社団法人　情報処理学会　研究報告
IPSJ SIG Technical Report

2004－ＱＡＩ－11　(4)
2004／5／26

Gridにおけるアプリケーションの実行性能へネットワーク特性が及ぼす影響に関する検討

北辻　佳憲[†]　山崎　克之[†]　鶴　正人[††]　小出　洋[††]　尾家　祐二[††]

† 株式会社 KDDI 研究所　〒 356–8502 埼玉県上福岡市大原 2–1–15
†† 九州工業大学 情報工学部　〒 820–8502 福岡県飯塚市津川 680–4
E-mail: {kitaji,yamazaki}@kddilabs.jp, {tsuru,oie}@cse.kyutech.ac.jp, koide@ai.kyutech.ac.jp

あらまし　Grid コンピューティングのネットワーク環境では、多様なアプリケーションが同一のネットワークで並行に実行され、複数のアプリケーションの通信に対して、限られたネットワーク資源の効果的な割り当てが課題となっている。資源の割り当てによってネットワーク特性が変化すると、アプリケーションの通信がその影響を受けて実行性能が劣化する恐れがある。さらに、その劣化の大きさはアプリケーションによって異なると考えられる。そこで、Grid 上で動作可能なアプリケーションや分散コンピューティングで広く使われている Message Passing Interface (MPI) などを用いたプログラムのトラヒック特性を調査する。それによって、アプリケーションにおいて実行性能に最も大きな影響を与えるネットワーク特性（通信品質）が何かを考察する。
キーワード　Grid コンピューティング、ネットワーク特性、トラヒックフロー特性、実行性能、遅延時間

Study for the Influences of Network Characteristics to Application Performance in the Grid Environment

Yoshinori KITATSUJI[†], Katsuyuki YAMAZAKI[†], Masato TSURU[††], Hiroshi KOIDE[††], and Yuji OIE[††]

† KDDI R&D Laboratories, Inc.　Ohara 2–1–15, Kamifukuoka-shi, Saitama, 356–8502 Japan
†† Faculty of Computer Science and Systems Engineering, Kyushu Institute of Technology, 680–4, Kawatsu, Iizuka-shi, Fukuoka, 820–8502 Japan
E-mail: {kitaji,yamazaki}@kddilabs.jp, {tsuru,oie}@cse.kyutech.ac.jp, koide@ai.kyutech.ac.jp

Abstract　In the Grid environment, it is a key issue how limited network resources are allocated to communications of various applications which are paralleled to perform in the identical networks. The change of network characteristics caused by the resource allocation to an application may affect the communications of other applications, which may, in turn, degrade their process performance. Additionally, we suppose that the level of degradation are various and depends on the applications. In this paper we examine the traffic characteristics (communication patterns) of a typical Grid application or programs using Message Passing Interface commonly used in the distributed computing technology. Furthermore, we study which characteristics of the network highly influence the process performance of these applications.
Key words　Grid Computing, Network Characteristics, Traffic Flow Characteristics, Processing Time, Latency

1. はじめに

インターネットの高速化、計算機の高性能化に伴い、ネットワークに接続された多数の計算機を集約して大規模演算を行う、並列分散コンピューティング技術が驚くべき早さで発達している。近年、分散コンピューティングの WAN(Wide Area Network) 環境への対応が考慮されるようになってきており、インターネット全域に遍在する計算機を集約して構成された仮想的な高性能計算機を利用した広域並列分散処理のことを、Grid コンピューティングと呼ぶ [1], [2]。

Grid コンピューティングのネットワーク環境では、多様なアプリケーションが同一のネットワークで並行に実行され、複数のアプリケーションの通信に対して、限られたネットワーク資源の効果的な割り当てが課題となっている。データ転送時間が、

アプリケーションの実行性能に無視できないほど大きい影響を与えるため、資源の割り当てによってネットワーク特性が変化すると、アプリケーションの通信がその影響を受けて、実行性能が劣化する恐れがある。さらに、アプリケーションによってその劣化の大きさは異なると考えられる。

そこで、Grid 上で動作可能なアプリケーションである電磁構造連成解析や分散コンピューティングで広く使われている Message Passing Interface (MPI) [3] などを用いた各種プログラムのトラヒック特性を調査する。そして、遅延やクロストラヒックなどのネットワーク特性が各アプリケーションに与える影響を調査し、実行性能に最も大きな影響を与えるネットワーク特性（通信品質）が何かを考察する。

2. 分散コンピューティングの特徴

分散コンピューティングにおける並列プログラムの処理は、Task-Framing 型、と Work Flow 型に分類することができる [4]。Task-Framing 型は、まず処理に必要となる入力データが各計算機へ転送され、計算機はタスク毎に設定されている実験パラメータを用いて処理を実行し、処理結果を出力する。各タスクは多数の試行により構成されており、それぞれの試行は異なるパラメータを用いて処理される。各パラメータに関する処理はそれぞれ独立であるため、1 つのタスク内で扱うパラメータの量を調整する事により、アプリケーションの処理負荷を複数のタスクへ分割することが可能である。通常、Task-Framing 型処理では、アプリケーションの処理をタスクに分割し計算機へ割り当てる Master と割り当てられたタスクを処理する Slave で構成されされることが多く、Master/Slave 型と呼ばれることもある。一方、Work Flow 型は、一連の処理を複数の段階に分けてパイプライン処理を行う。各段階は 1 つあるいは複数のタスクで構成され、タスクの出力結果は次の段階を構成するタスクの入力データとして扱われる。

この 2 種類の並列プログラムの処理は、共にタスク間のデータ交換や処理時間の同期について考慮する必要がある。通常、アプリケーションの実行中にタスク間で交換すべきデータが発生し、このようなデータは受信側のタスクの処理に必要不可欠であるため、データの到着までタスクの処理はブロックされる。このブロック時間を短縮するためには、ネットワーク資源に対して即時性が求められる。さらに、分散コンピューティングの性能は、最後に実行したタスクの終了時刻に左右されるため、資源を有効利用するために、タスクがすべて終了するまでの時間が最も短くなるように資源の選択を行う必要がある [5]。

3. 調査環境および調査モデル

本章では、ネットワーク特性が分散コンピューティングに与える影響を調査するため、調査対象のアプリケーションについて説明し、そのアプリケーションを実行するネットワークの構成を示す。

3.1 調査対象のアプリケーションの特徴

本稿では、Grid 上で動作可能なアプリケーションとして以下に挙げるアプリケーションを調査対象とした。

表 1 アプリケーションの分散処理の分類

アプリケーション	分散処理の形態
電磁構造連成解析	Task-Framing 型
NQueen 問題	Task-Framing 型
Povray	Task-Framing 型
ジグゾーパズル問題	Task-Framing 型
タスクスケジューリング問題	Work Flow 型

- 電磁構造連成解析　強磁場中に設置された導電性構造物に生じる渦電流による電磁力が構造物に作用するとともに、構造物の振動により生じる速度起電力が渦電流に影響を与える連成現象を解析するプログラム
- **NQueen 問題**　将棋版の様な正方形のますに、一辺のますの数と同数の駒をますの縦、横、斜めの列に重ならないように配置する探索プログラム
- **Povray**　Persistence of Vision Raytracer 方式で有名な 3 次元画像描画アプリケーション [6] に、分散処理の改良が加えられたフリーのプログラム
- **ジグゾーパズル問題**　2001 年 ACM 国際大学対抗プログラミングコンテストアジア予選函館大会の問題 C の計算機用ジグゾーパズル問題 [7] に対して、ピースの大きさとアルファベッド数の可変およびピースの回転も考慮する拡張を行った問題を解くプログラム
- **タスクスケジューリング問題**　標準タスクグラフ集 [8] を題材として、リストスケジューリング、クリティカルパス法、空きメモリを考慮したクリティカルパス法を用いて実際にタスクをスケジューリングするプログラム。リストスケジューリングは実行可能なタスクが発生するとそれを空いている計算機に割り当てるアルゴリズム、クリティカルパス法はタスクの重みと依存関係から最も長い時間がかかるタスクフロー（クリティカルパス）を求め，このタスクフロー上にあるタスクを優先的に実行するアルゴリズム、空きメモリを考慮したクリティカルパス法はクリティカルパス法のタスク割り当ての際、計算機にタスク処理に必要なメモリの空きがあるかを考慮するアルゴリズム [9], [10]

各アプリケーションの分散処理の分類を表 1 に示す。本稿で調査対象とする Task-Framing 型アプリケーションは、Master-Slave 間で通信が行われ、Slave 間では直接通信が行われない。タスクスケジューリング問題では Java で記述されたライブラリが用いられ、電磁構造連成解析では、Fortran 言語の MPICH ライブラリ [11] が、他のアプリケーションでは C 言語の MPICH ライブラリが用いられている。

Task-Framing 型処理ではタスクが実行される際に入力データ・出力結果の転送が行われるため、アプリケーションの起動後や、終了前に多量なデータ転送が発生しすることが予想される。一方、Work Flow 型処理では、連続するタスクの前後に入力データ・出力結果のデータ転送が非同期で行われ、各計算機において不均一なトラヒックが発生することが予想される。

3.2 調査モデル

本稿ではネットワーク特性が分散コンピューティングに与え

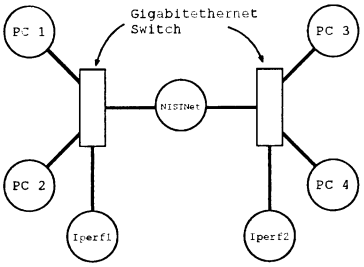

図 1　ネットワーク構成

表 2　各計算機の性能

Host	PC1, 2, 3, 4	NISTNet, Iperf1, 2
CPU	Dual Pentium4 3GHz	Xeon 3.06GHz
Memory	1GBytes	2 GBytes
NIC	Intel(R) PRO/1000	Intel(R) PRO/1000

る影響を調査するため、図 1 に示すネットワーク構成を用いてアプリケーションを実行し、様々なネットワーク状態における処理時間およびアプリケーションのトラヒック特性を計測した。

評価に使用した計算機の性能を表 2 に示す。なお、各アプリケーションは計算機 PC1 から PC4 の間で分散処理を行い、起動は PC1 から行う。Task–Framing 型アプリケーションでは PC1 において Master プロセスが動作し、PC1 を含め全ての計算機 (PC1〜PC4) において Slave プロセスが動作する。PC1, 2 と PC3, 4 の間には、通信遅延を挿入するために NIST Net [12] が動作するルータ (PC ルータ) を用いる。さらに、他のトラヒックの影響を調査するため、アプリケーションの動作中に外乱のトラヒックを挿入する計算機 iperf1 および iperf2 を各セグメントに接続した。各セグメントの計算機は、DELL 社製ギガビットイーサネットスイッチ (PowerConnect 5212) を用いて接続した。

計測は、PC1〜PC4 の送受信トラヒックを Finisar 社製 UTP タップを用いてキャプチャして解析を行った。PC1 と PC2 間、および、PC1 と PC3 の間の往復遅延時間の平均はそれぞれ 107 マイクロ秒および 712 マイクロ秒であった。

4. 通信特性の解析

本章では、前章で紹介した調査モデルを用いて、まずアプリケーションの基本性能を調査し、次に遅延およびクロストラヒックのネットワーク特性が分散コンピューティングの実行性能に与える影響を調査する。

4.1 アプリケーションの基本性能

まずはじめに、アプリケーションの基本性能を調べるため、往復遅延や外乱となるトラヒックを挿入せずに、アプリケーションを実行した場合のトラヒック特性を計測した。

4.1.1 通信コネクション

各アプリケーションの通信は TCP が用いられており、UDP は用いられていなかった。TCP コネクションの確立に用いられる待ち受けポート番号は表 3 のとおりであった。MPICH ライブラリおよび Java ライブラリとも、コネクション待ち受けのためのポート番号がある範囲で動的に選択されていることがわかった。

表 3　待ち受けポート番号

通信の種類	ポート番号
Task–Framing 型, Master	600, 33000–35000, 39000–41000, 54000–57000
Task–Framing 型, Slave	512, 54000–57000
Work Flow 型	8880, 36815, 43692, 55000–56000

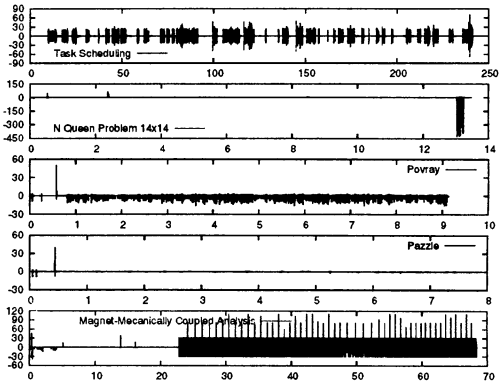

図 2　トラヒックの変化

図 2 に PC3 における送受信トラヒックの変化を示す。グラフは x 軸がアプリケーション開始後の経過時間 [秒] を示し、y 軸がトラヒックの 10ms ごとの平均速度 [Mbit/s] を示している。y 軸の正の値は PC3 への入りトラヒックを、負の値は出トラヒックを示している。Task–Framing 型アプリケーションのトラヒックの変化は、長時間に渡って定常的にトラヒックが転送される状態と、バースト的な転送が発生する状態があることがわかった。Work Flow 型のタスクスケジューリング問題では、3 つの分散基準で類似したトラヒックパターンを示したため、空きメモリを考慮したクリティカルパス法を基準とするスケジューリングについ示している。トラヒックは不均一であるが頻繁に転送されることがわかる。なお、PC2 および PC4 においてもトラヒックパターンの特徴はほぼ一致していた。

4.1.2 トラヒックフロー

つぎに、各アプリケーションのトラヒックフローについて調査した。本稿では、フローを計算機のアドレスおよびポート番号の組で区別され、コネクションの開始 (syn フラッグ) から終了 (fin フラグ) までの一連のパケットの集合と定義する。

Task–Framing 型アプリケーションでは、各アプリケーションで合計 50〜250 のフローが観測された。電磁構造連成解析では精度の高い演算を行うほどフロー数が増加し、NQueen 問題では正方配列の大きさが大きくなるほどフロー数が増加した。スケジューリング問題では、3 つの分散基準に大きな違いはなく、1 台の計算機が送受信するフローの合計が 1000 であった。

以下では、PC3 で送受されるトラヒックについて、フロー継続時間および発生間隔の分布を示す。Work Flow 型のタスクスケジューリング問題では、3 つの分散基準で類似した分布を示

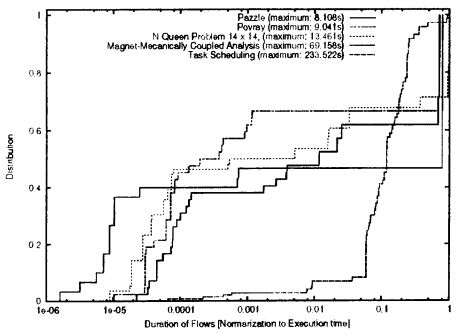

図 3　フロー継続時間の分布

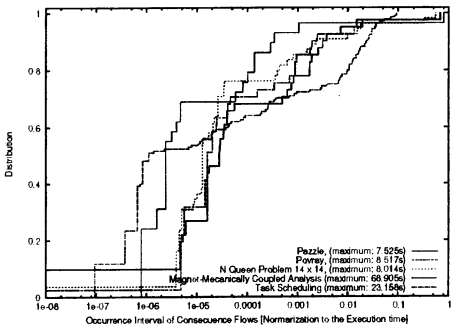

図 4　フロー発生間隔の分布

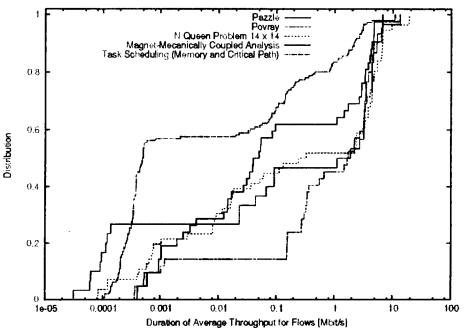

図 5　フローごとの平均スループットの分布

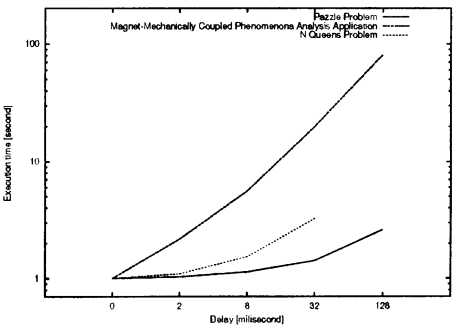

図 6　遅延による処理時間の変換

したため、空きメモリを考慮したクリティカルパス法を基準としたスケジューリングについて示す。

各アプリケーションのフロー継続時間の分布を図 3 に示す。x 軸は、アプリケーションの処理時間に対するフロー継続時間の比を示している。Task-Framing 型アプリケーションでは、半数前後のフローが非常に短い時間で終了するとともに、アプリケーションの実行時間に対して 7 割以上の長時間継続するフローも存在することがわかった。また、タスクスケジューリング問題では、Task-Framing 型アプリケーションに比べ、非常に短い時間で終了するフローの割合が少く、処理時間に対して長時間継続するフローが多いこともわかった。

各アプリケーションのフロー発生間隔の分布を図 4 に示す。x 軸は、アプリケーションの処理時間に対する発生間隔の比を示している。Task-Framing 型の分散処理を行うアプリケーションでは、多くの発生間隔が処理時間の 1/50 以下の値を示すが、全体の処理時間の約 5 割に相当する長い間隔が空く場合も観測された。タスクスケジューリング問題では、最長の間隔が処理時間の 1/10 となっていることがわかった。

各アプリケーションのフローの平均スループットの分布を図 5 に示す。x 軸は、平均スループットの実測値を示している。Task-Framing 型の分散処理を行うアプリケーションでは、全体の 3-5 割が 1Mbit/s 以上のスループットを示し、残り 5-7 割のフローが 100bit/s から 1Mbit/s のスループットに一様に分布している。それに対し、Work Flow 型のタスクスケジューリング問題では、全体の 5 割以上のフローが 1Kbit/s 未満のスループットを示している。Work Flow 型の分散処理は、Task-Framing 型に比べて対話型の通信の割り合いが多いことが想像される。

4.2　往復遅延を制限した場合の影響

本節では、図 6 の NISTNet が動作する計算機を用いて、通過するトラヒックに片道 1〜64 ミリ秒の往復遅延を挿入し、通信遅延が電磁構造連成解析、NQueen 問題、ジグゾーパズル問題の実行性能に与える影響を調査する。処理時間の計側は、アプリケーション自身が計測する場合にはその結果を用い、計測しない場合には time コマンドを用いて計測した。なお、計算機間の組み合わせを変えて実験を行ったが、実行性能に大きな違いは観測されず、遅延の増大によって特定の計算機間の通信が性能に大きく影響を与えことはないと考えられる。

図 6 は、アプリケーションの通信に往復遅延を挿入しなかった場合の処理時間を 1 としたときの、遅延挿入時の処理時間の比を示している。各値は 5 回の計測における平均値を示しており、往復遅延時間が大きくなるほど、実行性能の劣化が顕著となった。

各アプリケーションのフローごとの継続時間の分布は全てのフローが処理時間とほぼ等しい値を示し、トラヒックが比較的均一な電磁構造連成解析およびジグゾーパズル問題と、バースト性の高い N Queen 問題の間で特徴的な違いが現れなかった。また、各アプリケーションでは遅延の影響を受けるフロー (PC1 と PC3 の間および PC1 と PC4 の間) と遅延の影響を受けないフロー (PC1 と PC2 の間) の継続時間が、ともに全体処理時間とほぼ等しい値を示した。遅延の影響を受けた場合の電磁構造連成解析および N Queen 問題のフロー継続時間の分布

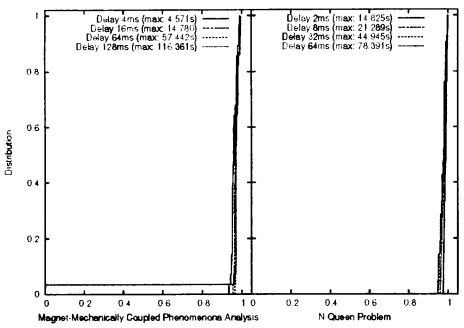

図 7　電磁構造連成解析および NQueen 問題のフロー継続時間の分布

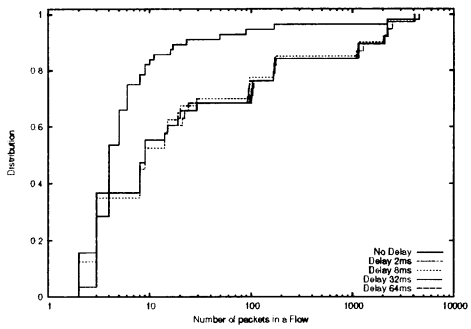

図 8　NQueen 問題のパケット数の分布の例

の例を図 7 に示す。x 軸は、アプリケーションの処理時間に対するフロー継続時間の比を示している。

- フロー数が 2〜5 割減少する
- 表 3 に示した 50000 番台の待ち受けポート番号が用いられなくなる
- パケット数が多いフローの数が増加する
- 平均パケット長が大きいフローが現れる

図 8 に、遅延があるときの N Queen 問題のフローごとのパケット数の分布の例を示す。遅延があるときはその大きさに関係なく、パケット数の少ないフローの割合が減少し、逆にパケット数の多い (数千パケットの) フローの割合が増えたことがわかる。このような変化は、電磁構造連成解析およびジグゾーパズル問題においても同様に観測された。

Work Flow 型アプリケーションであるタスクスケジューリング問題については、遅延環境における通信性能が安定して得られなかったため、更に調査を行う予定である。

4.3　クロストラヒックによる影響

アプリケーションの通信が、他のトラヒックによって受ける影響を調べるため、アプリケーションが動作中に回線の空き帯域を越えるトラヒックを流し、アプリケーションの実行性能を調査する。本節では、ある程度のトラヒックが発生する電磁構造連成解析を調査対象とした。計算機間に遅延がない状況では、電磁構造連成解析は約 40Mbit/s(IP ヘッダ含む) のトラヒックを発生させる。アプリケーションを実行中に、図 1 の、計算機 perf1 から perf2 へ 940Mbit/s の TCP または UDP のクロストラヒックを 1 秒間隔で 1 秒間流し、PC1 から PC3 へ流れるアプリケーショントラヒックの変化を観測した。このとき、IP ヘッダを含むクロストラヒックの速度は、TCP で 965.7Mbit/s、UDP で 959.1Mbit/s であった。図 9 に、TCP および UDP のクロストラヒックの影響を受けた電磁構造連成解析のトラヒックの変化を示す。グラフの x 軸はプログラム開始からの経過時間を、y 軸は 10ms ごとの平均速度を示している。

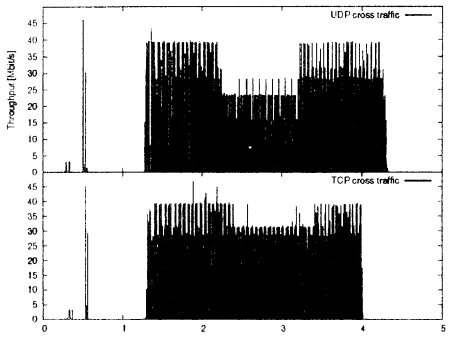

図 9　クロストラヒックの影響

TCP のクロストラヒックを流した場合、約 40Mbit/s の速度の電磁構造連成解析のトラヒックは約 32Mbit/s に減少し、クロストラヒックも 905–920Mbit/s(IP ヘッダを含めると 929.7–945.2Mbit/s) に減少した。同様の試験を 5 回繰り返したところ両者の減少はほぼ同じ範囲であった。このとき、電磁構造連成解析のトラヒックとクロストラヒックの両方にロスは発生しなかった。

UDP のクロストラヒックについても同じ試験を行ったところ、電磁構造連成解析のトラヒックは約 23Mbit/s に減少し、クロストラヒックの受信速度は 925-935Mbit/s(IP ヘッダを含めると 943.8–954.1Mbit/s) の間で変化し、0.47–1.4%のロスが観測された。

5.　考　　察

アプリケーションの通信にネットワーク資源を割り当る状況を考えた場合、トラヒックのポート番号からアプリケーションを特定することが考えられる。4.1.1 節の結果から、調査対象のアプリケーションのように、ポート番号が動的に割り当てられる状況では、アプリケーションを分類するルータあるいは計測装置がそのポート番号の範囲を予め知っていなければならない。

4.1.2 節の結果から、Work Flow 型のアプリケーションはフロー継続時間が長時間持続するフローの割合が多いことがわかった。タスクスケジューリング問題のように、頻繁にトラヒックを転送するアプリケーションでは、フローに基いた帯域予約がアプリケーションの良好な実行性能を実現できると考えられる。

4.2 節の実験結果から、調査対象としたアプリケーションは遅延の影響によって実行性能が大幅に低下した。このとき、フローの継続時間が処理時間にほぼ等しい値にまで延び、フロー数が減少し、パケットが多いフローが増加していた。これは、

データ交換の際 TCP コネクションが遅延によって長時間維持される間に、次のタスクの新たな入出力データが同一コネクションにバッファされ、同じコネクションが用いられたと考えられる。つまり、少量のデータを転送する多数の TCP コネクションを確立していたアプリケーションでは、遅延の影響によって確立される TCP コネクションが減少し、フローの一部はデータの転送量が増加したと考えられる。

アプリケーション開始の段階に Master から Slave へ割当られたタスクの情報が送信され、以後 Slave でタスクが次々に処理されるような Task-Framing 型の分散処理では、各 Slave で最後に処理されるタスクが終了した段階でそれまでに処理されたタスクの結果が送信されていれば、アプリケーションは良好な実行性能を示すと考えられる。遅延のある環境では、上述のように、1 つのタスクの処理の結果を送信し終る前に次のタスクが終了してその結果が蓄積されることが予想される。そのため、このような状況では、通信コネクションに蓄積されるデータを如何にまとめて送信できるかが課題と言え、以下に代表される遅延環境における TCP の広帯域伝送の手法が有効と思われる。

- パケットロス率をもとに送信レートを動的に制御する SABUL [13]、
- 輻輳回避モードにおける輻輳 window の制御を行う HSTCP [14] や Scalable TCP [15]
- 往復遅延を考慮した送信 window の制御を行う FAST [16]

4.3 節の結果から、アプリケーション実行中に TCP のクロストラヒックを流した場合、アプリケーショントラヒックとクロストラヒックの両方が減少し、UDP のクロストラヒックに比べてより多くの帯域が利用でき、結果として TCP のクロストラヒックの方が処理時間が短縮した。クロストラヒックは特定のパターンしか与えていないため、さらに調査が必要であるが、限られたネットワーク資源を複数の Grid アプリケーションで共有するような状況では、TCP や UDP などのプロトコルを基準にトラヒックを分類し、分類されるトラヒックタイプに対する資源割り当てが効果を発揮する可能性あると考えられる。

6. ま と め

本研究では、多様な Grid アプリケーションが同一のネットワークで並行に実行される環境において、複数のアプリケーションの通信に対して、限られたネットワーク資源の効果的な割り当てを検討するため、ネットワーク特性が分散コンピューティングの実行性能に与える影響を調査した。調査対象として、Task-Framing 型および Work Flow 型の分散処理を行うアプリケーションを用い、広域なネットワークを想定した環境において、帯域および外乱となるトラヒックのネットワーク特性がアプリケーションの処理時間に与える影響を調査した。

まず、アプリケーションの基本特性から、Work Flow 型のタスクスケジューリング問題は継続時間が比較的長いフローが多く、不均一ではあるが頻繁にトラヒックを発生させていた。そのため、フローに基づく帯域の予約が良好な実行性能を実現できると考えられる。

次に、往復遅延時間の変化が分散コンピューティングに与える影響を調査したところ、対象としたアプリケーションでは、往復遅延時間が大きくなるにつれて実行性能が著しく悪化することがわかった。このとき、計算機間の往復遅延が一致していなくても、全ての計算機間の通信コネクションが同様に長時間持続し、パケット数が少ないフローは減少し、フローの一部はパケット数が増加することがわかった。

次に、ネットワークを流れる他のトラヒックが分散コンピューティングに与える影響を調査したところ、クロストラヒックが TCP の場合は、対象アプリケーションの通信が TCP であることから双方のトラヒックが減少し、UDP のクロストラヒックの場合よりも多くの帯域を確保でき、良好な実行性能が得られた。限られたネットワーク資源を複数の Grid アプリケーションで共有するような状況では、TCP や UDP などのプロトコルを基準にトラヒックを分類し、分類されるトラヒックタイプに対する資源割り当てが効果を発揮する可能性があることを示した。

謝辞 本研究の一部は、文部科学省における超高速コンピュータ網形成プロジェクト（NAREGI）の支援を受けている。ここに記して謝意を表す。

文　献

[1] I. Foster and C. Kesselman: The GRID Blueprint for a New Computing Infrastructure, Morgan Kaufmann Publishers, 1998.
[2] I. Foster, C. Kesselman, and S. Tuecke, "The Anatomy of the Grid," International Journal of Supercomputer Applications, 15(3), 2001.
[3] Message Passing Interface Version 2, Message Passing Interface Forum, http://www-unix.mcs.anl.gov/mpi/ .
[4] R. Buyya: High Performance Cluster Computing: Programming and Applications, Volume 2, Prentice Hall PTR, 1999
[5] 西繁則, 山本寛, 池永全志, 二保知也, 尾家祐二, "ネットワーク特性が分散コンピューティングに与える影響," 電子情報通信学会 技術研究報告, NS2003-364, IN2003-319, pp.361-366, 沖縄県名護市, 2004 年 3 月 3 日〜5 日.
[6] Persistence of Vision Raytracer, http://www.povray.org/ .
[7] ICPC Asia Regional Contest, Japan Hakodate, http://www.fun.ac.jp/icpc/ .
[8] Kasahara Laoratory, Waseda University, http://www.kasahara.elec.waseda.ac.jp/schedule .
[9] 谷口慶介, 小出洋, " タスクスケジューリング自動化クラスにおけるタスク間通信の効率化," 情報処理学会 火の国情報シンポジウム 2004, 大分, 2004 年 3 月 5 日〜6 日.
[10] H. Koide, and Y. Oie, "A New Task Scheduling Method for Distributed Programs which Require Memory Management in Grids," Proc. Proposed SAINT2004 Workshop 8: High Performance Grid Computing and Networking, pp.666-676, Tokyo, 2004.
[11] MPICH – A Portable Implementation of MPI, http://www-unix.mcs.anl.gov/mpi/mpich/ .
[12] NIST Net Home Page, http://snad.ncsl.nist.gov/nistnet/ .
[13] Y. Gu, X. Hong, M. Mazzucco, and R.L. Grossman, "SABUL: A High Performance Data Transfer Protocol," http://www.rgrossman.com/pdf/sabul-hpdtp-11-02.pdf .
[14] HighSpeed TCP, http://www.icir.org/floyd/hstcp.html .
[15] Scalable TCP, http://www-lce.eng.cam.ac.uk/ ctk21/scalable/ .
[16] Fast AQM Scalable TCP, http://netlab.caltech.edu/FAST/.