Connections with Multiple Congested Gateways in Packet-Switched Networks Part 1: One-way Traffic

Sally Floyd*
Lawrence Berkeley Laboratory
1 Cyclotron Road
Berkeley, CA 94720
floyd@ee.lbl.gov

Abstract

In this paper we explore the bias in TCP/IP networks against connections with multiple congested gateways. We consider the interaction between the bias against connections with multiple congested gateways, the bias of the TCP window modification algorithm against connections with longer roundtrip times, and the bias of Drop Tail and Random Drop gateways against bursty traffic. Using simulations and a heuristic analysis, we show that in a network with the window modification algorithm in 4.3 tahoe BSD TCP and with Random Drop or Drop Tail gateways, a longer connection with multiple congested gateways can receive unacceptably low throughput. We show that in a network with no bias against connections with longer roundtrip times and with no bias against bursty traffic, a connection with multiple congested gateways can receive an acceptable level of throughput.

We discuss the application of several current measures of fairness to networks with multiple congested gateways, and show that different measures of fairness have quite different implications. One view is that each connection should receive the same throughput in bytes/second, regardless of roundtrip times or numbers of congested gateways. Another view is that each connection should receive the same share of the network's scarce congested resources. In general, we believe that the fairness criteria for connections with multiple congested gateways requires further consideration.

*This work was supported by the Director, Office of Energy Research, Scientific Computing Staff, of the U.S. Department of Energy under Contract No. DE-AC03-76SF00098.

1 Introduction

In this paper we investigate the throughput of connections in TCP/IP networks with multiple congested gateways. There are two distinct motivations for this investigation. One motivation of this paper is to consider the effect of multiple congested gateways on throughput. The second motivation of this paper is to consider general performance and fairness goals for networks that include connections with multiple congested gateways.

[M90b] reports on a measurement study of a network with multiple congested gateways, comparing the performance with Drop Tail and with Random Drop gateways. In that measurement study, the throughput for the longer connection was better for some scenarios with Drop Tail gateways, and better in other scenarios with Random Drop gateways. In this paper we show that for our scenario, when the simulations with Drop Tail gateways are constructed to remove traffic phase effects [FJ91a], the performance of networks with Drop Tail gateways and with Random Drop gateways are quite comparable. This paper gives a quantitative analysis of the throughput of connections with multiple congested gateways. We use both simulations and a heuristic analysis to investigate performance.

TCP/IP networks have a bias against connections passing through multiple congested gateways [DKS90] [M90b], a bias against connections with longer roundtrip times [H89] [Z89], and a bias against bursty traffic. We consider the interaction of these biases. In a previous paper [FJ91a] we reported briefly on investigations of the bias against connections with longer roundtrip times, and we discussed modifications to the TCP window-increase algorithm that could correct this bias. In this paper we principally consider the bias against connections passing through multiple congested gateways.

Section 2 discusses previous work discussing the biases in TCP/IP networks against connections with longer roundtrip times or against connections passing through multiple congested gateways. In section 3 we describe our simulator, and we define the TCP window modification algorithms and the gateway packet-dropping algorithms examined in our simulations. In section 4 we give the results of simulations. Subsection 4.4 discusses the bias against bursty traffic in simulations with Drop Tail and Random Drop gateways. Subsection 4.5 discusses the implications of the simulations for general traffic with multiple congested gateways.

Section 5 gives a heuristic analysis of the throughput for a network with multiple congested gateways. The heuristic analysis and the simulation results are in close agreement. Section 6 considers the conflicting network goals of maximizing system throughput and maintaining fairness for networks with multiple congested gateways. Section 7 gives conclusions, and discusses related work in progress.

In this paper we restrict our attention to networks with one-way traffic. In a second paper (in progress) we consider the complications introduced by two-way traffic.

2 Related work

In [M90b] Mankin presents a measurement study of a network with local and long distance traffic, with several congested gateways. The Random Drop and the Drop Tail gateway algorithms are compared. Three topologies are explored, with one, two, and three congested gateways respectively. The longer connection's throughput is better with Random Drop gateways for some topologies, and better with Drop Tail gateways for other topologies. (As [FJ91a] explains, we believe that these results should be interpreted keeping traffic phase effects in mind.) Mankin remarks that the longer connection is disproportionately likely to have packets dropped at the gateway, particularly with Drop Tail gateways.

In [DKS90] several simulations are run for a network with three congested gateways, with one long connection and three shorter connections. In the simulations with Fair Queueing or with the selective DECbit algorithm, the longer connection receives half of the available throughput at each gateway. For the simulations using the algorithms in 4.3 tahoe BSD TCP and FIFO Drop Tail gateways, the longer connection receives 29% of the throughput at each gateway.

Several researchers have discussed the bias in TCP/IP networks against connections with longer roundtrip times [H89] [M90b] [Z89]. [RCJ87] explores problems of fairness with the DECbit scheme [RJ90] with connections having different roundtrip times. [RCJ87] outlines two separate approaches for improving fairness in these conditions, a router-based approach and a transport-based approach. Examples of a router-based approach include the selective DECbit scheme [RCJ87] and Fair Queueing gateways [DKS90]. The transport-based approach explored briefly in [RCJ87] involves modifications to the window increase algorithm to ensure an additive increase in the throughput rate rather than in the window size. In this paper we combine a transport-based approach and a router-based approach to explore the range of performance possible for TCP/IP networks with multiple congested gateways.

3 Simulator algorithms

In this section we briefly describe our simulator, and we describe the different window modification algorithms investigated in our simulations. These include the Reduce-to-One and the Reduce-by-Half window-decrease algorithms and the Increase-by-One and the Constant-Rate window-increase algorithms. We also describe several gateway packet-dropping algorithms examined in our simulations. These include Drop Tail, Random Drop, and Random Early Detection (RED) gateways.

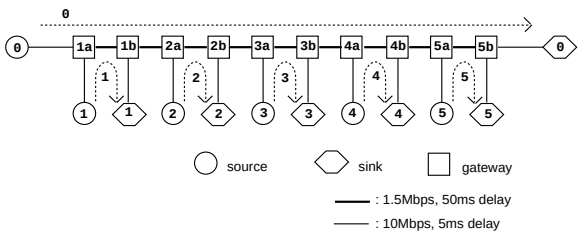

Figure 1: Simulation network with 5 congested gateways.

Figure 1 shows a simulation network with 6 connections, 10 gateways, and 5 congested gateways. The congested gateways in Figure 1 are gateways 1a, 2a, 3a, 4a, and 5a. The dotted lines show the connection paths; source i sends to sink i. Each connection has a maximum window just large enough so that, even when that connection is the only active connection, the network still occasionally drops packets.

We use a family of simulation networks similar to Figure 1, where the number of congested gateways n ranges from 1 to 10. Figure 1 only shows the network for $n = 5$ congested gateways. For a simulation network with n congested gateways for $n \geq 1$ there are $n + 1$ connections and $2n$ gateways. Connection 0 passes

through multiple congested gateways, and connections 1 through n each pass through one congested gateway. Connection 0 is roughly $2n - 1$ times longer than the other n connections. In these simulations all connection paths have the same maximum bandwidth. Therefore a connection with a longer roundtrip time also has a larger delay-bandwidth product.

Our simulator is based on the REAL simulator [K88], which is built on Columbia's Nest simulation package [BDSY88]. Our simulator has been extensively rewritten by Steven McCanne at LBL. FTP sources always have a packet to send and always send a maximal-sized packet as soon as the window allows them to do so. A sink immediately sends an ACK packet when it receives a data packet. The gateways use FIFO queueing. Data packets contain 1000 bytes, and ACK packets contain 40 bytes.

In our simulations each connection is a bulk data transfer with unlimited data. This is not intended to reflect any assumptions about network traffic. Our goal is to start with the simpliest possible network with multiple congested gateways, to see what insight can be obtained from simulations and from an analysis of this simple network. We discuss later in the paper the implications of these results for networks with a more realistic traffic mix.

Definitions: Reduce-to-One, Increase-by-One. Some of the simulations in the paper use the Reduce-to-One window decrease algorithm and the Increase-by-One window increase algorithm from 4.3 tahoe BSD TCP [J88]. Our simulator does not use the 4.3-tahoe TCP code directly but we believe it is functionally identical. Briefly, there are two phases to the window-adjustment algorithm. In the slow-start phase the window is doubled each roundtrip time until it reaches a certain threshold. Reaching the threshold causes a transition to the congestion-avoidance phase where the window is increased by roughly one packet each roundtrip time. In this paper we call the increment algorithm used in the congestion-avoidance phase the Increase-by-One algorithm. Packet loss (a dropped packet) is treated as a "congestion experienced" signal. The source uses timeouts or "fast retransmit" to discover the loss (if four ACK packets acknowledging the same data packet are received, the source decides a packet has been dropped) and reacts by setting the transition threshold to half the current window, decreasing the window to one packet and entering the slow-start phase. In this paper we call this the Reduce-to-One algorithm. □

In order to achieve the highest throughput for longer connections, some of these simulations use the Fast Recovery algorithm in 4.3 reno BSD TCP, along with Selective Acknowledgements (or SACKs). In this paper the Fast Recovery algorithm implemented in 4.3 reno BSD TCP is called the Reduce-by-Half algorithm. The use of the Reduce-by-Half window-decrease algorithm also simplifies the analysis of the behavior of the network.

Definitions: Reduce-by-Half window decreases and Selective Acknowledgements. With the Reduce-by-Half window decrease algorithm, when a packet loss is detected by the "fast retransmission" algorithm the connection reduces its window by half. The details of the Reduce-by-Half window decrease algorithm, which is implemented in our simulator as in 4.3 reno BSD TCP, are somewhat complex [J90]. For the purposes of this paper, the important feature of the Reduce-by-Half window decrease algorithm is that with the "fast retransmission" algorithm, the source retransmits a packet and reduces the window by half, rather than reducing its window to one packet. For the simulations in this paper, we use Selective Acknowledgement sinks with the Reduce-by-Half algorithm. With Selective Acknowledgement sinks, each ACK acknowledges not only the last sequential packet received for that connection, but also acknowledges all other (non-sequential) packets received. Selective Acknowledgements have been tested in a number of experimental Internet protocols, and their use in a proposed extension to TCP is described in [JB88]. □

Some of the simulations in this section use the TCP Increase-by-One window-increase algorithm for the congestion-avoidance phase of the window-increase algorithm. As shown in [FJ91a], this algorithm has a bias against connections with longer roundtrip times. In order to eliminate this bias, some of our simulations use the Constant-Rate algorithm instead in the congestion-avoidance phase. We are *not* proposing the Constant-Rate algorithm for current networks; we are in the process of investigating several alternatives to the current Increase-by-One window-increase algorithm. We simply are using the Constant-Rate algorithm to explore throughput in networks with a window-increase algorithm with no bias in favor of shorter-roundtrip-time connections.

Definitions: Constant-Rate window increases. In the Constant-Rate window-increase algorithm, each connection increases its window by roughly $a * r^2$ packets each roundtrip time, for some fixed constant a, and for r the calculated average roundtrip time. Using this algorithm, each connection increases its window by a pkts/sec in each second. For the simulations in this paper, we use $a = 4$. Connections with the Increase-by-One and the Constant-Rate window-increase algorithms both start with the slow-start algorithm of doubling the window each roundtrip time until a threshold has been reached [J88]. □

In this paper we examine networks with Drop Tail,

Random Drop, and RED gateways. As shown in [FJ91a], simulations and measurement studies with Drop Tail gateways are vulnerable to traffic phase effects; small changes in network parameters can result in large changes in the performance of the network. In order to avoid these phase effects in networks with Drop Tail gateways, in this paper the simulator adds a small random component to the roundtrip time for each packet in simulations using Drop Tail gateways. (This is discussed in [FJ91a].) Normally, our simulator charges zero seconds for the time required to process packets at the nodes. In this paper each source node adds a random time uniformly distributed in [0, b], for $b \approx 5.3$ ms. the bottleneck service time of the network, to the time required by the source node to process each ACK packet in the simulations with Drop Tail gateways. This is not intended to model any assumptions about realistic network behavior, but to eliminate problems with traffic phase effects. With this added random component the simulations with Drop Tail gateways give similar results to the simulations with Random Drop gateways.

To avoid the bias against bursty traffic common to Random Drop and to Drop Tail gateways, we also examine performance in simulations with Random Early Detection (RED) gateways, a modified version of Random Drop gateways that detect incipient congestion. RED gateways maintain an upper bound on the average queue size at the gateway. The use of RED gateways also simplifies the analysis of the behavior of the network.

Definitions: RED gateways. With our implementation of RED gateways [FJ91c], the gateway computes the average size for each queue using an exponential weighted moving average. When the average queue size exceeds a certain threshold, indicating incipient congestion, the gateway randomly chooses a packet to drop and increases the threshold. As more packets arrive at the gateway, the threshold slowly decreases to its previous value. The gateway chooses a packet to drop by choosing a random number n in the interval 1 to $range$, where $range$ is a variable parameter of the gateway. The gateway drops the nth packet to arrive at the gateway. With moderate congestion $range$ is large, and the probability that a packet from some connection is dropped is roughly proportional to that connection's average share of packets through that queue. With high congestion $range$ is decreased, decreasing the feedback time to the source node. RED gateways are described in more detail in a paper currently in progress [FJ91c]. □

One advantage of RED gateways is that, unlike Drop Tail and Random Drop gateways, RED gateways do not have a bias against bursty traffic. The bias of Drop Tail and of Random Drop gateways against bursty traffic and the correction of this bias in RED gateways are described in [FJ91a] and [FJ91b]. With Drop Tail or Random Drop gateways, the more bursty the traffic, the more likely it is that the queue will overflow and the Drop Tail or Random Drop gateway will drop a packet. This is because a burst of packets results in a temporary increase in the queue size at the gateway. With RED gateways the detection of congestion depends on the average queue size, not on the maximum queue size. Thus with RED gateways bursty traffic is less likely to result in the detection of congestion. With a RED gateway even when bursty traffic results in the detection of congestion at the gateway, the mechanism for dropping packets ensures that the bursty connection does not have a disproportionate probability of having a packet dropped. □

For the simulations in this paper the maximum queue size is 60 packets, and the RED gateways drop packets when the average queue size is between 10 and 20 packets. (The range from 10 to 20 packets for the average queue size for RED gateways is somewhat arbitrary; the optimum average queue size is still a question for further research.) For simulations with Random Drop and Drop Tail gateways, changes in the maximum queue size affect the total throughput, but have little effect of the distribution of that throughput between short and longer connections. The maximum windows for connections 1 to n are set sufficiently large to force occasional packet drops even in the absence of traffic from connection 0.

4 Simulation results

In this section we give the results of simulations for a family of networks as in Figure 1, with one connection with n congested gateways and n shorter connections each with one congested gateway. For connections with multiple congested gateways we investigate the effects on throughput of the TCP window-increase algorithm, the TCP window-decrease algorithm, and the gateway packet-dropping algorithms. Our simulations show that in a network with RED gateways, the Reduce-by-Half window-decrease algorithm, and the Constant-Rate window-increase algorithm, a connection with a long roundtrip time and multiple congested gateways can receive an acceptable level of throughput. However, we show that with Random Drop or Drop Tail gateways, the current Increase-by-One window-increase algorithm, and the current Reduce-to-One window-decrease algorithm, a connection with a long roundtrip time and multiple congested gateways can be effectively shut-out.

4.1 Simulations with only one active connection

Even with no congested gateways a connection with a long roundtrip time and a large delay-bandwidth product might increase its window fairly slowly with the Increase-by-One window-increase algorithm. In this section we examine the throughput of connection 0 in a family of networks as in Figure 1 in the absence of congestion. This demonstrates that the loss of throughput for connection 0 in simulations with all connections active is due to the congestion at the gateways, and not to any structural problems in the network. (Possible structural problems could include inadequate maximum windows for connection 0 or insufficiently high thresholds for the allowed average queue size at the RED gateways.)

The simulations in Figure 2 use a family of networks as in Figure 1 with only connection 0 active. In Figure 2 the x-axis shows half the total number of gateways. For example for $n = 5$ the simulation network is exactly like the network in Figure 1 with 10 gateways and 5 congested gateways, except that only connection 0 is active. For $n = 1$ connection 0 has a roundtrip time of 127 ms. in the absence of queues and a window of 24 packets is required to achieve 100% throughput. For $n = 10$ connection 0 has a roundtrip time of 2.027 seconds in the absence of queues, and a window of 380 packets is required to achieve 100% throughput. For each simulation connection 0's maximum window is set at the smallest window size sufficient to achieve 100% throughput. The simulations in Figure 2 use the Reduce-by-Half window decrease algorithm, Selective Acknowledgement sinks, and RED gateways.

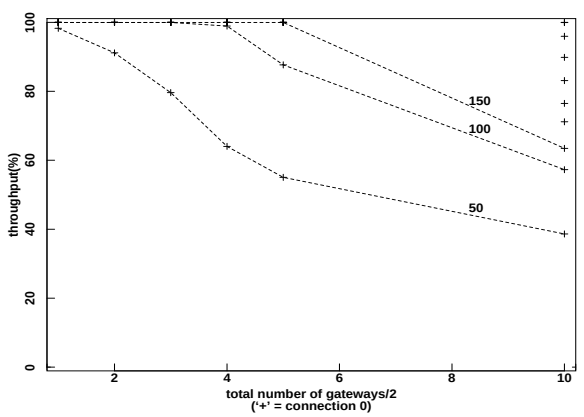

Figure 2: RED gateways, Increase-by-One.

The simulations in Figure 2 use the Increase-by-One window increase algorithm. Each simulation is run for 500 seconds, and there is a mark showing connection 0's throughput for each 50-second period of simulation. The bottom line shows connection 0's throughput in the first 50 seconds of the simulations, the next line shows connection 0's throughput in the second 50 seconds, and so on. For each simulation connection 0's throughput increases with each succeeding 50-second period. As Figure 2 shows, with the Increase-by-One window increase algorithm with $n = 10$ connection 0 requires 400 seconds to achieve its maximum window. □

4.2 Simulations with the Increase-by-One algorithm

Figures 3, 4, and 5 show the result of simulations with the Increase-by-One window increase algorithm. In these simulations the throughput for connection 0 decreases rapidly as the number of congested gateways increases. Figure 3 shows simulations with RED gateways and the Reduce-by-Half algorithm. Figure 4 shows simulations with Random Drop gateways and the Reduce-to-One algorithm. Figure 5 shows simulations with Drop Tail gateways, the Reduce-to-One algorithm, and an added random component to the roundtrip times to eliminate traffic phase effects. Each simulation was run for 500 seconds and each mark represents one 50-second period of that simulation. The x-axis shows the number n of congested gateways and the y-axis shows the throughput for connections 0 and 1 as a percentage of the maximum possible throughput through gateway 1a. The dashed lines show the average throughput for each connection from the simulations. The solid lines show the predictions by the heuristic analysis in Section 5 for the average throughput for each connection. As the number of congested gateways increases the loss of throughput for connection 0 becomes fairly severe.

The simulations in Figure 3 use RED gateways and the Reduce-by-Half window decrease algorithm. The simulations in Figure 5 are closest to the current conditions of the Internet; these simulations use Drop Tail gateways and the Reduce-to-One window decrease algorithm. Both of these choices result in a decrease in throughput for connection 0. Random Drop and Drop Tail gateways have a bias against bursty traffic, and therefore are more likely to drop packets from connection 0 than are RED gateways, as explored in Section 4.4. The use of the Reduce-to-One rather than the Reduce-by-Half window decrease algorithm is also more of a problem for connections with longer roundtrip times, as explained in Appendix A. □

Increase-by-One window-increase algorithm:

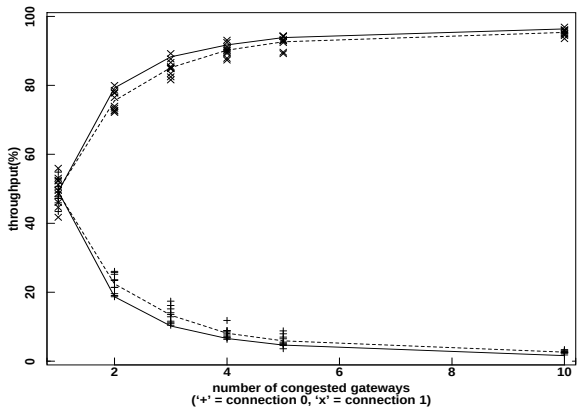

Figure 3: RED gateways, Reduce-by-Half.

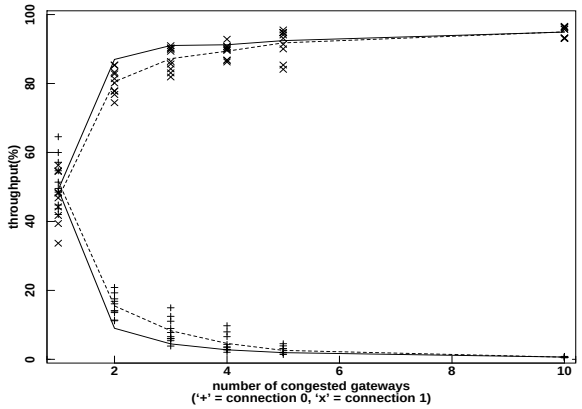

Figure 4: Random Drop gateways, Reduce-to-One.

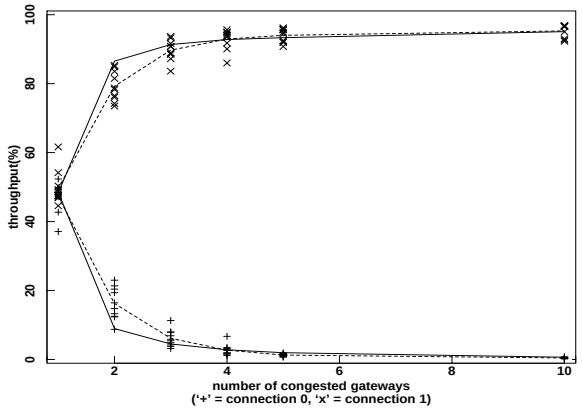

Figure 5: Drop Tail gateways, Reduce-to-One.

Constant-Rate window-increase algorithm:

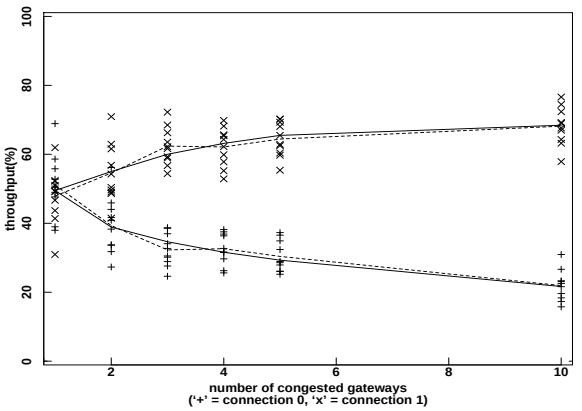

Figure 6: RED gateways, Reduce-by-Half.

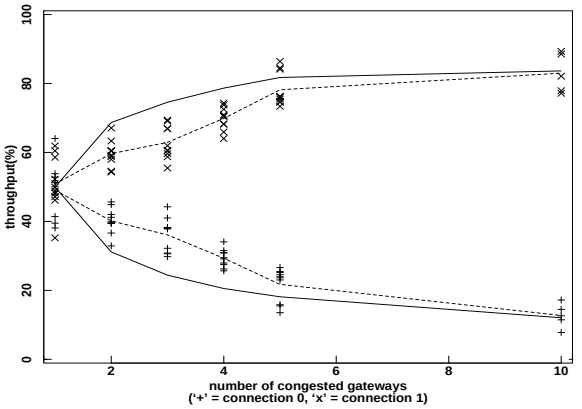

Figure 7: Random Drop gateways, Reduce-by-Half.

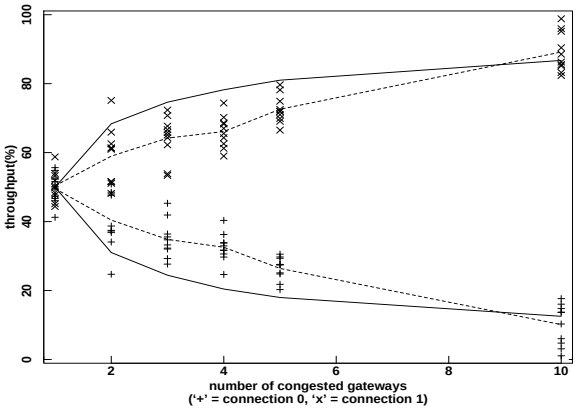

Figure 8: Drop Tail gateways, Reduce-by-Half.

4.3 Simulations with the Constant-Rate algorithm

Figures 6, 7, and 8 show the results of simulations with the Constant-Rate window increase algorithm. Because these simulations are designed to explore the best possible throughput for connections with multiple congested gateways, these simulations use the Reduce-by-Half window decrease algorithm instead of the Reduce-to-One algorithm. In the simulations with RED gateways, connection 0 receives reasonable throughput even with as many as 10 congested gateways. Connection 0's throughput is lower in the simulations with Drop Tail or Random Drop gateways but even in this case, with n congested gateways connection 0 receives at least $1/n$th of the throughput. (If the Reduce-by-Half window decrease algorithm is replaced by the Reduce-to-One window decrease algorithm, then connection 0's throughput suffers significantly.)

The simulations in Figures 3 and 6 differ only in that one uses the Increase-by-One window increase algorithm, and the other uses the Constant-Rate algorithm. Figure 6 shows that even for a connection that passes through multiple congested gateways, it is possible to achieve a reasonable level of throughput in an environment with no bias against connections with longer roundtrip times. □

4.4 The burstiness factor

The throughput is lower for connection 0 in the simulations with Random Drop and Drop Tail gateways than in the simulations with RED gateways because of the bias against bursty traffic in both Random Drop and Drop Tail gateways. This bias has been described in [FJ91a] and in [FJ91b].

Definitions: The burstiness measure. Consider the following measure of burstiness for one roundtrip time of traffic from a connection. Assume that a connection transmit n packets in one roundtrip time of r seconds. Let the *burstiness measure* for that connection in that roundtrip time be the maximum queue needed at the gateway for that connection, given an available bandwidth of exactly n/r packets per second on the bottleneck link. This burstiness measure reflects the increase in the queue caused by the uneven timing of packets arriving at a gateway. □

Given the burstiness measure defined above, even traffic from a large bulk data transfer can be quite bursty, depending on the conditions of the network. For a TCP connection with low throughput, a long roundtrip time, and somewhat frequent packet drops, packets are generally not transmitted at a smooth rate within each roundtrip time. This results from the mechanisms of the window-increase and window-decrease algorithms. Consider the simulations from Figures 7 and 8 for $n = 10$. In these simulations the traffic from connection 0 is likely to be much burstier than the traffic from connections 1 to n. As a result, with Random Drop or with Drop Tail gateways connection 0 receives a disproportionate share of the packet drops, as is shown in Figures 10 and 11.

Figure 9 shows that for the simulations in Figure 6 with RED gateways, connection 0's average share of the packet drops is proportional to connection 0's share of the throughput. For each simulation Figure 9 has a mark for each gateway and each 50-second time period in which there was at least one packet drop. The x-axis shows the number n of congested gateways. For each gateway and each 50-second time period, the y-axis shows connection 0's share of the packet drops divided by connection 0's share of the throughput at that gateway. In many of the simulations with RED gateways, the gateway drops only two or three packets in a 50-second time period. Therefore, for some gateways and time periods connection 0 receives no packet drops. The line shows the average ratio between connection 0's share of the packet drops and connection 0's share of the throughput. As this line shows, for each value of n this average ratio is close to one. (In Figure 9, some marks represent multiple data points.)

Figure 10 shows the average ratio between connection 0's share of the packet drops and connection 0's share of the throughput for the simulations with Random Drop gateways in Figure 7, and Figure 11 shows the average ratio for the simulations with Drop Tail gateways in Figure 8. (Figure 11 does not show all of the data points for simulations with $n = 10$ congested gateways.) Figures 10 and 11 show that for the simulations with Random Drop or Drop Tail gateways and 10 congested gateways, connection 0 receives a disproportionate share of the packet drops at each gateway.

The data in Figure 11 is displayed in a different format in Figure 12. The x-axis in Figure 12 shows connection 0's throughput at some gateway in some 50-second time period, and the y-axis shows connection 0's share of the packet drops divided by connection 0's share of the throughput at that gateway. Figure 12 shows that with Drop Tail gateways, as connection 0's throughput decreases, the ratio between connection 0's share of packet drops and connection 0's share of throughput increases. This is because as connection 0's throughput decreases, the burstiness of connection 0's traffic increases. The line shows the approximation function $y = 0.59 + 5 * e^{-5x}$ (using the density function for the exponential distribution). As is mentioned in Section 5, this approximation function overstates the bias somewhat when the fraction of throughput is greater than

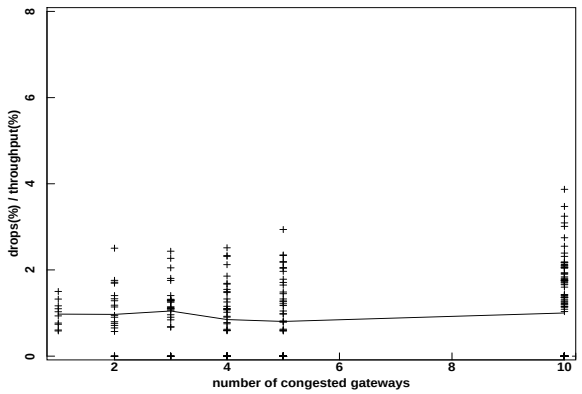

Figure 9: RED gateways, Reduce-by-Half.

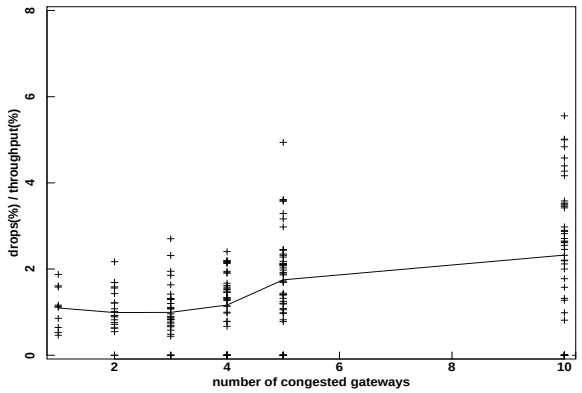

Figure 10: Random Drop gateways, Reduce-by-Half.

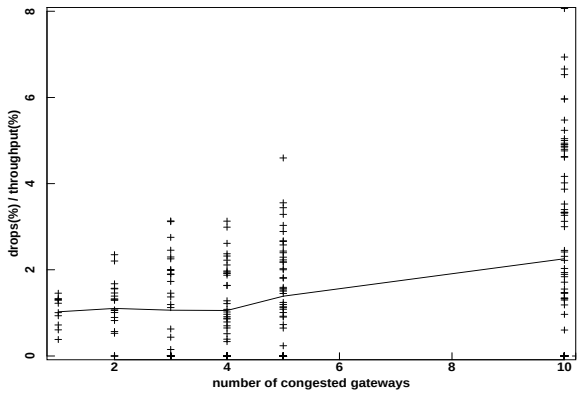

Figure 11: Drop Tail gateways, Reduce-by-Half.

1/4. (The function $y = -1.19 + 1.548/\sqrt{x}$ also gives a plausible fit to the data.)

4.5 Implications of the simulations

The results in this paper are not only of interest for large bulk-data-transfer connections. Consider the network in Figure 1 with $n = 5$ congested gateways. From Figure 2, when connection 0 is the only active connection connection 0 transmits more than 5000 packets in the first 50 seconds of simulation. In simulations in Figure 6 with RED gateways, the Constant-Rate window increase algorithm, and multiple congested gateways, connection 0 transmits 2507 packets in the first 50 seconds. However, in simulations in Figure 5 with Drop Tail gateways and the Increase-by-One window increase algorithm, connection 0 transmits only 178 packets in the first 50 seconds. This is poor performance. In current networks, it is not uncommon for connections to transmit several hundred packets [CDJM91]. (It might be less common in the current Internet for connections to pass through multiple gateways with persistent congestion.)

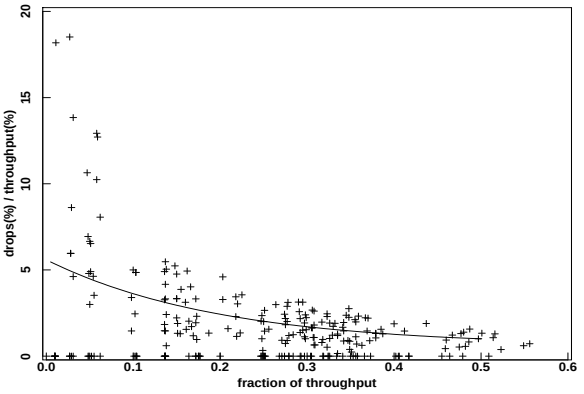

Figure 12: Drop Tail gateways, Reduce-by-Half.

The fact that the simulations uses connections 1 to n as bulk data transfers with unlimited data is not an essential factor in these results. The essential factor is simply that the congested gateways have persistent congestion (with from 11 to 25 dropped packets in each 50-second period of simulation, for the parameters in these simulations). The same level of congestion at the gateways could be achieved without using bulk transfer connections with unlimited data. Thus, the results in this paper could be extended to indicate the general throughput of connections with multiple congested gateways.

As the simulations in this section show, when the effects of multiple congested gateways are combined with the 4.3 tahoe BSD TCP Increase-by-One and Reduce-to-One window algorithms, the result is a disaster for a longer connection traversing multiple congested gateways. As a result of the Increase-by-One algorithm, a longer connection is slow to increase its throughput rate after a packet drop. As a result of the Reduce-to-One algorithm, connections with larger windows are more damaged by a packet drop. However, our simulations show that with the Constant-Rate window increase algorithm and the Reduce-by-Half decrease algorithm, a

connection traversing multiple congested gateways can receive a reasonable level of throughput.

We have several reasons for including simulations with Drop Tail, Random Drop, and with RED gateways. One reason is to compare the performance of Drop Tail and Random Drop gateways. The measurement study in [M90b] was conducted to compare the performance of Drop Tail and of Random Drop gateways for connections with different roundtrip times and multiple congested gateways. For those measurement studies, for some scenarios the longer connection had better performance with Drop Tail gateways, and for other scenarios the longer connection had better performance with Random Drop gateways. In this paper we show that for our simulation network with bulk-data-transfer connections and with a random component added to the simulations with Drop Tail gateways to eliminate traffic phase effects, the simulations with Drop Tail and with Random Drop gateways give quite similar performance.

A second reason to run simulations with Drop Tail, Random Drop, and RED gateways is to compare the simulations of one-way traffic in this paper with simulations of two-way traffic in a following paper. For the simulations of one-way traffic, connection 0's throughput with RED gateways is somewhat higher than connection 0's throughput with Drop Tail and Random Drop gateways. For simulations with two-way traffic the difference in throughput is more pronounced. In simulations with two-way traffic the traffic is much more bursty, due to compressed ACK packets at the gateways [WRM91] [ZC91]. As a result of this bursty traffic the performance for connection 0 suffers significantly with Drop Tail or with Random Drop gateways. In the simulations with two-way traffic, connection 0's performance is significantly better with RED gateways than it is with Drop Tail or Random Drop gateways.

There are several ways that moderate changes to the parameters of the simulation network could result in significant changes in the throughput. For the simulations in this paper, the maximum windows for connections 1 to n are set sufficiently large so that the congested gateways occasionally drop packets even in the absence of traffic from connection 0. If the congestion from connections 1 to n is reduced by reducing the maximum windows, then the throughput for connection 0 will be improved accordingly.

5 A heuristic analysis

In this section we present a heuristic analysis of the throughput in a family of networks as in Figure 1 with multiple congested gateways. In Section 5.1, we isolate the effect of multiple congested gateways on throughput. For the analysis in Section 5.1, all connections are assumed to increase their throughput at the same rate in pkts/sec/sec. In Section 5.2, we isolate the effect of different window-increase algorithms on throughput. This section examines throughput in a network with two connections with different roundtrip times and one shared gateway. Connection i increases its window by $g(r_i)$ packets per roundtrip time, where $g(r_i)$ is a function of the estimated roundtrip time of the connection. In Section 5.3, we consider the effect on throughput of multiple congested gateways and different window-increase algorithms taken together. This section concerns networks with multiple congested gateways and versions of the window increase algorithm defined above.

This is *not* a rigorous analysis of the behavior of the simulation network; this is a heuristic analysis involving several approximations and simplifying assumptions. The emphasis is on giving a simple and clear heuristic analysis for the behavior of the network with multiple congested gateways. Nevertheless, this heuristic analysis does validate the results of the simulations. The analysis gives some insight on the behavior of networks with multiple congested gateways and different window increase algorithms, and therefore also gives some insight into possibilities for modifying this behavior.

5.1 Multiple congested gateways with the Constant-Rate window increase algorithm

In this section we consider the family of networks in Figure 1 with multiple congested gateways, where all connections increase their throughput at the same rate in pkts/sec/sec. The model is based on the behavior of a network with RED gateways, the Constant-Rate window increase algorithm, and the Reduce-by-Half window decrease algorithm.

The multiple gateways model: First we describe the model. In the absence of congestion, all connections increase their throughput rate by the same number of packets/sec each second. (This could be either because all connections have the same roundtrip time, or because the window increase algorithm is the Constant-Rate algorithm.) A gateway detects congestion when the average arrival rate at the gateway equals the maximum transmission rate at the gateway. When a gateway detects congestion the gateway drops one packet, causing that connection to reduce its window. Each connection's probability of having a packet dropped equals that connection's share of the throughput at that gateway.

When a gateway drops a packet that connection instantaneously reduces its window by half, and this reduces its throughput rate by half. The connection again increases its throughput until another packet from that

connection is dropped. This assumes that each connection has a large maximum window. The gateways act asynchronously. Thus, each time some gateway drops a packet from connection 0, connection 0 reduces its window by half. Let M be the maximum possible throughput in pkts/sec through each congested gateway. □

Simplifying assumptions: For a heuristic analysis we make two simplifying assumptions to this model. First, we use an independent Poisson process with rate λ to approximate the number of packets dropped at each gateway. Second, let p be connection 0's average share of the throughput when congestion is detected at a gateway. We analyze the network as if for each packet drop at a gateway, a connection 0 packet is dropped with probability p. These two simplifying assumptions allow us to compute the average number of packet drops for each connection and the average time between packet drops. Using this information, we compute the throughput for each connection.

• **Assumption 1:** We *approximate* the number of packet drops at each congested gateway by an independent Poisson process with rate λ. It follows that each congested gateway has on the average λ packet drops per second, with an average time of $1/\lambda$ seconds between packet drops [R83, p.31]. This assumes that each congested gateway drops on the average the same number of packets, that these packet drops for different gateways are uncorrelated, and that the packet drops for a particular gateway are uniformly distributed in time. □

• **Assumption 2:** For this analysis, we assume that there is some p for $0 \leq p \leq 1$ such that the following two conditions hold: (1) for each gateway, connection 0's average share of the throughput when congestion is detected is p, and (2) for each packet drop at a gateway, a connection 0 packet is dropped with probability p. As a result, connection i's average share of the throughput at gateway ia when congestion is detected is $1 - p$, for $1 \leq i \leq n$. Thus, we analyze the network *as if* each time that gateway ia detects congestion, for $1 \leq i \leq n$, the gateway drops a connection 0 packet with probability p, and drops a connection i packet with probability $1 - p$.

Assumption #2 is a clearly unrealistic assumption, but it simplifies the analysis a great deal. In the initial model, connection 0's *average* probability of having a packet dropped after congestion is detected at a gateway is p, but connection 0's share of the throughput could be quite different from one packet drop to the next. Therefore, in the initial model connection 0's probability of having a packet dropped also could be quite different from one packet drop to the next. □

Claim 1 *Given the multiple gateways model with n congested gateways, and the two simplifying assumptions described above, connection 0's average throughput when congestion is detected at a gateway is $M * p$ pkts/sec for*

$$p = \frac{1}{1 + \sqrt{n}}.$$

Proof: From the two assumptions above, the number of packet drops for connection i is a Poisson process with rate $(1 - p) * \lambda$, for $1 \leq i \leq n$ [R83, p.38]. Thus for connection i the expected number of packet drops in 1 second is $(1 - p) * \lambda$, and the expected time between packet drops is $1/((1 - p) * \lambda)$ seconds. The number of packet drops for connection 0 is the sum of n independent Poisson processes, each with rate $p * \lambda$. This is a Poisson process with rate $n * p * \lambda$ [R83, p.51]. Thus for connection 0 the expected number of packet drops in 1 second is $n*p*\lambda$. The expected time between packet drops is $1/(n * p * \lambda)$ seconds.

From the second simplifying assumption, connection i's average throughput when a connection i packet is dropped is $M * (1 - p)$ pkts/sec, for $1 \leq i \leq n$. Because connection i reduces its window by half after a packet drop, connection i's average throughput after each packet drop is $M * (1 - p)/2$ pkts/sec. Connection i's throughput rate is therefore increased by an average of $M * (1 - p)/2$ pkts/sec between successive connection i packet drops. Similarly, connection 0's throughput rate is increased by an average of $M * p/2$ pkts/sec between successive connection 0 packet drops.

Therefore, in an average time of $1/((1 - p) * \lambda)$ seconds, connection i increases its throughput by an average of $M * (1 - p)/2$ pkts/sec, for $i > 0$. Similarly, in an average time of $1/(n * p * \lambda)$ seconds, connection 0 increases its throughput by an average of $M * p/2$ pkts/sec. From Lemma 8 in Appendix B, connection i's rate of throughput increase is

$$\frac{M * (1 - p)/2}{1/((1 - p) * \lambda)}$$

pkts/sec/sec, and connection 0's rate of throughput increase is

$$\frac{M * p/2}{1/(n * p * \lambda)}$$

pkts/sec/sec.

In our model, each connection increases its throughput at the same rate, in the absence on congestion. Therefore,

$$\frac{M * (1 - p)/2}{1/((1 - p) * \lambda)} = \frac{M * p/2}{1/(n * p * \lambda)},$$

$$(1 - p)^2 = n * p^2,$$

and

$$p = \frac{1}{1+\sqrt{n}}.$$

Thus connection 0's average throughput when a gateway detects congestion is $M/(1+\sqrt{n})$ pkts/sec. □

Figure 17 shows the results of this analysis. The x-axis shows the number n of congested gateways, and the y-axis shows the average throughput for connections 0 and 1 when congestion is detected at a gateway. The analytical results in Figure 17 are shown again by the solid line in Figure 6. In Figure 6 for each value of n the average total throughput has been calculated, and the analytical results are used to calculate connection 0's and connection 1's share of this throughput. The analytical results in Figure 6 and in Figure 17 differ only because the average total throughput in Figure 6 is less than 100%. As Figure 6 shows, the simulations and the analytical results are in close agreement.

This analysis used several simplifying assumptions described above. Nevertheless, the analysis gives some insight into the behavior of a network with multiple congested gateways, and the results of the analysis are in agreement with the simulations.

5.2 One congested gateway, different roundtrip times

In this section we examine the throughput in a network with two connections with different roundtrip times and one shared gateway, as in Figure 13. We consider the effect of different window increase algorithms on throughput.

Definitions: window increase algorithms, with r_i, $g(r_i)$. Let connection i's average roundtrip time including queueing delays be r_i seconds for $i \in \{0, 1\}$, and let $r_0 \geq r_1$. Let connection i increase its window by $g(r_i)$ packets each roundtrip time, where $g(r_i)$ is a function of the average roundtrip time r_i. Thus connection i increases its throughput rate by roughly $g(r_i)/r_i$ pkts/sec each r_i seconds, or by $g(r_i)/(r_i)^2$ pkts/sec/sec. For our analysis we assume that the roundtrip time is a constant r_i seconds, and that connection i increases its throughput rate by the constant rate of $g(r_i)/(r_i)^2$ pkts/sec/sec. □

The single gateway model: The single gateway model describing the network in Figure 13 is similar to the multiple gateway model in Section 5.1, except that it is no longer assumed that all connections increase their throughput rate by the same number of pkts/sec/sec. □

Simplifying assumptions: We use the same simplifying assumptions as in Section 5.1 above. Thus we analyze the network as if each time the gateway drops a packet, the gateway drops a connection 0 packet with probability p, and the gateway drops a connection 1 packet with probability $1-p$, where p is connection 0's average throughput at the gateway when congestion is detected. □

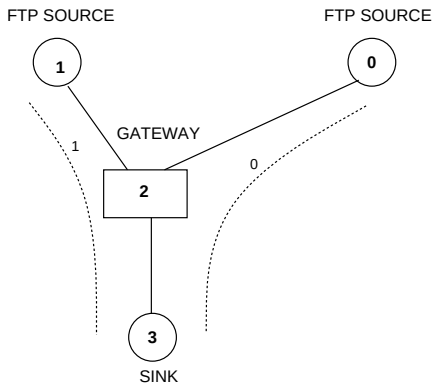

Figure 13: Simulation network with one congested gateway.

Claim 2 *Given the single gateway model, and the two simplifying assumptions, connection 0's average throughput when congestion is detected at a gateway is $M * p$ pkts/sec for*

$$p = \frac{1}{1+\frac{r_0}{r_1} * \sqrt{\frac{g(r_1)}{g(r_0)}}}.$$

Proof: From these two assumptions, the number of packet drops for connection 0 is approximated by a Poisson process with rate $p * \lambda$. Thus the expected number of packet drops for connection 0 in 1 second is $p * \lambda$, and the expected time between packet drops is $1/(p*\lambda)$ seconds. Similarly, for connection 1 the expected number of packet drops in 1 second is $(1-p) * \lambda$, and the expected time between packet drops is $1/((1-p) * \lambda)$ seconds.

From the second assumption, connection 0's average throughput through the gateway when a packet is dropped is $M*p$, and connection 1's average throughput is $M * (1-p)$. Each connection reduces its throughput by half when a packet from that connection is dropped. Thus connection 0's throughput is increased by an average of $M * p/2$ pkts/sec between successive connection 0 packet drops, and connection 1's throughput is increased by an average of $M * (1-p)/2$ pkts/sec between successive connection 1 packet drops.

Therefore, in an average time of $1/(p * \lambda)$ seconds, connection 0 increases its throughput by an average of $M * p/2$ pkts/sec. Connection 0's rate of throughput increase is

$$\frac{M * p/2}{1/(p * \lambda)} = \frac{M * \lambda * p^2}{2}$$

pkts/sec/sec. Similarly, connection 1's rate of throughput increase is

$$\frac{M * (1-p)/2}{1/((1-p) * \lambda)} = \frac{M * \lambda * (1-p)^2}{2}$$

pkts/sec/sec.

In our model, connection 0 is defined to increase its throughput rate by $g(r_0)/(r_0)^2$ pkts/sec/sec. Therefore,

$$\frac{g(r_0)}{(r_0)^2} = \frac{M * \lambda * p^2}{2}. \quad (1)$$

Similarly,

$$\frac{g(r_1)}{(r_1)^2} = \frac{M * \lambda * (1-p)^2}{2}. \quad (2)$$

From equations (1) and (2),

$$\left(\frac{r_1}{r_0}\right)^2 \frac{g(r_0)}{g(r_1)} = \left(\frac{p}{1-p}\right)^2,$$

or

$$p = \frac{1}{1 + \frac{r_0}{r_1} * \sqrt{\frac{g(r_1)}{g(r_0)}}}.$$

□

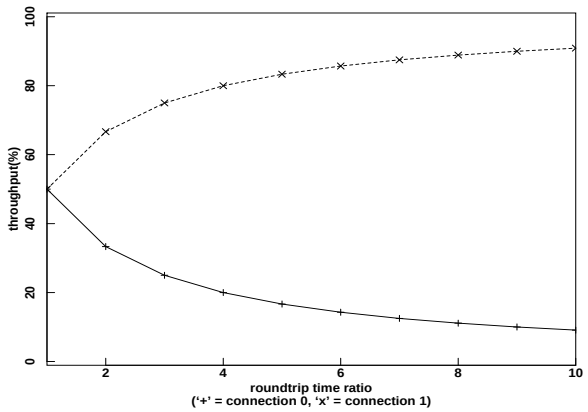

Figure 14: Analysis of throughput vs. roundtrip time ratio with one congested gateway, Increase-by-One increases, Reduce-by-Half decreases.

Corollary 3 *If $g(r_i) = 1$ as in the Increase-by-One algorithm, then connection 0's average throughput when the gateway detects congestion is $M * p$ pkts/sec for*

$$p = \frac{1}{1 + \frac{r_0}{r_1}}.$$

The two lines in Figure 14 show the throughput predicted by this analysis for connection 0 and connection 1. The x-axis gives the ratio between connection 0's roundtrip time and connection 1's roundtrip time.

Corollary 4 *If $g(r_i) = a * r_i^2$ for some constant a as in the Constant-Rate algorithm, then both connection 0 and connection 1's average throughput when the gateway detects congestion is $M/2$ pkts/sec, regardless of the roundtrip times of the two connections.*

Both of these corollaries concur with the results of simulations in [FJ91a]. (The simulations with the Increase-by-One algorithm in [FJ91a] have smaller maximum windows, and this results in a somewhat improved performance for the longer connection.) As discussed in [FJ91a], the results of Corollary 4 are consistent with the analysis in [CJ89] of window increase and decrease algorithms.

5.3 Multiple congested gateways and an unspecified window-increase algorithm

The second multiple gateways model: In this section we consider a family of networks with n multiple congested gateways, as in Figure 1 and in Section 5.1, where it is no longer assumed that each connection increases its throughput at the same rate. As in Section 5.2, assume that connection i increases its window by $g(r_i)$ packets each roundtrip time. Thus, connection i increases its throughput rate by $g(r_i)/(r_i)^2$ pkts/sec/sec. In this section we merge the analysis from the previous two sections.

Claim 5 *Given the second multiple gateways model with n congested gateways, and the two simplifying assumptions described earlier, connection 0's average throughput when congestion is detected at a gateway is $M * p$ pkts/sec for*

$$p = \frac{1}{1 + \sqrt{n} * \frac{r_0}{r_1} * \sqrt{\frac{g(r_1)}{g(r_0)}}}. \quad (3)$$

Proof: From Section 5.1, connection 0 increases its throughput rate by

$$\frac{M * p/2}{1/(n * p * \lambda)}$$

pkts/sec/sec, and connection 1 increases its throughput rate by

$$\frac{M * (1-p)/2}{1/((1-p) * \lambda)}$$

pkts/sec/sec.

Therefore

$$\frac{M * p/2}{1/(n * p * \lambda)} = \frac{g(r_0)}{(r_0)^2} \quad (4)$$

and

$$\frac{M * (1-p)/2}{1/((1-p) * \lambda)} = \frac{g(r_1)}{(r_1)^2}. \quad (5)$$

From equations (4) and (5),

$$\frac{n * p^2}{(1-p)^2} = \frac{g(r_0)}{g(r_1)} * \left(\frac{r_1}{r_0}\right)^2.$$

Then

$$p = \frac{1}{1 + \sqrt{n} * \frac{r_0}{r_1} * \sqrt{\frac{g(r_1)}{g(r_0)}}}.$$

□

Corollary 6 *For $g(r_0) = g(r_1)$ as in the Increase-by-One algorithm, and for $r_0/r_1 = 2n-1$ as in the network family in Figure 1, connection 0's average throughput when congestion is detected is $M * p$ pkts/sec for*

$$p = \frac{1}{1 + \sqrt{n} * (2n-1)}.$$

Figure 15 shows this result, where the x-axis shows the number of congested gateways. The analytical results in Figure 15 are shown again by the solid line in Figure 3. As Figure 3 shows, the simulations and the analytical results are in reasonable agreement.

Figure 17 with Constant-Rate increases shows a fairly modest decrease in throughput for connection 0 as the number of congested gateways increases. Figure 15 however, with Increase-by-One increases, shows poor throughput for connection 0 even for a small number of congested gateways. Thus this analysis shows that the combination of multiple congested gateways and the Increase-by-One window increase algorithm has a strong negative effect on the throughput of the longer connection.

The results in this section are valid for any window-decrease algorithm where the window is decreased by a multiplicative factor.

Corollary 7 considers a network similar to our simulation networks with Random Drop and Drop Tail gateways, where the average ratio between connection 0's share of the packet drops and connection 0's share of the throughput is some function $f(p)$, for p connection 0's share of the throughput. This allows us to model a network with gateways with a bias against bursty traffic.

Corollary 7 *Given the second multiple gateways model with n congested gateways and the simplifying assumptions described earlier add the following modification to the second simplifying assumption. For the heuristic analysis, analyze the network as if connection 0's average share of the throughput when congestion is detected is p, and for each packet drop at a gateway, a connection 0 packet is dropped with probability $f(p) * p$, for $f(p)$ some function of p. Then connection 0's average throughput when congestion is detected at a gateway is $M * p$ pkts/sec, for that p for $0 \leq p \leq 1$ such that*

$$\frac{n * p * f(p) * p}{(1-p)(1 - f(p) * p)} = \frac{g(r_0)}{g(r_1)} * \left(\frac{r_1}{r_0}\right)^2.$$

Proof: The proof is similar to the proof of Claim 5 above, and the details are omitted. □

Figure 18 gives the results of the heuristic analysis in Corollary 7 above for connection 0's throughput in a network with Constant-Rate increases, Reduce-by-Half decreases, and a bias against bursty traffic. As shown in Section 4.4 for our simulations with Drop Tail gateways, we let $f(p) = 0.59 + 5 * e^{-5*p}$ reflect the bias against bursty traffic of Drop Tail gateways for $0 \leq p \leq 0.5$. This models a network where, as a connection's share of the throughput decreases, the burstiness of that connection increases. The heuristic analysis in Figure 18 is shown again by the solid line in Figures 7 and 8. Numerical methods are used to obtain the result in Figure 18 from the equation in Corollary 7. Because the equation $f(p) = 0.59 + 5 * e^{-5*p}$ overstates the bias against bursty traffic of Random Drop and Drop Tail gateways for $p \geq 1/4$, the simulations and the analytical results are somewhat far apart. A more accurate function $f(p)$ for the bias against bursty traffic would give a more accurate heuristic analysis.

Figure 16 gives the results of the heuristic analysis in Corollary 7 above for connection 0's throughput with Increase-by-One increases, Reduce-by-Half decreases, and gateways with the bias against bursty traffic described above. The analysis in Figure 16 is shown again by the solid lines in Figures 4 and 5. Again, there is some discrepancy between the simulation results and the heuristic analysis, in part due to the lack of accuracy in the model for the bias against bursty traffic. The simulations use the Reduce-to-One algorithm, and the analysis in Figure 16 assumes the Reduce-by-Half algorithm.

Increase-by-One window-increase algorithm:

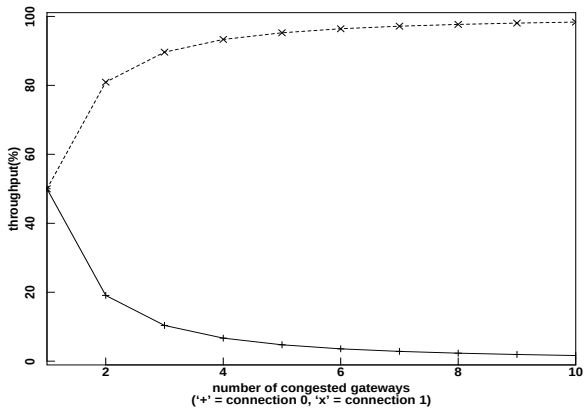

Figure 15: Analysis of throughput with Reduce-by-Half decreases.

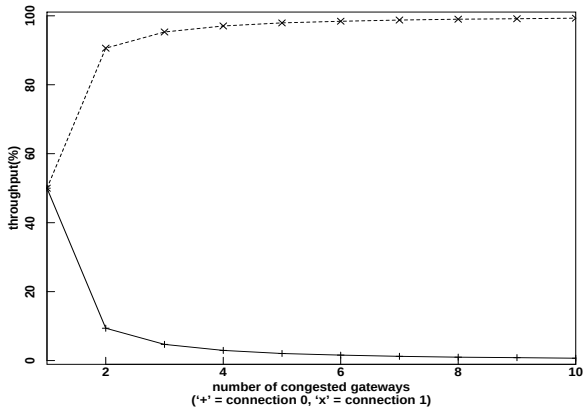

Figure 16: Analysis of throughput with Reduce-by-Half decreases and a bias against bursty traffic.

Constant-Rate window-increase algorithm:

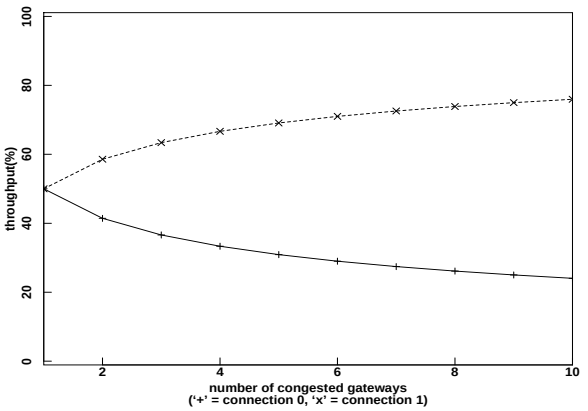

Figure 17: Analysis of throughput with Reduce-by-Half decreases.

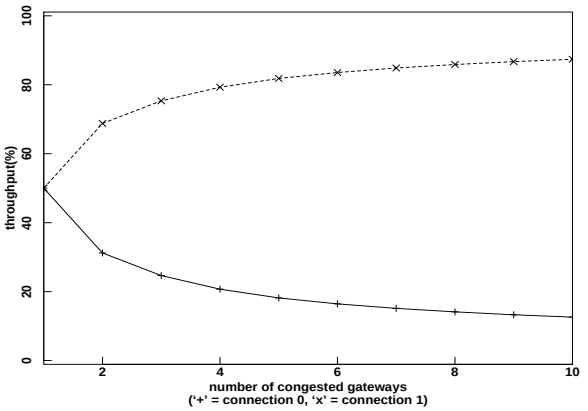

Figure 18: Analysis of throughput with Reduce-by-Half decreases and a bias against bursty traffic.

6 Fairness considerations

In previous sections, we used simulations and analysis to consider the performance of a simple network with multiple congested gateways. In this section we discuss the issue of fairness for such a network.

Some general network goals are to maximize throughput, minimize delay, and ensure fairness. Given a network with an upper bound on the average delay at each gateway, as in a network with RED gateways, these general goals can be reduced to the two goals of maximizing throughput and maintaining fairness. *System throughput* is measured as the total number of packets sent for all users divided by the total time. For the network in Figure 1, the system throughput is maximized when connection 0 receives zero throughput, but this is not very fair. Clearly the goals of maximizing throughput and maintaining fairness must be considered together.

There is no generally-agreed-upon definition for fairness in a computer network. Given two connections, one passing through n congested gateways and the other passing through one congested gateway, the connection with multiple congested gateways uses more of the network's scarce contested resources for each packet. Should each connection get the same share of the network's scarce resources? If so, then the connection with multiple congested gateways should get $1/n$ times the throughput of the connection with one congested gateway. On the other side, should each connection get the same throughput in pkts/sec?

We consider three different fairness measures: min-max fairness [HG86], the fairness index proposed in [JCH84], and the product measure, a variant of network power [BJ81]. Consider the throughput rate for each connection.

To satisfy the min-max fairness criteria, the smallest throughput rate must be as large as possible. Given this condition, the next-smallest throughput rate must be as large as possible, and so on. For the network in Figure 1, this min-max fairness criteria is satisfied when all $n+1$ connections receive the same throughput rate. As shown in [DKS90], this is achieved for a scenario similar to that in this paper by the use of the Fair Queueing gateway algorithm. [DKS90] also shows that the min-max fairness criteria is achieved by the selective DECbit algorithm [RCJ87].

The fairness index in [JCH84] is

$$F_1 = \frac{(\sum_{i=1}^{n} x_i)^2}{n(\sum_{i=1}^{n} x_i^2)},$$

where x_i is the resource allocation to the ith user. This fairness index ranges from 0 to 1, and is maximum when all users receive the same allocation. This index is k/n when k users equally share the resource, and the other $n-k$ users receive zero allocation. Examples of possible definitions of resource allocation include response time, throughput, throughput times hops, fraction of demand, and so on [JCH84].

If the resource allocation is defined as throughput, for a network in which one connection receives throughput p and the other n connections receive throughput $1-p$, this fairness index gives

$$F_1 = \frac{(p + n*(1-p))^2}{(n+1)*(p^2 + n*(1-p)^2)}.$$

This is maximized for $p = 1/2$, when all connections receive the same throughput rate.

If the resource allocation is defined as throughput times the number of gateways, then $x_0 = 2n*p$ and $x_i = 2*(1-p)$. The fairness index gives

$$\frac{1}{1 - 2*p + p^2 + n*p^2}.$$

As expected, this is maximized for $p = 1/(1+n)$, where each connection gets the same share of the contested resources.

The product measure

$$F_2 = \prod_{i=1}^{n} x_i,$$

the product of the throughput of the individual connections, is also used as a measure of fairness. For our purposes, let x_i be the throughput for the ith connection. (In other contexts x_i is taken as the power of the ith connection, and the product measure is referred to as *network power*.) The product measure is particularly sensitive to segregation; the product measure is zero if any connection receives zero throughput. In [MS90, p.15] it is shown that for a network with many connections and one shared gateway, the product measure is maximized when all connections receive the same throughput.

For a network in which one connection receives throughput p, and the other n connections receive throughput $1-p$, the product measure

$$F_2 = p(1-p)^n$$

is maximized for

$$(1-p)^n - n*(1-p)^{n-1}*p = 0,$$

or for $p = 1/(n+1)$. Thus for the network in Figure 1, the product measure is maximized when connection 0's throughput is $1/(n+1)$ of the maximum possible throughput. This is also connection 0's throughput

when each connection receives an equal share of the network's resources (where each congested gateway is considered as a separate resource).

The different prevailing measures of fairness shed little light on the appropriate throughput allocations for a network with multiple congested gateways. It is not sufficient to arbitrarily accept one of the prevailing definition of fairness simply because it is precise and easily measured. The min-max criterion and the fairness index based on throughput are maximized when connection 0 receives half of the maximum throughput. The product measure and the fairness index based on throughput times gateways, on the other hand, are maximized when connection 0 receives $1/(n+1)$ of the maximum throughput. For a network in which each connection increases its throughput at the same rate, simulations and analysis show that the throughput for connection 0 is less than the throughput that satisfies the min-max criterion, but greater than the throughput that maximizes the product measure. Therefore our simulations and analysis suggest that for a network with Constant-Rate increases, the throughput allocations for Figure 1 are within an acceptable range of fairness, in the absence of policy decisions on fairness in TCP/IP networks.

However, for the family of networks as in Figure 1 with the Increase-by-One algorithm, the network does not give fair performance by any of the accepted measures of fairness.

This paper shows that for the network family in Figure 1, the TCP window modification algorithms could be chosen to accommodate a wide range of fairness goals. Claim 5 gives a heuristic analysis for connection 0's throughput for this network. From this analysis, when each connection i increases its window by $g(r_i)$ packets each roundtrip time, connection 0's average share of the throughput is

$$p = \frac{1}{1 + \sqrt{n} * \frac{r_0}{r_1} * \sqrt{\frac{g(r_1)}{g(r_0)}}}. \qquad (6)$$

Thus the network modeled in this paper could accommodate a wide range of fairness goals, depending on the choice of the window increase algorithm represented by the function $g(r_i)$.

7 Conclusions and Future Work

The conclusions from the simulations and analysis in this paper are that for connections with long roundtrip times and multiple congested gateways, the use of the Increase-by-One window-increase algorithm in 4.3 tahoe BSD TCP and of Drop Tail or Random Drop gateways can result in unacceptably low throughput for a long connection with multiple congested gateways. As discussed in Section 3, this low throughput for connections with multiple congested gateways is not only a problem for networks characterized by long bulk-data transfers. However, our simulations and our analysis show that this low throughput for longer connections with multiple congested gateways is not unavoidable. With a window-increase algorithm such as the Constant-Rate window-increase algorithm, for example, it is possible for connections passing through multiple congested gateways to achieve a reasonable level of throughput.

As discussed in Section 6, there are several conflicting definitions of fairness in computer networks. Some of these definitions imply that a connection with multiple congested gateways should receive the same throughput as a connection with one congested gateways. Other definitions imply that a connection with n congested gateways should receive $1/n$th the throughput of a connection with one congested gateway. With this allocation, each connection receives the same share of the network resources, if each congested gateway is considered as a separate resource. In the absence of a policy decision on the desired fairness goal in current networks, we have no reason to consider a throughput allocation unfair that gives each connection the same share of the network resources.

It is generally agreed that segregation is to be avoided in current networks whenever possible. Our simulations with Random Drop or Drop Tail gateways and with the window modification algorithms in 4.3 tahoe BSD TCP essentially result in segregation against a longer connection with multiple congested gateways. We have shown that this segregation could be corrected by the use of the Reduce-to-Half window-decrease algorithm, gateways with no bias against bursty traffic, and a window-increase algorithm with no bias against connections with longer roundtrip times.

Our interest in the question of throughput allocation in networks with multiple congested gateways was initially sparked by the comparison in [M90b] of throughput in networks with multiple congested gateways with Random Drop and with Drop Tail gateways. We have shown in [FJ91a] that in networks with Drop Tail gateways, throughput can be determined partly by network phase effects, and this throughput can change dramatically with small changes in the network parameters. In this paper we show that when a sufficient random component is added to the roundtrip times in simulations using Drop-Tail gateways, reducing the traffic phase effects, then the throughput allocation is quite similar for Random Drop and for Drop Tail gateways for the network examined in this paper.

As we will show in a future paper (in progress), for

simulations with two-way traffic the throughput for a connection with multiple congested gateways is much higher in a network with RED gateways that it is in a network with Random Drop or Drop Tail gateways. For simulations with two-way traffic the compressed ACK packets cause bursty traffic, with a resulting loss of throughput for the longer connections. This dynamic does not occur in simulations with two-way traffic using RED gateways because of the ability of RED gateways to accommodate bursty traffic.

There are many open questions. The research in this paper is intended simply to explore the factors that affect throughput in networks with multiple congested gateways. We are specifically *not* proposing Constant-Rate window increase algorithms for current networks. We are currently investigating various alternatives to the current TCP Increase-By-One window increase algorithm. We explore RED gateways in more detail in a paper currently in progress [FJ91c].

The main result is simply that there is no *inherent* reason for long connections passing through multiple congested gateways in TCP/IP networks to receive unacceptably low throughput; this low throughput could be corrected by removing the network biases against connections with longer roundtrip times and against bursty traffic. We believe that more work is needed on fairness goals in networks and on the implications of fairness goals for other areas of network performance.

8 Acknowledgements

Several of the ideas in this paper resulted from joint work with Van Jacobson. I thank Scott Shenker, and Lixia Zhang for useful discussions on these issues. This work would have been impossible without Steven McCanne, who has extensively rewritten our version of the REAL simulator.

References

[BDSY88] Bacon, D,. Dupuy, A., Schwartz, J., and Yemimi, Y., "Nest: a Network Simulation and Prototyping Tool", *Proceedings of Winter 1988 Usenix Conference*, 1988, pp. 17-78.

[BJ81] Bharath-Kumar, K., and Jeffrey, J., *A New Approach to Performance-Oriented Flow Control*, IEEE Transactions on Communications, Vol.COM-29 N.4, April 1981.

[CDJM91] Cáceres, R., Danzig, P., Jamin, S., and Mitzel, D., "Characteristics of Wide-Area TCP/IP Conversations", to appear in SIGCOMM '91.

[CJ89] Chiu, D.-M., and Jain, R., "Analysis of the Increase and Decrease Algorithms for Congestion Avoidance in Computer Networks", Computer Networks and ISDN Systems, V. 17, pp. 1-14, 1989.

[DKS90] Demers, A., Keshav, S., and Shenker, S., "Analysis and Simulation of a Fair Queueing Algorithm", Internetworking: Research and Experience, Vol. 1, 1990, p. 3-26.

[FJ91a] Floyd, S., and Jacobson, V., *Traffic Phase Effects in Packet-Switched Gateways*, Computer Communication Review, V.21 N.2, April 1991.

[FJ91b] Floyd, S., and Jacobson, V., *On Traffic Phase Effects in Packet-Switched Gateways*, in preparation. (This is a longer version of the paper above.)

[FJ91c] Floyd, S., and Jacobson, V., *Random Early Detection gateways for congestion avoidance*, in preparation.

[H89] Hashem, E., "Analysis of random drop for gateway congestion control", *Report LCS TR-465*, Laboratory for Computer Science, MIT, Cambridge, MA, 1989.

[HG86] Hahne, E., and Gallager, R., *Round Robin Scheduling for Fair Flow Control in Data Communications Networks*, IEEE International Conference on Communications, June, 1986.

[J88] Jacobson, V., *Congestion Avoidance and Control*, Proceedings of SIGCOMM '88, August 1988.

[JB88] Jacobson, V., and Braden, R., "RFC 1072: TCP Extensions for Long-Delay Paths", October 1988.

[J90] Jacobson, V., *Berkeley TCP Evolution from 4.3-tahoe to 4.3-reno*, "Proceedings of the Eighteenth Internet Engineering Task Force", Vancouver, British Columbia, August, 1990.

[JCH84] Jain, R., Chiu, D.M., and Hawe, W., "A Quantitative Measure of Fairness and Discrimination for Resource Allocation in Shared Systems." DEC TR-301, Littleton, MA: Digital Equipment Corporation.

[K88] Keshav, S., "REAL: a Network Simulator", *Report 88/472*, Computer Science Department, University of California at Berkeley, Berkeley, California, 1988.

[M90a] Mankin, A., "A Measurement Study of Random Drop and Gateway Interaction", 1990.

[M90b] Mankin, A., *Random Drop Congestion Control*, Proceedings of SIGCOMM 90, Sept. 1990.

[MS90] Mitra, D. and Seery, J., *Dynamic Adaptive Windows for High Speed Data Networks: Theory and Simulations*, AT&T Bell Laboratories report, April 1990.

[RCJ87] Ramakrishnan, K.K., Chiu, D., and Jain, R., "Congestion Avoidance in Computer Networks with a Connectionless Network Layer, Part IV: A Selective Binary Feedback Scheme for General Topologies", DEC-TR-510, November, 1987.

[RJ90] Ramakrishnan, K.K., and Jain, Raj, "A Binary Feedback Scheme for Congestion Avoidance in Computer Networks", ACM Transactions on Computer Systems, V. 8, N. 2, p. 158-181, 1990.

[R83] Ross, S., Stochastic Processes, John Wiley & Sons, 1983.

[WRM91] Wilder, R., Ramakrishnan, K.K., and Mankin, A., "Dynamics of Congestion Control and Avoidance in Two-Way Traffic in an OSI Testbed", Computer Communication Review, V.21 N.2, April 1991, p.43-58.

[ZC91] Zhang, L., and Clark, D., "Observations on the Dynamics of a Congestion Control Algorithm: The Effects of Two-Way Traffic", to appear in SIGCOMM '91.

[Z89] Zhang, L., "A New Architecture for Packet Switching Network Protocols", MIT LCS TR-455, Laboratory for Computer Science, Massachusetts Institute of Technology, August 1989.

A The Reduce-to-One window decrease algorithm

In this appendix we briefly explain why longer-roundtrip-time traffic receives higher throughput with the Reduce-by-Half window decrease algorithm that with the Reduce-to-One algorithm. Let connection 1 have roundtrip time r_1, and let connection 0 have roundtrip time $r_0 = m * r_1$, for $m \geq 1$. Let connection 1's window be w_1 packets when a connection 1 packet is dropped. With the Reduce-to-One window decrease algorithm, connection 1's window is reduced to 1 packet, and then connection 1's current window is doubled each roundtrip time until it reaches $w_1/2$, at which point the congestion-avoidance window-increase algorithm is used. Connection 1 requires roughly

$$r_1 * \log_2(w_1/2)$$

seconds to increase its window back to $w_1/2$.

If connection 0 and connection 1 have the same throughput rate and connection 1's window is w_1, then connection 0's window is $w_0 = m * w_1$. If a connection 0 packet is dropped when connection 0's window is $m * w_1$, then connection 0 requires

$$r_0 * \log_2(w_0/2) = m * r_1 * (\log_2(w_1/2) + \log_2 m)$$

seconds to increase its window back to $m * w_1/2$. Thus, if connection 0's roundtrip time is m times longer that connection 1's roundtrip time, and connection 0 and connection 1 have the same throughput rate, then it takes connection 0 more that m times longer to increase its window to half of its old value.

B A short lemma

This Lemma is used in the proof of Claim 1. A connection increases its throughput by an average of R pkts/sec in one window-increase cycle, and the average window-increase cycle is T seconds. Further, the connection is known to increase its throughput rate by a fixed value of D pkts/sec/sec, for some D. Lemma 8 concludes that $D = R/T$.

Lemma 8 *Let some connection increase its throughput rate by a fixed value of D pkts/sec/sec. Let the connection increase its throughput by an average of R pkts/sec in one window-increase cycle interval, and the average window-increase cycle is T seconds. Then $D = R/T$.*

Proof: Let $f(x)$ be the density function for the length of a window-increase cycle. Then

$$T = \int_0^\infty x f(x) dx.$$

The average throughput increase in pkts/sec per window-increase cycle is

$$R = \int_0^\infty x * D * f(x) dx = D * T.$$

□