A Comparison of Reactive Host-based Flow Control Schemes in Reservationless Networks

Hemant Kanakia
Computing Science Research Center
AT&T Bell Laboratories
Murray Hill
kanakiaresearch.att.com

S. Keshav
Computing Science Research Center
AT&T Bell Laboratories
Murray Hill
keshavresearch.att.com

Partho P. Mishra
Systems Design and Analysis Group
Department of Computer Science
University of Maryland
parthocs.umd.edu

Abstract

A nationwide gigabit network with a high bandwidth delay product poses enormous challenges as well as opportunities in designing flow and congestion control schemes. In this paper we have focused on a class of flow control schemes used for bursty data communication traffic where bandwidth demands are not known apriori. We define reactive host-based congestion control schemes as schemes where network endpoints adjust their sending rates in response to explicit or implicit signals of network congestion. Although a number of such congestion control schemes has been proposed and the number keep growing every year, we lack a commonly accepted set of objective tests to be used as a benchmark to compare the proposed schemes. A benchmark suite would be valuable in judging the progress made in this field. The focus of this paper is in creating just such a benchmark suite. To remain focused on the effort, we have not considered congestion control mechanisms which assume that traffic demands are known and use reservation of resources to prevent congestion. However, we do note that the flow control schemes compared in this paper could also work in a framework that includes explicit resource reservation and usage monitoring.

For simplicity, we argue for using two user-oriented performance measures, namely the file transfer time of a fixed length file and the delay suffered by a single packet, to compare the performance of schemes. By carefully designing the topology of the network and the experimental environment, one can use these two measures to subsume other traditional performance measures such as link utilization, fairness and throughput. Our approach has the further advantage that using only two measures allows us to easily quantify the relative performance of the various flow/congestion control schemes. In addition to describing the test scenarios that are part of the benchmark suite, the paper also presents simulation results that compare several representative reactive congestion control schemes. Based on the results, we feel confident that the suite is effective and objective in differentiating among different schemes.

1 Introduction

Gigabit networks have a high bandwidth-delay product that makes the problem of congestion control a challenging research area. In future, congestion control mechanisms will probably involve several interrelated components such as resource reservation, Quality-of-Service based resource scheduling, and predictive and reactive flow control at network endpoints. In this paper, we consider congestion control schemes, implemented in the network and transport layer protocols, which try to optimize network and user performance under the restriction that the network does not specifically reserve resources for individual connections.

In recent years several new host-based reactive flow control schemes have been proposed, but we have lacked a set of objective tests for comparing their relative performance. This task would be easier if everyone agreed on the measurement environment and the performance measures to use. We focus here on convincing readers that the task of creating a credible, objective and useful benchmark is feasible and useful. To that end, we have designed a a framework that is not biased towards a particular network, set of applications or a control scheme.

We do not claim that the benchmark is suitable for declaring any one scheme as the best overall flow control scheme. In practice, the merits of a scheme must be evaluated not only on the basis of simulation benchmarks, but also on its implementation cost, the domain of use, and resource requirements. Our suite is not intended to compare these important, but orthogonal, issues. Our aim is simply to enable interested researchers to judge progress and to fine-tune the performance of a newly proposed flow control scheme. Past experience in designing and evaluating new flow and congestion control schemes leads us to believe that this is a useful exercise.

A quantitative comparison among host based reactive flow control schemes (we will simply call them congestion control schemes in the sequel) is meaningful because all such schemes have the same goal, namely, to prevent any part of a network from getting congested due to traffic overloads, while simultaneously maintaining a reasonable level of network utilization. At a high level of abstraction all congestion control schemes operate similarly. Network traffic is monitored either implicitly or explicitly to collect statistics that help presage the onset of congestion. If the traffic is monitored at a switch, rather than at a traffic source, the collected statistics have to be propagated to traffic sources. Traffic sources use the information to adjust the rate at which they inject traffic into the network so that overloads do not occur; at the same time making sure that network resources are not under-utilized. Traffic sources always have delayed statistical information about traffic conditions in the network. The method of collecting statistics and propagating statistics as well as the strategy used to throttle or to increase input traffic based on the traffic statistics is what differentiates the performance of various flow control schemes.

The perspective of a network user is important in comparing the performance of congestion control schemes. Network congestion can be defined as a degradation in network performance due to overload, and this degradation is important only to the extent that it affects users. A congestion control scheme impacts the performance in two ways. If congestion occurs, then packets are lost at the bottleneck and there are increases in the queueing delay. This reduces the throughput seen by users and also increase the response time. If an overly-conservative approach is used to prevent congestion at any cost, network resources will remain underutilized and a user will see reduced throughput. The task of congestion control schemes is to strike a balance between these two extremes. A user is also concerned about the long-term stability of the network. A prolonged denial of service while waiting for network congestion to clear is as acceptable as being put on hold on a telephone call. Furthermore, users are always sensitive to the quality of their service suffering due to the demands made by "greedy" and socially irresponsible users. Thus, we would like to measure the relative performance of various congestion control schemes to the extent that they affect the performance seen by a user.

2 Preliminary Considerations

Previous studies of congestion control schemes have used infinite data streams as traffic sources. Infinite data sources are used to ensure that measurements are not made in transient network conditions and to study the dynamics of the flow control schemes. However, they suffer from the problem that in reality, no file transfer is infinite. Indeed, as we move towards higher speed networks, the duration of transfer of a file becomes shorter and shorter. Thus, it is these very transients that will determine the perceived efficiency of a flow control scheme.

We propose to use only sources with finite amounts of data to send and use the total file transfer time as the primary metric to compare the performance of various schemes. We also use the per packet delay as a secondary metric. The file transfer time is defined as the interval between the instant at which the first byte of data is available at the transport-level of the sender and the instant at which the sender receives the acknowledgement that the destination has received the last byte of the file. Clearly the file transfer time directly reflects the throughput achieved by a congestion control scheme. The per packet delay serves to measure the response time seen by interactive users.

We argue that various other performance measures considered in previous studies are subsumed in these two measures, namely, the file transfer time and the packet delay time.

- **Bottleneck utilization** If the link remains underutilized even when there is data to send, the average file transfer rate over the link is lower and the file transfer time is higher.
- **Fairness** Several definitions of fairness are available in the literature. As a simple working definition, we consider that a user gets unfair service if its response time or throughput is dependent on where in the network its traffic is introduced. That is, the difference in the file transfer times for two identical sources, with equal amounts of data to send over different path lengths, is a measure of the unfairness in service provided by a flow control scheme.
- **Loss Rate** If a flow control scheme results in many packet losses then these packet losses have to be detected and the lost packets retransmitted. In some flow control schemes, packet losses are also used as an indicator to throttle back the sending rate. This in turn causes an increase in the file transfer time.

Thus, the metric of file transfer time subsumes many of the other metrics. Further, it has the virtue of being quite simple to compute. Although using a single user's file transfer time as a metric has not been used in previous studies of congestion control schemes, the approach has been used in a comparative study of application protocols in [6] and in the analysis of bulk transfers in [3].

The secondary metric that we consider is the mean and variance of single packet transfer delay. This is the time taken for a single packet from a source to reach the destination and an acknowledgement for it to be received back by the source. The mean single packet delay measures the response time seen by a user who uses the network for interactive applications, and the variance of this delay measures the variability in response. We believe that one goal of a congestion control scheme is to ensure that both the mean and the variance of the delay be as low as possible, and this is directly measured by the single packet delay. In practice, this is measured by sending a packet periodically from a source to a destination traversing a bottlenecked link.

One metric that is commonly used to compare such schemes is the network power, which is the net throughput divided by the net delay summed over all conversations. Since the thoroughput is the file size divided by its transfer time, and the delay is captured by the transfer time, the power is proportional to the inverse of the square of the transfer time. Since this does not add any information to our existing metric we do not explicitly use network power to compare the schemes.

How much information is lost by using only two metrics? These metrics are simple but not naive. We believe these metrics capture much of the motivation behind congestion control schemes. As mentioned earlier, in addition to these metrics, factors such as the implementation cost and resource requirements of the scheme can also serve as comparision metrics. However, we believe that such metrics are orthogonal to the performance of the schemes - it is precisely the performance aspects of the schemes that we believe is captured by our metric.

Another point to note is that a file transfer application is not as narrow an application as appears at first sight. Applications as diverse as database access, network news and shared whiteboards use the file transfer paradigm (i.e bulk data transfer at the fastest possible rate). Thus, the performance that we get for a finite file transfer directly relate to the performance of these other applications. Similarly, the single packet delay can be extrapolated to compare the performance of a large class of interactive applications.

3 Designing the Benchmark

Before describing the specific network configurations and traffic scenarios that constitute the benchmark, we make a few general observations about how we designed the benchmark suite.

First, congestion arises when there is a rate mismatch between the input and output of a server. Thus, our network topology must contain at least one possible point of overload.

Second, although networks are more complex in topology, the simple configuration used in all our traffic scenarios is adequate as long as we are not measuring network-wide statistics such as the average network delay. A successful design to test such statistics requires characterization of workload. Although many efforts have been made to measure traffic chractersitcs of existing low-speed networks, we believe that these studies offer an incomplete picture about what would be traffic characteristics in future high speed networks. The scenarios in our suite are not meant to be representative of any network, rather, they are meant to isolate specific potential problems in a congestion control scheme. Our experience shows that by evaluating a scheme on the entire suite, a designer can get a fairly accurate idea of the inherent problems with the scheme.

We explicitly have not considered complex topologies or large scale simulations. While overall network stability can be studied by a large scale simulation that runs for a long time, the large number of measurable quantities makes it unclear exactly what other comparative data would be available to differentiate among various congestion control schemes.

Third, we expect the major problem with congestion control protocols to be an inability to simultaneously deal with high-bandwidth and large propagation delay of a network. Thus, the topology and the values of parameters show exactly these two features. For better understanding of schemes that were designed for low bandwidth-delay networks, we have also included one test scenario where the network under study has low bandwidth-delay product.

Fourth, congestion is most likely when the network is heavily loaded. So, we aim to measure performance at high traffic loads.

Fifth, these experiments should be done on a real test-bed of a gigabit network and perhaps will be done in future on a test-bed that the authors have access to. However, currently access to such networks is not universal. Consequently, we have carried out the experiments with packet-level network simulators.

/* This section would perhaps continue to change. */ A number of different flow control schemes were compared using the benchmark suite. The comparison helped us to study the effectiveness of the benchmark suite in differentiating schemes and pointing out their weaknesses. The five flow control schemes that we compared are the TCP scheme implemented in the 4.3BSD-Tahoe operating system[7], the DECbit scheme[13], the Packet-Pair scheme described in [9], and the explicit end-to-end feedback and hop-by-hop congestion control schemes described in [11]. The scheduling discipline simulated were those assumed in the design of the flow control algorithm i.e. FCFS for TCP and DECbit, and Fair Queueing for the others. The performance was studied with two different packet-based event-driven network simulators. One of the simulator, named REAL, written by S.Keshav [10], is used currently by a number of different groups. This simulator was used to measure performances of JK, PP and DECbit schemes. The other simulator, originally written by L. Zhang [14], was modified to include switch modules dealing with the hop-by-hop rated based congestion control scheme described in [11]. We expect to expand the number of schemes compared as others undertake the task and provide us with results.

In Section 2, we present the ten different experiments included in the benchmark suite. After describing the detailed configuration of each scenario in the suite, we discuss the motivation behind each scenario and compare the performance of various schemes. /* the comparision part has been left out pending numbers */ (We intend to include most of the prominent schemes that have been proposed in the literature in the final version of this paper. We will also include other traffic scenarios if it turns out that we have overlooked an aspect of performance in this benchmark.) In Section 3, we discuss larger issues such as the lessons we learned in designing the benchmark and also our interpretation of the results observed so far.

4 Benchmark Suite and Results

In the following section, we describe the motivation for each experiment, its configuration and the parameters used. The values of the adjustable parameters of each control scheme were kept unchanged for all the experiments included in the test suite. The network configuration used in each experiment stays invariant for all the schemes except the hop-by-hop scheme. The configuration used for the hop-by-hop scheme is modified so that there are three additional intermediate hops between the source and the bottleneck; the round-trip delay path for each traffic source remains the same. This modification is meant to highlight the difference between hop-by-hop schemes and end-to-end schemes. For brevity, we exclude the exact details of the configurations used for hop-by-hop experiments.

We choose to model all sources and switches as being infinitely fast. Thus, bottlenecks always occur at the output queues of switches. In the scenarios, there are slow lines that act as bottlenecks, and fast lines that feed data to switches and bottleneck lines. Slow lines have a bandwidth of 40 Mbps and fast lines have a bandwidth of 100 Mbps. The propagation delay along each line is marked in each scenario. Sources are assumed to have finite amounts of data to send, as described in each scenario. The packet size is 500 bytes. This roughly corresponds to the measured mean value of 570 bytes in the DARPA Internet [1].

Sinks acknowledge each packet received, and set the sequence number of the acknowledgment packet to that of the highest in-sequence data packet received so far. Acknowledgement packets traverse the data path in the reverse direction. The acknowledgement (ack) packets are 40 bytes long. The switches have finite buffers whose sizes, for convenience, are measured in packets rather than bytes. For most of the experiments we assume that each switch has one bandwidth-delay product worth of buffers, although in one of the scenarios we use much fewer buffers. The switch buffers are assumed to be shared by all the active connections.

In all the experiments we measure the file transfer time for one or two primary connections. The file transfer sizes and the time at which each of the cross traffic streams switches on varies between experiments. However, in all the experiments the cross traffic streams come on after the primary connection and finish transmission before the primary connection(s). This allows us to use the file transfer time of the primary connection(s) to study the behavior of the flow control schemes in response to both increases and decreases in the available bandwidth. As a guide to how much better we could expect any scheme to be, we have calculated the optimal file transfer time for each experiment. The ideal transfer time is computed assuming that the sources simply place the entire file in the (infinitely large) bottleneck buffers at the start of transmission, and the bottleneck server

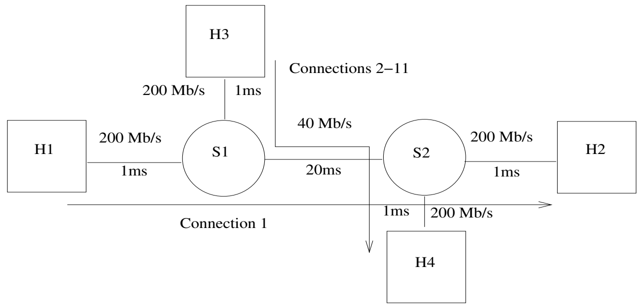

Figure 1: Network configuration for Experiments 1, 2 and 7

allocates each source its fair share. Single-packet delay is measured by a source that sends a 500 byte packet once every 20ms.

4.1 Slow Changes in Cross-Traffic

The configuration used for this experiment is shown in Figure 1. In this scenario, the cross-traffic load seen by the primary connection changes gradually. Connection 1, shown in Figure 1, starts up at time zero and has 2 Megabytes of data to send. The first cross-traffic stream, connection 2, starts at 100 ms and nine more cross-traffic streams come up at intervals of 44 ms (which is the round-trip propagation delay for all the connections). Since the fundamental time constant for reacting to changes in network state is one round trip time (RTT), the flow control mechanisms are highly stressed when new traffic sources come up spaced apart one RTT. Each cross traffic stream has 500 Kbytes of data to send. The transmission speeds of links and the size of files are chosen such that the cross-traffic load gradually increases and then decreases back to zero during the interval in which connection 1 is transmitting its data.

Experimental Results The file transfer times for connection 1 is shown in Table 4.1. The optimal transfer time is 1444 ms.

4.2 Sudden Jumps in Cross-Traffic

The configuration used for this experiment is shown in Figure 1. In this scenario, the cross-traffic load seen by the primary connection changes very rapidly. This scenario also indirectly evaluates the differences among schemes in terms of buffers required at the bottleneck in face of a large perturbation. Connection 1, shown in Figure 1, starts up at time zero and has 2 Megabytes of data to send. The first cross-traffic stream, connection 2, starts at 100 ms and nine more cross-traffic streams come up at intervals of 4 ms. This is much smaller than one RTT, so connection 1 experiences a large drop in available bandwidth within one RTT. At the same time, the interval of 4 ms is large enough to throw off any schemes that simply count the number of new connection

Scheme	Transfer time (ms)	Delay (ms)	
	Conn. 1	Mean	Std. Dev.
optimal	1444	0	0
hbh	1534	0.99	0.41
pp	1569	0.31	0.17
e2e-ef	1621	0.85	0.42
tcp (tahoe)	4430		
tcp (reno)	?		
tcp (w/red)	?		
tri-s	?		
dual	?		
decbit	5566		

Table 4.1: File Transfer time for Connection 1 with gradual changes in cross-traffic

Scheme	Transfer time (ms)	Delay (ms)	
	Conn. 1	Mean	Std. Dev.
optimal	1444		
hbh	1543	1.09	0.51
pp	1594	0.36	0.23
e2e-ef	1605	1.01	0.49
tcp	3775		
tcp (reno)	?		
tcp (w/red)	?		
tri-s	?		
dual	?		
decbit	5568		

Table 4.2: File transfer times for Connection 1 with a sudden large perturbation in cross-traffic

requests and accurately allocate bandwidth to existing and new connections. Each cross traffic stream has 500 Kbytes of data to send.

Experimental Results The file transfer times for connection 1 in Experiment 2 are shown in Table 4.2.

4.3 Transfer Time for Short Files

The transfer time is a critical performance measure for small file transfers. The main factor that increases the transfer time for short amounts of data transfers is the "slow-start" mechanism used by many flow control schemes. The term "slow-start" refers to the fact that when a new connection starts, it starts sending packets with a slowly increasing rate starting from an initially low rate. The slow increase in the rate at which new traffic is injected in the network allows existing traffic sources to lower their sending rates provided a bottleneck develops.

In this experiment, we aim to measure the differences that exists among the "slow-start" mechanisms used by flow control schemes. We want to isolate the slow start mechanism from the effect of lost packets and lost bandwidth due to cross traffic. Hence in this scenario, there is no cross

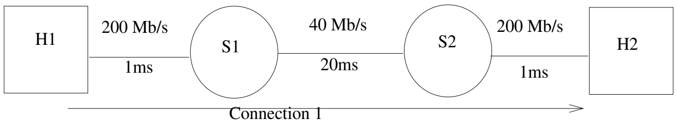

Figure 2: Network Configuration for Experiment 3

Scheme	Transfer time (ms)		
	4 pkts	20 pkts	200 pkts
optimal	44.4	46	64
hbh	53	73	122
pp	84	92	146
e2e-ef	47	63	150
tcp	120	200	327
tcp (reno)	?	?	?
tcp (w/red)	?	?	?
tri-s	?	?	?
dual	?	?	?
decbit	120	321	1203

Table 4.3: File transfer times for short files

traffic. The configuration used in this experiment is shown in Figure 2. File sizes of 4, 20 and 200 packets are used in this configuration.

Experimental Results The file transfer times for sending files of 4, 20 and 200 packets are shown in Table 4.3.

4.4 Fairness

Previous studies of TCP's performance have observed that connections on a longer delay path get a smaller share of the bandwidth of the bottleneck. This "unfairness" increases the file transfer times for traffic streams going over longer paths. In this experiment the fairness in bandwidth allocations is measured. The configuration used in this experiment is shown in Figure 3. Two primary connections, 1 and 2, are started at time zero and both have 2 Megabytes of data to send. Connection 2 goes over a path with approximately twice the end-to-end propagation delay of that of connection 1. Both connections see the same cross-traffic load at the bottleneck link shared by these connections. The starting times and file sizes of the cross traffic streams is the same as in Experiment 1.

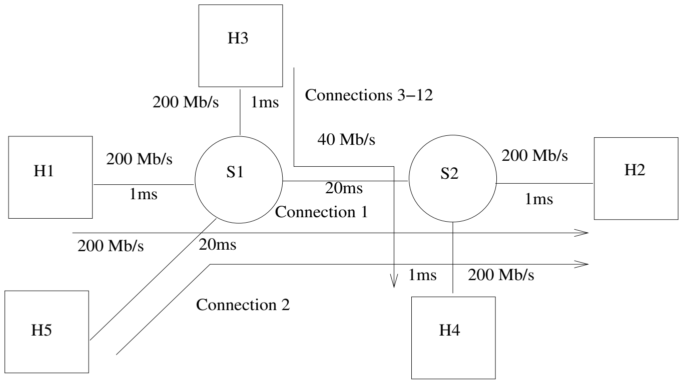

Figure 3: Network configuration for Experiment 4

Scheme	Transfer time (ms)	
	Conn. 1	Conn. 2
optimal	1844	1882
hbh	1859	1927
pp	1859	2022
e2e-ef	1877	2004
tcp	4030	8643
tcp (reno)	?	?
tcp (w/red)	?	?
tri-s	?	?
dual	?	?
decbit	6316	12010

Table 4.4: File transfer times for two connections with different round-trip delays

Experimental Results The file transfer times for sending files over the two different delay paths are shown in Table 4.4.

4.5 Migrating Bottlenecks

There will be only one bottleneck at any one time in the path of a connection. But the site of the bottleneck may migrate with changes in the network traffic. In this experiment, we explore the performance of flow control schemes in the presence of migrating bottlenecks. For simplicity, we have considered a generic case where a bottleneck first develops at a link, shifts to another link and then finally shifts back to the original link. The shifts in the site of the bottleneck are induced by introducing cross-traffic streams with different file transfer sizes over two separate links. The primary connection comes on at time zero with 2 Megabytes of data to send. Connection 2 switches on at 100 ms with 1 Megabyte of data to send. After connection 2 switches on, the S1-S2 link becomes the bottleneck for connection 1. At time 250 ms connections 3 and 4 switch on with

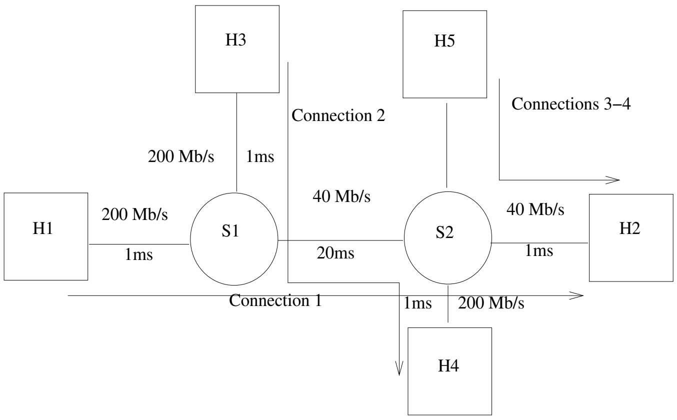

Figure 4: Network configuration for Experiment 5

Scheme	Transfer time (ms)	Delay (ms)	
	Conn.1	Mean	Std. Dev
optimal	644		
hbh	714	0.49	0.07
pp	771	0.082	0.041
e2e-ef	769	0.37	0.07
tcp (tahoe)	3237		
tcp (reno)	?		
tcp (w/red)	?		
tri-s	?		
dual	?		
decbit	5846		

Table 4.5: File transfer times for the migrating bottleneck scenario

250 Kilobytes of data each to send. This causes the bottleneck for Connection 1 to shift to the S2-H2 link. When connections 3 and 4 terminate the S1-S2 link again becomes the bottleneck for connection 1. The configuration used in this experiment is shown in Figure 4.

Experimental Results The file transfer times for sending files are shown in Table 4.5.

4.6 Two-Way Traffic

A traffic scenario with traffic flowing in both directions through a switch is important to examine because of the role played by acknowledgement packets in some flow control schemes [15]. The TCP scheme uses acks as a clocking mechanism to inject packets into the network at roughly the bottleneck rate. The packet-pair scheme uses the gap between acknowledgment packets to estimate the service rate at the bottleneck node. Unfortunately, whenever there is congestion or even burstiness in the traffic along the reverse path, there is a chance that two successive acks may

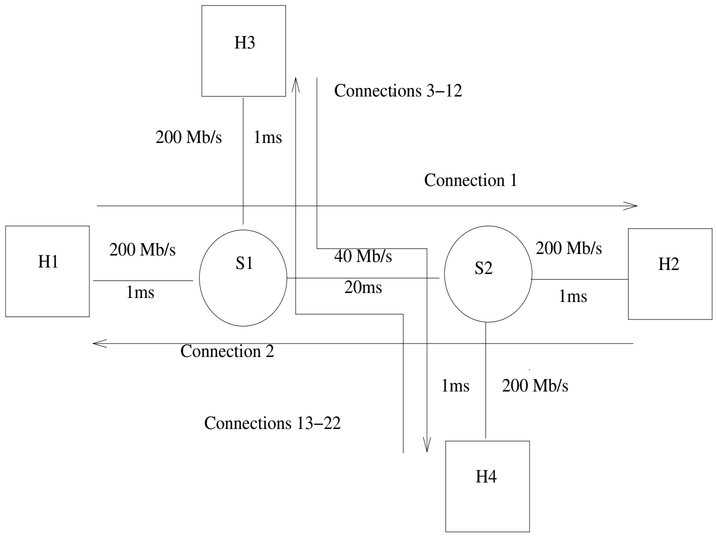

Figure 5: Network configuration for Experiment 6

Scheme	Transfer time (ms)		Delay (ms)	
	Conn. 1	Conn. 2	Mean	Std. Dev.
optimal	1556	1556		
hbh	1657	1651	5.67	4.06
pp	1677	1663	0.38	0.216
e2e-ef	1712	1722	1.23	0.55
tcp (tahoe)	5627	4435		
tcp (reno)	?	?		
tcp (w/red)	?	?		
tri-s	?	?		
dual	?	?		
decbit	7122	7122		

Table 4.6: File transfer times for 2-Way traffic scenario

get bunched up and thus adversely affect schemes that use the frequency of ack arrivals or inter-ack gaps in their control policies. This experiment is designed to detect any such conditions and to measure its impact on the performance.

The configuration used in this experiment is shown in Figure 5. The configuration is similar to that used in Experiment 1 with all the connections being duplicated and reversed so as to create an exactly symmetrical flow in both directions. Thus, the primary reverse connection is started by the host that is the sink for the primary forward connection. Both the primary forward and reverse connections start at time zero. Cross-traffic streams are also mirrored in a similar fashion.

Experimental Results The file transfer times for sending files in the forward and backward directions are shown in Table 4.6.

Scheme	Mean (standard deviation) of transfer time at cross traffic load				
	0.2	0.4	0.6	0.8	0.9
optimal	1444(0.0)	?	?	?	?
hbh	591 (3.74)	746 (8.52)	1063 (19.04)	2097 (29.08)	4094 (61.45)
pp	681 (8.5)	934 (7.9)	1331 (19.5)	2236 (26.8)	4112 (134.6)
e2e-ef	641 (2.21)	785 (6.89)	1088 (16.14)	2103 (31.82)	4092 (59.58)
tcp (tahoe)	3969	?	?	?	?
tcp (reno)	?	?	?	?	?
tcp (w/red)	?	?	?	?	?
tri-s	?	?	?	?	?
dual	?	?	?	?	?
decbit	6859	?	?	?	?

Table 4.7a: File transfer times with non-compliant cross-traffic streams

Scheme	Mean (standard deviation) of single packet delays at cross traffic load				
	0.2	0.4	0.6	0.8	0.9
optimal	0.0 (0)	0.0 (0)	0.0 (0)	0.0 (0)	0.0 (0)
hbh	0.59 (0.46)	0.61 (0.41)	0.69 (0.36)	0.94 (0.29)	1.37 (0.16)
pp	0.10 (0.02)	0.11 (0.06)	0.17 (0.05)	0.39 (0.09)	0.56 (0.10)
e2e-ef	0.46 (0.45)	0.50 (0.42)	0.58 (0.36)	0.84 (0.30)	1.28 (0.16)
tcp (tahoe)	3969	?	?	?	?
tcp (reno)	?	?	?	?	?
tcp (w/red)	?	?	?	?	?
tri-s	?	?	?	?	?
dual	?	?	?	?	?
decbit	6859	?	?	?	?

Table 4.7b: Single packet delays with non-compliant cross-traffic streams

4.7 Non-compliant Cross-traffic

In a network, one should not expect that all traffic streams will be subject to a single flow control discipline. For example, a real-time traffic source such as a packet video camera is unlikely to follow the same flow control scheme as that followed by a bursty data source such as a file transfer. Furthermore, there will always be some short data transfer applications (such as distributed computation) which will not be subject to any end-to-end flow control. Thus, it seems important to measure differences in the behavior of flow control schemes when other non-compliant traffic streams are present. This has been the motivation for the experiment.

The configuration used in this experiment is shown in Figure 1. The configuration is identical to that used in Experiment 1 except that the cross-traffic streams are uncontrolled and the packet arrival process for each cross traffic source is Poisson such that the aggregate arrival rate of all the cross connections is 90% of the bottleneck link bandwidth. As before the primary connection comes on at time zero and the cross traffic streams switch on at 44 ms intervals, starting at 100 ms.

Experimental Results The file transfer times are shown in Tables 4.7.

Scheme	Transfer time (ms)	Delay (ms)	
	Conn. 1	Mean	Std. Dev.
optimal			
hbh	x	x	x
pp	1991	0.04	0.03
e2e-ef	x	x	x
tcp (tahoe)			
tcp (reno)			
tcp (w/red)			
tri-s			
dual			
decbit			

Table 4.8: File transfer times with smaller number of buffers at a switch

4.8 Fewer Data Buffers at a Switch

In this experiment, we study the effect of having fewer buffers at the switch. While we believe that having at least one round-trip-time worth of buffering (shared among all conversations) is a minimum, it may sometimes be the case that this amount of buffering is not available. Here we study the performance of various feedback schemes when stressed by lack of buffers. The scenario is exactly as in experiment 1, except that the number of buffers has been reduced to 44 buffers from 440.

Experimental Results The file transfer times and single-packet delays are shown in Table 4.8.

4.9 A Low Bandwidth-Delay Network

The experiments so far have dealt with scenarios where the bandwidth-delay product is fairly large. Currently, such high bandwidths are not easily available. So it is interesting to study the peformance of various schemes where the product is smaller. In this scenario, we have a situation identical to that in Scenario 1, except that the bottleneck speed is only 1.5Mbps, and the roundtrip delay is 12ms. The bandwidth delay product is 5.5 packets, and there are only 20 buffers at the bottleneck. Because of the reduced bandwidth, single packet delay is probed by a source that sends a packet every 500ms.

Experimental Results The file transfer times and single-packet delays are shown in Table 4.9.

4.10 A Torture Test for Rate-based Schemes

This is a scenario suggested by a colleague as a tough one for rate-based schemes such as HBH or packet-pair schemes. The configuration used in this experiment is shown in Figure 6. In this scenario, we have Connection 2 with a round trip time of 20.2 ms and Connection 1 with rtt of 0.2 ms sharing a bottleneck link. Cross-traffic is provided by 2 on-off sources that start at times 1.5 ms and 3 ms respectively. Each on-off source sends 1/4 the total bottleneck link's capacity for 2.5 ms and then goes idle for 1.5 ms. The number of backlogged sources changes thus changes from 1 to 2 to 1 several times in a round-trip time of 1 source (20.2 ms rtt) whereas the other one gets the chance to adjust almost immediately.

Scheme	Transfer time (seconds)	Delay (s)	
	Conn. 1	Mean	Std. Dev.
optimal			
hbh	x	x	x
pp	1447	0.04	0.04
e2e-ef	x	x	x
tcp (tahoe)			
tcp (reno)			
tcp (w/red)			
tri-s			
dual			
decbit			

Table 4.9: File transfer times in a low bandwidth-delay network

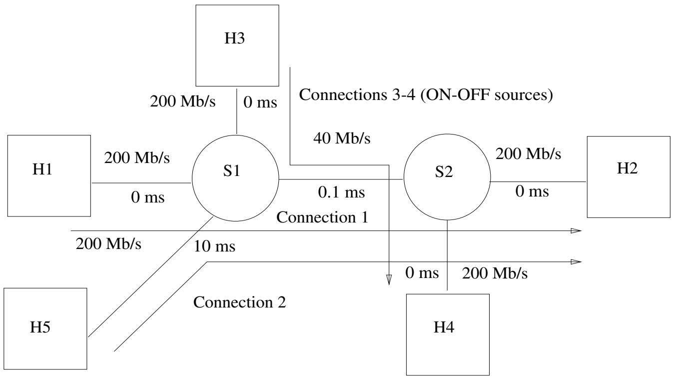

Figure 6: Network configuration for Experiments 10

Experimental Results The file transfer times and single-packet delays are shown in Table 4.10.

4.11 Additional Experiments?

We have strived hard to be inclusive and designed experiments to test all aspects of congestion control schemes without bias. If we have overlooked any aspect, we remain willing to add new experiments to the benchmark. The emphasis has definitely been on traffic scenarios for networks with a high bandwidth-delay product. Such scenarios accentuate the differences between the congestion control schemes. Moreover, we expect such scenarios to be common in future higher speed networks.

Scheme	Transfer time (ms) Conn. 1	Conn. 2
optimal	?	
hbh	1247	1254
pp	1113	1183
e2e-ef	1198	1237
tcp (tahoe)	?	
tcp (reno)	?	
tcp (w/red)	?	
tri-s	?	
dual	?	
decbit	?	

Table 4.10: File transfer times for the torture test of rate-based schemes

5 Conclusions

We have described in this paper an objective set of experiments to be used for evaluating the performance of one's favorite flow and congestion control scheme. It will be hard to prove that the benchmark proposed here is fair to all proposals on flow and congestion control schemes. But, with this beginning and with a willingness to add more experiments if warranted, we should end up with a benchmark suite that satisfies a large number of researchers in the field. Benchmarks such as this are quite useful from many perspectives.

- A benchmark serves to measure progress made over the years.
- A benchmark also helps in fine-tuning a scheme or in rejecting bad ideas before a lot of work is expended in building a system. The work we did in designing these experiments, has already been very helpful to us in this sense. The Packet-pair mechanism proposed in [9] did not have a slow-start mechanism. The source jumped rather quickly to its starting rate based on a single probe of a packet pair. On analyzing the performance of the Packet-Pair scheme, particularly for Scenario 2, a better approach was found and is now incorporated in PP scheme. The performance results shown here are for a version of PP with the inverse-exponential rule for increase the sending rate to the target rate. The modification reduces buffer build-up and thus reduces packet losses at the bottleneck.
- All previous studies of flow control schemes have focused on traffic dynamics with infinite data sources. In some cases, we have observed that network delay is reduced at the expense of higher wait times at a host for the data. By considering data transfers of a finite size, we have first of all simplified the comparison among schemes. Second, the file transfer time for finite sized data transfers also includes the wait time at a host. Various network-related aspects such as packet losses, oscillations in delays and optimal service rates, adaptation to dynamic traffic changes are all combined in a single measure, namely, the file transfer time.
- The performance as compared in this benchmark is from the perspective of a single user. We have adjusted the cross-traffic model to explore various aspects of congestion control, while using a single performance measure, file transfer time, to compare performances.
- We have also shown here performance results of the benchmark study of five different congestion control schemes. Supported by these results, we claim that the benchmark is detailed enough to highlight differences in performances among various schemes.

References

[1] R. Caceres, P. B. Danzig, S. Jamin and D. J. Mitzel. Characteristics of application conversations in TCP/IP wide-area internetworks. Proc. of ACM SIGCOMMM 91, September 1991.

[2] A. Demers, S. Keshav, and S. Shenker Analysis and Simulation of a Fair Queueing Algorithm *Journal of Internetworking Research and Experience* September 1990. pp. 3-26.

[3] R. Feix and B. Wolfinger Models for Bulk Transfers in Packet-Switched Networks *Submitted for Publication to INFCOM'91*

[4] K. W. Fendick, D. Mitra, I. Mitrani, M. A. Rodrigues, J. B. Seery, and A. Weiss. An approach to high-performance high-speed data networks. *IEEE Communications Magazine*, 30:74–82, Oct. 1991.

[5] K. W. Fendick, M. A. Rodrigues, and A. Weiss. Analysis of a rate-based feedback control strategy for long haul data transport. *Submitted for publication*, 1991.

[6] P. Gunningberg, M. Bjorkman, E. Nordmark, S. Pink, P. Sjodin and J.-E Stromquist Application Protocols and Performance Benchmarks *IEEE Communicatins Magazine*, pp. 30-36 June 1989

[7] V. Jacobson. Congestion avoidance and control. In *Proceedings of ACM SIGCOMM '88*, Aug 1988.

[8] S. Keshav Congestion Control in Computer Networks PhD thesis. University of California, Berkeley. August 1991

[9] S. Keshav. A control theoretic approach to flow control. In *Proceedings of ACM SIGCOMM '91*, Sept 1991.

[10] S. Keshav. REAL : A Network Simulator. Computer Science Dept. TR 88/472. University of California, Berkeley, December 1988.

[11] P. Mishra and H. Kanakia. A hop by hop rate based congestion control scheme. Technical Report 11273-920225-01TM, AT & T Bell Labs, Murray Hill, NJ, Mar 1992. To also appear in SIGCOMM'92. August 1992. Baltimore, USA.

[12] D. Mitra and J. B. Seery. Dynamic adaptive windows for high speed data networks: Theory and simulations. In *Proceedings of ACM SIGCOMM '90*, pages 20–30, Sept 1990.

[13] K. K. Ramakrishnan and R. Jain A Binary Feedback Scheme for Congestion Avoidance in Computer Networks ACM TOCS Vol. 8 No. 2 pp. 158-181. May 1990

[14] L. Zhang. Virtual clock: A new traffic control algorithm for packet switching network. In *Proceedings of ACM SIGCOMM '90*, pages 8–18, Sept 1990.

[15] L. Zhang, S. Shenker, and D. Clark. Observations on the dynamics of a congestion control algorithm: The effects of two way traffic. In *Proceedings of ACM SIGCOMM '91*, pages 30–39, Sept 1991.

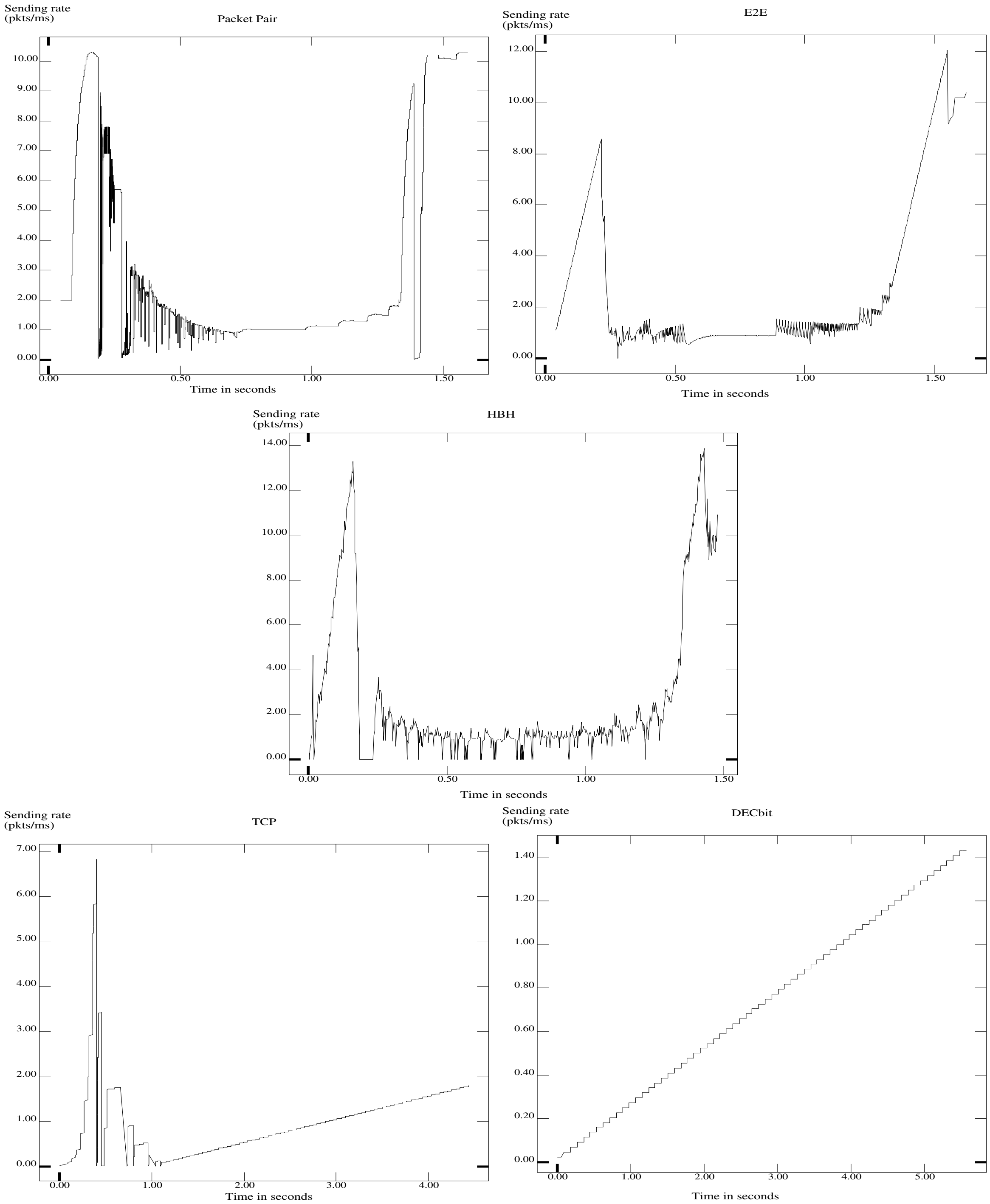

Figure 7: Sending rate for Connection 1 for all 5 schemes