Available online at www.sciencedirect.com
ScienceDirect
Computer Networks 51 (2007) 4561–4573
Computer Networks
www.elsevier.com/locate/comnet

GridJet: An underlying data-transporting protocol for accelerating Web communications

Frank Zhigang Wang *, Na Helian, Sining Wu, Yuhui Deng, Vineet Khare, Michael Parker

Centre for Grid Computing, Cambridge-Cranfield High Performance Computing Facility, MK43 0AL, United Kingdom

Available online 26 June 2007

Abstract

The World-Wide Web provides us with a distributed, hyperlinked document repository. Users attempting to access and share these hyperlinked documents via conventional HTTP and FTP often encounter long waits and frustration. To provide "local-like" access, a WAN/Grid-optimized protocol known as "GridJet" was incorporated into the Firefox Web browser that utilizes a wide range of technologies including the one of paralleling the remote file access. No change in the way of using software is required since the multi-streamed GridJet protocol remains fully compatible with existing IP infrastructures. Peer-to-peer clustered Web servers are also constructed to remove the scalability limitations and management problems associated with individual Web servers. Our recent progress includes a real-world test that Web applications over the GridJet protocol beats those over the classic ones by as much as five times where the transfer distance is over 10 000 km.

Keywords: Web communication infrastructure; World-Wide Web; Parallel data transfer; Grid computing; Bulk data transfer; Peer-to-peer

1. Introduction

The World-Wide Web (WWW) provides us with a distributed, hyperlinked document repository. The underlying infrastructure is communications protocols, responsible for implementing much of the structure of the document repository [1]. For example, in the current Web, when users choose to navigate from one Web page, using a hyperlink, to another, they set in motion in a request/response transaction between their Web browser and a Web server, acting in a client/server relationship, which implements that navigation.

From the perspective of the Web, there has been tremendous interest in extending the infrastructure of the Web, for example, through the study of Web communication and topology and their influence on Web browsing/search, virtual communities, collaborations and distributed information delivery. Curiously, there has been little focus on how advances in communications and networking can contribute to this research [1]. Web infrastructure topics could be discussed in relation to communications and networking, and similarly, advances in networking could be discussed in relation to their impact on the infrastructure of the Web [1].

* Corresponding author.
E-mail address: frankwang@ieee.org (F.Z. Wang).

1389-1286/$ - see front matter © 2007 Elsevier B.V. All rights reserved.
doi:10.1016/j.comnet.2007.06.016

Unfortunately, attempting to share or access hyperlinked documents via conventional HTTP and FTP across the WAN often proves to be a frustrating, time-consuming experience. For the users whose productivity and efficiency depend on being able to utilize these types of applications across the WANs, two problems must be resolved: WAN latency/instability and file-sharing performance in WAN environments.

To address the above problems, a middleware solution known as "GridJet" is developed in our laboratory, which melds distributed file system technology with high performance data transfer techniques to meet the needs of WAN/Grid-based virtual organizations. GridJet is available in binary or source formats. GridJet provides a high-speed remote document sharing and storage consolidation solution. GridJet alleviates WAN latency issues for hyperlinked documents as well as other applications based on WWW standards.

We fill the gap for two vital, missing capabilities: high-performance remote document access/bulk data transfer, and secure data management integrated with existing Grid authentication and authorization tools. GridJet is a new approach that extends established Internet architectures and protocols to meet the above immediate needs and is positioned to adapt to the future needs of WWW/Grid through the minor versioning provision of existing protocols.

We continue in Section 2 with a discussion in relation to communications and networking, their impact on the infrastructure of the Web, their related research and our uniqueness/innovation. Section 3 discusses our implementation of GridJet. Section 4 describes P2P clustered Web servers based on the GridJet engine. Section 5 reports the response time improvement compared to HTTP/FTP-based applications. Section 6 describes a real-world test that exploits the latency reduction of our GridJet prototype. Section 7 concludes with a discussion of ongoing research.

2. Related work and innovations

Firefox is the second most popular browser, with over 12% of the market, and runs on all modern platforms within the context of the Mozilla project [2]. Mozilla provides a suite of open-source applications including the Firefox browser, Thunderbird email client, and Sunbird calendar, all of which are built on a common core also called Mozilla. From the start, the Mozilla core framework was designed to work on all modern platforms and to be highly modular in its architecture.

Topaz [3] is an in-progress, open-source project. Topaz is a Firefox protocol extension for GridFTP and has been integrated in the Mozilla framework. The performance evaluation shows that the initial overhead due to the additional layer of abstraction introduced by Topaz becomes insignificant as the size of the transferred files increases, i.e., above 16 MB. However, Topaz does not support parallel stream data transfers and third-party transfers.

There are other download extensions for Firefox, some of which also support parallel downloads. As a peer-to-peer (P2P) communications protocol for file sharing, BitTorrent (BT) distributes large amounts of data widely [4]. Recipients each supply data to newer recipients, reducing the cost and burden on any given individual source, providing redundancy against system problems, and reducing dependence upon the original distributor. BitComet contains many advanced features for BT download and extends to HTTP/FTP to accelerate downloading [5]. As a download manager, FlashGot uses the MHT (multi-server hyper-threading transportation) technique and optimization arithmetic and it can split downloaded files into sections and supports multipoint transfers in parallel [6]. FlashGot supports HTTP, FTP, BT, MMS and other protocols. However, usage of BT and its variants accounts for significant traffic on the Internet.

Different from the above schemes, the GridJet protocol supports not only parallel stream data transfers but also single system image (SSI). It can run as a stand-alone tool or can be integrated in the Mozilla framework. GridJet's goal of acceptance amongst diverse organizations requires that it provide fast and secure means of cross-domain remote document access and bulk data transfer.

3. GridJet: an underlying WAN/Grid-optimized protocol for Web applications

3.1. Firefox interfaced with GridJet

In this work, we show how to integrate the GridJet protocol into the Firefox browser in order to enable the same behavior as standard FTP. The user simply supplies a gridjet URL (gridjet://server:port/path) in the address bar of the browser or in the hyperlinks of the objects on the Web pages, then can browse, upload, and download documents from

the Web server. GridJet currently supports Linux® and future work includes extension to Windows and MacOS X-based systems.

The Mozilla Firefox interfaced with a WAN/Grid-optimized Protocol, GridJet, is graphically depicted in Fig. 1. Two key technologies used in the Mozilla core framework are XPCOM and XUL. XPCOM [7] is similar to CORBA and provides most of the functionality in Mozilla. Components can be written in C/C++, Python, and Javascript and are grouped into libraries that handle everything from filesystem manipulation, to security, XSLT, and rendering. XPCOM components are all cross-platform and new components can be added with a minimum of effort. XUL [8] is used to create GUIs in Mozilla. XUL is HTML-like in its simplicity yet Java Swing-like in its power; it can be combined seamlessly with CSS, SVG, Java applets and can access virtually any XPCOM component via a thin layer of Javascript. The ease of XUL and the robust, cross-platform nature of XPCOM combine to make Mozilla an ideal framework for rapid application development.

The functionality of the GridJet client protocol implementation is wrapped into the Mozilla component framework. A protocol handler and channel are then created that support the GridJet scheme. After that, the new components are registered into the Mozilla runtime. An extensible user interface language (XUL) overlay is added that adds a file upload MenuItem to the Firefox menus to support uploading of files. The entire extension is packaged into a standard cross-platform installable (XPI) file. And finally, the extension is compiled.

There are several advantages in using Firefox to create a GridJet client. The browser provides users with an easy-to-use interface that most users are already familiar with. Firefox provides a number of features that are available to the protocol such as drag and drop support and dialog boxes for uploading files and providing passwords. Since Firefox handles the user interface, it makes it easier and faster to develop a GridJet client across multiple platforms.

3.2. Beyond HTTP/FTP/GridFTP

The default Web browser protocol that moves objects across WWW is HTTP (hypertext transport protocol). FTP/GridFTP are the principal methods by which people move files over the Internet. FTP (file transport protocol) is already directly built into

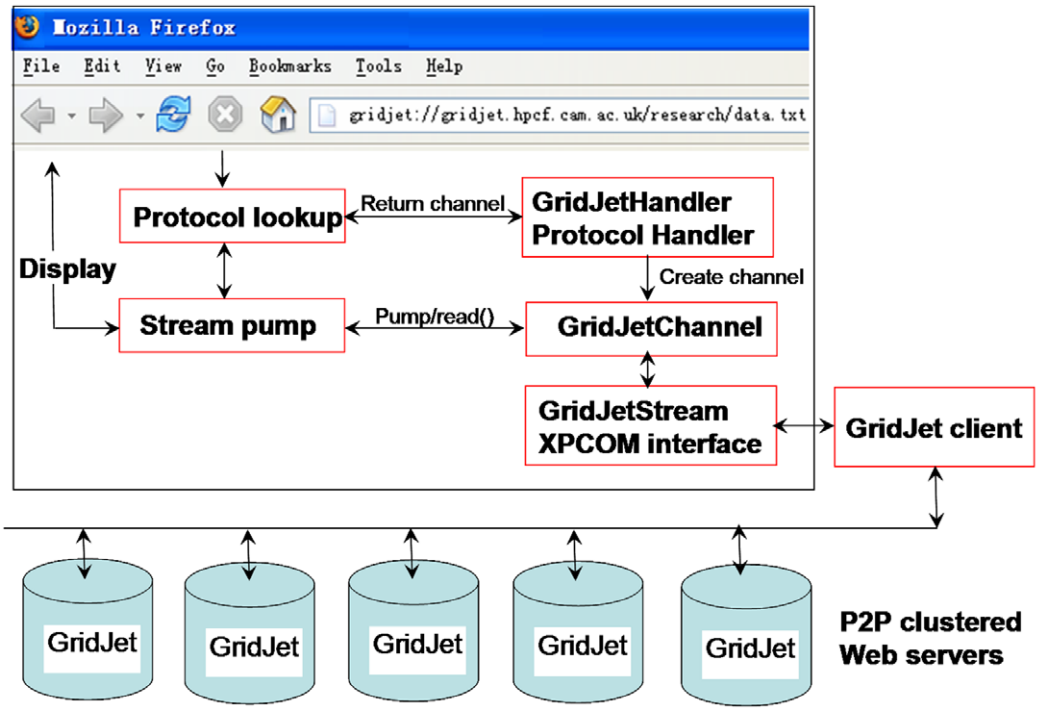


Fig. 1. Overview of the GridJet components of the Firefox extension.

most browsers, allowing users to simply type an FTP URL (ftp://server:port/path) in the address bar of the browser and browse, upload, and download their files over WWW. Topaz [3] is a Firefox protocol extension for GridFTP and has been integrated in the Mozilla framework, however Topaz does not support parallel stream data transfers [3]. In summary, because HTTP/FTP/Topaz are single-streamed, users attempting to work with Web Browser over HTTP/FTP/Topaz must often deal with unacceptable wait times in the WWW (60–1000 ms) [9].

In addition, a filesystem, in contrast to HTTP/FTP/GridFTP protocols, allows a user to use files on other network machines as if they were local. Furthermore, a filesystem also creates a single system image (SSI) unique to each user on each of a plurality of nodes in a virtual organization (VO).

3.3. GridJet security mechanism

Grid applications not only require efficient transfer of a large amount of data in wide area networks but also demand security and authentication services that enhance data integrity and enable data access control. The GridJet and its variations use public key infrastructure (PKI) associated with grid security infrastructure (GSI) to authenticate identities of WAN/Grid members and to secure resource allocation to these members [9]. The GridJet server requests and receives a host certificate from the CA [10]. The GridJet (service) also needs to be registered in the Firefoxrtual organization (VO) – lightweight directory access protocol (LDAP). Furthermore, the VO-LDAP generates a gridmap file to the GridJet server from the LDAP database for authorization and mapping. Beyond the role of certificate registration, the VO LDAP can perform other important roles for the VO, such as a service search engine and user authorization.

One of the interesting parts of a GridJet network is how often users communicate with the VO LDAP in relation to how often they communicate with GridJet servers. Essentially, the LDAP only needs to be accessed to initialize a session between a user and a GridJet server. After that, all communications can be handled between the user and the GridJet server. Data transfers that are broken off prematurely will need to be re-established, but it may not be necessary to authenticate again with the LDAP if the public key included in the certificate is still available for use. This design removes the VO LDAP as a potential storage I/O bottleneck.

3.4. GridJet implementation

The emergence of high speed wide area networks makes grid computing a reality. However, grid applications still have difficulties in achieving optimal TCP performance due to network tuning of TCP window size to improve the bandwidth and to reduce latency on a high speed wide area network.

Within the context of the UK Government project ‘‘grid-oriented storage (GridJet)’’ and the EC project ‘‘EuroAsiaGrid’’, a grid-based middleware solution known as ‘‘GridJet’’ has been developed in our laboratory, which melds distributed file system technology with high performance data transfer techniques to meet the needs of WAN/Grid-based virtual organizations. From its inception, the developed multi-streamed GridJet, via the fully-integrated parallel stream mechanism, is designed to deal with long-distance, bulk data, cross-domain and single-image file operations – a capability lacking in the current versions of single-streamed FTP. GridJet alleviates WAN latency issues for office/database documents as well as other applications based on FTP standards. The first-of-its-kind GridJet solution enables users to access and save office/database documents across the WAN/Grid at near-LAN speeds.

GridJet uses parallel streams to achieve very high transfer rates at a fraction of the memory cost. As shown in Fig. 2c, GridJet opens multiple socket streams between the sender and the receiver for applications. The application data are then partitioned into segments. The number of segments is equal to the number of streams. The segments are sent through streams in parallel by different threads and are then reassembled by the receiver. The final data are presented to applications as if they were transmitted through a single socket. Fig. 2 also illustrates why sending partitioned data through multiple socket streams can achieve near optimal bandwidth. In the figure the shaded areas represent socket buffer size and the large empty rectangles represent TCP pipe capacity (bandwidth*delay). Applications using a single socket stream will not fill the TCP pipe capacity. The capacity efficiency is about 1% when the buffer size is taken as its default value of 64 kB [11]. Applications using either a single socket stream with an optimal buffer size or mul-

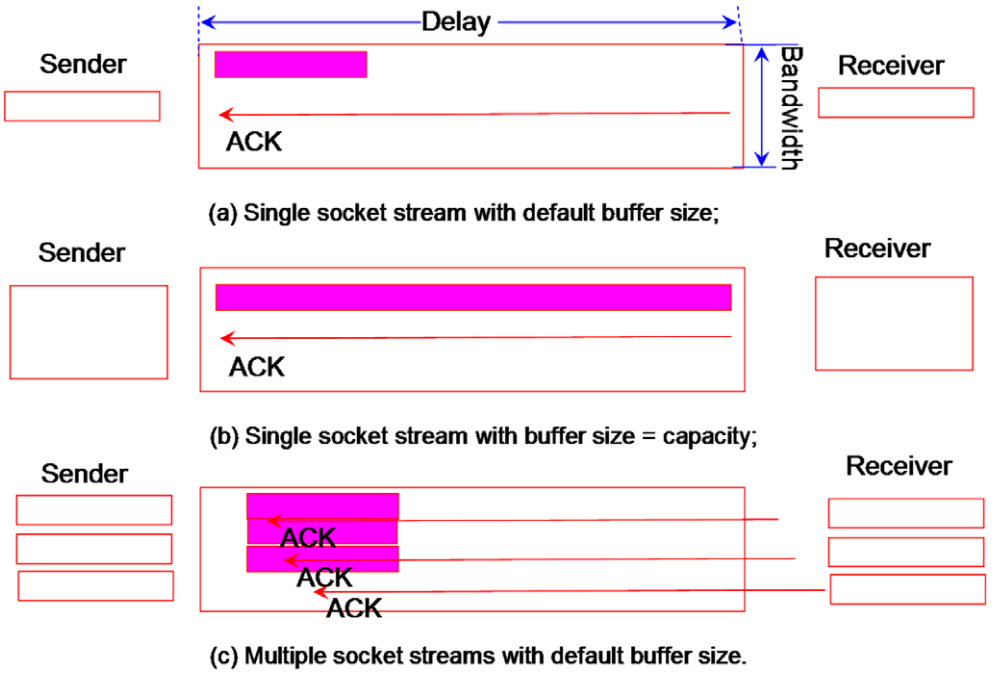


Fig. 2. Single or multiple TCP socket streams with different buffer sizes.

tiple socket streams can fill the TCP pipe, as shown in Fig. 2b and c. Chen [12] showed that using parallel streams is more effective than tuning TCP window size. In addition, acknowledging the entire file requires the entire file to be retransmitted even if a single packet is lost, as shown in Fig. 2b. Therefore applications using GridJet can obtain near optimal TCP performance without adjusting TCP window size.

Since Globus is becoming the overwhelming choice for grid development, non-Globus solutions will be difficult to melt seamlessly into popular grid environments. The GridJet uses Globus XIO to communicate with the parallel stream engine during a file read/write operation. As shown in Fig. 3, Globus XIO is broken into two main components: framework and drivers. The Globus XIO framework manages I/O operation requests that an application makes via the user API. The parallel stream driver is responsible for manipulating and transporting the user's data. The GSI driver performs the necessary messaging to authenticate a user and the integrity of the data. The TCP driver executes the socket level transport code contained within to establish a connection to the given contact string.

The VFS (Firefoxrtual filesystem switch) is implemented in the kernel. This implementation conforms naturally to the standard POSIX semantics and provides office/database applications with seamless access to GridJet. A kernel-memory module, the GridJet kernel client, acts as a VFS interface. The GridJet server nominates a user-space daemon, the GridJet user client, to penetrate the kernel/user boundary and to communicate with the kernel client [9].

The pair of the above kernel client and the user client is connected by three different message/data passing mechanisms for different considerations, respectively. The first mechanism is a pipe, one end of which is written by the kernel client and another end of which is read by the user client. The second mechanism is a message queue, in which each message generated by the user client stays until the kernel client reads it. The third is a new mechanism we call "data window". The "data window" mechanism (Fig. 4) breaks the space limit (32 MB) of the well-used IPC shared memory since we focus our attention on the bulk data transfer in the grid-oriented storage project. Like the IPC shared memory, the implemented "data window" mechanism also avoids copying data between the user space and the kernel space [9].

Hyperlinked documents in a Firefox browser are typically accessed in chunks of data (a page) at a time, slowing down the transfer of documents across the WAN. It is possible to move entire documents

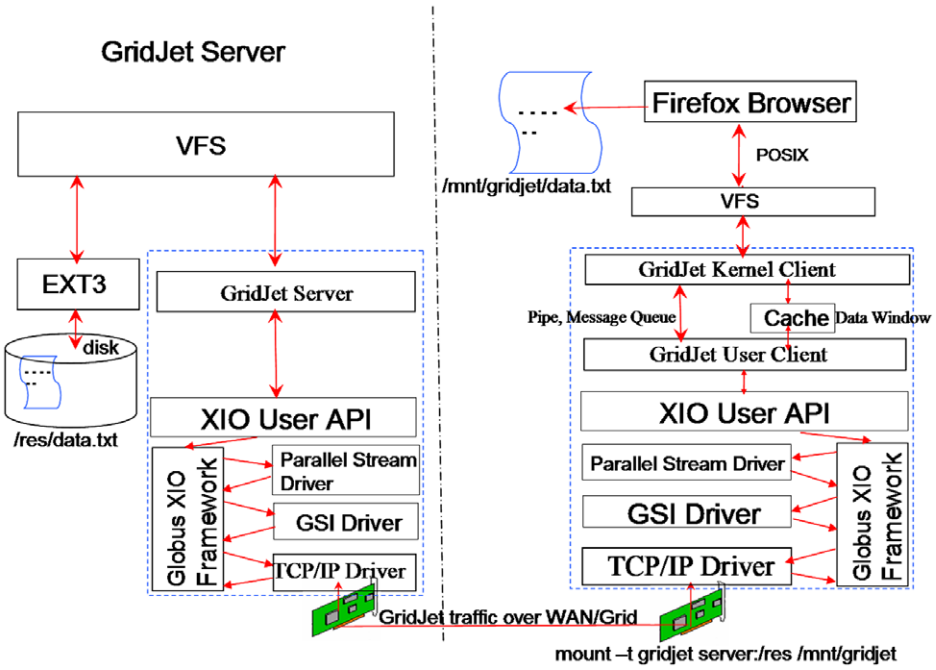


Fig. 3. The GridJet uses Globus XIO to communicate with the multi-stream engine during a database file read/write operation. A GridJet client mounts a file directory /res/data.txt on a remote file server to its local file tree /mnt/gridjet/ via the GridJet protocol, and thus the Firefox browser can access it as if it were local.

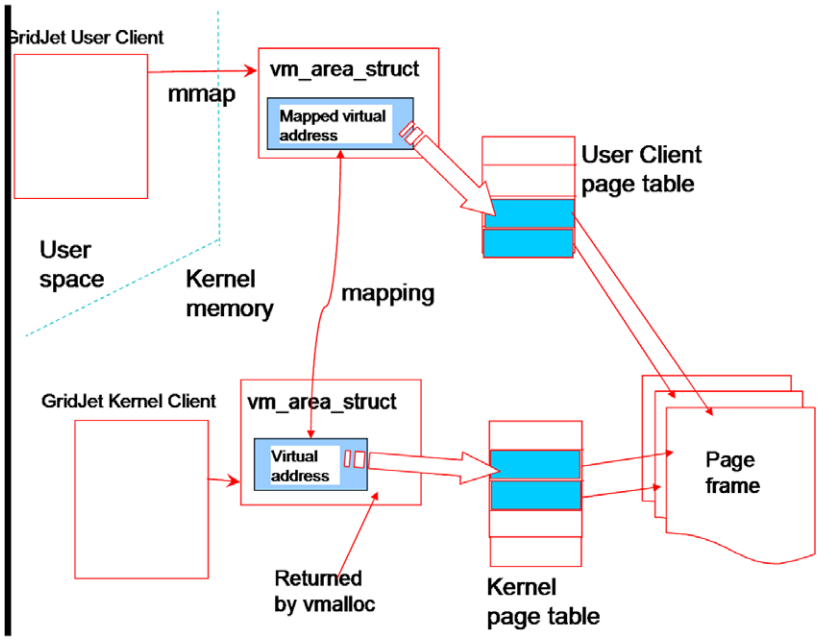


Fig. 4. The proposed "data window" mechanism breaks the space limit (128 MB) of the well-used IPC shared memory. A driver maps a virtual address to the User Client's user space (page table), which allows the User Client and the Kernel Client to access some common data structures.

across the WAN before the data are actually required by modifying the file size of the underlying file system calls (the default is 4 kB for Firefox). Therefore information is already localized when

users need it. Nevertheless, modifying the default file size may be in contradiction with the effort of making our GridJet accelerator universal. We had better use default configurations without applying any optimization. This is the usage condition for most scientists that normally have neither the needed expertise nor the time to configure the tools with high levels of optimization.

As shown in Fig. 3, a cache is introduced in the above data window of the GridJet to accumulate the file system calls (with default file size) originated by the browser to the underlying filesystem, GridJet. Pre-fetches means a number of pages (comprising the page being currently requested and the following ones) of the downloaded document are localized before those following are actually required. Note that Web-browsing would benefit a lot from those pre-fetches because sequential reads are dominant. Deferred writes are allowed to accumulate in the cache until a threshold value for the overall size of a number of file pages in cache is reached. File writes can often be flushed to GridJet in bulk file write operations that may be more efficient than the way the original application specified. The result is more data sent in fewer trips and less WAN waiting time. In addition, applications using multiple socket streams fill the TCP pipe to achieve optimal bandwidth. This combination reduces file access time while reducing the number of performance-impacting round trips over the WAN.

Note that the GridJet is a considerably complex piece of software. The source code of the developed GridJet is of 40 000 lines in length and we have spent four plus man years in developing and revamping it.

4. GridJet deployment: P2P clustered Web servers

The anticipated result of GridJet cluster is to remove the scalability limitations and management problems associated with individual Web servers. One of the design goals is to allow a single storage image to continue to expand, even if a single Web server reaches its capacity limitations. Built on the capabilities of multi-TCP stream and GSI-based data transport, GridJet provides a fast, lightweight, and reliable communication mechanism for building clusters of Web servers that communicate with each other via GridJet.

P2P clustered GridJet Web servers share a common, aggregate presentation of the data stored on all participating GridJet appliances. To borrow the overused parlance of the area, it offers users a single system image of the aggregate storage system. Each GridJet appliance manages its own internal storage space. The major benefit of this aggregation is that clustered GridJet storage can be accessed by users as a single mount point. This, in turn, enables user drive mappings to continue to grow beyond the capacity of a single appliance, which means user configurations do not need to be changed regardless of the number of different network connections and GridJet appliances used. Fig. 5 shows how three GridJet appliances present a unified storage system image from the three GridJet peers running on each node in a GridJet cluster.

The above P2P GridJet cluster is based on the small world model [13]. The small-world phenomenon is pervasive in networks arising from society, nature and technology. Any two individuals in the network are likely to be connected through a short sequence of intermediate acquaintances. In a famous experiment conducted by Stanley Milgram in 1960s, a principle of "six degrees of separation" existed in the American population.

One network construction that gives rise to small-world behavior is where each node in the network knows its physical neighbors, as well as a small number of randomly chosen distant nodes. The latter represent shortcuts in the network. It has been shown that this construction leads to graphs with small diameter, leading to a small routing distance between any two individuals. In the small world model, to try to minimize the amount of traffic and nodes involved in the transferal of packets around the network, the above directory server will be eliminated. Less reliance on central servers means that failures will affect very few servers. That means each server must contain some form of routing table (neighbor table in small world model).

When a user accesses data from the P2P GridJet cluster, either the GridJet appliance fulfils the request directly if the document resides on storage that it manages, or it requests, based on its neighbor table, the data from other GridJet appliance that manages it. The other appliance forwards the data directly to the requesting user. This process is shown in Fig. 5.

A P2P GridJet cluster has been constructed at our laboratory, consisting of three GridJet Web servers. We used a joint protocol [14] that constructs neighbor tables for new nodes and updates neighboring tables in existing nodes. Once a GridJet appliance is connected to the cluster, it can advertise

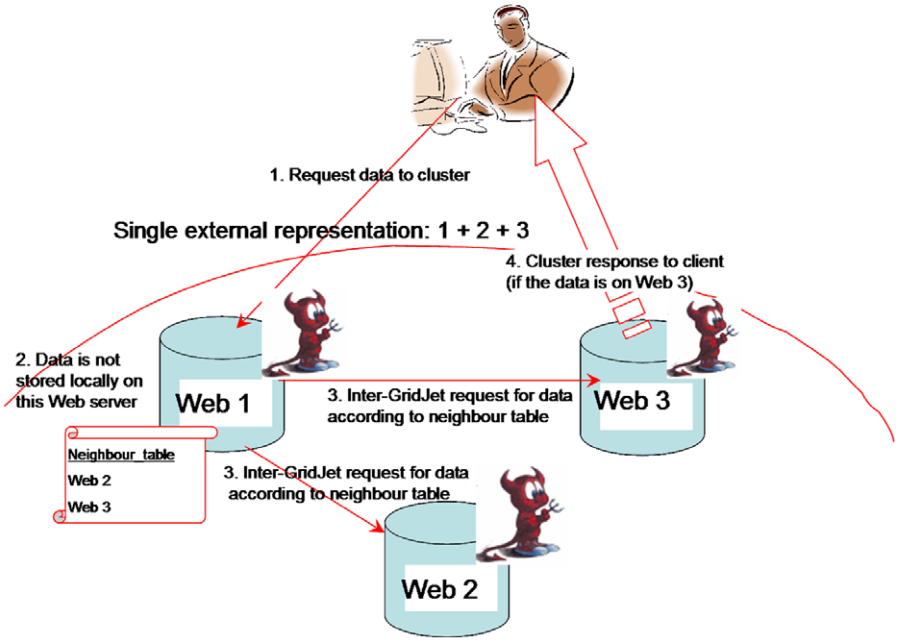


Fig. 5. Clustered GridJet-based Web servers share a common, aggregate presentation of the data stored on all participating GridJet appliances. An individual client machine interacts with one or more GridJet peers to retrieve information stored on any peer of the whole P2P cluster.

its availability without any intervention, featuring dynamic host configuration protocol (DHCP). The expected communication cost of advertising a new GridJet peer's availability and integrating a new GridJet peer into the cluster is shown to be small by both theoretical analysis and experimental measurements. On the other hand, the P2P routing module constitutes only a small fraction of the GridJet control software.

5. Tests over the emulated WAN

5.1. Test configuration and methodology

This section examines and investigates the relationship between distributed hyperlinked documents and the underlying GridJet communications infrastructure. Storage virtualization is usually done on a file level in the context of Web browser client I/O redirection. As exemplified in Fig. 3, a client mounts a file directory /res/data.txt on a remote file server to its local file tree/mnt/gridjet/ via the GridJet protocol, and thus the browser client can access it as if it were local. For example, a user simply supplies a gridjet URL (gridjet://server:port/path) in the address bar of the Firefox browser or in the hyperlinks of the objects on the Web pages to open GridJet-hyperlinked documents.

Since GridJet is built on top of Globus libraries and has been integrated into the Firefox browser, we introduce an additional layer as an abstraction of the communication infrastructure. To quantify any possible overhead, we compared the transfer times in seconds for opening documents between a browser client at Cambridge–Cranfield high performance computing facility and a GridJet server in Beijing using different transmission tools: HTTP, FTP and GridJet. We used a set of documents with different sizes (i.e., 256 KB, 1 MB, 4 MB, 16 MB, 64 MB) and we ran the different experiments three times for each protocol and each file size. The figures reported in this section are the average values of those measurements. For each tool, we measured the time from when the viewing command is issued by the user until the completion of the document transfer.

The WAN emulator, implemented using NIST-Net technology, was used to emulate an Ethernet network, producing user-settable delay and packet drop probability. Note that RTT (round-trip time) $>= 100$ ms represents pretty long distances since GridJet is designed to deal with World-wide Web operations. In a real-world test, the Cam-

bridge-Beijing datalink results in 539 ms of round-trip latency and 27 router hops where the geographic distance is over 10 000 km.

5.2. Accessing Web documents remotely via Gridjet

Extensive performance tests were conducted to measure the effectiveness of the Firefox/GridJet, in terms of the response-time improvements delivered to users. HTTP is chosen as the standard transfer protocol. Speedup is defined as the transfer time of Firefox/HTTP over the transfer time of Firefox deployed on another protocol. The GridJet security was also verified.

We measured the response times of a client requesting remote Web documents across the emulated 100 Mbps WAN link. Results in Table 1 show that 43 s were consumed when opening a 4 MB document via HTTP over a connection with 600 ms of RTT. When we added the GridJet server/client gateways to the link, the same 4 MB document required just 24 s to open across the link with #tcp = 64, representing a response time improvement of 1.8 ×. We ran the same test, this time viewing (transferring) a 64 MB document. Without the aid of the GridJet acceleration, the document is transferred in 324 s. When the GridJet server/client gateways were added to the test bed, the document access took 56 s across the link with #tcp = 64. The response time improvement jumps to 5.8 ×.

In particular, Fig. 6 shows the average speedup (y-axis) measured for opening documents with different sizes (x-axis). Namely Firefox is deployed over HTTP, FTP and GridJet (#tcp = 64), respectively. The figure shows that although GridJet is characterized by an initial transfer time that is larger than the other tools for small documents less than 2 MB, this gap tends to decrease as the size of the document increases and for documents larger than 2 MB the GridJet overtakes HTTP and FTP, reaching a speedup of 5.8 when opening a document of 64 MB. The reason for the additional overheads in

Table 1
Opening time in seconds

Document size	Firefox/HTTP	Firefox/FTP	Firefox/GridJet (#tcp)			
			Firefox/GridJet (1)	Firefox/GridJet (16)	Firefox/GridJet (64)	Firefox/GridJet (128)
256 kB	3	3	9	7	4	4
1 MB	11	12	15	13	12	13
4 MB	43	45	62	24	24	24
16 MB	165	175	186	38	31	31
64 MB	324	321	378	66	56	57

RTT = 600 ms.

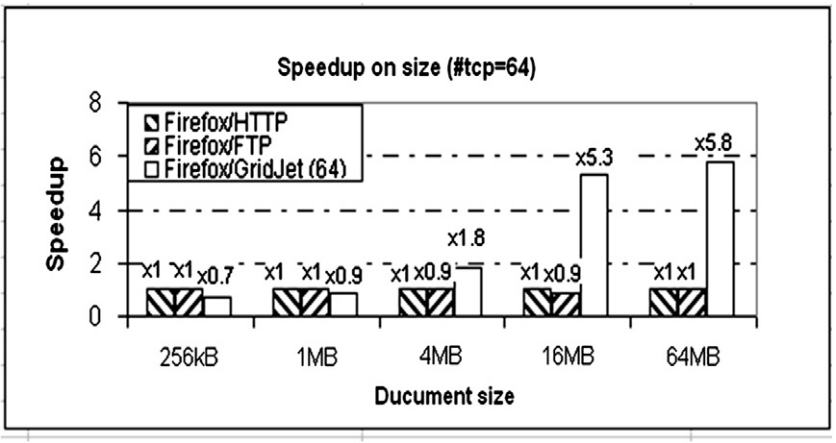


Fig. 6. The test measures the speedup as a function of the document size, namely Firefox over HTTP, FTP and GridJet (#tcp = 64), respectively. Although GridJet is characterized by an initial transfer time that is larger than the other tools for small documents less than 2 MB, this gap tends to decrease as the size of the document increases and for documents larger than 2 MB the GridJet overtakes HTTP and FTP, reaching a speedup factor of 5.8 when opening a document of 64 MB.

the GridJet functionalities is due to the additional layer of abstraction of GridJet on top of the Globus libraries (see Table 2).

Fig. 7 measures the response time improvement as a function of the number of tcp streams (#tcp). It is a standard open of a 16 MB document over the link with 600 ms RTT. Increases in the response time improvement with increased #tcp can be seen toward the right side of the graph. The maximum improvement goes up to 5.3× with 64 parallel streams. However, GridJet would eventually consume too much time in managing too many TCP streams, slowing down such an improvement with further increased #tcp.

The test described in Fig. 8 measures the speedup as a function of RTT (round-trip time) in ms. In the figure, GridJet outperforms the classic HTTP by factors ranging from 2 to 6, and this gap widens when RTT is increased further (RTTs of 60–1000 ms typify WWW environments). Note that the speed-of-light-in-fiber is 150 000 km/s so the equivalent physical distances are pretty long since GridJet is designed to deal with long-distance, cross-domain and single-image file operations, enabling users to connect remotely to the Web server.

6. Real-world gridjet-powered Web browsing on the euroasiagrid

The following real-world tests demonstrate the value of the GridJet-powered Web communications over the EuroAsiaGrid network resources. The EuroAsiaGrid project is funded by the European Commission (EC), consisting of over 10 European and Asian partner institutions. The time in seconds shows how GridJet protocol dramatically reduces Web document accessing times in real-world environments.

The purpose of this investigation was to examine the alpha release of the GridJet code, test and compare its performance to standard single-stream HTTP/FTP/Topaz transportations. Because the GridJet will be the underlying data transfer engine, it is very important to have a first-hand experience of its performance and capabilities.

Five EuroAsiaGrid sites, Cambridge, London, Portsmouth, Murcia and Beijing have been participating in the tests. At each site a Linux/Globus machine with installed GridJet client gateway was used as a client to access a dedicated Web server with installed GridJet server gateway at the Cam-

Table 2
The EuroAsiaGrid network configurations

Link	Router hops	Average RTT (ms)	LAN connectivity	Client memory (MB)
Cambridge–London	16 hops	17 ms	2 Mbps (London)	512 MB
Cambridge–Portsmouth	18 hops	19 ms	4 Mbps (Portsmouth)	512 MB
Cambridge–Murcia	23 hops	62 ms	10 Mbps (Murcia)	256 MB
Cambridge–Beijing	27 hops	539 ms	4 Mbps (Beijing)	128 MB

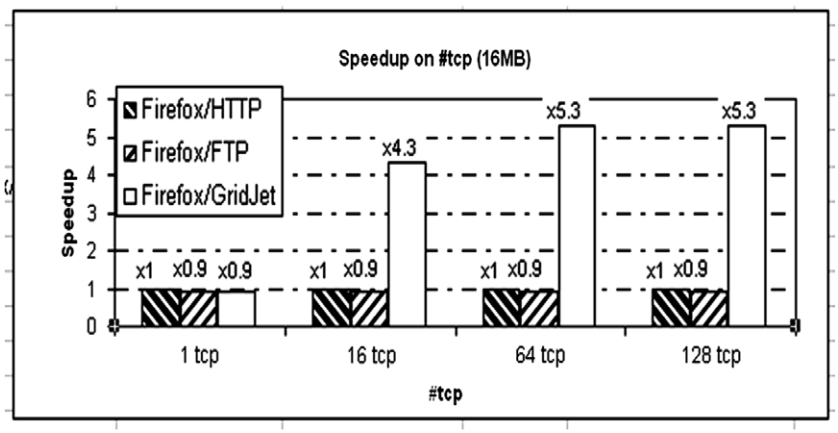


Fig. 7. The test measures the speedup as a function of the number of TCP streams, namely Firefox over HTTP, FTP and GridJet, respectively. A 16 MB document is used in all the measurements. The maximum improvement goes up to 5.3× with 64 parallel streams. However, GridJet would eventually consume too much time in managing too many TCP streams, slowing down such an improvement with further increased #tcp.

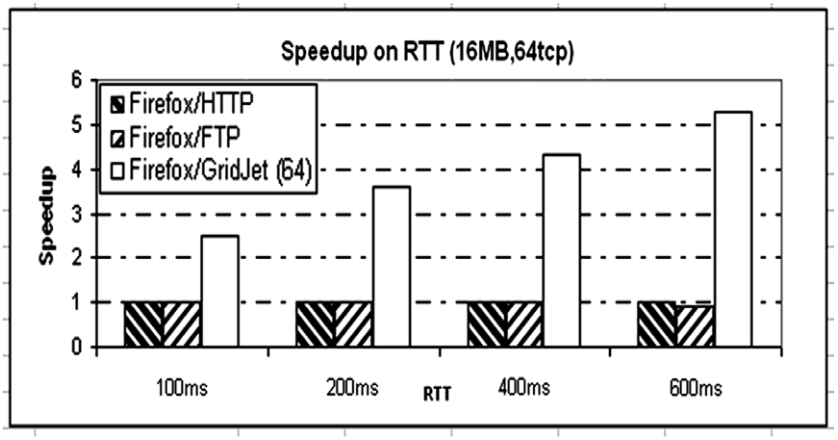


Fig. 8. The test measures the speedup as a function of the round trip time (RTT), namely Firefox over HTTP, FTP and GridJet (#tcp = 64), respectively. A 16 MB document is used in all the measurements. In the figure, GridJet outperforms the classic HTTP by factors ranging from 2 to 6, and this gap widens when RTT is increased further (RTTs of 60–1000 ms typify WWW environments).

bridge–Cranfield high performance computing facility (CCHPCF) in the investigation. The CCHPCF has a 155 Mbps permanent connection to the Internet using 5 Gbps SuperJANET backbone via the EastNet Cambridge node, while at the time of the investigation Murcia had a 10 Mbps connection, London had a 2 Mbps connection, Beijing and Portsmouth had a 4 Mbps connection.

During the investigation, it was found that the background load of the network link had a significant influence on the Web performance: for example, the read time of a document of 16 MB over the 32 tcp GridJet from Cambridge and Beijing could change from 134 s at 22:00 GMT to 1081 s at 14:00 GMT. Therefore, in order to eliminate the effect of changing network load, document operation times were compared only if the document operations were performed over the same network link and taken within the same short time interval; furthermore, all the un-reconstructable and outstanding values were discarded.

The main tests were carried out between Cambridge and Beijing. The Cambridge–Beijing datalink represents the longest network connection in these document downloading/uploading tests over GridJet (geographic distance 10 000 km, 27 router hops, average RTT = 539 ms). The CCHPCF has a 155 Mbps permanent connection to the Internet while Institute of Computing Technology, Chinese Academy of Sciences, in Beijing has a 4 Mbps LAN connection. The data travels from Cambridge to London, from London through a trans-Atlantic link to Huntington, the American West Virginia gateway, then through a trans-Pacific link via Australia, and finally arrives in Beijing.

Two sets of tests were performed: the first series of remote document operation measurements were taken between 19:00 and 22:00 GMT at a relatively low network load, while the second was performed over a congested network at peak time (10:00–15:00 GMT). The performance of the GridJet with respect to the number of parallel streams via remotely downloading/uploading the 4–128 MB document over HTTP/FTP was measured. Usage of the underlying GridJet engine radically increased the transfer rate of the document operations both over the congested and un-congested network, regardless of the load.

Point to point GridJet tests, carried out between the sites of Cambridge, London, Portsmouth, Murcia and Beijing, resulted in a remarkable performance increase of about 2–5 times, compared to those over single-stream HTTP/FTP. Parallel streams lead to increased bandwidth over both congested and unloaded networks. It is worth pointing out that in some of the cases not only the LAN or the actual hardware configuration but also numerous intermediate hops became the bottleneck in the WWW tests.

7. Conclusions

GridJet is an innovative combination of the accelerated communication and Web-based applications to expand the capabilities and usability of the Web and provides a simple access to grid technology.

The GridJet-based Web communications address the challenge of sharing files over a WAN/Grid with "LAN-like" performance, which is a highly beneficial aspect to the WWW/Grid user communities. In addition, the GridJet solution offers native integration with the Firefox browser and provides the ability to deliver data over the Wide Area Network using key security features inherent in the Globus platform. The first-of-its-kind GridJet solution optimizes Web data transfers across the WAN/grid to solve the needs of real-world users.

The core features include:

1. User communities demanding LAN-like performance for collaboration/communication, which is the failing of the current infrastructure and of the current protocols.
2. High-level architectural innovation. GridJet is an attempt to move grid computing from a research vision into production. The proposed P2P clustered GridJet Web servers share a common, aggregate presentation of the data, removing the scalability limitations and management problems associated with individual Web servers.
3. Technological innovation. It is demonstrated that using parallel streams is more effective than tuning TCP window size. On the other hand, using default configurations, which most users are familiar with, makes our GridJet accelerator universal.

In principle, the GridJet protocol can be combined with other data-intensive applications like content-based Web image search and Web-based online data mining. They are the highlighted issues on the GridJet roadmap in our laboratory. Future work will also include the integration of its third-party transfer functionality into GridJet. A third-party transfer will allow GridJet to connect to two GridJet Web servers simultaneously and initiate a direct file transfer between the servers. This will be accomplished by dragging a file from one Web browser window and dropping it in the other.

Acknowledgements

The project team is indebted to generous contributions from Dr. Jonathan Haynes, Dr. Les Oswald and Dr. Howard Jeffrey of Cambridge–Cranfield HPCF, Ms. Xiao Lijuan, Dr. Jin Xiong and Prof. Ninghui Sun of Institute of Computing Technology, Dr. Manuel Sanchez and Prof. Jose Garcia Carrasco of Murcia University, Dr. Mark Barker of Portsmouth University, and Dr. Mark Hayes of Cambridge e-Science Centre.

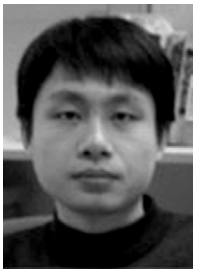
References

[1] Call for Papers, Special Issue of the Computer Networks journal on Innovations in Web Communications Infrastructure, 2006. <www.wmin.ac.uk/~courtes/iwi2006/journal. html>.
[2] Mozilla – An Open-source Web Browser. <http://www.mozilla.org.
[3] Richard Zamudio, Daniel Catarino, Michela Taufer, Brent Stearn, Karan Bhatia, Topaz: a Firefox protocol extension for GridFTP based on data flow diagrams, Technical Report UTEP-CS-06-18, April 2006.
[4] BitTorrent.org forum, <www.bittorrent.org> 2007.
[5] BitComet – A powerful C++ BitTorrent Client for file-sharing, <www.bitcomet.com> 2007.
[6] FlashGot – Firefox Download Manager Integration, flashgot.net, 2007.
[7] XPCOM – Cross Platform Component Object Model. <http://developer.mozilla.org/en/docs/XPCO>.
[8] XUL – XML User Interface Language. <http://www.xulplanet.com>.
[9] Frank Wang, Sining Wu, Na Helian, Andy Parker, Yike Guo, Yuhui Deng, Vineet Khare, Grid-oriented storage: a single-image, cross-domain, high-bandwidth architecture, IEEE Trans. Comput. 56 (4) (2007) 474–487.
[10] GlobusWORLD, <www.globusworld.com/program/program.php>, 2006.
[11] Andrew S. Tanenbaum, Computer Networks, fourth ed., ISBN-10: 0-13-066102-3, Prentice Hall, 2002.
[12] Jie Chen, Walt Akers, Ying Chen, William Watson III, Java Parallel Secure Stream for Grid Computing, <http://www.ihep.ac.cn/~chep01/abstract/10-008.htm>, 2005.
[13] J. Kleinberg, Navigation in a small-world, Nature 406 (2000).
[14] Huaiyu Liu, Simon S. Lam, Neighbor table construction and update in a dynamic peer-to-peer network, in: Proceedings IEEE ICDCS, Providence, RI, May 2003.

Frank Z. Wang is a Fellow of British Computer Society, Professor and Chair in e-Science and Grid Computing, Director of Centre for Grid Computing within the context of Cambridge–Cranfield High Performance Computing Facility (CCHPCF) [http://www.hpcf.cam.ac.uk/research.html]. His appointment is seen as crucial to the initiative of the CCHPCF, which is a collaborative research facility in the Universities of Cambridge and Cranfield. He has a publication record including one book titled "Encyclopedia of Grid Computing" to be pressed by an American publisher, 61 journal papers (IEEE Transactions on Computers, ACM Operating System Review, International Journal of Computer Networks, New Generation Computing, IEEE Transactions on Magnetics, Parallel Processing Letters, IEEE Distributed Systems Online, Journal of Applied Physics,

Applied Physics Letters, etc), 37 conference proceeding papers and two book chapters. He is on the Editorial Board of four international journals. He serves the High End Computing Panel for Science Foundation Ireland (SFI). He has been elected as the Chairman (UK & Republic of Ireland Chapter) of the IEEE Computer Society from January 2005. He is the holder of a UK Government EPSRC/DTI grant (£1 million) "Grid-Oriented Storage (GOS)", a UK Government EPSRC grant (£460 k) "Accelerating NFS/CIFS to produce a tenfold performance improvement for Office/Database applications over the WAN/Grid" and an EC grant (400 k euro) "EuroAsiaGrid", etc. Since January 2007, the Grid-oriented Storage (GOS) invention led to invitations for Frank to present talks at Princeton University, Cambridge University (Computer Lab), Rolls Royce, BBC, Xerox, Carnegie Mellon University, Birmingham University, CERN, York University, The Hong Kong University of Science and Technology, National Tsing Hua University, Manchester University, Oxford University, etc.

Na Helian received the Ph.D. degree in computer science in 1992. She has various working experiences in Japan, Singapore and UK. She is now a Senior Lecturer and the Director of MSc Data Mining Program at Department of Computing, Communication & Mathematics, London Metropolitan University, UK. She is the co-investigator of the UK Government EPSRC/DTI grant "Grid-oriented Storage (GOS)".



Sining Wu received his Ph.D. at the Institute of Computing Technology (ICT), Chinese Academy of Sciences in 2004. He was involved in the development of 10-TFLops Dawning, listed in the Top 500 Supercomputer List. He is now a Research Officer at the Centre for Grid Computing. His current research interests include distributed operating systems, storage systems and file systems.

Yuhui Deng obtained his Ph.D. in storage system in 2004. He had spent nearly four years in the National Key Laboratory of Data Storage System in China, being involved in three Chinese Natural Science Foundation projects and covering computer architecture, parallel I/O, network storage, virtual storage, etc. He is now a Research Officer at Centre for Grid Computing.

Vineet R. Khare received his BTech degree from India Institute of Technology (IIT) Kanpur, India, in 2001. His Ph.D. degree in Computer Science is to be awarded by The University of Birmingham in December 2006. Currently he is working as a research officer at the Centre for Grid Computing. His research interests include swarm intelligence, co-evolution, multiple neural network systems and multi-objective evolutionary algorithms.

Michael Andrew Parker holds a Ph.D. from the University of London, and an MA from Oxford. He is now the Director of the Cambridge eScience Centre. He is responsible for grid computing initiatives across the University, covering a wide variety of projects in physics, life sciences, earth sciences, medicine, chemistry and engineering, all requiring the sharing of large computational and data resources. He sits on the Management Committees of the Cambridge–Cranfield High Performance Computing Facility.