Figure 3 shows a simple simulation with RED gateways. The network is shown in Figure 4. The simulation contains four FTP connections, each with a maximum window roughly equal to the delay-bandwidth product, which ranges from 33 to 112 packets. The RED gateway parameters are set as follows: $w_q = 0.002$, $min_{th} = 5$ packets, $max_{th} = 15$ packets, and $max_p = 1/50$. The buffer size is sufficiently large that packets are never dropped at the gateway due to buffer overflow; in this simulation the RED gateway controls the average queue size, and the actual queue size never exceeds forty packets.

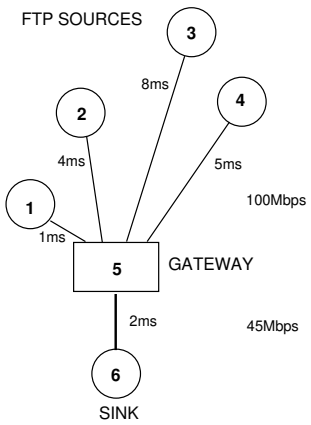

Figure 4: Simulation network.

For the charts in Figure 3, the x-axis shows the time in seconds. The bottom chart shows the packets from nodes 1-4. Each of the four main rows shows the packets from one of the four connections; the bottom row shows node 1 packets, and the top row shows node 4 packets. There is a mark for each data packet as it arrives at the gateway and as it departs from the gateway; at this time scale, the two marks are often indistinguishable. The y-axis is a function of the packet sequence number; for packet number n from node i, the y-axis shows $n \bmod 90 + (i-1)100$. Thus, each vertical 'line' represents 90 consecutively-numbered packets from one connection arriving at the gateway. Each 'X' shows a packet dropped by the gateway, and each 'X' is followed by a mark showing the retransmitted packet. Node 1 starts sending packets at time 0, node 2 starts after 0.2 seconds, node 3 starts after 0.4 seconds, and node 4 starts after 0.6 seconds.

The top chart of Figure 3 shows the instantaneous queue size q and the calculated average queue size avg. The dotted lines show min_{th} and max_{th}, the minimum and maximum thresholds for the average queue size. Note that the calculated average queue size avg changes fairly slowly compared to q. The bottom row of X's on the bottom chart shows again the time of each dropped packet.

This simulation shows the success of the RED gateway in controlling the average queue size at the gateway in response to a dynamically changing load. As the number of connections increases, the frequency with which the gateway drops packets also increases. There is no global synchronization. The higher throughput for the connections with shorter roundtrip times is due to the bias of TCP's window increase algorithm in favor of connections with shorter roundtrip times (as discussed in [6, 7]). For the simulation in Figure 3 the average link utilization is 76%. For the following second of the simulation, when all four sources are active, the average link utilization is 82%. (This is not shown in Figure 3.)

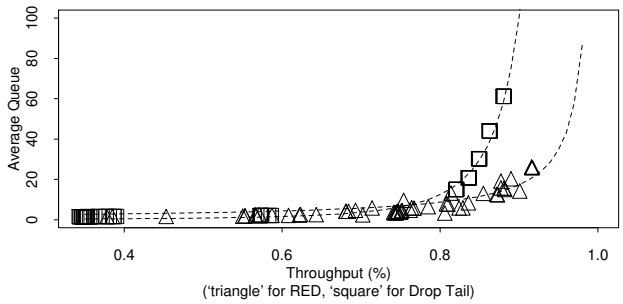

Figure 5: Comparing Drop Tail and RED gateways.

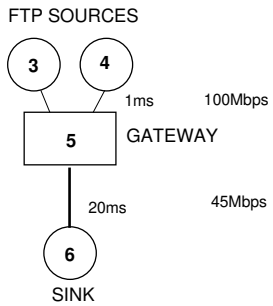

Figure 6: Simulation network.

Because RED gateways can control the average queue size while accommodating transient congestion, RED gateways are well-suited to provide high throughput and low *average* delay in high-speed networks with TCP connections that have large windows. The RED gateway can accommodate the short burst in the queue required by TCP's slow-start phase; thus RED gateways control the *average* queue size while still allowing TCP connections to smoothly open their windows. Figure 5 shows the results of simulations of the network in Figure 6 with

two TCP connections, each with a maximum window of 240 packets, roughly equal to the delay-bandwidth product. The two connections are started at slightly different times. The simulations compare the performance of Drop Tail and of RED gateways.

In Figure 5 the x-axis shows the total throughput as a fraction of the maximum possible throughput on the congested link. The y-axis shows the average queue size in packets (as seen by arriving packets). Five 5-second simulations were run for each of 11 sets of parameters for Drop Tail gateways, and for 11 sets of parameters for RED gateways; each mark in Figure 5 shows the results of one of these five-second simulations. The simulations with Drop Tail gateways were run with the buffer size ranging from 15 to 140 packets; as the buffer size is increased, the throughput and the average queue size increase correspondingly. In order to avoid phase effects in the simulations with Drop Tail gateways, the source node takes a random time drawn from the uniform distribution on [0, t] seconds to prepare an FTP packet for transmission, where t is the bottleneck service time of 0.17 ms. [7].

The simulations with RED gateways were all run with a buffer size of 100 packets, with min_{th} ranging from 3 to 50 packets. For the RED gateways, max_{th} is set to $3\,min_{th}$, with $w_q = 0.002$ and $max_p = 1/50$. The dashed lines show the average delay (as a function of throughput) approximated by $1.73/(1-x)$ for the simulations with RED gateways, and approximated by $0.1/(1-x)^3$ for the simulations with Drop Tail gateways. For this simple network with TCP connections with large windows, the network power (the ratio of throughput to delay) is higher with RED gateways than with Drop Tail gateways. There are several reasons for this difference. With Drop Tail gateways with a small maximum queue, the queue drops packets while the TCP connection is in the slow-start phase of rapidly increasing its window, reducing throughput. On the other hand, with Drop Tail gateways with a large maximum queue the average delay is unacceptably large. In addition, Drop Tail gateways are more likely to drop packets from both connections at the same time, resulting in global synchronization and a further loss of throughput.

Later in the paper, we discuss simulation results from networks with a more diverse range of connections. The RED gateway is not specifically designed for a network dominated by bulk data transfer; this is simply an easy way to simulate increasingly-heavy congestion at a gateway.

6 Calculating the average queue length

The RED gateway uses a low-pass filter to calculate the average queue size. Thus, the short-term increases in the queue size that result from bursty traffic or from transient congestion do not result in a significant increase in the average queue size.

The low-pass filter is an exponential weighted moving average (EWMA):

$$avg \leftarrow (1 - w_q)avg + w_q\, q. \tag{1}$$

The weight w_q determines the time constant of the low-pass filter. The following sections discuss upper and lower bounds for setting w_q. The calculation of the average queue size can be implemented particularly efficiently when w_q is a (negative) power of two, as shown in Section 11.

6.1 An upper bound for w_q

If w_q is too large, then the averaging procedure will not filter out transient congestion at the gateway.

Assume that the queue is initially empty, with an average queue size of zero, and then the queue increases from 0 to L packets over L packet arrivals. After the Lth packet arrives at the gateway, the average queue size avg_L is

$$\begin{aligned} avg_L &= \sum_{i=1}^{L} i\, w_q (1-w_q)^{L-i} \\ &= w_q (1-w_q)^L \sum_{i=1}^{L} i \left(\frac{1}{1-w_q}\right)^i \\ &= L + 1 + \frac{(1-w_q)^{L+1} - 1}{w_q}. \end{aligned} \tag{2}$$

This derivation uses the following identity [9, p. 65]:

$$\sum_{i=1}^{L} i x^i = \frac{x + (Lx - L - 1)\, x^{L+1}}{(1-x)^2}.$$

Figure 7 shows the average queue size avg_L for a range of values for w_q and L. The x-axis shows w_q from 0.001 to 0.005, and the y-axis shows L from 10 to 100. For example, for $w_q = 0.001$, after a queue increase from 0 to 100 packets, the average queue size avg_{100} is 4.88 packets.

Given a minimum threshold min_{th}, and given that we wish to allow bursts of L packets arriving at the gateway, then w_q should be chosen to satisfy the following equation for $avg_L < min_{th}$:

$$L + 1 + \frac{(1-w_q)^{L+1} - 1}{w_q} < min_{th}. \tag{3}$$

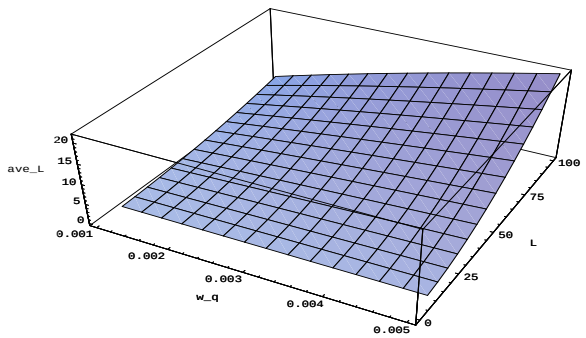

Figure 7: avg_L as a function of w_q and L.

Given $min_{th} = 5$, and $L = 50$, for example, it is necessary to choose $w_q \leq 0.0042$.

6.2 A lower bound for w_q

RED gateways are designed to keep the calculated average queue size avg below a certain threshold. However, this serves little purpose if the calculated average avg is not a reasonable reflection of the current average queue size. If w_q is set too low, then avg responds too slowly to changes in the actual queue size. In this case, the gateway is unable to detect the initial stages of congestion.

Assume that the queue changes from empty to one packet, and that, as packets arrive and depart at the same rate, the queue remains at one packet. Further assume that initially the average queue size was zero. In this case it takes $-1/\ln(1-w_q)$ packet arrivals (with the queue size remaining at one) until the average queue size avg reaches $0.63 = 1 - 1/e$ [35]. For $w_q = 0.001$, this takes 1000 packet arrivals; for $w_q = 0.002$, this takes 500 packet arrivals; for $w_q = 0.003$, this takes 333 packet arrivals. In most of our simulations we use $w_q = 0.002$.

6.3 Setting min_{th} and max_{th}

The optimal values for min_{th} and max_{th} depend on the desired average queue size. If the typical traffic is fairly bursty, then min_{th} must be correspondingly large to allow the link utilization to be maintained at an acceptably high level. For the typical traffic in our simulations, for connections with reasonably large delay-bandwidth products, a minimum threshold of one packet would result in unacceptably low link utilization. The discussion of the optimal average queue size for a particular traffic mix is left as a question for future research.

The optimal value for max_{th} depends in part on the maximum average delay that can be allowed by the gateway.

The RED gateway functions most effectively when $max_{th} - min_{th}$ is larger than the typical increase in the calculated average queue size in one roundtrip time. A useful rule-of-thumb is to set max_{th} to at least twice min_{th}.

7 Calculating the packet-marking probability

The initial packet-marking probability p_b is calculated as a linear function of the average queue size. In this section we compare two methods for calculating the final packet-marking probability, and demonstrate the advantages of the second method. In the first method, when the average queue size is constant the number of arriving packets between marked packets is a geometric random variable; in the second method the number of arriving packets between marked packets is a uniform random variable.

The initial packet-marking probability is computed as follows:

$$p_b \leftarrow max_p(avg - min_{th})/(max_{th} - min_{th}).$$

The parameter max_p gives the maximum value for the packet-marking probability p_b, achieved when the average queue size reaches the maximum threshold.

Method 1: Geometric random variables. In Method 1, let each packet be marked with probability p_b. Let the intermarking time X be the number of packets that arrive, after a marked packet, until the next packet is marked. Because each packet is marked with probability p_b,

$$Prob[X = n] = (1 - p_b)^{n-1} p_b.$$

Thus with Method 1, X is a *geometric* random variable with parameter p_b, and $E[X] = 1/p_b$.

With a constant average queue size, the goal is to mark packets at fairly regular intervals. It is undesirable to have too many marked packets close together, and it is also undesirable to have too long an interval between marked packets. Both of these events can result in global synchronization, with several connections reducing their windows at the same time, and both of these events can occur when X is a geometric random variable. □

Method 2: Uniform random variables. A more desirable alternative is for X to be a *uniform* random variable from $\{1, 2, ..., 1/p_b\}$ (assuming for simplicity that $1/p_b$ is an integer). This is achieved