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.

GridFTP-APT: データ転送プロトコル GridFTP の 並列 TCP コネクション数調整機構

伊藤　建志† 　大崎　博之† 　今瀬　眞†

† 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　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、複数の TCP コネクションを並列に確立することによりスループットの向上を図る、「並列データ転送」と呼ばれる機能をサポートしている。しかし、GridFTP が高いスループットを実現するためには、並列 TCP コネクション数を、ネットワーク環境に応じて適切に設定しなければならない。本稿では、グリッドのミドルウエア層から計測できる情報のみを用いて、並列 TCP コネクション数を調整する機構 GridFTP-APT (GridFTP with Automatic Parallelism Tuning) を提案する。シミュレーション実験により、GridFTP-APT を用いることにより、さまざまなネットワーク環境で高いスループットを実現できることを示す。
キーワード　グリッド、GridFTP、並列 TCP コネクション、黄金分割探索法

GridFTP-APT: Automatic Parallelism Tuning Mechanism for Data Transfer Protocol GridFTP

Takeshi ITO†, 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　GridFTP has been used as a data transfer protocol to effectively transfer a large volume of data in Grid computing. GridFTP supports a feature called *parallel data transfer* that improves throughput by establishing multiple TCP connections in parallel. However, in order to achieve high GridFTP throughput, the number of TCP connections should be optimized based on the network status. In this paper, we propose an automatic parallelism tuning mechanism called *GridFTP-APT (GridFTP with Automatic Parallelism Tuning)* that adjusts the number of parallel TCP connections only using information measurable in a Grid middleware. Through simulation experiments, we demonstrate that GridFTP-APT significantly improves the performance of GridFTP in various network environment.
Key words　Grid, GridFTP, Parallel TCP Connections, Golden Section Search Method

1　は じ め に

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のプロトコルとして、GridFTP が提案されている [1, 2]。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、GridFTP の制御パラメータ設定方法に関する研究がいくつか行われている。例えば、文献 [3] は、GridFTP の TCP ソケットバッファサイズを、自動的に設定する手法を提案している。これは、GridFTP サーバ–クライアント間で、ラウンドトリップ時間および帯域を計測することにより、ネットワークの

帯域遅延積を計算し、TCP ソケットバッファサイズを決定するというものである。しかし、文献 [3] で提案されている手法は、GridFTP プロトコルの変更が必要であり、既存の GridFTP サーバとの相互接続性を実現できない。また、文献 [3] では、TCP ソケットバッファサイズのみに着目しており、並列 TCP コネクション数に関しては考慮されていない。

文献 [4] は、並列 TCP コネクションのスループットを、簡単な解析モデルによって求めている。しかし、この解析モデルは、2 章で述べるような、並列 TCP コネクション数が大きすぎる時の、スループットの低下をモデル化していない。

一方、文献 [5] では、数学的解析によって、GridFTP の最適な制御パラメータを導出している。文献 [5] では、定常状態における GridFTP のグッドプットを導出することで、GridFTP の制御パラメータ (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) をどのように設定すればよいかを明らかにしている。しかし、現実のネットワークにおいて文献 [5] の結果を利用するためには、ネットワークのラウンドトリップ時間およびボトルネックリンク帯域が既知でなければならない。しかし、文献 [5] では、GridFTP のラウンドトリップ時間およびボトルネックリンクの帯域をどのように計測するかについては、議論されていない。

また、文献 [6] では、GridFTP の自動パラメータ設定機構を提案している。ここで提案されている手法は、転送したいファイルをチャンクと呼ばれるブロック単位で転送し、そのチャンク転送ごとにネットワークの状態 (ラウンドトリップ時間、グッドプット) を計測するというものである。そして、文献 [5] の解析結果を用いることにより、並列 TCP コネクション数を調整している。並列 TCP コネクション数の調整法として、異なる 3 種類のモードが提案されている。しかし、帯域遅延積が非常に大きいネットワークなど、いくつかの条件下では、GridFTP のグッドプットがそれほど向上しないという問題がある [6]。

そこで本稿では、GridFTP における並列 TCP コネクション数の自動調整機構 GridFTP-APT (GridFTP with Automatic Parallelism Tuning) を提案する。GridFTP-APT は、GridFTP クライアント上で動作し、グリッドのミドルウエア層から計測できる情報のみを利用する。GridFTP-APT は、GridFTP のグッドプットが、並列 TCP コネクション数に関して、上に凸の関数であるという性質を利用する。GridFTP-APT は、最大化問題の数値計算アルゴリズムを用いて、最適な並列 TCP コネクション数を探索する。本稿では、シミュレーション実験により、GridFTP-APT の性能評価を行う。その結果、GridFTP-APT を用いることにより、さまざまなネットワーク環境で高いスループットを実現できることを示す。

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、GridFTP の並列 TCP コネクション数調整機の設計方針と、我々が提案する GridFTP-APT の基本的なアイディアを述べた後、GridFTP-APT のアルゴリズムを説明する。4 章では、シミュレーション実験により GridFTP-APT 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既存の FTP (File Transfer Protocol) [7–9] を拡張したプロトコルであり、現在、GGF (Global Grid Forum) [10] 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 では、既存の FTP に対して、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といった機能が追加されている [1]。

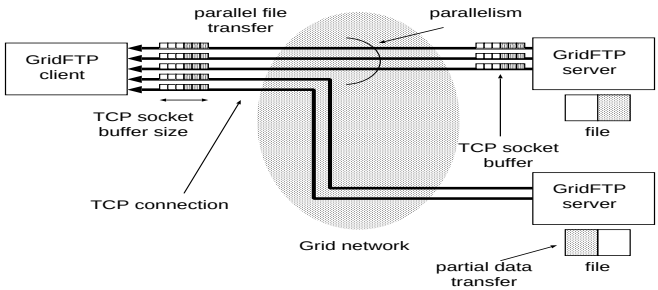

図 1　GridFTP における並列データ転送
Fig. 1　Parallel data transfer in GridFTP

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を追加することによって実現されている [1]。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 [11] では、GridFTP バージョン 1 (GridFTP v1) に準拠した、GridFTP サーバおよび GridFTP クライアントが含まれている。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指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 [2]。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 においても検討が行なわれていない。

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 [12]。

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、より大きな帯域を利用できる。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 並列 TCP コネクション数調整機構

3.1 設計方針

まず、GridFTP の並列 TCP コネクション数調整機構の設計方針を述べる。

GridFTP の並列 TCP コネクション数調整機構が、既存の GridFTP との互換性を実現することは非常に重要である。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、並列 TCP コネクション数調整機構は、GridFTP クライアント側のみの変更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、並列 TCP コネクション数調整機構を実現することが望ましい。並列 TCP コネクション数調整機構を GridFTP クライアント側で実現する場合、GridFTP における第三者転送をサポート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 の並列 TCP コネクション数調整機構が、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重要である。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、GridFTP の並列 TCP コネクション数調整機構は、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、並列 TCP コネクション数調整機構を実現することが望ましい。つまり、並列 TCP コネクション数調整機構は、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。

3.2 GridFTP-APT の基本的なアイディア

文献 [5] 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 コネクション当たりの 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。

本稿では、GridFTP のグッドプットが、並列 TCP コネクション数に関して、上に凸の関数であるという性質を利用し、並列 TCP コネクション数を、自動的に調整する機構 GridFTP-APT を提案する。

GridFTP-APT の基本的なアイディアは、GridFTP クライアントが、転送したいファイルを「チャンク」と呼ばれるブロックに分割し、チャンク転送ごとに、並列 TCP コネクション数を調整するというものである。GridFTP の拡張ブロックモードを用いることにより、このようなチャンク単位の転送が可能となる。GridFTP-APT では、最大化問題の数値計算アルゴリズムを用いて、チャンク転送ごとに GridFTP のグッドプットを計測する。その結果に応じて、GridFTP のグッドプットが最大化されるように、並列 TCP コネクション数を調整する。

GridFTP-APT では、最大化問題の数値計算アルゴリズムとして、黄金分割探索法 (Golden Section Search Method) を用いる [13]。黄金分割探索法とは、ある関数 $f(x)$ が範囲 $[x_l : x_r]$ において上に凸であるという性質を持つ時に、その範囲内で $f(x)$ を最大化する x を数値的に探索する手法である。GridFTP-APT では、黄金分割探索法を用いて、GridFTP のグッドプットが最大となる、並列 TCP コネクション数を数値的に探索する。

なお、GridFTP のグッドプットは、拡張ブロックモード [1] 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。具体的には、GridFTP の拡張ブロックモードを用いて、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 [1] ため、転送したチャンクの大きさ X 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。

3.3 並列 TCP コネクション数調整アルゴリズム

まず、黄金分割探索法を適用するために、GridFTP のグッドプットが上に凸となる、並列 TCP コネクション数の範囲 (以下、ブラケットと呼ぶ) を探索する。

並列 TCP コネクション数を小さな値から開始し、GridFTP のグッドプットが減少し始めるまで、各チャンク転送ごとに、並列 TCP コネクション数を乗算的に増加させる。これにより、GridFTP のグッドプットが最大となる点を含む、並列 TCP コネクション数の範囲 (ブラケット) を決定する。

以下では、チャンク転送に用いる並列 TCP コネクション数を N、その時に計測された GridFTP のグッドプットを $G(N)$ と表記する。また、k 回前のチャンク転送時に用いた並列 TCP コネクション数を N_{-k} と表記する。

GridFTP-APT は、以下のようにブラケットを探索する。

(1) 並列 TCP コネクション数 N を初期化

$$N \leftarrow N_0$$

ここで、N_0 は並列 TCP コネクション数の初期値である。

(2) チャンクを転送し、その時の GridFTP のグッドプット $G(N)$ を計測する。

(3) 以下の不等式が成立すれば、ブラケットを (N_{-2}, N_{-1}, N) と決定し、アルゴリズムを終了する。成立しなければステップ (4) へ進む。

$$G(N) < G(N_{-1})$$

（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。

$$N \leftarrow \alpha \times N$$

ここで $\alpha(>1)$ は制御パラメータである。なお、並列 TCP コネクション数は 1 以上の整数であるため、実際には、上記によって得られる N に最も近い整数を用いる。

上記のように決定されたブラケット (l, m, r) に対して、チャンク転送中に黄金分割探索法を適用し、GridFTP のグッドプットが最大となる、並列 TCP コネクション数を探索する。

GridFTP-APT は、以下のように、最適な並列 TCP コネクション数 N を探索する。

（1） 並列 TCP コネクション数 N を以下のように決定する。

$$N \leftarrow \begin{cases} l + (m-l)\nu & \text{if } m-l > r-m \\ m + (r-m)\nu & \text{otherwise} \end{cases} \quad (3)$$

ここで ν は黄金分割比 $(= (3-\sqrt{5})/2)$ である。なお、並列 TCP コネクション数は 1 以上の整数であるため、実際には、上記によって得られる N に最も近い整数の値を用いる。

（2） チャンクを転送し、その時の GridFTP のグッドプット $G(N)$ を計測する。

（3） 以下の不等式が成立すれば、ステップ (4) へ進む。

$$G(N) > G(m)$$

成立しなければ、ブラケットを以下のように変更し、ステップ (1) に戻る。

$$(l, m, r) \leftarrow \begin{cases} (l, m, N) & \text{if } m < N \\ (N, m, r) & \text{otherwise} \end{cases}$$

（4） ブラケットを以下のように変更し、ステップ (1) に戻る。

$$(l, m, r) \leftarrow \begin{cases} (m, N, r) & \text{if } m < N \\ (l, N, m) & \text{otherwise} \end{cases} \quad (4)$$

3.4 チャンクサイズ決定アルゴリズム

各チャンク転送における、チャンクの大きさ (チャンクサイズ) X をどのように決定するかは、重要な問題である。

ブラケットの探索や、黄金分割探索法による並列 TCP コネクション数の探索を速くするためには、チャンクサイズをできるだけ小さく取ることが望ましい。これにより、上記のアルゴリズムの各ステップが短時間で終了するため、結果として、最適な並列 TCP コネクション数へ、より速く収束することが期待できる。

しかし、チャンクサイズが小さすぎると、TCP の特性上 [14]、各 TCP コネクションのグッドプットを、正確に計測することができない。結果として、並列 TCP コネクション数が、最適な値に収束せずに、GridFTP のグッドプットが低く抑えられてしまう。このため、チャンクサイズは、TCP のグッドプットが正確に計測できる程度に大きくする必要がある。

TCP のグッドプットを正確に計測するために、チャンクサイズをどの程度大きく取ればいいかは、ネットワークの帯域によって決定される。このため、チャンクサイズは、ネットワークの帯域に応じて変化させる必要がある。しかし、当然のことながら、チャンク転送時の、GridFTP のグッドプットを事前に知ることはできない。GridFTP-APT では、GridFTP の拡張ブロックモードを用いてチャンク転送を行うが、GridFTP の拡張ブロックモードでは、事前に転送するブロックの大きさを指定しなければならないという制限がある。

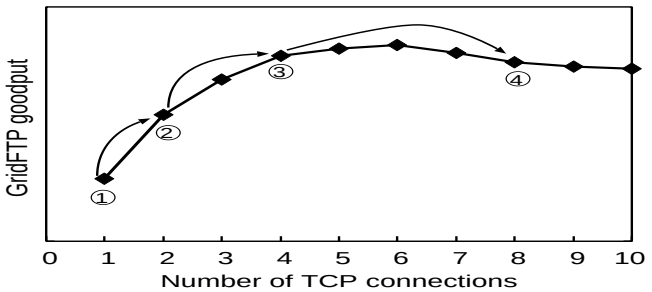

図 2 GridFTP-APT の動作例 (ブラケットの探索時)
Fig. 2 Example of GridFTP-APT operation (searching for bracket)

以上のような問題に対処するため、GridFTP-APT では、次回のチャンク転送時の GridFTP のグッドプットを予測することにより、チャンク転送に要する時間ができるだけ一定となるよう、チャンクサイズを動的に変更する。具体的には、GridFTP-APT は、以下のようにチャンクサイズ X を決定する。

ブラケットを探索する時は、次回のチャンク転送時の GridFTP のグッドプットを、前回および前々回の GridFTP グッドプットの比から $G(N_{-1}) \times G(N_{-1})/G(N_{-2})$ と予測し、以下のようにチャンクサイズを決定する。

$$X \leftarrow G(N_{-1}) \frac{G(N_{-1})}{G(N_{-2})} \Delta$$

ここで、Δ は制御パラメータであり、チャンク転送時間の目標値を意味している。

なお、最初のチャンク転送時には、前回および前々回の GridFTP のグッドプット $G(N_{-1})$、$G(N_{-2})$ が不明であるため、以下のようにチャンクサイズ X を決定する。

$$X \leftarrow \frac{N_0\, W}{R} \Delta$$

ここで、W は、TCP ソケットバッファサイズであり、R はラウンドトリップ時間である。ラウンドトリップ時間は、制御チャンネルにおけるコマンドの応答時間から計測する。また、2 回目のチャンク転送時には、前々回の GridFTP のグッドプット $G(N_{-2})$ が不明であるため、以下のようにチャンクサイズ X を決定する。

$$X \leftarrow \alpha\, G(N_{-1}) \Delta$$

黄金分割法を適用し、最適な並列 TCP コネクション数を探索する時は、次回のチャンク転送時の GridFTP のグッドプットを、ブラケット (l, m, r) 内の 2 点の GridFTP グッドプットの内挿によって予測し、以下のようにチャンクサイズを決定する。

$$X \leftarrow \begin{cases} ((1-\xi)\, G(l) + \xi\, G(m))\, \Delta & \text{if } N < m \\ ((1-\xi)\, G(m) + \xi\, G(r))\, \Delta & \text{otherwise} \end{cases}$$

$$\xi \leftarrow \begin{cases} \frac{N-l}{m-l} & \text{if } N < m \\ \frac{N-m}{r-m} & \text{otherwise} \end{cases}$$

3.5 GridFTP-APT の動作例

ここで、GridFTP-APT の動作例を説明する。図 2 に、GridFTP-APT がブラケットを探索する例を示す。ここでは、$N_0 = 1$ および $\alpha = 2$ としている。また、円中の数字 k は、k 回目のチャンク転送を意味している。

GridFTP-APT は、GridFTP のグッドプットが上に凸となる、並列 TCP コネクション数の範囲 (ブラケット) を探索する。ま

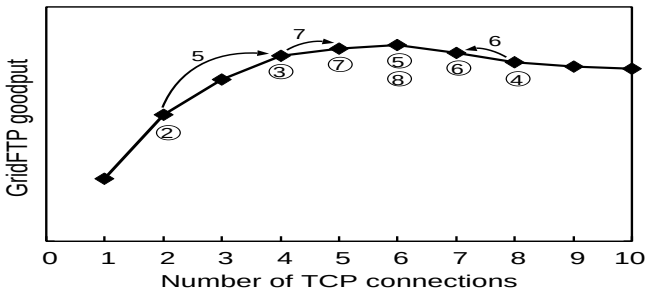

図 3　GridFTP-APT の動作例 (最適な並列 TCP コネクション数の探索時)

Fig. 3　Example of GridFTP-APT operation (searching for the optimal number of TCP connections)

ず、並列 TCP コネクション数 N を 1 (= N_0) から開始する。GridFTP のグッドプットが増加から減少に転じるまで、チャンク転送ごとに、並列 TCP コネクション数 N を、$1 \to 2 \to 4 \to 8$ と増加させる。並列 TCP コネクション数 N が $4 \to 8$ の時に、GridFTP のグッドプットが減少しているため、ブラケットが (2, 4, 8) と決定される。

次に、図 3 に、GridFTP-APT が最適な並列 TCP コネクション数を探索する例を示す。図 3 において、矢印上の数字 k は、k 回目のチャンク転送においてブラケットを変更することを意味している。

ブラケット (2, 4, 8) に対して、チャンク転送中に黄金分割探索法を適用し、GridFTP のグッドプットが最大となる、並列 TCP コネクション数を探索する。ブラケットが (2, 4, 8) であるため、5 回目のチャンク転送に用いる並列 TCP コネクション数 N は、式 (3) から $N = 6$ となる。5 回目のチャンク転送におけるグッドプットは $G(6)$ であるが、$G(4) < G(6)$ であるため、式 (4) より、ブラケットは (4, 6, 8) へと更新される。以下、同様に並列 TCP コネクション数 N が $6 \to 7 \to 5$ と変更され、それに伴いブラケットが $(4, 6, 8) \to (4, 6, 7) \to (5, 6, 7)$ と更新される。最終的に、ブラケットが (5, 6, 7) の時点で、グッドプットが最大となる $N = 6$ に並列 TCP コネクション数が固定される。

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、 GridFTP-APT の有効性を定量的に評価する。

シミュレーションに用いた、ネットワークのトポロジを図 4 に示す。図 4 における、ボトルネックリンクの伝搬遅延を変化させて、シミュレーションを行った。GridFTP サーバとクライアントが、2 台のルータを介して接続されている。GridFTP クライアントから GridFTP サーバに対して、連続的にデータ転送を行った。また、GridFTP-APT の基本的な特性を明らかにするため、今回はバックグラウンドトラヒックが存在しない環境でシミュレーションを行った。なお、シミュレーションには、ns-2 シミュレータ [15] を修正して使用した。

シミュレーションで用いたパラメータ設定を表 1 に示す。以降のシミュレーションでは、特に断りのない限り、表 1 のパラメータを用いている。

図 5 に、ボトルネックリンクの伝搬遅延を 10 [ms] と設定した時の、GridFTP-APT の並列 TCP コネクション数の時間的変動を示す。この図より、GridFTP-APT の並列 TCP コネクション数が、転送開始から 15 [s] 程度で $N = 13$ に収束していることが分かる。この例では、転送開始から 10 [s] 程度でブラケットが決定され、その後 5 [s] 程度で並列 TCP コネクション数の最適化が完了している。

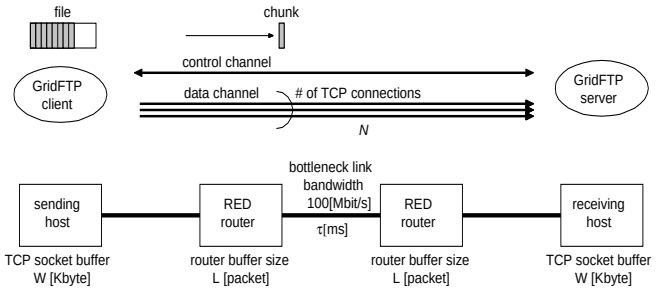

図 4　シミュレーションに用いたネットワークのトポロジ

Fig. 4　Network topology used in simulation

表 1　シミュレーションで用いたパラメータ設定

Table 1　Parameter configuration used in simulation

ボトルネックリンク帯域	100	[Mbit/s]
ボトルネックリンクの伝搬遅延	10, 20	[ms]
RED ルータのバッファサイズ	100	[packet]
RED ルータの制御パラメータ max_{th}	75	
RED ルータの制御パラメータ min_{th}	25	
RED ルータの制御パラメータ max_p	0.1	
RED ルータの制御パラメータ w_q	0.002	
TCP ソケットバッファサイズ	64	[Kbyte]
TCP のパケット長	1000	[byte]
並列 TCP コネクション数の初期値 N_0	4	
並列 TCP コネクション数の乗算増加量 α	2	
チャンク転送時間の目標値 Δ	1.0	[s]

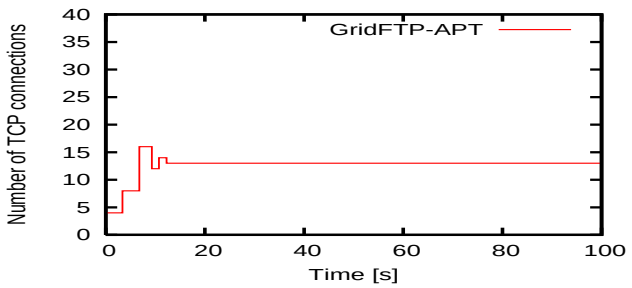

図 5　GridFTP-APT の並列 TCP コネクション数の時間的変動 (τ = 10 [ms])

Fig. 5　Number of TCP connections in GridFTP-APT for $\tau = 10$ [ms]

この時の、GridFTP-APT のグッドプットを図 6 に示す。比較のため、並列 TCP コネクション数を 1、4、8、16、32 と固定した時の、GridFTP の定常状態におけるグッドプットもあわせて示している。まず、並列 TCP コネクション数を固定した時のグッドプットから、最適な並列 TCP コネクション数が 8 ~ 16 の間に存在することが分かる。GridFTP-APT は、転送開始から 10 [s] ほどで、約 80 [Mbit/s] のグッドプットに到達している。さらに、その後 5 [s] 程度でグッドプットが 85.3 [Mbit/s] に収束している。なお、今回のシミュレーションでは、RED ルータのバッファサイズが 100 [packet] であるため、並列 TCP コネクション数が最適化された状態でも、85 [Mbit/s] 程度が最大のグッドプットであった。つまり、GridFTP-APT は、転送開始から 15 [s] 程度で、ネットワーク資源を有効に利用できている。

さらに、ボトルネックリンクの伝搬遅延を 20 [ms] に設定した時の、GridFTP-APT の並列 TCP コネクション数およびグッドプットの時間的変動を、それぞれ図 7 および図 8 に示す。

図 7 より、転送開始から 25 [s] 程度で、並列 TCP コネクション数が収束していることがわかる。ボトルネックリンクの伝搬遅延が 10 [ms] の場合 (図 7) よりも大きくなっているため、並

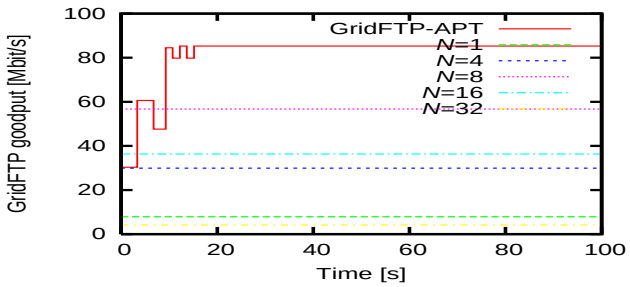

図 6　GridFTP-APT のグッドプットの時間的変動 ($\tau = 10$ [ms])
Fig. 6　Evolution of goodput in GridFTP-APT for $\tau = 10$ [ms]

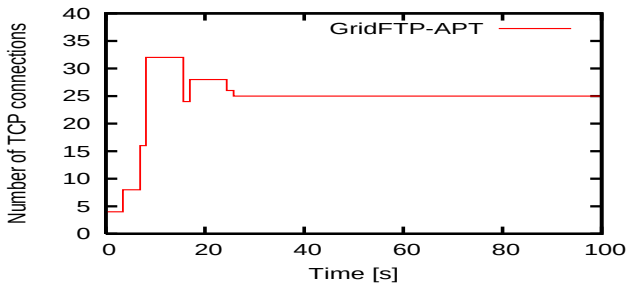

図 7　GridFTP-APT の並列 TCP コネクション数の時間的変動 ($\tau = 20$ [ms])
Fig. 7　Number of TCP connections in GridFTP-APT for $\tau = 20$ [ms]

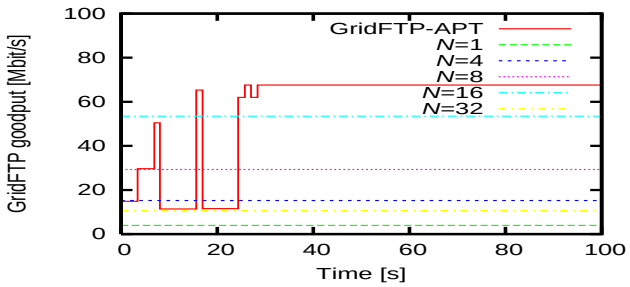

図 8　GridFTP-APT のグッドプットの時間的変動 ($\tau = 20$ [ms])
Fig. 8　Evolution of goodput in GridFTP-APT for $\tau = 20$ [ms]

列 TCP コネクション数が収束するまでの時間が長くなっていると考えられる。また、この時の並列 TCP コネクション数は $N = 25$ であり、ボトルネックリンクの伝搬遅延が 10 [ms] の場合 ($N = 13$) と大きく異なっていることも分かる。

図 8 から、この時、GridFTP-APT が並列 TCP コネクション数を調整することにより、高いグッドプットを実現できていることが分かる。図 8 から分かるように、GridFTP-APT が、転送開始直後にブラケットを特定するまではグッドプットは一時的に低下しているが、チャンク転送ごとに並列 TCP コネクション数を調整することにより、最終的に高いグッドプットを実現している。なお、この条件下では、並列 TCP コネクション数が最適化された状態でも、67.5 [Mbit/s] 程度が最大のグッドプットであった。つまり、GridFTP-APT は、ボトルネックリンクの伝搬遅延が変化した場合であっても、ネットワーク資源を有効に利用できていることが分かる。

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 の並列 TCP コネクション数調整機構 GridFTP-APT を提案した。GridFTP-APT は、GridFTP のグッドプットが、並列 TCP コネクション数に関して、上に凸の関数であるという性質を利用し、最大化問題の数値計算アルゴリズムを用いて、最適な並列 TCP コネクション数を探索する。本稿では、シミュレーション実験により、GridFTP-APT が、さまざまなネットワーク環境で高いスループットを実現できることを示した。

今後の課題として、より一般的なネットワーク環境における GridFTP-APT の性能評価が挙げられる。特に、バックグラウンドトラヒックの影響や、複数の GridFTP-APT が混在する環境における性能評価が必要である。また、提案する GridFTP-APT を実装し、現実のネットワークにおいて性能評価を行う予定である。

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に感謝いたします。また、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ー網形成プロジェクト (NAREGI) および、文部科学省における科学研究費補助金特定領域研究 (課題番号 16016261) の支援を受けている。ここに記して感謝いたします。

文　　献

[1] W. Allcock *et al.*, "GridFTP: Protocol extensions to FTP for the Grid," *GGF Document Series GFD.20*, Apr. 2003. Also available as `http://www.gridforum.org/GFD.20.pdf`.

[2] I. Mandrichenko, W. Allcock, and T. Perelmutov, "GridFTP v2 protocol description," *GGF Document Series GFD.47*, May 2005. Also available as `http://www.gridforum.org/GFD.47.pdf`.

[3] 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.

[4] T. J. Hacker and B. D. Athey, "The end-to-end performance effects of parallel TCP sockets on a lossy wide-area network," in *Proceedings of the 16th IEEE-CS/ACM International Parallel and Distributed Processing Symposium (IPDPS)*, Aug. 2001.

[5] T. Ito, H. Ohsaki, and M. Imase, "On parameter tuning of data transfer protocol GridFTP in wide-area Grid computing," in *Proceedings of Second International Workshop on Networks for Grid Applications (GridNets 2005)*, pp. 415–421, Oct. 2005.

[6] T. Ito, H. Ohsaki, and M. Imase, "Automatic parameter configuration mechanism for data transfer protocol GridFTP," to be presented at *the 2006 International Symposium on Applications and the Internet (SAINT 2006)*, Jan. 2006.

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

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

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

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

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

[12] 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.

[13] W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling, *Numerical Recipes in C: The Art of Scientific Computing*. Cambridge University Press, 1992.

[14] N. Ehsan and M. Liu, "Analysis of TCP transient behavior and its effect on file transfer latency," in *Proceedings of IEEE International Conference on Communications (ICC2003)*, vol. 26, pp. 1806–1811, May 2003.

[15] "The network simulator – ns2." available at `http://www.isi.edu/nsnam/ns/`.