Evaluation of Rate-based Transport Protocols for Lambda-Grids

Xinran (Ryan) Wu and Andrew A. Chien
Department of Computer Science and Engineering
University of California, San Diego
{xwu, achien}@ucsd.edu

Abstract

Lambda-Grids are richly interconnected collections of plentiful, geographically-distributed computing and storage resources. This rich connectivity is enabled by dedicated dense wavelength division multiplexing (DWDM) optical paths. With abundant bandwidth in the center of the network (many DWDM links), contention and sharing bottlenecks move from the network core to end systems. In such networks, traditional TCP is insufficient to provide acceptable performance. We identify the key communication characteristics of this radically different network, introducing a new multipoint-to-point communication pattern for data-intensive application. We evaluate several promising rate-based data transport protocols (RBUDP, SABUL/UDT and GTP) for lambda-Grids under a range of communication patterns (single stream, multiple parallel streams, converging streams, and rapid transitions). Our experiments use a range of performance metrics, including sustained throughput and loss rate, inter- and intra-protocol fairness, protocol overhead, and rate adaptation speed to flow transitions. The results provide insights into the capabilities of these three protocols and also for improvements in design and implementation of rate-based protocols.

1. Introduction

Continuing advances in optical networking are producing increases in bandwidth which exceed the rapid geometric increases in semiconductor chip capacity as predicted by Moore's Law. Recently, *Dense Wavelength Division Multiplexing* (DWDM) has emerged as an efficient technique to exploit terabit fiber bandwidths, multiplexing large numbers of wavelengths (lambdas) onto a single fiber. Exploiting this trend, a wide range of research in systems and applications is being pursued to develop the *lambda-Grid* (sometimes called a *Distributed Virtual Computer* or DVC) [1, 2]. In the lambda-Grid, distributed grid [3] resources, including computing clusters, petabyte data repositories and high-resolution scientific instruments, can be tightly coupled by dedicated optical connections. In lambda-Grids, the central architectural element is optical networking instead of end systems. The OptIPuter project [4] and other efforts such as CANARIE [5] are exploring new opportunities and challenges from applications to system design [6-8] which arise from dedicated optical connections. The work described in this paper is part of the OptIPuter project.

Compared to traditional IP networks which have millions of endpoints, shared links, and are packet-switched, lambda-Grids are characterized by fewer endpoints (e.g. 10^3, not 10^8), dedicated high speed links (1Gbps, 10Gbps, etc.), and optical packet switching or circuit switching. These differences effectively mean that lambda-Grids have no internal network congestion. Since end-to-end dedicated link bandwidth matches or exceeds processing speeds in end systems, contention and sharing bottlenecks are pushed to the end systems. In grids generally, applications are rapidly evolving from a point-to-point model (e.g. data transfer from single server to a client) to a collection of clients and distributed servers of large data sets. Such applications exhibit communication patterns with multipoint-to-point (e.g. fetching large quantity of data from distinct servers to feed local computation or visualization) and multipoint-to-multipoint structure. In the lambda-Grid, these structures are combined with extremely high speed. Together, these differences imply a radically different set of communication challenges in lambda-Grids than in traditional IP networks [9].

Even for point-to-point communication in high bandwidth-delay product links, high performance bulk data transfer has been a long standing research challenge. Traditional TCP [10] was designed for shared low-bandwidth networks, and its performance is strongly dependent on the bandwidth-delay product of the network[11]. TCP's slow start and its *Additive Increase Multiplicative Decrease* (AIMD) congestion control balance non-aggressive competition and end-to-end performance. However, on high speed paths, slow start causes TCP to take a long time both to reach full bandwidth and to recover from packet loss when round trip time (RTT) is large. A number of TCP variants (e.g. [12-17]) have been developed to improve performance for shared, packet switched networks.

Recently, the Grid and high performance computing community has proposed a number of high performance

data transport protocols (e.g. [18-22]) based on UDP. These protocols are rate-based, enabling them to fill high bandwidth-delay product networks, using explicitly specified or negotiated transmission rates. These protocols also provide reliable transport services. We consider three representatives of these protocols, RBUDP, SABUL, and our GTP.

Each of these three protocols is different both in the intended environment of use and performance characteristics. Among them, *Reliable Blast UDP* (RBUDP) [19] targets fast, fixed-rate reliable data transfer on dedicated or QoS-enabled high speed links. It requires users to explicit configure (and reconfigure) the protocol based on link capacity. *Simple Available Bandwidth Utilization Library* (SABUL) [18] is designed for a shared network and conducts application level congestion and rate control over UDP. The newest version of SABUL, UDT[23], employs a delay-based rate adjustment scheme to improve the performance. Tsunami [21] targets at efficient file transfer over high speed links, the performance of which is limited by the I/O processing (and disk speed) of two ends. Recent work on the *Group Transport Protocol* (GTP) [22] focuses on the challenge of achieving high performance with a more complex multipoint-to-point communication pattern in lambda-Grids. GTP is a receiver–driven transport protocol which exploits information across multiple flows to manage receiver contention and fairness.

In this paper, we formulate the key communication problems for lambda-Grids, distilling them to four exemplar communication patterns (point-to-point single flow and parallel flows, multipoint-to-point converging flows, and rapid flow speed transitions). With these workloads, we study a range of rate-based protocols using Dummynet [24] emulation and measurements on the TeraGrid [25]. The primary contributions of this paper are summarized below.

- Definition of the key communication challenges for lambda-Grids captured in four model communication patterns: high-speed single and parallel flows, multipoint-to-point, multipoint-to-multipoint, and high speed transitions.
- Evaluation of three rate-based protocols (RBUDP, SABUL, GTP) for converging multipoint-to-point flows: all achieve high bandwidth, but vary widely (as much as 1000x) in packet loss rate with GTP achieving by far the lowest loss rate.
- Evaluation of three rate-based protocols on intra-protocol fairness which shows all exhibit good intra-protocol fairness for parallel flows. For converging flows only GTP maintains fairness that is independent on the difference of RTT's of flows.
- Evaluation of three rate-based protocols on inter-protocol fairness, which shows UDT is more TCP friendly than the other two, and
- Evaluation for workloads with rapid flow changes which shows that RBUDP and SABUL do not capture the available bandwidth efficiently. GTP manages rapid flow transitions better, efficiently exploiting the available bandwidth and maintaining low loss rates through a range of transitions.

Our results suggest that managing receiver contention for new multipoint-to-point communication pattern in lambda-Grids is a challenging problem. Our experiments show that receiver-based approach, of which GTP is an exemplar, is a promising direction and deserve further investigation for the new networking environment of lambda-Grids.

The remainder of the paper is organized as follows. In Section 2, we describe the communication problem and challenges in lambda-Grids. We provide an overview and comparison of the three rate-based protocols in Section 3. In Section 4, we present evaluation experiment results, followed by a summary of our results, and a discussion of future research directions.

2. Data Communication in Lambda-Grids

2.1 Modeling Lambda-Grid Communications

A lambda-Grid is a set of distributed resources directly connected with DWDM links (with 1-10Gbps per wavelength (lambda), and hundreds of lambdas per optical fiber). Lambda-grids are distinguished from traditional shared packet-switched IP networks by their dynamic configuration and dramatically higher performance and quality of service. The key distinguishing characteristics of lambda-Grid networks are:

(a) High speed (1Gbps, 10Gbps, etc.) dedicated links using one or multiple lambdas connecting a small numbers of endpoints (e.g. 10^3, not 10^8), and possibly with long delays (e.g. 60ms RTT from SDSC to NCSA) between sites. Switching and edge structure as described above. The dedicated lambdas have better QoS than shared internet (little jitter, low loss on fiber).

(b) End-to-end network bandwidth that matches or exceeds the data processing capabilities (computing/IO processing) of attached systems. The abundant network resources create a relative scarcity of end-system resources, pushing the congestion from internal network links to the endpoints.

(c) Network congestion occurs primarily at the end systems (e.g. data buffer overflow) or at the "last switch" where multiple high speed streams converge (e.g. two flows sharing the same receiver).

In a lambda-Grid, one can view the optical connections between end systems as fast, dedicated connections, in contrast to shared links which are packet-switched by IP routers. We illustrate this in Figure 1,

showing that internal network contention shifts from internal links to endpoints (or their access links, where multiple dedicated optical connections terminate).

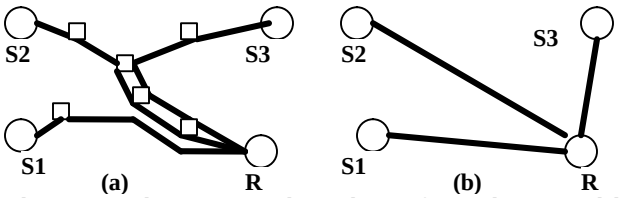

Figure 1: The connection view of receiver R with three senders. (a) Shared IP connection: senders connect with receiver via shared links and intermediate nodes. (b) Dedicated lambda connections: dedicated capacity between each sender/receiver pair.

2.2 Multipoint-to-point Communication Pattern

The advent of large-scale computation and data sharing in wide-area Grids and peer-to-peer applications is driving an evolution in communication patterns from point-to-point connections to multipoint-to-point and multipoint-to-multipoint structures. One example is P2P Content Delivery Networks (CDNs) such as Kazaa [26] and BitTorrent [27], where multiple replicated sites and accessed simultaneously to retrieve data as fast as possible.

The change in communication structure is even more rapid in lambda-Grids where high-speed dedicated wavelength connections are used to access large distributed data collections (which may be 100's of petabytes). In this architecture, applications fetch data from multiple sites concurrently and operate on that data locally. Novel to lambda-Grids, there is plentiful network bandwidth in the network core, so when multiple dedicated lambda connections converge, their aggregate capacity far exceeds the data handling speed of the end system. In short, the critical contention occurs at endpoints, not within the network.

2.3 Communication Challenges in Lambda-Grids

Data transport protocols for lambda-Grids are subject to a range of performance considerations for design and implementation. More complex communication patterns such as multipoint-to-point and multipoint-to-multipoint share traditional challenges of point-to-point high bandwidth-delay product transmission such as achieving high aggregate throughput while keeping loss rate low, but introduce several new challenges.

High Throughput Low-Loss Transmission for Parallel Flows With many projects utilizing multiple flows (e.g. parallel flows between two ends, and multipoint-to-point), the communication solution to lambda-Grids should be aggressive enough to employ all of the receiver's communication capacity with multiple connections, achieving high throughput and still maintaining low average loss rate.

Intra and Inter Protocol Fairness Among Flows An important design goal for multi-flow communication is to provide predictable performance to flows. This requires the rate allocation (or bandwidth sharing) of multiple flows to meet certain fairness criteria, such as Max-min fairness [28], proportional fairness[29], etc. Intra-protocol fairness assures all flows following the same protocol receive the same level of the service. Inter-protocol fairness addresses fair competition among traffic flows from multiple protocols, including the interaction between rate-based protocols with TCP (the notion of "TCP friendliness"[30]).

Quick Response to Flow Dynamics An ideal solution would react quickly to flows joining and departing, efficiently utilizing the available network capacity and maintaining low loss rates. Smooth and efficient transitions would approach the maximum feasible network performance.

3. Protocols

In this section we give a brief overview of three rate-based protocols: RBUDP[19], SABUL/UDT[23] and GTP[22]. A summary of the key characteristics of these protocols can be found in Table 1.

3.1 RBUDP

RBUDP [19] is a point-to-point data transfer protocol, intended for dedicated (e.g. dedicated wavelength) or Quality-of-Service (QoS) enabled network environment. The RBUDP sender starts by transmitting all data blocks over UDP at a fixed speed, which is specified by the user (or NIC speed). The receiver maintains a bitmap to keep track of received/lost data blocks. After the sender finishes sending all the data, the receiver sends the updated bitmap back to the sender through TCP (TCP is used for the purpose of reliable transmission). The sender then resends the lost data blocks according to the bitmap in the next round. The above procedure repeats until receiver successfully receives all the data blocks. RBUDP does not perform any rate adaptation, so in order to avoid network congestion, it requires explicit control by the protocol user (for example running Iperf [31]). Thus, dynamic rate scenarios are beyond the scope of the protocol (must include external control). Protocol overhead includes the delay between each round of transmission due to the bitmap transmission, which becomes expensive when the number of rounds of the transmission increases (due to heavy network congestion). We expect this problem to be solved in their upcoming streaming version of RBUDP.

	RBUDP	SABUL/UDT	GTP
Initial Rate	Specified by the User or NIC speed	Slow start, exponential increment	Negotiated by the sender and receiver
Reliable Transmission	Yes	Yes	Yes
Multipoint-to-point	No.	No.	Yes.
Rate Adaptation	No	Rate-based with delay compensation	Rate adaptation and estimation
Intra-protocol Fairness	Not Considered	To some extent	Max-min fairness among flows at the receiver side
TCP Friendliness	No	Yes	No. (could be extended to manage TCP flows together with GTP flows [22])
Transition Management	No	Rate adaptation according to its AIMD law	Explicit transition management to flow changes, and the ability of quick exploring available bandwidth.
Implementation	User level	User level	User level

Table 1: Summary Comparison of Three Rate-based Protocols

3.2 SABUL/UDT

SABUL [23] (Simple available bandwidth utilization library [18]) is designed for data-intensive applications in high bandwidth-delay product networks with user level implementation and control. The newest version of SABUL, UDT [23], combines rate-based, window-based and delay-based control mechanisms to deliver high throughput and low loss data transmission. UDT implements slow start and AIMD control scheme for flow control (which makes it to be more TCP friendly than other rate-based protocols) and window-based control for controlling the number of outstanding packets in flight. UDT also deploys rate adjustment based on delay monitoring, providing improved performance over common AIMD control laws. However this also makes UDT sensitive to network and end system conditions. Therefore UDT maximizes its performance on dedicated connections. Since the control scheme in UDT is combined from several different control mechanisms, it would be interesting to provide thorough theoretical analysis of its congestion control mechanisms to further illustrate its behavior.

3.3 GTP

GTP [22] is a receiver–driven request-response transport protocol designed for efficient multipoint-to-point data transmission. GTP implements two levels of flow control. For each individual flow, the receiver explicitly controls the sender's transmission rate. This allows the flow's rate to be adjusted quickly in response to packet loss (detected at the receiver side). Across the incoming flows at each receiver, there is a scheduler. This structure exploits the insight that in lambda-Grids, congestion usually occurs at the end systems, especially the receivers. The scheduler at the receiver manages across multiple flows, dealing with any congestion or contention and performing max-min rate amongst them. The receiver actively measures per-flow throughput, loss rate, and uses it to estimate bandwidth capacity. It then allocates the available receiver capacity (can be limited by resource or the final link) across flows. This allocation is done once for each control interval in Max-min fair manner. Correspondingly, the senders adjust to transmit at the revised rates. This receiver-driven centralized rate allocation scheme enables GTP to significantly reduce receiver side packet loss and respond quickly to transitions (flows join or terminate).

4. Protocol Evaluation

4.1 Methodology

We compare RBUDP, SABUL, GTP, and standard untuned TCP (PSockets[16] is used to generate parallel TCP streams). Throughout our experiments we use the latest available versions of the protocols (RBUDP v0.2, SABUL/UDT 1.1, and GTP prototype) and use emulations with Dummynet [24] delay router, and measurements on TeraGrid [25] with two end points at SDSC (San Diego Supercomputer Center) and NCSA (National Center for Supercomputing Applications) to model a broad range of network structure. The end-to-end bandwidth between SDSC and NCSA of each connection is 1Gbps (NIC speed limit). Dummynet is used to introduce a range of various round trip delays to the experiments performed in local cluster environment (see Figure 2).

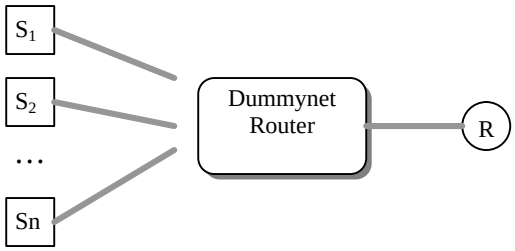

Figure 2: Experiment environment on a local cluster (dual 2.2GHz Intel Xeon processors, 2 Gigabit NICs and 2GB memory). A Dummynet router is installed to introduce various round trip delays.

We consider several communication patterns for our experiments. These are chosen to model our point-to-point and multi-point-to-point applications' expected behaviors. They include:

- Point-to-point with single flow (Figure 3a);
- Point-to-point parallel flows (Figure 3b);
- Converging flows (multiple senders and single receiver) with varied delay and bandwidth for each link (Figure 3c);
- Flow dynamics where new flows join or existing flows terminate.

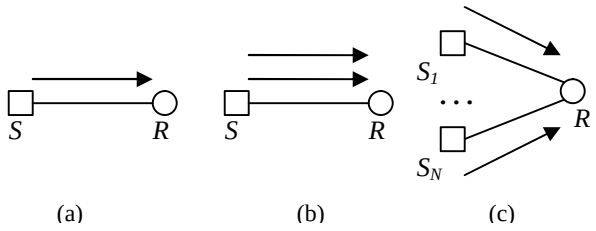

Figure 3: Three data transmission patterns. (a) point-to-point, single flow; (b) point-to-point, parallel flows; (c) Multipoint-to-point, converging flows.

Considering the requirements of high-performance data-centric e-science applications which are the focus of much of the work in grids and lambda-Grids, we have defined a representative set of performance metrics which we use. These metrics include:

- Sustained throughput and loss ratio for a 10GB data transfer (Point-to-point and multipoint-to-point);
- Intra-protocol fairness (the ratio of minimum to maximum flow throughput);
- Inter-protocol fairness, and their interaction with TCP (the ratio of TCP throughput with and without without rate-based protocol);
- Loss ratio in the first 50 RTT after flow arrival and throughput in first 50 RTT after flow departure.

These metrics measure throughput, fairness in several forms, and dynamic response of the protocols. We present the results of our evaluation in the following subsections.

4.2 Throughput and Loss Measurements

To provide a performance baseline, we present an evaluation of the performance of rate-based protocols with two general metrics, sustained throughput and loss ratio. In particular, we summarize for the three data transmission patterns (see Figure 3) on the TeraGrid and present the results in Table 2. For all three scenarios we measure the sustained throughput of transferring 10GB data between SDSC and NCSA. The bandwidth on each single point-to-point link is 1Gbps, and the round trip delay is approximately 58ms.

		TCP	RBUDP	UDT	GTP
(a) Single flow	Average Throughput (Mbps)	4.88	881	898	896
	Avg. Loss	unknown[1]	0.07%	0.01%	0.02%
(b) Parallel flows	Aggregate Rate (Mbps)	14.5	931	912	904
	Avg. Loss	unknown	2.1%	0.1%	0.03%
(c) Convergent flows	Aggregate Rate[2] (Mbps)	677	443	811	865
	Avg. Loss	unknown	53.3%	8.7%	0.06%

Table 2: Throughput and loss measurements made on the TeraGrid. (a) Single flow between NCSA and SDSC. (b) Parallel flows between the same sender and receiver from SDSC to NCSA. (c) Converging flow with three senders (two at NCSA and one at SDSC) to one receiver at SDSC.

Our results show that for single flows, the three rate-based protocols achieve much higher throughput than traditional TCP while maintaining a low-loss ratio. All three rate-based protocols also perform well when there are parallel flows between the same sender and receiver. While RBUDP and UDT achieve slightly higher throughput than GTP with their aggressiveness, it incurs the expense of a much higher loss ratio. GTP's receiver-based control scheme provides high throughput and a low loss ratio. For converging flows, all three rate-based protocols achieve high throughput, but loss rates vary over a range of 1000x, and GTP has the lowest loss rate by a large margin.

[1] We are not able to measure instant TCP loss rate, due to the lack of root privileges on TeraGrid.

[2] Aggregate rate and loss rate vary for RBUDP and SABUL, and numbers listed are the average values of several measurements.

Note that throughout all the experiments (in this and following subsections) RBUDP is configured with a capacity estimate for each flow of 1Gbps (their network interface speed), which is reasonable and close to the achievable bandwidth measured by Iperf [31] for the case of single connection on a dedicated link. While it may be argued that other fixed settings would produce lower packet loss rate, RBUDP provide no assistance in choosing such settings. This is because unlike mode transport protocols, RBUDP includes no mechanisms for capacity estimatation or rate adaptation.

4.3 Intra-protocol Fairness

Providing stable and predictable data transmission service requires high speed protocols share resources stably amongst competing flows. We first consider sharing amongst flows based on the same rate-based protocol, and use equal allocation (fairness) of the bandwidth on the link as the metric. We use the following definition for fairness: Given a set of N flow with rate allocation $R = \{r_1, r_2, \ldots, r_N\}$, where r_{max} and r_{min} are the maximum and minimum rates of those flows, we define the fairness index f_R of flow rate allocation R as

$$f_R = r_{min} / r_{max.}$$

All of the protocols achieve good fairness for a single link with 4 and 8 parallel flows (see Figure 4).

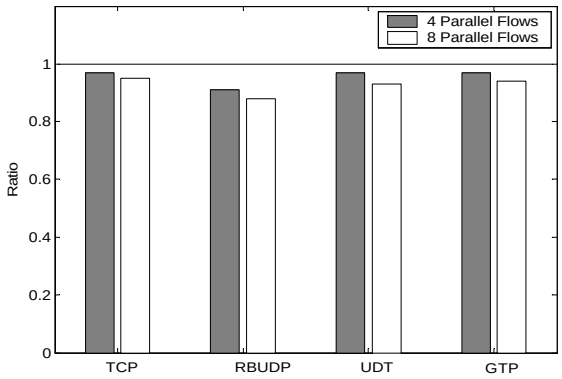

Figure 4: Fairness index of 4 and 8 parallel flows on a single link.

It is a longstanding research challenge (e.g. [32], [33]) to get converging flows using the same protocol, but with different RTT's to achieve a fair rate allocation. Consider a scenario with two converging flows as shown in Figure 3c. We fix one flow's RTT at 50ms, and vary the other flow's RTT from 5ms to 50ms using Dummynet. Figure 5 plots the achieved fairness index for each of the rate-based protocols.

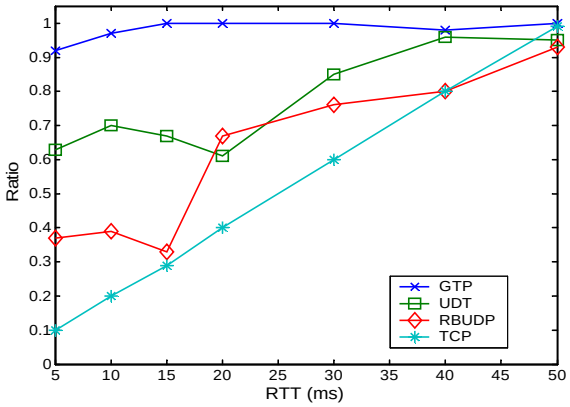

Figure 5: Fairness index of GTP, UDT, RBUDP, TCP with two converging flows, where flow 1 has a fixed RTT=50ms and RTT of flow 2 varies from 5ms to 50ms.

TCP is well-known to deliver throughput inversely proportional to the RTT [34, 35], so the fairness index of TCP increases with RTT. For UDT and RBUDP, a difference in RTT has a similar, but less pronounced effect, reducing the achieved fairness index. For UDT this is because a flow with shorter RTT adapts faster and thus increases its rate more quickly. For RBUDP, both flows will have the same sending rate, but the one with smaller RTT transfers retransmission bitmaps back more quickly, reducing the protocol overhead. In contrast, GTP maintains a better fairness index (close to 1), across a range of RTT differences. This is because GTP explicitly allocates receiver bandwidth to flows at the receiver, enabling each flow to achieve a fair share of the throughput.

4.4 Inter-Protocol Fairness

Because applications and networks may employ many protocols, we consider inter-protocol fairness of these flows (and TCP) from two perspectives. First, we study how the three rate-based protocols interact with each other. Second, we study the interaction of each of these protocols with TCP. All of these experiments are performed using Dummynet.

Scenario 1: Rate-based protocols on a single link. We start single RBUDP, UDT and GTP flows staggered in time as parallel flows on a single link (as in Figure 3b). First, UDT and GTP share the bandwidth efficiently, dividing it equally (see Figure 4). When RBUDP is introduced, it captures less than an equal share while UDT and GTP continue to share the remaining bandwidth equally. All three rate-based protocols are thus able to co-exist as parallel flows. This also shows that UDT and GTP have more aggressive implementations than RBUDP.

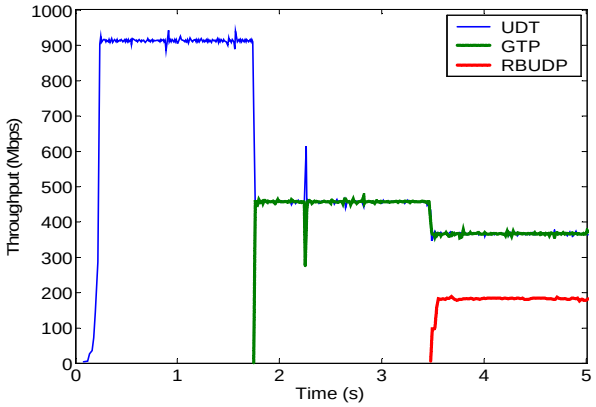

Figure 6: Three flows sharing the same sender and receiver. GTP and RBUDP start at time 1.8s and 3.5s, respectively.

Scenario 2: Rate-based protocols with converging flows. We initiate the three flows staggered in time and originating from distinct senders to a single receiver (the scenario in Figure 3c). Our results show that in this case, RBUDP takes bandwidth with GTP and UDT being largely shut out (see Figure 7). This result can be explained by how the congestion at the end system (receiver) is resolved. Both GTP and UDT detect the congestion and reduce their flow rates in response to loss, while RBUDP continues to transmit at high rates, eventually driving GTP and UDT traffic to close to zero.

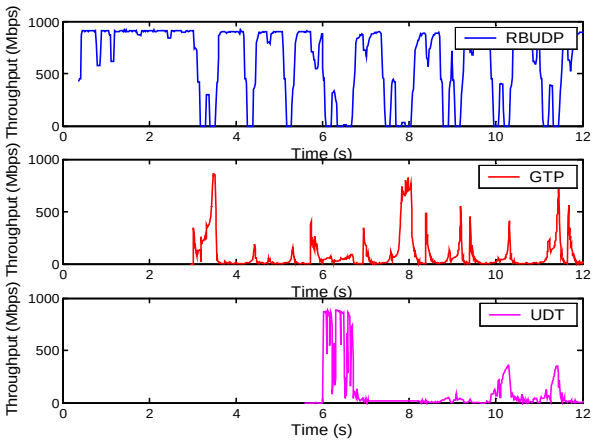

Figure 7: Three contending flows from three senders terminating at one receiver. GTP and UDT start at time 3s and 6s, respectively.

This also illustrates that to be fair to other flows, RBUDP needs to be modified to adapt its rate. Again, such rate adaptation is intentionally not addressed in the design of RBUDP, as it is intended for other environments.

Scenario 3: Rate-based protocols and TCP. We study how the operation of these rate-based protocols affects traditional TCP flows. Since web services and a widely-used grid data transfer tool GridFTP [36] are based on single or multiple TCP flows, our study also provides some insight to how these protocols will interact. We measure TCP throughput in the presence of each of the rate-based protocols and compare to TCP running alone. An ideal ratio is 50% (equal sharing), with lower ratios indicating that the TCP traffic is suffering.

Experiments with parallel TCP and rate-based protocol flows show good sharing properties (see Table 3) in local cluster environment. On high bandwidth-delay product networks, the situation is different (see Table 4). In the presence of RBUDP and GTP, TCP is not able to achieve the same level of the throughput. This is because RBUDP and GTP are aggressive. TCP is only able to obtain an equal share of the network capacity with UDT because it employs a similar increase/decrease flow control mechanism as in TCP.

	Rate-based and TCP		Single TCP Throughput	Influence Ratio
	Rate-based	TCP		
RBUDP	467Mbps	450Mbps	912Mbps	49.3%
UDT	552Mbps	380Mbps	912Mbps	41.6%
GTP	612Mbps	328Mbps	912Mbps	35.9%

Table 3: RBUDP, UDT, GTP each runs with a single TCP flow, point-to-point on a 1Gbps link on the cluster.

	Rate-based and TCP		Single TCP Throughput	Influence Ratio
	Rate-based	TCP		
RBUDP	771Mbps	2.1Mbps	24.3 Mbps	8.6%
UDT	751Mbps	23.6Mbps	24.3Mbps	97.2%
GTP	760Mbps	9.7Mbps	24.3Mbps	40.0%

Table 4: RBUDP, UDT, GTP each runs with a single TCP flow, point-to-point on a simulated 800Mbps Dummynet link with 30ms RTT.

4.5 Transition Management

The ability to respond quickly and stably to rapid flow transitions (begin or end) is an important capability for transport protocols in high speed networks whose goal is to provide the maximum physical bandwidth to large flows. However, achieving maximum throughput, stable behavior, and rapid transitions is challenging. For example, how to respond to the beginning and end of a multi-gigabit flow when the network is operating at 100% capacity? We use a three-stage scenario to evaluate the three rate-based protocols. We begin with a single flow (flow 1), and a second flow (flow 2) with 40ms RTT begins around 5 seconds later. Flow 2 ends about 5 seconds after beginning. The trajectories for each flow's

throughput are depicted in Figures 10-12. The network link speed to the receiver is 800Mbps in the experiments.

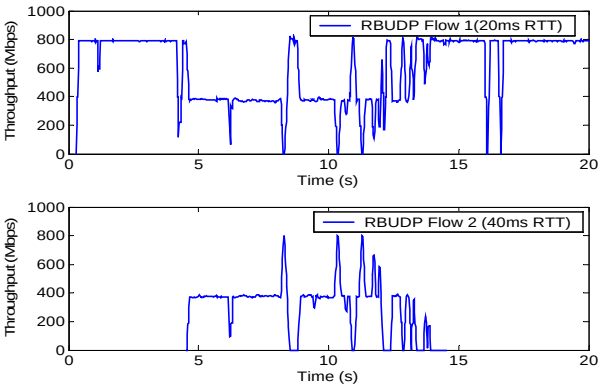

Figure 10: Two RBUDP flows share the same receiver but from different senders. Flow 2 joins at 5 seconds, and terminates more than 5 seconds later.

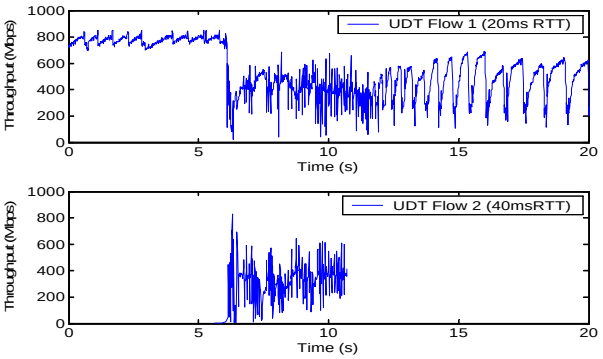

Figure 11: Two UDT flows share the same receiver but from different senders. Flow 2 begins at about 5 seconds and ends 6 seconds later..

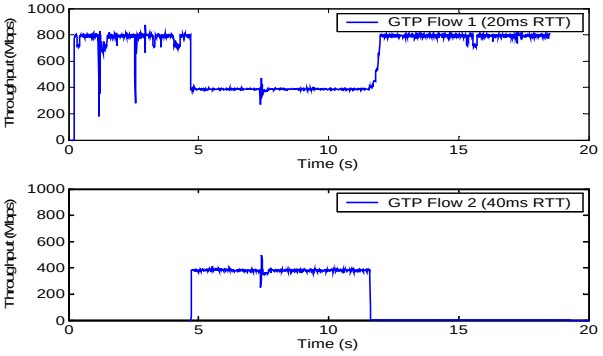

Figure 12: Two GTP flows share the same receiver but from different senders. Flow 2 begins at around t=5, and terminates about 7 seconds later.

Of the three protocols, GTP achieves smooth transitions, but RBUDP and UDT both give erratic behavior. RBUDP gives rapid transition at a flow beginning, but incurs high (30%) packet loss due to its lack or rate control. UDT's sensitivity to packet loss and delay produces rate oscillations in our Dummynet environment and slower recovery speed after flow 2 terminates. Because Dummynet is an emulation tool, there is the possibility that its behavior is not true to real networks. For example, better transition and fairness results are reported in [37], where the link rate used for real measurement is low (100Mbps). GTP performance is the best amongst the three, producing clean transition and quick rate adaptation to flow changes. GTP has a fundamental advantage in its centralized receiver-based rate allocation scheme, providing a global perspective across flows, and thereby enabling dramatically better network performance.

To quantify the response to flow transitions, we define two additional performance metrics. First, we use the loss ratio in the first 1 second (50 RTTs) after flow 2 begins to characterize each protocol's response to new flows. Second, we calculate the throughput of flow 1 during the first 1 second (50 RTTs) after flow 2 ends, and its long-term sustained throughput without flow 2, and utilize the ratio of these two throughputs to characterize the protocol's ability to return bandwidth to flow 1. These performance metrics are shown in Table 6.

		RBUDP	UDT	GTP
M1	Loss Ratio	29.3%	17.7%	0.7%
M2	Throughput after flow 2 leaves	740Mbps	359Mbps	687Mbps
	Sustained Throughput	793Mbps	787Mbps	773Mbps
	Ratio	0.93	0.45	0.88

Table 6: RBUBP, UDT, GTP each runs with a single TCP flow, point-to-point on a simulated 800Mbps Dummynet link with 30ms RTT.

GTP achieves good results for both metrics. RBUDP does not adjust its transmission rate, so huge losses are incurred when flow 2 begins, and flow 1 fills the bandwidth fast when flow 2 ends. We see oscillations of UDT along with the introduction of flow 2 in our Dummynet environment.

5 Summary and Future Work

Lambda-Grids involve a new set of communication challenges where networks have plentiful bandwidth but limited end-system capacity. This change moves congestion from the internals of the network to the endpoints and makes new communication patterns such as multipoint-to-point and multipoint-to-multipoint important. We study the performance of three promising

rate-based protocols (RBUDP, UDT, and GTP) in a wide range of different circumstances. Our results show that all three rate-based protocols can achieve high performance for point-to-point connection with single or parallel flows. However, when converging flows are considered, the situation is quite different. With its receiver-driven architecture and rate allocation across flows, GTP outperforms RBUDP and UDT providing lower loss, higher throughput, lower CPU overhead, and rapid, stable transitions as flows begin and end. These results suggest that receiver-driven architectures should be more broadly studied for lambda-Grid transport protocols.

Future Work Major challenges remain for rate-based protocols in lambda-Grid environment. First, we need to explore more techniques for end system contention management which is critical when networks are fast enough to move congestion to the end systems. Second, implementation techniques which are efficient and incur low CPU overhead are needed to make these new transport protocols usable. Third, a wide range of challenges remain in achieving fast and clean transition in response to network changes, and inter-protocol and intra-protocol fairness (especially TCP friendliness). Fourth, we will explore integrating TCP traffic estimation and management into the frame work of GTP, reserving a bandwidth share for TCP, so as to provide a satisfactory level of quality of service. Finally, it is a critical and challenging research topic to model these rate-based protocols analytically, including a formal proof of their properties (e.g. convergence, fairness, TCP friendliness, etc.).

Acknowledgements

This work is supported in part by the National Science Foundation under awards NSF EIA-99-75020 Grads and NSF Cooperative Agreement ANI-0225642 (OptIPuter), NSF CCR-0331645 (VGrADS), NSF NGS-0305390, and NSF Research Infrastructure Grant EIA-0303622. Support from Hewlett-Packard, BigBangwidth, Microsoft, and Intel is also gratefully acknowledged.

6. References

[1] N. Taesombut and A.A. Chien, *Distributed Virtual Computer (DVC): Simplifying the Development of High Performance Grid Applications.* In Proceedings of the Workshop on Grids and Advanced Networks (GAN 04), Aprial 2004.

[2] T. DeFanti, C.d. Laat, J. Mambretti, K. Neggers and B.S. Arnaud, *TransLight: a Global-scale LambdaGrid for e-Science.* Communications of the Association for Computing Machinery (CACM), 47(11), November 2003.

[3] I. Foster and C. Kesselman, *The Grid: Blueprint for a New Computing Infrastructure.* Morgan Kaufmann, 1999.

[4] L. Smarr, A. Chien, T. DeFanti, J. Leigh and P. Papadopoulos, *The OptIPuter.* Communications of the Association for Computing Machinery 47(11), November 2003.

[5] *CANARIE.* http://www.canarie.ca.

[6] H.B. Newman, M.H. Ellisman and J.A. Orcutt, *Data-intensive e-Science Frontier Research.* Communications of the Association for Computing Machinery (CACM), 47(11), November 2003.

[7] I. Foster and R.L. Grossman, *Data Integration in a Bandwidth-rich World.* Communications of the Association for Computing Machinery (CACM), 47(11), November 2003.

[8] T.A. DeFanti, J. Leigh, M.D. Brown, D.J. Sandin, O. Yu, C. Zhang, R. Singh, E. He, Alimohideen, N.K. Krishnaprasad, R. Grossman, M. Mazzucco, L. Smarr, M. Ellisman, P. Papadopoulos, A. Chien and J. Orcutt, *Teleimmersion and Visualization with the OptIPuter.* Proceedings of the 12th International Conference on Artificial Reality and Telexistence (ICAT 2002), The University of Tokyo, Japan, December 3-6, 2002.

[9] A. Falk, T. Faber, J. Bannister, A. Chien, R. Grossman and J. Leigh, *Transport Protocols for High Performance.* Communications of the Association for Computing Machinery (CACM), 47(11), November 2003.

[10] J.B. Poster, *Transmission Control Protocol.* RFC 793, Sep 1981.

[11] Braden Jacobson, *TCP Extensions for High Performance.* RFC 1323, May 1992.

[12] S. Floyd, *HighSpeed TCP for Large Congestion Windows.* Internet draft.

[13] C. Jin, D.X. Wei and S.H. Low, *FAST TCP: Motivation, Architecture, Algorithms, and Performance.* in Proceedings of IEEE INFOCOM 2004, Hong Kong, March 2004.

[14] D. Katabi, M. Handley and C. Rohrs, *Internet Congestion Control for High Bandwidth Delay Product Network.* in Proceedings of ACM SIGCOMM 2002, Pittsburgh, Aug 2002.

[15] T. Kelly, *Scalable TCP: Improving Performance in Highspeed Wide Area Networks.* Submitted for publication, December 2002.

[16] H. Sivakumar, S. Bailey and R.L. Grossman, *PSockets: The Case for Application-level Network Striping for Data Intensive Applications using High Speed Wide Area Networks.* In Proceedings of Supercomputing 2000.

[17] S.H. Low, L. Peterson and L. Wang, *Understanding Vegas: A Duality Model.* Journal of ACM, 49(2):207-235, March 2002.

[18] H. Sivakumar, R. Grossman, M. Mazzucco, Y. Pan and Q. Zhang, *Simple Available Bandwidth Utilization Library for High-Speed Wide Area Networks.* Submitted for publication.

[19] E. He, J. Leigh, O. Yu and T. DeFanti, *Reliable Blast UDP: Predictable High Performance Bulk Data Transfer.* IEEE Cluster Computing, 2002: p. 317.

[20] P. Dickens, *FOBS: A Lightweight Communication Protocol for Grid Computing.* in proceedings of Euro-Par 2003.

[21] *Tsunami.* http://www.indiana.edu/~anml/anmlresearch.html.

[22] R.X. Wu and A.A. Chien, *GTP: Group Transport Protocol for Lambda-Grids.* In Proceedings of the 4th IEEE/ACM International Symposium on Cluster Computing and the Grid, Aprial 2004.

[23]Y. Gu, X. Hong, M. Mazzucco and R.L. Grossman, *SABUL: A High Performance Data Transfer Protocol*. Submitted for publication.
[24] L. Rizzo, *Dummynet: a Simple Approach to the Evaluation of Network Protocols.* Computer Communication Review, 27, January 1997.
[25] D.A. Reed, *Grids, the TeraGrid, and Beyond.* IEEE Computer, 36(1) 62-68, 2003.
[26] *Kazaa.* http://www.kazaa.com.
[27] *BitTorrent.* http://bitconjurer.org/BitTorrent/.
[28] D.P. Bertsekas and R. Gallager, *Data Networks, Second Edition.* Prentice-Hall, Englewood-Cliffs, New Jersey, 1992.
[29] F. Kelly, A. Maulloo and D. Tan, *Rate Control for Communication Networks: Shadow Prices, Proportional Fairness and Stability.* Journal of the Operational Research Society, 49, pp. 237-252.
[30] J. Mahdavi and S. Floyd, *TCP-Friendly Unicast Rate-Based Flow Control.* Technical note sent to the end2end-interest mailing list, Jan 1997. http://www.psc.edu/networking/papers/tcpfriendly.html.
[31] *Iperf Tool.* http://dast.nlanr.net/Projects/Iperf/.
[32] T. Henderson, E. Sahouria, S. McCanne and R. Katz, *On Improving the Fairness of TCP Congestion Avoidance.* In Proc. IEEE Globecom '98, volume 1, pp. 539--44, Sydney, Australia, November 1998.
[33] J. Mo and J. Walrand, *Fair end-to-end window-based congestion control.* IEEE/ACM Transactions on Networking 8, 556--567.
[34] S. Floyd, *Connections with Multiple Congested Gateways in PacketSwitched Networks Part 1: One-way traffic.* Computer Communication Review, Vol. 21, No. 5, pp. 30-47, October 1991.
[35] J. Padhye, V. Firoiu, D. Towsley and J. Kurose, *Modelling TCP Throughput: A Simple Model and its Empirical Validation.* EEE/ACM Trans. on Networking, vol. 8, No. 2. Apr. 2000.
[36] B. Allcock, J. Bester, J. Bresnahan, A.L. Chervenak, I. Foster, C. Kesselman, S. Meder, V. Nefedova, D. Quesnel and S. Tuecke, *Data Management and Transfer in High-performance Computational Grid Environments.* Journal of Parallel Computing, 28(5) 749-771, May 2002.
[37] Y. Gu and R.L. Grossman, *End-to-End Congestion Control for High Performance Data Transfer*. Submitted for publication.