Discrete-Time Modeling of TCP Reno Under Background Traffic Interference with Extension to RED-Based Routers*

Ahmed E. Kamal
Department of Electrical and Computer Engineering
Iowa State University
Ames, IA 50011-2252
U.S.A.
E-mail: kamal@iastate.edu
Telephone: (515) 294-3580
Fax: (515) 294-1152

Abstract

This paper introduces a discrete-time model which captures the essential protocol features of the congestion control mechanism used by the TCP Reno protocol, subject to interference from other sources. Under this model, a single target session is modeled according to the TCP Reno mechanism, including slow start, congestion avoidance, fast retransmit and fast recovery. At the same time, other sources are modeled as a background process using a discrete batch Markov arrival process (D-BMAP). The D-BMAP process has been modified such that the transitions between the phases are dependent on the number of lost packets from the background process. This introduces a feedback process, which can be used to model an aggregation of TCP sources. In order to capture all the TCP Reno protocol features, two levels of Markov process modeling are used: a microscopic level, at the packet transmission time boundaries, and a macroscopic one, at the start of the new transmission windows. In addition, it is shown how the model can be extended to model networks with RED-based routers. Several performance measures are derived, and numerical examples which demonstrate the protocol features are presented.

I Introduction

The Transmission Control Protocol (TCP) [1, 2] is the reliable transport layer protocol of the Internet. It is part of the TCP/IP suite of protocols, and guarantees error free, and sequenced data delivery between the end systems. One of the important functions of the TCP protocols is the congestion control function, which in part tries to avoid congestion, and in another part responds to congestion by reducing the source's transmission rate. As far as this congestion control function is concerned, there are several versions of the TCP protocol, most importantly, Tahoe and Reno [3]. These versions differ in how they detect packet losses, and how they react to such situations.

Since congestion is detected by the TCP protocol upon the loss of a packet due to the lack of

*This research was supported in part by a Carver Trust Grant from Iowa State University.

buffer inside the network[1], different active queue management protocols have been proposed and implemented to signal the occurrence of congestion to the end systems. Most notably, the Random Early Detection (RED) [4] is the one implemented several commercial routers. With RED, on the onset of congestion, the router drops packets randomly so that congestion can be signaled to the sources, and they take appropriate actions to reduce the traffic they feed to the network.

The purpose of this paper is to introduce an accurate performance model for the TCP Reno version in the presence of interference traffic, which may cause congestion in common routers. Although several models of TCP Reno have been introduced before, this model is different in several aspects. The model keeps track of the exact way in which windows evolve under TCP Reno in a target source, and without making any assumptions about the loss process. In addition, interfering traffic is assumed and is modeled aggregately by a discrete batch Markovian arrival process (D-BMAP) [5], which is modified to react to losses[2] This process can be used to model UDP traffic, a collection of TCP sources, etc. The packet dropping mechanism is therefore modeled *exactly*, without assuming any random, or independent packet loss distribution. That is, a packet will be only dropped when the router is congested.

The paper is organized as follows. The next section provides some background material, in terms of the protocol description, and an overview of the relevant work in the literature. The following section introduces the model, while Section IV shows how the transition probabilities can be derived. Section V shows how this model can be extended to model RED-based routers. An approximate approach is also introduced. The evaluation of the performance measures of interest is described in Section VI, while Section VII introduces several numerical examples. Finally, Section VIII concludes the paper with a few remarks.

For ease of reference, we include a table of symbols in an appendix.

II Background

In this section we present some background material in terms of the protocols, the models and related work.

II.1 The TCP Congestion Control Mechanism:

The TCP protocol is an end-to-end protocol, in the sense that the protocol data is only handled by the sender and the receiver, and not by the network [1]. Every segment that is transmitted by the

[1]Flow problems can cause packets to be also discarded, but by the end systems. In this paper we do not consider the effect of flow problems, since the receivers can adjust their advertised window sizes to overcome this problem.

[2]It is to be noted that this approach was proposed by the author independently in [6]. Similar approaches in which continuous time BMAP, and Markov modulated Poisson process (MMPP) are used to model IP traffic were introduced recently, and independently in [7, 8].

source is kept in a buffer at the source until it is finally acknowledged by the receiver, or a timer expires, at which time it must be retransmitted. The receiver, upon receiving a data segment, whether it is the expected segment or not, always sends back an acknowledgment to the source indicating the data sequence number it is expecting. The TCP congestion control mechanism uses the sliding window mechanism, and is part preventive, and part reactive. In the preventive phase, there are two stages. In the first, called *slow start*, the source starts with a window size of one data segment, and for every acknowledgment it increases the window size by one data segment[3], until the window reaches a threshold level, which is initially set arbitrarily high [1]. At this point in time, the TCP protocol enters the *congestion avoidance stage* in which the window size is incremented by one maximum segment size every time a window full of segments is transmitted and acknowledged (linear window size increase). The reactive phase takes place when a segment[4] is lost inside the network and is not acknowledged. Segment losses can be detected through the timeout mechanism, in which case the source reduces its window size to one maximum segment size, and sets the window threshold to [9]

$$\max(\frac{\text{unacknowledged data}}{2}, 2 \times \text{max. segment size}) \tag{1}$$

It then enters the slow start stage again, until the window size reaches the threshold, at which time it switches to the congestion avoidance phase, and so on.

II.2 TCP Reno

The TCP Reno version implements two other mechanisms which detect, and react to congestion faster. The first mechanism, called *fast retransmit* detects the loss of a data segment when it receives a new acknowledgment followed by three duplicate acknowledgments of the same segment. The lost segment is then retransmitted. The second mechanism, *fast recovery*, then follows. The source then halves the window size, sets the threshold to that new window size, increments the window size by three, enters the congestion avoidance stage, and keeps on transmitting new segments. Duplicate acknowledgments increment the window size, while a new acknowledgment resets the window to the threshold value. The last event causes the start of the congestion avoidance phase. In case the timer expires at the source before three duplicate acknowledgments are received, the source reduces its window size to one maximum segment size, sets the slow start threshold to the value in equation (1), and enters the slow start phase.

[3]The window size is incremented by one for every acknowledgement, in addition to the recovery of the credit conveyed by the acknowledgement. This will eventually result in doubling the window size, which happens after acknowldgements for all segments in the window are received, hence increasing the window size exponentially.

[4]Since a TCP segment is transmitted by the IP layer as a packet, when an IP packet is lost, this means that the TCP segment is lost. Therefore, although the modeling of the window evolution process is at the TCP layer, it actually refers to IP layer packet transmission.

II.3 RED

RED was proposed in [4] as a means of signaling congestion to sources, so that they can reduce their input traffic. RED uses a first order filtering mechanism to calculate the average queue size seen by the nth packet arrival, a_n, in terms of a_{n-1}, and the actual queue size, q_n, using the following equation:

$$a_n = (1-g)a_{n-1} + gq_n. \tag{2}$$

The parameter g determines how fast the average queue size responds to changes in the actual queue size. In order to drop packets, RED uses the following procedure:

$$P(packet\ discarding) = \begin{cases} 0 & 0 \le a_n \le minTh \\ \frac{a_n - minTh}{maxTh - minTh} P_{max} & minTh \le a_n \le maxTh \\ 1 & maxTh \le a_n \end{cases}$$

where $minTh$ and $maxTh$ are minimum and maximum threshold values of the average queue size. The packet dropping probability is 0 if the average queue size upon packet arrival does not exceed $minTh$, while it is 1 if the average queue size exceeds $maxTh$. P_{max} is the maximum probability of packet discarding when $minTh \le a_n \le maxTh$.

II.4 The D-BMAP Process

The discrete batch Markov arrival process (D-BMAP) is the discrete time version of the BMAP process introduced in [10]. The BMAP is a versatile process, and is equivalent to Neuts' N-process [11], while being simpler to handle. It lends itself to modeling a large number of arrival processes, including correlated ones. The discrete time process is characterized by the presence of $\mathcal{F}$ phases, $\{0, 1, \ldots, \mathcal{F}-1\}$, where the process makes a transition from phase i to phase j at the end of the time slot with probability β_{ij}. $\mathbf{B}$ is the transition probability matrix for this process. Upon making a transition to phase j, the process generates k messages with probability α_{jk}. If matrix $\mathbf{A_k} = \mathbf{diag}\{\alpha_{\mathbf{0k}}, \alpha_{\mathbf{1k}}, \ldots, \alpha_{\mathcal{F}-\mathbf{1},\mathbf{k}}\}$, then the matrix $\mathbf{BA_k}$ is the matrix whose (i,j) element denotes the transition probability from phase i to phase j with k message arrivals.

The D-BMAP process can be used to model any general process using the appropriate number of phases, and the appropriate transition probabilities between phases. Therefore, it will be employed in this paper to represent the aggregate background traffic, after extending it to be dependent on the packet loss process.

II.5 Related work

Several models of the TCP congestion control mechanism have been introduced. Reference [12] introduced simple models for TCP connections, which did not take into account several of the TCP

operating features, but were the first analytical models to provide insight into the operation of the TCP flow control mechanism. Reference [13] considers a single TCP (Tahoe or Reno) connection in a network with a high bandwidth-delay product. Packet losses occur randomly. A simple analysis of multiple connections was carried out in order to illustrate the bias towards connections with short propagation delays. The work in [14] considers the modeling of different versions of TCP under lossy links. The work in [15, 16] used a discrete time approach to model a single TCP Reno connection, in which packet losses are independent and random, and [17] extended the analysis in [15] to model the slow start stage. Reference [18] further extended the analyses in [17] and [16] to evaluate the session latency under TCP Tahoe, TCP Reno, and TCP SACK. Reference [19] introduced a Markov chain model of TCP Reno over ATM that approximated the protocol operation. Reference [20] used a fluid flow approach to model multiple connections with different round trip times. The work in [21] introduced two separate models: one for the network, and one for an aggregation of multiple on-off sources served by multiple TCP connections. An iterative approach was used for model tuning. In [22] the model in [13] was extended to study the behavior of TCP connections with different types of lossy channels. Using a mixture of simulation and analysis, reference [23] showed the self-similar nature of TCP connections. Multiple TCP sources were modeled in [24] by approximating them by a Poisson process. A fluid flow model for TCP over ABR virtual connections in ATM networks was introduced in [25]. Another approximate model for TCP over ABR, but without fast retransmit or fast recovery, was introduced in [26]. The model assumed an ABR service rate which is controlled by a 2-state Markov chain. In [27], Altman introduced a model for TCP Reno when losses occur according to a bursty process, and was extended in [28] for the case of a general stationary loss process. Approximations, and bounds for the maximum window size case were introduced, and approximations for the detection of loss through timeouts were also presented. Reference [29] used max-plus algebra to model the TCP Tahoe and Reno versions while crossing an arbitrary number of routers, and with different service rates. The effect of cross traffic was modeled using stochastic service rates. The choice of this service rate was not discussed, and the effect of the buffer size was not taken into account. Moreover, the computational cost of formulae increases in a non-polynomial way.

The RED queue management protocol was modeled in very few, and very recent studies. In references [30, 31] approximate analyses were carried out in which average queue size was approximated by the instantaneous queue size. Reference [32] introduced a fluid-flow analysis of the transient performance of RED and TCP sources, while reference [33] used a feedback control model as an approximate model to find the steady state behavior of active queue management schemes, including RED. Reference [34] presented an approximate model for a TCP Reno flow passing through a RED-controlled router. The model has some similarity to the model of this paper. Interference traffic is present, and is represented by a Markov Modulated Poisson Process (MMPP), but its rate is not dependent on packet losses. Therefore, it was only used to model UDP sources. However,

the solution method in [34] is totally different from our solution method, and it employed several approximations.

Reference [35] is a good article that outlines the difficulties, and the sources of error in modeling TCP protocols. We point out that several of such sources of error are absent from the modeling approach introduced in this paper.

III The Model

The model of this paper is based on the following assumptions:

1. We consider a bottlenecked router, and assume that queuing delays, and transmission times at other routers are constant, and are included in a constant factor, τ, which also includes the round trip propagation delay. This is a standard modeling assumption, which has been widely used, e.g., [13, 14].

2. Packet lengths are assumed constant, and the packet transmission time is assumed to be the time unit. In addition, the system is assumed to be time synchronous, and the time slot is also equal to the packet transmission time. This is also a standard modeling assumption [16], since the target node is modeled under heavy traffic, and in this case the node transmits packets with maximum size[5].

3. We consider a *target session* which is modeled as follows:

   (a) The target source uses the *Reno* version of TCP. That is, when a packet is discarded, it is detected by the receipt of three duplicate acknowledgments, and triggers the fast retransmit and fast recovery phases. Or, if the timer expires before such three duplicate acknowledgments are received, the packet loss is also recovered from, and the slow start phase is entered.

   (b) It is assumed that no losses take place from the target source during the slow start phase. We believe that this assumption is very reasonable for two reasons. First, the target source transmits a limited number of packets during this phase, where the maximum window size is half the window size before the loss. Second, because of the exponential increase in the window size during the Slow Start phase, the duration of the slow start phase is very short, namely, $\lceil \log_2 i/2 \rceil$, where i is the window size when the packet loss took place.

   (c) It is assumed that the time-out period, χ, satisfies the following relation

[5] In reality, packets can be variable in size. However, the assumption of a constant packet size is made for the sake of tractability.

$$\chi > 2\tau + \frac{\text{B}}{\text{transmission rate at bottleneck router}}$$

where B is the buffer size at the bottleneck router. Although this assumption is made for mathematical tractability, it is in line with the actual operation of TCP, and is also satisfied through the use of timers with coarse granularities, e.g., 500 ms.

(d) The target source is assumed to be constantly backlogged, i.e., it always has packets to send.

4. Other sessions are modeled aggregately using a background process, which is modeled as a *modified* Discrete Batch Markovian Arrival Process (D-BMAP) [5]. The background process consists of a number of phases, $\mathcal{F}$, such that:

   (a) Arrivals from the background process are dependent on the current phase.

   (b) Transitions between phases are also governed by the current phase. Moreover, transitions are also dependent on the number of packets discarded from the process. This last assumption is an **extension** that we introduce to the D-BMAP process, and it enables one to introduce a correlation between packet losses, and the packet generation process from the background traffic source. This introduces a feedback loop which is useful in using the D-BMAP to model other TCP sources, which reduce their transmission rates because of packet losses.

Notice that in the above, no assumption was made about the packet discarding probability or strategy, from either of the target source or the background traffic. Therefore, packet dropping follows the actual mechanics of the network and the router operation, and is solely due to buffer overflow. This avoids the need to accurately specify the packet loss process, which is a major source of errors in TCP models [35].

The system is assumed to be time synchronous, where the time slot is the packet transmission time at the bottlenecked router, and all time intervals are normalized to the packet transmission time. All packet arrivals and departures are synchronized to the slot boundaries. The system is modeled at two levels, namely a macroscopic level, and a microscopic level:

- At the **macroscopic level**, or **window evolution level**, which is the main model, the system is observed at the instants when the target session updates its window size, given that there are no outstanding packet losses. The system is modeled as an embedded Markovian chain at such instants.

- The transitions between different states at the macroscopic level are dependent on all the activities between the embedding points, including window adjustments, packet losses and detection of such losses, round trip delays, including queue size effects, etc. It is not possible to account for all such effects at the macroscopic level only. Therefore, the transition probabilities

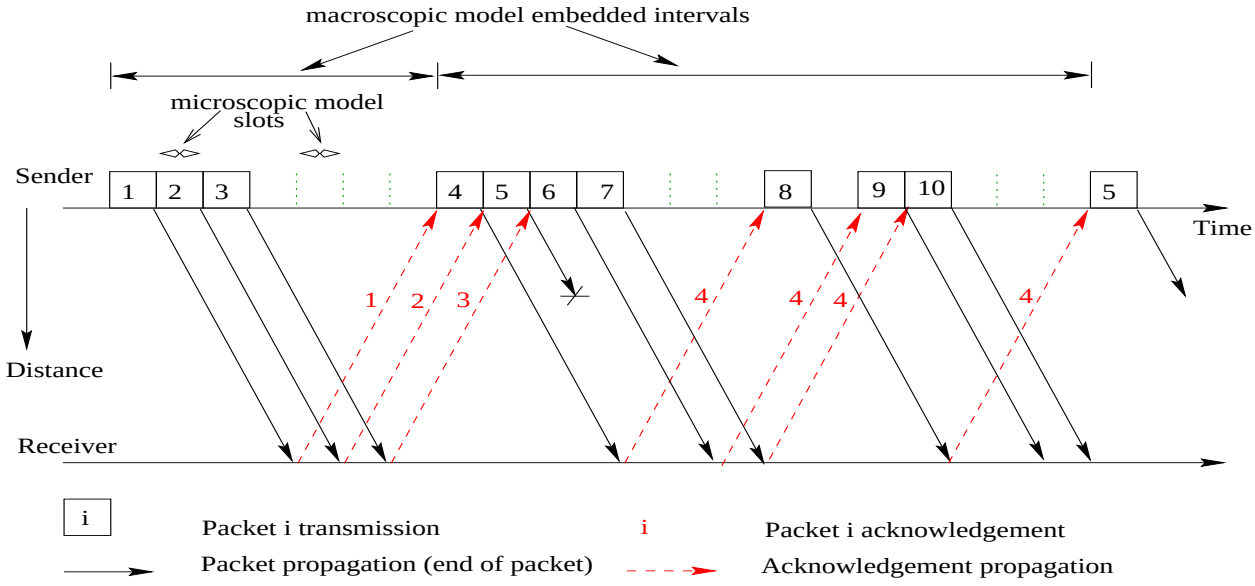

Figure 1: Typical scenario to illusrate the microscopic and macroscopic levels of analysis

between two successive embedding points are then computed using the **microscopic model**, or the **slot level model**, by considering the transitions between the system states at all the slot boundaries between these two embedding points.

The system state at the embedding points (**macroscopic model**) consists of the tuple $\Psi = (\Omega, \Theta, \Phi)$, where

Ω the window size of the target session, which can take values from 2 to W_{max}.

Θ the router's queue size, where the maximum buffer size is B

Φ the phase of the background process, i.e., the modified D-BAMP process.

The system state at slot boundaries within an embedded interval (microscopic model) is similarly given by the tuple $\psi = (\omega, \theta, \phi)$, where the window size, ω, is fixed over such an interval. Note that θ and ϕ are taken at the beginning of a slot, just after a packet has been removed from the queue. During the evolution of the phase state variable, ϕ, transitions between phases at the end of a slot, given the number of losses from this process are governed by

$\beta_{ff'}(l) = \Pr(\text{next phase is } f' | \text{ current phase is } f, \text{ and } l \text{ packets have been discarded})$.

Also, the probability that n packets arrive from the background process are phase dependent and given by

$\alpha_f(n) = \Pr(n \text{ packet arrivals} \mid \text{current phase is } f)$

such that $\sum_{n=0}^{n_{max}} \alpha_f(n) = 1$, where n_{max} is the maximum number of possible packet arrivals within a slot.

In Figure 1 we show an example that illustrates the two abstraction levels, namely, the microscopic and the macroscopic levels. In this example, two embedded intervals at the macroscopic level are shown. In the first, the window size is three packets, and all packets are transmitted

successfully. Within this interval, each slot corresponds to one packet transmission time. Notice also that the duration of this embedded period is equal to one packet transmission time, in addition to the delay encountered within the network in the forward and reverse paths.

In the second embedded interval, the window size has become four packets, and packet number 5 is lost. Three duplicate acknowledgments of packet number 4 will invoke the retransmission of the the lost packet. Notice that the embedded interval in this case ends when the lost packet is retransmitted.

IV Transition Probabilities

In this section we show how to evaluate the transition probabilities at the two levels.

IV.1 Microscopic Level Analysis

We first analyze the system at the microscopic level, which is the slot level. Note that all transition probability matrices between slot boundaries at this level will be denoted by $\mathbf{S}_x$, where the subscript x will be used to indicate a slot that contains success (s) or loss (l) from the target source, or a slot in which the target source is idle (i).

IV.1.i Microscopic level analysis: no loss from the target session

Consider the transition between two successive packet slot boundaries in which the target session does not encounter a loss. In this case, the target session is partly active for the duration of its window, and its packets are accepted, and partly inactive awaiting acknowledgments.

We use ρ to refer to the ordered pair (θ, ϕ), which is a subset of the state, ψ. Assume that ρ has the value (q, f). Notice that within an embedded *window evolution* interval, the window size, ω, does not change.

Define the following probability

$x(i, m|j, n, l, \rho) = \mathrm{P}$(within a slot, the buffer accepts i packets from the target session, given that j packets have already been accepted from the same session, and also accept m out of n packets from the background session, given l packets have been accepted, and the system state at the beginning of the slot is ρ)

The conditions $i, j \in \{0, 1\}$, $i + j = 1$ and $m \leq n$ must be satisfied.

We also define $P(accept|l, \rho)$ ($P(discard|l, \rho)$) as the probability of accepting (discarding) a packet given that l packets have been accepted, and the state is ρ. We have the following recursive relations:

$$\begin{aligned} x(1, m|0, n, l, \rho) &= P(accept|l, \rho) \times \left[\frac{1}{n+1} \times x(0, m|1, n, l, \rho) + \frac{n}{n+1} \times x(1, m-1|0, n-1, l+1, \rho)\right] \\ &\quad + P(discard|l, \rho) \times \left[\frac{n}{n+1} \times x(1, m|0, n-1, l, \rho)\right] \end{aligned} \tag{3}$$

$$x(0,m|1,n,l,\rho) = P(accept|l+1,\rho) \times x(0,m-1|1,n-1,l+1,\rho) + P(discard|l+1,\rho) \times x(0,m|1,n-1,l,\rho) \quad (4)$$

The two terms in each of the above equations take into account the cases in which the packet from the target source is accepted and rejected, respectively. In equation (3), the first term is further divided into two terms in which the accepted packet can be from either the target source or the background traffic, respectively, and then the queue subsequently accepts 0 or 1 more packets from the target session. In the second term of the equation, the discarded packet must be from the background process, and we must accept one packet from the target session. In these two equations we use

$$P(discard|l,\rho) = \begin{cases} 1 & l+q^+ = B \\ 0 & l+q^+ < B \end{cases} \quad (5)$$

where $q^+ = \max(q-1,0)$, and $P(accept|l,\rho) = 1 - P(discard|l,\rho)$.

The probabilities, $x()$, can be calculated recursively using equations (3) and (4), and the initial conditions:

$$x(0,0|1,0,l,\rho) = 1 \quad \text{for } l \geq 0 \text{ and } q^+ + l + 1 \leq B,$$
$$x(i,m|j,n,l,\rho) = 0 \quad \text{for } m > n, \text{ and}$$
$$x(i,m|j,n,l,\rho) = 0 \quad \text{for } i+j+m+l+q^+ > B$$

When the target session is inactive, i.e., awaiting acknowledgments, we define $y(0,m|0,n,l,\rho)$ similar to $x()$, except that no packets are generated or accepted from the target session. We therefore have the following relation, which is derived in a manner similar to equations (3) and (4):

$$y(0,m|0,n,l,\rho) = P(accept|l,\rho) \times y(0,m-1|0,n-1,l+1,\rho) + P(discard|l,\rho) \times y(0,m|0,n-1,l,\rho) \quad (6)$$

with the initial conditions:

$$y(0,0|0,0,l,\rho) = 1 \quad 0 \leq l+q^+ \leq B$$
$$y(0,m|0,n,l,\rho) = 0 \quad m > n \;\; or \;\; m+l+q^+ > B$$

Based on the above, we define the transition probability matrix between slot boundaries when a packet from the target session is accepted (i.e., success), $\mathbf{S}_s = [s_s(\rho'|\rho)]$, whose elements are the transition probabilities from state $\rho = (q,f)$ to state $\rho' = (q',f')$, and are given by

$$s_s(\rho'|\rho) = \sum_{n=q'-q^+-1}^{n_{max}} x(1,q'-q^+-1|0,n,0,\rho) \cdot \alpha_f(n) \cdot \beta_{ff'}(n-q'+q^++1)$$

We also define the transition probability matrix between slot boundaries when the target session is not active (i.e., idle), $\mathbf{S}_i = [s_i(\rho'|\rho)]$, where its elements are given by

$$s_i(\rho'|\rho) = \sum_{n=q'-q^+}^{n_{max}} y(0,q'-q^+|0,n,0,\rho) \cdot \alpha_f(n) \cdot \beta_{ff'}(n-q'+q^+)$$

IV.1.ii Microscopic level analysis: loss from the target session

Slightly different from the above, we define

$z(i, m|j, n, l, \rho)$ = P(within a slot, i packets are discarded from the target session, given j have already been discarded, and also m packets are accepted out of n packets from the background session, given l have been accepted, and given the system state at the beginning of the slot is ρ).

Similar to $x()$, i and j also take values of 0 and 1 and sum to 1.

Therefore, $z()$ can be evaluated for the cases of $m + l + i + j + q^+ < B$ using the following equations:

$$\begin{aligned} z(1, m|0, n, l, \rho) &= P(discard|l, \rho) \times \left[\frac{1}{n+1} \times z(0, m|1, n, l, \rho) + \frac{n}{n+1} \times z(1, m|0, n-1, l, \rho)\right] \\ &\quad + P(accept|l, \rho) \times \left[\frac{n}{n+1} \times z(1, m-1|0, n-1, l+1, \rho)\right] \qquad (7) \\ z(0, m|1, n, l, \rho) &= P(accept|l, \rho) \times z(0, m-1|1, n-1, l+1, \rho) \\ &\quad + P(discard|l, \rho) \times z(0, m|1, n-1, l, \rho) \qquad (8) \end{aligned}$$

and the initial conditions:

$$z(0, 0|1, 0, l, \rho) = 1 \quad l + q^+ <= B \qquad (9)$$

$$z(1, 0|0, 0, l, \rho) = 1 \quad l + q^+ = B \qquad (10)$$

$$z(i, m|j, n, l, \rho) = 0 \quad m > n \text{ or } m + l + q^+ > B \qquad (11)$$

With the above definition of $z()$, we can define the transition probability matrix between slot boundaries in which the target session encounters a loss as $\mathbf{S}_l = [s_l(\rho'|\rho)]$, with

$$s_l(\rho'|\rho) = \sum_{n=q'-q^+}^{n_{max}} z(1, q' - q^+|0, n, 0, \rho) \cdot \alpha_f(n)\beta_{ff'}(n - q' + q^+)$$

The transition probability matrix between slot boundaries is therefore

$$\mathbf{S} = \mathbf{S}_s + \mathbf{S}_l + \mathbf{S}_i$$

IV.2 Macroscopic Level Analysis

Next, we analyze the system at the macroscopic level, which is the window evolution level. The system is modeled as an embedded Markov chain at the instants where the model starts a new window which did not encounter any loss, as has been described earlier. The window will therefore increase, in the case of no loss, or decrease in the case of a loss. The amount of decrease also depends on the method of loss detection. In the case of loss detection through three duplicate acknowledgments, the window is halved, while if the loss is detected through a timer expiry, the window will be reset to one maximum segment size, and will have to increase exponentially through the slow start phase. Transition probability matrices at this level will be referred to by $\mathbf{W}$, with different subscripts as will be explained in the text.

IV.2.i Macroscopic level analysis: an interval without loss

Now we consider an embedded interval that starts in state $(\Omega, \Theta, \Phi) = (w, q, f)$, and does not encounter any losses from the target TCP session. The transition probability matrix to state (w', q', f'), where $w' = \min(w+1, W_{max})$ will be denoted by $\mathbf{W}_{i+}$.

Let the block matrix row of the matrix $\mathbf{S}_s$ corresponding to an initial queue size of q be denoted by $S_s(q)$. Therefore, the block matrix row of the matrix $\mathbf{W}_{i+}$ corresponding to an initial queue size of q is also denoted by $W_{i+}(q)$, and is given by

$$W_{i+}(q) = S_s(q)\mathbf{S}_s{}^{i-1}\mathbf{S}_i{}^{q^+ + \tau + 2 - i} \tag{12}$$

In the above, it has been assumed that all packets in the window will encounter the same queuing delay at the router, as that seen by the first packet, which is equal to q^+. Equation (12) can be calculated efficiently in a recursive manner.

IV.2.ii Macroscopic level analysis: an interval with loss

Unlike section IV.2.i, this interval does not end with the arrival of the following window, since at the beginning of the following window the packet loss would not have been recovered from yet (see Figure 1). As such, the following window is an extension of the current window, and is used to recover from lost packets. In fact, this interval ends with the arrival of a new window which starts with the retransmission of the discarded packet. Denote by $\mathbf{W}_{i-}$ the transition probability matrix to state (w', q', f'), where $w' = \max(2, \lceil \frac{w}{2} \rceil)$, and $w = i$ is the initial window size. This matrix is used when a packet loss takes place and is detected.

It should be noted that for the case of an initial window size, i, with $i \geq 4$, $\mathbf{W}_{i-}$ consists of several components, depending on whether the loss is detected through duplicate acknowledgments, or through time-out. And, in the former case, there are also two components depending on whether the three packets resulting in the duplicate acknowledgments are transmitted within the same window, or during the next window. These components will be discussed separately below.

Detection of loss through duplicate acknowledgments:

Observe that a packet loss cannot be detected through duplicate acknowledgments if the window size is less than four[6].

We define

$\mathbf{U}^{(k)}(j)$ = Transition probability matrix with exactly j packets being accepted from the target session in the router's queue within k slots.

[6]In the second embedded macroscopic interval in Figure 1, since the window size is four, loss detection through duplicate acknowledgments is possible. Had the window size been three, or had multiple packets been discarded, then not enough duplicate acknowledgments will be received, and loss detection through this mechanism is infeasible.

We can therefore use the following recursive relations to compute $\mathbf{U}()$

$$\mathbf{U}^{(k)}(2) = \mathbf{S}_l\mathbf{U}^{(k-1)}(2) + \mathbf{S}_s\mathbf{U}^{(k-1)}(1) \quad (13)$$

$$\mathbf{U}^{(k)}(1) = \mathbf{S}_l\mathbf{U}^{(k-1)}(1) + \mathbf{S}_s\mathbf{U}^{(k-1)}(0) \quad (14)$$

$$\mathbf{U}^{(k)}(0) = \mathbf{S}_l\mathbf{U}^{(k-1)}(0) \quad (15)$$

starting from $\mathbf{U}^{(0)}(0) = \mathbf{I}$, $\mathbf{U}^{(0)}(1) = \mathbf{0}$, and $\mathbf{U}^{(0)}(2) = \mathbf{0}$, where $\mathbf{I}$ is the identity matrix, and $\mathbf{0}$ is the zero matrix.

There are two cases to consider when the loss is detected through duplicate acknowledgments:

1. The case in which the loss is detected within the same window. This requires that after the discarded packet there be at least three more packets in the same window. Therefore, for the case of $i \geq 4$, the component of $W_{i-}(q)$ is denoted by W_{i-}^{DS} (for *D*etection within the *S*ame window), and is given by

$$W_{i-}^{DS}(q) = \sum_{n=0}^{i-5} S_s(q)\mathbf{S}_s^n\mathbf{S}_l \sum_{k=2}^{i-n-3} \mathbf{U}^{(k)}(2)\mathbf{S}_s[\mathbf{S}_s + \mathbf{S}_l]^{i-k-n-3}\mathbf{S}_i^{q^+ + \tau + 2 - i}[\mathbf{S}_s + \mathbf{S}_l]^{n+1}\mathbf{S}_i^{k+1}$$
$$+ S_l(q) \sum_{k=2}^{i-2} \mathbf{U}^{(k)}(2)\mathbf{S}_s[\mathbf{S}_s + \mathbf{S}_l]^{i-k-2}\mathbf{S}_i^{q^+ + \tau + k + 3 - i} \quad (16)$$

   The first term in the above equation corresponds to the first lost packet being packet $n + 2$, for $n \geq 0$, while the second term corresponds to the first packet in the window being lost.

2. The other case is the one in which the loss is detected within the following window. This requires that, if the first packet discarded is packet k in the first window, where $k > 1$, then the third acknowledgment must be in response to packet number l in the following window, where $l < k$. Therefore, also for $i \geq 4$, the component of $W_{i-}(q)$ for this case is denoted by W_{i-}^{DN} (for *D*etection within the *N*ext window), and can be expressed as

$$W_{i-}^{DN}(q) = \sum_{j=0}^{2} \sum_{n=2-j}^{i-2-j} S_s(q)\mathbf{S}_s^n\mathbf{S}_l\mathbf{U}^{(i-n-2)}(j)\mathbf{S}_i^{q^+ + \tau + 2 - i}$$
$$\cdot \sum_{k=2-j}^{n} \mathbf{U}^{(k)}(2-j)\mathbf{S}_s[\mathbf{S}_s + \mathbf{S}_l]^{n-k}\mathbf{S}_i^{k+\tau+q^+-n+1} \quad (17)$$

In this equation, a loss must take place in packet number $n + 2$, for $n \geq 0$, and fewer than three packets are not discarded from the same window, and after the occurrence of the first loss.

Notice that in equations (16) and (17), we consider transitions between slots whose states correspond to the no loss, loss, or idle from the target session point of view.

Detection of loss through time-out:

Detection of loss through time-out, where the time-out interval is assumed fixed and equal to χ slots, measured from the end of the packet transmission, will take place in one of two cases:

1. The window size is less than four, in which case W_{i-}^T for $i < 4$ is given by

$$W_{1-}(q) = S_l(q)\mathbf{S}_i^{\chi} \tag{18}$$

$$W_{2-}^T(q) = S_l(q)[\mathbf{S}_s + \mathbf{S}_l]\mathbf{S}_i^{\chi-1} + S_s(q)\mathbf{S}_l\mathbf{S}_i^{q^+ +\tau}[\mathbf{S}_s + \mathbf{S}_l]\mathbf{S}_i^{\chi-q^+-\tau-2} \tag{19}$$

$$W_{3-}^T(q) = S_l(q)[\mathbf{S}_s + \mathbf{S}_l]^2\mathbf{S}_i^{\chi-2} + S_s(q)\mathbf{S}_l[\mathbf{S}_s + \mathbf{S}_l]\mathbf{S}_i^{q^+ +\tau-1}[\mathbf{S}_s + \mathbf{S}_l]\mathbf{S}_i^{\chi-q^+-\tau-2} + S_s(q)\mathbf{S}_s\mathbf{S}_l\mathbf{S}_i^{\tau+q^+-1}[\mathbf{S}_s + \mathbf{S}_l]^2\mathbf{S}_i^{\chi-q^+-\tau-3} \tag{20}$$

Equation(18) is straightforward, and corresponds to the case in which the only packet in the window is lost. In equation (19), the two terms correspond to the first loss occurring in the first, and second packets, respectively. Equation (20), three terms correspond to the first loss occurring in the first, second, and third terms, respectively.

2. The window size is greater than or equal to four, but not enough packets are accepted from the target session (at least three) in order for the three duplicate acknowledgments to be sent back to the source. Therefore, for $i \geq 4$, the corresponding component of $W_{i-}(q)$ is denoted by $W_{i-}^T(q)$ is given by

$$W_{i-}^T(q) = \sum_{k=0}^{2}\sum_{j=0}^{2-k} S_s(q)\sum_{n=0}^{i-2}\mathbf{S}_s^n\mathbf{S}_l\mathbf{U}^{(i-n-2)}(j)\mathbf{S}_i^{q^+ +\tau+2-i}\mathbf{U}^{(n+1)}(k)\mathbf{S}_i^{\chi-q^+-\tau-1} + \sum_{j=0}^{2} S_l(q)\mathbf{U}^{(i-1)}(j)\mathbf{S}_i^{\chi-i+1} \tag{21}$$

Based on equations (16) – (21), the matrix $\mathbf{W}_{i-}$ for $i \geq 4$ can be obtained from

$$\mathbf{W}_{i-} = \mathbf{W}_{i-}^{DS} + \mathbf{W}_{i-}^{DN} + \mathbf{W}_{i-}^{T} \quad .$$

The transition probability matrix between *window evolution* embedding points (macroscopic level), given that the window size at the initial embedding point was i, is given by

$$\mathbf{W}_i = \mathbf{W}_{i+} + \mathbf{W}_{i-}$$

We also define the transition probability matrix, $\mathbf{W}_{i-}^D$, which takes into account the reduction in window size when a loss is detected through duplicate acknowledgment only,

$$\mathbf{W}_{i-}^D = \mathbf{W}_{i-}^{DS} + \mathbf{W}_{i-}^{DN} \quad ,$$

while the transition probability matrix for the window evolution when there is a loss that is detected through a time-out is given by $\mathbf{W}_{i-}^T$.

IV.3 Transition Probability Matrix

We now define the transition probablity matrix, $\mathbf{P}$, at the macroscopic level embedding points. Notice that these embedding points correspond to the start of the transmission of a new window from the target source in which there are no outstanding losses, and the target source is in the congestion avoidance phase. Here, we invoke assumption 3(b) in Section III, namely, that no packets are lost from the target source while in the Slow Start phase. When a loss is detected through time-out, and the target source enters the Slow Start phase, the embedding interval under the macroscopic model is defined to end in this case when the slow start phase ends, and the window size reaches half the window size before the loss[7]. We define the matrix $\mathcal{W}_{k+}$ as the transition probability matrix between two successive window evolution embedding points, started with a window size of k, where the target source is in the slow start phase. Because of the assumption of no loss from the target source during the slow start phase, we have

$$\mathcal{W}_{k+}(q) = (S_s(q) + S_l(q))(\mathbf{S}_s + \mathbf{S}_l)^{k-1}\mathbf{S}_i^{q^+ + \tau + 2 - k}$$

Notice that since this matrix will be used in the Slow Start phase, where the window size grows exponentially, k in the above equation will always assume a value which is a power of 2. Also, note that all window evolutions within the slow start phase are considered to form an embedded interval, and the transition probability matrix from such an interval may involve the product of several of the $\mathcal{W}_{k+}$ matrices, as will be shown below.

The elements of the transition probability matrix between the *Slow Start* embedding points, $\mathbf{P}$ are therefore given by

$$P_{i,j} = \begin{cases} W_{i+} & j = \min(i+1, W_{max}) & \text{(term 1)} \\ W_{i-}^D & i \geq 4, j = \lceil \frac{i}{2} \rceil, \text{ and } j \neq \text{power of 2} & \text{(term 2)} \\ W_{1-}^T & i = 1, j = 1 & \text{(term 3)} \\ W_{2-}^T & i = 2, j = 1 & \text{(term 4)} \\ W_{i-}^T \left(\prod_{l=0}^{\lceil \log_2 i \rceil - 2} \mathcal{W}_{2^l +} \right) & i > 2,\ j = 2^{\lceil \log_2 i \rceil - 1}, j \neq \lceil \frac{i}{2} \rceil & \text{(term 5)} \\ W_{i-}^T \left(\prod_{l=0}^{\lceil \log_2 i \rceil - 2} \mathcal{W}_{2^l +} \right) + W_{i-}^D & i > 2,\ j = 2^{\lceil \log_2 i \rceil - 1}, j = \lceil \frac{i}{2} \rceil & \text{(term 6)} \\ 0 & \text{otherwise} & \text{(term 7)} \end{cases} \quad (22)$$

The different terms in the transition probability matrix are derived as follows:

term 1: the increase in the window size by 1 due to a successful transmission by the target source, up to a maximum of W_{max}.

term 2: the halving of the window size because of a loss that was detected by three duplicate acknowledgments. The case of j being a power of two is included in **term 6**.

[7]Since the window size is in terms of units of the maximum segment size, then the threshold is set to the smallest integer, greater than or equal to half the window size when loss took place.

term 3: the resetting of the window size to 1 because of a loss that is detected through timeout. The slow start threshold is equal to 1 in this case.

term 4: similar to **term 3**, except that the initial window size is 2.

term 5: the resetting of the window size to 1 because of a loss that is detected through timeout, and the successive exponential increase in the window size up to $\lceil \frac{i}{2} \rceil$, where i is the window size before the loss. The case of $j = \lceil \frac{i}{2} \rceil$ is treated in **term 6**.

term 6: this term includes a term similar to **term 5**, in addition to a term similar to **term 2** since j is both a power of 2, and is equal to $\lceil \frac{i}{2} \rceil$. Therefore, a transition from i to this value of j is possible due to loss detection through time-out followed by a slow start phase, and also due to loss detection by three duplicate acknowledgments

term 7: these are invalid transitions.

From the above, it can be easily seen that the matrix **P** has either two or three block matrices in any row, i, where i is a window size. Therefore, the steady state probability matrix can be solved using the efficient method introduced in [36].

V Model Extension: RED-Based Routers

In this section we show how the model can be extended to the case of routers implementing the RED strategy.

The system state must be expanded to include a characterization of the average queue size as measured by RED. Therefore, it is represented by the quadruple $\Psi = (\Omega, \Theta, \Delta, \Phi)$, where Δ is a function of the average queue size and the actual queue size, and is used in calculating the average queue size as follows. According to the RED strategy, the average queue size upon the nth packet arrival is computed using equation (2). In reality, a_n is a discrete, non-integer, variable, but the number of discrete levels may be on the order of B^B. It is therefore impossible to provide an exact, and tractable model of RED, even with a small buffer size. Instead, we employ the state variable, Δ, which, when measured after accepting the nth packet, corresponds to $\eta(q_n - a_{n-1})$, where η is a scaling factor[8] in the range of 1 to $\frac{1}{g}$.

The above representation is based on the observation that the average queue value, a, follows the instantaneous queue queue value, q, although at a much slower rate. That is, a moves in the same direction as q. As such, we assume that the difference between a and q will be limited.

According to equation (2), we have

$$\eta(q_n - a_{n-1}) = \eta(\frac{a_n - a_{n-1}}{g})$$

[8]The η factor is chosen to be inversely proportional to the load.

As such, a_n can be computed as $q-(1-g)d/\eta$, where d is the value of Δ. Since this value is not necessarily an integer, we use the following approximation for Δ.

$$\Delta = \eta\lfloor q - a + 0.5\rfloor.$$

During the calculation of the transition probabilities at the microscopic level, the tuple $\rho = (\theta, \phi)$ is also expanded to capture changes in Δ. Therefore, ρ is replaced by (θ, δ, ϕ), and δ takes the value d at the beginning of slot, and d' at the end of the slot.

The discarding probability in equation (5) must also be revised in order to reflect the probability of packets being discarded according to the RED strategy. Accordingly, the following probability should be used instead:

$$P(discard|l,\rho) = \begin{cases} 1 & l + q^+ = B \text{ or } a(l,\rho) > maxTh \\ \frac{a(l,\rho)-minTh}{maxTh-minTh} \times P_{max} & l + q^+ < B \text{ and } minTh < a(l,\rho) \leq maxTh \\ 0 & a(l,\rho) \leq minTh \end{cases} \quad (23)$$

where $a(l,\rho)$ is the average queue size after l packets have been accepted in the queue, given the state of the system at the beginning of the slot is ρ. This is computed as

$$a(l,\rho) = (1-g)^l a(0,\rho) + \sum_{i=1}^{l} g(1-g)^{l-i}(q^+ + i) \quad (24)$$

$a(0,\rho)$ can be obtained from ρ using the relation $a = q - (1-g)d/\eta$.

The calculation of d' given d in the $\mathbf{S}_s$, $\mathbf{S}_i$ and $\mathbf{S}_l$ matrices, follows the determination of q'. The value of d' can then assume one value only which is determined using equation (24) by setting $l = q' - q^+$. One exception to this is the case in which both q^+ and l are equal to zero. In this case, the average queue size will be updated by multiplying it by $1-g$, and $d' = -\eta(1-g)a(0,\rho)$.

The accuracy of the above approach increases with δ. However, this also increases the computational requirements. In order to reduce teh computational requirements, an approximate approach is used in which the system dependence on δ is relaxed. In other words, δ is estimated given the queue size, θ, and the arrival process. then, the queue size is determined by this estimate of δ, and the system dynamics.

The approximate approach uses two Markov chains:

- The original Markov chain of the model, but after removing the state variables Δ and δ from the tuples Ψ and ψ, respectively, and
- A second Markov chain in which the state consists of the pair (δ, θ).

A reasonable assumption for the value of the conditional probability $P(\delta|\theta)$ is made, and the first Markov chain is solved to obtain the distribution of the instantaneous queue size, θ. The second Markov chain then uses this distribution of θ, together with the arrival process and solves for the

joint distribution of δ and θ. From this joint distribution, $P(\delta|\theta)$ is computed, and is used in the first Markov chain. The process is repeated until convergence. It was found that two to three iterations are usually required for convergence, and that the throughput results obtained from the original model, and the approximate model differ by less than 1%, but with a significant saving in computation.

VI Performance Measures

Denote the steady state probability vector by

$$\mathbf{\Pi} = \{\vec{\pi}_1, \vec{\pi}_2, \ldots, \vec{\pi}_{W_{max}}\}$$

where $\vec{\pi}_i$ is the steady state probability vector of the target session having a window size of i. This vector in turn has several component vectors in terms of the state ρ,

$$\vec{\pi}_i = \{\vec{\pi}_{i_0}, \vec{\pi}_{i_1}, \ldots, \vec{\pi}_{i_{\rho max}}\}$$

Given the transition probability matrix in equation (22) with its regular structure[9], we applied the efficient solution method in [36] to solve for the vector Π. Several performance measures can be directly computed from Π, including the distributions of queue size and the window size, as well as their moments. In addition, we can calculate the distribution of losses within windows of different sizes.

The following performance measures can be computed:

1. Mean queue size $= \sum_{i=1}^{W_{max}} \sum_{\rho,\Theta=q\in\rho} q\vec{\pi}_{i_\rho}\vec{1}$
   where $\vec{1}$ is an appropriately dimensioned column vector of 1's.

2. Also, some probability distributions can be computed including:

   (a) $\mathrm{P(window = i)} = \sum_\rho \vec{\pi}_{i_\rho}\vec{1}$

   (b) The probability of loss given window size $= i$ can be expressed as:

   $$\mathrm{P(loss|window = i)} = \frac{\mathrm{P(loss\ and\ window = i)}}{\mathrm{P(window = i)}} = \frac{\vec{\pi}_i \mathbf{W}_{i-}\vec{1}}{\sum_\rho \vec{\pi}_{i_\rho}\vec{1}}$$

   and similarly,

   $$\mathrm{P(window = i|loss)} = \frac{\mathrm{P(loss\ and\ window = i)}}{\mathrm{P(loss)}} = \frac{\vec{\pi}_i \mathbf{W}_{i-}\vec{1}}{\sum_{i=2}^{W_{max}} \mathrm{P(loss\ and\ window = i)}}$$

   (c) $\mathrm{P(loss\ detected\ by\ duplicate\ ACK|loss\ when\ window = i)} = \frac{\vec{\pi}_i(\mathbf{W}_{i-}^{DS}+\mathbf{W}_{i-}^{DN})\vec{1}}{\mathrm{P(loss\ and\ window=i)}}$

   (d) $\mathrm{P(loss\ detected\ by\ timeout|loss\ at\ window = i)} = \frac{\vec{\pi}_i \mathbf{W}_{i-}^{T}\vec{1}}{\mathrm{P(loss\ and\ window=i)}}$

[9]Since the Markovian chain is finite, irreducible and aperiodic, the steady state probability vector, Π, exists [37].

(e) P(first loss is at n$^{\text{th}}$ position|window = i, loss in window) $= \frac{\vec{\pi}_i \mathbf{S}_s^{n-1} \mathbf{S}_l \vec{1}}{\text{P(loss and window=i)}}$

3. **Throughput:** The system throughput is more involved to compute and can be obtained using the definition:

$$\text{Throughput} = \frac{\text{E(number of successfully received packets in a cycle)}}{\text{E(cycle length)}}$$

where a cycle is the duration between two successive embedding points under the macroscopic analysis. There are two cases to consider in computing the numerator and the denominator of the above expression:

(a) **A cycle without loss.**

In this case

$$\text{E(number of packets in a cycle without loss}|\Omega = \text{i}, \Theta = \text{q}) = i\vec{F}(i,q)W_{i+}(q)\vec{1} \quad (25)$$

and

$$\text{E(length of a cycle without loss}|\Omega = \text{i}, \Theta = \text{q}) = (q^+ + 1 + \tau)\vec{F}(i,q)W_{i+}(q)\vec{1} \quad (26)$$

The vector $\vec{F}(i,q)$ in the above equations is the steady state probability vector of Φ given that $\Omega = i$ and $\Theta = q$. This vector can be calculated easily from $\vec{\pi}_i$ and the joint steady state probability of $\Omega = i$ and $\Theta = q$, which is obtained from

$$\sum_{\rho, \Theta = q \in \rho} \vec{\pi}_{i_\rho} \vec{1}$$

(b) **A cycle with loss.**

To compute the cycle length and the number of packets transmitted during this cycle in an exact manner can be computationally expensive. This is especially true since we have to distinguish between a cycle containing a loss which is detected by the expiry of the timer, and a cycle, also containing a loss, but is detected by the receipt of three duplicate acknowledgments. Instead, we take an approximate, but accurate, approach in handling cycles with losses in which we keep track of the first loss in a cycle exactly. However, further losses are considered to depend on the first loss, but to be independent among themselves.

To compute the cycle length and the number of packets transmitted during this cycle we require several auxiliary probabilities:

- The probability of the first loss at position m, when the window and queue sizes are i and q, respectively, is given by $L_{i,q}(m)$, for $1 \leq m \leq i$. This can be expressed as:

$$L_{i,q}(1) = \vec{F}(i,q)S_l(q)\vec{1} \quad (27)$$

$$L_{i,q}(m) = \vec{F}(i,q)S_s(q)\mathbf{S}_s^{m-2}\mathbf{S}_l\vec{1} \text{ for } m \geq 2, \quad (28)$$

and

- The probability of loss at position n, given that the first loss was at position m, and that the window and queue sizes are i and q respectively, is denoted by $l_{i,q,m}(n)$, for $1 \leq m < n < i + m$. Notice that n can be greater than i, which means that it is in the following sub-window of packet transmissions corresponding to the first $m-1$ acknowledgments. As such,

$$l_{i,q,m}(n) = \frac{\mathrm{P(loss\ at\ n,\ first\ loss\ at\ m}|\Omega = \mathrm{i}, \Theta = \mathrm{q})}{\mathrm{P(first\ loss\ at\ m}|\Omega = \mathrm{i}, \Theta = \mathrm{q})} \tag{29}$$

where

$$\mathrm{P(loss\ at\ n,\ first\ loss\ at\ m}|\Omega = \mathrm{i}, \Theta = \mathrm{q}) =$$

$$\begin{cases} \vec{F}(i,q)S_l(q)(\mathbf{S}_s + \mathbf{S}_l)^{n-2}\mathbf{S}_l\vec{1} & \text{for } m = 1,\ 2 \leq n \leq i \\ \vec{F}(i,q)S_s(q)\mathbf{S}_s^{m-2}\mathbf{S}_l(\mathbf{S}_s + \mathbf{S}_l)^{n-1-m}\mathbf{S}_l\vec{1} & \text{for } 1 < m < n \leq i \\ \vec{F}(i,q)S_s(q)\mathbf{S}_s^{m-2}\mathbf{S}_l(\mathbf{S}_s + \mathbf{S}_l)^{i-m} & \\ \quad \cdot \mathbf{S}_i^{(q^+ + 1 + \tau - i)}(\mathbf{S}_s + \mathbf{S}_l)^{n-i-1}\mathbf{S}_l\vec{1} & \text{for } 1 < m \leq i < n < i + m \end{cases} \tag{30}$$

Now we can express the contribution of cycles with losses to the mean cycle time and the mean number of packets transmitted within those cycles according to the window size:

i. Ω =1, 2 or 3, in which the loss is always detected by the time-out mechanism.

$$\begin{aligned} &\mathrm{E(number\ of\ successful\ packets\ in\ a\ cycle\ containing\ a\ loss}| \\ &\quad \Omega = i, 1 \leq i \leq 3, \Theta = q) \\ &= \sum_{j=1}^{i} L_{i,q}(j)[j - 1 + \sum_{k=1}^{i-1}(1 - l_{i,q,j}(j+k)) + \sum_{l=0}^{R(i)} 2^l] \end{aligned} \tag{31}$$

$$\begin{aligned} &\mathrm{E(length\ of\ a\ cycle\ that\ contains\ a\ loss}|\Omega = \mathrm{i}, 2 \leq \mathrm{i} \leq 3, \Theta =, \mathrm{q}) \\ &= \sum_{j=1}^{i}(\chi + j - 1 + R(i) \cdot (q^+ + 1 + \tau))L_{i,q}(j) \end{aligned} \tag{32}$$

The term $R(i)$ above is the number of rounds in the slow start phase, and is used in the above two equations in order to calculate the number of successful packets in that phase, and the actual length of the phase, respectively. It is given by

$$R(i) = \begin{cases} 0 & i \leq 2 \\ \lceil \log_2 i \rceil - 1 & i > 2 \end{cases}$$

ii. $\Omega \geq 4$, in which case the loss can be detected either by the time-out mechanism, or the triple duplicate acknowledgment mechanism. The cycle length and the number of packets transmitted in a cycle in such cases are calculated similar to equations (31) and (32), except that we have to take into account all combinations of losses in the cycle:

$$\begin{aligned} &\mathrm{E(number\ of\ successful\ packets\ in\ a\ cycle\ with\ loss}|\Omega = \mathrm{i}, \mathrm{i} \geq 4, \Theta = \mathrm{q}) \\ &= \sum_{j=1}^{i} L_{i,q}(j) \left[\sum_{k=0}^{2}[(j - 1 + k) + \sum_{l=0}^{R(i)} 2^l] + \sum_{k=3}^{i-1}(j - 1 + k) \right] \end{aligned}$$

$$\cdot \text{P(k successful packet transmissions)} \tag{33}$$

$$\begin{aligned}
&\text{E(length of a cycle with loss}|\Omega = \text{i}, 4 \le \text{i}, \Theta =, \text{q)} \\
&= \sum_{j=1}^{i} L_{i,q}(j)(\chi + j - 1 + R(i) \cdot (q^{+} + 1 + \tau)) \\
&\qquad \cdot \text{P(less than 3 successful transmissions after first loss)} \\
&+ \sum_{j=1}^{i} L_{i,q}(j) \sum_{k=j+3}^{i} (k + q^{+} + 1 + \tau) \\
&\qquad \cdot \text{P(3rd successful transmission after first loss is at k)} \\
&+ \sum_{j=1}^{i} L_{i,q}(j) \sum_{k=i+1}^{j+j-1} [2(q^{+} + 1 + \tau) + k - i] \\
&\qquad \cdot \text{P(3rd successful transmission after first loss is at k)}
\end{aligned} \tag{34}$$

Although the number of combinations in the above equations can be large, but the calculation of their respective probabilities is straightforward and fast since they only involve the scalar probabilities $L_{i,q}$ and $l_{i,q,m}(n)$.

Now, based on equations (25), (31) and (33),

$$\begin{aligned}
&\text{E(number of successful packets in a cycle)} \\
&= \sum_{i} \sum_{q} [\text{E(number of packets in a cycle without loss}|\Omega = \text{i}, \Theta = \text{q)} \\
&+ \text{E(number of packets in a cycle with loss}|\Omega = \text{i}, \Theta = \text{q)}] \Pr(\Omega = \text{i}, \Theta = \text{q})
\end{aligned} \tag{35}$$

and from equations (26), (32) and (34)

$$\begin{aligned}
&\text{E(cycle length)} \\
&= \sum_{i} \sum_{q} [\text{E(length of a cycle without loss}|\Omega = \text{i}, \Theta = \text{q)} \\
&+ \text{E(length of a cycle with loss}|\Omega = \text{i}, \Theta = \text{q)}] \Pr(\Omega = \text{i}, \Theta = \text{q})
\end{aligned} \tag{36}$$

VII Numerical Examples

In this section we present several numerical examples based on the above model. This section consists of a number of subsections, each addressing a certain aspect of the model.

VII.1 Model Verification

We first verify the accuracy of the model via simulation. We used the network shown in Figure 2, which consists of two sources connected to a router with a buffer size of 20 packets. Both sources send to the same destination node, which is also connected to the same router. The first source has

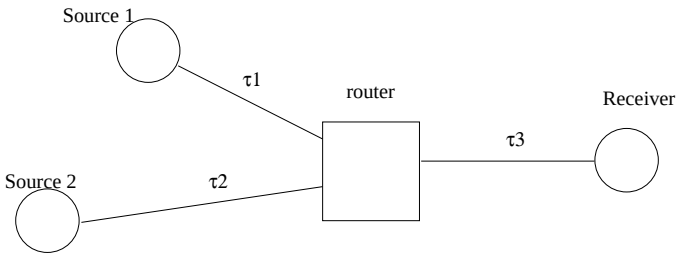

Figure 2: Sample network

Table 1: Throughput values for the sources in Figure 2 using the model, and the *ns-2* simulator

	Source 1	Source 2
Scenario 1 ($\tau_1 = 4$)	0.684002 (0.659 $\pm$ 0.0153)	0.339428 (0.341 $\pm$ 0.0153)
Scenario 2 ($\tau_1 = 8$)	0.572 (0.592 $\pm$ 0.0124)	0.422 (0.407 $\pm$ 0.0121)

a maximum window size of 16 packets, while the second source has a maximum window size of 8 packets. The propagation delays, $\tau 2 = 4$, and $\tau 3 = 4$, both in terms of packet transmission times. The propagation delay between source 1 and the router, $\tau 1$, assumes two values, namely 4 and 8 packet transmission times [10]. For this network, two models were constructed, one for each of the two sources, while the other source was modeled approximately using a Markov modulated Poisson process (MMPP). For example, for the model in which source 1 is the target source, source 2 was modeled as an MMPP generating background traffic. The MMPP used here is a special case of the modified D-BMAP used in this model, in which the phase changes result in reducing or increasing the transmission rate. The probability of phase changes themselves are functions of whether a loss is encountered from the process or not. The parameters of the MMPP are based on first order statistics which are obtained from the other model, in which source 2 is the target source. The two models were run in an alternating manner, and iteratively, until convergence was obtained. Although convergence was based on the mean buffer size of the router, other criteria could have been used. The throughput values obtained from the model are shown in Table 1. We also show the results obtained from the *ns-2* simulator [38], together with 95% confidence intervals, in the same table, and in parentheses. As shown in the table, the results from the model are very close to those from simulation, and the error does not exceed 4%.

We also modeled a network consisting of two sources connected to a RED enabled router with a buffer size of 10 packets, and maximum and minimum thresholds of 10 and 5 packets, respectively. Both sources send to the same destination node, which is also connected to the same router. Both sources have a maximum window size of 8. τ for the first source is always 8, while for the second source it assumes the values 8 and 16, respectively. Two models were used, one for each target

[10] This network corresponds to a network with links operating at a DS-3 rate of 44.736 Mb/s, packet lengths of 1400 bytes, and propagation delays $\tau 1 = 1$ and 2 ms, $\tau 2 = 1$ ms, and $\tau 3 = 1$ ms.

Table 2: Throughput of two sources when the router is RED enabled, and not RED enabled; $\tau_1 = 8$

	Without RED		With RED	
	Source 1	Source 2	Source 1	Source 2
$\tau_2 = 8$	0.507 (0.503±0.0146)	0.507 (0.497±0.0126)	0.508 (0.497±0.0031)	0.508 (0.503±0.0050)
$\tau_2 = 16$	0.644 (0.649±0.0110)	0.347 (0.350±0.0112)	0.633 (0.642±0.0056)	0.362 (0.357±0.0118)

Table 3: Throughput of the target TCP source in the presence of 5 UDP background sources

Background load	Analytical Results	Simulation Results
0.3	0.4851	0.4974 ±0.0067
0.5	0.3460	0.3506 ± 0.0031
0.7	0.2261	0.2311±0.0051

source, and an iterative procedure was used to represent the target source of one model by a D-BMAP in the other model. In Table 2, we show the throughput results for the two cases where the router operates with, and without RED. The table also shows results obtained from the *ns-2* simulator, together with the 95% confidence intervals. The results from the model are very close to the simulation results, and the error is limited to less than 2%. The results show that when RED is used, the effect of the different propagation delays on the throughput is slightly reduced. This is particularly apparent in the second case in the table when the two sources have different propagation delays.

We also modeled a TCP source, in addition to 5 UDP sources, i.e., the backoff probability is 0, who transmit to a common destination through a common router. Each of the UDP sources alternates between exponentially distributed on and off periods, with the average duration of the on period being sufficient to transmit five packets in the constant bit rate mode. The off period average duration is adjusted to reflect one of three load levels, namely, 0.3, 0.5 and 0.7. The router buffer size is 15 packets, the maximum window size of the target TCP soruce is 16 packets, the round trip propagation delay is 25 packets for all sources, and the time-out interval of the TCP source is 75 packet transmission times. Each segment is 512 bytes, and all links have a transmission rate of 45 Mb/s. This is the same scenario modeled in Figure 3 when the backoff probability is 0. The throughput of the TCP source, as obtained from the model and the *ns-2* simulation are shown in Table 3. Note that the 95% confidence intervals are also shown for the simulation results.

VII.2 Effect of System Paramters

We next consider the effect of different parameters, such as the window size, the propagation delay, the background interference traffic burstiness, as well as its responsiveness to lost segments[11]. The background interference traffic is called p_b-responsive if it backs off immediately with probability p_b when a packet is discarded due to congestion. UDP traffic can therefore be regarded as 0-responsive, while a 1-responsive traffic source backs off immediately, e.g., has a zero propagation delay. TCP traffic sources back off, but the effect of backoff, and rate reduction, does not appear at the router until after the segment loss is detected at the source. We can therefore model this effect using p_b-responsive sources, where $0 < p_b < 1$. In the examples below, we consider three levels of responsiveness of interference traffic, namely, 0, 0.5 and 1.

Effect of Background Traffic

In the first example, shown in Figure 3, we show the throughput achieved by the target TCP source when the router buffer size is 15 packets, the maximum window size of the target source is 16 packets, the round trip propagation delay is 25 packets[12], and a time-out interval of 75 packet times[13]. The packet and window sizes are default values in some Sun Solaris implementations [3]. In the figure, the target TCP source throughput is shown versus the load offered by the interference traffic. The interference traffic is modeled as a group of five on-off sources, where the distributions of both the on and the off periods are exponential. During the on period, which has a mean duration of 5 slots, each source generates a segment in a slot with probability 1. The length of the off period is adjusted in order to achieve the desired load level. As mentioned above, three responsiveness levels are used.

As expected, it is clear from the figure that as the interference load increases, the target source throughput decreases almost linearly with the load. As has also been expected, as the responsiveness level of the background traffic increases, the target node throughput will increase. This is also clear from the figure. However, the throughput achievable under the 1-responsive case is not much better than that under the 0-responsive case. This can be attributed to the fact that when the queue is full, it is very likely that it will drop packets from most active sessions, including the target source. Therefore, the target session will reduce its window in all cases. However, with the 1-responsive case, there is a slightly greater probability that a packet will not be dropped from the target source since the background sources have already backed off.

[11] Examples run with and without RED-based routers yielded results which are marginally different. Therefore, for the remainder of this section we only present results without using RED-based routers.

[12] This is equivalent to about 250 Km one-way separation between the source and the destination, assuming a DS-3 (45 Mb/s) link, and 512-byte TCP segments, and an advertised receiver window size of 8 Kbytes.

[13] Using the same assumptions for calculating the round trip propagation delay, this time-out interval is equivalent to 6.75 ms.

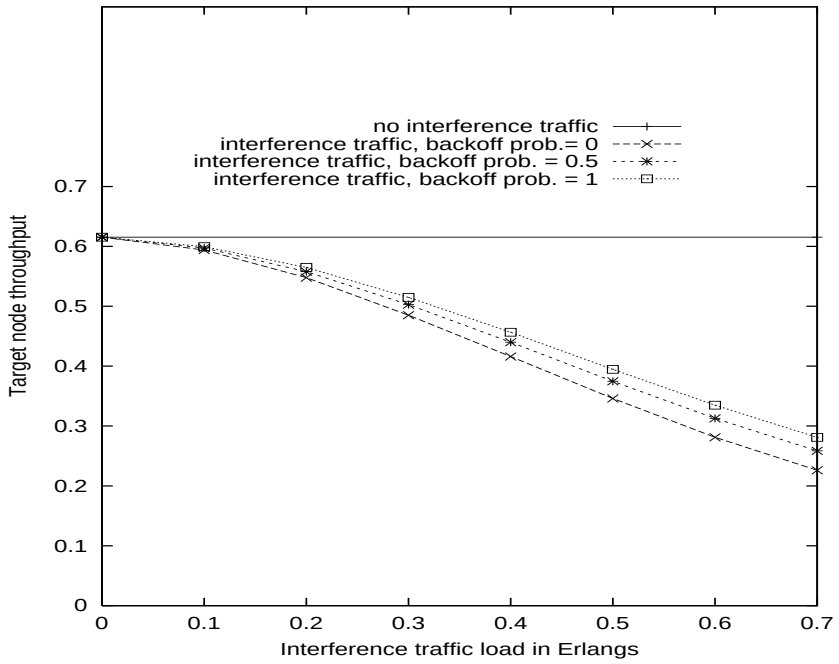

Figure 3: Target TCP source throughput: window size = 16, buffer size =15, round trip delay = 25 and time-out interval = 75

We also study the effect of the background traffic burstiness, as shown in Figure 4. We retain the same parameters of Figure 3, including the mean duration of the off periods, except that we reduce the background traffic burstiness inside the on period. Within the on period, the interval between successive packets is exponentially distributed with a mean of 5 slots, and the average number of packets within this period is kept at 5. Surprisingly enough, the reduced burstiness does not help the target source at all. In fact, the target source achievable throughput is reduced. Taking the dynamics of the TCP protocol into account, suppose the background packets arrive back to back. Then, they may cause successive packets from the target source to be discarded, and the window size will be reduced by one half. However, when the background traffic packets arrive with a time separation, then it is likely that these packets will overlap more than one window, especially under heavy load, and when the active period and the propagation delay are of the same order. Thus, they cause the window size of the TCP source to be reduced twice, hence reducing the throughput. Notice also that in this case increasing the responsiveness of the background traffic has a slightly more positive effect on the TCP source throughput, which is due to the fact that the background traffic refrains from sending any more packets that may interfere with the following window(s).

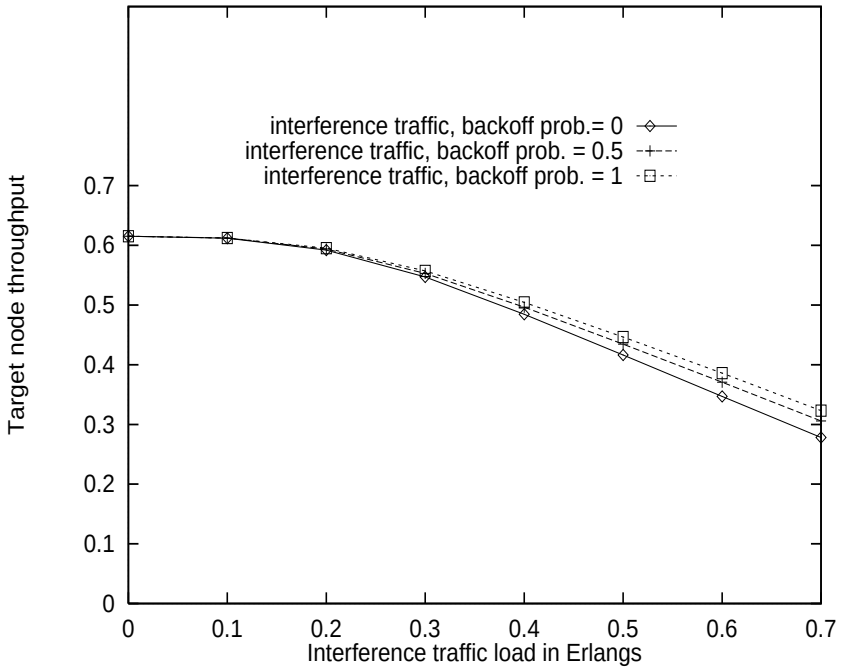

Figure 4: Target TCP source throughput: window size = 16, buffer size = 15, round trip delay = 25 and time-out interval = 75; reduced burstiness

Effect of Buffer Size

In the next example, shown in Figure 5, we double the buffer size to 30. The effect of this increase in the buffer size is an increase the throughput by about 10% at heavy background load. It will therefore take a substantial increase in the buffer size in order to achieve any measurable improvement in the target source throughput.

Effect of Window Size

In Figure 6, we study the effect of changing the window size. While we keep all the parameters of the example in Figure 3, the target TCP source maximum window size is reduced to 8 packets. Although the achievable throughput in the absence of background traffic has now been reduced to one half, which is due to the limit imposed by the maximum window size, the achievable throughput under heavy background traffic is only slightly worse than the first case, i.e., with a maximum window size of 16. This is due to that fact that under such load conditions the router is congested, packets are dropped frequently, and the TCP source window size is very small. The mean window size under the 0.7 load level in both cases is comparable: 8.79 and 6.84 for the 16 and 8 maximum window sizes, respectively, and with $p_b = 1$ for the background traffic.

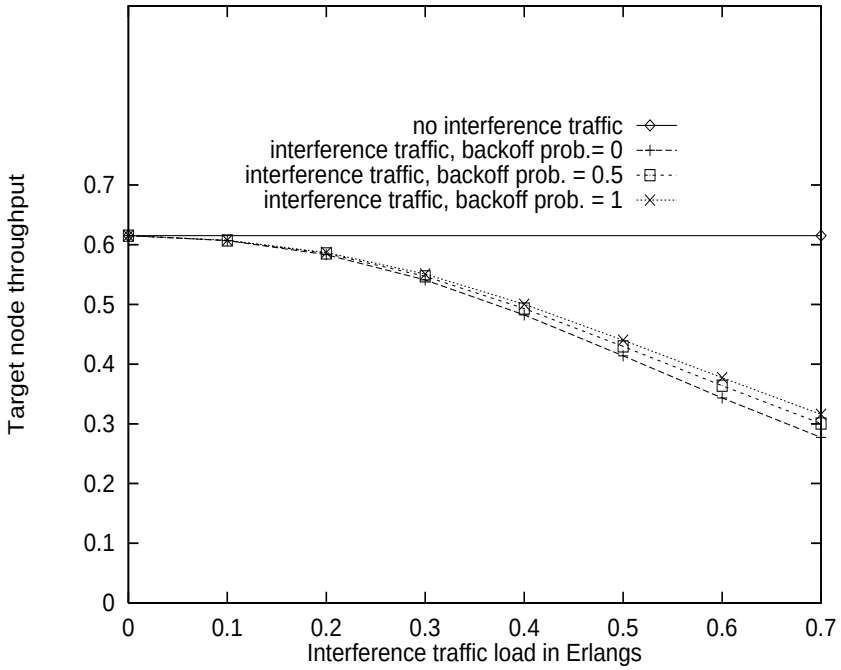

Figure 5: Target TCP source throughput: window size = 16, buffer size = 30, round trip delay = 25 and time-out interval = 75

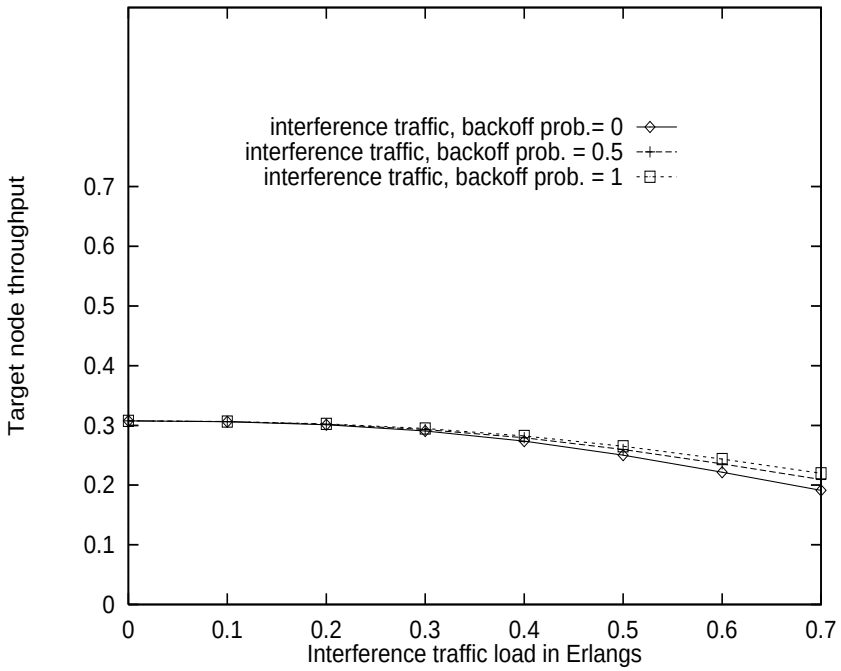

Figure 6: Target TCP source throughput: window size = 8, buffer size = 15, round trip delay = 25 and time-out interval = 75

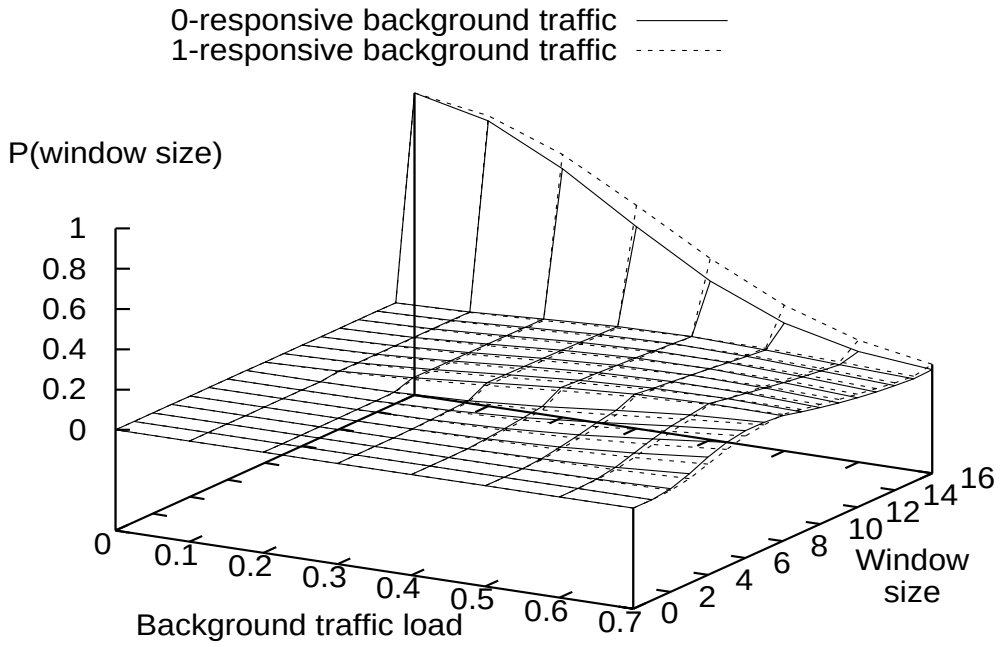

Figure 7: Window size probability mass function

VII.3 Window Size Distribution

Next, we study window distribution functions under different conditions. In Figure 7 we show the probability mass function of the window size using the same scenario of Figure 3, for different levels of background traffic load. We consider two background traffic responsiveness levels, 0 and 1. It is clear that under light load the window is at its maximum size, 16, most of the time. Also, the window size almost never goes below 8 (half the maximum window size), which indicates that if a packet is dropped from the TCP source, this is done when the window size is at the maximum window size, 16. Under heavy load, the mass of the window probabilities are close to half the maximum window size. This is due to the fact either losses take place in this region, or take place at higher values, and then the window is halved. This is evident from Figure 8 which shows the window size probability mass function when a packet is discarded (for both 0-responsive and 1-responsive background traffic cases). Under heavy load, the probability that the window size is at its maximum when a loss occurs is relatively large (around 0.1), especially with the 1-responsive traffic. This is due to the fact that this value is a boundary where the window cannot increase further, and as such the system stays at this value for some time. This also contributes to the window distribution being centered around half the maximum window size due to packet discarding, and halving of the window size. It is to be also noted that when the background traffic

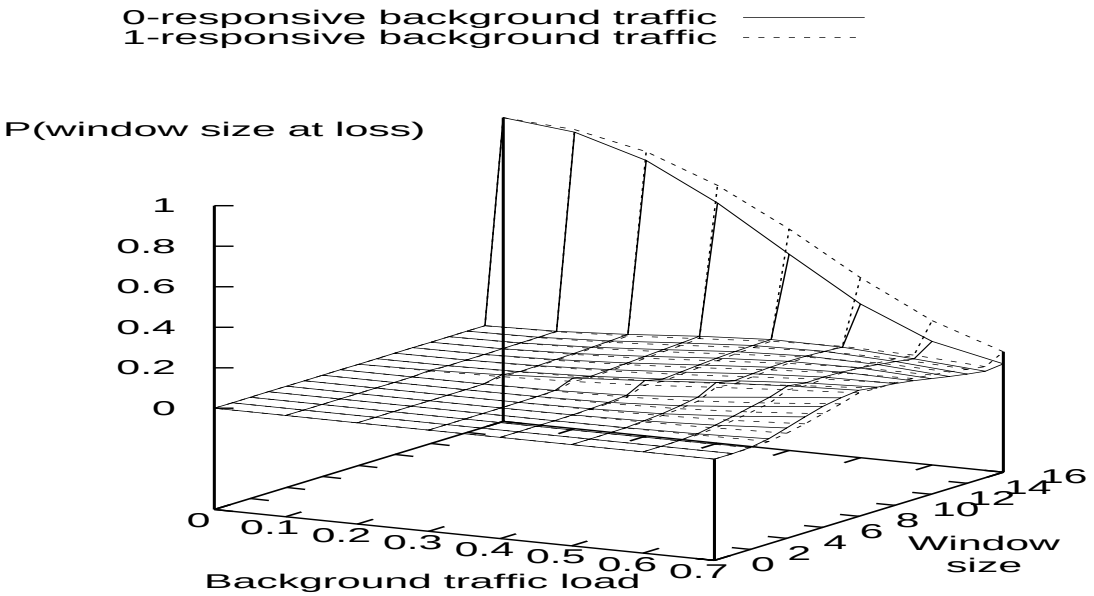

Figure 8: Probability mass function of window size when a loss occurs

is responsive, the window tends to be slightly larger. For example, under the 0.7 load case, the mean window size is 7.67 with the 0-responsive traffic, while it is 8.79 with the 1-responsive traffic.

If we compare the probability mass functions of the unconditional window size (Figure 7) and the window size when a loss occurs (Figure 8) we observe that smaller window sizes have slightly larger probabilities in the former figure (the comparison is shown in Figure 9 for the case of 0-responsive traffic). This is due to the fact that an unconditional probability of a being at certain window size is due to two components: a window size increase from a smaller window size, or window size halving because of packet discarding. The latter component is smaller if a window size has just been reached after a packet loss.

VII.4 Packet Discarding Distribution

Finally, we take a closer look at the packet discarding probability within different windows. In Figure 10 we plot the probability of the position of the first packet loss for different window sizes, and for two cases of background traffic load, namely 0.1 and 0.7. The responsiveness levels are 1 and 0 for the two traffic cases, respectively. This corresponds to two extreme cases. Under light load, the first loss tends to occur towards the end of the window when the buffer size grows during the window active period. However, there is a slightly greater probability for the first loss to occur

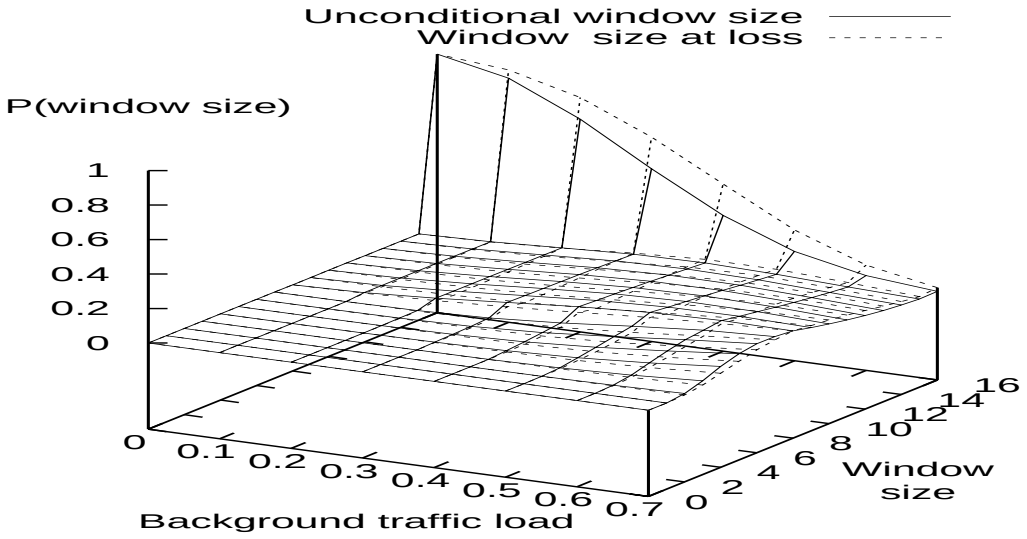

Figure 9: Comparison between the probability mass functions of the unconditional window size, and the window size at loss instants

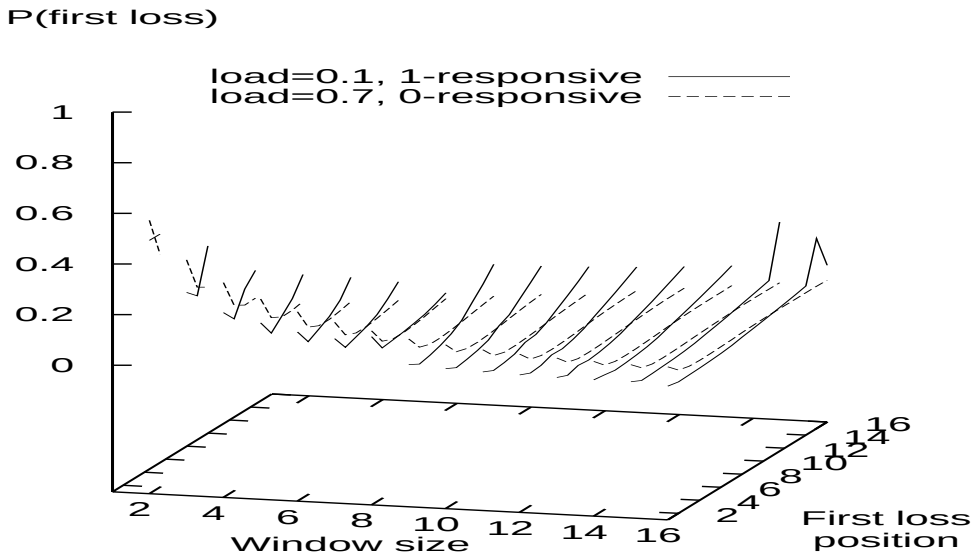

Figure 10: Probability mass function of first loss position

at the start of the window, namely on the first packet. This is due to accumulating traffic from the previous window and the background traffic. Notice also that at large window sizes near or at the maximum window size, there is a large mass of loss probability. This is because under light load the window stays there for a long time. Under heavy load there is a greater tendency for the first loss to occur in the first few positions at the beginning of the window, since the queue is already full. If that does not happen, the position of the first loss is almost evenly distributed over the entire window, since the buffers are already full.

It is also of interest, and importance, to explore the probability of a packet being discarded in a certain position, given that loss has already occurred in the window by discarding an earlier packet. This is shown in Figure 11, which depicts the distribution of the distance to the following loss positions, given the position of the first loss. The figure is for the case in which the window size is 12. It is shown that once a loss occurs, the following packet in the same window is discarded with a relatively large probability. The following packets in the same window are also discarded with a smaller, but relatively also large probabilities[14]. Such losses from the background traffic occur very close to the point where the first packet is lost from the TCP source. However, the probability of discarding packets from the following partial window (distances greater than the window size - the

[14]In the case of 1-responsive traffic, the probability of loss drops significantly due to background source back off.

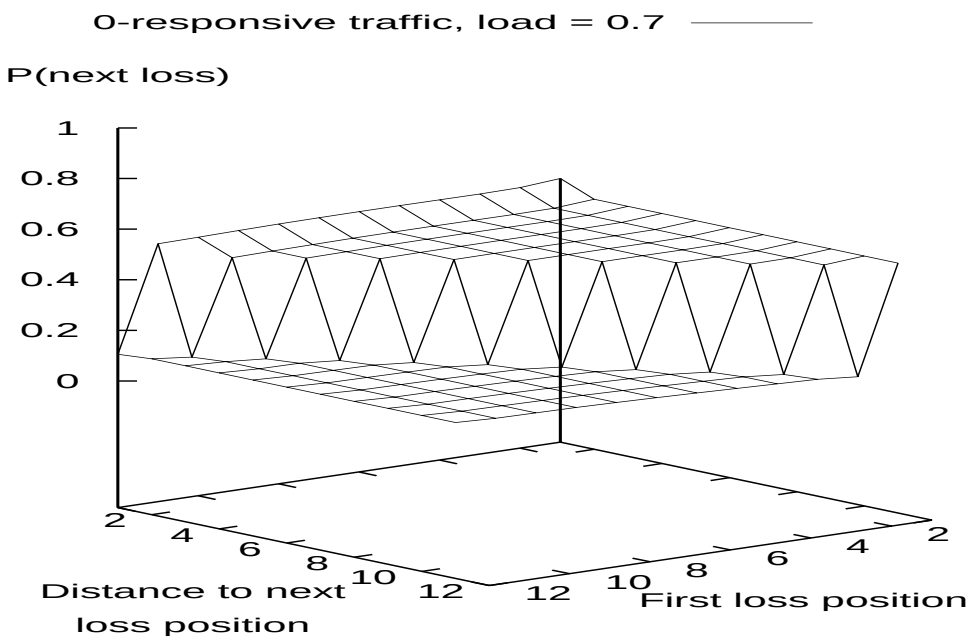

Figure 11: Probability mass function of successive loss positions, given the position of the first loss

first loss position) is very small, and almost negligible.

These observations reveal that the assumption made in [15, 16] in which all packets following the first discarded packet in a window are also discarded, holds in an approximate manner only, and under heavy load. However, such assumptions are not expected to impact the throughput calculations under light load significantly, since packet losses are already very rare. It is to be noted that the assumption may hold in other situations in which the on period of the background traffic is much longer.

VII.5 Summary of Results

This section has shown that the model can accurately capture the dynamics of TCP Reno, with and without RED routers, as it accurately keeps track of the buffer evolution process, and the background arrival process. The model has been also used to study the effect of the background traffic intensity and burstiness, the router buffer size, and the maximum window size on the throughput of the target source. We summarize such throughput results in Table 4, where there are four columns of results. The first column is a baseline case, in which the maximum window size is 16, the router buffer size is 15, and the five background traffic sources each generates a packet in every slot in the on period. The next three columns correspond to cases where:

Table 4: Summary of results

	$W_{max} = 16,\ B = 15$ bursty traffic	$W_{max} = 16,\ B = 30$ bursty traffic	$W_{max} = 8,\ B = 15$ bursty traffic	$W_{max} = 16,\ B = 15$ reduced burstiness
$p_b = 0.0$	0.226	0.277	0.191	0.278
$p_b = 0.5$	0.258	0.301	0.209	0.306
$p_b = 1.0$	0.281	0.316	0.220	0.323

- the buffer size is increased to 30,
- the maximum window size is reduced to 8, and
- the background sources burstiness is reduced by spacing packet generation according to an expontial distribution with mean 5 slots.

All results are at background traffic load of 0.7 Erlangs. Also, three backoff probabilities, namely, 0, 0.5 and 1, are used.

Finally, it is to be noted that we have investigated the effect of increasing the time-out interval on the throughput, and it was found that it has very little effect, even at heavy load, since most packet losses are detected using the duplicate acknowledgment method. For example, with 0.5-responsive background traffic and a 0.7 load level, the throughput was 0.258, 0.253 and 0.244 when the χ was set to 75, 150 and 300 packet times, respectively.

VIII Conclusions

This paper has presented an accurate discrete time model of the TCP Reno protocol in the presence of interference from background traffic. The background traffic was modeled as a D-BMAP process, which was modified such that the transitions between the phases are dependent on the number of packet losses from the process. Two levels of modeling were used, one at the packet transmission time level, and the other at the window transmission level. In this model, the basic features of the TCP Reno protocol were modeled, and several performance measures were derived. Several numerical results were obtained using this model, including throughput and window and packet loss distributions. Although the results included a limited number of sources, namely, two and six sources, which were modeled using a MMPP, they were presented as a proof of concept, and to enable comparison to simulation. Procedures for evaluating the parameters of the D-BMAP similar to those presented in [7, 8] can be employed.

The model was also extended to include the case in which routers employing the RED strategy are used. The RED strategy was modeled by keeping track of a function of the difference between the instantaneous and average queue sizes. Since this extension resulted in a very large state space, some approximations were introduced to make the state space manageable. Comparison to simulation results showed that this approximation still results in an acceptable accuracy.

The work in this paper can be extended in a number of ways. The model can be extended to include other active queue management schemes. In addition, it can also be extended to include several TCP options, such as selective acknowledgment (SACK) [39] and forward acknowledgment (FACK) [40]. The model can also be modified to use the explicit congestion notification (ECN) option by IP routers [41] instead of packet discarding by the routers. Modifying the model to implement TCP pacing [42] is also a simple task. Other extensions include the modeling of variable segment sizes. This, however, can be an involved extension since the round trip delay, τ, needs to be expressed in terms of variable packet transmission times.

Acknowledgements

The author wishes to acknowledge the comments made by the anonymous reviewers, which helped improve the presentation of the paper. The author also would like to thank J. Al-Karaki for his help with the *ns-2* simulator.

References

[1] M. Allman, V. Paxson, and W. Stevens, "Tcp congestion control." Network Working Group Request for Comment, RFC 2581, Apr. 1999.

[2] T. Socolofsky and C. Kale, "A tcp/ip tutorial." Network Working Group Request for Comment, RFC 1180, Jan. 1991.

[3] W. R. Stevens, *TCP/IP Illustrated, Volume 1: The Protocols.* Addison-Wesley, 1994.

[4] S. Floyd and V. Jacobson, "Random early detection gateways for congestion avoidance," *IEEE/ACM Transactions on Networking*, vol. 1, pp. 397–413, Aug. 1993.

[5] C. Blondia and O. Casals, "Performance analysis of statistical multiplexing of vbr sources," in *Proceedings of the IEEE INFOCOM*, pp. 828–83, 1992.

[6] A. E. Kamal, "A discrete-time model of tcp reno with background traffic interference," in *in the proceedings of the IEEE/ACM International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS)*, pp. 445–452, Oct. 2002.

[7] A. Klemm, C. Lindemann, and M. Lohmann, "Modeling ip traffic using the batch markovian arrival process," *Performance Evaluation*, vol. 54, pp. 149–173, 2003.

[8] D. P. Heyman and D. Lucantoni, "Modeling multiple ip traffic streams with rate limits," *IEEE/ACM Transactions on Networking*, vol. 11, pp. 948–958, Dec. 2003.

[9] G. R. Wright and W. R. Stevens, *TCP/IP Illustrated, Volume 2: The Implementation*. Addison-Wesley, 1995.

[10] D. M. Lucantoni, "New results on the single server queue with a batch markovian arrival process," *Communications in Statistics: Stochastic Models*, vol. 7, pp. 1–46, 1991.

[11] M. F. Neuts, "The c-server queue with constant service times and a versatile markovian arrival process," in *Applied Probability - Computer Science: The Interface*, pp. 31–67, 1981.

[12] S. Floyd, "Connections with multiple congested gateways in packet-switched networks, part 1: One-way traffic," *ACM Computer Communication Review*, pp. 30–47, Oct. 1991.

[13] T. V. Lakshman and U. Madhow, "The performance of tcp/ip for networks with high bandwidth-delay products and random loss," *IEEE/ACM Transactions on Networking*, vol. 5, pp. 336–350, June 1997.

[14] A. Kumar, "Comparative performance analysis of versions of tcp in a local network with a lossy link," *IEEE/ACM Transactions on Networking*, vol. 6, pp. 485–498, Aug. 1998.

[15] J. Padhye, V. Firoiu, d. Towsley, and J. Kurose, "Modeling tcp throughput: A simple model and its empirical validation," in *Proceedings of ACM SIGCOMM Symposium*, pp. 303–314, 1998.

[16] J. Padhye, V. Firoiu, d. Towsley, and J. Kurose, "Modeling tcp reno performance throughput: A simple model and its empirical validation," *IEEE/ACM Transactions on Networking*, vol. 8, pp. 133–145, Apr. 2000.

[17] N. Cardwell, S. Savage, and T. Anderson, "Modeling tcp latency," in *Proceedings of the IEEE INFOCOM*, 2000.

[18] B. Sikdar, S. Kalyanaraman, and K. S. Vastola, "Analytic models for the latency and steady-state throughput of tcp taho, reno and sack," *IEEE/ACM Transactions on Networking*, vol. 11, no. 6, pp. 959–971, 2003.

[19] M. A. Marsan, E. de Souza e Silva, R. L. Cigno, and M. Meo, "A markovian model for tcp over atm," *Telecommunication Systems Journal*, vol. 12, pp. 341–368, 1999.

[20] P. Brown, "Resource sharing of tcp connections with different round trip times," in *Proceedings of the IEEE INFOCOM*, 2000.

[21] C. Casetti and M. Meo, "A new approach to model the stationary behavior of tcp connections," in *Proceedings of the IEEE INFOCOM*, 2000.

[22] A. Abouzeid, S. Roy, and M. Azizoglu, "Stochastic modeling of tcp over lossy links," in *Proceedings of the IEEE INFOCOM*, 2000.

[23] A. Veres and M. Boda, "The chaotic nature of tcp congestion control," in *Proceedings of the IEEE INFOCOM*, 2000.

[24] O. Gusak and T. Dayar, "A generalization of a tcp model: Multiple source-destination case with arbitrary lan as the access network," in *System Performance Evaluation: Methodologies and Applications* (E. Gelenbe, ed.), pp. 39–49, CRC, 2000.

[25] J.-L. Costeux, "Fluid analysis of tcp connections over abr vcs," in *System Performance Evaluation: Methodologies and Applications* (E. Gelenbe, ed.), pp. 97–111, CRC, 2000.

[26] S. Shakkottai, A. Kumar, A. Karnik, and A. Anvekar, "Tcp performance over end-to-end rate control and stochastic available capacity," *IEEE/ACM Transactions on Networking*, vol. 9, no. 4, pp. 377–391, 2001.

[27] E. Altman, K. Avrachenkov, and C. Barakat, "Tcp in the presence of bursty losses," *Performance Evaluation*, vol. 42, pp. 129–147, 2000.

[28] E. Altman, K. Avrachenkov, and C. Barakat, "A stochastic model of tcp/ip with stationary random losses," in *Proceedings of ACM SIGCOMM Symposium*, 2000.

[29] F. Baccelli and D. Hong, "Tcp is max-plus linear and what it tells us on its throughput," in *Proceedings of ACM SIGCOMM Symposium*, pp. 219–230, 2000.

[30] T. Bonald, M. May, and J.-C. Bolot, "Analaytic evaluation of red performance," in *Proceedings of the IEEE INFOCOM*, 2000.

[31] H. Alazemi, A. Mokhtar, and M. Azizoglu, "Stochastic approach for modeling random early detection gateways in tcp/ip networks," in *Conference Record of the International Conference on Communications (ICC)*, 2001.

[32] V. Misra, W.-B. Gong, and D. Towsley, "Fluid-based analysis of a network of aqm routers supporting tcp flows with an appliction to red," in *Proceedings of ACM SIGCOMM Symposium*, (Stockholm, Sweden), pp. 151–160, Aug. 2000.

[33] V. Firoiu and M. Borden, "A study of active queue management for congestion control," in *Proceedings of the IEEE INFOCOM*, 2000.

[34] V. Sharma and P. Purayastha, "Performance analysis of tcp connections with red control and exogenous traffic," in *Proceedings of the Conference on Global Communications (GLOBECOM)*, pp. 1794–1799, 2001.

[35] C. Barakat, "Tcp/ip modeling and validation," *IEEE Network*, vol. 15, pp. 38–47, May/June 2001.

[36] A. E. Kamal, "Efficient solution of multiple server queues with applications to the modeling of atm concentrators," in *Proceedings of the IEEE INFOCOM*, pp. 248–254, 1996.

[37] J. G. Kemeny and J. L. Snell, *Finite Markov Chains*. New York: van Nostrand, 1960.

[38] "http://www.isi.edu/nsnam/ns."

[39] S. F. M. Mathis, J. Mahdavi and A. Romanow, "Tcp selective acknowledgement options." Network Working Group Request for Comment, RFC 2018, Oct. 1996.

[40] M. Mathis and J. Mahdavi, "Forward acknowledgement: Refining TCP congestion control," in *ACM SIGCOMM'96*, pp. 281–291, 1996.

[41] K. Ramakrishnan and S. Floyd, "A proposal to add explicit congestion notification (ecn) to ip." Network Working Group Request for Comment, RFC 2481, Jan. 1999.

[42] J. Kulik *et al.*, "Paced tcp for high bandwidt-delay networks," in *IEEE/ACM WOSBIS*, 1999.

Appendix

Table 5: List of Symbols

Symbol	Meaning
τ	round trip propagation delay
χ	timeout interval
B	router's buffer size
W_{max}	maximum (receiver advertized) window size
$\mathcal{F}$	number of D-BMAP phases
$\alpha_f(n)$	Pr(n packet arrivals | current phase is f)
$\beta_{ff'}(l)$	Pr(next phase is f'| current phase is f, and l packets have been discarded)
Ψ	the system state at the embedding points (macroscopic model)
Ω	the window size of the target session at the embedding points
Θ	the router's queue size at the embedding points
Φ	the D-BMAP phase at the embedding points
ψ	the system state at the slot boundaries (microscopic model)
ω	the window size of the target session at the slot boundaries
θ	the router's queue size at the slot boundaries
ϕ	the D-BMAP phase at the slot boundaries

Symbol	Meaning
$x(i,m\|j,n,l,\rho)$	P(within a slot, the buffer accepts i packets from the target session, given that j packets have already been accepted from the same session, and also accept m out of n packets from the background session, given l packets have been accepted, and the system state at the beginning of the slot is ρ)
$y(0,m\|0,n,l,\rho)$	similar to $x()$, except that no packets are generated or accepted from the target session
$z(i,m\|j,n,l,\rho)$	P(within a slot, i packets are discarded from the target session, given j have already been discarded, and also m packets are accepted out of n packets from the background session, given l have been accepted, and given the system state at the beginning of the slot is ρ).
$P(accept\|l,\rho)$	probability of accepting a packet given that l packets have been accepted, and the state is ρ.
$P(discard\|l,\rho)$	probability of discarding a packet given that l packets have been accepted, and the state is ρ.
$\mathbf{S}$	Transition probability matrix between slot boundaries. A subscript of s, i, or l is added if the event of success, idle of loss from the target is jointly considered
$\mathbf{W}$	Transition probability matrix between embedding points. A subscript i is added ($\mathbf{W}_i$) if the initial window size is i. The subscript is $i+$ or $i-$, if success or loss from the target source is encountered within the interval.
$\mathbf{W}^T$	Similar to $\mathbf{W}$, but joint with the event of loss that is detected through timeout. The window size may not increase, therefore $\mathbf{W}^T_{i+} = 0$.
$\mathbf{W}^D$	Similar to $\mathbf{W}^T$, but when loss is detected through three duplicate acknowledgments.
$\mathbf{W}^{DS}$	Similar ot $\mathbf{W}^D$, and loss is detected during the same window as the loss.
$\mathbf{W}^{DN}$	Similar ot $\mathbf{W}^D$, and loss is detected during the window following the window of the loss.
$W(q)$	Block matrix row of $\mathbf{W}$ when $\Psi = q$. Similar notation is used for $\mathbf{W}^T$, $\mathbf{W}^D$, $\mathbf{W}^{DS}$ and $\mathbf{W}^{DN}$.
$\mathcal{W}$	Transition probability matrix between two successive window evolution embedding points, started with a window size of k, when the target source is in the slow start phase.
$\mathbf{U}^{(k)}(j)$	Transition probability matrix with exactly j packets being accepted from the target session in the router's queue within k slots.
q_n	Instantaneous queue size when the nth packet arrives.
a_n	Average queue size when the nth packet arrives.
η	A scaling factor used for scaling the difference between q_n and a_{n-1}.
Δ	A function of the difference between the average and instantaneous queue sizes.