社団法人　電子情報通信学会
THE INSTITUTE OF ELECTRONICS,
INFORMATION AND COMMUNICATION ENGINEERS

信学技報
TECHNICAL REPORT OF IEICE.

# データ転送プロトコル GridFTP の自動パラメータ設定機構の提案

伊藤　建志† 　大崎　博之† 　今瀬　真†

† 大阪大学 大学院情報科学研究科
〒 565-0871 大阪府吹田市山田丘 1-5
E-mail: †{t-itou,oosaki,imase}@ist.osaka-u.ac.jp

あらまし　近年、地理的に分散した計算機資源をネットワークにより接続することにより、計算機資源の有効利用を図るとともに、大規模な科学技術計算を可能にする、グリッドコンピューティングが注目を浴びている。グリッドコンピューティングでは、大容量のファイルを転送するために GridFTP と呼ばれるデータ転送プロトコルが用いられている。GridFTP は、既存の TCP の問題点を解消するため、複数の TCP コネクションを並列に確立する、GridFTP サーバと GridFTP クライアント間で TCP ソケットのバッファサイズを動的に交渉するといった機能を持つ。本稿では、GridFTP の定常状態解析の結果をもとに、GridFTP の自動パラメータ設定機構を提案し、その有効性をシミュレーション実験により明らかにする。
キーワード　グリッドコンピューティング、GridFTP、TCP (Transmission Control Protocol)、パラメータ設定

# On Automatic Parameter Configuration Mechanism for Data Transfer Protocol GridFTP

Takeshi ITOU†, Hiroyuki OHSAKI†, and Makoto IMASE†

† Graduate School of Information Science and Technology, Osaka University
1-5 Yamadaoka, Suita, Osaka, 565-0871 Japan
E-mail: †{t-itou,oosaki,imase}@ist.osaka-u.ac.jp

**Abstract**　In recent years, Grid computing has got a lot of attention. Grid computing is a concept of connecting geographically-distributed computational resources via networks, which is expected to realize efficient usage of computational resources and large-scale scientific and engineering computation. In Grid computing, a data transfer protocol called GridFTP has been used for large file transfer. GridFTP has several features for solving problems of the existing TCP. First, multiple TCP connections can be established in parallel. Second, the TCP socket buffer size can be negotiated between GridFTP server and client. In this paper, we propose an automatic parameter configuration mechanism of GridFTP based on our steady state analysis, and show its effectiveness through simulation experiments.
**Key words**　Grid Computing, GridFTP, TCP (Transmission Control Protocol), Parameter Tuning

## 1 はじめに

近年、ネットワークの大容量化・広域化が急速に進んでいる。実験的な広域ネットワークでは、ボトルネックリンクの帯域が数 Tbps、エンド–エンド間の伝搬遅延は数百 msec にも達する。一方、ネットワークを介して、地理的に分散した複数の計算機資源を接続する、グリッドコンピューティングが注目を浴びている [1]。グリッドコンピューティングにより、これまで十分に利用されていなかった計算機資源を有効に活用するとともに、大規模な科学技術計算が可能になることが期待されている。しかし、グリッドコンピューティングを用いて大規模な科学技術計算を実現するためには、ネットワークを介して計算機間で膨大なファイルを転送する必要がある。このため、大容量のデータを効率的に転送できる、高速なデータ転送プロトコルへの要求が高まりつつある。

現在のインターネットでは、データ系トラヒックの転送に TCP (Transmission Control Protocol) が広く用いられている [2]。TCP にはさまざまなバージョンが存在するが、現在のインターネットで最も広く普及しているのは、TCP バージョン Reno (TCP Reno) およびその派生バージョンである [3]。TCP Reno は、輻輳

回避フェーズにおいて、AIMD (Additive Increase Multiplicative Decrease) 型のウィンドウフロー制御を行うことにより、ネットワークへのパケット転送レートを調節している。TCP Reno は 1 ラウンドトリップ時間ごとにウィンドウサイズを 1 つ上昇させるが、ネットワーク中でのパケット棄却を検出した時にはウィンドウサイズを半減させる。このため、ネットワークの帯域遅延積が大きい場合に、TCP Reno は少数のパケット棄却によってスループットが大きく低下する。このため、ネットワークのパケット棄却率が高い環境下において、高いスループットを維持することが困難であるといった問題などが指摘されている。

グリッドコンピューティングでは、大容量のデータを効率的に転送するためのプロトコルとして、GridFTP が提案されている [4–6]。GridFTP は、既存の TCP の問題点を解消することを目的として設計されており、さまざまな機能が追加されている。例えば、複数の TCP コネクションによる並列データ転送や、TCP ソケットバッファサイズの自動交渉といった機能を持つ。GridFTP の有効性は、並列 TCP コネクション数などの制御パラメータの設定に大きく依存することが知られている。しかし、GridFTP の制御パラメータをどのように設定すれば良いかに関して、これまで十分な検討がなされていない。例えば、GridFTP のプロトコル体系には、GridFTP のサーバ–クライアント間で並列 TCP コネクション数を指定するコマンドが含まれているが、並列 TCP コネクション数をどのように決定すれば良いかは、GridFTP のプロトコルではまったく規定されていない。

これまでに GridFTP の制御パラメータの設定方法に関する研究はいくつか行われている [7, 8]。文献 [7] では、GridFTP サーバ–クライアント間で帯域遅延積を計測することにより、必要な TCP ソケットバッファサイズを決定する手法を提案している。これは、GridFTP の拡張ブロックモードにおける SBUF コマンドを拡張することにより実現されている。文献 [8] では、TCP の流体近似モデルを用いることにより、最適な並列 TCP コネクション数および TCP ソケットバッファサイズを導出している。しかし、文献 [8] の解析結果を用いて、GridFTP の並列 TCP コネクション数を最適化するためには、TCP コネクションのラウンドトリップ時間や、ネットワークのパケット棄却率および利用可能帯域を事前に知る必要がある。このため、文献 [8] の解析結果を、現実の GridFTP にそのまま適用することはできない。

そこで本稿では、文献 [8] の解析結果を利用することにより、GridFTP における並列 TCP コネクション数の最適化手法 (自動パラメータ設定機構) を提案する。提案する手法は、まず、GridFTP クライアントにおいて、ネットワークの状態 (GridFTP のグッドプットやラウンドトリップ時間など) を計測する。次に、これらの情報をもとに並列 TCP コネクション数を調整するといったものである。本稿では、並列 TCP コネクション数の調整方法が異なる、3 種類の動作モード (MI モード、MI+ モード、AIMD モード) を考える。さらに、シミュレーション実験により、提案する自動パラメータ設定機構の性能評価を行う。その結果、提案する自動パラメータ設定機構により GridFTP の性能が大幅に向上することを示す。

本稿の構成は以下の通りである。まず、2 章において GridFTP の概要と GridFTP の特徴的な機能の 1 つである並列データ転送について述べる。3 章では、我々が提案する自動パラメータ設定機構の設計方針と基本的なアイディアを述べた後、アルゴリズムの詳細を説明する。4 章では、シミュレーション実験により自動パラメータ設定機構の性能評価を行い、その有効性を示す。最後に 5 章において、本稿のまとめと今後の課題を述べる。

## 2 GridFTP

GridFTP は、グリッドコンピューティングにおいて、大容量のデータを効率的に転送するために設計されたデータ転送プロトコルである。GridFTP は、現在広く用いられている FTP (File Transfer Protocol) [9–11] を拡張したものであり、現在、GGF (Global Grid Forum) [12] において標準化が進められている。GridFTP は、トランスポート層の通信プロトコルとして TCP を使用するが、TCP のさまざまな問題点を解消するように設計されている。例えば、GridFTP には既存の FTP に対して、以下のような機能が追加されている。TCP ソケットバッファサイズの自動交渉、並列データ転送、データ転送の第三者制御、部分ファイル転送、セキュリティ、信頼性のあるデータ転送である [6]。これら GridFTP 固有の機能の多くは、拡張ブロックモードと呼ばれる新しい転送モードによって実現されている [5]。現在、グリッドコンピューティングの代表的なミドルウェアである Globus Toolkit [13] では、GridFTP バージョン 1 (GridFTP v1) に準拠した、GridFTP サーバおよび GridFTP クライアントが実装されている。ただし、この GridFTP の実装では、TCP ソケットバッファサイズの自動交渉機能は実装されておらず、並列データ転送に用いる並列 TCP コネクション数も利用者が手動で設定しなければならない。また GGF では、GridFTP v1 の問題点に関する議論も行われており、それらの問題点を解決する、GridFTP バージョン 2 (GridFTP v2) の検討も進められている。GridFTP v2 では、GridFTP v1 の拡張ブロックモードにおいて、データ転送が単方向という制限が解消され、またデータ転送中にデータチャンネルを動的に確立・切断できるなど、さまざまな機能が追加されている。しかし、並列 TCP コネクションなどの GridFTP の制御パラメータをどのように設定すれば良いかについては、GridFTP v2 においても検討がなされていない。

### 2.1 並列データ転送

GridFTP では、OPTS RETR コマンドを用いることにより、複数の TCP コネクションを並列に確立することができる (図 1)。これにより、単一のファイルを、単一のサーバから、複数の TCP コネクションを通して転送することができる。複数の TCP コネクションを集約することにより、単一の TCP コネクションを用いる場合に比べて、より高いスループットが期待できる [14]。

これは、以下の理由によって説明できる。(1) 複数の TCP コネクションを集約することにより、TCP の輻輳回避フェーズにおいて、競合する他の TCP コネクションよりも、より大きな帯

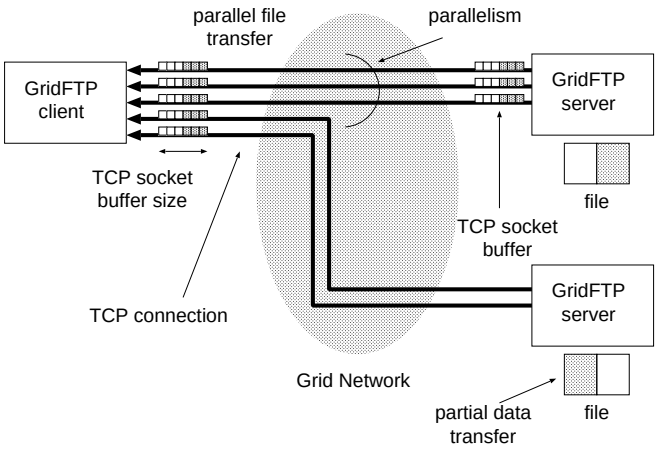


図 1　GridFTP における並列データ転送
Fig. 1　Parallel data transfer in GridFTP

域を利用できる [15]。これは、TCP の輻輳回避フェーズでは、AIMD 型のウィンドウフロー制御が行われており、パケット棄却率が小さいネットワークでは、複数の TCP コネクションを集約することにより有利にデータ転送を行うことができるからである。(2) 複数の TCP コネクションを集約することにより、1 ファイルの転送に利用できる TCP ソケットバッファサイズの総量が大きくなる。これは、$N$ 本の TCP コネクションを集約することにより、$N$ 倍の TCP ソケットバッファサイズを利用できるからである。(3) 複数の TCP コネクションを集約することにより、TCP のスロースタートフェーズにおける、転送レートの立ち上がりが速くなる。スロースタートフェーズでは、1 ラウンドトリップ時間ごとに、輻輳ウィンドウが 2 倍になる。このため、$N$ 本の TCP コネクションを集約することにより、転送レートの立ち上がりが $N$ 倍になる。

しかし、集約する TCP コネクション数 $N$ が大きすぎると、以下のような理由によりスループットが低下する。(1) TCP コネクションあたりのウィンドウサイズが小さくなり、タイムアウトが頻繁に発生してしまう。(2) サーバにおいて、TCP プロトコルスタックの処理に要するオーバヘッドが大きくなる。このため、ネットワークの状況に応じて、集約する TCP コネクション数 $N$ の最適値を決定する必要がある。しかし、GridFTP において並列 TCP コネクション数 $N$ をどのように決定するかについて、これまで十分な検討は行われておらず、未解決の問題のままである。

## 3　GridFTP の自動パラメータ設定機構

### 3.1　設 計 方 針

まず、GridFTP の自動パラメータ設定機構を設計するための基本方針を述べる。

GridFTP の自動パラメータ設定機構を設計する際には、既存の GridFTP との互換性を実現することが非常に重要である。GridFTP は Globus Toolkit に実装され、現在急速に普及が進んでいる。すでに多数の GridFTP サーバが稼働しているため、自動パラメータ設定機構は、GridFTP クライアント側で実現することが望ましい。また、既存の GridFTP サーバとの相互接続性を実現するために、既存の GridFTP プロトコルを変更せずに、自動パラメータ設定機構を実現することが望ましい。自動パラメータ設定を GridFTP クライアント側で行う場合、GridFTP における第三者転送において自動パラメータ設定を実現することは困難である。しかし、ほとんどの転送は GridFTP サーバ–クライアント間で行われるため、大きな問題にはならないと考えられる。

次に、GridFTP の自動パラメータ設定機構が、グリッドコンピューティング環境へ容易に導入できることが望ましい。一般に、グリッドコンピューティングの大きな特徴の一つとして、グリッドを構成する計算機やネットワークが不均一であるという点が挙げられる。このため、自動パラメータ設定機構は、さまざまな計算機環境やネットワーク環境においても動作することが求められる。このため、グリッドのミドルウェア層において、自動パラメータ設定機構を実現することが望ましい。つまり、自動パラメータ設定機構は、計算機上で稼働しているオペレーティングシステムや、搭載しているネットワークインターフェースなどの固有の機能を利用しないことが重要である。現在、ほとんどの計算機では、TCP バージョン Reno をベースにしたトランスポート層通信プロトコルを利用している。このため、トランスポート層通信プロトコルが TCP バージョン Reno であることを仮定するのは妥当であると考えられる。

### 3.2　基本的なアイディア

文献 [8] において、定常状態における GridFTP のグッドプットが近似的に次式のように導出されている。

$$G \simeq \min\left(\frac{N\,W}{R}, \frac{N\,(1-p^*)}{2\,R}\left(-3+\frac{\sqrt{6+21\,p^*}}{\sqrt{p^*}}\right)\right) \quad (1)$$

$$p^* \simeq \left(-2+\frac{2\,B\,R}{N}+\frac{2}{3}\left(\frac{B\,R}{N}\right)^2\right)^{-1} \quad (2)$$

ここで、$G$ は GridFTP のグッドプット、$N$ は並列 TCP コネクション数、$W$ は各 TCP コネクションあたりの TCP ソケットバッファサイズ、$B$ はボトルネックリンクの帯域、$R$ は TCP コネクションのラウンドトリップ時間、$p^*$ はネットワーク中のパケット棄却率である。式 (1) より、GridFTP のグッドプット $G$ は、並列 TCP コネクション数 $N$ に関して上に凸の関数となることが分かる。このため、並列 TCP コネクション数 $N$ は、GridFTP のグッドプット $G$ を最大化するように選択すれば良いことが分かる。しかし、式 (1) から分かるように、最適な $N$ を決定するためには、ボトルネックリンクの帯域 $B$ や TCP コネクションのラウンドトリップ時間 $R$ などが既知でなければならない。本稿では、式 (1) の解析結果を利用することにより、GridFTP クライアント側で並列 TCP コネクション数を自動的に設定する機構を提案する。

基本的なアイディアは、GridFTP クライアント側において、ネットワークの状態 (GridFTP のグッドプットおよびラウンドトリップ時間) を計測しながら、式 (1) を最大化する並列 TCP コネクション数 $N$ を探索するというものである。式 (1) より、

並列 TCP コネクション数 $N$ が小さすぎる状況では、GridFTP のグッドプットが $NW/R$ で制限されることがわかる。このため、並列 TCP コネクション数 $N$ を固定して、転送したいファイルの一部 (以下、チャンクと呼ぶ) の転送を行い、GridFTP クライアント側でグッドプット $G$ およびラウンドトリップ時間 $R$ を計測する。チャンクの転送終了後に、$G$ と $NW/R$ の関係を調べることにより、現在の並列 TCP コネクション数 $N$ が最適な値よりも大きいかどうかを判定することができる。この結果をもとに、並列 TCP コネクション数 $N$ を更新し、新たなチャンクの転送を行う。このような手順を繰り返すことにより、GridFTP のグッドプットを最大化するように、並列 TCP コネクション数を自動的に設定することが可能となる。

### 3.3 ネットワーク状態の計測方法

まず、GridFTP クライアントにおいて、ネットワークの状態をどのように計測するかを説明する。ミドルウェア層では、GridFTP のグッドプット $G$、TCP ソケットバッファサイズ $W$、ラウンドトリップ時間 $R$ を計測することが可能である。

GridFTP のグッドプットは、拡張ブロックモード [5] を用いて、転送したいファイルの一部 (チャンク) を転送し、その転送に要した時間とチャンクの大きさから計算することができる。具体的には、GridFTP の拡張ブロックモードを用いて、固定長のチャンクを ERET コマンドもしくは ESTO コマンドを用いて転送し、その時の応答時間 $T$ を計測する。ERET コマンドもしくは ESTO コマンドの応答は、チャンクの転送が完了した時点で返される [5] ため、転送したチャンク $X$ の大きさと、その時の応答時間 $T$ から、GridFTP のグッドプットを $G = X/T$ のように計算できる。

GridFTP クライアント側の TCP ソケットバッファサイズは、ソケット API を用いて容易に取得することができる。GridFTP には、ソケットバッファサイズを静的に設定する機能 (SBUF コマンド) [5] があるため、これを利用することによりサーバ側の TCP ソケットバッファサイズを取得する (正確には、GridFTP クライアントが指定した値に設定する) ことが可能である。GridFTP クライアントおよび GridFTP サーバの TCP ソケットバッファサイズのうち、小さい方の値が式 (1) における $W$ となる。

ラウンドトリップ時間 $R$ は、GridFTP の制御チャネルにおけるコマンドの応答時間を計測することにより求めることができる。具体的には、GridFTP クライアントから、GridFTP サーバ側に対して USER/PASS/SITE/FEAT/TYPE/MODE/SIZE/OPTS/PBSZ 等のコマンドを実行した時に、その応答時間 $R_i$ を計測する。これらのコマンドの処理は、GridFTP サーバ側において瞬時に完了すると考えられる。このため、ラウンドトリップ時間 $R$ は、これらのコマンドの応答時間によって近似できると考えられる。具体的には、ラウンドトリップ時間 $R$ の値として、過去 $M$ サンプルの応答時間の平均値 $R = \Sigma_{i=1}^{M} R_i/M$ を用いる。

上記のように、GridFTP の制御パケットやデータパケットを利用することにより、ネットワークに負荷を与えないパッシブな手法で計測が可能である。

一方、式 (1) に含まれる変数のうち、ボトルネックリンクの帯域 $B$ およびネットワーク中でのパケット棄却率 $p^*$ を GridFTP クライアント側で計測するのは困難である。現状では、ボトルネックリンクの帯域 $B$ やパケット棄却率 $p^*$ をミドルウェア層からパッシブな手法で計測することは困難である。計算機上で稼働しているオペレーティングシステムや、搭載しているネットワークインターフェースに固有の機能を使用すれば、ボトルネックリンクの帯域 $B$ やパケット棄却率 $p^*$ の計測も不可能ではないが、グリッドコンピューティング環境への導入容易性が損われてしまう。ミドルウエア層で、計測用の UDP/ICMP パケットを送出するというアクティブな手法を用いることによって、ボトルネックリンクの帯域 $B$ やパケット棄却率 $p^*$ を計測することも不可能ではないが、既存の GridFTP との互換性が損われてしまう。

### 3.4 並列 TCP コネクション数の調整方法

ネットワークの状態を計測した後、GridFTP のグッドプットを最大化するようにどのように並列 TCP コネクション数 $N$ を調整するかを説明する。本稿で提案する自動パラメータ設定機構は、転送したいファイルの一部をチャンク単位で転送し、その時の GridFTP のグッドプット $G$、TCP ソケットバッファサイズ $W$、ラウンドトリップ時間 $R$ を計測する。これらの情報をもとに、並列 TCP コネクション数 $N$ を調整するが、並列 TCP コネクション数 $N$ の調整方法として、以下に示すような 3 種類の動作モードを考える。

- MI (Multiplicative Increase) モード

MI モードは、並列 TCP コネクション数 $N$ を小さな値から開始し、TCP ソケットバッファサイズが GridFTP のグッドプットのボトルネックとならなくなるまで、並列 TCP コネクション数 $N$ を乗算的に増加させるモードである。以下では、並列 TCP コネクション数の初期値を $N_0$ とし、乗算増加量を $\gamma(>1)$、制御パラメータを $\eta\ (0 < \eta \leqq 1)$ と表記する。MI モードのアルゴリズムは以下の通りである。

( 1 ) 並列 TCP コネクション数 $N$ を初期化。

$$N \leftarrow N_0 \tag{3}$$

( 2 ) 固定長のチャンクを転送し、その時の GridFTP グッドプット $G$ およびラウンドトリップ時間 $R$ を計測する。

( 3 ) 以下の不等式が成立すれば、TCP ソケットバッファサイズがボトルネックと判定し、ステップ (4) へ進む。成立しなければアルゴリズムを終了する。

$$G > \eta \times \frac{NW}{R} \tag{4}$$

( 4 ) 並列 TCP コネクション数 $N$ を次式のように増加させ、ステップ (2) に戻る。

$$N \leftarrow \gamma \times N \tag{5}$$

- MI+ (Multiplicative Increase Plus) モード

MI+ モードでは、並列 TCP コネクション数 $N$ を小さな値か

ら開始し、TCP ソケットバッファサイズがボトルネックとならなくなるまで、並列 TCP コネクション数 $N$ を乗算的に増加させる。TCP ソケットバッファサイズがボトルネックではないと判断した段階で、定常状態解析の結果をもとに並列コネクション数 $N$ を最適化するモードである。以下では、並列 TCP コネクション数の初期値を $N_0$ とし、乗算増加量を $\gamma\ (>1)$、制御パラメータを $\eta\ (0<\eta\leq 1)$ と表記する。MI+ モードのアルゴリズムは以下の通りである。

( 1 ) 並列 TCP コネクション数 $N$ を初期化。

$$N \leftarrow N_0 \tag{6}$$

( 2 ) 固定長のチャンクを転送し、その時の GridFTP グッドプット $G$ およびラウンドトリップ時間 $R$ を計測する。

( 3 ) 以下の不等式が成立すれば、TCP ソケットバッファサイズがボトルネックと判定し、ステップ (4) へ進む。成立しなければステップ (5) へ進む。

$$G > \eta \times \frac{N\,W}{R} \tag{7}$$

( 4 ) 並列 TCP コネクション数 $N$ を次式のように増加させ、ステップ (2) に戻る。

$$N \leftarrow \gamma \times N \tag{8}$$

( 5 ) 式 (1) の右辺第二項を最大化する、並列 TCP コネクション数 $N$ を数値的に計算し、並列 TCP コネクション数 $N$ をその値に設定し、アルゴリズムを終了する。

- AIMD (Additive Increase and Multiplicative Decrease) モード

AIMD モードでは、並列 TCP コネクション数 $N$ を小さな値から開始し、TCP ソケットバッファサイズがボトルネックであれば、並列 TCP コネクション数 $N$ を加算的に増加させる。そうでなければ、並列 TCP コネクション数 $N$ を乗算的に減少させるというモードである。並列 TCP コネクション数の初期値を $N_0$ とし、加算増加量を $\alpha\ (>1)$、乗算減少量を $\beta\ (0<\beta<1)$、制御パラメータを $\eta\ (0<\eta\leq 1)$ と表記する。AIMD モードのアルゴリズムは以下の通りである。

( 1 ) 並列 TCP コネクション数 $N$ を初期化。

$$N \leftarrow N_0 \tag{9}$$

( 2 ) 固定長のチャンクを転送し、その時の GridFTP グッドプット $G$ およびラウンドトリップ時間 $R$ を計測する。

( 3 ) 以下の不等式が成立すれば、TCP ソケットバッファサイズがボトルネックと判定し、ステップ (4) へ進む。成立しなければステップ (5) へ進む。

$$G > \eta \times \frac{N\,W}{R} \tag{10}$$

( 4 ) 並列 TCP コネクション数 $N$ を次式のように増加させ、ステップ (2) に戻る。

$$N \leftarrow N + \alpha \tag{11}$$

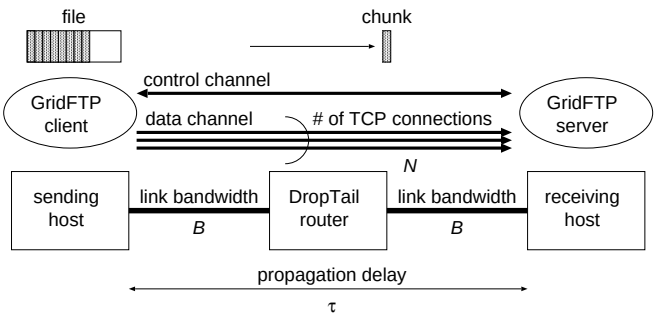


図 2　シミュレーションで用いたネットワークのトポロジ
Fig. 2　Network topology used in simulation

( 5 ) 並列 TCP コネクション数 $N$ を次式のように減少させ、ステップ (2) に戻る。

$$N \leftarrow N(1-\beta) \tag{12}$$

## 4　シミュレーション

本章では、シミュレーション実験により、提案する自動パラメータ設定機構の有効性を評価する。シミュレーションで用いた、ネットワークのトポロジを図 2 に示す。GridFTP サーバと GridFTP クライアントが単一のルータを介して接続されているというトポロジである。GridFTP クライアントから GridFTP サーバに対して、10 [Gbyte] のファイルを転送した。提案する 3 種類の自動パラメータ設定機構を用いて、並列 TCP コネクション数を動的に変化させた。

シミュレーションには ns-2 (version 2.27) を用いた。なお、特に断りのない限り、シミュレーションでは以下のようなパラメータを用いた。ボトルネックリンクの帯域 $B = 500$ [Mbit/s] ネットワークの伝搬遅延 $\tau = 100$ [ms]、TCP ソケットバッファサイズ $W = 1$ [Mbyte]、TCP のパケット長 1500 [byte]、ラウンドトリップ時間の平均を計算するサンプル数 $M = 10$、並列 TCP コネクション数の初期値 $N_0 = 1$、チャンクサイズ $X = N \times 100$ [Mbyte]、制御パラメータ $\eta = 0.9$、MI モードおよび MI+ モードの乗算増加量 $\gamma = 2.0$、AIMD モードの加算増加量 $\alpha = 1.0$、AIMD モードの乗算減少量 $\beta = 0.5$。

図 3 および図 4 に、それぞれ GridFTP のグッドプット $G$ の時間的変動、および自動パラメータ設定機構によって設定された並列 TCP コネクション数 $N$ の時間的変動をそれぞれ示す。これらの図では、10 [Gbyte] のファイル転送が終了するまで、チャンク転送ごとに計測した GridFTP のグッドプットおよび設定された並列 TCP コネクション数をプロットしている。

図 3 および図 4 より、MI モード、MI+ モードともに良好な性能を示していることが分かる。MI モード、MI+ モードともに、転送開始後に並列 TCP コネクション数 $N$ を指数的に増加させ、転送開始後 120[s] 程度で並列 TCP コネクション数 $N$ がそれぞれ 16 および 11 に収束している。この時の、GridFTP のグッドプットは、それぞれ 479 [Mbit/s] (MI モード)、454 [Mbit/s] (MI+ モード) であった。これにより、MI モードもしくは MI+ モー

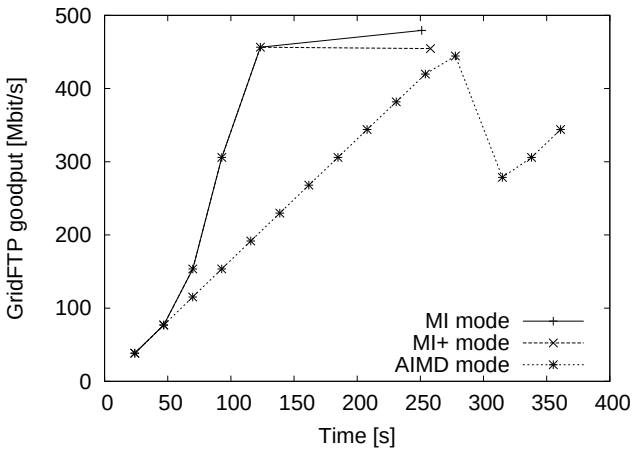


図 3　GridFTP のグッドプット $G$ ($B = 500$ [Mbit/s], $\tau = 200$ [ms], $W = 1$ [Mbyte])

Fig. 3　GridFTP goodput $G$ ($B = 500$ [Mbit/s], $\tau = 200$ [ms], $W = 1$ [Mbyte])

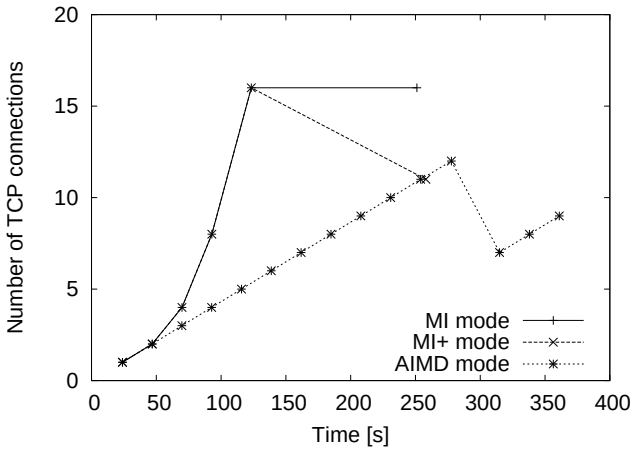


図 4　並列 TCP コネクション数 $N$ ($B = 500$ [Mbit/s], $\tau = 200$ [ms], $W = 1$ [Mbyte])

Fig. 4　Number of TCP connections $N$ ($B = 500$ [Mbit/s], $\tau = 200$ [ms], $W = 1$ [Mbyte])

ドを用いることにより、GridFTP がネットワークの利用可能帯域を十分に利用できていることが分かる。また、AIMD モードでは、並列 TCP コネクション数 $N$ が線形に増加している、MI モードや MI+ モードに比べて、並列 TCP コネクション数の立ち上がりが遅くなっていることが分かる。ただし、AIMD モードの過渡特性は、加算増加量 $\alpha$ および乗算減少量 $\beta$ に大きく依存するため、これらの制御パラメータの影響について今後検討を行ってゆく予定である。

## 5　まとめと今後の課題

本稿では、特に GridFTP の並列データ転送機構に着目し、GridFTP の自動パラメータ設定機構を提案した。提案した自動パラメータ設定機構は、GridFTP サーバ–クライアント間で転送したいファイルをチャンクと呼ばれる単位で転送し、GridFTP のグッドプットおよびラウンドトリップ時間を計測する。計測した GridFTP のグッドプットとラウンドトリップ時間をもとに、GridFTP のグッドプットを最大化するように、並列 TCP コネクション数を調整するというものである。並列 TCP コネクション数の調整方法として、MI モード、MI+ モード、AIMD モードの 3 種類の方式を提案し、それぞれの有効性をシミュレーション実験により評価した。その結果、MI モードおよび MI+ モードを用いることにより、提案する自動パラメータ設定機構は GridFTP のグッドプットを向上させ、ネットワークの利用可能帯域を有効に利用できることを明らかにした。

今後は、さまざまな条件下でシミュレーションを行い、提案する自動パラメータ設定機構が、多様なネットワーク環境において良好に動作するためのパラメータ領域を明らかにする予定である。

## 謝　　辞

本研究を実施するにあたり、有意義な議論をしていただいた、大阪大学大学院情報科学研究科の村田正幸氏、山本英之氏に感謝いたします。

## 文　　献

[1] I. Foster and C. Kesselman, *The Grid: Blueprint for a New Computing Infrastructure*. San Francisco: Morgan Kauffman, 1999.

[2] J. Postel, "Transmission control protocol," *Request for Comments (RFC) 793*, Sept. 1981.

[3] J. Padhye and S. Floyd, "On inferring TCP behavior," *ACM SIGCOMM Computer Communication Review*, vol. 31, pp. 287–298, Aug. 2001.

[4] The Globus Project, "GridFTP: universal data transfer for the Grid," *White Paper*, Sept. 2003. available at `http://www.globus.org/datagrid/deliverables/C2WPdraft3.pdf`.

[5] W. Allcock, J. Bester, J. Bresnahan, S. Meder, P. Plaszczak, and S. Tuecke, "GridFTP: Protocol extensions to FTP for the Grid," *Grid Working Draft (Recommendation)*, Apr. 2003.

[6] The Globus Project, "GridFTP update January 2002," 2002. available at `http://www.globus.org/datagrid/deliverables/GridFTP-Overview-200201.pdf%`.

[7] S. Thulasidasan, W. Feng, and M. K. Gardner, "Optimizing GridFTP through dynamic right-sizing," in *Proceedings of IEEE International Symposium on High Performance Distributed Computing*, June 2003.

[8] 伊藤 建志, 大崎 博之, 今瀬 真, "広域グリッドコンピューティングにおけるデータ転送プ ロトコル GridFTP のパラメータ設定方法に関する検討," 電子情報通信学会技術研究報告 (IN2004-39), July 2004.

[9] J. Postel and J. Reynolds, "File transfer protocol (FTP)," *Request for Comments (RFC) 959*, Oct. 1985.

[10] R. Elz and P. Hethmon, "FTP security extensions," *Request for Comments (RFC) 2228*, Oct. 1997.

[11] P. Hethmon and R. Elz, "Feature negotiation mechanism for the file transfer protocol," *Request for Comments (RFC) 2389*, Aug. 1998.

[12] "Global Grid Forum." `http://www.ggf.org/`.

[13] "Globus Toolkit." available at `http://www.globus.org/`.

[14] L. Qiu, Y. Zhang, and S. Keshav, "On individual and aggregate TCP performance," in *Proceedings of Internetl Conference on Network Protocols*, pp. 203–212, Oct. 1999.

[15] S. Floyd, "Highspeed TCP for large congestion windows," *Internet Draft* `draft-ietf-tsvwg-highspeed-01.txt`, Aug. 2003.