「次世代 IT 基盤構築のための研究開発」の研究開発領域「将来のスーパー コンピューティングのための要素技術の研究開発」（平成 17〜19 年度）の一つ) によるものである．

参考文献

[1] F. Ercal , J. Ramanujan , P. Sadayappan, “Task allocation onto a hypercube by recursive mincut bipartitioning,” Journal of Parallel and Distributed Computing, v.10 n.1, p.35-44, Sep. 1990

[2] T. Hatazaki, “Rank Reordering Strategy for MPI Topology Creation Function,” PVM/MPI ’98, LNCS 1497, pp.188-195,1998.

[3] Jesper Larsson Traff, “Implementing the MPI process topology mechanism,” in Conference on High Performance Networking and Computing, Proceedings of the 2002 ACM/IEEE conference on Supercomputing,in 2002.

[4] T. Agarwal, A.Sharma, L. V. Kale, “Topology-aware task mapping for reducing communication contention on large parallel machines,” Proceedings of IEEE International Parallel and Distributed Processing Symposium 2006.

[5] Gyan Bhanot, Alan Gara, Philip Heidelberger, Eoin Lawless, James Sexton, Robert Walkup, “Optimizing task layout on the Blue Gene/L supercomputer.” IBM J. Res. & Dev. 49, No. 2/3, 489-500, in 2005.

[6] “TOP500 Supercomputer Sites,” `http://www.top500.org/.`

[7] “The ASCI Purple Benchmark,” `http://www.llnl.gov/asci/purple/benchmarks/limited/code_list.html`

[8] “NAS Parallel Benchmark,” `http://www.nas.nasa.gov/Resources/Software/npb.html`