Submitted for publication to CCGrid 2003

Network Emulation Tools for Modeling Grid Behavior

Xin Liu, Huaxia Xia, and Andrew A. Chien
Department of Computer Science and Engineering
University of California, San Diego
{lxin, huaxia, achien} @ csag.ucsd.edu

Abstract

There is increasing interest in Grids that use compute, storage, and network resources in configurations involving both flexible sharing and tight-coupling. Such usage produces complex dynamics, subject to complex non-linear behavior. We believe that detailed packet level simulation is both possible and necessary forhigh fidelity grid simulations. We present the MicroGrid, a tool set which allows researchers to emulate Grid resources and applications directly. The MicroGrid tools include a scalable network emulatorMaSSF whose design can support emulation thousands of grid resources and application elements.. We demonstrate the MicroGrid, byapplyingthree types of applications: 1) basic parallel applications (NASNAS Parallel Benchmarks), 2) peer to peer applications (Gnutella file sharing), and 3) in a detailed evaluation of GridFTP on the TeraGrid. These studies exploit the packet-level simulation capabilities of MaSSF to expose detailed dynamic communication behavior. In the cases of Gnutella and GridFTP, these dynamic characteristics provide critical insights into how to use and improve the application and resource configurations.

1. Introduction

The advent of high speed wide-area networks, the emergence of a shared grid software infrastructure, and most recently the endorsement by major companies are creating an accelerating momentum behind Computational Grids, the large scale federation of compute and data resources and coordinated sharing of such resources [1]. Such technologies have tremendous potential in enabling flexible collaboration, resource sharing, and even the vision of compute utilities. However, realizing these promises still faces major challenges. While Grid-enabled applications can access and exploit geographically distributed computing, communication, and storage resources, they also face many challenges.

Grids are beginning to be deployed (i.e. the NASA Information Power Grid, UK E-science Grid, the AP Grid, and the TeraGrid), and the resulting systems are emerging as extremely complex resource environments, with wide ranges of network, compute, and data resource capabilities in the Grid. The dynamic behavior of these resource environments, when subject to actual application workloads, complicated by adaptive behavior on the part of the applications themselves [2, 3] is complex, and difficult to predict or model. The coupling of resources by grid applications causes complex, non-linear dynamics which are critical to Grid resource stability. While we already have tools for modeling tools for processor, network and disk respectively, it is hard to put them together and build a reasonable performance model for the complex Grid application. To date there are neither tools nor a methodology to characterize the dynamic behavior of grid resources or applications. Traditional practice is to perform actual test-bed experiments (at moderate scale) or simplistic simulations that are not validated. This challenge is the focus of this work, providing tools and using them to characterize dynamic Grid behaviors.

We have developed a set of simulation tools called the MicroGrid that enable systematic design and evaluation of middleware, applications, and network services for the Computational Grid. These tools provide an environment for scientific and repeatable experimentation [4]. The simulation detail of these tools is designed to match the needs indicated by a range of published studies illustrating performance anomalies in Grid computing systems. The MicroGrid includes compute, memory, and disk I/O performance models and combines them with a scalable packet-level network emulator. In addition, the MicroGrid provides a virtualized resource environment which enables Globus applications to run unmodified in the virtual performance environment. Thus, entire real applications can be used in performance studies [4]. These tools have been validated on a range of existing grid applications and common resource configurations.

It is our hope that the MicroGrid will catalyze the development of experimental methodologies to robustly extrapolate grid simulation results and rational

Grid design and management. The key component of MicroGrid is network emulator MaSSF that support realistic grid software environments, modeling of a wide variety of resources, and scalable performance.

MaSSF is a detailed network emulator, which will model the behavior of each single IP packet as ns. This will guarantee the accuracy of MaSSF but raise the scalability issue. Our approach is to build the network emulator over a distributed simulation engine to exploit the availability of scalable hardware (like clusters with high speed connection). We adapt the DaSSF simulation engine [5] for emulation by adding real-time synchronization mechanism. Network protocol stack (IP, TCP, OSPF and BGP4) are developed over DaSSF.

We focus on demonstrating the capabilities of the MicroGrid approach to Grid resource and application modeling. First we apply MicroGrid to a number of simple Grid applications to show the breadth of its capability. Second, we explore the emergent behavior of peer-to-peer applications, characterizing their detailed network behavior, based on detailed packet-level simulations. Finally, we use the MicroGrid and MaSSF to make a deep study of GridFTP on the yet-to-be-deployed TeraGrid. By introducing link loss, router loss, link speed variation in the emulator, we can study how GridFTP behaves in the future configuration of the TeraGrid. The specific insights gained from these studies include:

- deployment of peer-to-peer file sharing techniques can offload backbones, improving scalability by three times in our study,
- detailed network simulations using MicroGrid and MaSSF show that LAN bandwidth usage increases dramatically, three to five times, but only after an initial "warmup" period,
- GridFTP performance on Teragrid network (backplane) are limited by TCP's ability to fill the high bandwidth delay product channels, and deliver only 16MB/s of a possible 120MB/s, and
- the performance is extremely sensitive to bursty packet losses, with examples of 3x bandwidth variability depending on how peering and load distribution is performed by the routers. This suggests that careful study of these alternatives is important in the actual Teragrid deployment.
-

The remainder of the paper is organized as follows. Section 2 formalizes the research problem of modeling grids. A description of our MicroGrid approach can be found in Section 3. Section 4 describes the application of the MicroGrid tools to a range of applications and the resulting insights. Section 5 discusses a detailed study of GridFTP performance. Section 6 discusses related work, and Section 7 summarizes and describes some future directions.

2. The Problem

Modeling the dynamic behavior of Grid resources under load and Grid applications is challenging due to the non-linear behavior of system elements and the vast scale of the resources as well as state space that must be explored. To effectively support the development of robust grid applications and middleware software as well as the rational design and configuration of Grid resources, tools must:

- use real grid applications and middlware software,
- provide high fidelity models for resource performance,
- include accurate models for processing, memory, input/output, and networks,
- exploit detailed resource configuration information where available [6], and
- scale to millions of grid application, middleware, and resource elements.

These requirements suggest a set of tools which virtualize the dominant Grid middleware environment (i.e. Globus), use parallel computing resources to scale, and use detailed network modeling (packet level simulation) to preserve the most critical dynamics in Grid software behavior – communication and network behavior.

3. MicroGrid and MaSSF

3.1. MicroGrid Architecture

The basic functionality of the MicroGrid is to allow Grid experimenters to simulate their applications on a virtual Grid environment. The simulation tool should be made flexible to allow accurate experimentation on heterogeneous physical resources, as shown in Figure 1. Our MicroGrid implementation supports Grid applications which use the Globus Grid middleware infrastructure.

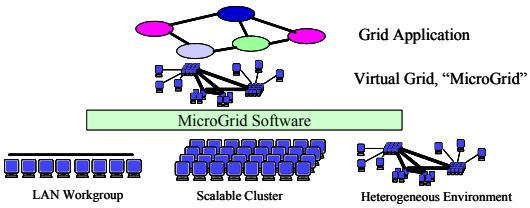

Figure 1. Architecture of the MicroGrid tools.

The basic architecture of the MicroGrid is shown in Figure 1, and each element in the figure addresses one of the key challenges in constructing a high-fidelity virtual Grid. Specifically:

Virtualization: The application perceive only the virtual Grid resources (host names, networks), independent of the physical resources being utilized. This is achieved by virtulizing the Grid information services, virtualizing/emulating the appropriate resources.

Global Coordination: We provide a coherent global simulation of varying numbers of varying virtual resources on heterogeneous physical resources. One major function is to coordinate the simulation speed of different virtual resources.

Resource Simulation: Each virtual resource (host, cpu, network, disk, etc.) is modeled accurately as an element of the overall simulation.

Our approach towards overcoming these challenges was described in detail in [4]; here we focus on the new scalable network emulation component.

3.2. MaSSF

MaSSF (pronounced "massive") is a packet-level network emulator build on distributed simulation engine DaSSF, which mainly consists of four parts.

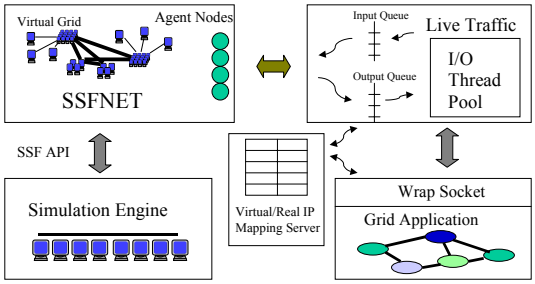

Figure 2. The MaSSF Scalable Network Emulation System.

Simulation Engine: We adapt the DaSSF as the underlying simulation engine. DaSSF can utilize MPI-connected cluster systems to achieve scalable performace. This is one of the key components to make MaSSF scalable. We extend DaSSF with a real-time scheduler, which will enable best-effort emulation. This scheduler can also running on a scale down mode when the emulated system is too large to be acted in real time. With the *Global Coordination* of MicroGrid, this feature provides extreme flexibility to emulate any large scale networks accurately.

Network Modeling: DaSSF provides a generic programming interface (SSFAPI) that the network protocol stack can be built above it. Based on this API, we re-implement all the necessary network protocols, such as IP, TCP/UDP, OSPF, and BGP. We try to simplify these protocols and maintain their behavior characteristics at the same time. Parts of our codes are taken from a JAVA implementation SSFNET [5].

Emulation Capability: The third step is to add emulation capability to the network protocol stack. The major issue here is how to match things executing in emulation and real time.

Live traffic intercept and redirection: intercept live network streams from real application and redirect them to the network emulator.

4. Modeling Parallel and Peer-to-Peer Applications

We present experimental data derived from exercising MaSSF on parallel and peer-to-peer applications. These studiesdemonstrate the generality and flexibility of the system.

4.1. NAS Parallel Benchmarks

MaSSF can be used to emulate large scale scientific computation over LAN, Campus network, and Wide Area Network. Here we emulate NPB Class A benchmark [7] over 64 processors. The network configurations are shown in Table 1.

	# Procs	Network
Cluster	64	100BaseT Switched
Campus	64	Two 32P clusters, 1Gbps MAN
TeraGrid	64	Four 16P clusters, 40Gbps WAN

Table 1. Network Configurations for NPB Studies

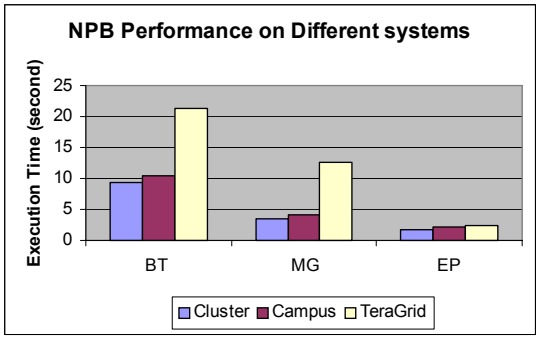

Figure 3. NPB Performance on different systems

The results shown in Figure 3, demonstrate how the performance of the NAS Parallel Benchmarks scales over different network capabilities and topologies (BT is scaled down 10 time for better presentation). Because the bisection bandwidth varies from 3.2Gbps to 1Gbps to 40Gbps, while latency increases from 1ms to 3ms to 30ms, it is clear that latency dominates. With the exception of EP, increased latency, producing poor performance when systems are more widely

distributed. This confirms that despite the high bandwidth of Teragrid's backplane, Grid jobs which span Teragrid the commonly need be latency tolerant to achieve good performance.

4.2. File sharing: Centralized or Peer-to-Peer

One of the most exciting areas of distributed systems design today focuses on decentralized systems / protocols often referred to generically as "peer-to-peer". The design of these systems is particularly challenging, as the large-scale emergent behavior [8] is the objective, and often not directly inferable from the small-scale behavior and detailed design of the protocols. We apply the the MaSSF network simulation to study loosely-coupled peer-to-peer applications.

File sharing in a large organization and company is always a big concern. Traditional approaches include a high-end NFS/DFS file server or a web server. The typical scaling limit on these centralized solutions is the CPU/traffic load on the server and backbone network. Using the MaSSF tools, we explore the use of peer-to-peer tools to distribute the files. Consider the scenario depicted in Figure 4.

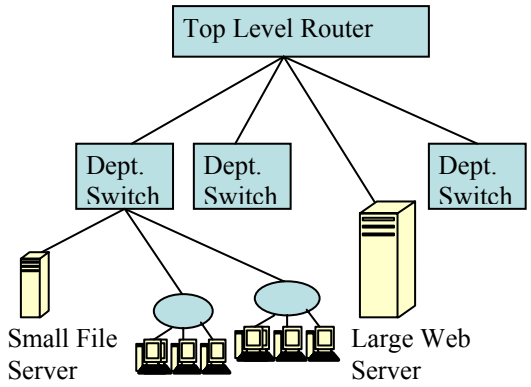

Figure 4. A typical corporate network.

The centralized approach deploys a large file server, connected to the "backbone" router. Users access files thru their local LAN network, up through a departmental switch, and then eventually up into the backbone router. Scalability limits for this system include the central server, the backbone network links, and the routers. All of these elements are expensive both in terms of hardware cost and management, so adding capacity can be expensive.

Peer-to-peer approaches first distribute files from a central server, and then move the files using the peer-to-peer file sharing protocol. The files are cached on various client (peer) nodes throughout the system, and the file sharing protocols enables users' clients to locate the needed files nearby and access them. In this discussion, we focus on the read-only traffic, which most often is the dominant component and defer the write traffic to a future study. The particular peer-to-peer file sharing protocol we study is Gnutella [3].

For workload, we use synthesized file requests against a set of synthetic files. The files averaged 2 megabytes and we used 5000 distinct files. 200 users accessed the files from different client machines, with a random access pattern and 30 second think time distributed as a poisson process. Five departments (segments) are assumed. Varying these parameters allows control over the intensity of file traffic and study the total throughput of two approaches. With our detailed network simulation, we can not only detect the scalability limits (see Figure 5), but also detailed network usage behavior, including instantaneous bandwidth. For example, we can characterize the resulting network traffic on the backbone network (which is the scalability limit) as well as the departmental and LAN networks (to explore the impact on local network infrastructure). Because these local infrastructures are rather costly to update, they are often rather outdated in many enterprises (e.g. 10Mbit switched or shared or 100Mbit shared).

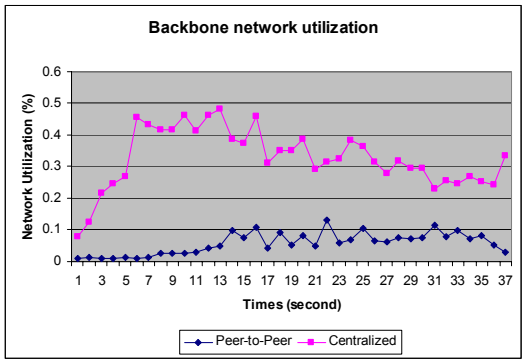

Figure 5. Network Backbone utilization over time for Centralized server solution and Gnutella based file distribution.

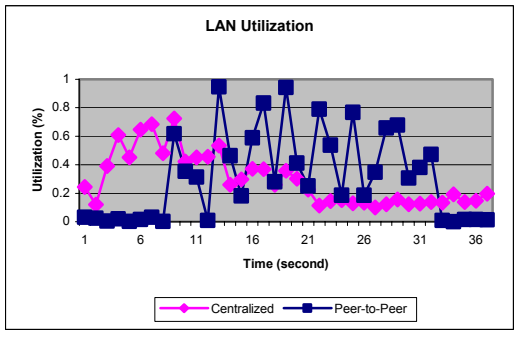

Figure 6. Departmental network (LAN) utilization over time for Centralized server solution and Gnutella based file distribution.

The results in Figures 5 and 6 clearly illustrate the capability of peer to peer technology to offload the central server and the backbone network. After a cold start period, the peer-to-peer approach reduces backbone bandwidth consumption three-fold, freeing that capacity for other uses. Figure 6 shows a complementary increase in LAN bandwidth consumption – to be expected, but its magnitude of increase is remarkable. LAN network usage increase 3-5x after the cold start period. Some of this increase can be attributed to the earlier completion of the file access trace under the peer-to-peer approach due to its more rapid servicing of file requests performance (see tail of Figure 6.). Our detailed packet-level simulation not only reveals the detailed bandwidth, but could for example detail packet sizes and the benefits of supporting larger MTU's in the LANs (or other such network level performance issues).

5. Modeling Future Grids: GridFTP Performance on the TeraGrid

The MicroGrid and MaSSF can not only be used to model existing hardware systems, but it can also be used to model future and even hypothetical hardware Grid systems. Such capability can be used for planning (Should I add this resource? How much will it help?), design (How should these elements and systems be combined in a Grid?), and even retrospective diagnosis (What happened?). In this case, we employ the MicroGrid to model the future behavior of a major Grid infrastructure being deployed in the United States, the Teragrid (see http://www.teragrid.org/).

5.1. TeraGrid

The TeraGrid is being deployed within the NSF TeraScale initiative. The initial TeraGrid design, planned for operation in early 2003, includes five large-scale Linux clusters at ANL, Caltech, NCSA, PSC, and SDSC. Modeling the TeraGrid is challenging because its network and cluster node performance represents an extreme test of scale. In addition, because the Teragrid is an important system and a major investment, we seek to understand its behavior well before all of the decisions which affect its performance are committed. Naturally, the fact that the Teragrid does not yet exist provides a showcase for the MicroGrid's ability to model future infrastructure.

5.2. GridFTP

The GridFTP protocol [9, 10] is a FTP protocol extension proposed by the Globus Project to provide a secure, efficient data transport mechanism over Grid. Since GridFTP will be the underlying data transfer engine of many Grid projects, it is very important to be able to predict its performance and capabilities over varied network configuration. A major feature of the GridFTP protocol is parallel data transfers (the use of multiple TCP streams between the same source and destination to improve aggregate bandwidth over wide-area network). The GridFTP spec also includes support for striped transfers, but no implementation supporting such features was available as of this writing. Our focus here is using MaSSF tools to characterize current performance and suggest directions for improvement.

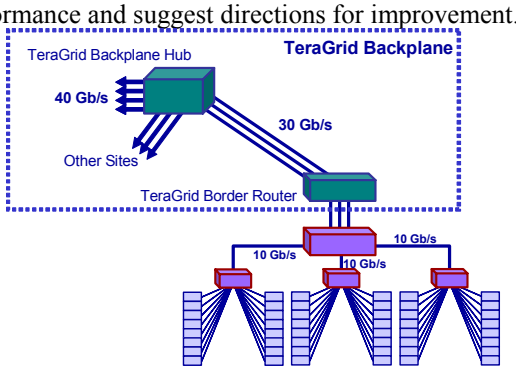

Figure 6. TeraGrid Site Network Architecture for any of the five sites.

5.3. Experiments

The globus-url-copy tool and in.ftpd server, released the Globus Toolkits 2.2 were used. A test file of a size of 1GByte is transferred between sites, and the bandwidth is calculated based on the transfer time. Throughput measurements with respect to different number of parallel threads were performed between NCSA and SDSC. We also study the GridFTP behavior under different background traffic, TCP send buffer size, and link loss rate.

5.3.1. Baseline Data Transfer Performance

To achieve high bandwidth, TeraGrid uses parallel Layer 2 links which are grouped to look like a single link to Layer 3. Internally the routers pick physical links round robin for different flow and hash based on the flow ID so that a flow is restricted to a single physical link. So if an application uses only one TCP flow, it cannot benefit directly from parallel physical links. For GridFTP, we expect at least four parallel flows to be needed to capture the benefits of the TeraGrid Backplane.

We study performance along a path which goes from SDSC to LA through a 3x10Gbit/s link, then to Chicago through a 4x10Gbit/s link, and then to NCSA through a 3x10Gbit/s link. At the local level on both sides (as shown in Figure 6), individual hosts are

connected via a 1Gbit/s link to a 10Gbit/s uplink. Performing parallel download and varying the number of parallel flows from 1 to 32 increased throughput from 8.3 MB/s to 16.5MB/s (See Figure 7). However, when competition with seven other gridftp sessions (each with 6 parallel flows) is introduced, the maximum bandwidth is reduced to 13.8MB/s. Note that the benefit of parallelism is increased, nearly 6-fold from 2.56MB/s to 13.8MB/s. In all cases, GridFTP is far from the hardware limit of 1Gbit/s.

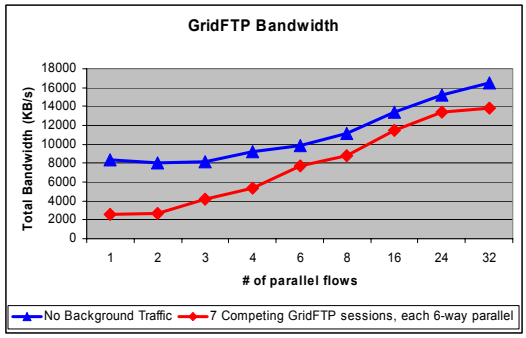

Figure 7. Baseline Data Transfer Performance

Using the detailed packet-level simulation of MaSSF, we can explore the detailed dynamic behavior, which is much more complex. We plot the instantaneous network bandwidth delivered by GridFTP in Figure 8. As you can see, the bandwidth is unstable, and varies widely – making poor use of the network resources, and potentially introducing oscillation in the system. This effect appears to be due to TCP's congestion control. With MaSSF, this can be easily verified by dumping the TCP window sizes for each TCP session. A typical trace is shown in Figure 9, and we can see the effect of additive increase, multiplicative decrease of window size in TCP. This appears to also be the performance limiting facter, since due to the large bandwidth-delay product of TeraGrid (either 1Gbps * 12ms or 10Gbit * 12ms), TCP needs a large slide window size to fill the pipe.

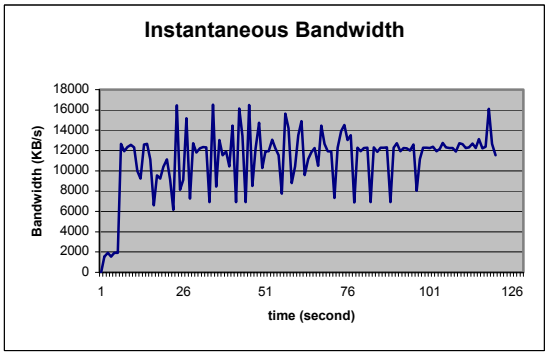

Figure 8. Instantaneous Bandwidth delivered by GridFTP over TeraGrid.

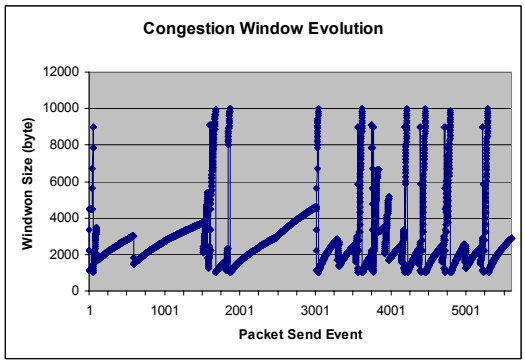

Figure 9. Congestion window size over time for a single TCP session used by GridFTP over TeraGrid.

5.3.2. TCP Send Buffer Sizes

Also due to the large bandwidth and latency product, GridFTP needs a large send buffers to deliver performance. GridFTP protocol supports automatic TCP buffer size negotiation, but this feature is not implemented in the current release. So users can manually set the TCP buffer size.

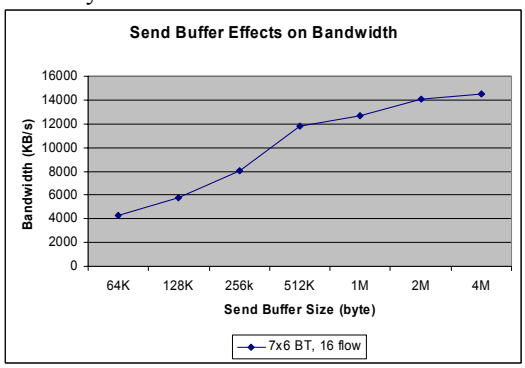

Figure 10. Send Buffer Effect on Bandwidth

Using MaSSF, we study how send buffer size influences GridFTP performance, increasing the TCP buffer from 64KB to 4MB and the result bandwidths are shown in Figure 10. We study the effect under different parallel thread numbers, which has the similar behaviors.

5.3.4. Link Loss Rates

Recall each flow is restricted to a single physical link. This works well for typical internet workloads of many small, independent flows. However, with programs like GridFTP which use collections of TCP sessions in concert, it is not clear how the performance will be affected if one link is heavy-loaded or becomes faulty. With MaSSF, we can study the effects of packet loss and observe how GridFTP responds.

We introduce the burst loss rate of 0.5% and 0.1% on 4x10Gbit/s links which comprise the Teragrid linear, and use a burst length of 5 packets. The achieved performance is shown in Figure 11. Monolithic assumes a single 40Gbps channel, the other two bars for each loss rate scenario correspond to individual trials with distinct mappings of flows to the faulty and non-faulty links.

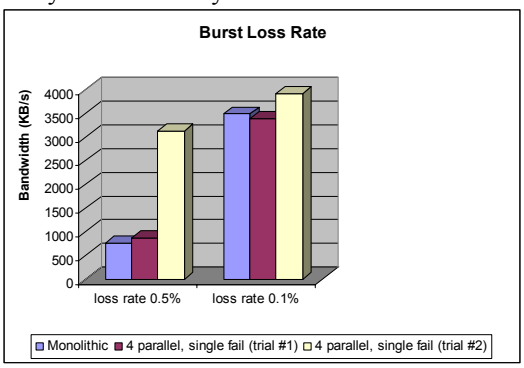

Figure 11. Burst loss effects on GridFTP performance.

We can see that bursty losses dramatically reduce performance. Even at 0.1% burst loss, best case performance is down nearly 75% from the no loss case. We attribute this to dramatic reductions in congestion windows at each loss. The monolithic case shows that performance is strongly dependent on link organization. Performance varies significantly on distinct trials, depending on how many of GridFTP's sessions wind up on the faulty link.

We also observed that differential loss rates will also effect the relative progress of the TCP sessions, increasing skew in completion time and stressing GridFTP's ability to balance across the parallel flows. This is an area for future improvement in GridFTP.

5.4. Evaluation

Our experiments show that GridFTP can achieve solid performance on Teragrid, but will peak far below the hardware limits on performance. Their parallel transfer feature is especially important to exploit the parallel L2 link pervasive in the TeraGrid backplane. However, the performance is fragile to any packet loss. Dynamic diagnosis of faulty links and reconfiguration to avoid their use will result in best performance.

6. Related Work

There have been many different emulation/simulation tools developed for network, distributed system, and grid research [cite many here]. However, few of these systems have exploited live execution of applications, coupling the modeling of computation, storage and networking. Network simulators such as ns [11] and GloMoSim [12] can simulate for large numbers of network nodes, but do not enable easy coupling to live applications (and thus cannot capture resource coupling and complex adaptive application behavior), or simplify the network modeling, sacrificing accuracy for scale.

Network emulation tools enable coupling with live application programs [11], but to date have mostly focused on static, and relatively small-scale systems. Some use only simple models [13] or have limited scalability [11]. One recent different approach, Modelnet [14], has an objective similar to ours -- providing a scalable Internet emulation environment that enable direct experimentation with application software. Their approach to scalability simplifies both network topology (a network of pipes) and routing (assuming a simple routing protocol based on shortest path. In contrast, MaSSF implements the full stack of OSPF and BGP4 routing protocol.

Emulab [15] represents an approach based on reconfiguration of routers to emulate in hardware the desired network behavior. Users can set up both the target network and operating system, in a single physical testbed. This approach has the advantage of speed of emulation, but provides little in the way of convenient control of speed and modeling to the experiment designer. However, they are currently exploring the addition of tools that can integrate other simulators and emulators to address these limitations.

MaSSF is built above the distributed discrete-event simulation engine DaSSF. While there have been many efforts on use PDES on network simulation [16], we know of no other modeling efforts that seek this level of detailed, general purpose emulation of computational Grid.

7. Conclusion

We have described both our MicroGrid and MaSSF modeling systems for computational grid and peer to peer applications. These systems enable repeatable modeling of dynamic resource, middleware, and application behavior in complex resource and network environments. We have applied these tools to a range of applications from traditional parallel to emergent peer to peer to grid data movement libraries. These studies have produced the following insights:

- deployment of peer-to-peer file sharing can increase file sharing performance by 3x, but at a high price in local LAN bandwidth,
- GridFTP performance on Teragrid network (backplane) is limited by TCP's ability to fill the high bandwidth delay product channels,

and delivers only 16MB/s of a possible 120MB/s, and

- GridFTP (and TCP) performance is extremely sensitive to bursty packet losses, and this variability indicates that router configuration is critical to achieving good performance.

Acknowledgements

Supported in part by the Defense Advanced Research Projects Administration through United States Air Force Rome Laboratory Contracts AFRL F30602-99-1-0534 and the National Science Foundation thru NSF EIA-99-75020 Grads and NSF Cooperative Agreement ANI-0225642 (OptIPuter) to the University of California, San Diego Support from Hewlett-Packard is gratefully acknowledged.

The authors also acknowledge the contributions of Alex Olugbile to the system infrastructure which made this work possible.

References

1. Ian Foster, C.K.e., *The Grid: Blueprint for a New Computing Infrastructure*. 1999: Morgan Kaufmann.
2. Francine Berman, A.C., Keith Cooper, Jack Dongarra, Ian Foster, Dennis Gannon, Lennart Johnsson, Ken Kennedy, Carl Kesselman, John Mellor-Crummey, Dan Reed, Linda Torczon, and Rich Wolski., *The GrADS Project: Software Support for High-Level Grid Application Development.* International Journal of High Performance Computing Applications,, 2001. **15**(4): p. 327-344.
3. Oram, A., *Peer-to-Peer: Harnessing the Power of Disruptive Technologies*. March 2001: O'Reilly.
4. H. Song, X.L., D. Jakobsen, R. Bhagwan, X. Zhang, K. Taura and A. Chien. *The MicroGrid: a Scientific Tool for Modeling Computational Grids*. in *IEEE Supercomputing (SC 2000)*. 2000. Dallas, USA.
5. James Cowie, H.L., Jason Liu, David Nicol and Andy Ogielski. *Towards Realistic Million-Node Internet Simulations*. in *Proceedings of the 1999 International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA'99)*. 1999. Las Vegas, Nevada.
6. V.Paxson, S.F.a., *Difficulties in simulationg the Internet.* IEEE/ACM Transactions on Networking, 2001. **9**(4).
7. William Saphir, R.V.d.W., Alex Woo, and Maurice Yarrow, *New implementation and results for the nas parallel benchmarks 2*. 1997, NASA Ames Research Center.
8. Minsky, M.L., *The Society of Mind*. March 1988: Simon & Schuster.
9. B. Allcock, J.B., J. Bresnahan, A. L. Chervenak, I. Foster, C. Kesselman, S. Meder, V. Nefedova, D. Quesnal, S. Tuecke, *Data Management and Transfer in High Performance Computational Grid Environments.* Parallel Computing Journal, 2002. **28**(5): p. 749-771.
10. W. Allcock, J.B., J. Bresnahan, A. Chervenak, L. Liming, S. Meder, S. Tuecke, *GridFTP Protocol Specification*, in *GGF GridFTP Working Group Document*. 2002.
11. Lee Breslau, D.E., Kevin Fall, Sally Floyd, John Heidemann, Ahmed Helmy, Polly Huang, Steven McCanne, Kannan Varadhan, Ya Xu, and Haobo Yu, *Advances in Network Simulation.* IEEE Computer, 2000. **33**(5): p. 59-67.
12. Lokesh Bajaj, M.T., Rajat Ahuja, Ken Tang, Rajive Bagrodia, and Mario Gerla, *GloMoSim: A Scalable Network Simulation Environment*. 1999, UCLA Computer Science Department Technical Report 990027.
13. Rizzo, L. *Dummynet and Forward Error Correction*. in *Proc. of the 1998 USENIX Anuual Technical Conf.* 1998. New Orleans, LA: USENIX Association.
14. Brian White, J.L., Leigh Stoller, Robert Ricci, Shashi Guruprasad, Mac Newbold, Mike Hibler, Chad Barb, and Abhijeet Joglekar. *An Integrated Experimental Environment for Distributed Systems and Networks*. 2002.
15. Amin Vahdat, K.Y., Kevin Walsh, Priya Mahadevan, Dejan Kostic, Jeff Chase, and David Becker. *Scalability and Accuracy in a Large-Scale Network Emulator*. 2002.
16. Rob Simmonds, R.B., and Brian Unger. *Applying parallel discrete event simulation to network emulation*. in *14th Workshop on Parallel and Distributed Simulation (PADS 2000)*. 2000. Bologna, Italy.