ELSEVIER

Computer Networks 40 (2002) 557–576

COMPUTER NETWORKS

www.elsevier.com/locate/comnet

Predicting bottleneck bandwidth sharing by generalized TCP flows

Archan Misra [a,*], Teunis Ott [b], John Baras [c]

[a] IBM Research, 19 Skyline Drive, Hawthorne, NY 10532, USA

[b] Department of Computer and Information Science, New Jersey Institute for Technology, University Heights, Newark, NJ 07102, USA

[c] Department of Electrical and Computer Engineering, University of Maryland, College Park, MD 20742, USA

Received 19 October 2000; received in revised form 7 February 2002; accepted 30 April 2002

Responsible Editor: G. Morabito

Abstract

The paper presents a technique for computing the individual throughputs and the average queue occupancy when multiple TCP connections share a single bottleneck buffer. The bottleneck buffer is assumed to perform congestion feedback via randomized packet marking or drops. We first present a fixed point-based analytical technique to compute the mean congestion window sizes, the mean queue occupancy and the individual throughputs when the TCP flows perform idealized congestion avoidance. We subsequently extend the technique to analyze the case where TCP flows perform generalized congestion avoidance and demonstrate the use of this technique under the Assured Service model, where each flow is assured a minimum traffic rate. Simulations are used to demonstrate the accuracy of this technique for relatively low values of packet dropping probability and a much wider range of packet marking probability.

Keywords: TCP; Throughput; RED; ECN; Congestion avoidance; Queues

1. Introduction

In this paper, we present a mathematical technique for computing how competing TCP flows share the link capacity and buffer space of a bottleneck queue. In particular, we first consider the interaction of multiple *persistent* TCP flows, each performing *idealized congestion avoidance* [1], with a buffer performing congestion control via random packet drops/marking. We develop an analytical technique, resulting in a fixed-point iteration scheme, to obtain the 'center' of the queue occupancy and the individual TCP windows and use such values to determine the individual throughput of each TCP flow. Such a technique is used to numerically predict the manner in which TCP flows share resources in the presence of a queue using algorithms such as Random Early Detection (RED) [2] and Explicit Congestion Notification (ECN) [3].

[*] Corresponding author. Tel.: +1-914-784-7116; fax: +1-914-784-6201.

E-mail addresses: archan@us.ibm.com (A. Misra), ott@oak.njit.edu (T. Ott), baras@isr.umd.edu (J. Baras).

1389-1286/02/$ - see front matter © 2002 Elsevier Science B.V. All rights reserved.
PII: S1389-1286(02)00296-7

We subsequently extend the analysis to consider the case of TCP flows performing *generalized congestion avoidance*. Under this generalization of TCPs current window adjustment algorithm, a TCP flow increments its congestion window, *cwnd*, [1] from its current value W by $c_1 W^{\alpha}$ in the absence of congestion and decreases the window by $c_2 W^{\beta}$ in the presence of congestion (where α, β, c_1 and c_2 are constants that parameterize the window adjustment algorithm). This model of window adjustment corresponds to the family of binomial congestion control algorithms studied in [4]. We present the extensions to our fixed-point technique necessary for this generalized case. To further demonstrate the flexibility of this fixed-point mechanism, we consider the case where such generalized TCP flows are regulated by the Assured Service [5] model. Under this model, each flow is guaranteed a minimum *assured* traffic rate; packets from a flow should ideally experience no congestion as long as its offered traffic does not exceed this rate. A suitable modification of the fixed-point analysis technique leads to a reasonably accurate method for predicting the individual TCP window sizes and throughputs under this service model as well.

Mathematically speaking, we treat the evolution of the congestion window of a TCP flow as an idealized stochastic process. In particular, we consider the TCP Reno version [6] of window adaptation, where detection of congestion (through duplicate acknowledgments or via explicit setting of a congestion indicator bit) results in an halving of the congestion window. Of course, most TCP versions do not respond to multiple congestion indicators (packet markings or drops) within a single window, but rather assume that the indicators collectively signal a single congestion episode and thus halve their window only once. We shall later explain why our model provides reasonably accurate approximation of such behavior as long as the notification probability is moderately low. By disregarding transient phenomena such as fast recovery [1] and timeouts, we can model the window evolution of the generalized TCP flow as a Markov process with the following state-transition probabilities:

$$P\{W_{n+1} = w + c_1 w^{\alpha} | W_n = w\} = 1 - p(w), \qquad (1.1)$$

$$P\{W_{n+1} = w - c_2 w^{\beta} | W_n = w\} = p(w), \qquad (1.2)$$

where $p(w)$ is the (state-dependent) congestion notification probability. The classical congestion avoidance algorithm is obtained by setting $\alpha = -1$, $\beta = 1$, $c_1 = 1.0$ and $c_2 = 0.5$. The value of the TCP window that corresponds to 'zero-drift', whereby the probability of window increase equals the probability of window decrease, is assumed to represent the 'center' of the flow's window distribution. It should be noted that, strictly speaking, when multiple TCP flows interact with a single queue, the probability for congestion notification for a specific flow depends not just on its window, but also on the instantaneous window sizes of all the other connections. [2]

Congestion notification is an abstract event in our analysis; packet drops and packet marking are thus fundamentally equivalent events. Our fixed-point analysis technique thus applies irrespective of whether the buffer's congestion control uses packet drops or ECN-based packet marking. To illustrate the accuracy of our analysis for conventional TCP flows, we shall first present simulation studies based on feedback via *randomized dropping*. For numerical studies of the generalized congestion avoidance algorithm, we shall however use a buffer that performs congestion control via *packet marking*. The use of such examples seem justified, since modifications to TCP congestion avoidance are likely to occur jointly after the de-

[1] While *cwnd* in actual TCP implementations is expressed in bytes and is consequently integer-valued, we assume that, in Eqs. (1.1) and (1.2), W is real-valued and is expressed in maximum segment size (MSS) units. The congestion window in the rest of this paper is assumed to be real-valued. We will explicitly mention the situations where the congestion window is expressed in bytes.

[2] An accurate model of the window evolution process for N TCP connections would require an N-dimensional Markov model, where the state space would be a N-dimensional vector consisting of the window sizes of each individual connection. The transition probabilities between states would depend on the state of the entire system (the instantaneous windows of each connection), making useful analysis impossible.

ployment of explicit congestion notification mechanisms, such as ECN, in the Internet. We shall see that, while our generic fixed-point formulation applies to both packet marking and dropping-based congestion control, our specific use of the "square-root formula" is more accurate over a much wider range of packet *marking* probabilities.

Our analytical technique computes only the 'center' of the queue occupancy (and not higher order statistics). As an indirect fallout of our simulations, we shall however also demonstrate how such randomized congestion feedback typically causes the TCP windows to be *negatively correlated*. Negative correlation implies that the TCP windows tend to vary 'out-of-phase'; such behavior causes the variance of the queue occupancy to be smaller that the sum of the variances of the individual flows.

The rest of the paper is organized as follows. Section 2 describes our mathematical model for the interaction between multiple TCP flows and a RED-like buffer. Section 3 presents the analytical technique for multiple flows performing idealized congestion avoidance and provides simulation results to demonstrate the accuracy of our analysis. This section also reveals how our use of the "square-root formula" leads to higher accuracy when notification is primarily achieved through ECN-based feedback. Section 4 considers the extension of the analysis for the case of multiple generalized TCPs under the Assured Service model and presents comparisons with simulated results. Finally, Section 5 concludes the paper.

1.1. Related work and model applicability

While several papers (e.g., [2,7]) have used simulations and experiments to consider the effect of RED-like randomized feedback algorithms on TCP throughput, relatively little work has been published on analytical techniques for computing such throughput sharing. The fixed-point method for analyzing the sharing of a bottleneck buffer was first presented in [8,9] later presented a similar analysis based on a control-theoretic model. The extension of the fixed-point technique for generalized TCP flows was investigated in [10]. This paper combines the results in [8,10] into a common framework. Ref. [8] also shows how such fixed-point analysis is exploited to evaluate two alternative techniques for computing the *window distribution* of a TCP flow in such a multi-flow case.

The role of negative correlation in stabilizing the queue occupancy was explored in [8,11], which also showed how the use of averaged values of past queue occupancies and drop-biasing techniques could quantitatively modify such correlation behavior. Our analytical technique only computes the mean queue occupancy; we do not make any claims on the dynamic behavior of the queue. Ref. [9] treats the queue as a dynamical system and shows that such a queue can exhibit instabilities and oscillatory behavior; such behavior is mathematically motivated in [12].

Our analytical technique applies to queues where the drop or marking probability is based only on the queue occupancy and is independent of the number of active flows. Active queue management algorithms, such as SRED [13] and BLUE [14] attempt to stabilize the bottleneck queue occupancy by dynamically adjusting the drop/marking thresholds based on the offered load. While such algorithms could be incorporated into our analysis by appropriate adjustment of the notification probabilities, we have not explicitly considered such enhancements in this paper.

The 'drift-based computation' of a flow's mean window size was previously used in several papers (e.g. [7,16,17]) to compute the throughput of a TCP flow subject to a constant packet drop probability. Such an approach leads to the 'square-root formula' for classical congestion avoidance, which states that the mean window of a TCP connection is inversely proportional to the square-root of the loss probability. More detailed models of TCP behavior, that consider the effect of timeouts and fast recovery, are considered in [18,19], and essentially show that the TCP throughput becomes inversely proportional to the packet drop probability at moderately high loss rates. Our idealized analytical technique is demonstrated using the classical square-root formula and is thus applicable only when packet loss rates are relatively low and transmission timeouts are relatively rare events.

The generic analytical framework has however been also used in conjunction with other more accurate models of TCP behavior to provide better approximations for higher packet loss rates. For example, [20] used the fixed point framework with the PFTK formula [19] for TCP behavior to demonstrate better agreement under higher rates of RED-based packet dropping. We focus primarily on establishing the principles of the fixed-point formulation, rather than studying specific models of TCP response. However, our simulation studies will demonstrate that the use of the fixed-point method with the square-root formula is accurate for a wide range of ECN-based congestion feedback—in essence, our model is accurate as long as the packet *loss* rates can be kept low enough to restrict the occurrence of retransmission timeouts.

2. Mathematical model and problem approach

In this section, we first describe the TCP source model for classical congestion avoidance, and the random drop-based buffer management algorithm. The corresponding extensions for generalized congestion avoidance and marking-based queue management are obvious and are presented thereafter. We also present the Assured Service model, which we shall analyze later.

2.1. TCP sources

The TCP connections are *persistent* (sending infinite-sized data files), with the congestion window acting as the only constraint on the injection of new packets by the sender. We assume that the connection never times out, that the data is always sent in equal-sized segments (although segment sizes could vary between connections) and that acknowledgments are never lost. For the purposes of presentation, we assume that the receiver acknowledges every received packet separately (delayed acknowledgments are not enabled); delayed acknowledgments can be incorporated into the model using the approximation in Appendix C.

Let N be the number of concurrent TCP connections under consideration. The ith TCP flow, TCP_i, has a MSS of M_i bytes. Moreover, the round trip time of the ith TCP connection is assumed to consist of two components: a *fixed* component denoted by RTT_i (seconds) and a *variable* queuing component. Since our model assumes that each flow essentially faces only one bottleneck, the delay in the rest of the traffic path can be assumed to be fixed and determined largely by the propagation and transmission delays of the constituent links (queuing delays in such non-bottleneck nodes are assumed to be negligible). The queuing delay at the bottleneck node is explicitly modeled and contributes to the variable component of the delay. Let W_i denote the window size (in MSSs) of the ith connection. Note that while the process model (Eqs. (1.1) and (1.2)) represents the window state in segments, we shall also occasionally refer (explicitly) to the window size in bytes.

2.2. Queue behavior

For the analysis in Section 3, we consider a RED-like queue which subjects all incoming packets to *random* packet drops/marking, with a probability that depends on the *instantaneous* queue occupancy. The specialized 'ORED' queue behavior for the analysis of the Assured Service model will be presented in Section 2.3.

The service rate (bandwidth) of the queue is C bytes/s. In general, let Q be the buffer occupancy of the random drop/marking queue and Q_i (in bytes) be the amount of traffic from connection i that is buffered in the queue (so that $\sum_{i=1}^{N} Q_i = Q$). The drop/marking function is denoted by $f(Q)$. For the simulation results, we use the linear drop/marking model, with $f(Q)$ given by

$$f(Q) = \begin{cases} 0 & \forall Q < \min_{\mathrm{th}} \\ p_{\max} & \forall Q > \max_{\mathrm{th}} \\ p_{\max} \dfrac{Q - \min_{\mathrm{th}}}{\max_{\mathrm{th}} - \min_{\mathrm{th}}} & \forall \min_{\mathrm{th}} \leqslant Q \leqslant \max_{\mathrm{th}}, \end{cases} \tag{2.1}$$

where, as per standard notation, $\max_{\mathrm{th}}$ and $\min_{\mathrm{th}}$ are the maximum and minimum drop/marking thresholds (in bytes) and $p_{\max}$ is the maximum packet drop/marking probability. (This is in fact, an even gentler version of the 'gentle' model of RED behavior recommended in [21].) From an analytical viewpoint, we merely need $f(Q)$ to be non-decreasing in Q; this is true for all sensible drop functions.

Although our analysis is primarily focussed on algorithms that do not maintain flow-specific state (and do not distinguish between flows), a slight generalization, which allows the actual packet congestion notification probability to be flow-dependent, is possible. To that extent, we suppose that the congestion notification probability for a packet of flow i, which arrives when the queue occupancy is Q, is given by the function $f_i(Q)$. $f_i(Q)$ is related to our afore-mentioned drop/marking function $f(Q)$ by the expression:

$$f_i(Q) = c_i^2 f(Q), \tag{2.2}$$

where the c_i are arbitrary non-zero constants. Our model thus permits the notification function for different connections to be *scalar multiples* of one another; the scalar values are represented as c_i^2 instead of c_i for future notational convenience.

This scalar model permits us, for example, to capture the *byte-mode* of operation of RED where the probability of a *packet* drop is proportional to the size of the packet (by setting $c_i^2 = M_i$). [3] Also, for convenience, we shall use $p_i(W)$ to represent the (as yet unknown) relationship between the packet drop/marking probability of TCP$_i$ and its window size W. The reader may note that packet drops or marking in RED, unlike our reference model, are not truly conditionally independent; a simple correction for our model in such a situation is discussed in Appendix B.

2.3. Assured Service model and buffer behavior

The Assured Service model [5] describes a framework for differential bandwidth sharing, where each flow (user) is guaranteed a minimum or *assured* rate as part of their service profile. Adequate capacity provisioning is assumed to ensure that packets from a flow experience minimal congestive losses/marking as long as its transmission rate lies within this assured rate. Flows are allowed to inject additional (opportunistic) packets beyond this assured rate; such packets are treated as best-effort and have lower priority. To enable network buffers to differentiate between such packets, [5] proposes a tagging mechanism at the network edge. Packets which stay within the profiled rate are tagged as *in* packets while packets that violate the profile are tagged as *out* packets. Mechanisms such as a leaky bucket [22] or modifications thereof [5] may be used to implement the tagging operation. *In* packets are provided preferential treatment in network buffers via the RIO (RED with In/Out) discard algorithm; RIO is similar to RED except that it uses different thresholds for *in* and *out* packets to ensure that *out* (opportunistic) packets are dropped before *in* packets. For simulation-based studies involving the generalized congestion avoidance algorithm, we assume that our bottleneck queue uses the ORED [10] buffer management algorithm; ORED is similar to RIO but differs in two respects:

- ORED marks *out* packets instead of dropping them.
- ORED does not signal congestion notification for *in* packets, except when the buffer overflows and packets are dropped.

As in the classical congestion avoidance case, the generalized TCP flow TCP$_i$ has an MSS of M_i bytes and a round-trip time of RTT$_i$ s. Additionally, TCP$_i$ is assumed to have an assured rate of R_i bytes/s and can consequently expect to receive no congestion feedback as long as its transmission rate ρ_i is less than R_i. The flows interact with an ORED buffer serving a link of capacity C bytes/s. Our analysis assumes that [4]

$$C > \sum_{i=1}^{N} R_i, \tag{2.3}$$

[3] Our ‘scalar-multiple’ model of flow-dependent notification probabilities can capture a much richer set of randomized feedback settings than apparent at first glance. For example, it can represent a setting of weighted RED where the different classes have the same min$_{\text{th}}$ and max$_{\text{th}}$ thresholds but different $p_{\max}$. We do not explore the validation of such settings further in this paper.

[4] If $C < \sum_{i=1}^{N} R_i$, then ECN marking will occur even though at least one TCP flow obtains less than its assured rate. This is clearly a violation of the Assured Service model.

i.e., the link capacity is greater than the sum of the *assured* rates of the individual flows.

The marking function of the ORED buffer (for *out* packets) is given by the traditional linear model $f(Q)$ presented in Eq. (2.2).

3. Estimating the mean queue occupancy for classical congestion avoidance

In this section, we consider the interaction of TCP flows performing classical congestion avoidance with a bottleneck buffer performing feedback through random packet drops or marking. For the classical congestion algorithm, the state-transition probabilities of the ith Markovian process are given by

$$P\left\{ W_{n+1} = w + \frac{1}{w} \middle| W_n = w \right\} = 1 - p_i(w), \tag{3.1}$$

$$P\left\{ W_{n+1} = \frac{w}{2} \middle| W_n = w \right\} = p_i(w). \tag{3.2}$$

We first use a drift-based argument to determine the center of the queue occupancy, denoted by Q^*, and the centers of the *cwnd*-s of the individual connections, denoted by W_i^*, $i = \{1, \ldots, N\}$. To estimate the *center* of the queue occupancy, we use a set of fixed point mappings. The basic idea is to find values for the average window sizes, such that the average queue size given by those set of values is consistent with the average notification probability that is implied by the window sizes. The derivation of the 'square-root' formula via the drift-based technique is borrowed from [23]. As noted earlier, let Q^* be this mean or center value of the queue occupancy and let W_i^*, $i \in \{1, 2, \ldots, N\}$ be the center of the ith TCP flow.

3.1. Formulating the fixed point equations

Define the *drift* of the congestion window of a TCP flow (momentarily dropping the flow-specific subscript) by the expected change, ΔW, in its window size. Since, for a window size of w, the window size (in packets) increases by $1/w$ with probability $1 - p(w)$ and decreases by $w/2$ with probability $p(w)$, we have

$$\Delta W = (1 - p(w)) \frac{1}{w} - p(w) \frac{w}{2}. \tag{3.3}$$

From the above equation, the *center* or '0-drift' value of W, called W^*, is seen to be

$$W^* \approx \sqrt{2 \frac{1}{p(W^*)}}, \tag{3.4}$$

where the approximation is quite accurate as p is usually quite small [5] (for current TCP versions, if the drop probability exceeds 0.05, timeouts and slow start phenomena begin to dominate TCP behavior).

The notification probability for flow i, $p_i(W)$, for a given value, Q of the buffer occupancy is given by the relationship $p_i(W) = f_i(Q)$. Accordingly, in the multi-TCP case, the zero-drift analysis gives the following expression (in packets) for the mean window size for flow i:

$$W_i'(\text{packets}) \simeq \sqrt{\frac{2}{f_i(Q^*)}}. \tag{3.5}$$

By incorporating expression (2.2) in the above equation and noting that each packet of flow i is M_i bytes in size, we get the mean window size (in bytes) as:

$$W_i^* \simeq \frac{M_i}{c_i} \sqrt{\frac{2}{f(Q^*)}}. \tag{3.6}$$

Now, let C_i be the average bandwidth obtained by TCP i. Assuming that there is no significant buffer underflow and that the link is fully utilized (after all, this is a bottleneck queue), we get the relation $\sum_{i=1}^{N} C_i = C$. C_i can also be computed by a different method: by noting that a TCP connection sends one window worth of data in one effective round trip time. Since a queue of size Q will contribute a buffering delay of Q/C, the effective round trip time of connection i is $\mathrm{RTT}_i + Q/C$; thus, we can relate C_i to W_i by the expression

[5] A more accurate analysis [23] reveals that the mean window occupancy, in ack time, is given by $W^* \approx 1.5269/\sqrt{p}$. It is this value that we used in all our experimental results; for notational ease, however, we shall continue using the $\sqrt{2}$ approximation in our exposition.

$$C_i = \frac{W_i^*}{\mathrm{RTT}_i + Q/C}. \tag{3.7}$$

On summing the C_is from the above equation and equating them to C, we get

$$C = W \sum_{i=1}^{N} \frac{M_i/c_i}{\mathrm{RTT}_i + Q/C}, \tag{3.8}$$

or, upon simplification,

$$W = \frac{1}{\sum_{i=1}^{N} \frac{M_i/c_i}{Q + C*\mathrm{RTT}_i}}, \tag{3.9}$$

where $W = \sqrt{2/f(Q)}$. For notational convenience, let the RHS of Eq. (3.9) be denoted by the function $g(Q)$ so that

$$g(Q) = \left(\sum_{i=1}^{N} \frac{M_i/c_i}{Q + C * \mathrm{RTT}_i} \right)^{-1}.$$

The *fixed point* solutions for the 'average' TCP window sizes and the queue occupancy is then given by the set of values that provide a solution to the following simultaneous equations:

$$W = \sqrt{\frac{2}{f(Q)}}, \tag{3.10}$$

$$W = \left(\sum_{i=1}^{N} \frac{M_i/c_i}{Q + C * \mathrm{RTT}_i} \right)^{-1} = g(Q). \tag{3.11}$$

After solving these simultaneous equations, we can get the 'average' congestion window for the ith TCP flow (in bytes) using the relation

$$W_i^* = \frac{M_i}{c_i} W^*. \tag{3.12}$$

We can then obtain the throughput, ρ_i, of TCP$_i$ by using the relation

$$\rho_i = \frac{W_i^*}{\mathrm{RTT}_i + Q^*/C}. \tag{3.13}$$

3.2. More generic models for TCP behavior

It is well-known that current TCP versions show appreciable deviation from the "square-root" formula if the packet loss rate is larger than ≈5%. (This deviation occurs primarily due to the overhead of retransmission timeouts and slow start caused by multiple packet losses within a single window worth of packets). The above fixed point model can, however, be easily extended to consider more accurate or generic models of TCP response. For example, by consider the effects of TCP fast retransmits and timeouts, Ref. [19] showed that the window size (expressed in MSS) of a single TCP flow subject to a *loss* probability $f_i(Q)$ is well approximated by:

$$W_i = \mathrm{RTT}_i \bigg/ \left\{ \mathrm{RTT}_i \sqrt{\frac{2f_i(Q)}{3}} + T_0 \min\left(1, \sqrt{\frac{27 f_i(Q)}{8}}\right) f_i(Q)(1 + 32 f_i^2(Q)) \right\}, \tag{3.14}$$

where T_0 represents the base retransmission timeout interval. Since the total RTT of TCP$_i$ is clearly a function of Q, it follows that Eq. (3.14) can also be represented in the form $W_i = g(Q)$, with an appropriately defined $g(\cdot)$. This is fundamentally similar to the form of Eq. (3.11); accordingly, the same fixed point technique can be used to solve for Q^* even in this case. To maintain our focus on the fixed point technique itself (rather than the precise form of $g(Q)$), we do not consider such refinements any further in this paper.

Our Markovian model for TCP window evolution assumes that the TCP window halves on the receipt of *every* congestion indicator, even if they occur in fairly close succession. This is, of course, an idealized behavior, single most current TCP implementations treat multiple packet drops/markings within a single window as indicative of a single congestion event and halve their window only once. We argue that our model is reasonably accurate since the number of random packet drops/markings within a single window should be either 0 or 1 in a well-behaved queuing system. To see this, assume that the router congestion notification probability stays constant at p. Then, the average window size of a TCP flow subject to feedback from such a router is $W^*(p) \simeq \sqrt{2/p}$ (from Eq. (3.4)). Let X represent the random variable representing the number of notification events in such a window $W^*(p)$. Then, the

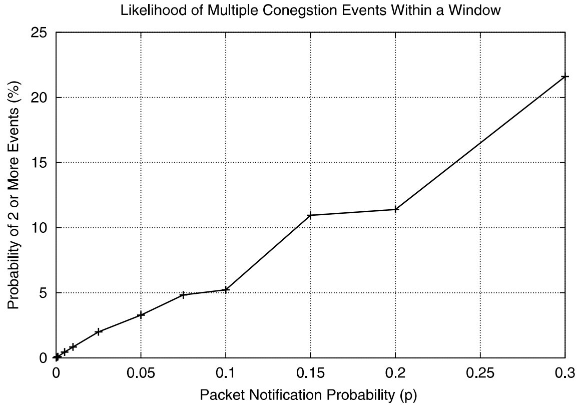

Fig. 1. Probability of two or more loss/marking events in a congestion window.

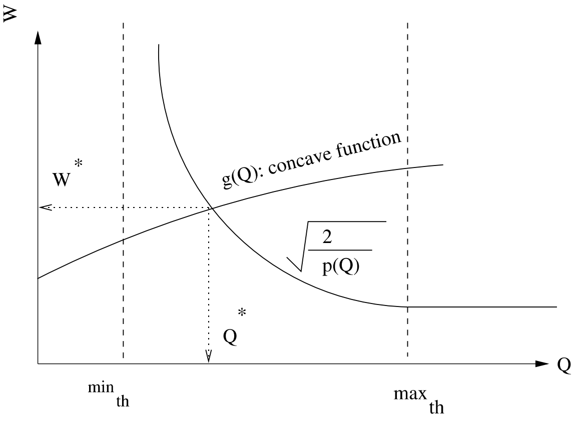

Fig. 2. Typical relationship between W and Q for random drop queues.

probability of multiple losses, $\Pr\{X \geqslant 2\}$ under a truly *independent* feedback model is given by the binomial model:

$$\begin{aligned}\Pr\{X \geqslant 2\} &\approx 1 - \Pr\{X = 0\} - \Pr\{X = 1\} \\ &\approx 1 - (1-p)^{W^*(p)} \\ &\quad - \binom{W^*(p)}{1} p(1-p)^{W^*(p)-1}. \qquad (3.15)\end{aligned}$$

Fig. 1 plots this probability as a function of p. It is easy to see that the probability of multiple losses is fairly small even for moderately large values of p (around 10% for $p = 0.2$). This low probability is easily explained by observing that a larger p also leads to a smaller value of the "average" window size, thus reducing the likelihood of multiple losses within a single window. Accordingly, our stochastic model of TCP window evolution seems to be a reasonably good approximation to TCP behavior (sans timeouts of course).

3.3. Existence and solution of fixed point

We now prove the existence of a unique solution to the above simultaneous equations (i.e., (3.10) and (3.11)) and also provide a numerical technique for its rapid computation.

The existence of a unique solution can be demonstrated graphically (see Fig. 2) by simultaneously plotting Eqs. (3.10) and (3.11) on the (Q, W) axes. Since $f(Q)$ is assumed non-decreasing in Q, we have W in Eq. (3.10) to be a non-increasing function of Q. On the other hand, $g(Q)$ in Eq. (3.11) can be seen to be an increasing function of Q. The two plots will therefore intersect at a single point, which is our 'zero-drift' solution for W^* and Q^*.

In Appendix A, we prove that the function $g(Q)$ is concave; accordingly we can see that the function $f(Q)$, defined by the difference between the RHS of Eqs. (3.10) and (3.11), is convex in Q:

$$s(Q) = \sqrt{\frac{2}{f(Q)}} - \left\{ 1 \Big/ \sum_{i=1}^{N} \frac{M_i/c_i}{Q + C * \mathrm{RTT}_i} \right\}. \qquad (3.16)$$

Hence, we use the Newton gradient technique, which is guaranteed to converge and provide a solution to the equation $s(Q) = 0$, to solve for the fixed point. We start with an initial estimate of $Q_0 = \min_{\mathrm{th}} + \delta$ (an initial value to the left of Q^*) and proceed with repeated iteration. In this particular setting, the derivative $s'(Q_j)$ at the jth iteration is given by

$$\frac{f'(Q_j)}{\sqrt{2} f(Q_j)^{3/2}} - \left\{ \sum_{i=1}^{N} \frac{M_i/c_i}{(Q_j + C * \mathrm{RTT}_i)^2} \Big/ \left(\sum_{i=1}^{N} \frac{M_i/c_i}{Q_j + C * \mathrm{RTT}_i} \right)^2 \right\}. \qquad (3.17)$$

3.4. Insights from above analysis

The drift analysis technique provides some insights for predicting or controlling the stationary behavior of persistent TCP connections and for understanding the accuracy of our approximation technique. For example, our analysis shows that:

- TCP connections with the same round trip time but different packet sizes will see the same 'average' window size (in bytes) if $c_i = \alpha M_i \ \forall i$, where α is an arbitrary constant. In other words, to ensure fair sharing of throughput among TCP connections with different packet sizes, the packet dropping probability should be proportional to the square of the packet size. Contrast this with current byte-mode drop schemes where the packet drop probability is normally proportional to the packet size.
- TCP connections which are identical, except for different round trip times, will observe relative throughput that is inversely proportional to the round trip times. This unfairness towards TCP connections with larger round-trip times is well known.
- Since W^* (the 'fixed point' that satisfies both Eqs. (3.10) and (3.11)) is identical for all flows, it should be clear from Eq. (3.12) that the mean value of the window size (in packets) for all TCP flows, which have the same drop function (same p_is), will be the same, irrespective of their round-trip times and segment sizes. The point is more subtle than apparent at first glance: the means are identical only when expressed in MSSs and when the distribution is taken with respect to *ack time*. When sampled in *clock time*, the mean window size of a TCP connection will indeed depend on its round-trip delay (which influences the rate of progress of the connection). We can, however, easily compute the distribution in clock time from that in ack time, if the round-trip delay for a specific connection is non-varying (through the relation $\mathrm{d}F_{\mathrm{ack}}(x) = \{x\,\mathrm{d}F_{\mathrm{clock}}(x)/\int_0^\infty y\,\mathrm{d}F_{\mathrm{clock}}(y)\}$). As the number of flows increases, we shall later see that the buffer occupancy (and hence, the queuing delay) shows relatively smaller variation; estimates of clock-time distributions from our ack-time calculations are progressively more accurate.

3.5. Simulation results for the mean window sizes

We used a wide variety of simulations, with various combinations of segment sizes and round trip times, to verify the accuracy of our fixed point-based prediction technique. All simulations are performed with New Reno TCP sources on the ns-2 [24] simulator; the fixed point technique converges in a few seconds compared to the O(min) duration necessary with simulations. To study the accuracy of our drift analysis, we simulated both RED (Random Early Detection) and ERD (Early Random Drop) [15] queues. The differences between these algorithms and the necessary corrections to our model (for RED) are presented in Appendix B.

A set of illustrative examples are presented in Figs. 3 and 4. In both simulations, two TCP connections, with 512 byte packets, interact with a single bottleneck queue. The random drop queue has $\mathrm{min}_{\mathrm{th}} = 25$ packets, $\mathrm{max}_{\mathrm{th}} = 75$ packets and buffer size equal to 150 packets (similar to recommendations in [25]); $p_{\max}$ was varied between the values outlined in the plots. Fig. 3 considers two TCP connections with identical round-trip times, while Fig. 4 shows the results when the nominal RTT of the second connection is double that of the first (called the BaseRTT in the figure). By varying $p_{\max}$, we change the slope of the drop function and hence, the 'zero-drift' point of the queue occupancy. Similar simulations have also been performed for a variety of round-trip times and MSSs. In general, the accuracy of our predictions is slightly lower for larger RTT values, although in all cases the agreement was within 10% of the predicted values. This is expected because a larger RTT essentially increases the chance of buffer underflow (which invalidates our model) by increasing the feedback time of the TCP control loop. Since our model does not account for phenomena like fast recovery (during which the queue size reduces), we tend to predict larger queue occupancies than those obtained via simulation.

To further illustrate the utility of our analytical technique in determining the relative sharing of the

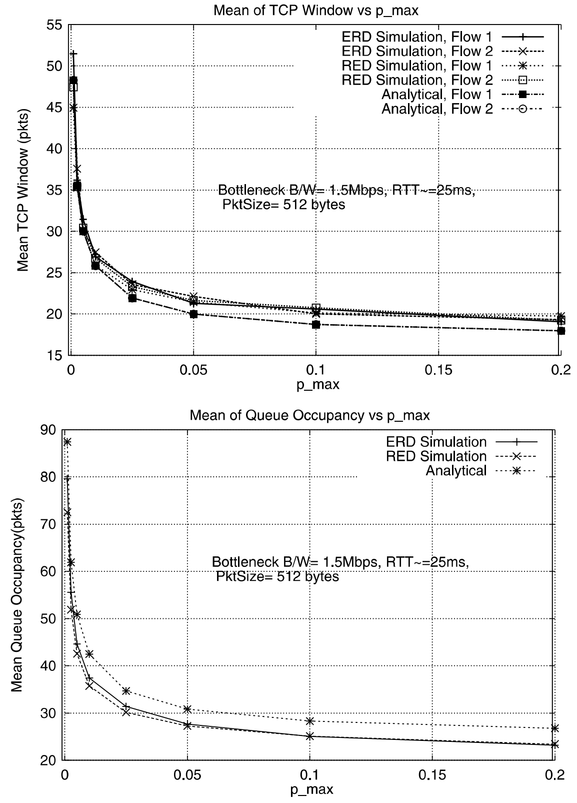

Fig. 3. Mean behavior with two identical connections.

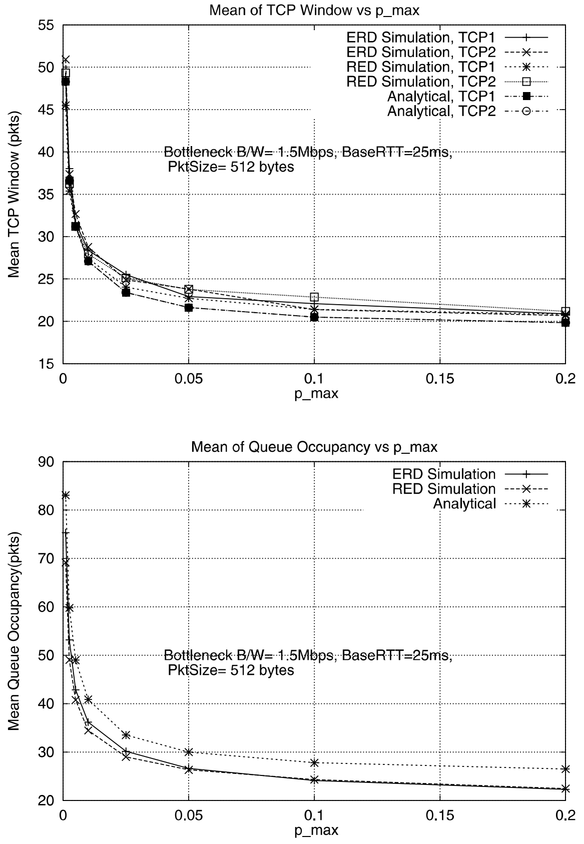

Fig. 4. Mean behavior with two dissimilar connections.

bottleneck capacity among the TCP flows, we consider four TCP flows sharing a bottleneck ERD buffer. The round-trip times of the four flows were 50, 70, 90 and 110 ms respectively. Table 1 shows the simulated and analytically predicted values for the throughput of all the four flows. We can again see that our analytical technique predicts the true sharing of TCP bandwidth with reasonable accuracy.

Our simulations also validate our analysis, which states that the means of the TCP windows (in segments) should be identical (in ack time), even though the round-trip times of the various flows and the segment sizes are different. It should also be noted that the negative correlation among window sizes (discussed shortly) helps to reduce the variation in packet loss probability and improves the accuracy of our technique.

Table 1
Results for 4 TCPs with different RTTs

TCP flow	Throughput analytical (Mbps)	Throughput simulated (Mbps)
1	0.451	0.441
2	0.392	0.370
3	0.346	0.352
4	0.311	0.315

3.6. Model accuracy for varying N

While the above examples clearly validate the fundamentals behind the fixed-point approach, it would be interesting to study the effect of our choice of the “square-root formula” on the accuracy of our model. To study this behavior, we performed extensive simulations by varying N, the

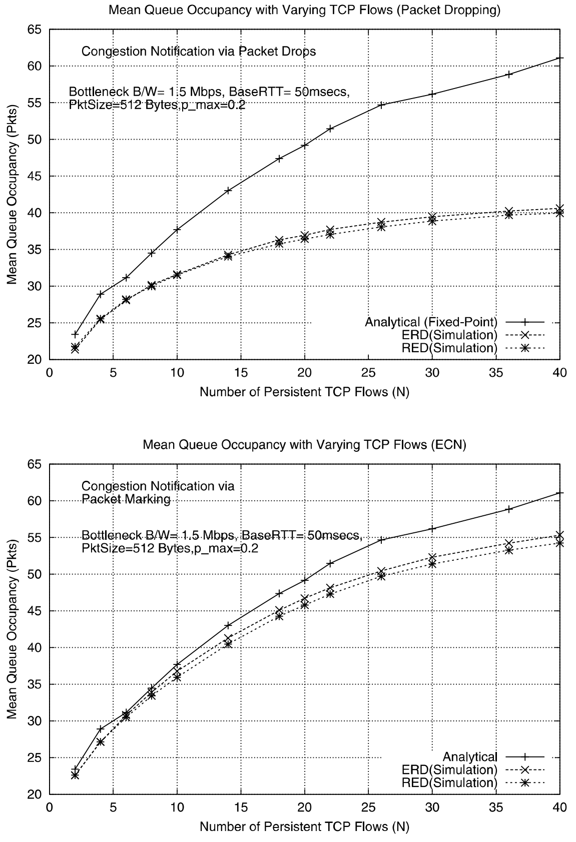

Fig. 5. Queue occupancy variation for router drop/marking.

number of persistent TCP flows. More importantly, we used simulations to obtain the difference between the simulated average queue occupancy when the router performed congestion feedback using either packet drops or packet marking (ECN). Fig. 5 plots the simulated queue occupancy of both RED and ERD queues, as well as the analytically predicted value using the fixed-point method. Since the fixed point method is based on an abstract concept of congestion notification, the analytic prediction is identical for both packet dropping and marking behavior. The plots presented in Fig. 5 correspond to a bottleneck buffer setting of $\min_{\text{th}} = 25$, $\max_{\text{th}} = 75$ and $p_{\max} = 0.2$ respectively. The TCP flows were grouped in pairs, with the RTT of the first pair set to 50 ms, and the RTT of every successive pair set to 1.2 times that of the previous pair.

It is easy to see that the analytic predictions over-estimate the actual occupancy (and consequently TCP throughput) when congestion notification is performed via packet drops. The reason for this is not hard to find—the square-root formulation fails to consider the effect of timeouts, which occur when TCP flows encounter packet *losses*. As N increases, the average queue length and the average packet dropping rate increases, leading to greater inaccuracy in the square-root based fixed-point model. When notification is performed via packet *marking*, such loss-related timeouts are extremely rare events, and congestion control is achieved principally through congestion avoidance. Accordingly, our analytic model proves reasonably accurate (within 10% in all cases) for ECN-based bottleneck queues, even when the feedback rate is high and number of TCP flows are fairly large. ECN-based marking is clearly the preferred mode of congestion notification in the future; moreover, the adoption of advanced mechanisms such as TCP SACK should further reduce the likelihood of TCP timeouts. Accordingly, the results demonstrate that the "square-root formula" based fixed-point technique is likely to be *increasingly useful as a predictor of network performance in the future.*

3.7. Negative window-size correlation and its consequences

On observing the results of our simulations, we found that the window sizes of the different flows were *negatively correlated.* Negative correlation essentially implies that the windows tend to vary out of phase: when the window size of one flow is large, the other flows have smaller than usual window sizes, and vice versa. The queue occupancy thus exhibits lower variability and tends to be less dependent on variations in the window size of a single flow. To demonstrate the presence of such correlation for an arbitrary number of flows, we sample the queue size and the individual windows to obtain the variance of the sum of the window sizes $\mathrm{Var}(\sum_{i=1}^{N} W_i)$ and the sum of the individual variances $\sum_{i=1}^{N}\mathrm{Var}(W_i)$. We know that the two should be equal if the flows are ideally

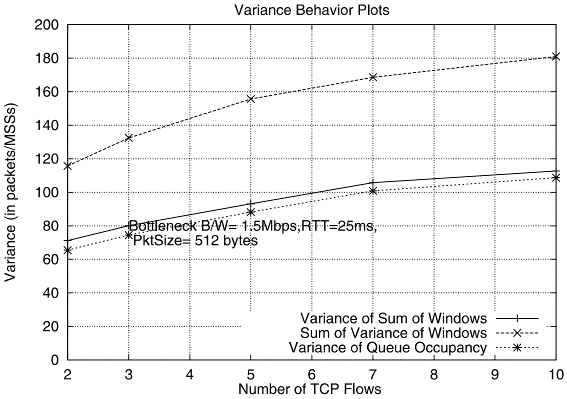

Fig. 6. Variance plots for TCP flows over an ERD queue.

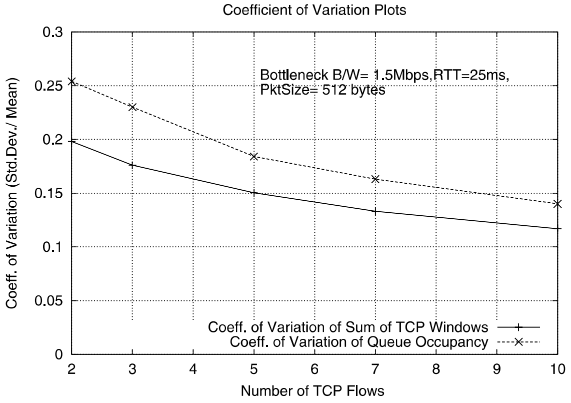

Fig. 7. Coefficient of variation behavior.

uncorrelated. For negative correlation, the sum should exhibit lower variance ($\mathrm{Var}(\sum_{i=1}^{N} W_i) < \sum_{i=1}^{N} \mathrm{Var}(W_i)$), while for positive correlation, the sum should exhibit larger variance ($\mathrm{Var}(\sum_{i=1}^{N} W_i) > \sum_{i=1}^{N} \mathrm{Var}(W_i)$). This follows from the general relationship

$$\mathrm{Var}\left(\sum_{i=1}^{N} W_i\right) = \sum_{i=1}^{N} \mathrm{Var}(W_i) + \sum_{i \neq j} \mathrm{Cov}(W_i, W_j). \tag{3.18}$$

Hence, if the covariance terms are negative, then the LHS of Eq. (3.18) is less than the RHS.

Fig. 6 shows the behavior of these statistical metrics for different values of N, the number of simultaneous flows sharing a bottleneck buffer. The figure shows that $\mathrm{Var}(\sum_{i=1}^{N} W_i)$ is always less than $\sum_{i=1}^{N} \mathrm{Var}(W_i)$ (and, in fact, $\mathrm{Var}(Q)$ is even lower than $\mathrm{Var}(\sum_{i=1}^{N} W_i)$). This indicates the presence of 'negative correlation' among the TCP flows. Ref. [8] shows how such a negative correlation can be exploited to obtain more accurate estimates of the stationary distribution of the congestion windows, while [10] discusses how such negative correlation is exploited by intelligent 'drop-biasing' strategies to reduce the variability of the queue occupancy. Fig. 7, on the other hand, shows that the coefficient of variation of the queue occupancy decreases with an increase in the number of simultaneous TCP flows. Accordingly, we can expect our analytical predictions to be more accurate in the presence of a larger number of flows, as long as the presence of a larger number of flows does not increase the *drop* probability and the incidence of retransmission timeouts.

4. Analysis extension for generalized congestion avoidance

We now extend the fixed-point technique to compute the mean window sizes and throughputs for TCP flows performing generalized congestion avoidance. In general, a process that performs window-based congestion control under the TCP paradigm can be thought of as increasing its window by a function $\mathrm{incr}(W)$ on receiving an acknowledgment in the absence of congestion and decreasing its window by $\mathrm{decr}(W)$ on receiving an acknowledgment indicating congestion. As stated earlier, we consider a special case of window adjustment where:

$$\mathrm{incr}(W) = c_1 W^{\alpha}, \quad \mathrm{decr}(W) = c_2 W^{\beta},$$

where α, β, c_1 and c_2 are arbitrary constants. The class of algorithms having $\alpha = 0$ and $\beta = 1$, are called additive-increase, multiplicative-decrease (AIMD) algorithms in literature. We shall also refer to the class of algorithms having $\alpha = -1$ and $\beta = 1$ as sub-additive increase, multiplicative-decrease (SAIMD) algorithms in the remainder of this paper.

To demonstrate the utility of our fixed-point analysis to a variety of problems, we tailor our analysis to the Assured Service model and consider the interaction with an ORED buffer. As we shall see shortly, this is a relatively harder case, since the marking probability for packets from a flow is not simply dependent on the queue occupancy, but on the flow's window size as well. (The case of generalized TCP flows interacting with a random notification buffer without any minimum assured rate is a simpler version of this problem and follows almost immediately.) Also, the ORED buffer marks (sets the ECN bit) on only *out* packets. Since *in* packets are never marked, the only possible form of congestion indication experienced by *in* packets occurs due to losses during buffer overflow. The model thus essentially assumes that marking *out* packets with a sufficiently aggressive probability is adequate to ensure that a congestion window does not grow without limit. Mathematically speaking, this will be true as long as $\lim_{w\uparrow\infty} \mathrm{incr}(W)/\mathrm{decr}(W) \to 0$. i.e., while $\alpha < \beta$, which is true in all practical cases of interest.

4.1. Formulating the fixed point equations

As before, we define the *drift* in the congestion window of the ith flow by the expected change, ΔW_i, in its window size as a function of its window size W_i. The window size increases by $c_1 W_i^{\alpha}$ with a probability $1 - p_i(W)$ and decreases by $c_2 W_i^{\beta}$ with a probability $p_i(W)$, where $p_i(W)$ is the probability of a packet being marked (ECN bit set). Thus, the drift is 0 (corresponding to the 'mean' or center of the window) when W_i satisfies the condition

$$c_1 W_i^{\alpha}(1 - p_i(W_i)) = c_2 W_i^{\beta} p_i(W_i). \tag{4.1}$$

Accordingly, given a specific function $p_i(\cdot)$, we can obtain the mean value of the congestion window by solving

$$\frac{c_2}{c_1} W_i^{\beta-\alpha} = \frac{1 - p_i(W_i)}{p_i(W_i)}. \tag{4.2}$$

Clearly, relation (4.2) defines a set of N equations for $i = 1, \ldots, N$.

If the mean ORED buffer occupancy is Q (bytes), we can determine the corresponding function $p_i(\cdot)$. In this case, the marking probability for *out* packets is given [6] by $f(Q)$. Now, if a fraction γ_i of the packets from flow i are marked as *out*, the unconditional marking probability for packets of flow i is $\gamma_i f(Q)$. Unfortunately, when more than 1 TCP flow is present, γ_i is *itself a function of both* W_i and Q. To see this, note that, when the queue occupancy is Q, the total round-trip time for flow i is given by $\mathrm{RTT}_i + Q/C$. Since the flow control algorithm transmits $W_i M_i$ bytes every round-trip time, the achieved throughput ρ_i is given by

$$\rho_i = \frac{W_i M_i}{\mathrm{RTT}_i + Q/C}. \tag{4.3}$$

The probability of a packet being tagged as *out* is assumed to be equal to the fraction by which the achieved throughput exceeds the assured rate R_i. γ_i is thus given by $\gamma_i = (\rho_i - R_i)/\rho_i$ or, upon using Eq. (4.3):

$$\gamma_i = 1 - \frac{R_i(\mathrm{RTT}_i + Q/C)}{W_i M_i}. \tag{4.4}$$

Accordingly, the marking probability $p_i(W_i)$ is given by

$$p_i(W_i) = \left(1 - \frac{R_i(\mathrm{RTT}_i + Q/C)}{W_i M_i}\right) f(Q),$$

which on substituting into Eq. (4.2) yields the following relationship (one for each $i = (1, \ldots, N)$)

$$\frac{c_2}{c_1} W_i^{\beta-\alpha} = \left(1 - \frac{R_i(\mathrm{RTT}_i + Q/C)}{W_i M_i} f(Q)\right)^{-1} - 1. \tag{4.5}$$

We denote the solution for W_i of the above equation as $h_i(Q)$ to explicitly indicate that the above equation is really a function of the queue occupancy Q. We shall elaborate on a technique for solving the above equation (to obtain $h_i(Q)$) in Section 4.2.

Given a value for Q, we can then (at least in principle) solve the set of N equations (Eq. (4.5) for $i = 1, \ldots, N$) to obtain the N values, $h_i(Q)$,

[6] As stated earlier, our formulation can also be used when different flows have marking probabilities that are scalar multiples of each other, i.e., $f^i(Q) = \kappa_i f(Q)$ where κ_i are arbitrary constants.

$i = 1, \ldots, N$. However, our solution must satisfy another constraint: in the absence of buffer underflow, the sum of the throughputs of the N flows must equal the link capacity C, i.e., $\sum_{i=1}^{N} \rho_i = C$. For a specific value of Q, we note that $\rho_i = h_i(Q)M_i/(\mathrm{RTT}_i + Q/C)$ and hence, after trivial algebraic manipulations arrive at the other constraint:

$$\sum_{i=1}^{N} \frac{h_i(Q)M_i}{Q + \mathrm{RTT}_i * C} = 1. \tag{4.6}$$

The basis of our fixed-point theory should now be clear. As we vary Q and solve for the $h_i(Q)$ according to expression (4.5), there will be one value for which the constraint (4.6) is satisfied. This value of the queue occupancy is denoted by Q^*. The corresponding solutions for $h_i(Q^*)$ provides the theoretical mean window sizes W_i^*; the corresponding throughput for connection i is then computed by $W_i^* M_i/(\mathrm{RTT}_i + Q^*/C)$.

4.2. Existence and solution of fixed point

The existence of a unique solution can be verified by varying Q from $\min_{\mathrm{th}}$ to ∞. At values close to $\min_{\mathrm{th}}$, $f(Q) \approx 0$ and hence, from Eq. (4.5), we see that $h_i(Q)$ will be very large. Accordingly, the LHS of Eq. (4.6) will be much larger than 1. On the other hand, as $Q \uparrow \infty$, the value of $h_i(Q)$ also increases (since it is clearly always larger than $R_i(\mathrm{RTT}_i + Q/C)$). In that case, if we neglect the constant term of 1 in the RHS of Eq. (4.5), we can easily see, after elementary manipulation, that the expression (4.5) reduces to

$$\frac{c_2 M_i}{c_i} W_i^{\beta-\alpha} = \frac{c_2}{c_1} R_i \left(\mathrm{RTT}_i + \frac{Q}{C} \right) W_i^{\beta-\alpha-1} + M_i, \tag{4.7}$$

which, for large values of Q and W_i, yields

$$W_i M_i = h_i(Q) M_i \approx R_i \left(\mathrm{RTT}_i + \frac{Q}{C} \right). \tag{4.8}$$

By plugging expression (4.8) into the LHS of constraint (4.6), we can see that the LHS turns out to be equal to $\sum_{i=1}^{N} R_i/C$. But by our assumption (2.3), this is clearly less than 1. We can further show that as Q increases from $\min_{\mathrm{th}}$ to ∞, the LHS of (4.6) decreases monotonically and crosses 1 at some point. Such a value of Q accordingly defines the unique solution of the fixed point.

Our algorithm for solving the fixed point essentially consists of varying Q and solving for $h_i(Q)$ until the condition (4.6) is satisfied.

An iterative gradient scheme (based on the Newton method) can be used to solve for $h_i(Q)$. A value of W_i that satisfies Eq. (4.5) is essentially the unique zero of the function $g(W)$ defined by

$$\left(1 - \frac{R_i(\mathrm{RTT}_i + Q/C)}{W_i M_i} f_{\mathrm{mark}}(Q) \right)^{-1} - 1 - \frac{c_2}{c_1} W_i^{\beta-\alpha} \tag{4.9}$$

Define

$$g_1(W_i) = \left(1 - \frac{R_i(\mathrm{RTT}_i + Q/C)}{W_i M_i} f_{\mathrm{mark}}(Q) \right)^{-1} - 1$$

and

$$g_2(W) = \frac{c_2}{c_1} W_i^{\beta-\alpha}.$$

By taking derivatives, we can see that $g_1(W_i)$ is convex and decreasing in W_i while $g_2(W_i)$ is increasing in W_i (since $\beta > \alpha$). Furthermore, if $\beta - \alpha < 1$, then $g_2(W_i)$ is also concave. Accordingly, we start with a value of W_i slightly larger than $R_i(\mathrm{RTT}_i + Q/C)$ and repeat the iterations until we converge. In particular, if $\beta - \alpha \leqslant 1$, $g(W_i)$ is convex and hence, we can guarantee convergence without any overshoot. When $\beta - \alpha > 1$, we have the possibility of overshoot and hence, need to take special care in our numerical procedure. However, in all our numerical calculations, we were able to attain convergence using the Newton iterative method using the iteration

$$W_i^{j+1} = W_i^j + \frac{g(W_i^j)}{g'(W_i^j)}. \tag{4.10}$$

Here, $g'(W_i^j)$, the derivative of $g(W)$, is given by the expression:

$$g'(W_i) = \frac{-R_i(\mathrm{RTT}_i + Q/C)}{f(Q) W^2 M_i (1 - R_i(\mathrm{RTT}_i + Q/C)/W_i M_i)^2} - \frac{c_2}{c_1} (\beta - \alpha) W_i^{\beta-\alpha-1}.$$

The appropriate value for Q i.e., Q^*, on the other hand, can be obtained by a binary search procedure, since we have established that $\sum_{i=1}^{N} h_i(Q)M_i/(\mathrm{RTT}_i + Q/C)$ is monotonically decreasing and smaller than C when $Q > Q^*$ and larger than C when $Q < Q^*$. Thus, the entire algorithm consists of two loops: an outer loop varying Q via a binary search method and an inner loop evaluating $h_i(Q)$ via the Newton gradient method.

4.3. Simulations and comparative results

We performed fairly extensive tests using ns-2 to compare the accuracy of our analytical/numerical results with those obtained via simulations. Our modifications to ns-2 included incorporation of the generalized incr(W) and decr(W) functions in the TCP code and augmentation of the RED code to implement the ORED mechanism.

For ease of illustration, we principally present plots for the case of only two generalized flows. (We have however used between 2 and 20 TCP flows in additional simulations to verify the accuracy of our technique.) Both flows had the same segment size of 512 bytes. To provide illustrative results, we use four parameter sets.

1. Parameter set 1: ($\alpha = -1$, $\beta = 1$, $c_1 = 1$ and $c_2 = 0.5$), i.e., the current TCP window adaptation procedure.
2. Parameter set 2: ($\alpha = 0$, $\beta = 1$, $c_1 = 0.2$ and $c_2 = 0.1$), i.e., an interesting choice of AIMD parameters.
3. Parameter set 3: ($\alpha = -1$, $\beta = 1$, $c_1 = 0.5$ and $c_2 = 0.1$), i.e., SAIMD with a reduction in the coefficients for window increase and decrease.
4. Parameter set 4: ($\alpha = 0$, $\beta = 1$, $c_1 = 0.4$ and $c_2 = 0.2$), i.e., AIMD with larger coefficients for window increase and decrease than parameter set 2.

The link capacity was varied between 4.5 and 12 Mbps. While $\min_{\mathrm{th}}$ and $\max_{\mathrm{th}}$ was maintained at 20 and 100 respectively for both parameter sets, $p_{\max}$ was kept at 0.01 for parameter set 1 and 3, and at 0.1 for parameter sets 2 and 4. This was done to ensure reasonable mean window sizes: for identical marking probabilities, parameter sets 2 and 4 would have much larger mean window sizes than parameter sets 1 and 3. We present here the results of two different experiments.

In the first set of experiments, which we shall call *Experiment A*, we kept the round-trip times identical for both flows but provided them different profiled rates. TCP flow 1 had a profile of 1.5 Mbps and TCP 2 had a profile of 3 Mbps. Both flows were tagged by a leaky bucket-based conditioner with a moderate bucket size of 20 packets. Fig. 8 shows the theoretical and simulated TCP mean window sizes/throughputs for parameter set 1 as the link capacity C is varied. Fig. 9 shows the corresponding plots for parameter set 2 (we do not provide plots for the other parameter sets due to space limitations). The figures show close agreement between our analytical predictions and the simulated results. We conducted similar experiments where N varied from 2 to 20; our predictions

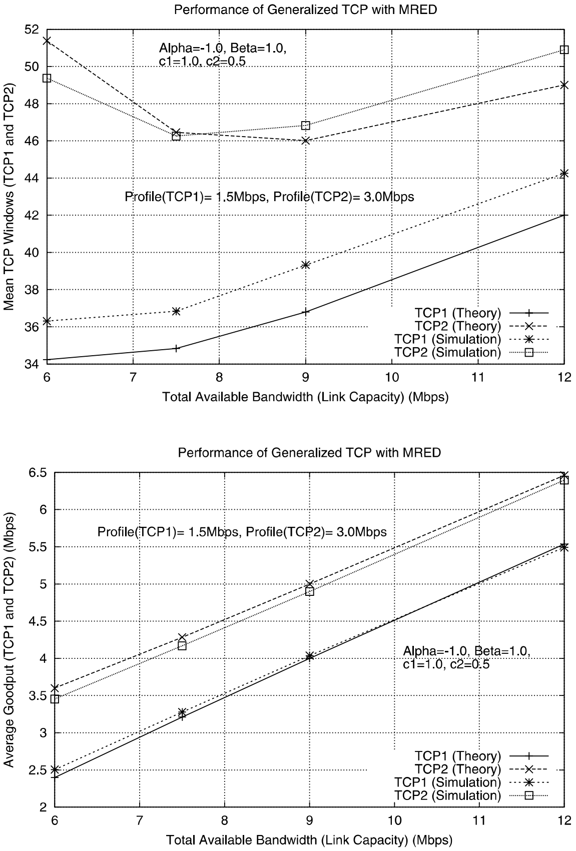

Fig. 8. Mean window sizes and throughputs for parameter set 1 (different rate profiles).

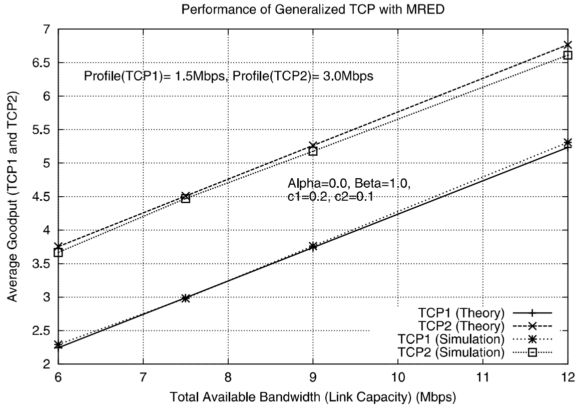

Fig. 9. Mean window sizes and throughputs for parameter set 2 (different rate profiles).

were always within 5% of the values obtained via simulations.

In the second set of experiments, which we shall call *Experiment B*, the two TCP flows had identical profiled rates (1.5 Mbps) but different round-trip times. Flow 1 had an RTT of 20 ms while flow 2 had an RTT of 100 ms. Fig. 10 shows the theoretical and simulated TCP mean window sizes/throughputs for parameter set 1 as the link capacity C is varied; we see the close agreement between the analytical predictions and the simulated values. Similar agreement is obtained with the other parameter sets; we omit the figures due to space constraints.

4.3.1. Accuracy of technique for larger N

We have also studied the accuracy of our model for larger values of N under a variety of settings. Table 2 shows the predicted and simulated values of the goodput for $N = 10$ flows. In this particular study, each odd flow has a guaranteed (assured) rate of 400 Kbps and each even flow has an assured rate of 800 Kbps, while the channel bandwidth is set to 9 Mbps. Moreover, flows were grouped in pairs, with the first pair having an RTT of 25 ms, and the RTT of each subsequent pair being set to 1.6 times the RTT of the previous pair. We tabulate the results for two of the parameter sets enumerated earlier: for parameter set 1 (TCP), the ORED buffer parameters were $\min_{th} = 25$, $\max_{th} = 75$ and $p_{\max} = 0.1$, while for parameter set 2, the buffer parameters were $\min_{th} = 20$, $\max_{th} = 100$ and $p_{\max} = 0.1$.

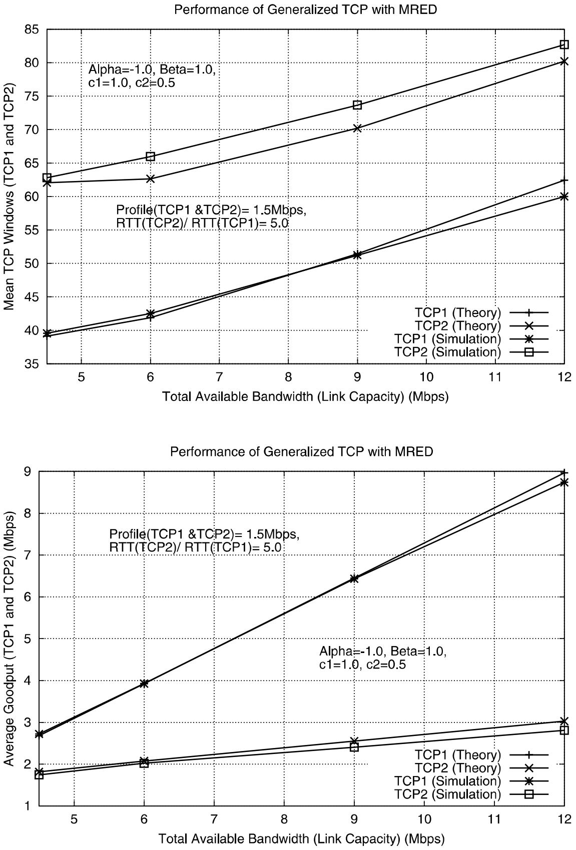

Fig. 10. Mean window sizes and throughputs for parameter set 1 (different RTT values).

Table 2
Mean TCP goodputs for $N = 10$

TCP flow	Parameter set 1 ($\alpha = -1$)		Parameter set 2 ($\alpha = 0$)	
	Analytical (Mbps)	Simulated (Mbps)	Analytical (Mbps)	Simulated (Mbps)
1	1.095	1.108	0.759	0.763
2	1.362	1.392	1.165	1.160
3	0.873	0.889	0.736	0.744
4	1.160	1.149	1.142	1.132
5	0.696	0.682	0.705	0.712
6	1.006	0.998	1.110	1.114
7	0.567	0.554	0.667	0.660
8	0.905	0.887	1.071	1.066
9	0.485	0.481	0.622	0.616
10	0.849	0.806	1.025	1.004

The close agreement between the simulated and predicted values validates the applicability of our technique across a wide variety of operating parameters. Ref. [10] also uses this approach to consider how changes to the adaptation parameters affect the bandwidth sharing paradigm between multiple TCP flows. Such studies are useful in evaluating the possible effects of suggested changes to TCP's current congestion control algorithm. Of course, as stated earlier, this 'square-root' based approach works only when retransmission timeouts are rare events. When packet losses are the sole means of congestion notification, this technique is accurate only as long as the loss probabilities are relatively small (below $\approx$5%). However, when packet marking mechanisms, such as ECN, are used, packet losses are relatively infrequent events; accordingly, our analysis holds over a much wider range of marking probabilities.

5. Conclusions

In this paper, we have demonstrated an analytical and numerical technique to obtain the centers of the TCP window sizes and the associated queue occupancy when multiple persistent TCP flows share a bottleneck buffer performing randomized congestion feedback. The technique essentially uses a drift-based argument to relate the center of a TCP window performing congestion avoidance to the average packet marking/dropping probability. We use a set of fixed-point conditions, which seek an operating point for the buffer where the drop/marking probability is consistent with the sizes of the individual TCP windows. While we use the "square-root" formula as a specific instance of our fixed-point formulation, the mechanism is general enough to incorporate other models for TCP response to congestion notification. For simple cases, where an individual flow is not subject to any rate-constraints, we prove how the use of a Newton gradient-based technique results in fast convergence.

We subsequently extend the technique to consider TCP flows performing generalized congestion avoidance, where the flow increases and decreases its congestion window by $c_1 W^\alpha$ and $c_2 W^\beta$ respectively. As an example of a more complicated model, we consider the case of an Assured Service framework and derive the mean window sizes and throughputs under this model. The solution technique now combines binary search in an outer loop with gradient-based iterative search in an inner loop. Simulation results attest to the accuracy of our analysis technique.

While performing our simulations, we had observed the presence of negative correlation among the TCP windows. Mechanisms that indirectly exploit this negative correlation to smoothen the buffer occupancy have been suggested in [11] and should be explored in greater detail. Our analysis does not consider the effects of transients such as fast recovery and timeouts. Accordingly, our fixed point approximation technique was seen to provide very accurate results, over a wide variety of loads and operating conditions, when ECN-based *marking* was used to provide congestion feedback. When congestion notification is achieved via randomized packet *dropping*, our technique suffers from a degradation in accuracy when the average packet loss rate exceeds $\sim$5%.

The numerical technique presented here appears to be a promising way to develop hierarchical fixed point algorithms that provide reasonably accurate estimates of network behavior at a fraction of the cost of detailed simulations. In the future, we hope to explore the use of such techniques for determining the operational point of packet-based networks, especially in the presence of multiple bottlenecks.

Appendix A. Proof that *s(Q)* is convex

We prove here that the function $s(Q)$ defined in Eq. (3.16) is convex. First, some notation: let $\frac{M_i}{c_i}$ be denoted by b_i and $C * \mathrm{RTT}_i$ be denoted by d_i. The function $g(Q)$ is then given by $g(Q) = (\sum_i b_i/(Q + d_i))^{-1}$. On differentiating this function we obtain

$$g'(Q) = g(Q)^2 \sum_i \frac{b_i}{(Q + d_i)^2}. \tag{A.1}$$

Since from above, $g'(Q) > 0\ \forall Q$, $g(Q)$ is an increasing function of Q. Differentiating again, we have the second derivative given by

$$g''(Q) = 2g(Q)g'(Q)\sum_i \frac{b_i}{(Q+d_i)^2} - 2(g(Q))^2 \sum_i \frac{b_i}{(Q+d_i)^3},$$

or on rearranging

$$g''(Q) = 2(g(Q))^3 \left\{ \left(\sum_i \frac{b_i}{(Q+d_i)^2} \right)^2 - \left(\sum_i \frac{b_i}{(Q+d_i)^3} \right) \left(\sum_i \frac{b_i}{Q+d_i} \right) \right\}. \tag{A.2}$$

We now prove that the term in the curly braces in Eq. (A.2) is negative. To see this, let $\beta = \sum_i b_i$ and let $a_i = (Q + d_i)\ \forall i \in \{1, 2, \ldots, N\}$ (note that a_i is always positive). Consider a random variable A which takes on the value a_i with probability $\pi_i = b_i/\beta$. Then, the second derivative can also be written (with $E[]$ denoting the expectation operation) as

$$g''(Q) = 2\beta^2 (g(Q))^3 \{E^2[A^2] - E[A^3]E[A]\}. \tag{A.3}$$

Now, we know if A is a random variable that has $\text{Prob}(A > 0) = 1$, then $\log E[A^m]$ is convex in m $\forall m \geqslant 0$. Thus, we have $\log E[A^2] \leqslant (\log E[A] + \log E[A^3])/2$, so that $E^2[A^2] - E[A^3]E[A] \leqslant 0$. Applying this result to expression Eq. (A.3), we see that $g''(Q)$ is negative and hence, $g(Q)$ is a *concave* function of Q.

As the term $\sqrt{2/f(Q)}$ is easily seen to be convex (its second derivative is positive), we can conclude that $s(Q)$ is a convex function of Q.

Appendix B. Modeling RED behavior

In this appendix, we discuss the applicability of our mode to ERD and the RED queues. The important differences between RED and ERD are:

- RED operates on the average (and not the instantaneous) queue length. The drop probability, p, is thus a function of the weighted average Q_{avg} of the queue occupancy, i.e., p is a function not just of Q_n but of $(Q_n, Q_{n-1}, Q_{n-2}, \ldots)$ with an exponential decay.
- To avoid unbounded inter-drop gaps, RED increases the drop probability for every accepted packet. This property, which we call *drop-biasing*, is achieved by using a variable, cnt, which increments with every successive accepted packet; the true dropping probability is then given by $p(Q)/(1 - \text{cnt} * p(Q))$. This results in a inter-drop period that is *uniformly* distributed between $(1, \ldots, \lfloor 1/p(Q) \rfloor)$ as opposed to the *geometrically* distributed inter-drop gap caused by an independent packet drop model.
- Some RED implements have $f(Q) = 1$ when Q_{avg} exceeds $\max_{\text{th}}$; this contrasts with our assumption of random drop throughout the entire range of the buffer occupancy. Our RED queues however have $f(Q) = p_{\max}$ for Q_{avg} larger than $\max_{\text{th}}$.

To capture the effects of drop-biasing in RED, we change the function $f(Q)$ such that the average inter-drop gap is the same for both RED $(1/2p)$ and ERD $(1/p)$. We achieve this by setting the $p_{\max}$ value in RED simulations to half that used for ERD simulations and in our analytical technique.

Appendix C. Correction for delayed acknowledgments

Delayed acknowledgments essentially imply that the TCP process increments its window only once for every K (K is usually 2) acknowledgments. A simple way to capture this effect is to alter Eq. (3.1) to

$$P\left\{ W_i^{n+1} = w + \frac{c_1 W^{\alpha}}{K} \,\middle|\, W_i^n = w \right\} = 1 - p_i(w), \tag{C.1}$$

i.e., approximate window evolution by a process that increments its window by $c_1 W^\alpha / K$ for every congestion-free acknowledgment. Accordingly, the zero-drift condition in Eq. (4.1) becomes:

$$c_1 W_i^\alpha (1 - p_i(W_i)) = K c_2 W_i^\beta p_i(W_i). \tag{C.2}$$

References

[1] V. Jacobson, M. Karels, Congestion avoidance and control, in: Proceedings of ACM SIGCOMM, September 1988.

[2] S. Floyd, V. Jacobson, Random early detection gateways for congestion avoidance, IEEE/ACM Transactions on Networking 1 (4) (1993) 397–413.

[3] K.K. Ramakrishnan, S. Floyd, A Proposal to add explicit congestion notification (ECN) to IP, RFC 2481, IETF, January 1999.

[4] D. Bansal, H. Balakrishnan, Binomial congestion control algorithms, in: Proceedings of IEEE INFOCOM'01, April 2001.

[5] D. Clark, W. Fang, Explicit allocation of best effort packet delivery service, IEEE/ACM Transactions on Networking 6 (4) (1998) 367–373.

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

[7] T.V. Lakshman, U. Madhow, B. Suter, Window-based error recovery and flow control with a slow acknowledgment channel: a study of TCP/IP performance, in: Proceedings of IEEE INFOCOM'97, April 1997.

[8] A. Misra, T. Ott, J. Baras, The window distribution of multiple TCPs with random loss queues, in: Proceedings of IEEE GLOBECOM'99, December 1999.

[9] V. Firoiu, M. Borden, A study of active queue management for congestion control, in: Proceedings of IEEE INFOCOM'00, March 2000.

[10] A. Misra, J. Baras, T. Ott, Generalized tcp congestion avoidance and its effect on bandwidth sharing and variability, in: Proceedings of IEEE GLOBECOM'00, December 2000.

[11] A. Misra, T. Ott, J. Baras, Using 'Drop-Biasing' to stabilize the occupancy of random-drop queues with tcp traffic, in: Proceedings of IEEE International Conference on Communication Systems (ICCS), November 2000.

[12] A. Misra, T. Ott, Effect of exponential averaging on the variability of a RED queue, in: Proceedings of IEEE International Conference on Communications (ICC), June 2001.

[13] T. Ott, S. Lakshman, L. Wong, SRED: stabilized RED, in: Proceedings of IEEE INFOCOM'99, March 1999.

[14] W. Feng, D. Kandlur, D. Saha, K. Shin, Blue: An alternative approach to active queue management, in: Proceedings of NOSSDAV 2001, June 2001.

[15] E. Hashem, Analysis of random drop for gateway congestion control, MIT Tech Report, MIT-LCS-TR-506.

[16] S. Floyd, Connections with multiple congested gateways in packet-switched networks. Part 1: One-way traffic, Computer Communication Review (1991).

[17] M. Mathis, J. Semke, J. Mahdavi, T. Ott, The macroscopic behavior of the TCP congestion avoidance algorithm, Computer Communications Review (1997).

[18] A. Kumar, Comparative performance analysis of versions of TCP in a local network with a lossy link, IEEE/ACM Transactions on Networking 6 (4) (1998) 485–498.

[19] J. Padhye, V. Firoiu, D. Towsley, J. Kurose, Modeling TCP throughput: a simple model and its empirical validation, in: Proceedings of ACM SIGCOMM'98, September 1998.

[20] T. Bu, D. Towsley, Fixed point approximations for TCP behavior in an AQM network, in: Proceedings of ACM SIGMETRICS'00, September 2000.

[21] S. Floyd, Recommendations on using the 'gentle' variant of RED, March 2000. Available from http://www.aciri.org/floyd/red.html.

[22] M. Schwartz, Broadband Integrated Networks, Prentice Hall, Englewood Cliffs, NJ, 1997.

[23] T. Ott, M. Matthis, J. Kemperman, The stationary behavior of idealized congestion avoidance, August 1996. Available from ftp://ftp.telcordia.com/pub/tjo/TCPwindow.ps.

[24] The ns-2 network simulator. Available from http://www-mash.CS.Berkeley.EDU/ns.

[25] S. Floyd, Notes on RED in the end-to-end-interest mailing list, 1998.

Archan Misra is currently a Research Staff Member with the Pervasive Security and Networking Department at the IBM TJ Watson Research Center, Hawthorne, NY. He is presently working on next-generation Wireless Internet services, autonomic connectivity and peer discovery algorithms for mobile computing, mobility management techniques for pervasive devices and distributed middleware for location and context aware programming. Before joining IBM in March 2001, Archan spent 3.5 years at Telcordia Technologies (formerly called Bellcore), where he was responsible for several initiatives in the areas of IP-based mobility management, congestion control, QoS architectures and autoconfiguration of heterogeneous networks. As part of his research efforts, Archan co-invented the IDMP mobility management and the DCDP autoconfiguration protocols. He has published more than forty research papers in the areas of wireless networking, congestion control and mobility management and received the Best Paper award in IEEE MILCOM 2001. Archan received his Ph.D. in Electrical and Computer Engineering from the University of Maryland at College Park, 2000, and his B.Tech in Electronics and Communication Engineering from IIT Kharagpur, India in July 1993.

Teunis J Ott, since September 2001 is a Professor in the Department of Computer Science at the New Jersey Institute of Technology, Newark, NJ. Before moving to NJIT, he spent 23 years in Bell Laboratories, Bellcore, and Telcordia as Member of Technical Staff, Supervisor, Director, and Senior Scientist. In the last few years, he has been performing research on Internet Routing and on modeling of transport protocols, in particular TCP and aspects of ECN and received the best paper award in IEEE MILCOM 2001. For the audience of this journal his best known publication is probably "The Stationary Distribution of Ideal TCP Congestion Avoidance" (August 1996), with Joop Kemperman and Matt Mathis (http://web.njit.edu/~ott etc.), which is often cited and exists only on the web. Dr. Ott holds four patents on Telephony Routing and aspects of IP Traffic.

John S. Baras holds the Lockheed Martin Chair in Systems Engineering at the University of Maryland at College Park. He is also the Founding Director of the Center for Hybrid and Satellite Communication Networks (CHSCN), a NASA center for the commercial development of space, as well as a Professor in the Department of Electrical and Computer Engineering at the University of Maryland. Dr. Baras has published numerous journal and conference papers in the areas of stochastic control and scheduling, risk-sensitive and intelligent control, wavelets and real-time signal processing algorithms, and satellite and hybrid communication systems. He is on the editorial boards of several journals, including the IMA Journal of Mathematical Control and Information. Dr. Baras received his M.S. and Ph.D. degrees in Applied Mathematics from Harvard University in 1971 and 1973 respectively, and a B.S. in Electrical Engineering from the National Technical University of Athens, Greece in 1970.