Performance Evaluation of Data Transfer Protocol GridFTP for Grid Computing

Hiroyuki Ohsaki and Makoto Imase

***Abstract*— In Grid computing, a data transfer protocol called *GridFTP* has been widely used for efficiently transferring a large volume of data. Currently, two versions of GridFTP protocols, GridFTP version 1 (GridFTP v1) and GridFTP version 2 (GridFTP v2), have been proposed in the GGF. GridFTP v2 supports several advanced features such as data streaming, dynamic resource allocation, and checksum transfer, by defining a transfer mode called *X-block* mode. However, in the literature, effectiveness of GridFTP v2 has not been fully investigated. In this paper, we therefore quantitatively evaluate performance of GridFTP v1 and GridFTP v2 using mathematical analysis and simulation experiments. We reveal the performance limitation of GridFTP v1, and quantitatively show effectiveness of GridFTP v2. Through several numerical examples, we show that by utilizing the data streaming feature, the average file transfer time of GridFTP v2 is significantly smaller than that of GridFTP v1.**

***Keywords*— Grid Computing, GridFTP, Performance Evaluation, Queuing Theory**

I. Introduction

TO realize effective Grid computing on a wide-area network, various issues must be resolved. In Grid computing, the amount of data transferred between distributed computers is enormous and round-trip time of a network is large. For example, when analyzing high-energy physical phenomena, the size of data for a single measurement runs into several Tbytes [1]. Additionally, round-trip time of a wide-area network sometimes reaches several hundreds milliseconds.

Thus, for high-performance Grid computing, a data transfer protocol that can effectively transfer a large volume of data over a wide-area network has been awaited [2]. However, FTP (File Transfer Protocol) [3] and HTTP (Hyper Text Transfer Protocol) [4], transfer protocols widely used on the Internet at present, were not been designed to deal with such large data transfer. If such existing data transfer protocols are used, large data transfer will be lengthy due to the limits inherent in their processing overheads and the transport layer protocol, TCP (Transmission Control Protocol) [2].

GridFTP was developed as a protocol that aims to effectively transfer a large amount of data in Grid computing [5, 6]. GridFTP extends the widely used FTP by circumventing the problems with TCP. In particular, GridFTP supports a feature to negotiate the size of the TCP socket buffer (i.e., automatic negotiation of the TCP socket buffer size) as well as a feature to use multiple TCP connections to transfer a single file in parallel (i.e., parallel data transfer). Because of these features, GridFTP has proved to be more effective than traditional FTP in transferring data especially over a network with large bandwidth-delay product [7].

Authors are with Graduate School of Information Science and Technology, Osaka University, Japan (e-mails: {oosaki,imase}@ist.osaka-u.ac.jp).

The Global Grid Forum (GGF) has completed the specification development of the GridFTP version 1 (GridFTP v1) [5] and has been actively standardizing the GridFTP version 2 (GridFTP v2) [6].

Several performance evaluations have been undertaken for GridFTP v1. For instance, the literature [7] modeled GridFTP v1 as a continuous-time system by multiplexing fluid-flow approximation models of TCP. It derived the throughput and the packet loss probability of GridFTP using the model. It also derived an optimal configuration of the number of parallel TCP connections and the TCP socket buffer size that maximizes the GridFTP throughput. The literature [8, 9] proposed an automatic parameter configuration mechanism for parallel data transfer of GridFTP. In particular, it demonstrated that the GridFTP goodput can be increased by adjusting the number of TCP connections based on the current GridFTP goodput and round-tip time. However, the effectiveness of GridFTP v2 has not been fully investigated.

In this paper, we clarify the performance limitation of GridFTP v1 and quantitatively show the effectiveness of GridFTP v2 through their performance comparison using mathematical analysis and simulation experiments. We illustrate the effectiveness of the data streaming of GridFTP v2 on a network with a single server and a single client by using a mathematical analytic method. Specifically, we model GridFTP v1 as an $M/G/1$ queue [10] and GridFTP v2 an $M/G/1-PS$ queue [10] to derive and compare the means and standard deviations of their file transfer time.

The structure of this paper is as follows. Firstly, Section II outlines GridFTP. We explain in particular both data streaming and dynamic resource allocation, which are the features newly introduced in GridFTP v2. In Section III, we evaluate the effectiveness of the data streaming of GridFTP v2 by using a mathematical analytic method. Lastly, Section IV describes a summary of this paper and future tasks.

II. Data Transfer Protocol GridFTP

GridFTP is a protocol designed for effective large amount data transfer in Grid computing [5, 6]. It is an extension of FTP protocol, which has been widely used on the Internet at present.

GGF has completed the specification development of GridFTP v1 [5] and has been actively standardizing GridFTP v2 [6]. GridFTP v1 and GridFTP v2 support the features listed in Tab. I, respectively.

TABLE I
COMPARISON OF GRIDFTP V1 AND GRIDFTP V2 FEATURES

	GridFTP v1	GridFTP v2
Automatic negotiation of TCP socket buffer size	○	○
Parallel data transfer	○	○
Third-party control of data transfer	○	○
Partial file transfer	○	○
Security	○	○
Reliable data transfer	○	○
Data streaming	×	○
Dynamic resource allocation	×	○
GET/PUT commands	×	○
EOF communication in stream mode	×	○
Checksum transmission	×	○

GridFTP v1 extended FTP protocol to a minimal extent to enable effective data transfer. This means that it also inherited several problems caused by the limitations of FTP [11]. For example, since GridFTP succeeded to the semantics of FTP, it does not allow pipelined transfer of multiple files on the same data channels. Another limitation is that it cannot adjust the number of data channels during file transfer.

On the other hand, GridFTP v2 defines X-block mode, which is further extension of Extended block mode (E-block mode) of GridFTP v1 [6]. With X-block mode, GridFTP v2 supports features to transfer multiple files over the same data channels in a pipelined manner (i.e., *data streaming*) and to dynamically adjust the number of data channels during file transfer, or in other words, the number of TCP connections used for parallel data transfer (i.e., *dynamic resource allocation*). It also supports a feature to detect data corruption by file or by data block of X-block mode (i.e., *checksum transmission*).

In what follows, we present this newly defined X-block mode of GridFTP v2. Especially, we explain two features enabled by X-block mode: data streaming and dynamic resource allocation. GridFTP v2 supports several features besides them. These features include: *GET/PUT commands* to specify files to be transferred and establish data channels simultaneously; *EOF communication in stream mode* to send a signal of the end of file transfer to the receiver host in stream mode; and *checksum transmission* as stated above (Tab. I). For the detail of these features, please refer to the literature [6].

The following is the explanation on data streaming and dynamic resource allocation.

- Data streaming
  GridFTP v2 has implemented a feature called data streaming [6], which transfers multiple files on the same data channels in a pipelined manner with the use of the identifier of transaction, TID (Transaction ID), of X-block mode (Fig. 1).
  Since GridFTP v1 succeeded to the semantics of FTP, it cannot allow pipelined transfer of multiple files on the same data channels [5, 6]. In other words, GridFTP v1 cannot transfer multiple files over the same channels at the same time. In GridFTP v1, each file is transferred after the transfer of the previous file is finished.
  X-block mode of GridFTP v2 resolves this problem with GridFTP v1 by adding Transaction ID to the header of data block. GridFTP servers/clients specify different transaction IDs for different transferred files so that they are transferred over the same data channels in a pipelined manner. This feature of GridFTP v2 can diminish the overhead for sequential file transfer in GridFTP v1, and therefore the efficiency of file transfer can be improved.
- Dynamic resource allocation
  GridFTP v2 has also implemented dynamic resource allocation, in which the number of data channels can be adjusted with the use of READY/CLOSE/BYE commands during file transfer (Fig. 2).
  GridFTP v1 cannot adjust the number of data channels during file transfer, since it expanded FTP to support parallel data transfer. Additionally, GridFTP v1 has a limitation that establishment of data channels is allowed only to sender GridFTP servers/clients, to avoid race conditions between data channels [6].
  In contrast, GridFTP v2 has incorporated READY/CLOSE/BYE commands to dynamically adjust the number of data channels during file transfer. Furthermore, not only sender but also receiver GridFTP servers/clients are able to establish data channels.

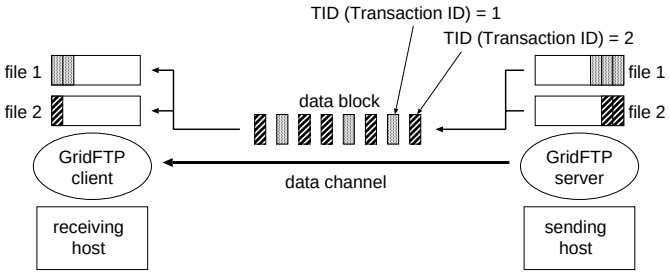

Fig. 1: Pipelined file transfer using the data streaming: GridFTP v2 can transfer multiple files on the same data channels using the TID (Transaction ID) of X-block mode.

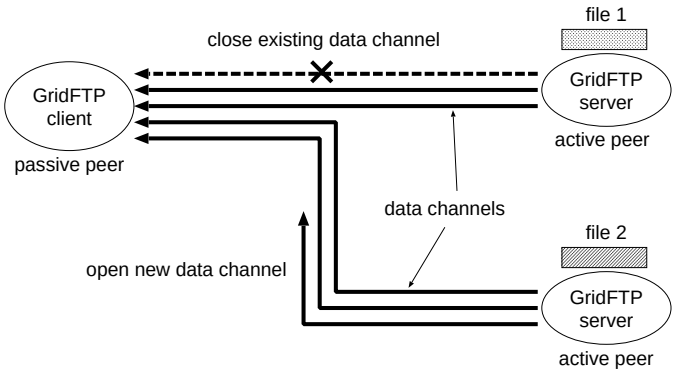

Fig. 2: Dynamic data channel allocation using the dynamic resource allocation: GridFTP v2 can adjust the number of data channels using READY/CLOSE/BYE commands during file transfer.

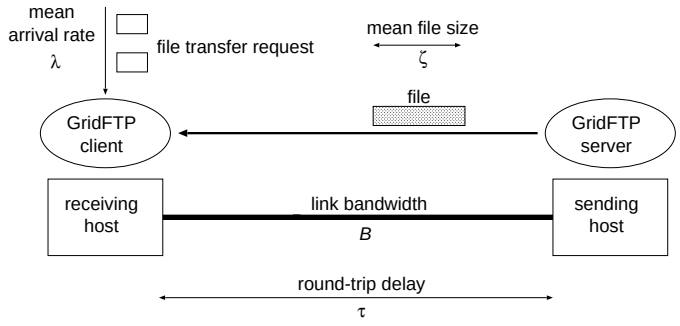

Fig. 3: Analytic model for the data streaming: files are transferred from the GridFTP server to the GridFTP according to the Poisson arrival of transfer requests.

Incidentally, only GridFTP servers/clients that initially established data channels, i.e., active peers, can open additional data channels in GridFTP v2. The other GridFTP servers/clients, i.e., passive peers, can only shut down open data channels.

III. Performance Evaluation of Data Streaming

In this chapter, we demonstrate the effectiveness of the data streaming of GridFTP v2 on a network consisting of a server and a client using a mathematical analytic method. We model both GridFTP v1 and GridFTP v2 as queuing systems to derive the means and standard deviations of their file transfer time.

A. Analysis

The analytic model is shown in Fig. 3. Suppose files are transferred from a GridFTP server to a GridFTP client. Let B denote the link bandwidth and τ denote the round-trip time between them. We assume that the occurrence of requests for file transfer is a Poisson with an arrival rate of λ. We also assume that the size of the files transferred from a GridFTP server to a GridFTP client follows exponential distribution with mean ζ.

First, we focus on GridFTP v1. In GridFTP v1, if the GridFTP client requests for file transfer, the GridFTP server sequentially transfers files to that client using RETR and ERET commands. Since GridFTP cannot process commands in a pipelined manner, there is a delay of at least the round-trip time τ after transferring a file until the transfer of the subsequent file starts. Given the delay between two consecutive file transfers is τ (that is, given each subsequent file transfer instantaneously starts upon a request), the file transfer of GridFTP v1 can be modeled as an M/G/1 queue.

Let $F_x(x)$ denote the probability distribution function of the service time $\tilde{x}$ (the time required for a single file transfer except the waiting time in queue). We define $\mu \equiv B/\zeta$. Assuming that the bandwidth between the GridFTP server and client, B, is constant, $F_x(x)$ is given by the equation

$$F_x(x) \quad = \quad 1 - e^{-\mu\,(x-\tau)}. \tag{1}$$

This yields the probability density function

$$f_x(x) \quad \equiv \quad \frac{dF_x(x)}{dx} = \mu e^{-\mu\,(x-\tau)}. \tag{2}$$

Thus, the first and second moments of the service time $\tilde{x}$ are

$$\overline{x^1} \quad \equiv \quad \int_0^\infty x f_x(x) dx = \frac{e^{\mu\,\tau}}{\mu}, \tag{3}$$

$$\overline{x^2} \quad \equiv \quad \int_0^\infty x^2 f_x(x) dx = \frac{2\,e^{\mu\,\tau}}{\mu^2}. \tag{4}$$

The mean system waiting time T of an $M/G/1$ queue is

$$T \quad = \quad \overline{x^1} + \frac{\lambda\,\overline{x^2}}{2\,(1-\lambda/\mu)}. \tag{5}$$

From Eqs. (3) and (4), the mean file transfer time of GridFTP v1, T, is given by

$$T \quad = \quad \frac{e^{\mu\,\tau}\,\lambda}{(1-\lambda/\mu)\;\mu^2} + \frac{e^{\mu\,\tau}}{\mu}. \tag{6}$$

On the other hand, the nth moment of the system waiting time of an $M/G/1$ queue, $\overline{s^n}$, is given by

$$\overline{s^n} \quad = \quad \sum_{i=0}^{n} \left(\begin{array}{c} n \\ i \end{array} \right) \overline{w^{n-i}}\,\overline{x^i}, \tag{7}$$

where

$$\overline{w^1} \quad = \quad \frac{\lambda\,\mu\,\overline{x^2}}{2(\mu-\lambda)}, \tag{8}$$

$$\overline{w^2} \quad = \quad \frac{\lambda\,\mu\,\left(3\,\lambda\,\mu\,\overline{x^2}^2 - 2\,\overline{x^3}(\mu-\lambda)\right)}{6\,(\mu-\lambda)^2}. \tag{9}$$

Therefore, the standard deviation of the file transfer time of GridFTP v1, σ, is given by

$$\begin{aligned} \sigma \quad &\equiv \quad \sqrt{\overline{s^2} - \overline{s^1}^2} \\ &= \sqrt{\frac{e^{\mu\,\tau}\;(2(\mu-\lambda) - e^{\mu\,\tau}\;(\mu - 2\,\lambda))}{(\mu-\lambda)^2\,\mu}} \end{aligned} \tag{10}$$

Next, we focus on GridFTP v2. In GridFTP v2, it is possible to simultaneously transfer multiple files from the GridFTP server to the client by specifying the Transaction ID, which is unique to each file, in the header of data block. Therefore, the file transfer of GridFTP v2 can be modeled as an M/G/1-PS queue.

Again, assuming that the link bandwidth between the GridFTP server and client, B, is constant, the mean service time (the time required for a single file transfer), x, is given by

$$x \quad = \quad \frac{\zeta}{B}. \tag{11}$$

For an $M/G/1-PS$ queue, $T(x)$, the mean system waiting time of a customer with the service time x, is

$$T(x) \quad = \quad \frac{\mu\,x}{\mu-\lambda}. \tag{12}$$

The mean file transfer time of GridFTP v2, T, is therefore given by

$$T \quad = \quad \int_0^\infty T(x) f_x(x) dx, \tag{13}$$

TABLE II

PARAMETER CONFIGURATION USED IN NUMERICAL EXAMPLES

Link bandwidth	B	100	[packet/ms]
Round-trip time	τ	10	[ms]
Traffic load	ρ	0.9	
Mean file size	ζ	5000	[packet]

where $f_x(x)$ denotes the probability density function of the file transfer time $\tilde{x}$. Since the file size follows exponential distribution with mean ζ, it follows that

$$f_x(x) \quad = \quad \mu e^{-\mu x}. \tag{14}$$

On the other hand, in an $M/G/1-PS$ queue, the variance of the system waiting time for a customer with the service time x is given by

$$\sigma^2(x) \quad = \quad \frac{2\rho x}{\mu(1-\rho)^3} - \frac{2\rho}{1-\rho}\left[1 - e^{-\mu(1-\rho)x}\right]. \tag{15}$$

Therefore, the standard deviation of the file transfer time of GridFTP v2 is given by

$$\sigma^2 \quad \equiv \quad \int_0^\infty \sigma(x) f_x(x) dx$$
$$= \int_0^\infty \frac{1}{\mu e^{\mu x}} \frac{2\left\{\lambda(e^{(\lambda-\mu)x} - 1)(\lambda-\mu)^2 + \mu^2 x\right\}}{(1-\lambda/\mu)^3} dx. \tag{16}$$

B. Numerical Examples

We demonstrate the effectiveness of the data streaming of GridFTP v2 by several numerical examples. The parameter configuration used in the numerical examples is listed in Tab. II. For these numerical examples, we obtained the mean occurrence rate of file transfer requests, λ, corresponding to our specified traffic load $\rho(\equiv \lambda/\mu)$. In the following numerical examples, we used the parameters listed in Tab. II unless stated otherwise.

We first look at the impact of the link bandwidth on the average file transfer time of GridFTP v1 and GridFTP v2. Figure 4 shows the average file transfer time T of GridFTP v1 and GridFTP v2 when the link bandwidth B was changed in the range of 1–1000 [packet/ms].

It can be seen from Fig. 4 that the average file transfer time of GridFTP v2 is much smaller than that of GridFTP v1. The difference in the average file transfer time between GridFTP v1 and GridFTP v2 is significant when the link bandwidth is large. This can be explained as follows. As the link bandwidth becomes larger, the required time for file transfer decreases. However, overhead, i.e., delay in processing successive file transfer, hinges on round-trip time in GridFTP v1. Therefore, the overhead of GridFTP v1 remains large relative to that of GridFTP v2 even when the link bandwidth becomes large. This seems to have resulted in the difference in the file transfer time.

The standard deviation of the file transfer time σ is shown in Fig. 5. This figure shows that GridFTP v2 realizes much smaller variation in the file transfer time than GridFTP v1 does. With the data streaming, GridFTP v2 transfers multiple files over the same data channels in a pipelied manner, leading

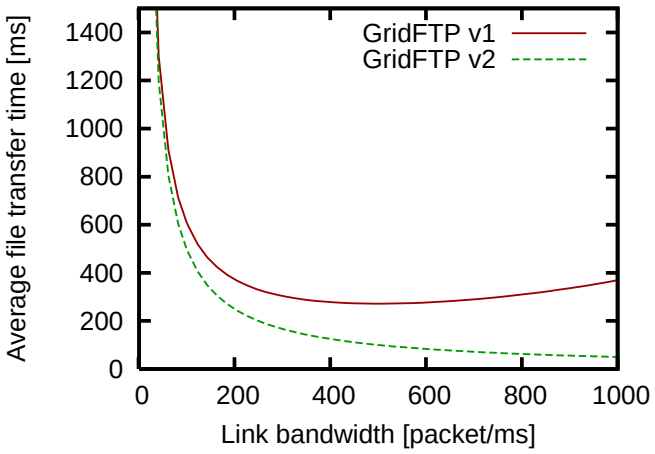

Fig. 4: Link bandwidth B vs. average file transfer time T, showing the average file transfer time of GridFTP v2 is much smaller than that of GridFTP v1.

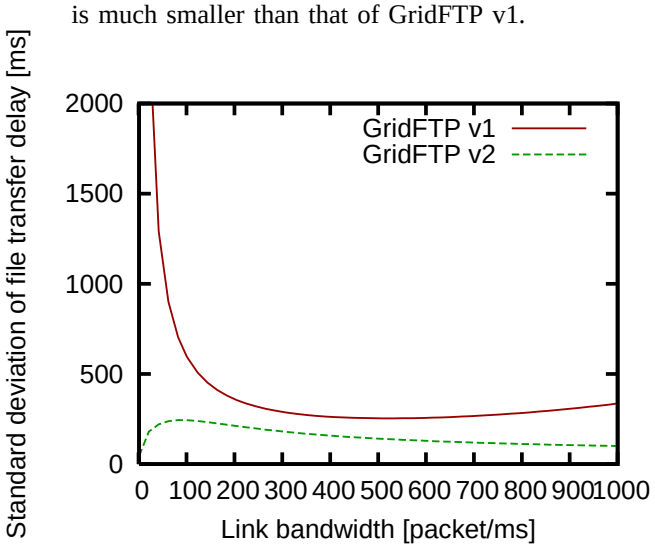

Fig. 5: Link bandwidth B vs. standard deviation of file transfer time σ, showing GridFTP v2 realizes much smaller variation in the file transfer time than GridFTP v1 does.

significantly small variation in the file transfer time. This result clearly indicates superiority of GridFTP v2 over GridFTP v1.

Figure 6 presents the average file transfer time T of GridFTP v1 and GridFTP v2 when the round-trip time τ was changed in the range of 1–90 [ms]. Also, Fig. 7 presents the standard deviation of the file transfer time σ. Figure 6 shows that the average file transfer time increases in GridFTP v1 as the round-trip time τ becomes larger. In contrast, the average file transfer time is constant in GridFTP v2 regardless of the round-trip time τ. Thus, it can be concluded that the data streaming of GridFTP v2 is effective especially on a wide-area network with large round-trip time.

Lastly, Fig. 8 presents the average file transfer time T of GridFTP v1 and GridFTP v2 when the traffic load ρ was changed in the range of 0.8–1.0. Also, Fig. 7 presents the

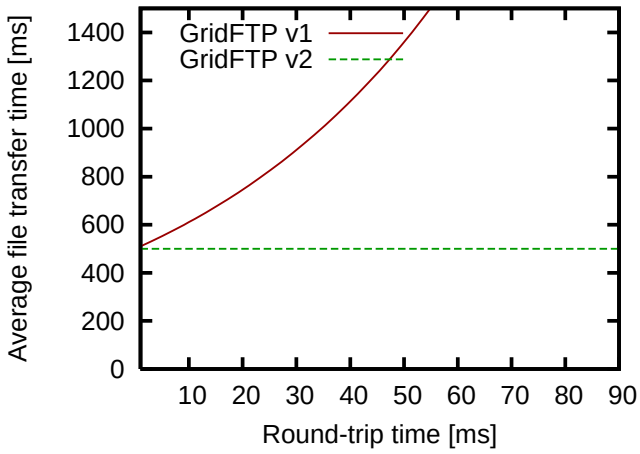

Fig. 6: Round-trip time τ vs. average file transfer time T, showing the average file transfer time of GridFTP v2 is constant regardless of the round-trip time τ.

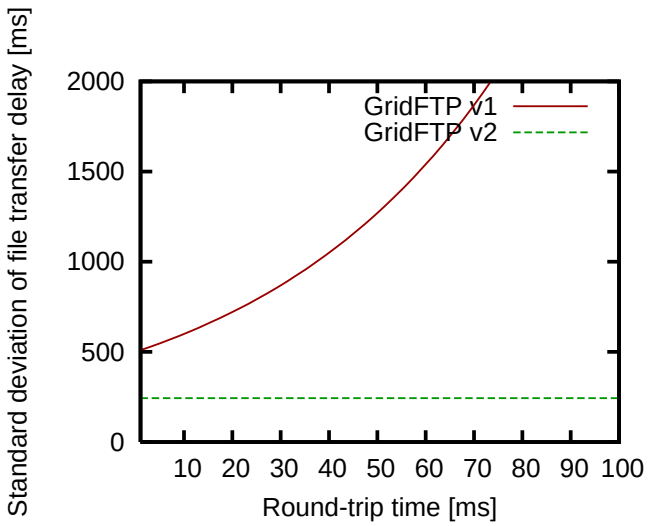

Fig. 7: Round-trip time τ vs. standard deviation of file transfer time σ, showing variation in the file transfer time of GridFTP v2 is constant regardless of the round-trip time τ.

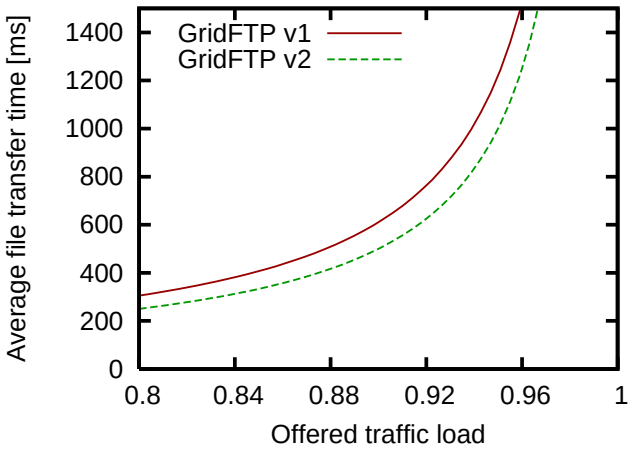

Fig. 8: Offered traffic load ρ vs. average file transfer time T, showing the average file transfer times T of both GridFTP v1 and GridFTP v2 rapidly increase as the traffic load ρ becomes larger.

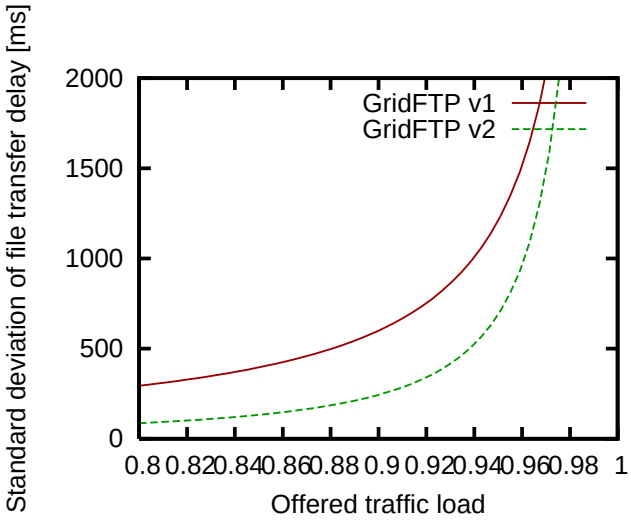

Fig. 9: Offered traffic load ρ vs. standard deviation of file transfer time σ, showing variation in the file transfer delay of GridFTP v2 is smaller than that of GridFTP v1.

standard deviation of the file transfer time σ. In Fig. 8, the average file transfer times T of both GridFTP v1 and GridFTP v2 rapidly increase as the traffic load ρ becomes larger. The difference in the average file transfer times of GridFTP v1 and GridFTP v2 increases as the traffic load ρ becomes larger. However, the ratio between the average file transfer time of GridFTP v1 and that of GridFTP v2 is approximately constant regardless of the traffic load ρ. In summary, while the data streaming of GridFTP v2 is effective in reducing average file transfer time, it does not have the effect of improving the tolerance to traffic load.

IV. Summary and Future Tasks

In this paper, we clarified the performance limitation of GridFTP v1 and quantitatively showed the effectiveness of GridFTP v2 through their performance comparison using mathematical analysis and simulation experiments. We showed the effectiveness of the data streaming of GridFTP v2 by using a mathematical analytic method.

Our future tasks include the investigation into the effectiveness of other features supported in GridFTP v2. In particular, it is necessary to evaluate the effectiveness of GET/PUT commands, EOF communication in stream mode, and checksum transmission.

ACKNOWLEDGMENTS

We would like to express our appreciation to Prof. Masayuki Murata and Mr. Yasuhiko Yoshimura of the Graduate School of Information Science and Technology, Osaka University, for joining meaningful discussions and theoretical arguments.

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] H. B. Newman, "Data intensive Grids and networks for high energy and nuclear physics," *InterAct Magazine*, Sept. 2002, also available as http://netlab.caltech.edu/FAST/references/InterAct_NetworksGridsforHEP_%hbn091502.doc.

[2] V. Sander *et al.*, "Networking issues for Grid infrastructure," *OGF Document Series GFD.37*, Nov. 2004, also available as http://www.ggf.org/documents/documents/GFD.37.pdf.

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

[4] R. Fielding *et al.*, "Hypertext transfer protocol – HTTP/1.1," *Request for Comments (RFC) 2616*, June 1999.

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

[6] I. Mandrichenko, W. Allcock, and T. Perelmutov, "GridFTP v2 protocol description," *OGF Document Series GFD.47*, May 2005, also available as http://www.ggf.org/documents/GFD.47.pdf.

[7] 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)*, Oct. 2005, pp. 415–421.

[8] ——, "Automatic parameter configuration mechanism for data transfer protocol GridFTP," in *Proceedings of the 2006 International Symposium on Applications and the Internet (SAINT 2006)*, Jan. 2006, pp. 32–38.

[9] ——, "GridFTP-APT: Automatic parallelism tuning mechanism for data transfer protocol GridFTP," in *Proceedings of 6th IEEE International Symposium on Cluster Computing and the Grid (CCGrid2006)*, May 2006, pp. 454–461.

[10] L. Kleinrock, *Queueing systems, volume II: computer applications*. John Wiley & Sons, Inc., 1976.

[11] I. Mandrichenko *et al.*, "GridFTP protocol improvements," *OGF Document Series GFD.21*, July 2003, also available as http://www.ggf.org/documents/GFD.21.pdf.