Using 'Drop-Biasing' to Stabilize the Occupancy of Random-Drop Queues with TCP Traffic

Archan Misra archan@research.telcordia.com
Teunis Ott tjo@research.telcordia.com
John Baras baras@isr.umd.edu

Abstract— **The paper describes how the use of 'drop-biasing', a technique to control the *distribution* of the gap between consecutive packet losses in random drop queues (such as RED) can be used to reduce the variability of the queue occupancy with TCP traffic. Reducing the variance of the queue occupancy reduces delay jitter for buffered packets, as well as decreases the likelihood of buffer underflow. We find that modifying the packet drop probabilities to ensure a minimum separation between consecutive packet drops serves to decrease the variability in the queue occupancy. This is really achieved as a result of the *increased* negative correlation among the congestion windows of the constituent TCP flows. Such negative correlation explains why the use of simple drop-biasing strategies can reduce the queue variability without increasing the likelihood of bursts of packet losses. The results of our investigations have relevance for the design and deployment of RED-like algorithms for congestion control in the Internet.**

I. Introduction

In this paper, we study random drop-based queue management algorithms used for buffering TCP traffic in the Internet. In particular, we study how the use of a simple 'drop-biasing' technique to alter the pattern of packet drops generated by algorithms such as Random Early Detection (RED) [1] *can significantly lower the variability of the queue occupancy, without resulting in any substantial change in the ability of such algorithms to absorb transient bursts.*

The primary objective of algorithms such as RED [1], is to prevent buffer underflow and consequent under-utilization of available bandwidth. As long as this primary objective is unaffected, a randomized dropping strategy should also attempt to reduce the variability of the queue occupancy. Lower occupancy variability not only increases queue stability and reduces the possibility of buffer underflow but also *reduces queuing delay jitter.* Reduced jitter is beneficial, especially for real-time application traffic, such as Voice-over-IP, which might be multiplexed on the same queue, and yet may be only a small component of the traffic load in the immediate future. While such real-time traffic should ideally be buffered separately, widespread deployment of such service differentiation is yet to be realized.

We define *drop-biasing* as the technique by which random dropping queues can alter the distribution of the gap between consecutive packet drops, without changing the mean dropping behavior. Under this technique, the drop probability for an incoming packet is adjusted, based on the number of packets accepted since the last drop. This is usually performed through the use of a variable cnt, which is incremented by 1 for every accepted packet and reset to 0 whenever a packet is selected for random drop.

Archan Misra and Teunis Ott are with Telcordia Technologies, 445 South Street, Morristown, NJ 07960. John Baras is with the Center for Satellite and Hybrid Communication Networks at the University of Maryland, College Park, MD 20742.

We find that using drop-biasing to mandate a minimum separation between successive packet drops decreases queue variability, occasionally to a significant degree. In particular, we use simulations to demonstrate that suitable drop-biasing techniques reduce the queue variance for both persistent and Web TCP flows; for reasons explained later, the reduction is much higher in the case of persistent TCP flows. We also show that this reduction is achieved without compromising the ability of the queue to absorb transient TCP bursts without leading to correlated losses.

We show how the variability in the queue occupancy is related to the correlation among the congestion windows of the competing TCP connections. [9] reported that when TCP flows are buffered by a random drop queue which bases its packet drops on the instantaneous queue occupancy alone, the congestion windows exhibit negative correlation. Due to negative correlation, the congestion windows tend to vary 'out of phase'; this causes a reduction in the variance of the sum of the congestion windows and hence a smaller fluctuation in the queue occupancy. Additional studies, reported elsewhere [22], show that the use of an averaged value of the queue occupancy (as is suggested in RED to reduce the sensitivity to transient packet bursts) can decrease this negative correlation. This paper shows that proper use of drop-biasing can increase this negative correlation among TCP windows, *irrespective of the use of either the instantaneous or the averaged queue occupancy in the dropping process*. As a result of this changing correlation value, the queue variance will change, even though the overall drop probability and the individual TCP window distributions do not change noticeably.

Our results can be used to devise useful drop-biasing strategies for choosing packets for random drop. In scenarios where packet drops are the only way to signal incipient congestion to TCP, our results can lead to improved buffer stability.

A. Related Work

Randomized packet dropping as a congestion indicator for TCP traffic was first proposed in [2], where the dropping probability was based on the instantaneous queue occupancy. The well known paper by Floyd and Jacobson [1] introduced the mechanism of Random Early Detection (RED), which continues to be the most popular random dropping strategy currently employed. This version of RED (which we call 'classical RED' in this paper) bases its dropping probability on a weighted-average of the past queue occupancy and employs a technique to generate a uniform distribution for the gap between successive packet drops. Various modifications to the basic RED algorithm have been proposed. For example, SRED [6] and BLUE [7]

are mechanisms that adaptively tune RED parameters based on estimates of the number of active flows or the buffer overflow and idling events respectively. The drop-biasing feature studied in this paper is orthogonal to any of these original or modified dropping algorithms; it can be used to complement the performance of any of these algorithms.

Several papers have analyzed the TCP window dynamics when subject to random losses. [12], [20], [13] and [8] consider the case when the loss probability is constant; [9] and [16] consider the window distribution of TCP flows when the loss probability is not constant but depends on a function of the connection's window size.

Some recent publications, such as [14], indicate only limited performance improvements with RED in experimental studies and suggest the need for further investigation of random dropping strategies before their widespread adoption on the Internet. Our paper is different from conventional analyses in that it investigates a specific feature of random drop algorithms from the viewpoint of the variation in the queue dynamics, rather than the performance of the TCP flows themselves. However, we take care to show that our proposed modifications does not compromise the fundamental capability of these algorithms to absorb transient TCP bursts.

II. Models and Techniques under Investigation

In this section, we describe the mathematical model for drop-biasing (along with a brief description of how an averaged queue occupancy is used in classical RED). We also indicate the two different traffic source models for TCP traffic used in our simulations, along with their implications on the results of this paper. Finally, we clarify the network topology and metrics used in our simulation studies.

A. Models for Random Drop-based Queuing

In this paper, the packet dropping probability of the buffer is based on some function of the buffer occupancy; flow-specific differentiation is not considered. Let Q be the buffer occupancy[1] of the random drop queue. The **drop function**, which determines the base packet dropping probability, is denoted by $p(Q)$. While $p(Q)$ can, in general, be any non-decreasing function of Q, our simulation studies use the popular linear drop model, given by

$$\begin{aligned} p(Q) &= 0 && \forall\ Q < min_{th} \\ &= 1 && \forall\ Q > max_{th} \\ &= \frac{p_{max} * (Q - max_{th})}{max_{th} - min_{th}} && \forall\ min_{th} \leq Q \leq max_{th} \end{aligned} \quad (1)$$

where, as per RED's standard notation, max_{th} and min_{th} are the maximum and minimum drop thresholds and p_{max} is the maximum packet drop probability. Since all simulations reported here use equal-sized TCP/ UDP packets, all thresholds and queue occupancies are reported in packets (segments) instead of bytes.

[1] Depending on the context, Q represents either the instantaneous queue occupancy, Q_{curr}, or some mapping of the queue occupancies in the past. For classical RED, Q is really a weighted average of the past queue occupancies; for ERD, Q is identical to Q_{curr}.

A.1 Inter-Drop Gap Determination Strategy (Drop-Biasing)

Given a specific drop function, $p(Q)$ provides an estimate of the *averaged independent drop probability*: if the queue occupancy were to remain constant at Q, on an average, *one out of every* $\frac{1}{p(Q)}$ *packets* should be dropped. We can then use various drop-biasing techniques to alter the *distribution* of the gap between drops, without altering the average gap of $\frac{1}{p(Q)}$.

Classical RED performs drop-biasing by using the variable cnt, introduced earlier, to modify the dropping probability of an incoming packet. In classical RED, the packet dropping probability, denoted by p_{drop} is given by the equation

$$p_{drop} = \frac{p(Q)}{1 - cnt * p(Q)}. \quad (2)$$

This code is present in the publicly available ns-2 simulator [18], which we have used for our simulations. Under this approach, the inter-drop gap that is uniformly distributed between $(1, \ldots, \lfloor \frac{1}{p(Q)} \rfloor)$. Neglecting the integer constraints, the *mean* inter-drop gap is then $\approx \frac{1}{2*p(Q)}$, if the queue occupancy Q remains constant. We call this model as the **Uniform** drop-biasing strategy.

Early Random Drop, as discussed in [2] or as applied in [16], on the other hand, computes the dropping probability for each incoming packet by the equation $p_{drop} = p(Q)$, i.e., the drop probability of an incoming packet does not depend on the treatment applied to the past packets. If $p(Q)$ is constant, this approach results in a *geometric distribution* for the duration of an inter-drop period. We call this dropping strategy as the **Geometric** model; note that under this method, a constant Q (and hence $p(Q)$) results in a *mean* inter-drop gap of $\frac{1}{p(Q)}$ packets.

Both the above drop-biasing strategies *do not impose any minimum gap between successive packet losses*; back to back packet losses are indeed possible. We shall later see that introducing such a minimum gap can appreciably reduce the variability of the queue occupancy. A minimum gap between consecutive packet drops can be specified in the uniform dropping strategy of classical RED by delaying random dropping until at least $\frac{1}{p(Q)}$ packets have been accepted. The following pseudo-code (also available in ns-2) uses the variable cnt to achieve this gap:

$$\begin{aligned} if\ cnt <= \frac{1}{p(Q)}, &\quad then \quad p_{drop} = 0, \\ if\ \frac{1}{p(Q)} < cnt <= \frac{2}{p(Q)}, &\quad then \quad p_{drop} = \frac{2}{2 - cnt * p(Q)}, \\ if\ cnt > \frac{1}{p(Q)}, &\quad then \quad p_{drop} = 1. \end{aligned}$$

We call this scheme as the **Delayed Uniform** drop-biasing strategy, since it results in a uniformly distributed gap between $(\frac{1}{p(Q)}, \ldots, \frac{2}{p(Q)})$; the *mean* inter-drop gap in this case is $\frac{3}{2*p(Q)}$.

As a natural corollary to the Delayed Uniform model, we have the **Delayed Geometric** model where the gap between successive packet losses is at least $\frac{1}{p(Q)}$; once $\frac{1}{p(Q)}$ packets have been accepted, each new incoming packet is likely to be dropped with a probability of $p(Q)$. If the drop function $p(Q)$ is constant, this results in a shifted-geometric distribution for the inter-drop gap, with a *mean* inter-drop gap of $\frac{2}{p(Q)}$.

An additional drop-biasing strategy is interesting for its simplicity and resultant insight. This model, which we call the **Deterministic** model, causes every $\frac{1}{p(Q)}^{th}$ packet to be dropped. For a constant $p(Q)$, there is indeed *nothing random* about this packet dropping strategy; accordingly, special artifacts such as synchronization and phase effects [19], that unfairly penalize specific connections, are possible. However, such effects are unlikely in the real Internet where links and traffic paths exhibit random delays. Moreover, a simple randomization scheme which distributes the loss probability over a few packets around the $\frac{1}{p(Q)}^{th}$ packet can effectively counteract this phenomenon. The Deterministic model is the simplest approach for introducing a mandatory (yet unpredictable) separation between successive packet drops. For a fixed value of $p(Q)$, the Deterministic model results in a *mean* inter-drop gap of $\frac{1}{p(Q)}$, as in the Geometric model.

In section III we shall report on the relative performance of random drop queues under these different drop-biasing strategies. Figure 1 provides a visual understanding of how the various dropping strategies result in different shapes for the cumulative distribution function (cdf) for the inter-drop gap.

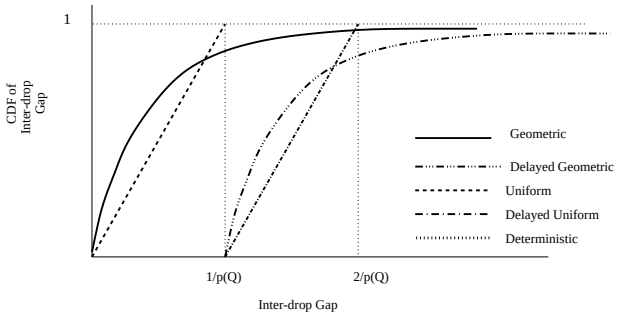


Figure 1: Different CDFs for the Inter-drop Gap

A.2 Past Memory in Drop Function

The drop function, $p(Q)$, can be based either on the instantaneous queue occupancy or on some function of the past queue occupancy. RED uses the *exponentially weighted moving average* model to incorporate the past queue occupancy in the drop pattern; in this model, an *average* queue occupancy Q_{avg} is computed according to the iterative relation

$$Q_{avg}^{i+1} = (1-\alpha)Q_{avg}^{i} + \alpha * Q_{curr}^{i+1}, \qquad (3)$$

where the superscript refers to the arrival of the $i+1^{th}$ packet and Q_{curr} refers to the instantaneous queue occupancy. Note that if the *weight factor*, α, equals 1, the drop function depends only the instantaneous queue occupancy; as $\alpha \downarrow 0$, the memory of the averaging process increases. By varying α within the interval $(0, 1]$, we can obtain the entire range of memory in the dropping process. While the results presented here primarily involve instantaneous queue occupancies ($\alpha = 0$), we have verified that our results apply qualitatively over the entire range of realistic values of α. (We shall in fact provide a few plots to illustrate how our suggested drop-biasing strategies perform equally well when $\alpha = 0.01$.) When $\alpha = 1$, i.e., when the instantaneous queue occupancy is used, we shall refer to the buffer as an ERD queue; when $\alpha \neq 1$, i.e., when averaging is performed, we shall refer to the queue as a RED queue.

B. TCP Source Models and Version

We used two source models in our simulations, since they emphasize two different phases of TCP window evolution. The *persistent* source model assumes infinite-sized file transfers; under this model, the sender's congestion window acts as the only constraint on the injection of new data packets by the sender. In such a situation, when the loss rates are relatively low, TCP primarily exhibits the stationary *congestion avoidance* behavior.

The *Web TCP* model, on the other hand, mimics the effects of Web-based TCP transactions and involves the transfer of finite-sized files. The model and its parameters are based on [15] and consists of a cycle of a single Web *transaction* (each consisting of multiple file transfers) alternating with *inactive off-periods* when no data transfer takes place. Each of the multiple file transfers in a single transaction occurs *sequentially* and on distinct TCP connections. Since most files are only a few KBytes in size, the bulk of the transfers are completed during TCP's initial slow-start transient (where TCP congestion control is less effective). More importantly, the number of active TCP connections fluctuates rapidly under this model; since the occupancy of a RED buffer depends on the number of active flows, the queue occupancy will fluctuate as well [6].

Current Web transfer protocols (e.g., HTTP 1.1 [21]) use *persistent* TCP connections [2]; the same TCP connection is used for multiple transfers, even across multiple transactions. The effective quantity of data transferred by a single connection consequently increases; in comparison to our Web model, a greater portion of the data is now transferred during TCP's stationary congestion avoidance phase. Results in section III show that improvements with drop-biasing strategies are more pronounced for persistent TCP traffic rather than our model of Web traffic. As persistent TCP connections become commonplace in Web transfers, we expect the performance improvement obtained with our drop-biasing strategies to become more significant.

C. Simulation Parameters

Our simulations are performed using the version of TCP New Reno provided in the ns-2 [18] simulator. To provide representative simulation results, we use a simulation topology involving a single random drop queue with a capacity (C) of 1.5 Mbps, a min_{th} of 20 packets, a max_{th} of 200 packets, a p_{max} of 0.05 and a maximum buffer size of 500 packets. (Results from other parameter specifications are qualitatively similar and are not discussed here.) Furthermore, all TCP and UDP connections have a packet size of 512 bytes. To remove the possible synchronization effects among different TCP flows, we ensured that, in almost all simulations, each of the constituent TCP flows had a different round trip time within the range between 25msec and 250msec. (This is achieved by spacing the RTTs of the individual TCP flows uniformly over the interval $(50, 250)$msec.) To

[2] In the context of HTTP, the use of the word 'persistent' implies the use of a single TCP connection for multiple file transfers. This is different from the earlier definition of persistent TCP source models, which refers to the transfer of infinitely large files over a single TCP flow.

generate the appropriate time-series, the queue occupancy and TCP window sizes are sampled every 50msec in all our simulations. The number of persistent TCP sources is varied between $2-15$ while the number of 'Web TCP' sources is varied between $30-120$.

We have seen how, for the same $p(Q)$, different dropping strategies give rise to different mean inter-drop gaps. To provide a fair comparison of the queue variance, we ensured that the mean queue occupancy is nearly the same for all drop-biasing strategies. We can ensure this by having the *mean* inter-drop gap, for a fixed Q, equal for all strategies. In other words, we need $\frac{1}{p_{Geom}(Q)} = \frac{2}{p_{DelayedGeom}(Q)} = \frac{1}{2*p_{Uniform}(Q)} = \frac{3}{2*p_{DelayedUnif}(Q)} = \frac{1}{p_{Deter}(Q)}\ \forall Q$. For $p(Q)$ as in equation (1), this is achieved by setting the p_{max} values for the DelayedGeometric, Uniform, DelayedUniform and Deterministic models to be respectively 2, $\frac{1}{2}$, $\frac{3}{2}$ and 1 times the value of p_{max} for the Geometric model.

D. Jitter Formulation

To measure the effect of alternative drop-biasing strategies on the delay jitter, we sometimes used a probe stream to directly obtain the delay variation. The probe stream injects packets periodically into the queue at a relatively low intensity (64 Kbps).

We used two alternative definitions of jitter. Both definitions involve the specification of a time interval. Under the *percentile* definition, the jitter for that interval is computed as the difference between the 95^{th} and the 5^{th} percentile of the packet delays. The alternative *RTP-based* definition [23] employs a moving average computation over the delay variation between consecutive packets ('per-packet jitter') using

$$\begin{aligned} JitterMov(i) &= (1.0-\beta) * JitterMov(i-1) \\ &\quad +\beta * PerPacketJitter(i) \end{aligned}$$

where $\beta = \frac{1}{16}$. The RTP-jitter for that interval is defined as the maximum value of $JitterMov$ in that interval. Graphs in this paper use intervals of 250msec and 10sec, corresponding to a sample size of ≈ 4 and ≈ 160 probe packets respectively.

E. Effect of Drop-Biasing on Response to Traffic Bursts

To ensure that the drop-biasing strategies do not increase the incidence of bursty losses, we use a set of loss-related metrics per flow and subsequently derive average performance metrics by aggregating over the individual flows.

The simplest such metric of packet losses is the *runlength of packet drops*, which represents the distribution of continuous bursts of losses. To obtain this distribution for a runlength k, we count the total number of packet losses, say L, as well as the number of loss episodes that correspond to a burst of k consecutive losses (on a per-flow basis), say L_k,; the corresponding fraction of losses with runlength k is then $\frac{L_k * k}{L}$.

The runlength is, however, not a very suitable metric, since TCP flows rarely lose *consecutive* packets. More importantly, TCP behavior exhibits timeouts and performance degradation when multiple losses occur in a window; the losses need not be back to back. To study the presence of such extended loss bursts, we study the number of losses in a blocks of consecutive packets. To derive this, we first obtain flow-specific time-series by considering *the number of packet drops* in contiguous blocks (called clusters) of P packets, and hence, the stationary distribution of the number of packet losses per block of P packets (over all the constituent flows). (In our studies, we chose P to be approximately half the reciprocal of the average packet loss rate. This ensures that, in the case of random and independent packet drops, the number of losses in a block is typically either 0 or 1.) To investigate the possible existence of loss bursts of length larger than the block size P, we also determined, for each individual flow, the *auto-covariance* function $C(j),\ j = 0, 1, \ldots$ of the time-series formed by the number of packet drops in each consecutive cluster. (The auto-covariance plots presented here are obtained by averaging over all the flows). In general, a drop-biasing strategy that results in a larger spread of the distribution of the number of packet losses per cluster or larger values of the average auto-covariance $C_{avg}(k)$ for $k = (1, 2, \ldots)$ is less capable of absorbing transient bursts.

III. Effect of Drop-Biasing Techniques on Queue Occupancy Variability

In this section, we investigate how the five dropping strategies enumerated in section II affect the variability of the queue occupancy (and consequently the jitter experienced by the buffered traffic). To study the effect of drop-biasing strategies in isolation, we mostly vary the number of TCP flows while keeping the exponential weight α constant. Most graphs presented in this section involve ERD queues (based on instantaneous queue occupancies); as stated earlier, we have observed similar results for RED queues (with various values of α).

[16] showed that the TCP windows exhibit negative correlation when an ERD queue is used. Negative correlation implies that the window sizes of the TCP connections tend to vary out-of-phase: when the window size of one flow is large, the other flows have smaller window sizes. In such a situation, the sum of the window sizes (and indirectly the buffer occupancy) at any instant would exhibit less variability. Mathematically speaking, we can observe the correlation behavior by comparing the variance of the sum of the window sizes $Var(\sum_{i=1}^{N} W_i)$ against the sum of the individual variances $\sum_{i=1}^{N} Var(W_i)$. When the windows are uncorrelated, the two are equal; for negative correlation, the sum should exhibit lower variance ($Var(\sum_{i=1}^{N} W_i) < \sum_{i=1}^{N} Var(W_i)$), while for positive correlation, the sum should exhibit larger variance ($Var(\sum_{i=1}^{N} W_i) > \sum_{i=1}^{N} Var(W_i)$). This follows from the general relationship

$$Var(\sum_{i=1}^{N} W_i) = \sum_{i=1}^{N} Var(W_i) + \sum_{i \neq j} Cov(W_i, W_j) \quad (4)$$

Thus, the correlation among the windows can be observed from comparisons of the variance of the sum of the windows (or, almost equivalently, the variance of the queue occupancy, $Var(Q)$) with the sum of the variances of the individual windows, $\sum_{i=1}^{N} Var(W_i)$.

We now qualitatively motivate why the introduction of a minimum spacing between consecutive random drops might reduce the queue variability. Suppose a random drop queue drops a packet from TCP flow i at time instant t. If there is no minimum separation between two consecutive packet losses, packets

from other TCP flows may also encounter packet drops soon after t. Since TCP reduces its congestion window in response to a packet drop, such drops can lead to a reduction of the window sizes of multiple TCP connections at around the same time. Imposing a minimum inter-drop gap, on the other hand, ensures that multiple TCP flows do not reduce their windows simultaneously. After a packet from a flow is dropped, packets from other flows are guaranteed to be accepted for the duration of the gap; this process effectively increases the negative correlation among the TCP windows. As a secondary benefit, ensuring a minimum gap between successive packet drops reduces the likelihood of multiple random packet drops from the same flow within a congestion window. Multiple drops within a window can lead to TCP transients such as timeouts and slow start, which increase the burstiness of the offered traffic. Both the above reasons suggest that a minimum inter-drop gap can dampen the fluctuation in the queue occupancy. We now provide the results that we have observed with persistent and Web TCP connections.

A. Queue Behavior with Persistent TCP

Figure 2 shows the occupancy statistics of an ERD queue (as a function of the total number of TCP flows) for different drop-biasing strategies, when the TCP flows have RTTs ranging from 25msec to 250msec. We see that the Deterministic strategy provides the least variance among the five proposed strategies; it also shows the least increase in variance with an increase in the number of TCP flows. Observe also the fairly large reduction in variance between the delayed and non-delayed versions of the Geometric and Uniform dropping models. The above results suggest that introducing a minimum inter-drop gap is much more significant than specifying the exact distribution of the drop pattern. We also note the success of our strategy of adjusting the p_{max}s for different drop-biasing strategies to ensure almost equal mean queue occupancies. The role of negative correlation among the TCP windows can also be obtained from observing figure 2. While the variance of the queue occupancy is different for different drop-biasing strategies, the sums of the variances of the congestion windows of the individual TCP flows is fairly independent of the choice of the drop-biasing strategy. As explained earlier, this establishes that the Deterministic and DelayedUniform drop-biasing strategies lead to stronger 'out-of-phase' behavior among the TCP flows.

To study the impact of the choice of drop-biasing strategy on the burstiness of the packet losses, we present plots of the various burstiness-related metrics in figure 3, for the case of $N = 15$. In this case, the average packet drop probability was obtained (from the corresponding mean of the queue occupancy) to be ~ 0.15, leading to choice of the cluster size of 30. Our plot of the runlength distribution shows that, not only do the Deterministic and DelayedUniform strategies offer lower queue variability, they also reduce the likelihood of back to back losses. Thus, while $\sim 99.5\%$ of losses occur in bursts of 1 for any of the strategies that specify a minimum gap between consecutive packet losses (namely Deterministic, DelayedUniform and Delayed Geometric), the figure is much lower for alternative strategies ($\sim 98\%$ for Uniform and $\sim 96.7\%$ for Geometric). On the other hand, plots of the distribution and auto-covariance function of the number of losses in a block of 30 packets are almost identical for different drop-biasing strategies and are thus not very interesting. We use such plots to assert that the DelayedUniform or Deterministic drop-biasing strategies can offer lower queue variance without leading to burstier losses. We had also experimented with network topologies where all the TCP flows had very similar RTTs; once again, the Deterministic and DelayedUniform drop-biasing strategies outperformed the rest.

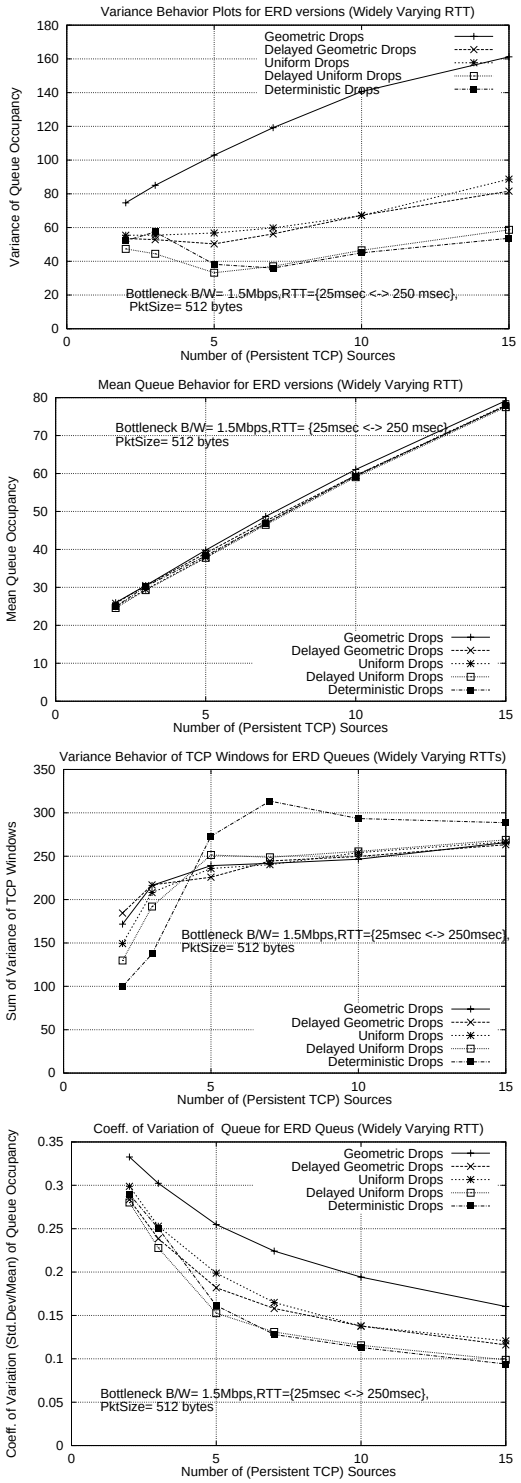


Figure 2: Persistent TCP and ERD Queues

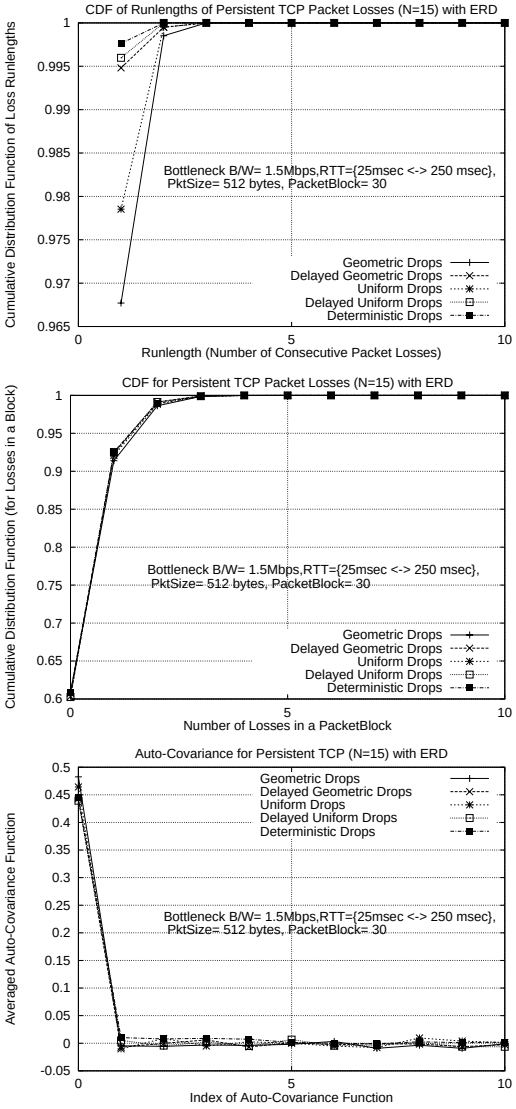


Figure 3: Burstiness-Related Metrics for Persistent TCP and ERD Queues

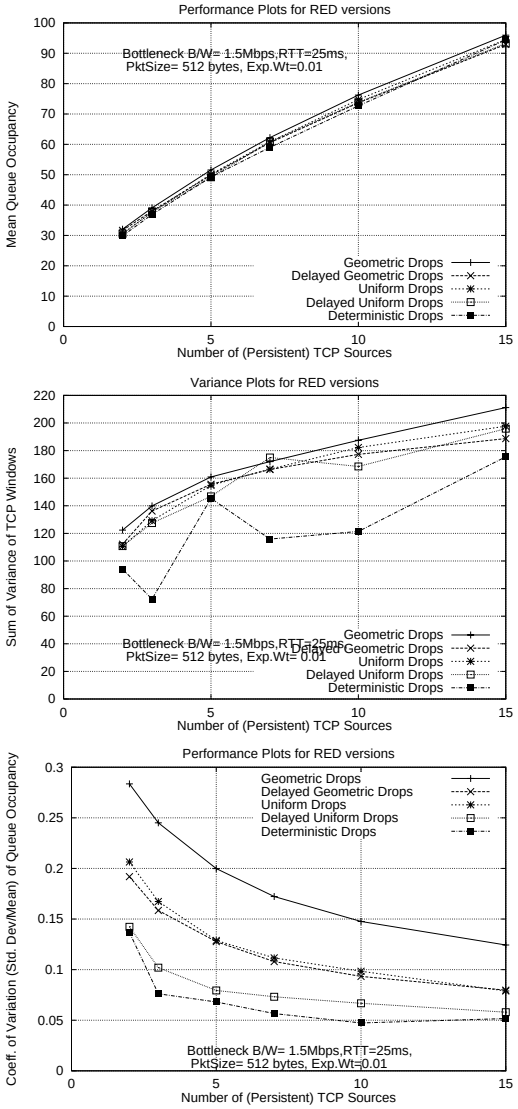
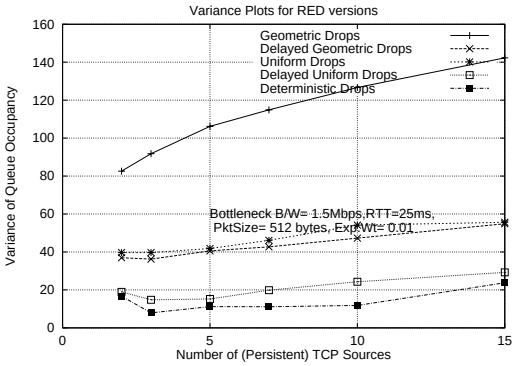


Figure 4: Persistent TCP and RED Queues (Similar RTTs)

Figure 4 shows the results for the same experimental setup as that of figure 2, except that the ERD queue has been replaced by a RED queue with an exponential weight $\alpha = 0.01$ and the RTTs of the TCP flows are all ≈ 25msec. Behavior similar to that mentioned for the instantaneous (ERD) case can be observed. In particular, we have used extensive simulations to verify that the relative performance of the different drop-biasing strategies is qualitatively unaffected by variations in α.



B. *Queue Behavior with Web TCP*

Figure 5 shows the plots for ERD queue behavior with Web TCP traffic (when all the flows had round-trip times of approximately 25msec). Simulations of RED (with exponential averaging) with Web TCP provide similar results, as do simulations with TCP flows with widely divergent round trip times, and are accordingly not presented here. As before, we can observe that the Deterministic and the Delayed Uniform dropping models provide lower queue variance (for the same mean queue occupancies) than alternative dropping strategies.

Note that, compared to the persistent TCP case, the variances are much larger and the difference in variance between the different drop-biasing strategies is relatively lower. (In fact, the differences between the coefficients of variation of the queue occupancies are much lower.) As we have seen, the Web model implies that the number of active TCP connections can vary, even over relatively short time scales. Accordingly, a significant portion of the observed queue variance is simply due to the variability in the number of active connections.

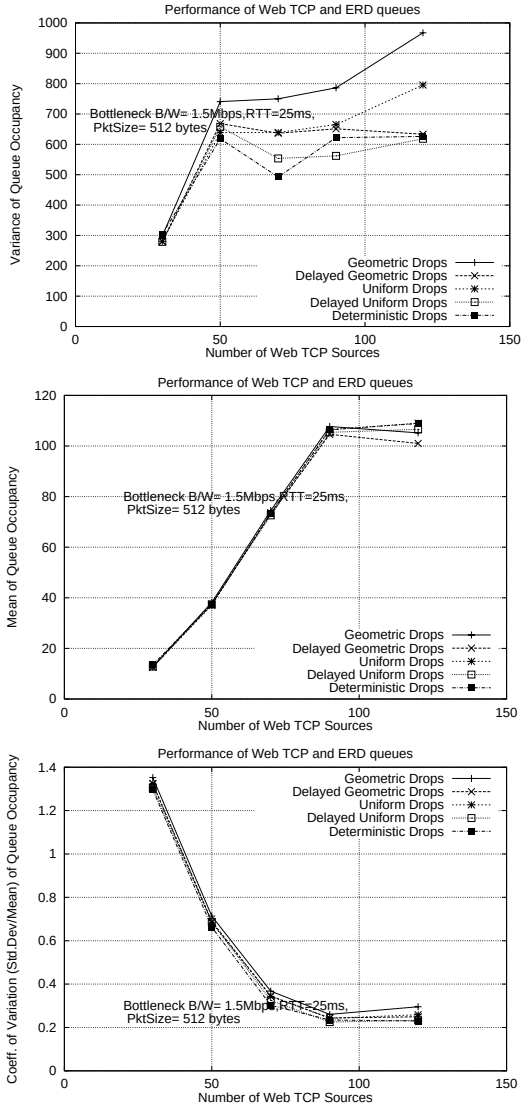


Figure 5: Web TCP and ERD Queue Behavior

To isolate the dependence of the queue variance on the drop-biasing strategies themselves, we also poll the number of active connections at each sampling instant and derive the *conditional variance* of the queue occupancy, i.e. the variance of the queue occupancy as a function of the number of active connections. Plots of the conditional mean and variance of the queue occupancy are provided in figure 6, for the case of 70 Web TCP connections. We also provide the probability distribution of the number of active connections in this case. We can see that the number of active connections lies between $(10, 30)$ most of the time; furthermore, there were never more than 45 active connections present at any sampling instant. The value of 0 for the mean and variance graphs for $N_{active} > 40$ is simply a placeholder indicating the absence of any samples. The graphs of figure 6 clearly reveal that while the conditional means are about the same for each strategy, the conditional variances are very different. The variance of the Deterministic strategy (≈ 200) in the region of $N_{active} = (10, 30)$, where most of the samples are located, is *consistently lower than that of all the alternative drop-biasing strategies*; for example, contrast this with the the variance of the Geometric strategy (≈ 600) in the same region. Mechanisms such as SRED attempt to reduce the dependence of the queue occupancy on the number of active connections. Combining the drop-biasing strategies with such mechanisms will permit us to observe the dependence of the queue variance on the drop-biasing strategies for Web traffic more directly.

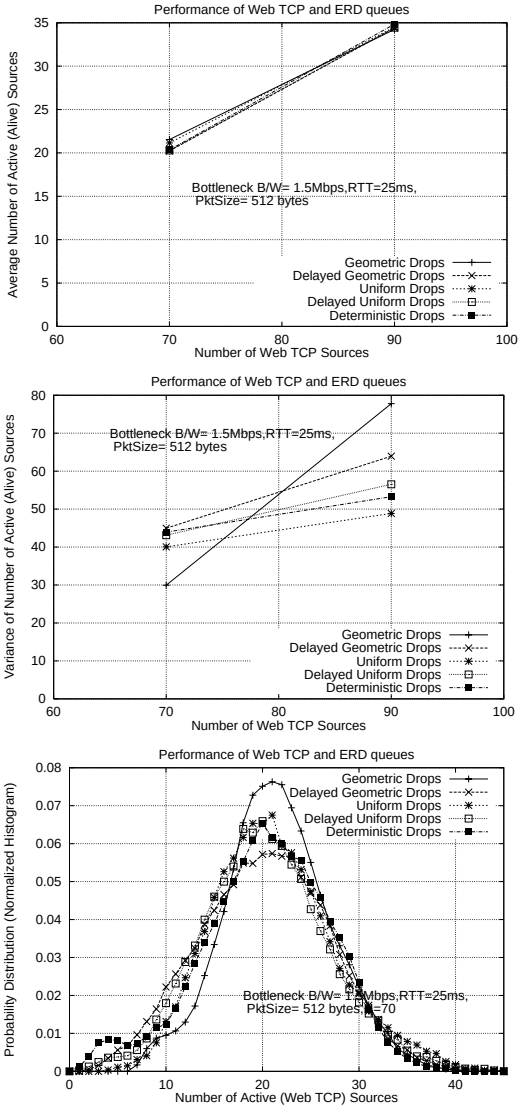


Figure 6: Conditional Plots for Web TCP and ERD

C. Queue Occupancy adn Jitter Plots

For a more direct measurement of the queue variability and packet jitter, we also plotted the dynamics of the queue occupancy as well as the jitter experienced by our periodically generated probe packets. As an illustrative example, we present the plots when the TCP flows had approximately similar round trip times of 25msec.

We first present the results when 10 persistent TCP sources interact with the random drop queue. In this specific instance, we simply present plots of the queue occupancy (sampled at 50msec intervals) for the various drop-biasing strategies in figure 7. These plots are adequate to visually illustrate how, in this case, the Deterministic drop-biasing strategy provides a much smoother queue and smaller packet jitter than the Geometric and

Uniform drop-biasing models.

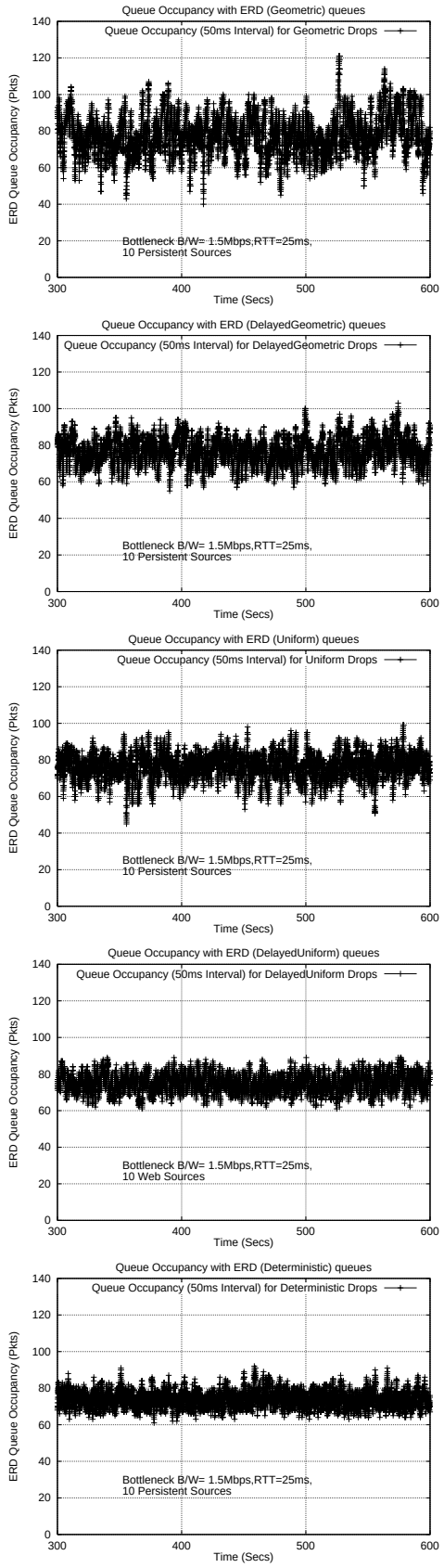


Figure 7: Queue Occupancy with Persistent TCP

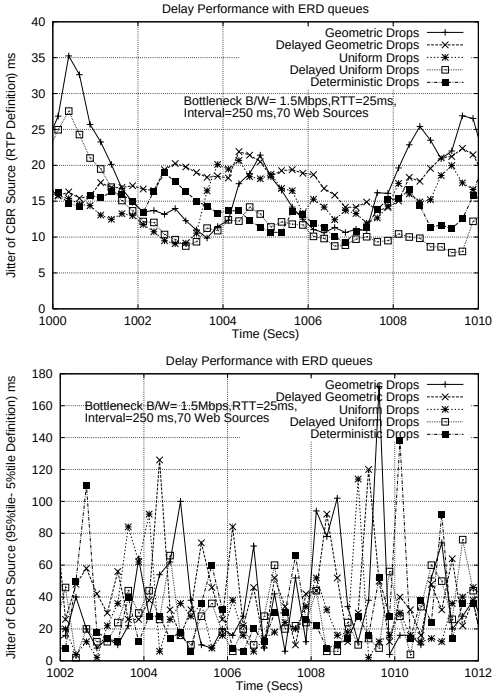


Figure 8: ERD Jitter (250msec Interval) for Web TCP

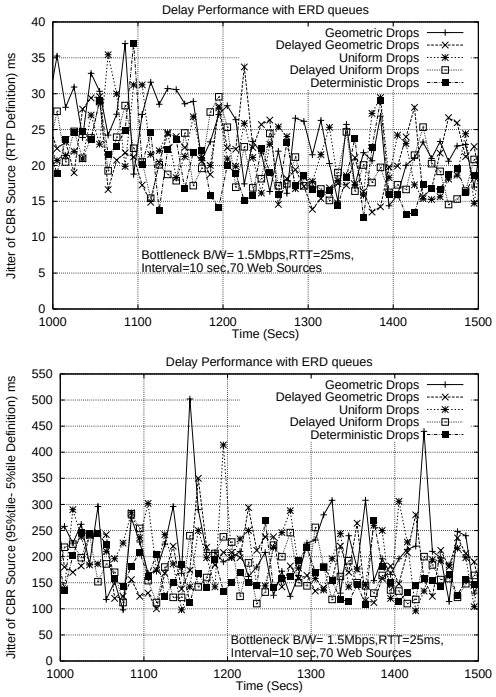


Figure 9: ERD Jitter (10s Interval) for Web TCP

Similar plots, for 70 Web TCP streams, also reveal the reduction in queue variability and delay jitter through the Deterministic and the Delayed Uniform drop-biasing strategies but are less visually apparent than their persistent TCP counterparts. To provide a better visual illustration, we therefore provide the delay jitter plots (as per the two definitions outlined in Section II.E) in figures 8 and 9 for intervals of 250msec and 10sec respectively.

The RTP-based moving averaged jitter is more appropriate for a 250msec interval; on the other hand, the percentile-based definition of jitter is more appropriate for an interval of 10sec. These plots show that, generally speaking, the Deterministic and Delayed Uniform approaches have lower jitter than the other models for inter-drop gap.

D. Main Inferences

Our simulation-based studies show that an appropriate drop-biasing strategy can indeed result in a significant reduction in the variance of the queue occupancy in a random drop queue. Introducing a minimum packet gap between successive random drops provides significant reduction in jitter by increasing the negative correlation among TCP windows and reducing the fluctuations in the queue.

In particular, the Deterministic drop model and the Delayed Uniform drop model were found to perform better than alternative drop-biasing models. The results indicate that specifying a minimum gap between consecutive packet drops affects queue variability much more than detailed specification of the probability distribution of the dropping pattern. The performance improvement was more significant with persistent TCP sources than with Web TCP sources; this is due to the fact that Web TCP sources inherently result in rapid variation in the number of TCP connections. When conditional queue variances were observed, as in figure 6, we could study the performance improvement due to better drop-biasing strategies in isolation.

While the simulations reported here involved a low speed bottleneck (1.5 Mbps bandwidth), we have performed similar simulations at higher link speeds (e.g., 45 Mbps) to study the relevance of our conclusions for higher-speed backbone links. The results obtained are similar and indicate that our conclusions apply for buffers both at the network edges and in the backbone. However, for a given choice of drop-biasing strategy, the coefficient of variation of the queue occupancy is lower at higher link speeds (due to the improved traffic aggregation). Accordingly, relatively speaking, the reduction in queue variance through the use of appropriate drop-biasing schemes is more significant (in terms of the actual reduction in delay jitter) at slower edge links than at faster backbone links.

IV. Conclusions

In this paper, we have shown that the choice of an appropriate drop-biasing strategy can significantly reduce the variability of the occupancy of a random drop queue, without compromising on the buffer's ability to absorb transient bursts. Such reduction in the queue variability can significantly decrease the delay jitter experienced by buffered packets.

In particular, simply introducing a minimum inter-drop gap between successive packet drops significantly reduced the variance of the queue occupancy; we identified the Deterministic and the DelayedUniform schemes as two attractive drop-biasing strategies. These strategies were shown to outperform other candidate drop-biasing strategies irrespective of whether an averaged or instantaneous queue occupancy was used in the determination of the random dropping probability. We also identified and demonstrated how this reduction was achieved through an increase in the negative correlation among the congestion windows of the constituent TCP flows. Since the Deterministic strategy is the least computationally complex, it appears to be an attractive drop-biasing technique.

We have also observed how the improvements are less dramatic for conventional RED-like dropping algorithms when the number of TCP flows is itself variable or when a significant fraction of data transfer occurs during TCP's slow start phase. New mechanisms that make the queue occupancy relatively insensitive to the number of active connections (e.g., SRED) or reduce the variance of TCP's congestion windows (e.g., ECN) should be researched and integrated into router buffers to provide improved performance.

References

[1] S Floyd and V Jacobson, "Random Early Detection Gateways for Congestion Avoidance", IEEE/ACM Transactions on Networking, August 1993.

[2] E Hashem, "Analysis of Random Drop for Gateway Congestion Control", MIT-LCS-TR-506.

[3] B Braden, D Clark et al, "Recommendations on Queue Management and Congestion Avoidance on the Internet", RFC 2309.

[4] V Jacobson, "Congestion Avoidance and Control", SIGCOMM 1988.

[5] S Floyd, TCP and Explicit Congestion Notification, ACM Computer Communication Review, October, 1994.

[6] T Ott, S Lakshman and L Wong, "SRED: Stablized RED", INFOCOM, March 1999.

[7] W Feng, D Kandlur, D Saha and K Shin, "Blue: A New Class of Active Queue Management Algorithms", UM CSE-TR-387-99, 1999.

[8] T Ott, M Matthis and J Kemperman, "The Stationary Behavior of Idealized Congestion Avoidance", ftp://ftp.bellcore.com/pub/tjo/TCPwindow.ps, August 1996.

[9] A Misra and T Ott, "The Window Distribution of Idealized TCP Congestion Avoidance with Variable Packet Loss", Proceedings of Infocom '99, March 1999.

[10] J Padhye, V Firoiu, D Towsley and J Kurose, "Modeling TCP Throughput: a Simple Model and its Empirical Validation", Proceedings of Sigcomm '98, September 1998.

[11] A Kumar, "Comparative Performance Analysis of Versions of TCP in a Local Network with a Lossy Link", IEEE/ACM Transactions on Networking, August 1998.

[12] T V Lakshman, U Madhow and B Suter, "Window-based Error Recovery and Flow Control with a Slow Acknowledgement Channel: a Study of TCP/IP Performance", Proceedings of Infocom '97, April 1997.

[13] S Floyd, "Connections with Multiple Congested Gateways in Packet-Switched Networks Part 1: One-way Traffic", Computer Communication Review, Vol.21, No. 5, October 1991.

[14] M May, J Bolot, C Diot and B Lyles,"Reasons not to deploy RED", INRIA technical report, June 1999.

[15] Barford M and Crovella M, Generating Representative Workloads for Network and Server Performance Evaluation, Boston University Technical Report, BU-CS-97-006.

[16] A Misra, T Ott and J Baras," The Window Distribution of Multiple TCPs with Random Loss Queues", GLOBECOM, December 1999.

[17] V Jacobson, "Modified TCP congestion avoidance algorithm", April 30, 1990, end2end-interest mailing list.

[18] The ns-2 network simulator, http://www-mash.CS.Berkeley.EDU/ns.

[19] S Floyd and V Jacobson, "On Traffic Phase Effects in Packet-switched Gateways", Internetworking Research and Experience, September 1992.

[20] M Mathis, J Semke, J Mahdavi and T Ott, "The Macroscopic Behavior of the TCP Congestion Avoidance Algorithm", Computer Communications Review, July 1997.

[21] J Fielding, J Gettys et al, Hypertext Transfer Protocol- HTTP/1.1, RFC 2616.

[22] A Misra and T Ott, "Effect of Memory on the Stability and Variability of Random Drop Queues", under submission.

[23] H Schulzrinne, S Casner, R Frederick and V Jacobson, "RTP: A Transport Protocol for Real-Time Applications", RFC 1889.