PAPER

GridFTP-APT: Automatic Parallelism Tuning Mechanism for GridFTP in Long-Fat Networks

Takeshi ITO[†a)]**,** ***Student Member,*** **Hiroyuki OHSAKI**[†b)]**,** ***and*** **Makoto IMASE**[†c)]**,** ***Members***

SUMMARY In this paper, we propose an extension to GridFTP that optimizes its performance by dynamically adjusting the number of parallel TCP connections. 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, for achieving 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 according to information available to the Grid middleware. Through simulations, we demonstrate that GridFTP-APT significantly improves the performance of GridFTP in various network environments.

key words: *automatic parallelism tuning, Golden Section Search method, Grid computing, GridFTP, GridFTP-APT, parallel TCP connections, parameter tuning*

1. Introduction

GridFTP has been proposed as a protocol to effectively transfer a large volume of data in Grid computing [1]–[3]. GridFTP is designed to solve the existing TCP problems and has various additional features for this purpose. These features include, for instance, parallel data transfer using multiple TCP connections and automatic negotiation of TCP socket buffer size. It is known that the effectiveness of GridFTP depends largely on control parameter configuration such as the number of parallel TCP connections [4], [5]. However, examination has not been conducted sufficiently so far regarding how to configure GridFTP control parameters. For example, although a command is defined in the GridFTP protocol to specify the number of parallel TCP connections between GridFTP server and client, it is not specified at all in the GridFTP protocol how to determine the number of parallel TCP connections.

There have been several studies on methods for configuring the GridFTP control parameters. In [6], the authors propose a method to determine the required TCP socket buffer size for GridFTP. By measuring the round-trip time and the bandwidth between GridFTP server and client, this method calculates the bandwidth-delay product of the network, and determines the TCP socket buffer size. However, since the GridFTP protocol needs to be modified when using the method proposed in [6], interoperability with existing GridFTP servers is unrealizable. Moreover, in [6], the authors focus only on the TCP socket buffer size, and do not consider the number of parallel TCP connections.

In [4], the throughput of parallel TCP connections is derived using a simple analytic model. However, the analytic approach in [4] does not model degradation of throughput when the number of parallel TCP connections is too large.

In [7], the optimal control parameters of GridFTP are derived using mathematical analysis. In [7], the authors clarify the configuration method of GridFTP control parameters (i.e., the number of parallel TCP connections and the TCP socket buffer size) by deriving the GridFTP goodput in steady state. However, to apply the result in [7] to a real network, the round-trip time and the bottleneck link bandwidth of a network must be known in advance. However, it is not discussed in [7] how the round-trip time and the bottleneck link bandwidth are measured by GridFTP.

In [8], the authors propose an automatic parameter configuration mechanism for GridFTP. The proposed method transfers a file as a series of blocks called *chunk*, and measures network status (i.e., the round-trip time and GridFTP goodput) for every chunk transfer. Based on the analytic result in [7] and measurement results, the number of parallel TCP connections is adjusted. In [8], three operation modes are proposed for heuristically adjusting the number of parallel TCP connections. However, there is a problem that GridFTP goodput does not improve so much in some conditions — for instance, in a network with a large bandwidth-delay product [8].

In this paper, we therefore propose a GridFTP-APT (GridFTP with Automatic Parallelism Tuning) mechanism that automatically adjusts the number of parallel TCP connections of GridFTP. GridFTP-APT operates on a GridFTP client according to information available to the Grid middleware. GridFTP-APT utilizes the fact that GridFTP goodput is a convex function for the number of parallel TCP connections. GridFTP-APT searches for the optimal number of parallel TCP connections using a numerical computation algorithm for a maximization problem. We evaluate the performance of GridFTP-APT through simulations. Consequently, we show that GridFTP-APT can realize high throughput in various network environments.

The structure of this paper is as follows. First, Section 2 provides a general description of GridFTP as well as the explanation on parallel data transfer, one of the notable features of GridFTP. Section 3 discusses design principles of an automatic parallelism tuning mechanism for GridFTP.

[†]Graduate School of Information Science and Technology, Osaka University, Japan

a) E-mail: t-itou@ist.osaka-u.ac.jp

b) E-mail: oosaki@ist.osaka-u.ac.jp

c) E-mail: imase@ist.osaka-u.ac.jp

We then explain the basic ideas and algorithm of our proposed GridFTP-APT. Section 4 demonstrates the effectiveness of GridFTP-APT through simulations. Finally, Section 5 summarizes the paper and mentions future tasks.

2. GridFTP

GridFTP is a data transfer protocol, which is designed to effectively transfer a large volume of data in Grid computing. GridFTP is an extension to FTP (File Transfer Protocol) [9]–[11] that has been widely used, and was standardized in OGF (The Open Grid Forum, formerly known as GGF (The Global Grid Forum)) [12]. GridFTP, which uses TCP as its transport layer communication protocol, is designed to solve several problems of TCP. For example, besides the features of the existing FTP, it has additional features such as automatic negotiation of TCP socket buffer size, parallel data transfer, third-party control of file transfer, partial file transfer, security, and reliable data transfer [2].

Most of these specific features of GridFTP are realized by a new transfer mode called extended block mode [2]. Currently, GridFTP server and client software conforming to GridFTP version 1 (GridFTP v1) is included in the Globus Toolkit [13], which is the de facto standard middleware for Grid computing. However, in this specific GridFTP implementation, the feature of the automatic negotiation of TCP socket buffer size is not implemented, so a user must manually specify the number of parallel TCP connections for parallel data transfer. In addition, OGF has been discussing the problems with GridFTP v1 and has finished standardization of GridFTP v2 (version 2) as a solution to these problems [14]. Additional features have been incorporated in GridFTP v2. These features relax several limitations of the extended block mode in GridFTP v1, such that data transfer is restricted to a single direction and unable to open/close data channels in the midst of the data transfer. However, how to configure the GridFTP control parameters regarding parallel TCP connections has not been addressed even in the standardization of GridFTP v2.

Multiple TCP connections can be established in parallel in GridFTP using `OPTS RETR` command (Fig. 1). With this feature, a single file can be transferred from a single server through multiple TCP connections. Higher throughput can be expected by aggregating multiple TCP connections in comparison to using a single TCP connection [5].

This can be explained by the following reasons: (1) By aggregating multiple TCP connections, larger bandwidth can be gained in the TCP congestion avoidance phase than those gained by other competing TCP connections. This situation results from the fact that, in the TCP congestion avoidance phase, the AIMD window flow control is being executed, and therefore aggregated multiple TCP connections can transfer data more advantageously through the network with a smaller packet loss probability. (2) By aggregating multiple TCP connections, the total TCP socket buffer size available to the file transfer increases. Aggregation of N TCP connections can utilize as N times TCP socket buffer size as that of a single TCP connection. (3) By aggregating multiple TCP connections, ramp-up time of the transfer rate in the TCP slow-start phase is shortened. In the slow-start phase, the congestion window doubles every round-trip time. For this reason, through the aggregation of N TCP connections the speed of the transfer rate increase becomes N times as fast as that of a TCP connection.

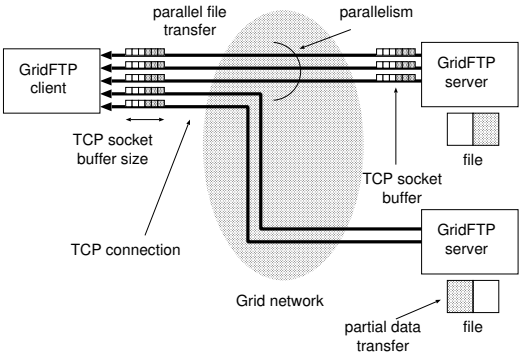


Fig. 1: Parallel data transfer in GridFTP

On the contrary, the throughput drops if the number of aggregate TCP connections, N, becomes too large since this may cause the following situations: (1) The window size per TCP connection becomes smaller, so TCP timeout would frequently occur. (2) The overhead required for the server to process TCP protocol stack would increase. Therefore, the optimal number of TCP connections, N, should be determined based on the network status. Nevertheless, how to optimize the number of parallel TCP connections, N, in GridFTP has not sufficiently been studied and still remains as an open issue.

3. Automatic Parallelism Tuning Mechanism for GridFTP

3.1 Design Principles

First, we explain the basic principles for designing an automatic parallelism tuning mechanism for GridFTP.

It is extremely important to provide compatibility with existing GridFTP servers in designing an automatic parallelism tuning mechanism for GridFTP. GridFTP is implemented in the Globus Toolkit and has been spreading rapidly in recent years. Since a number of GridFTP servers have already been in operation, it is preferable to realize the automatic parallelism tuning mechanism at the side of GridFTP client. Additionally, it is also favorable to realize the automatic parallelism tuning mechanism without changing the existing GridFTP protocol so as to enable interconnection with existing GridFTP servers. If the automatic parallelism tuning is executed on the side of GridFTP client, it will be difficult to realize the automatic parallelism tuning during the third party transfer. Nonetheless, we believe that this would not be so problematic because most of the transfers are executed between GridFTP servers and clients.

It is desirable that the automatic parallelism tuning

mechanism for GridFTP can easily be installed in Grid computing environment. Generally, Grid computing is characterized by the heterogeneity of computers and networks constituting Grid. Therefore, the automatic parallelism tuning mechanism needs to operate in various computer environments as well as various network environments. For this reason, it is preferable for the automatic parallelism tuning mechanism to be realized in the Grid middleware layer. In other words, it is important for the automatic parallelism tuning mechanism to avoid using any function specific to certain operating systems or network devices on the computers.

3.2 Basic Ideas of GridFTP-APT

In [7], the GridFTP goodput G in steady state is approximately derived as

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

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

where N is the number of parallel TCP connections, W is the TCP socket buffer size for each TCP connection, B is the bottleneck link bandwidth, and R is the round-trip time of the TCP connections.

Equation (1) indicates that the GridFTP goodput G is a convex function for the number N of parallel TCP connections. Thus, the number of parallel TCP connections, N, should be selected so as to maximize the GridFTP goodput G.

In this paper, we propose a GridFTP-APT mechanism that automatically adjusts the number of parallel TCP connections. GridFTP-APT utilizes the fact that the GridFTP goodput is a convex function for the number of parallel TCP connections. The basic idea of GridFTP-APT is that a GridFTP client divides a file to transfer into blocks called *chunk*, and adjusts the number of parallel TCP connections at the end of every chunk transfer. Chunk-based transfer can be realized using the extended block mode of GridFTP. More specifically, GridFTP-APT measures the goodput at every chunk transfer. According to measurement results, GridFTP-APT adjusts the number of parallel TCP connections so that the GridFTP goodput is maximized using a numerical computation algorithm for a maximization problem.

GridFTP-APT uses the GSS (Golden Section Search) algorithm, one of numerical computation algorithms for a maximization problem [15]. GSS algorithm is a technique of numerically searching for x that maximizes $f(x)$, when $f(x)$ is a convex function in the range of $[x_l : x_r]$ and derivatives of $f(x)$ are unknown. It is known that GSS algorithm is guaranteed to work in the worst possible case [16]. GridFTP-APT numerically searches for the optimal number of parallel TCP connections that maximizes the GridFTP goodput using GSS algorithm.

The GridFTP goodput of each chunk transfer can be calculated from the chunk size and its transfer time, which can be measured by transferring the chunk in the extended block mode [2]. Specifically, a chunk is transferred by `ERET` or `ESTO` command in the extended block mode of GridFTP while measuring its response time, T. Since the response to `ERET` or `ESTO` command is returned when the chunk transfer is completed [2], the GridFTP goodput can be calculated as $G = X/T$ from the chunk size, X, and the response time, T. Note that in parallel data transfer, after examining reception of all packets belonging to the chunk the receiver notifies the sender of chunk transfer completion.

3.3 Adjusting the Number of Parallel TCP Connections

First, for applying the GSS algorithm, GridFTP-APT searches for a range of the number of parallel TCP connections called *bracket*, in which the GridFTP goodput takes a convex form.

GridFTP-APT starts from a small number of parallel TCP connections, and multiplicatively increases the number of parallel TCP connections at every chunk transfer until the GridFTP goodput decreases. GridFTP-APT determines the bracket — the range of the number of parallel TCP connections covering the optimal value that maximizes the GridFTP goodput.

In what follows, N is the number of parallel TCP connections used for a chunk transfer, $G(N)$ the GridFTP goodput measured at the chunk transfer, and N_{-k} the number of parallel TCP connections used for the k-th last chunk transfer.

GridFTP-APT searches for the bracket as follows.

1. Initialize the number of parallel TCP connections:

   $$N \leftarrow N_0,$$

   where N_0 is the initial number of parallel TCP connections.
2. Transfer a chunk while measuring the GridFTP goodput $G(N)$.
3. If the following inequality is satisfied, determine the bracket as (N_{-2}, N_{-1}, N) and terminate the algorithm.

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

   Otherwise, proceed to the step 4.
4. Increase the number of parallel TCP connections as follows, and return to the step 2.

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

   where $\alpha(> 1)$ is a control parameter. Note that, since the number of parallel TCP connections is a positive integer, the integer closest to N in the above equation is used as the number of parallel TCP connections.

Using the GSS algorithm, GridFTP-APT searches for

the number of parallel TCP connections that maximizes the GridFTP goodput within the bracket (l, m, r) during succeeding chunk transfers.

GridFTP-APT searches for the optimal number N of parallel TCP connections as follows.

1. Update the number N of parallel TCP connections:

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

   where ν is the golden ratio ($= (3-\sqrt{5})/2$) [15]. Note that, since the number of parallel TCP connections is a positive integer, the integer closest to N in the above equation is used as the number of parallel TCP connections.
2. Transfer a chunk while measuring the GridFTP goodput $G(N)$.
3. If the following inequality is satisfied, proceed to the step 4.

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

   If the above inequality is not satisfied, change the bracket as follows and return to the step 1.

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

4. Change the bracket as follows, and return to the step 1.

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

3.4 Determining Chunk Size

It is an important problem to appropriately determine the size of a chunk in every chunk transfer.

With a large chunk size, the number of parallel TCP connections converges to the optimal value slowly, so that the GridFTP goodput is degraded. For accelerating search of the optimal number of parallel TCP connections, it is desirable to keep the chunk size as small as possible. With a small chunk size, since each step in the algorithm described above is completed more shortly, it is expected that the number of parallel TCP connections converges faster to the optimal value.

On the contrary, if the chunk size is too small, the goodput of each TCP connection cannot be measured accurately because of TCP's characteristics [17]. Consequently, since the number of parallel TCP connections does not converge to the optimal value, the GridFTP goodput cannot be maximized. For this reason, it is necessary to increase the chunk size so that the TCP goodput can be measured accurately.

The chunk size required for accurately measuring the TCP goodput is affected by the network bandwidth. Hence, it is necessary to determine the chunk size according to the network bandwidth. However, of course, we cannot know the network bandwidth *before* the chunk transfer. Since GridFTP-APT uses the extended block mode of GridFTP for chunk transfers, there is a limitation that the block size must be specified before starting the block transfer.

For solving this problem, GridFTP-APT predicts the GridFTP goodput of the next chunk transfer, and dynamically configures the chunk size so that the chunk transfer time becomes as fixed as possible. Specifically, GridFTP-APT determines the chunk size X as follows.

When searching for the bracket, GridFTP-APT predicts the GridFTP goodput of the next chunk transfer as $G(N_{-1}) \times G(N_{-1})/G(N_{-2})$ from the ratio of the last two chunk transfers, and determine the chunk size as

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

where Δ is a control parameter, which is the target value of the chunk transfer time.

Note that, at the time of the first chunk transfer, the GridFTP goodput $G(N_{-1})$ and $G(N_{-2})$ are unknown. Assuming that the GridFTP goodput at the time of the first chunk transfer is limited by the TCP socket buffer size (i.e., $G(N) = N\,W/R$), the chunk size X is determined as follows:

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

where W is the TCP socket buffer size and R is the round-trip time. The round-trip time is measured from response times of commands on the GridFTP control channel. Note that, at the time of the second chunk transfer, the GridFTP goodput $G(N_{-2})$ is unknown. Assuming that the GridFTP goodput of the second chunk transfer is also limited by the TCP socket buffer size, the chunk size X is determined as follows.

$$X = N/N_{-1}\,G(N_{-1})\Delta$$

When GridFTP-APT searches for the optimal number of parallel TCP connections with the GSS algorithm, the GridFTP goodput of the next chunk transfer is predicted by the interpolation of two samples of the GridFTP goodput in the bracket (l, m, r). Namely, the chunk size is determined as follows.

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

where

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

3.5 Example of GridFTP-APT Operation

In what follows, we illustrate an example operation of GridFTP-APT. Figure 2 shows an example operation of GridFTP-APT when searching for a bracket with $N_0 = 1$ and $\alpha = 2$. The number k shown in a circle indicates the k-th chunk transfer.

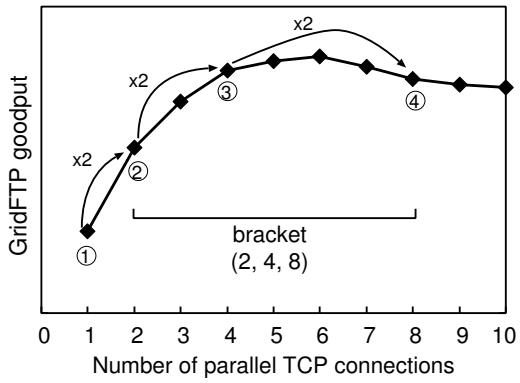


Fig. 2: Example of GridFTP-APT operation when searching for a bracket

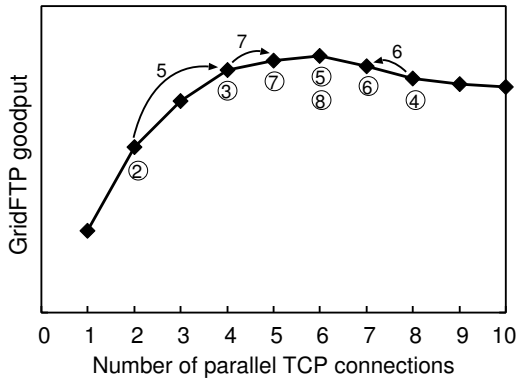


Fig. 3: Example of GridFTP-APT operation when searching for the optimal number of TCP connections

GridFTP-APT searches for a bracket (i.e., the range of the number of parallel TCP connections covering the optimal value). First, GridFTP-APT initialize the number N of parallel TCP connections to N_0 (= 1). GridFTP-APT multiplicatively increases the number of parallel TCP connections as $1 \rightarrow 2 \rightarrow 4 \rightarrow 8$ at every chunk transfer until the GridFTP goodput starts to decrease. Since the GridFTP goodput decreases when the number N of parallel TCP connections changes as $4 \rightarrow 8$, the bracket is determined as (2, 4, 8).

Next, Fig. 3 shows an example operation of GridFTP-APT when searching for the optimal number of parallel TCP connections. In Fig. 3, the number k on an arrow indicates that the bracket is changed at k-th chunk transfer.

The GSS algorithm is applied to the bracket (2, 4, 8) during chunk transfers, and the number of parallel TCP connections is adjusted for maximizing the GridFTP goodput. Since the bracket is (2, 4, 8), the number of parallel TCP connections at the 5-th chunk transfer N is determined as $N = 6$ from Eq. (3). The GridFTP goodput in the 5-th chunk transfer is $G(6)$, and since $G(4) < G(6)$ is satisfied, the bracket is updated as (4, 6, 8) from Eq. (4). Hereafter, in a similar way, GridFTP-APT changes the number of parallel TCP connections N as $6 \rightarrow 7 \rightarrow 5$, and updates the bracket as (4, 6, 8) $\rightarrow$ (4, 6, 7) $\rightarrow$ (5, 6, 7). Finally, when the bracket is (5, 6, 7), the number of parallel TCP connections is fixed at $N = 6$, which maximizes the GridFTP goodput.

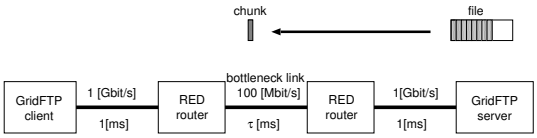


Fig. 4: Network model used in simulation for investigating GridFTP-APT operations and the effect of GridFTP-APT control parameters

Table 1 Parameter configuration used in simulation

bottleneck link bandwidth	100	[Mbit/s]
propagation delay of the bottleneck link	10	[ms]
buffer size of RED router L	100	[packet]
control parameter of RED router max_{th}	0.75 L	
control parameter of RED router min_{th}	0.25 L	
control parameter of RED router max_p	0.1	
control parameter of RED router w_q	0.002	
TCP socket buffer size	64	[Kbyte]
TCP packet size	1,500	[byte]
initial number of parallel TCP connections N_0	2	
GridFTP-APT control parameter α	2	
target value of chunk transfer time Δ	2.0	[s]
file size transfered using GridFTP F	1, 5, 10	[Gbyte]

4. Simulation

In this section, we quantitatively evaluate the effectiveness of GridFTP-APT through simulations.

4.1 GridFTP-APT Operation

First, we examine the evolution of the number of parallel TCP connections and the GridFTP goodput when using GridFTP-APT for demonstrating the GridFTP-APT operation.

Figure 4 shows the network model used in simulation. In this network model, a GridFTP server and client are connected via two RED routers. We mainly use RED routers for its effectiveness [18]. Performance of GridFTP-APT with DropTail routers will be investigated in Section 4.6. For preventing global synchronization [5], establishment of each TCP connection is randomly delayed between 0 and 0.1 [s]. Note that the timing of establishment of parallel TCP connections is not specified in the GridFTP protocol but implementation dependent.

Data transfer was continuously performed from the GridFTP server to the GridFTP client with TCP Reno. Moreover, for clarifying the fundamental characteristics of GridFTP-APT, we first performed simulation without background traffic. ns-2 simulator [19] with our GridFTP-APT implementation was used for simulation. Table 1 shows the parameter configuration used in simulation. Unless explicitly stated, parameters shown in Tab. 1 are used in the following simulations.

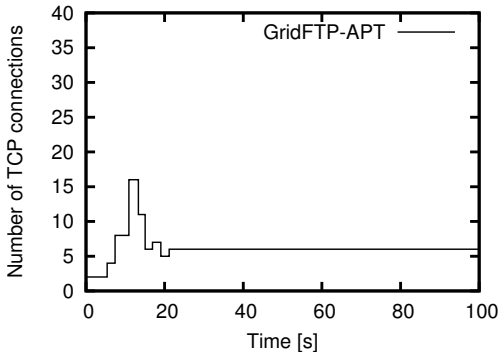


Fig. 5: The number of TCP connections in GridFTP-APT for $\tau = 10$ [ms]

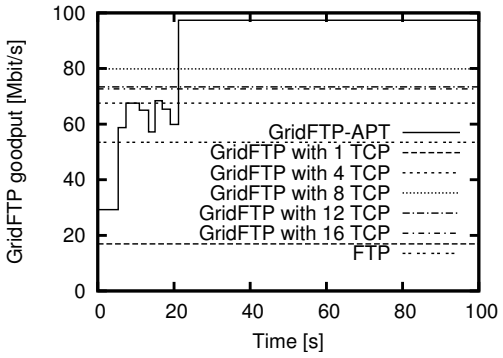


Fig. 6: Evolution of goodput in GridFTP-APT for $\tau = 10$ [ms]

The evolution of the number of parallel TCP connections of GridFTP-APT when the propagation delay of the bottleneck link is 10 [ms] is shown in Fig. 5. This figure shows that the number of parallel TCP connections of GridFTP-APT converges to $N = 6$ at approximately $t = 21$ [s]. In this case, the bracket was determined at approximately $t = 13$ [s]. After approximately 8 [s], the number of parallel TCP connections was optimized.

The evolution of goodput in GridFTP-APT in this scenario is shown in Fig. 6. For comparison purposes, GridFTP goodput in steady state when fixing the number of parallel TCP connections at 1, 4, 8, 12, and 16 are also plotted in the figure. Also, FTP goodput with a sufficiently large socket buffer (i.e., 10 [Mbyte]) is plotted in the figure. First, one can find that the optimal number of parallel TCP connections seems to exist between 4–16 from the GridFTP goodput with the fixed number of parallel TCP connections. GridFTP-APT reaches goodput of approximately 68 [Mbit/s] at approximately $t = 7$ [s]. After approximately 14 [s], the GridFTP goodput converges to 97 [Mbit/s]. Note that, since the buffer size of RED routers is small (i.e., 100 [packet]) in our simulations, the maximum goodput was approximately 97 [Mbit/s] even with the optimal number of parallel TCP connections. This indicates that GridFTP-APT can utilize the network resource quite effectively after approximately 21 [s] from starting the transfer. Namely, GridFTP-APT realizes significantly better performance than the original FTP and GridFTP.

Evolutions of the number of parallel TCP connections and the GridFTP goodput when the propagation delay of the bottleneck link is 20 [ms] are shown in Figs. 7 and 8, respectively.

Figure 7 shows that the number of parallel TCP connections converges at approximately $t = 25$ [s]. Although the propagation delay of the bottleneck link is larger than the previous case (Fig. 5), the convergence time of the number of parallel TCP connections does not become larger. One can find that the optimal number of parallel TCP connections is $N = 10$, which is different from that in the previous case ($N = 6$).

Figure 8 shows that GridFTP-APT can realize high

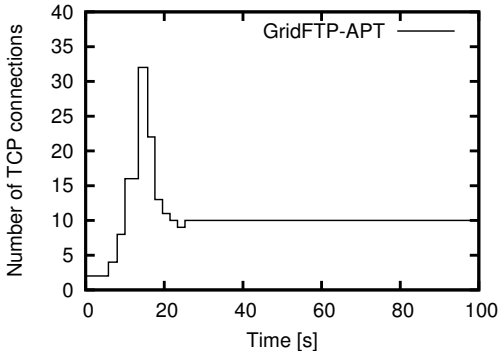


Fig. 7: The number of TCP connections in GridFTP-APT for $\tau = 20$ [ms]

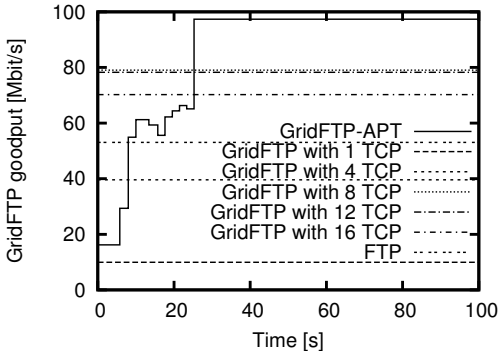


Fig. 8: Evolution of GridFTP goodput for $\tau = 20$ [ms]

goodput by appropriately adjusting the number of parallel TCP connections. Note that, in this simulation, due to small buffer of RED routers and a large propagation delay, the maximum GridFTP goodput was approximately 97 [Mbit/s] even with the optimal number of parallel TCP connections. Namely, this indicates that GridFTP-APT can utilize the network resource quite effectively regardless of the propagation delay of the bottleneck link.

Evolutions of the number of parallel TCP connections and the GridFTP goodput when the bottleneck link bandwidth is 1,000 [Mbit/s] and the propagation delay of the bot-

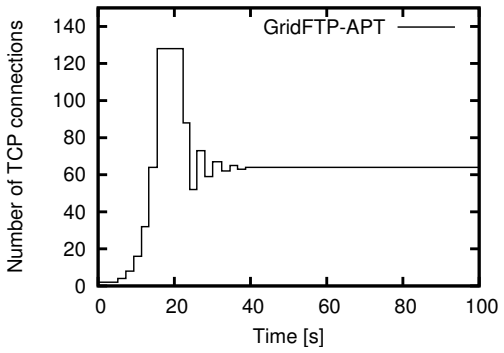


Fig. 9: Number of TCP connections in GridFTP-APT for $B = 1,000$ [Mbit/s], $\tau = 10$ [ms]

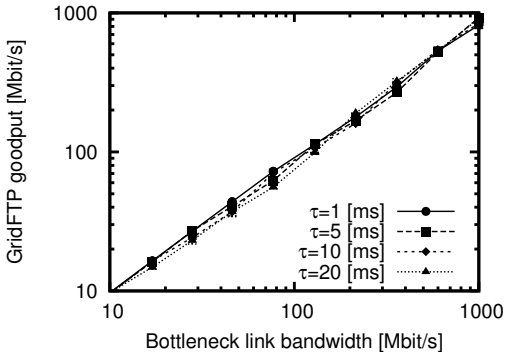


Fig. 11: Bottleneck link bandwidth vs. GridFTP goodput for τ = 1, 5, 10 and 20 [ms]

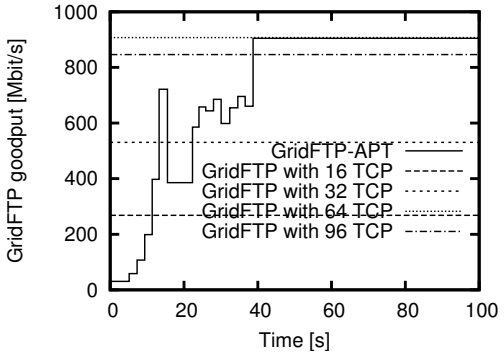


Fig. 10: Evolution of GridFTP goodput for $B = 1,000$ [Mbit/s], $\tau = 10$ [ms]

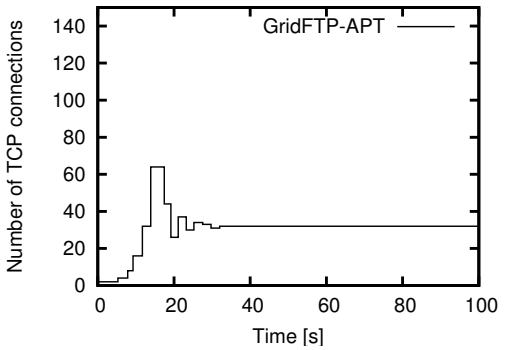


Fig. 12: Number of TCP connections in GridFTP-APT for $B = 1,000$ [Mbit/s], $\tau = 10$ [ms], transmission rate of UDP packets 500 [Mbit/s]

tleneck link is 10 [ms] are shown in Figs. 9 and 10, respectively. Figure 9 shows that the number of parallel TCP connections of GridFTP-APT converges to $N = 64$ at approximately $t = 38$ [s]. Figure 10 shows that GridFTP-APT can realize high goodput by appropriately adjusting the number of parallel TCP connections.

Note that, since the buffer size of RED routers is small (i.e., 100 [packet]) in our simulations, the maximum goodput was approximately 910 [Mbit/s] even with the optimal number of parallel TCP connections. Namely, this indicates that GridFTP-APT can utilize the network resource quite effectively even when the bottleneck link bandwidth is large.

4.2 Effect of Network Bandwidth and Delay

For investigating the effectiveness of GridFTP-APT in various network environments, we run simulations while changing the bandwidth B and propagation delay τ of the bottleneck link. Figure 11 shows the GridFTP goodput for τ = 1, 5, 10, and 20 [ms] and F = 10 [Gbyte] when changing the bottleneck link bandwidth B. Figure 11 clearly illustrates the effectiveness of GridFTP-APT; i.e., GridFTP-APT can efficiently utilize the bottleneck link bandwidth regardless of the bandwidth and/or propagation delay of the bottleneck link. Namely, GridFTP-APT can realize high throughput in various network environments.

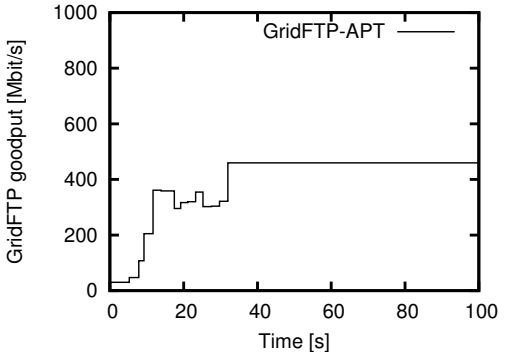


Fig. 13: Evolution of GridFTP goodput for $B = 1,000$ [Mbit/s], $\tau = 10$ [ms], transmission rate of UDP packets 500 [Mbit/s]

4.3 Effect of Router Buffer Size

For investigating the effectiveness of GridFTP-APT in various network environments, we run simulations while changing the buffer size L of RED routers. Figure 14 shows the GridFTP goodput in steady state for τ = 1, 5, 10, and

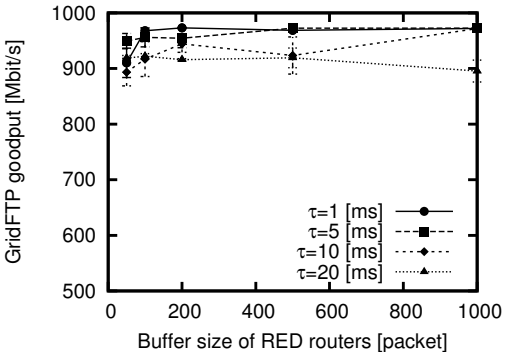


Fig. 14: Buffer size of RED router vs. GridFTP goodput for τ = 1, 5, 10 and 20 [ms]

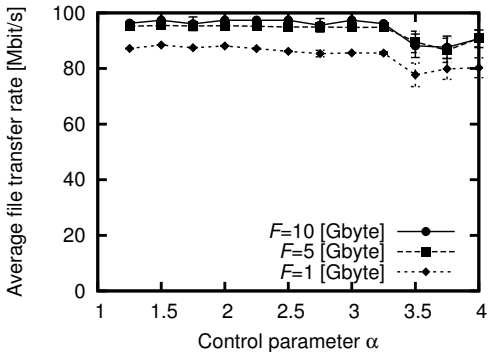


Fig. 15: GridFTP-APT control parameter α vs. average file transfer rate ($\tau = 10$ [ms])

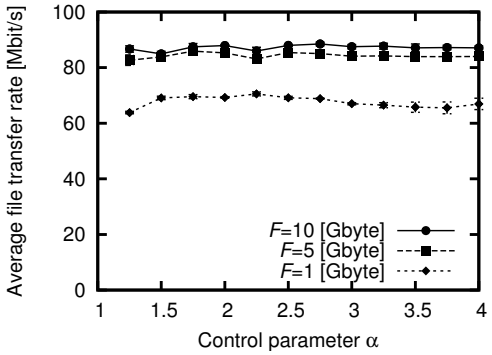


Fig. 16: GridFTP-APT control parameter α vs. average file transfer rate ($\tau = 50$ [ms])

20 [ms], B = 1,000 [Mbit/s], and F = 10 [Gbyte]. This figure shows that the GridFTP-APT performs effectively for different buffer sizes of RED routers.

4.4 Effect of GridFTP-APT Control Parameters

We investigate the effect of the configuration of GridFTP-APT control parameters on the performance of GridFTP-APT. Specifically, we clarify the effect of the configuration of the GridFTP-APT control parameter α, the target value of chunk transfer time Δ, and the initial number of parallel TCP connections N_0 on the GridFTP performance.

We use the topology in Fig. 4 for the following simulations. Unless explicitly stated, parameters shown in Tab. 1 are used in the following simulations.

First, we investigate the effect of the GridFTP-APT control parameter α on the GridFTP-APT performance. It is expected that the control parameter α affects the time and accuracy of bracket search in GridFTP-APT.

It is known that the goodput of parallel TCP connections depends on the propagation delay and bandwidth of a network, the TCP window size, and the number of parallel TCP connections [7]. In what follows, we only show simulation results for different propagation delays τ of the bottleneck link and file sizes F as system parameters due to the page limitation. Generally, the bottleneck link bandwidth B is also one of the important system parameters. However, we found from a number of simulations that the bottleneck link bandwidth B did not affect the configuration of the GridFTP-APT control parameters. Therefore, simulation results for different bottleneck link bandwidths are not included in this paper.

Figures 15 and 16 show the average file transfer rate of GridFTP when changing the GridFTP-APT control parameter α as 1.25–4 for $\tau = 10$ [ms] and $\tau = 50$ [ms], respectively. We repeated the simulation 20 times and calculated the average value of these results. In these figures, their confidence intervals are also plotted.

These figures show that the average file transfer rate of GridFTP depends on the GridFTP-APT control parameter α. For $\tau = 10$ [ms], the average file transfer rate of GridFTP degrades when α is large. This can be explained as follows. When the control parameter α is too large, GridFTP-APT increases the number of parallel TCP connections suddenly at the bracket search. The bracket becomes too wide so that the time for adjusting the optimal number of parallel TCP connections using the GSS algorithm becomes long. Consequently, GridFTP-APT cannot effectively utilize the network bandwidth until the number of TCP connections is optimized, so that the average file transfer rate degrades.

From these observations, we conclude that the control parameter α should be around 2–3 regardless of the propagation delay of a network.

Next, we investigate the effect of the GridFTP-APT control parameter Δ used for chunk size determination on the GridFTP-APT performance. Basically, chunk transfer time is approximately proportional to the control parameter Δ. For this reason, it is expected that the control parameter Δ directly affects the convergence time of the number of parallel TCP connections.

Figures 17 and 18 show the average file transfer rate of GridFTP when changing the GridFTP-APT control parameter Δ as 1–30 [s] for $\tau = 10$ [ms] and $\tau = 50$ [ms], respectively. We repeated the simulation 20 times and calculated the average value of these results. In these figures, their confidence intervals are also plotted. These figures show

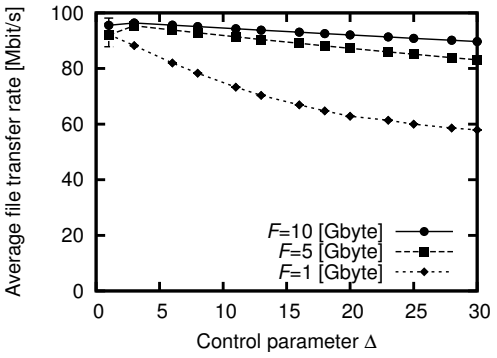


Fig. 17: GridFTP-APT control parameter Δ vs. average file transfer rate ($\tau = 10$ [ms])

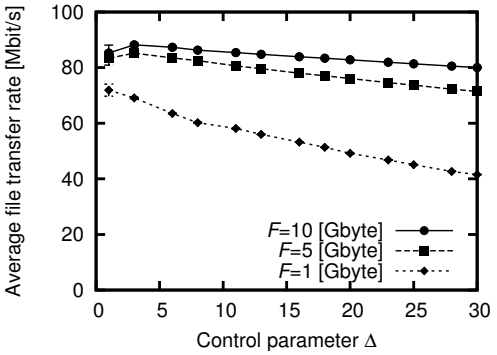


Fig. 18: GridFTP-APT control parameter Δ vs. average file transfer rate ($\tau = 50$ [ms])

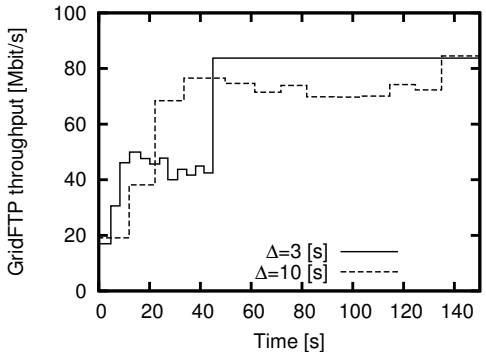


Fig. 19: Evolution of GridFTP throughput ($\tau = 50$ [ms])

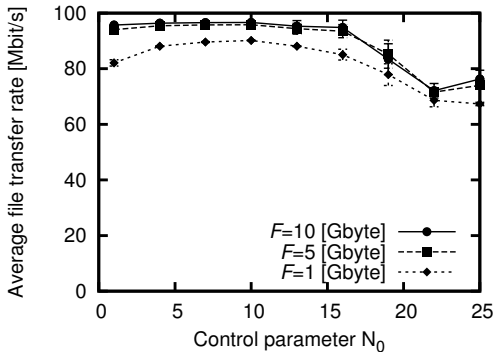


Fig. 20: GridFTP-APT control parameter Δ vs. average file transfer rate ($\tau = 10$ [ms])

that the average file transfer rate of GridFTP depends on the control parameter Δ. The average file transfer rate of GridFTP degrades with large Δ. This can be explained as follows. Each chunk transfer time becomes long with large Δ. Therefore, the time for optimizing the number of parallel TCP connections becomes long, so that the file transfer rate of GridFTP degrades.

Evolutions of the average file transfer rate of GridFTP for $\Delta = 3$ [s] and $\Delta = 10$ [s] are shown in Fig. 19. In this figure, evolutions of the GridFTP goodput on every chunk transfer are plotted. This figure shows that the time for optimizing the number of parallel TCP connections largely depends on the setting of the control parameter Δ.

Figure 18 shows that the average file transfer rate of GridFTP is low for a small Δ. This can be explained as follows. The GridFTP goodput cannot be measured correctly with a small Δ. Consequently, the number of parallel TCP connections cannot be optimized correctly, so that the average file transfer rate of GridFTP becomes low. One can find that when the control parameter Δ is approximately 3 [s], GridFTP-APT achieves the highest average file transfer rate regardless of the propagation delay of the bottleneck link and the file size.

From these observations, we conclude that the control parameter Δ should be set to approximately 3 [s] regardless of the propagation delay of the bottleneck link and the file size.

Next, we investigate the effect of the GridFTP-APT control parameter N_0 on the GridFTP-APT performance. It is expected that the initial value of the number of parallel TCP connections N_0 affects the time for optimizing the number of parallel TCP connections.

Figures 20 and 21 show average file transfer rates of GridFTP when changing the GridFTP-APT control parameter N_0 as 1–25 for $\tau = 10$ [ms] and $\tau = 50$ [ms], respectively. In these figures, their confidence intervals are also plotted. These figures show that the average file transfer rate of GridFTP depends on the GridFTP-APT control parameter N_0. These figure also show that the optimal setting of the control parameter N_0 depends on the propagation delay of the bottleneck link, but hardly depends on the file size. When the propagation delay of the bottleneck link is large (Fig. 21), GridFTP-APT can achieve high average file transfer rate with wide range of N_0.

From these observations, we conclude that N_0 should be set to approximately 4–8 regardless of the file size.

4.5 Effect of Background Traffic

We investigate the effect of the background traffic on the GridFTP-APT performance. Figure 22 shows the network

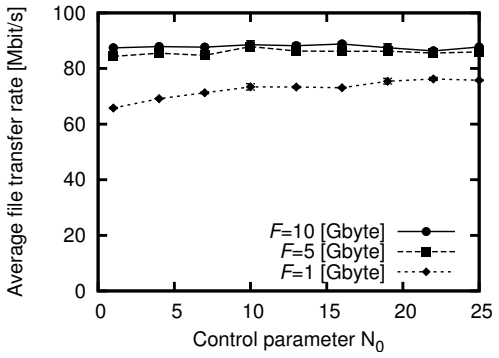


Fig. 21: GridFTP-APT control parameter Δ vs. average file transfer rate ($\tau = 50$ [ms])

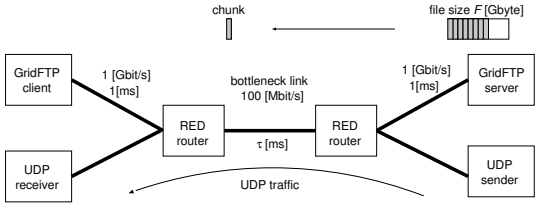


Fig. 22: Network topology used in simulation for investigating the effect of background traffic

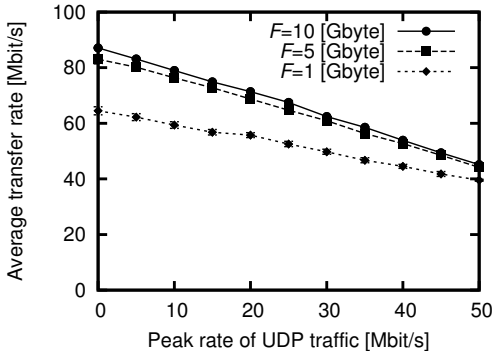


Fig. 23: Peak rate of UDP traffic vs. average transfer rate ($\tau = 50$ [ms])

model used in simulation. In this network model, a GridFTP server and a client are connected via two RED routers. Data transfer was continuously performed from the GridFTP server to the GridFTP client. In addition, a UDP sender and a UDP receiver are connected via the same two RED routers. As background traffic, the UDP sender transmits UDP packets at a fixed rate. The interval of UDP packets is exponentially distributed. The packet length of UDP packets is 1,500 [byte].

Figure 23 shows the average file transfer rate of GridFTP when changing the peak rate of UDP traffic as 0–50 [Mbit/s]. This figure shows that GridFTP-APT can utilize the available bandwidth effectively regardless of the amount of the background traffic.

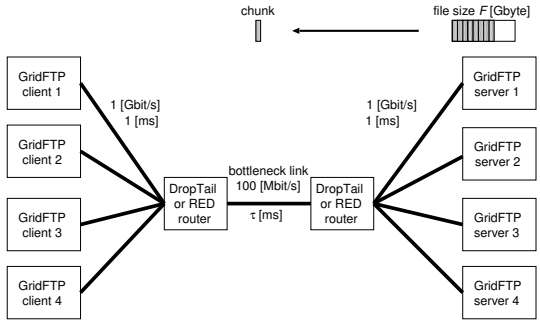


Fig. 24: Network topology used in simulation for investigating the fairness among multiple GridFTP sessions

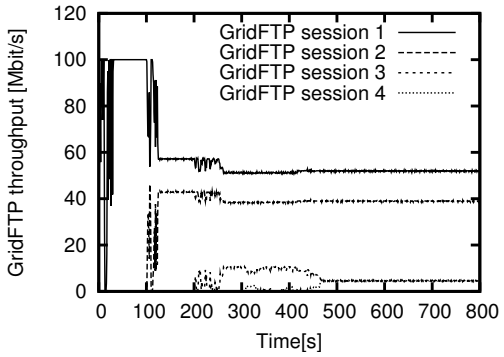


Fig. 25: Evolution of GridFTP throughput with DropTail routers ($\tau = 10$ [ms])

4.6 Fairness among GridFTP sessions

Finally, we investigate fairness among GridFTP sessions when multiple GridFTP sessions compete for the shared network resource. We measure evolutions of the throughput of GridFTP sessions, and analyze fairness among GridFTP sessions.

Figure 24 shows the network model used in simulation. In this network model, four GridFTP servers and four clients are connected via either two DropTail routers or two RED routers.

First, we show the results with DropTail routers. Figure 25 shows evolution of the throughput of each GridFTP session. In this figure, each GridFTP session initiated every 100 [s]. This figure shows that fairness among GridFTP sessions degrades when the number of GridFTP sessions increases. GridFTP-APT has no mechanism to adapt to the change in the network bandwidth, so that fairness among GridFTP sessions degrades. TCP itself has a problem when combined with DropTail routers; it is known that with DropTail routers, fairness among TCP connections degrades when the buffer size of the router is small [20].

By replacing DropTail routers to RED routers, fairness among GridFTP sessions can be improved. Figure 26 shows evolutions of the throughput of each GridFTP ses-

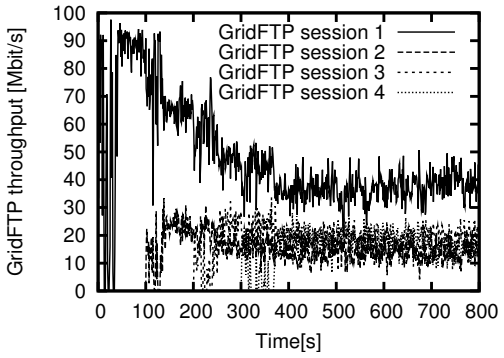


Fig. 26: Evolution of GridFTP throughput with RED routers ($\tau = 10$ [ms])

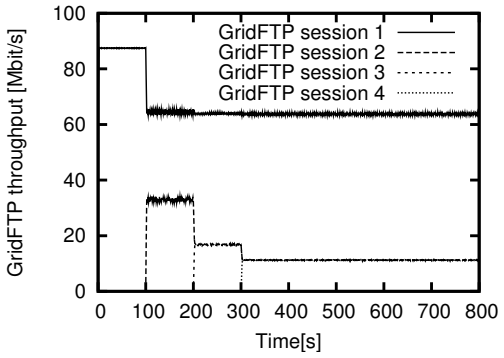


Fig. 27: Evolution of GridFTP throughput with DropTail routers ($\tau = 10$ [ms], $N = 4$)

sion with RED routers. In this figure, the queue discipline of routers is RED and other parameters are the same value used for Fig. 25. Figure 26 shows that fairness among GridFTP sessions is improved largely with RED routers. This is because TCP connections in GridFTP sessions operate stably with RED routers. However, the GridFTP session that starts transfer at 0 [s] gains large bandwidth even with RED routers. To improve fairness, we should modify the GridFTP-APT algorithm in the future.

Fairness degradation of GridFTP-APT is a side-effect of achieving high performance. For instance, the original GridFTP realizes better fairness than GridFTP-APT. Figures 27 and 28 show evolutions of the original GridFTP throughput with DropTail routers and RED routers, respectively. GridFTP-APT gains higher total throughput (i.e., 100 [Mbit/s] and 90 [Mbit/s] in Figs. 25 and 26) than the original GridFTP (i.e., 87 [Mbit/s] and 87 [Mbit/s] in Figs. 27 and 28). From these observations, we believe GridFTP-APT is suitable for bandwidth-intensive applications in a long-fat networks. GridFTP-APT has no mechanism to adapt to change in the network configuration once the number of parallel TCP connections is optimized. But modest change in the available bandwidth should be absorbed by the congestion control mechanism of TCP.

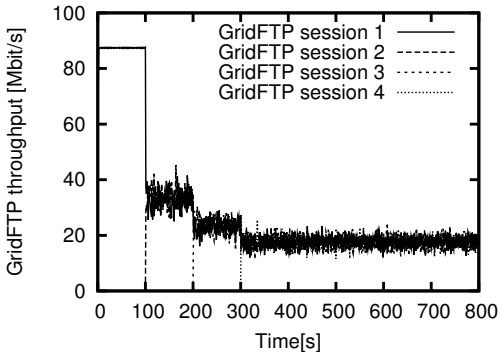


Fig. 28: Evolution of GridFTP throughput with RED routers ($\tau = 10$ [ms], $N = 4$)

5. Conclusion

In this paper, we have proposed GridFTP-APT, an automatic parallelism tuning mechanism for GridFTP, mainly focusing on the parallel data transfer feature for GridFTP. GridFTP-APT utilizes the fact that GridFTP goodput is a convex function for the number of parallel TCP connections. GridFTP-APT searches for the optimal number of parallel TCP connections using the GSS algorithm, one of numerical computation algorithms for a maximization problem. In this paper, we have shown that GridFTP-APT realized high throughput in several network environments.

As future work, we are planning to further improve GridFTP-APT performance. GridFTP-APT works quite effectively for a large data transfer in a long-fat network. However, as we have found from simulations, GridFTP-APT lacks adaptability to change in the amount of background traffic. We are therefore planning to design an extension to GridFTP-APT for detecting and adapting to change in the amount of background traffic.

Acknowledgements

We would like to express our appreciation to Prof. Masayuki Murata for joining meaningful discussions. This work is supported by the NAREGI (National Research Grid Initiative) Project from the Ministry of Education, Culture, Sports, Science and Technology, Japan.

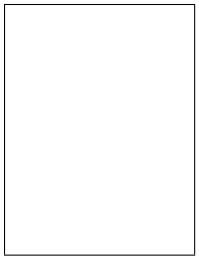
References

[1] The Globus Project, "GridFTP: universal data transfer for the Grid," White Paper, Sept. 2003. available at `http://www.globus.org/toolkit/docs/2.4/datagrid/deliverables/C2WPdraft3.pdf`.

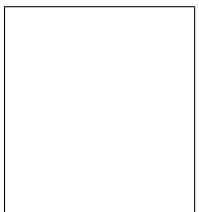
[2] W. Allcock *et al.*, "GridFTP: Protocol extensions to FTP for the Grid," OGF Document Series GFD.20, April 2003. Also available as `http://www.ggf.org/documents/GFD.20.pdf`.

[3] The Globus Project, "GridFTP update January 2002," 2002. available at `http://www.globus.org/toolkit/docs/2.4/datagrid/deliverables/GridFTP-Ov%erview-200201.pdf`.

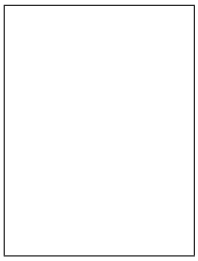
[4] T.J. Hacker *et al.*, "The end-to-end performance effects of parallel TCP sockets on a lossy wide-area network," Proceedings of the 16th IEEE-CS/ACM International Parallel and Distributed Processing Symposium (IPDPS), pp.434–443, April 2002.
[5] L. Qiu *et al.*, "On individual and aggregate TCP performance," Proceedings of Internetl Conference on Network Protocols, pp.203–212, Oct. 1999.
[6] S. Thulasidasan *et al.*, "Optimizing GridFTP through dynamic right-sizing," Proceedings of IEEE International Symposium on High Performance Distributed Computing, pp.14–23, June 2003.
[7] T. Ito *et al.*, "On parameter tuning of data transfer protocol GridFTP in wide-area Grid computing," Proceedings of Second International Workshop on Networks for Grid Applications (GridNets 2005), pp.415–421, Oct. 2005.
[8] T. Ito *et al.*, "Automatic parameter configuration mechanism for data transfer protocol GridFTP," Proceedings of the 2006 International Symposium on Applications and the Internet (SAINT 2006), pp.32–38, Jan. 2006.
[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] "Open Grid Forum." `http://www.ogf.org/`.
[13] "Globus Toolkit." available at `http://www.globus.org/`.
[14] I. Mandrichenko *et al.*, "GridFTP v2 protocol description," OGF Document Series GFD.47, May 2005. Also available as `http://www.ggf.org/documents/GFD.47.pdf`.
[15] W.H. Press *et al.*, Numerical Recipes in C: The Art of Scientific Computing, Cambridge University Press, 1992.
[16] G.E. Forsythe, M.A. Malcom, and C.B. Moler, Computer Methods for Mathematical Computations, Prentice-Hall, 1977.
[17] N. Ehsan and M. Liu, "Analysis of TCP transient behavior and its effect on file transfer latency," Proceedings of IEEE International Conference on Communications (ICC2003), pp.1806–1811, May 2003.
[18] E. Altman, C. Barakat, E. Laborde, P. Brown, and D. Collange, "Fairness analysis of TCP/IP," Proceedings of the 39th IEEE Conference on Decision and Control, pp.61–66, Dec. 2000.
[19] "The network simulator – ns2." available at `http://www.isi.edu/nsnam/ns/`.
[20] S. Floyd and V. Jacobson, "Random early detection gateways for congestion avoidance," IEEE/ACM Transactions on Networking, vol.1, no.4, pp.397–413, Aug. 1993.



Takeshi Ito received B.E. degree in the Information and Computer Sciences from Osaka University. He also received the Master of Information Science and Technology degree from Osaka University, Osaka, Japan, in 2006. He is currently a Ph.D. candidate at Department of Information Networking, Graduate School of Information Science and Technology, Osaka University, Japan. His research work is in the area of Grid networks. He is a student member of IEEE and Institute of Electronics, Information, and Computer Engineers of Japan (IEICE).



Hiroyuki Ohsaki received the M. E. degree in the Information and Computer Sciences from Osaka University, Osaka, Japan, in 1995. He also received the Ph. D. degree from Osaka University, Osaka, Japan, in 1997. He is currently an associate professor at Department of Information Networking, Graduate School of Information Science and Technology, Osaka University, Japan. His research work is in the area of traffic management in high-speed networks. He is a member of IEEE and Institute of Electronics, Information, and Computer Engineers of Japan (IEICE).



Makoto Imase received his B.E. and M.E. degrees in information engineering from Osaka University in 1975 and 1977, respectively. He received D.E. degree from Osaka University in 1986. From 1977 to 2001, he was engaged Nippon Telegraph and Telephone Corporation (NTT). He has been a Professor of Graduate School of Information Science and Technology at Osaka University since 2002. His research interests are in the area of information networks, distributed systems and graph theory. He is a member of IPSJ, JSIAM, and IEICE.