High Performance Data Transfer in Grid Environment Using GridFTP over InfiniBand

Hari Subramoni, Ping Lai, Raj Kettimuthu, and Dhabaleswar K. Panda

Department of Computer Science and Engineering, The Ohio State University

{subramon, laipi, panda}@cse.ohio-state.edu

Mathematics and Computer Science Division, Argonne National Labs

{kettimut}@mcs.anl.gov

Technical Report
OSU-CISRC-11/09-TR53

Abstract

GridFTP has established itself as a popular tool for data transfer in the grid environment. GridFTP is designed on top of the Globus XIO framework which makes the task of integrating new transport protocols and storage systems into it a very easy one. The performance of GridFTP is highly dependant on the disk I/O techniques, as well as the underlying network communication protocols. Though GridFTP has many optimizations for disk I/O operations, the relatively low communication bandwidth offered by the existing network protocols presents a bottleneck to next generation exascale applications which would want to transfer exabytes of data across long distances. On the other hand, modern interconnects such as InfiniBand have introduced many advanced communication features, like zero-copy protocol and RDMA operations which can greatly improve the communication efficiency. Moreover, the recently introduced InfiniBand WAN routers such as Obsidian Longbows now give us the ability to extend the reach of InfiniBand to WAN distances.

In this paper, we take on the challenge of combining the ease of use of the Globus XIO framework and the high performance achieved through InfiniBand communication, thereby natively supporting GridFTP over InfiniBand based networks. The Advanced Data Transfer Service (ADTS) designed in our previous work provides the low level InfiniBand support to the Globus XIO layer so that it can be used by GridFTP as well as by end-user Grid applications built on top of Globus XIO. In order to achieve efficient disk based data transfers, we introduce the concepts of I/O staging in the Globus XIO ADTS driver. We evaluate our design in both LAN and WAN environments using microbenchmarks as well as communication traces from several real world applications. We also provide insights into the communication performance with some in-depth analysis. The results of our experimental evaluation shows a performance improvement of up to 100% for ADTS based data transfers as compared with TCP or UDP based ones in LAN as well as high delay WAN scenarios.

Keywords: *GridFTP, RDMA, Zero-Copy, Globus-XIO, Cluster-of-Clusters, InfiniBand, Obsidian Longbow XR, InfiniBand WAN, iWARP*

1 Introduction

Ever increasing demands in high end computing and intensive data exchange together with the cost effectiveness of high performance commodity systems have led to massive deployments of compute and storage systems on a global scale. In such grid based scenarios, as shown in Figure 1, bulk data transfer within and across these physically separated clusters or data-centers has become an inescapable requirement for the uses of scientific data-set distribution, content replication, remote data backup, etc. Generally, File Transfer Protocol (FTP) [1] is used for handling bulk data transfers. Through the years since the earliest FTP implementation based on TCP/IP, there have been a lot of efforts on its improvement and extensions [2, 3, 4, 5, 6, 7]. Among these, GridFTP, designed specifically for high-bandwidth wide area networks, has emerged as the most popular FTP implementation in the Grid environment.

On the other hand, System Area Networks (SAN) such as InfiniBand (IB) [8] and 10 Gigabit Ethernet/iWARP [9] are rapidly gaining momentum for designing the high-end clusters and data-centers. These high performance interconnects have revolutionized the communication capabilities of modern systems. In addition to providing high bandwidth and low latency, they also provide advanced features like zero-copy communication and Remote Direct Memory Access (RDMA) that enable the design of novel communication protocols. Furthermore, industry vendors [10, 11] have recently introduced IB WAN routers to extend these capabilities beyond a single cluster or data-center, e.g., across multiple campuses or even across WAN range. Hence, communication libraries are now capable of zero-copy communications over WAN, which also provides a new scope for designing newer FTP mechanisms. The Advanced Data Transfer Services (ADTS) [12] is one such high performance FTP library designed to run natively over IB based networks and thereby leverage the various performance benefits offered by it.

Though IB based communication offers excellent performance benefits, it is not easy to port an existing application to use IB due to the difference in communication semantics. The Globus XIO framework [13], used to design GridFTP, offers an easy to use interface to the end users as well as applications, to perform file transfers over the network without having to worry about the communication semantics of the underlying devices (network or disk). In this paper, we take on the challenge of combining the ease of use of the Globus XIO framework and the high performance achieved through InfiniBand communication by natively supporting GridFTP over InfiniBand based networks. In particular, ADTS library is ported to the Globus XIO framework so that it can be utilized by GridFTP to transfer files. We also enhance the disk I/O performance of the existing ADTS library by adding support to decouple the network processing from the disk I/O operations.

We evaluate our design at both the microbenchmark and application levels. The results of the microbenchmark level

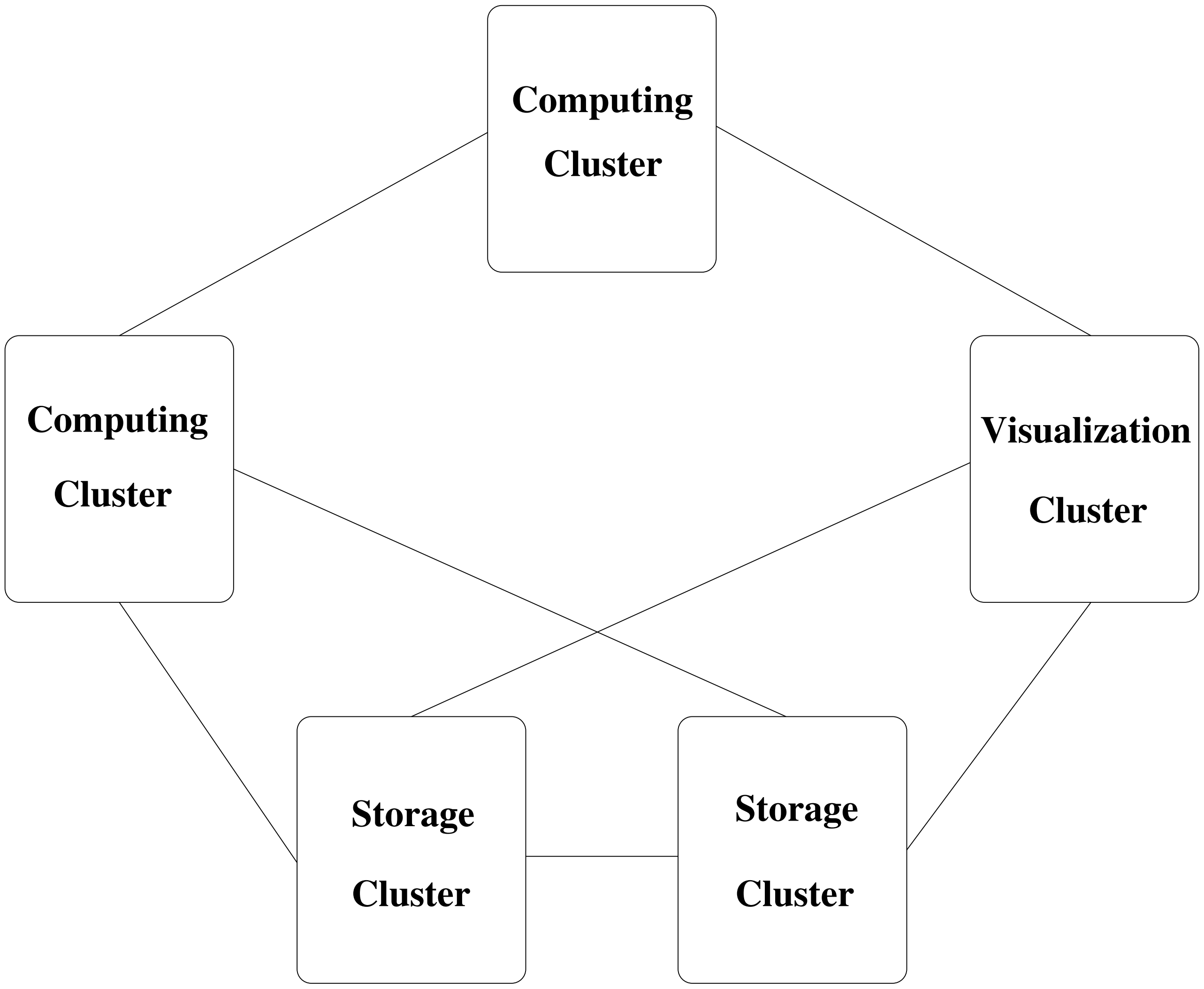

Figure 1. Typical Scenario in Modern Grid Environment

experiments show that the implementation of the Globus XIO framework over the ADTS library offers up 100% improvement in performance over the TCP or UDP based ones (IPoIB) in both LAN as well as WAN scenarios. We also see that the separation of network processing from the disk I/O processing has an impact on the performance, especially when we go to WAN scenarios where delays can be of the order of tens or even hundreds of milliseconds. Similar trends are seen with the application level evaluation as well. We simulate the communication patterns of various high end computing applications such as the Community Climate System Model [14] and Ultra Scale Visualization [15] which transfer large quantities of data over high delay networks.

The remainder of this paper is organized as follows: Section 2 describes the motivation of this work. Section 3 gives a brief overview of GridFTP and InfiniBand. In Section 4.2 we present our design of Globus ADTS XIO driver. We evaluate and analyze the performance in various scenarios in Section 5, describe the related work in Section 6, and summarize the conclusions and possible future work in Section 7.

2 Motivation

Distributed Petascale Science computations and experiments require unprecedented wide-area, end-to-end capabilities in the form of high throughput data transport. Such requirements arise in a number of science areas including climate, high energy physics, astrophysics, combustion, nanoscience, genomics, and others. The modeling of complex systems, such as climate [16] or supernova [17], at higher fidelity generates proportionately larger volumes of data that must be visualized, examined, and studied by widely dispersed scientific teams for gaining insights and making new discovery. Unfortunately, the amount of data now being created by major computational codes exceeds the capability of current network-based data distribution, despite the available 10Gbps capacities of network backbone connections. Several examples exist, including supernova simulation and combustion modeling where data distribution has been delayed or thwarted. However, one of the most dramatic examples is the collection of computed data from climate models from remote supercomputer centers for the current 5-year international normalization. It is being done using a large wheeled RAID (Redundant Array of Inexpensive Disks) array that is physically shipped to participating institutions such as Oak Ridge National Laboratory (ORNL), where it is filled and returned to Livermore where the effort is coordinated.

The current network transport limitations are no longer an artifact of the limited capacity of the network backbone as originally surmised. Indeed, the 10Gbps backbones of either ESnet [18] or Science Data Network (SDN) can currently offer such capacity to connect pairs of sites for extended periods, which corresponds to 500 Megabytes per second or 50 Terabytes per day. The fact that science users neither see nor use this bandwidth is symptomatic of much deeper challenges, which will only get worse with the next generation of Petascale and Exascale projects. Historically, wide-area data transport is handled mostly by Transmission Control Protocol (TCP), which has been the basis for File Transfer Protocol (FTP), GridFTP [2, 19], and HyperText Transfer Protocol (HTTP). The unprecedented demands that Petascale applications place on data transport will push TCP well beyond its useful envelope. The fundamental problem with TCP ultimately reduces to its treatment of bandwidth as a shared resource, and there have been a number of efforts to develop high-performance versions of TCP [20, 21, 22, 23] and reliable layers on top of UDP [24, 25] but only with limited success. Existing research shows that a reliable UDP (UDT) is able to achieve a much higher throughput than a TCP connection. However, the user level processing incurred by UDT requires GridFTP to consume a lot more resources then even a regular TCP connection would incur[26].

On the other hand, InfiniBand [8], originally designed for short range data transfers between high-performance storage and computing systems now have WAN capabilities thanks to devices such as the Longbow ER/XR routers [10] from Obsidian. Such capabilities and the introduction of newer communication technologies such as RDMA over Ethernet (RDMAoE) [27] is also accelerating the acceptance of high performance IB as a suitable candidate for WAN communication. Given all these, there now exists multiple ways for the end user to perform a long distance data transfer, which we summarize in Figure 2.

Of the various communication options shown in Figure 2, we have already seen that TCP, UDP and protocols adapted from them (e.g., IPoIB or SDP) cannot achieve good performance in IB based systems. Our earlier work on ADTS [12] shows, taking GridFTP as an example, the various application level limitations imposed by these adaptations. Given this scenario, we are left with two choices to design our high performance FTP library - IB Verbs and RDMA over Ethernet. These options are highlighted in gray color in Figure 2. We focus on the first (IB Verbs), of the two design choices in this paper. Due to technical issues, we are not able to perform experiments on RDMA over Ethernet enabled HCA's at present. We will include these preliminary numbers in the camera-ready version of the paper.

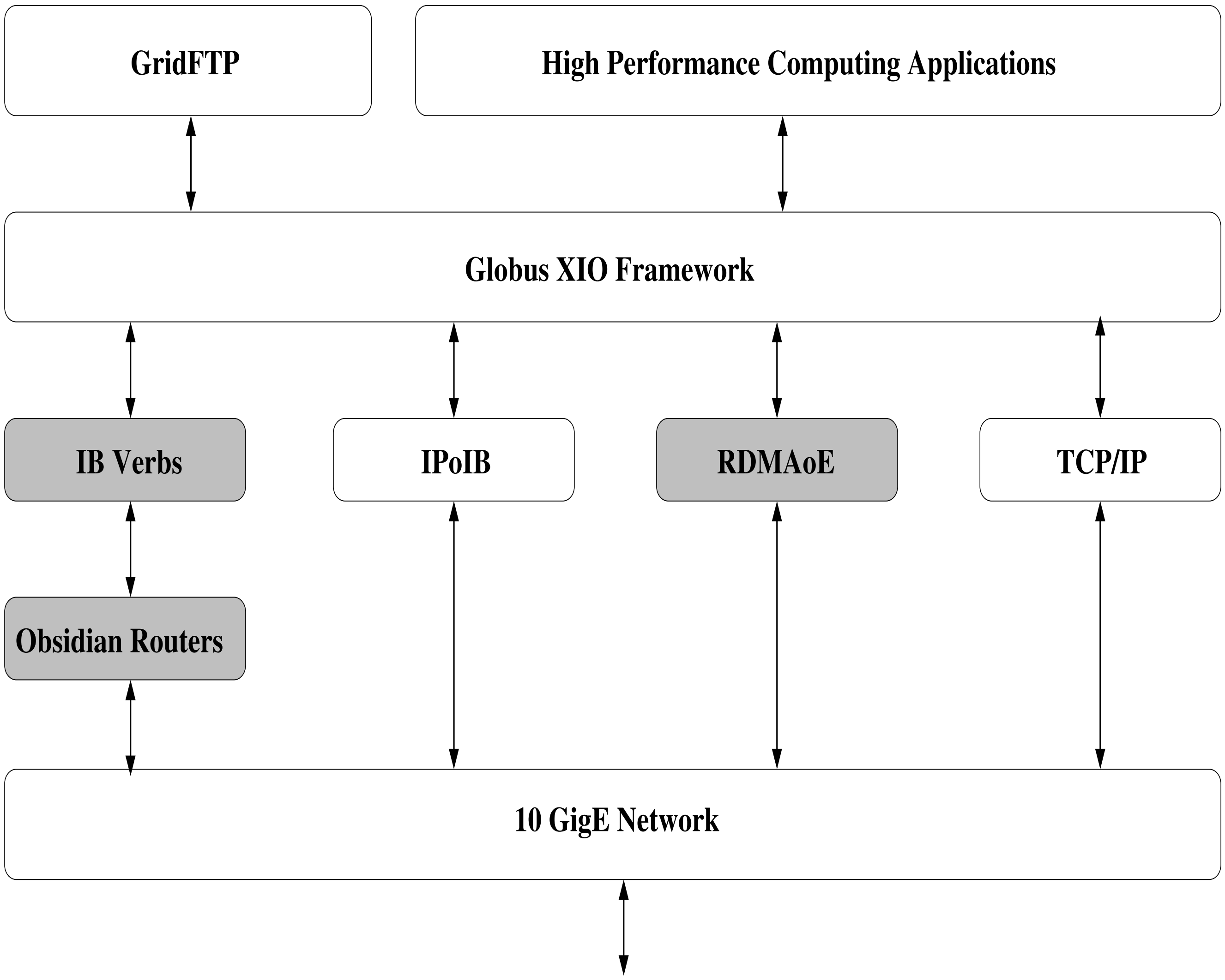

Figure 2. Communication Options Available in Modern Grid Environment

3 Background

In this section we give some brief background material on the topics covered in the paper.

3.1 GridFTP

GridFTP [2] is a high-performance, secure, reliable extension of the standard FTP data transfer protocol, optimized for high-bandwidth, wide area networks. The Globus implementation of GridFTP [19] provides a software suite optimized for a broad range of data access applications, including bulk file transfer and data extraction from complex storage systems. Thousands of installations around the world rely on GridFTP for over 6 million transfers per day. GridFTP has been used in record-setting DOE/HEP Large Hadron Collider Computing Grid data service challenges [28] and is a fundamental building block for a wide range of distributed data management and computing systems. Many other projects and facilities use GridFTP for data transfers, including;

- The Earth System Grid project uses GridFTP to move datasets for analysis,
- CMS and ATLAS employ GridFTP for data transport,
- Advanced Photon Source facility at Argonne uses GridFTP to transfer Terabytes of data to various collaborator locations around the world,
- the FLASH project employs GridFTP to move massive data sets from LLNL to various sites for visualization, and
- DOE's leadership computing facilities use GridFTP to move data in and out of their HPSS storage system.

Some key advantages of GridFTP include:

- Performance: GridFTP provides order of magnitude performance improvements over standard FTP by using parallel streams to minimize bottlenecks inherent in TCP and non-TCP protocols such as UDT. GridFTP performs coordinated data transfer using multiple computer nodes at the source and destination, resulting in further performance increases. GridFTP allows the client to simultaneously maintain multiple outstanding, unacknowledged transfer commands, which greatly improves performance when transferring large numbers of small files.
- Security: GridFTP supports the PKI/X.509 based Grid Security Infrastructure (GSI). A dynamically loadable authorization module allows for site-specific authorization of every operation. Simple options are available to encrypt data or perform integrity checking. GridFTP also supports SSH security, allowing users to initiate transfers using their SSH credentials without the need to obtain X.509 certificates.
- Robustness: Restart markers allow interrupted transfers to restart with minimal delay overhead.
- Extensibility: The Data Storage Interface (DSI) completely shields the user from the complexities of the underlying storage systems. DSIs are currently available for POSIX-compliant storage systems, HPSS (tape archive) and the Storage Resource Broker. Moreover, GridFTP utilizes a read-write-open-close abstraction called the Globus eXtensible IO (XIO) system that makes it transport protocol agnostic.

3.2 InfiniBand and InfiniBand WAN

InfiniBand Architecture (IBA) [8] defines a switched network fabric for interconnecting processing nodes and I/O nodes. It uses a queue-based model. A Queue Pair (QP) consists of a send queue and a receive queue. The send queue contains instructions for transmitting data and the receive queue contains the instructions describing the location of the receive buffer. At the low level, InfiniBand supports different transport services including Reliable Connection (**RC**) and Unreliable Datagram (**UD**).

IBA supports two types of communication semantics: Channel Semantics (Send-Receive communication model) for RC and UD, and Memory Semantics (RDMA communication model) for RC. Remote Direct Memory Access (RDMA) [29] operations allow processes to access the memory of a remote process without the intervention of the remote node's CPU. Both the approaches can perform zero-copy data transfers. i.e. the data can directly be transferred from the application source buffers to the destination buffers without additional host level memory copies. IBA also supports Shared Receive Queue (**SRQ**) mechanism that enables high server scalability while providing efficient flow control. With SRQ, receive buffers can be posted on a common receive queue for multiple connections. Further, it also provides a feature which notifies the application when the number of such receive buffers fall below a threshold. The receiver can post additional buffers upon getting the notification making sure that sufficient buffers are always available for incoming data. Thus flow control designs are significantly simplified.

TCP/IP network protocol stack can be adapted for use on InfiniBand through IPoIB driver [30]. When it is applied, an InfiniBand device is assigned an IP address and can be accessed just like any regular TCP/IP device.

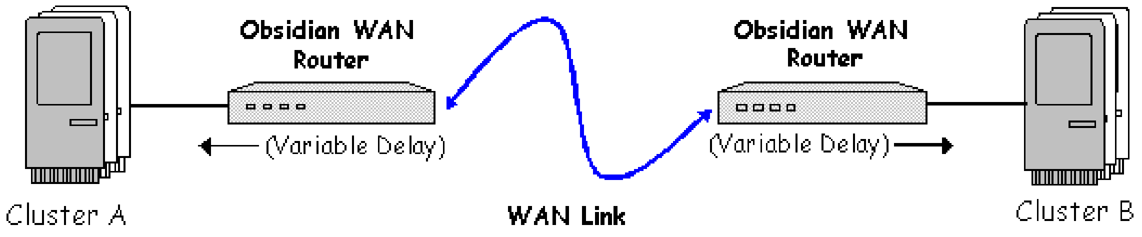

Figure 3. Cluster-of-Clusters Connected with Obsidian Longbow XRs

InfiniBand Range Extension with Obsidian Longbows: Obsidian Longbows [10] primarily provide range extension for InfiniBand fabrics over modern 10 Gigabit/s Wide Area Networks (WAN), supporting IB traffic at SDR rates (8 Gbps). The Longbows work in pairs, establishing point-to-point links between two clusters with one Longbow at each end of the link as shown in Figure 3. The Longbows unify both the networks into one InfiniBand subnet which is transparent to the InfiniBand applications and libraries, except for the increased latency added by the wire delays.

Typically, a delay of 5 *us* is expected per each km of wire length in WAN. The Obsidian Longbow routers provides a web interface for each to specify the delay. We leverage this feature to emulate cluster-of-clusters with varying degrees of separation in our experiment.

4 Design and Methodology

In this section we describe our approach to transferring large volume data over the grid.

4.1 Design of the Globus ADTS XIO Driver

The ADTS library provides us with a standard set of Open, Close, Read, Write (OCRW) interfaces similar to what expected by the Globus XIO framework from an underlying transport or transform driver. Given this, the job of creating a new transport driver for GridFTP using the ADTS library was one that involved little difficulty. The research challenges are encountered when we try to perform disk based file transfers over LAN as well as high delay WAN scenarios. The ADTS library was initially designed for memory based file transfers, and the performance bottlenecks introduced by devices with a slower data rate such as a hard disk or RAID's [31] were not given much importance. However, considering that most of the current as well as next generation grid based scientific applications would require movement of petabytes or exabytes of data, support for disk based transfer becomes critical to achieve high performance file transfers for real world applications.

Due to the relatively low bandwidth offered by the existing networking stacks (TCP or UDT), the disk I/O is not such a big performance bottleneck. However, with the introduction of the Globus ADTS XIO driver, the LAN as well as WAN performance will take a big leap as we saw in our previous work on the ADTS library [12]. In this scenario, it becomes imperative for us to effectively decouple the network from the disk I/O in order to achieve faster file transfers. For this purpose, we re-design the ADTS library to introduce multiple threads (read, write, and network thread) as well as a set of buffers to stage the data and thus decouple the disk from the network. The new architecture of the ADTS library is shown in Figure 4. Such decoupling of network from the slower speed mass storage devices will especially be useful in high delay WAN scenarios where keeping the network pipe full for good performance is a challenge.

As shown in Figure 4, the read thread pre-fetches a set of locations from the disk and keeps it ready for the network thread to send over the physical link. To prevent too many context switches between the threads and ensure that the network thread gets a large amount of the processing time, we define low and high water marks for the read and write threads. We begin reading only when the number of available buffers goes below the low water mark. The high water mark is set to the maximum size of the circular buffer. We have similar low and high water marks for the write thread as well. The values need to be carefully tuned to achieve best performance for a given network delay.

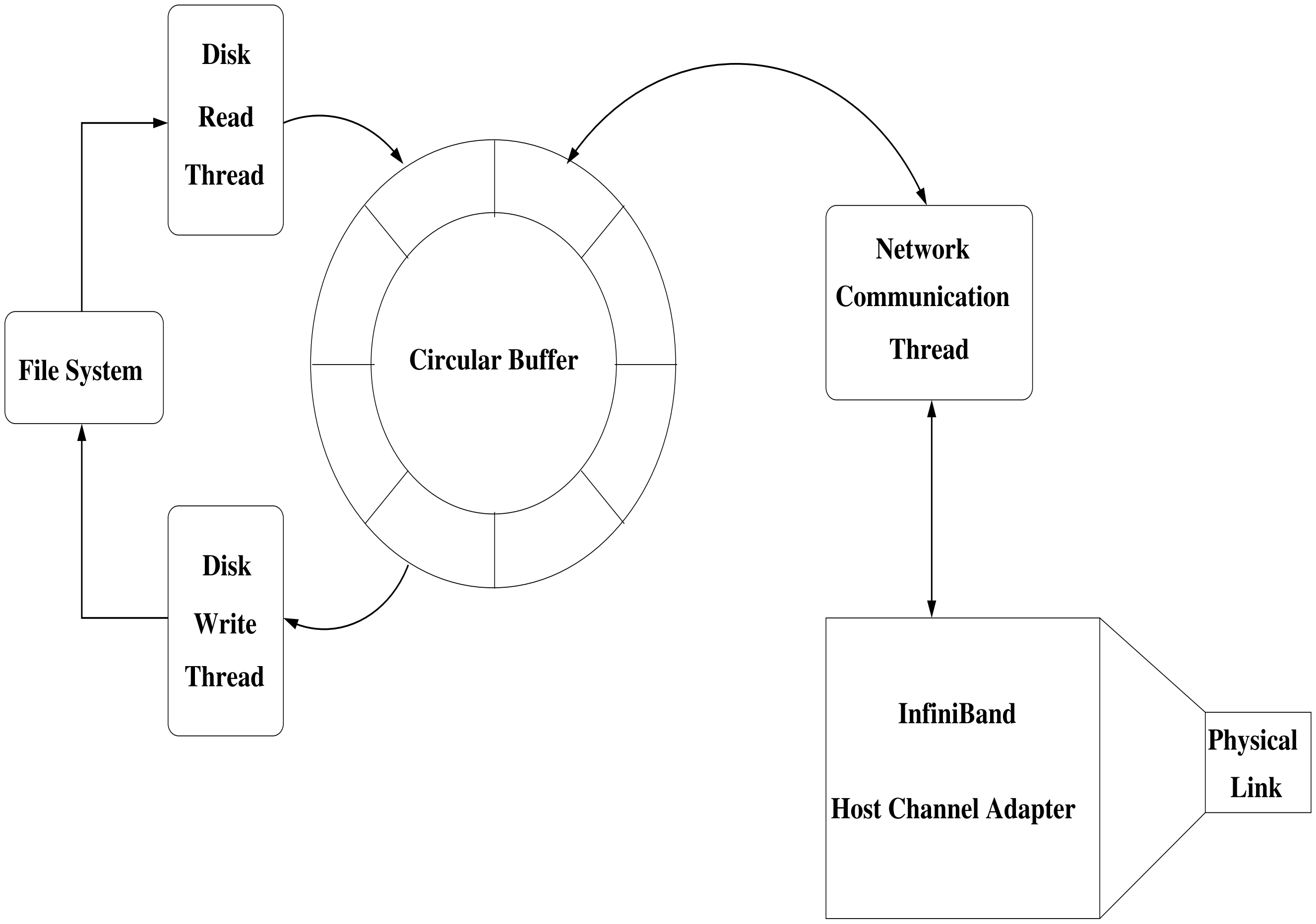

Figure 4. I/O Staging Design for ADTS XIO Driver

There are multiple buffers with configurable sizes within the ADTS XIO library.

- Network Buffers: This is the size of a buffer that is injected / retrieved from the network in one operation by the ADTS driver.
- File Buffers: This is the size of the circular buffer used by the ADTS library to buffer the received packets before writing it to the disk.

From our earlier work on the ADTS library [12], we found out that we are able to saturate the link bandwidth in LAN as well as high delay WAN scenarios at a packet size of 1 MB. So, for all our experiments we set the network buffer size to be 1 MB. The file buffer is a configurable parameter and will be critical in determining the level of decoupling obtained between the network and the disk I/O. There is always a trade off between the amount of memory consumed and the file transfer performance obtained, based on this parameter. A smaller buffer size would mean less memory consumed as well as more frequent writes or reads from the disk. This would result in spending more time performing disk I/O operations than network file transfers. The impact of such frequent disk I/O operations may not be readily visible in low latency networks where it is easier to keep the pipeline full. But in WAN scenarios with high delays, we will see the true impact of buffer size on the performance of applications.

4.2 Methodology

”Grid computing is the combination of computer resources from multiple administrative domains applied to a common task, usually to a scientific, technical or business problem that requires a great number of computer processing cycles or the need to process large amounts of data” [32, 33]. According to this statement, we can envision a scenario where we have many applications running on computational systems in geographically distributed locations, attempting to work on pieces of huge dataset. Such a mechanism would automatically involve sending and retrieving data and results from other applications or a central location. For such an architecture it is more natural for the application to fetch the data in-band, i.e while the application is in operation, rather than out of band, i.e work on a piece of data which was fetched previously and consequently could be stale at this point in time. Hence it would make more sense for such applications to use the Globus XIO framework directly (integrating it into applications) to retrieve data in-band instead of relying on GridFTP to pre-fetch the data for it in an out of band fashion. This will not make any difference in terms of performance because GridFTP ultimately uses the Globus XIO framework itself for transferring data.

Hence, apart from the standard set of comparisons using GrdiFTP, we also use this approach to evaluate our design. We integrate the Globus XIO based file transfer mechanisms into sample applications based on the client - server model similar to the one shown in Figure 5. As we can see, with the Globus XIO framework, it takes very little effort for an application to effectively use a high performance file transfer library such as the ADTS.

5 Experimental Results

In this section we describe the experimental setup, provide the results of our experiments and also give in-depth analysis of the results.

5.1 Experimental Setup

The experimental setup consists of two hosts connected through the Obsidian WAN routers as shown in Figure 6. The hosts are equipped with the Intel Nehalem series of processors with Dual quad-core processor nodes operating at 2.40GHz with 12 GB RAM and a PCIe 2.0 interface. Though we have DDR/QDR rate cards available, the total available bandwidth is limited by the ingress and egress transmission rates offered by the Obsidian routers. The Obsidian router provides one IB SDR port and two GigE ports at the ingress and one 10 GigE Sonet port at the Egress. So in effect, the bandwidth attainable through the Obsidian routers for an IB device is limited to the 8Gbps transmission rate the ingress SDR port affords us. Red Hat Enterprise Linux Server release 5.3 with Linux Kernel version 2.6.18-128 was used on both the hosts. For all IPoIB (TCP/IP) based tests, auto-tuning of the socket buffers was enabled. Due to the bug [34] with the *CUBIC* congestion control protocol [23] in the 2.6.18 kernels, the *HTCP* congestion control mechanism was used. OFED version 1.4.2 was installed on both the hosts.

5.2 Microbenchmark Level Evaluation

In this section we present the results of the microbenchmark level experiments carried out on the experimental systems mentioned above.

5.3 Memory to Memory Data Transfer

In this experiment, the data transfers were performed from host memory directly to target memory. This was done so as to measure the raw performance of the underlying network protocol with varying network delay. As we can see from Figures 7 (a) and (b), while the ADTS based implementation is able to either saturate the link bandwidth or consume a substantial portion of it successfully, the IPoIB based implementation is only able to utilize 10% - 15% of the total link bandwidth. Each point in the graph represents the throughput number obtained while transmitting 128 GB of aggregate data as multiples of 256 MB files. An interesting thing to note here is that, for low delay scenarios, the best performance is obtained when we

```
int
main(
    int                             argc,
    char *                          argv[])
{
    globus_result_t                 res;
    globus_xio_driver_t             adts_driver;
    globus_xio_stack_t              adts_stack;
    globus_xio_handle_t             handle;
    globus_size_t                   nbytes;
    char *                          contact_string = NULL;
    char                            buf[256];
    int                             server = 0;

    contact_string = argv[1];
    adts = argv[2];
    server = argv[3];


    globus_module_activate(GLOBUS_XIO_MODULE);
    res = globus_xio_driver_load(
            "adts",
            &adts_driver);
    assert(res == GLOBUS_SUCCESS);

    res = globus_xio_stack_init(&adts_stack, NULL);
    assert(res == GLOBUS_SUCCESS);
    res = globus_xio_stack_push_driver(adts_stack,
                                        adts_driver);
    assert(res == GLOBUS_SUCCESS);

    res = globus_xio_handle_create(&handle,
                                    adts_stack);
    assert(res == GLOBUS_SUCCESS);

    res = globus_xio_open(handle, contact_string, NULL);
    assert(res == GLOBUS_SUCCESS);

    if (server) {
        res = globus_xio_write(handle, buf,
                sizeof(buf) - 1, 1, &nbytes, NULL);
        assert(res == GLOBUS_SUCCESS);
    } else {
        res = globus_xio_read(handle, buf,
                sizeof(buf) - 1, 1, &nbytes, NULL);
        assert(res == GLOBUS_SUCCESS);
    }
    globus_xio_close(handle, NULL);

    globus_module_deactivate(GLOBUS_XIO_MODULE);

    return 0;
}
```

Figure 5. Example Globus XIO ADTS Program

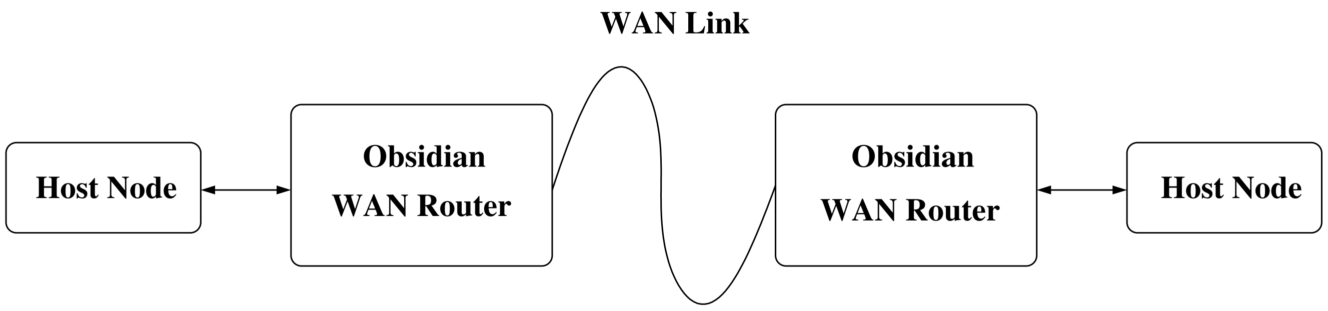

Figure 6. Experimental Setup

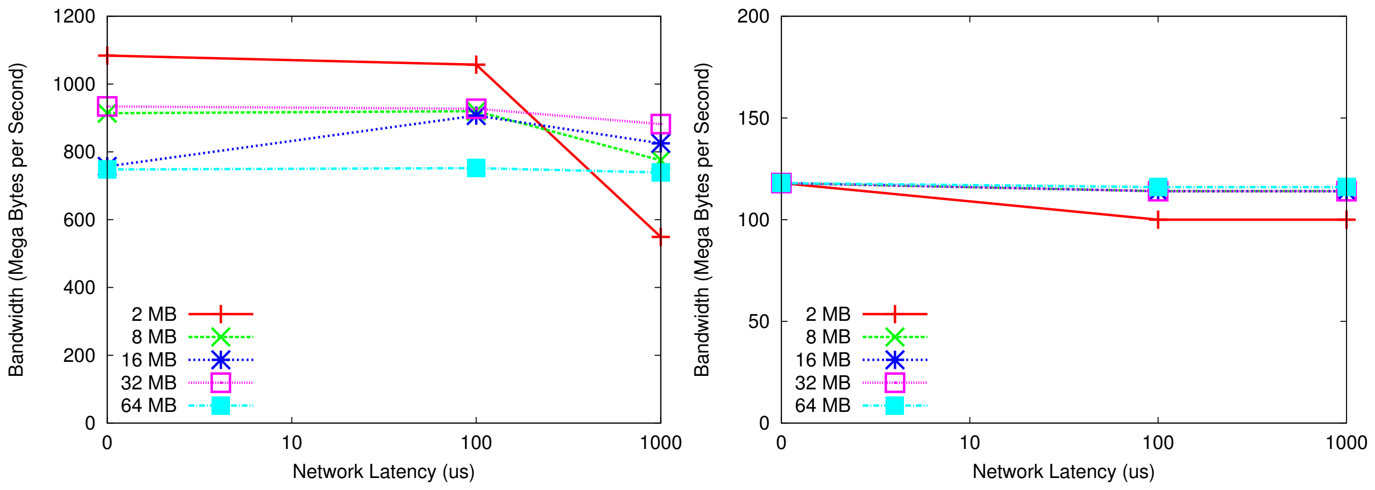

Figure 7. Performance Memory Based FTP *Get* Operation for (a) ADTS Driver and, (b) TCP Driver

use smaller file buffers. Buffering packets helps in improving file transfer performance only when I/O performance is worse than the network performance or when there is high delay in the network. Since we are doing memory transfers and the performance of memory operations is much better than that of network based operations, we will in fact see a reduction in performance in low delay scenarios as the size of the file buffer size. But when there is considerable delay in the network, as is the case with 1 ms network delay, buffering the packet actually helps as we are able to dedicate more time to polling the network. Overall, we see that a file buffer size of 32 MB gives the best performance for varying network delays.

5.4 Disk to Disk Data Transfer

The performance comparison of ADTS and IPoIB based GridFTP *Get* operation for disk to disk data transfer is shown in Figure 8. Though the performance is considerably less as opposed to the memory based file transfers, the trends seen are similar to those observed for the memory to memory transfers. From the figure, we can see that the performance of ADTS based GridFTP is almost twice as better compared to that of IPoIB based GridFTP in almost all scenarios. As the IPoIB numbers do not show any difference in performance for varying delays for any of the file buffer sizes used, we only show the case which gives the best performance (64MB file buffer size). From this graph, we can conclude that there is no one size of the file buffer that suits all possible network delays. The file buffer size will have to chosen carefully and dynamically based the network characteristics prevalent at that time.

5.5 Disk to Disk Data Transfer with Data Staging and Asynchronous I/O

The performance of staged asynchronous disk to disk I/O based GridFTP operations is shown in Figure 9. Here we can clearly see that we are able to get better performance by using larger file buffer size. As expected, the staged asynchronous I/O based approach does better than the synchronous I/O based approach seen in the previous section for medium to large delays as it helps in keeping the pipeline full by allowing the network thread to poll the network more frequently. The IPoIB based approach gets saturated at a much lower level as usual. As the IPoIB numbers do not show any difference in performance for varying delays for any of the file buffer sizes used, we only show the case which gives the best performance (64MB file buffer size).

5.6 Application Level Evaluation

We have mentioned in Section 3.1 that GridFTP is widely used in many applications. In this section, we use two applications to evaluate the performance of our design. The first application is the replication of Community Climate System Model (CCSM) [14] hosted at National Center for Atmospheric Research (NCAR) as part of the Earth System Grid [16] project. The intention is replicate large volumes of data (in the order of 160 TBytes) sourced at NCAR to other participating sites

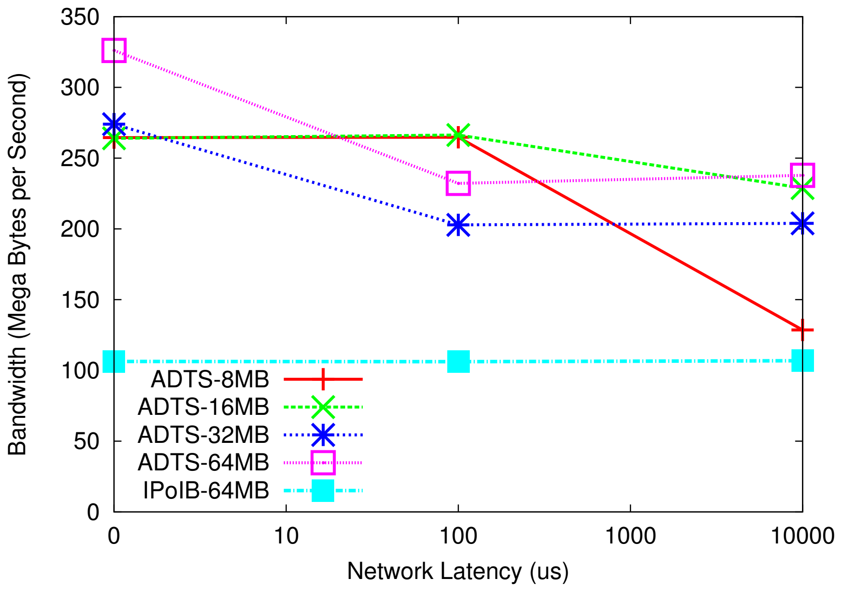

Figure 8. Performance of FTP *Get* Operation for Disk Based Transfers

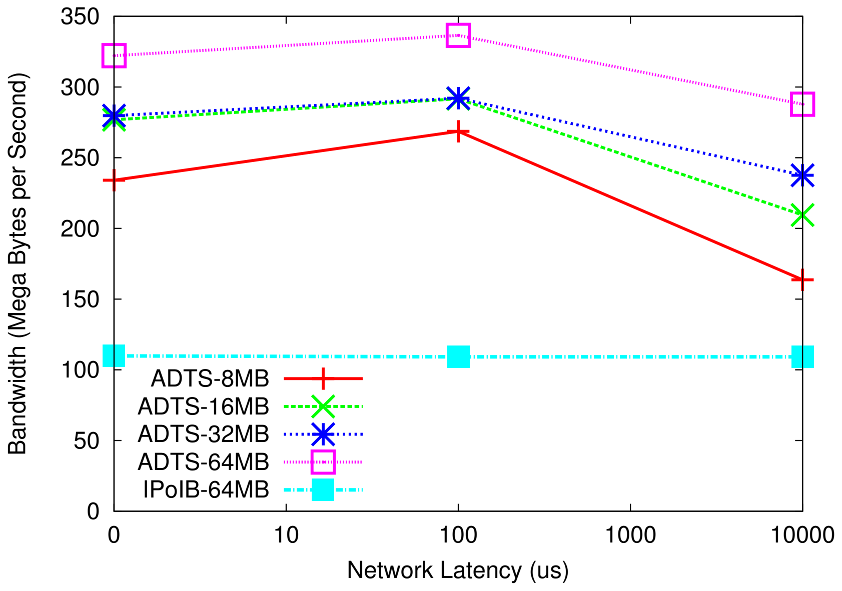

Figure 9. Performance of FTP *Get* Operation for Disk Based Transfers with Staging and Asynchronous I/O

such as Lawrence Livermore National Laboratory (LLNL). The files are in the order of 256 MBytes. We use our design to emulate this data replication between two clusters connected with Obsidian Longbows. The latency between NCAR and LLNL is around 30 ms which we use to configure the Longbow latency. The other application is the Ultra Scale Visualization project that transfers large numbers of 2.6 GB files between ORNL and UC Davis. As before, we use Obsidian Longbows to simulate the distance between these two sites by setting the latency to 80 ms.

Figure 10 shows the bandwidth of our design and the IPoIB based design during the data replication. We clearly see that the new design outperforms the IPoIB based design by more than a factor of two. Even with such large delay, our design can sustain good performance. This presents the prospective future use for the new design in real grid scenarios.

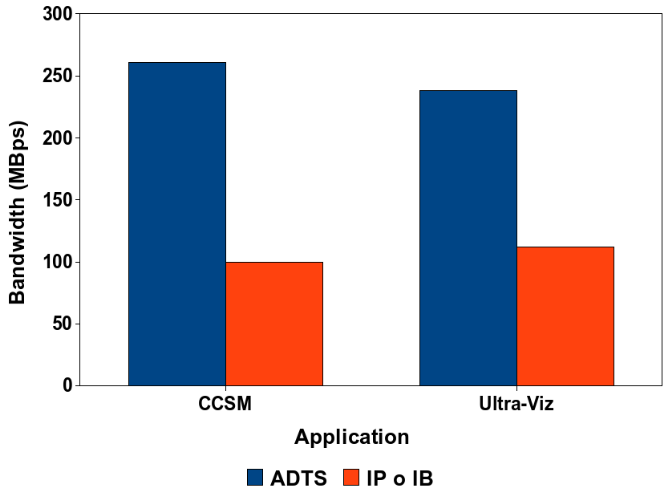

Figure 10. Performance of FTP *Get* Operation for CCSM Application and Large Scale Visualization Traces

6 Related Work

Researchers have investigated FTP from multiple angles including security, performance, distributed anonymous FTP and extensibility [3, 35]. The extension to support IPv6 and transfer files over Network Address Translators (NATs) is introduced in [7]. In [2], the authors have proposed GridFTP which performs efficient TCP based transfers through the use of multiple streams for each transfer. Also, scientists aimed to improve multiple file transfers using SCTP multi-streaming, parallel transfers [6]. The use of UDP based transfers has been explored in [5, 4], in order to overcome some of the limitations in TCP. Our group has investigated the use of high performance transport protocols supported by advanced interconnects in FTP design [12]. In this work, we designed and implemented the ADTS layer and evaluated its impact on the FTP performance.

There has been an active development of solutions to address limitations of TCP's AIMD-based congestion control mechanism [36]. These solutions include improvements to TCP such as Binary Increase Congestion Control [37], CUBIC [23], High-Speed TCP (HSTCP) [38], Hamilton TCP (HTCP) [22], Scalable TCP [21], new transport protocols such as XCP [39], XTP [40] and reliable layers on top of UDP [24, 25, 41, 42] that target high data rates over wide-area connections. But the task of sustaining end-to-end throughput at rates of 10 Gbps over thousands of miles still remains complex. Recently, UDT has been integrated into GridFTP as an alternative transport protocol for TCP [26], However, the user level processing in UDT require more CPU and memory resources than TCP.

Regarding InfiniBand area, researchers have explored its advanced features to design better middleware and applications. The work on NFS over RDMA demonstrates better performance [43] due to the benefits of RDMA in several scenarios. Literature [44, 45, 46, 47] investigate various aspects of the performance characteristics over IB WAN.

7 Conclusions and Future Work

In this paper we take on the challenge of designing an easy to use, high performance FTP library, drawing on the advantages of ease of use given by the Globus XIO framework used by GridFTP and the high performance nature of the ADTS

FTP library designed by us. We enhance the existing ADTS design by adding new elements to efficiently decouple the network and disk I/O processing. Such decoupling allows us to get better performance for real world applications in WAN scenarios with large delays as evidenced by the results of our microbenchmark as well as application level evaluations. The results of our experimental evaluation clearly indicates the benefits of using a high performance transport driver like ADTS for GridFTP. We see that the ADTS based design outperforms the existing designs based on IPoIB (TCP/IP, UDP) by a factor of 2 in LAN as well as WAN scenarios.

As part of future work we plan to evaluate the performance of GridFTP running over ADTS on InfiniBand interfaces that have support for the RDMA over Ethernet protocol. This will allow us to run directly on real 10GigE networks and perform more in-depth analysis of the effects of TCP/IP cross traffic on IB flows in ethernet networks. Such an experimental setup would also allow us to compare IB directly with TCP/IP - something that we have not been able to do well till now. We also plan to run I/O benchmarks such as IOZone [48], Bonnie [49] and FIO [50] to evaluate how effective our disk I/O performance is as opposed to the best possible value that can be obtained as shown by these benchmarks.

8 Acknowledgments

We would like to thank Jason Gunthorpe and David Southwell of Obsidian Strategics for their valuable help and support in making this paper possible. We would also like to thank Michael Link of Argonne National Labs for the help offered with building GridFTP with the ADTS library.

References

[1] J. Postel and J. Reynolds, “File Transfer Protocol. RFC 959.”

[2] W. Allcock, “GridFTP: Protocol Extensions to FTP for the Grid. Global Grid ForumGFD-R-P.020,2003.”

[3] M. Allman and S. Ostermann, “Multiple Data Connection FTP Extensions,” Ohio University, Tech. Rep., 1996.

[4] D. Bush, “UFTP,” http://www.tcnj.edu/ bush/uftp.html.

[5] K. R. Sollins, “The Trivial File Transfer Protocol – TFTP 2 RFC 1350. July, 1992.”

[6] S. Ladha and P. D. Amer, “Improving Multiple File Transfers Using SCTP Multistreaming,” in *Int’l on Performance, Computing, and Communications Conference*, April 2004.

[7] M. Allman, S. Ostermann, and C. Metz, “FTP Extensions for IPv6 and NATs. RFC 2428. Sep. 1998.”

[8] “Infiniband trade association,” http://www.infinibandta.org.

[9] “Architectural Specifications for RDMA over TCP/IP,” `http://www.rdmaconsortium.org/`.

[10] “Obsidian research corp.” http://www.obsidianresearch.com/.

[11] Net NX 5010 High Speed Exchange, http://www.net.com/products/products-nx.shtml.

[12] P. Lai, H. Subramoni, S. Narravula, A. Mamidala, and D. K. Panda, “Designing Efficient FTP Mechanisms for High Performance Data-Transfer over InfiniBand,” in *the 2003 International Conference on Parallel Processing (ICPP 09)*, Sep. 2009.

[13] William Allcock and John Bresnahan and Rajkumar Kettimuthu and Joseph Link, “The Globus eXtensible Input/Output System (XIO): A Protocol Independent IO System for the Grid,” *Parallel and Distributed Processing Symposium, International*, vol. 5, p. 179a, 2005.

[14] Community Climate System Model, http://www.ccsm.ncar.edu.

[15] Institute for Ultra Scale Visualization, “Ultravis: Ultra Scale Visualization,” http://ultravis.ucdavis.edu/.

[16] “Earth System Grid,” http://www.earthsystemgrid.org/.

[17] "SciDAC Computational Astrophysics Consortium," http://www.supersci.org/.

[18] "Energy Sciences Network (ESnet)," http://www.es.net/.

[19] W. Allcock, J. Bresnahan, R. Kettimuthu, and M. Link, "The Globus Striped GridFTP Framework and Server," in *Proceedings of the 2005 ACM/IEEE conference on Supercomputing*, 2005.

[20] C. Jin, D. X. Wei, and S. H. Low, "FAST TCP: Motivation, Architecture, Algorithms, Performance," in *Proceedings of IEEE Infocom*, 2004.

[21] T. Kelly, "Scalable TCP: Improving Performance in Highspeed Wide Area Networks," in *First International Workshop on Protocols for Fast Long Distance Networks*, 2003.

[22] M. Allman, V. Paxson, and W. Stenvens, "TCP Congestion Control," in *IETF, RFC-2581*, 1999.

[23] H. S, I. Rhee, and L. Xu, "CUBIC: A New TCP-Friendly High-Speed TCP Variant," in *SIGOPS Operating System Rev*, 2008.

[24] Y. Gu and R. L. Grossman, "UDT: UDP-based Data Transfer for High-speed Wide Area Networks," in *Computer Networks: The International Journal of Computer and Telecommunications Networking*, 2007.

[25] "Faster Bulk Transfer Starring: UDP," http://www.csm.ornl.gov/ dunigan/netperf/udp/UDP_RBUDP.html.

[26] J. Bresnahan, M. Link, R. Kettimuthu, and I. Foster, "UDT as an Alternative Transport Protocol for GridFTP," in *Proceedings of the 7th International Workshop on Protocols for Future*, 2009.

[27] "Open fabrics enterprise distribution," http://www.openfabrics.org/.

[28] "LHC: Large Hadron Collider," http://en.wikipedia.org/wiki/Large_Hadron_Collider.

[29] R. Consortium., http://www.rdmaconsortium.org/home/draft-recio-iwarp-rdmap-v1.0.pdf.

[30] Mellanox OFED Stack for Linux Users Manual, http://www.mellanox.com/pdf/products/software/ Mellanox_OFED_Linux_User_Manual_1_20.pdf.

[31] Davtd A Patterson, Garth Gibson, and Randy H Katz, "A Case for Redundant Arrays of Inexpensive Disks (RAID)," University of California, Berkeley, Tech. Rep., 1988.

[32] "Grid Computing," http://en.wikipedia.org/wiki/Grid_computing.

[33] Ian Foster, Carl Kesselman, Steven Tuecke, "The Anatomy of the Grid."

[34] Office of Science, DOE, http://fasterdata.es.net/TCP-tuning/linux.html.

[35] F. Anklesaria and et.al, " The Internet Gopher Protocol. RFC 1436. Network Working Group, Mar. 1993."

[36] M. Allman, V. Paxson, and W. Stevens, "Tcp congestion control," United States, 1999.

[37] L. Xu, K. Harfoush, and I. Rhee, "Binary increase congestion control (bic) for fast long-distance networks," in *IEEE Infocom*. IEEE, 2004. [Online]. Available: http://www.ieee-infocom.org/2004/Papers/52_4.PDF

[38] "HighSpeed TCP," http://www.icir.org/floyd/hstcp.html.

[39] D. Katabi, M. Handley, and C. Rohrs, "Congestion control for high bandwidth-delay product networks," in *SIGCOMM '02: Proceedings of the 2002 conference on Applications, technologies, architectures, and protocols for computer communications*. New York, NY, USA: ACM, 2002, pp. 89–102.

[40] W. T. Strayer, M. J. Lewis, and R. E. Cline, Jr., "Xtp as a transport protocol for distributed parallel processing," in *HSNS'94: Proceedings of the High-Speed Networking Symposium on USENIX 1994 High-Speed Networking Symposium*. Berkeley, CA, USA: USENIX Association, 1994, pp. 6–6.

[41] . Tsunami Network Protocol Implementation, http://www.indiana.edu/ uits/cpo/tsunami.

[42] R. L. Grossman, M. Mazzucco, H. Sivakumar, Y. Pan, and Q. Zhang, “Simple available bandwidth utilization library for high-speed wide area networks,” *J. Supercomput.*, vol. 34, no. 3, pp. 231–242, 2005.

[43] “Sun Microsystems, Inc. and The Ohio State University. NFS over RDMA Design, Version 1.1, Aug 2007.”

[44] S. Carter, M. Minich, and N. S. V. Rao, “Experimental evaluation of infiniband transport over local- and wide-area networks,” in *High Performance Computing Symposium*, 2007.

[45] W. Yu, N. S. Rao, P. Wyckoff, and J. S. Vetter, “Performance of rdma-capable storage protocols on wide-area network,” in *Peta-byte Storage Workshop, in conjunction with SC08*, 2008.

[46] S. Narravula, H. Subramoni, P. Lai, R. Noronha, and D. K. Panda, “Performance of HPC Middleware over InfiniBand WAN,” in *Int’l Conference on Parallel Processing*, Sep. 2008.

[47] N. S. V. Rao, W. Yu, W. R. Wing, S. W. Poole, and J. S. Vetter, “Wide-area performance profiling of 10GigE and InfiniBand technologies,” in *Proceedings of the 2008 ACM/IEEE conference on Supercomputing*, 2008.

[48] “IOzone Filesystem Benchmark,” http://www.iozone.org/.

[49] “Bonnie - IO Throughput Benchmark,” http://www.garloff.de/kurt/linux/bonnie/.

[50] “FIO - Linux IO Benchmarking Tool,” http://freshmeat.net/projects/fio/.