Steady State Analysis of the RED Gateway: Stability, Transient Behavior, and Parameter Setting

Hiroyuki Ohsaki, Masayuki Murata, and Hideo Miyahara

Graduate School of Engineering Science, Osaka University
1-3 Machikaneyama, Toyonaka, Osaka 560-8531, Japan

(Phone) +81-6-6850-6588
(Fax) +81-6-6850-6589
(E-mail) oosaki@ics.es.osaka-u.ac.jp

Abstract

Several gateway-based congestion control mechanisms have been proposed to support an end-to-end congestion control mechanism of TCP (Transmission Control Protocol). One of promising gateway-based congestion control mechanisms is a RED (Random Early Detection) gateway. Although effectiveness of the RED gateway is fully dependent on a choice of control parameters, it has not been fully investigated how to configure its control parameters. In this paper, we analyze the steady state behavior of the RED gateway by explicitly modeling the congestion control mechanism of TCP. We first derive the equilibrium values of the TCP window size and the buffer occupancy of the RED gateway. Also derived are the stability condition and the transient performance index of the network by using a control theoretic approach. Numerical examples as well as simulation results are presented to clearly show relations between control parameters and the steady state behavior. Our findings are: (1) max_p (maximum packet marking probability) mostly affects the RED's buffer occupancy, (2) the network becomes more stable as the number of TCP connections or the bandwidth–delay product increases, and (3) min_{th} (minimum threshold) is a key parameter for optimizing the transient performance. We finally discuss how control parameters of the RED gateway should be configured for achieving better performance.

keywords: RED (Random Early Detection) gateway, TCP (Transmission Control Protocol), Stability, Transient behavior, Paramter setting, Control theory

1 Introduction

In a packet-switched network, a feedback-based congestion control mechanism is essential to realize efficient data transfer services. Its main objective is to prevent packet losses in the network, as well as to utilize network resources effectively. The current Internet uses a window-based flow control mechanism in its TCP (Transmission Control Protocol), as a feedback-based congestion control mechanism. For instance, a version of TCP called *TCP Reno* uses packet losses in the network as congestion indication of the network [1, 2]. Until a packet loss occurs in the network, TCP Reno gradually increases its window size. As the window size exceeds its available bandwidth, excess packets are waited at the buffer of an intermediate gateway (or router) for some period. If the window size increases further, the buffer of the gateway overflows, leading to a packet loss. The source host detects the occurrence of the packet loss by, for example, receiving duplicate ACKs, and quickly reduces its window size. After reduction of the window size, congestion in the network will soon be relieved. Once congestion in the network is over, the source host increases its window size again. In short, the congestion control mechanism of TCP first increases its window size, and as soon as it detects packet losses in the network, it reduces its window size. The congestion control mechanism of TCP repeats this process indefinitely.

The congestion control mechanism of TCP was designed to work without any knowledge on underlying network components including a gateway. Namely, the congestion control mechanism of TCP assumes nothing about a gateway's operation algorithm. It is because neither a packet scheduling discipline nor a packet discarding algorithm of a gateway is known to the source host in a real network. Actually, separation of TCP's congestion control mechanism from the gateway's operation algorithm is desirable when several types of congestion control mechanisms and gateway's algorithms co-exist in a network as in the current Internet. However, such generality of the congestion control mechanism of TCP may significantly limit the network performance.

Accordingly, several gateway-based congestion control mechanisms have been proposed to support an end-to-end congestion control mechanism of TCP [3, 4, 5]. One of promising gateway-based congestion control mechanisms is a RED (Random Early Detection) gateway [4]. The key idea of the RED gateway is to keep the average queue length (i.e., the average number of packets in the buffer) low. Basically, the RED gateway randomly discards an incoming packet with a probability that is proportional to the average queue length. The operation algorithm of the

RED gateway is so simple that the RED algorithm can be easily implemented. The authors of [4] have claimed advantages of the RED gateway over a conventional drop-tail gateway as follows: (1) the average queue length is kept low, so that an end-to-end delay of a TCP connection is also kept small, (2) the RED gateway has no bias against bursty traffic as in the drop-tail gateway, and (3) a global synchronization problem of TCP connections found in the drop-tail gateway is avoided.

There have been several simulation studies of the RED gateway [4, 6, 7]. But there still remain open issues. The biggest one is how to choose several control parameters of the RED gateway. Despite the fact that effectiveness of the RED gateway heavily relies on a choice of control parameters [6, 8, 9], it has not been fully investigated how to configure those control parameters. The authors of [4] have proposed a recommended set of control parameters, but it is just an empirical guideline without any theoretical basis. There are only a few analytical studies on the RED gateway. In [10], the authors have analyzed the performance of the RED gateway for both bursty and less bursty traffic. However, their analytic model is very simple and not realistic; input traffic to the RED gateway is modeled by a batch Poisson process, which completely neglects the dynamics of the congestion control mechanism of TCP. In [11], the authors have analyzed the dynamics of the RED gateway, but the input traffic is still limited to either a renewal process or an MMPP (Markov Modulated Poisson Process). Since the RED gateway was originally designed to cooperate with the congestion control mechanism of TCP, effect of TCP must be taken into account to understand a substantial property of the RED gateway.

In this paper, we analyze steady state behavior of the RED gateway by explicitly modeling the congestion control mechanism of TCP. We first derive equilibrium values of TCP's window size and the buffer occupancy of the RED gateway. Also derived are a stability condition and a transient performance index by using a control theoretic approach. Numerical examples as well as simulation results are presented to clearly show relations between control parameters and steady state behavior. Our findings are — (1) max_p (maximum packet marking probability) mostly affects RED's buffer occupancy, (2) a network becomes more stable as the number of TCP connections or a bandwidth–delay product increases, and (3) min_{th} (minimum threshold) is a key paramter for optimizing transient performance. We finally discuss how control parameters of the RED gateway should be configured for achieving better performance.

This paper is organized as follows. In Section 2, a network model that we will use throughout this paper is described, followed by a brief explanation of the RED gateway. In Section 3, we analyze the steady state behavior of the RED gateway when incoming traffic is controlled by the congestion control mechanism of TCP. Section 4 derives the stability condition and the transient performance index by using a control theoretic approach. In Section 5, we present several numerical examples to clearly show how control parameters of the RED gateway affect the steady state behavior. Simulation results are also presented to validate our approximate analysis. In Section 6, we discuss how control parameters of the RED gateway should be configured for achieving better performance. Finally, Section 7 concludes the current paper with a few remarks.

2 RED Algorithm and Analytic Model

Figure 1 illustrates our analytic model that will be used throughout this paper. It consists of a single RED gateway and the number N of TCP connections. All TCP connections have an identical (round-trip) propagation delay, which is denoted by τ [ms]. We assume that the processing speed of the RED gateway, which is denoted by B [packet/ms], is the bottleneck of the network. Namely, transmission speeds of all links are assumed to be faster than the processing speed of the RED gateway.

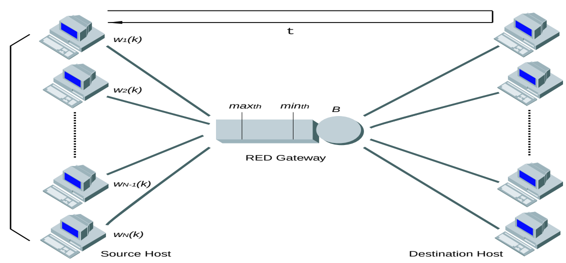


Figure 1: Analytic model.

We model the congestion control mechanism of TCP version Reno [2, 12] at all source hosts. A source host has a window size, which is controlled by the congestion control mechanism of TCP. By letting w_n be a current window size of the source host n ($1 \leq n \leq N$), it is allowed to send the number w_n of packets without receipt of ACK (Acknowledgement) packets. In other words, the source host n can send a bunch of w_n packets during a round-trip time. We then model the entire network as a discrete-time system, where a time slot of the system corresponds to a round-trip time of TCP connections. Note that a slot length changes at every slot since a round-trip time changes because of a queueing delay at the RED gateway. We define $w_n(k)$ [packet] as the window size of the source host n at slot k. All source hosts are assumed to have enough data to transmit; the source host n is assumed to always send the number $w_n(k)$ of packets during slot k.

The RED gateway has several control parameters. Let min_{th} and max_{th} be the *minimum threshold* and the *maximum threshold* of the RED gateway, respectively. These threshold values are used to calcurate a packet marking probability for every incoming packet. The RED gateway maintains an average queue length (i.e., the average number of packets waiting in the buffer). The RED gateway uses an exponential weighted moving average (EWMA), which is a sort of low-pass filters, to calculate the average queue length from the current queue length. More specifically, let q and $\overline{q}$ be the current and the average queue length. At every packet arrival, the RED gateway updates the average queue length $\overline{q}$ as

$$\overline{q} \quad \leftarrow \quad (1 - w_q)\,\overline{q} + w_q\, q \tag{1}$$

where w_q is a weight factor. We define $q(k)$ [packet] and $\overline{q}(k)$ [packet] be the current and the average queue lengths at slot k. We assume that both q and $\overline{q}$ will not change during a slot. As discussed in [10], this assumption is realistic for a small w_q.

Using the average queue length, the RED gateway calculates a packet marking probability p_b at every arrival of an incoming packet. Namely, the RED gateway calculates p_b as

$$p_b \quad = \quad \begin{cases} 0 & \text{if } \overline{q} < min_{th} \\ 1 & \text{if } \overline{q} \geq max_{th} \\ max_p \left(\frac{\overline{q} - min_{th}}{max_{th} - min_{th}} \right) & \text{otherwise} \end{cases} \tag{2}$$

where max_p (*maximum packet marking probability*) is another control parameter (see Fig. 2). The RED gateway randomly discards each incoming packet with probability p_b. The packet marking mechanism of the RED gateway is not per-flow basis, and the same packet marking probability is used for all TCP connections. Refer to [4] for the detailed algorithm of the RED gateway. A typical setting of control parameters of the RED gateway, which has been recommended by the authors of [4], is shown in Table 1. In latter half of this paper, we will discuss how control parameters of the RED gateway should be configured for achieving better performance.

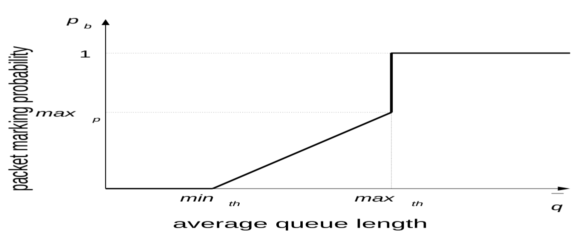


Figure 2: Calculation of the packet marking probability p_b from the average queue length $\overline{q}$.

Table 1: A recommended set of RED's control parameters

min_{th}	min. threshold value	5 [packet]
max_{th}	max. threshold value	15 [packet]
max_p	max. packet marking probability	0.1
w_q	weight factor for averaging	0.002

3 Steady State Analysis

3.1 Derivation of State Transition Equations

We first obtain the packet dropping probability at slot k. At the beginning of slot k, the source host n sends the number $w_n(k)$ of packets into the network. The RED gateway marks each arriving packet based on the average queue length. Since the average queue length $\overline{q}$ is assumed to be fixed within a slot, the packet marking probability $p_b(k)$ at slot k is also fixed. Provided that the average queue length $\overline{q}$ is between min_{th} and max_{th}, $p_b(k)$ is obtained from Eq. (2) as

$$p_b(k) \quad = \quad max_p \left(\frac{\overline{q}(k) - min_{th}}{max_{th} - min_{th}} \right) \tag{3}$$

Every arriving packet at the RED gateway is marked with probability [4]

$$\frac{p_b(k)}{1 - count \cdot p_b(k)}$$

where $count$ is the number of unmarked packets that have arrived since the last marked packet. The number of unmarked packets between two consecutive marked packets can be represented by an uniform random variable in $\{1, 2, \cdots, 1/p_b(k)\}$. Namely,

$$P_k[X = n] \quad = \quad \begin{cases} p_b(k) & 1 \leq n \leq 1/p_b(k) \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

Let $\overline{X}_k$ be the expected number of unmarked packets between two consecutive marked packets at slot k. $\overline{X}_k$ is obtained as

$$\begin{aligned} \overline{X}_k \quad &= \quad \sum_{n=1}^{\infty} n\, P_k[X = n] \\ &= \quad \frac{1/p_b(k) + 1}{2} \end{aligned} \tag{5}$$

We next derive state transition equations of the network. The state transition equations that we will derive hereafter are the basis of our steady state analysis. If all packets sent from a source host are *not* marked at the RED gateway, corresponding ACK packets will be returned to the source host after a round-trip time. In this case, the congestion control mechanism of TCP increases the window size by one packet. On the contrary, if any of packets are discarded by the RED gateway, the congestion control mechanism of TCP throttles the window size to half. We assume that all packet losses are detectable by duplicate ACKs [12]. Namely, the number of discarded packets in a slot is assumed to be less than three. The probability that at least one packet is discarded from $w_n(k)$ packets is given by

$$\frac{w(k)}{1/p_b(k)} \quad = \quad w(k)\, p_b(k)$$

Therefore, the window size at slot $k+1$ is given by

$$w(k+1) \quad = \quad \begin{cases} \frac{w(k)}{2} & \text{with prob. } w(k)\, p_b(k) \\ w(k)+1 & \text{otherwise} \end{cases} \tag{6}$$

By our definition of a slot, all packets that have sent in slot k are to be acknowledged until the beginning of slot $k+1$. The current queue length at slot $k+1$ is given by

$$\begin{aligned} q(k+1) \quad &= \quad q(k) + \sum_{n=1}^{N} w_n(k) - B\left(\tau + \frac{q(k)}{B}\right) \\ &= \quad \sum_{n=1}^{N} w_n(k) - B\tau \end{aligned} \tag{7}$$

We then focus on a relation between $\overline{q}(k)$ and $\overline{q}(k+1)$. As explained in Section 2, the RED gateway updates the average queue length at every packet arrival according to Eq. (1). In other words, the average queue length is updated $\sum_{n=1}^{N} w_n(k)$ times during slot k. Recall that the current queue length $q(k)$ is assumed to be fixed during a slot. The average queue length in slot $k+1$ is then given by

$$\begin{aligned} \overline{q}(k+1) \quad &= \quad (1-w_q)^{\sum_{n=1}^{N} w_n(k)} \overline{q}(k) \\ &\quad + \left\{1 - (1-w_q)^{\sum_{n=1}^{N} w_n(k)}\right\} q(k) \end{aligned} \tag{8}$$

The network model depicted in Fig. 1 is fully described by state transition equations given by Eqs. (6)–(8), and the state vector $\mathbf{x}(k)$

$$\mathbf{x}(k) = \begin{bmatrix} w_1(k) \\ \vdots \\ w_N(k) \\ q(k) \\ \overline{q}(k) \end{bmatrix} \tag{9}$$

3.2 Derivation of Average State Transition Equations

The state transition equations given by Eqs. (6)–(8) contain a probability, because the RED gateway marks each arriving packet in a probabilistic way. To analyze the steady state behavior of the RED gateway, we introduce *average state transition equations* that represent a typical behavior of TCP connections and the RED gateway. We also introduce a *sequence*, which is a series of adjacent slots in which all packets from a source host have been unmarked by the RED gateway. A typical evolution of a window size within a sequence is illustrated by Fig. 3. Note that this figure shows the case where the RED gateway discards one or more packets in slot $k-1$. The window size $w_n(k)$ is changed according to Eq. (6). Namely, as long as no packets from the source host is discarded by the RED gateway, the window size is incremented by one packet at every slot. If one or more packets are discarded, the window size is halved. Such a process will be repeated indefinitely by the congestion control mechanism of TCP.

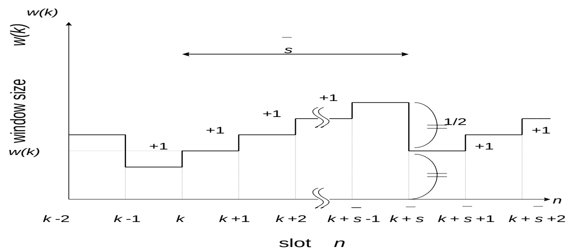


Figure 3: A typical evolution of the window size in a sequence.

The key idea in our steady state analysis is to treat the network as a discrete-time system where a time slot corresponds to a sequence, instead of a slot defined in Section 2. Let $\overline{s}_k$ be the average number of slots that consists a sequence beginning from slot k. Then, $\overline{s}_k$ satisfies the following inequality.

$$\overline{X}_k \quad \leq \quad \sum_{i=0}^{\overline{s}_k - 1} \sum_{n=1}^{N} w_n(k+i).$$

In what follows, due to space limitation, we only show the case where window sizes of all source hosts are synchronized; the window size of the source host n is equally given by $w(k)$. However, our steady state analysis can be easily applied to the case where window sizes of all source hosts are not identical. When window sizes of all source hosts are identical, the above inequality is rewritten as

$$\begin{aligned} \overline{X}_k \quad &\leq \quad \sum_{i=0}^{\overline{s}_k - 1} N\, w(k+i) \\ &= \quad N\left\{\overline{s}_k\, w(k) + \frac{\overline{s}_k(\overline{s}_k - 1)}{2}\right\} \end{aligned} \tag{10}$$

Solving for $\overline{s}_k$ yields

$$\overline{s}_k \quad = \quad \frac{1}{2} - w(k) + \frac{\sqrt{N^2(1-2w(k))^2 + 8N\overline{X}_k}}{2N} \tag{11}$$

Assuming that the number of slots in a sequence is deterministically given by $\overline{s}_k$, we derive the average state transition equations from slot k to $k+\overline{s}_k$. Since the RED gateway discards one or more packets during slot $k+\overline{s}_k-1$ (see Fig. 3), the average state transition equation from $w(k)$ to $w(k+\overline{s}_k)$ is obtained from Eq. (6) as

$$\begin{aligned} w(k+\overline{s}_k) &= \frac{w(k+\overline{s}_k-1)}{2} \\ &= \frac{w(k)+\overline{s}_k-1}{2} \end{aligned} \tag{12}$$

Similarly, the average state transition equation from $q(k)$ to $q(k+\overline{s}_k)$ is obtained from Eq. (7) as

$$\begin{aligned} q(k+\overline{s}_k) &= N\,w(k+\overline{s}_k-1) - B\,\tau \\ &= N\,(w(k)+\overline{s}_k-1) - B\,\tau \end{aligned} \tag{13}$$

We next derive the average state transition equation from $\overline{q}(k)$ to $\overline{q}(k+\overline{s}_k)$. Using Eq. (8), $\overline{q}(k+\overline{s}_k)$ is obtained as

$$\begin{aligned} \overline{q}(k+\overline{s}_k) &= (1-w_q)^{A_k}\,\overline{q}(k+\overline{s}_k-1) \\ &\quad + \left\{1-(1-w_q)^{A_k}\right\}\, q(k+\overline{s}_k-1) \end{aligned} \tag{14}$$

where

$$A_k = N\,w(k+\overline{s}_k-1)$$

Recall that $\overline{X}_k$ is the average number of unmarked packets between two consecutive marked packets, i.e., the average number of unmarked packets in a sequence. By assuming that the current queue length does not change excessively, $\overline{q}(k+\overline{s}_k)$ is approximated as

$$\overline{q}(k+\overline{s}_k) \simeq (1-w_q)^{\overline{X}_k}\,\overline{q}(k) + \left\{1-(1-w_q)^{\overline{X}_k}\right\} q(k) \tag{15}$$

The average state transition equations given by Eqs. (12), (13), and (15) describe the average behaviors of the window size, the current queue length, and the average queue length, respectively. Our approximated analysis will be validated in Section 5.

3.3 Derivation of Average Equilibrium Value

Because of the nature of TCP's congestion control mechanism, the window size of a source host oscillates indefinitely and is never converged to a equilibrium state. In what follows, we therefore derive a *average equilibrium value*, which is defined as the expected value in steady state, to understand a typical behavior of TCP connections and the RED gateway. Let w^*, q^*, and $\overline{q}^*$ be the average equilibrium value of the window size $w(k)$, the current queue length $q(k)$, and the average queue length $\overline{q}(k)$, respectively. The average equilibrium value can be easily obtained from Eqs. (12), (13), and (15) by equating $w(k) = w(k+\overline{s}_k)$ and so on.

$$\begin{aligned} w^* &= \sqrt{\frac{1}{4} + \frac{1}{3N}\left(\frac{max_{th}-min_{th}}{max_p(\overline{q}^*-min_{th})}+1\right)} \\ &\quad -\frac{1}{2} \end{aligned} \tag{16}$$

$$q^* = N\,w^* - B\,\tau \tag{17}$$

$$\overline{q}^* = q^* \tag{18}$$

Note that w^* does not represent the average window size in steady state. Instead, it represents the expected value of the *minimum* window size since w^* is the equilibrium value of the window size at the beginning of a sequence (see Fig. 3). The average window size in steady state is obtained from w^* as

$$\lim_{k\to\infty} \frac{\sum_{i=0}^{\overline{s}} w(k+i)\left(\tau + \frac{q(k+i)}{B}\right)}{\sum_{i=0}^{\overline{s}}\left(\tau + \frac{q(k+i)}{B}\right)} \simeq \frac{3w^*}{2} - 1 \tag{19}$$

where $q(k+i)$ is approximated by $q(k)$ for $0 \le i \le \overline{s}$. Note that solving Eqs. (16) and (17) gives closed-form solutions for w^* and q^*. However, we do not include here because those are lengthy.

4 Stability and Transient Behavior

In this section, we analyze stability and transient behavior of the network by using a control theoretic approach. Since incoming traffic at the RED gateway is flow-controlled by the congestion control mechanism of TCP, the window size of a source host oscillates and never converges to a equilibrium value. The operation of the RED gateway is changed by a probability, so it is difficult to analyze its stability and transient behavior. We therefore focus on their *average behaviors* by utilizing the average state transition equations derived in Section 3. In particular, we derive the stability condition and the transient performance index of the network by considering the average behaviors of the TCP connections and the RED gateway.

In Section 2, the average state transition equations for the discrete-time system illustrated by Fig. 1 are obtained in Eqs. (12), (13), and (15). The average equilibrium values of the system states are also obtained in Eqs. (16)–(18). Let us introduce $\delta\mathbf{x}(k)$ as the difference between the state vector $\mathbf{x}(k)$ and the average equilibrium point.

$$\delta\mathbf{x}(k) \equiv \begin{bmatrix} w(k)-w^* \\ q(k)-q^* \\ \overline{q}(k)-\overline{q}^* \end{bmatrix} \tag{20}$$

By lineally approximating $w(k)$, $q(k)$, and $\overline{q}(k)$ around their average equilibrium values, $\delta\mathbf{x}(k+\overline{s})$ can be written as

$$\delta\mathbf{x}(k+\overline{s}) \quad = \quad \mathbf{A}\,\delta\mathbf{x}(k)$$

where **A** is a state transition matrix. It is known that the stability and transient behavior of the system given by Eqs. (12), (13) and (15) around the equilibrium point are determined by the roots of the characteristic equation [13].

$$|s\mathbf{A} - \mathbf{I}| \quad = \quad 0 \tag{21}$$

Let $s_i (1 \le i \le 3)$ be the roots of the above characteristic equation. The system is stable if and only if all s_i's lie in the unit circle (*stability condition*). The stability condition for a given state transition matrix can be easily computed by, for example, Jury's criterion [13].

Also known is that the transient behavior of the system given by Eqs. (12), (13) and (15) around the equilibrium point is determined by s_i's. More specifically, the transient performance of the system is mostly determined by the following value (*transient performance index*).

$$s_{max} = \max_i(|s_i|) \tag{22}$$

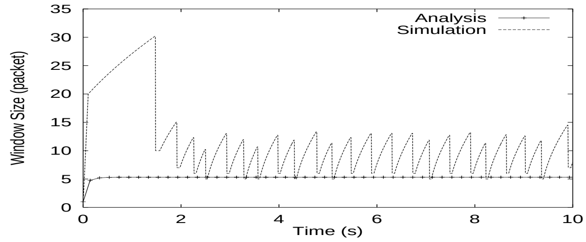
The smaller the transient performance index is, the faster the converges to the equilibrium point.

5 Numerical Examples and Simulation Results

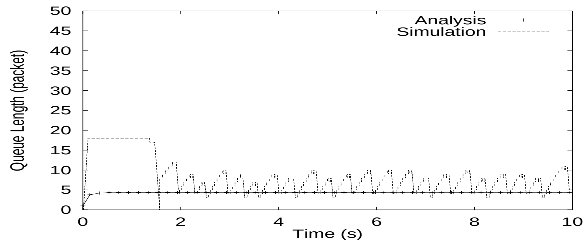
5.1 Validity of Approximated Analysis

We present a couple of simulation results to demonstrate the validity of our analysis since we have made several assumptions. We run simulation experiments for the same network model given by Fig. 1 using a network simulator *ns* [14]. We use the following network parameters in simulation: the processing speed of the RED gateway $B = 2$ [packet/ms] (i.e., about 1.5 Mbit/s for the packet size of 1,000 bytes) and the propagation delay $\tau = 1$ [ms]. For RED control parameters, the values shown in Table 1 are used.

Evolutions of TCP's window size and the current queue length of the RED gateway in simulation experiments are plotted in Figs. 4 and 5 for $N = 1$ and $N = 5$, respectively. We numerically compute the average window size $w(k)$ and the average queue length $q(k)$ (i.e., the minimum values in each sequence) from Eqs. (12), (13) and (15). In Figs. 4 and 5, those analytic results are also plotted. One can find from these figures that our analytic results match the minimum values of the window size and the current queue length, indicating a close agreement of our analytic results with simulation ones.

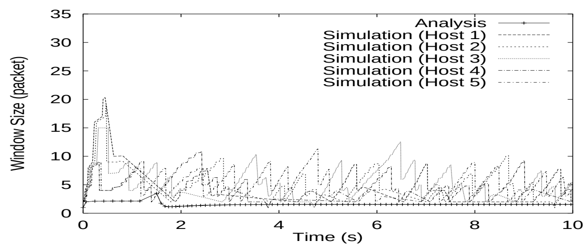


(a) Window size of TCP connection(s)

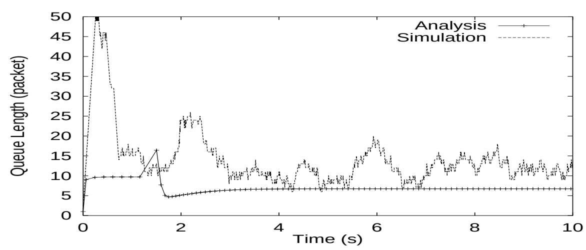


(b) Current queue length of RED gateway

Figure 4: Comparison between analytic results and simulation results ($B = 2$ [packet/ms], $\tau = 1$ [ms], $N = 1$).



(a) Window size of TCP connection(s)



(b) Current queue length of RED gateway

Figure 5: Comparison between analytic results and simulation results ($B = 2$ [packet/ms], $\tau = 1$ [ms], $N = 5$).

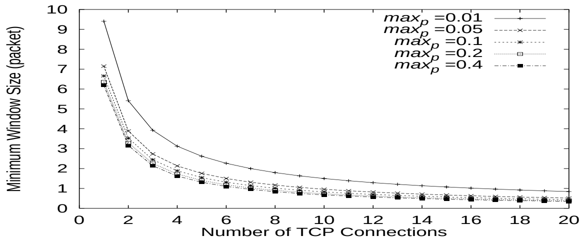


Figure 6: Average equilibrium value of the window size for different numbers of TCP connections ($B =$ 2 [packet/ms], $\tau = 1$ [ms]).

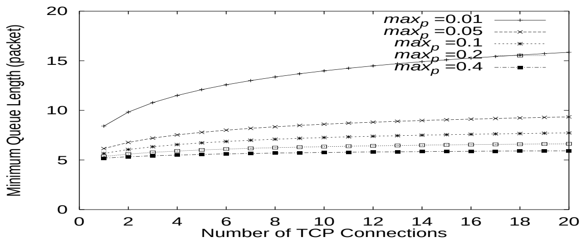


Figure 7: Average equilibrium value of the queue length for different numbers of TCP connections ($B =$ 2 [packet/ms], $\tau = 1$ [ms]).

5.2 Discussion on Average Equilibrium Values

Equations (16)–(18) indicate several interesting properties of the RED gateway. For instance, Eqs. (16) and (17) suggest that the window size of a source host and the queue length decrease as the number of TCP connections N or the maximum packet marking probability max_p increases. Such a tendency is clearly illustrated in Figs. 6 and 7, where the average equilibrium values of the window size w^* and the queue length q^* are plotted, respectively. Recall that w^* and q^* represent expected values of the minimum window size and the minimum queue length in steady state. We used the following network parameters: the processing speed of the RED gateway $B = 2$ [packet/ms] and the (round-trip) propagation delay $\tau = 1$ [ms]. The number of TCP connections N is varied between 1 and 20. The maximum packet marking probability max_p is also varied between 0.001 and 0.4, while other control parameters of the RED gateway are set to the values shown in Table 1. These figures indicate that the window size is heavily dependent on the number of TCP connections N, and that the queue length is mostly determined by the control parameter max_p.

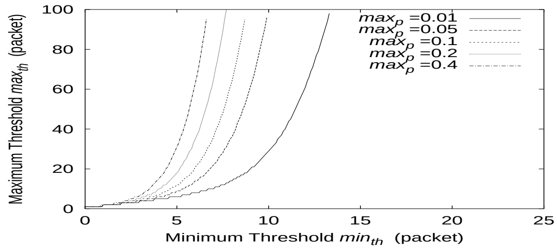


Figure 8: Stability region in the min_{th}–max_{th} plane ($B = 2$ [packet/ms], $N = 1$, $\tau = 1$ [ms]).

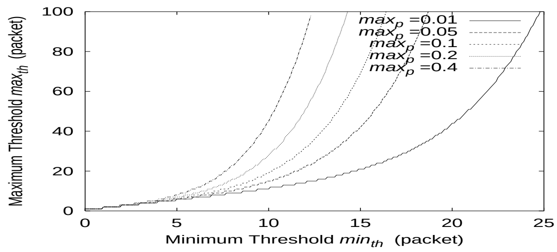


Figure 9: Stability region in the min_{th}–max_{th} plane ($B = 2$ [packet/ms], $N = 5$, $\tau = 1$ [ms]).

5.3 Discussion on Stability and Transient Behavior

Figures 8 and 9 show stability regions of the network in the max_{th}–min_{th} plane. In these figures, the number of TCP connections N is set to 1 or 5, respectively. Each line in the figures shows a boundary line of a stability conditions for different values of max_p. These figures indicate that the network is stable if the point (min_{th}, max_{th}) resides a left-side of the boundary line. For other control parameters, we used the same values used in Fig. 6: $B = 2$ [packet/ms] and $\tau = 1$ [ms]. One can find that the stability region becomes large as the maximum packet marking probability max_p decreases. This shows that the network can be stabilized with a large min_{th} if the packet marking probability max_p is set to a small value. Comparison between Figs. 8 and 9 suggests that the stability region becomes large as the number TCP connections N increases.

We next investigate how the stability region changes when the processing speed of the RED gateway B or the propagation delay of TCP connections τ is increased. Figure 10 is the case with a larger processing speed $B =$ 10 [packet/ms]. Figure 11 is the case with a larger propagation delay $\tau = 5$ [ms]. Other parameters are identical to those used in Fig. 8. By comparing Figs. 10 and 11, one can find that the stability regions in these figures are completely identical. This indicates that for system stability, the effect of increasing the processing speed B is same with the effect of increasing the propagation delay τ. This

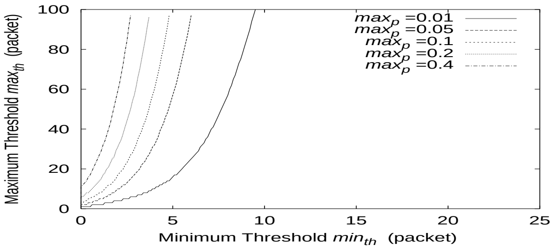


Figure 10: Stability region with a larger processing speed ($B = 10$ [packet/ms], $N = 1$, $\tau = 1$ [ms]).

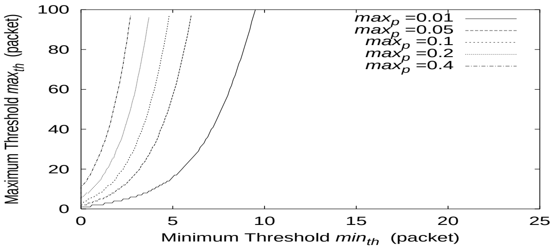


Figure 11: Stability region with a larger propgation delay ($B = 2$ [packet/ms], $N = 1$, $\tau = 5$ [ms]).

phenomenon can be explained from Eq. (13) where B and τ are shown in the product form of $B \times \tau$.

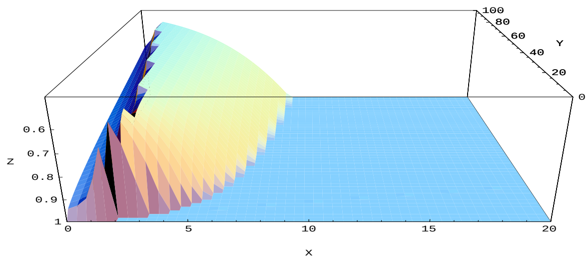


Figure 12: Relation among transient performance index and two threshold values ($B = 2$ [packet/ms], $N = 1$, $\tau = 1$ [ms], $max_p = 0.1$)

As explained in Section 4, stability and transient behavior of the network is determined by the roots of the characteristic equation given by Eq. (21). In Figs. 12 and 13, we plot the transient performance index s_{max} defined by Eq. (22) for different values of the maximum threshold max_{th} and the minimum threshold min_{th}. These figures correspond to the case of $max_p = 0.1$ in Figs. 8 and 9, respectively. Three axes represent the minimum threshold min_{th} (x), the maximum threshold max_{th} (y), and the transient performance index s_{max} (z). These figures indicates that the network is stable if s_{max} is less than 1, and that the transient performance is better (e.g., faster convergence) with a smaller s_{max}. One can find that the optimal

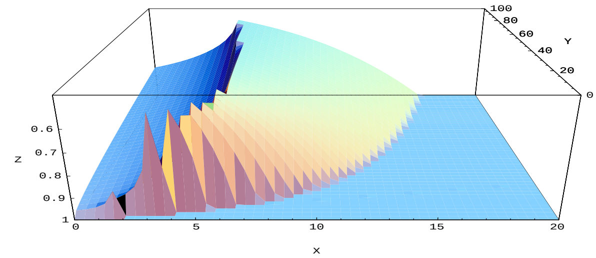


Figure 13: Relation among transient performance index and two threshould values ($B = 2$ [packet/ms], $N = 5$, $\tau = 1$ [ms], $max_p = 0.1$)

point (min_{th}, max_{th}) is almost independent of max_{th}. In other words, the transient performance is mostly determined by the minimum threshold min_{th}.

6 Discussion on Parameter Setting

The above discussion clearly depicts the fact that the steady state behavior of the RED gateway is significantly affected by a choice of control parameters. In what follows, based on findings from our steady state analysis, we discuss how control parameters of the RED gateway should be configured for achieving better performance. The RED gateway only knows two system parameters a priori: its processing speed B and its buffer capacity. For a proper configuration of control parameters, other system parameters such as the number of TCP connections N and the propagation delay of TCP connections τ should be estimated.

Step 1: *Estimate the number of TCP connections N.*

First, the number of TCP connections is estimated at the RED gateway. One approach for such an estimation is for the RED gateway to monitor the number of TCP flows during a certain period. It is possible to identify each TCP flow by analyzing the IP packet header since it contains source/destination IP addresses and port numbers. This approach does not always estimate the number of TCP connections correctly if the estimation period is not adequate. If the estimation period was too long, the RED gateway would overestimate the number of TCP connections since terminated TCP connections might be included. On the other hand, if the estimation period is too short, it would underestimate since it might fail to include slow TCP connections. We have found that the stability region becomes large as the number of TCP connections N increases in Section 5. Thus, a shorter estimation period would be appropriate for safer operation. In [9], another approach has been proposed to estimate the number of TCP connections. This approach is less complex since it does not require per-flow accounting.

Step 2: *Estimate the propagation delay of TCP connections τ.*

Second, a propagation delay of TCP connections τ is estimated at the RED gateway. In our analysis, propagation delays of all TCP connections are assumed to be identical. However, it is rarely the case in real networks; every TCP connection usually has a different propagation delay. A simple approach for applying our analytic results to real networks is to use the average propagation delay of all TCP connections as τ. Since we have found a tendency that the stability region becomes large as the propagation delay τ increases in Section 5. A smaller estimation of the propagation delay is therefore desirable for more stable operation. Similary, if the propagation delay cannot be estimated at all, a small value would be used.

Step 3: *Set the maximum threshold max_{th} to the buffer size.*

Third, the maximum threshold max_{th} is set to the possible largest value, i.e., the buffer size of the RED gateway. As explained in Section 2, the RED gateway discards all incoming packets when the average queue length exceeds the maximum threshold max_{th}. Therefore, in most cases, the maximum threshold max_{th} should not be set to less than the buffer size. If the requirement on an end-to-end transfer delay of TCP connections is tight, a small value of max_{th} would be appropriate to shorten the queuing delay at the RED gateway.

Step 4: *Determine max_p and min_{th} using our analysis.*

Finally, the maximum packet marking probability max_p and the minimum threshold min_{th} are determined from our steady state analysis. These control parameters have great importance on controlling the average queue length of the RED gateway as well as stability and transient behavior of the network. Therefore, these control parameters must be chosen very carefully using our analytic results in Sections 3 and 4. More specifically, the smaller value of max_p is desired for stability, as long as the average queue length given by Eq. (17) can be kept at a reasonable level. The minimum threshold min_{th} should be determined to optimize the transient performance. Namely, the minimum threshold min_{th} should be chosen to minimize the transient performance index given by Eq. (22).

7 Conclusion and Future Works

In this paper, we have analyzed the steady state behavior of the RED gateway when incoming traffic is flow-controlled by the congestion control mechanism of TCP. We have derived equilibrium values of TCP's window size and the buffer occupancy of the RED gateway. We have also derived the stability condition and the transient performance index for the network. Several numerical examples and simulation results have been presented to clearly understand relations between control parameters of the RED gateway and the steady state behavior. Of all our findings, most important results were: (1) max_p (maximum packet marking probability) mostly affects the RED's buffer occupancy, (2) the network becomes more stable as the number of TCP connections or the bandwidth–delay product increases, and (3) min_{th} (minimum threshold) is a key parameter for optimizing the transient performance. We finally have discussed a method for parameter tuning of the RED gateway. The outline of the parameter tuning method was: (1) estimate the number of TCP connections, (2) estimate the average propagation delay of all TCP connections, (3) set max_{th} (maximum threshold) to the buffer size, and (4) determine max_p and min_{th} using our analytic results to optimize the transient behavior.

As a future work, it would be of importance to extend our steady state analysis to relax strong assumptions. In particular, extending our steady state analysis to allow different propagation delays of TCP connections would give us much insight for better understanding of the RED gateway. More simulation experiments would be necessary to investigate the behavior of the RED gateway in complex network configurations.

References

[1] V. Jacobson, "Congestion avdoidance and control," in *Proceedings of SIGCOMM '88*, pp. 314–329, August 1988.

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

[3] E. Hashem, "Analysis of random drop for gateway congestion control," *Technical Report MIT-LCS-TR-465*, 1989.

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

[5] B. Barden et al., "Recommendations on queue management and congestion avoidance in the Internet," *Request for Comments (RFC) 2309*, April 1998.

[6] D. Lin and R. Morris, “Dynamics of random early detection,” in *Proceedings of ACM SIGCOMM ’97*, September 1997.

[7] M. May, J. Bolot, C. Diot, and B. Lyles, “Reasons not to deploy RED,” in *Proceedings of IWQoS ’99*, pp. 260–262, March 1999.

[8] W. chang Feng, D. D. Kandlur, D. Saha, and K. G. Shin, “A self-configuring RED gateway,” in *Proceedings of IEEE INFOCOM ’99*, March 1999.

[9] T. J. Ott, T. V. Lakshman, and L. Wong, “SRED: Stabilized RED,” in *Proceedings of IEEE INFOCOM ’99*, March 1999.

[10] M. May, T. Bonald, and J.-C. Bolot, “Analytic evaluation of RED performance,” in *Proceedings of IEEE INFOCOM 2000*, March 2000.

[11] V. Sharma, J. Virtamo, and P. Lassila, “Performance analysis of the random early detection algorithm,” available at `http://keskus.tct.hut.fi/tutkimus/com2/publ/redanalysis.ps`, September 1999.

[12] W. R. Stevens, “TCP slow start, congestion avoidance, fast retransmit, and fast recovery algorithms,” *Request for Comments (RFC) 2001*, January 1997.

[13] R. Isermann, *Digital control systems, Volume 1: fundamentails, deterministic control*. Springer-Verlag Berlin Heidelberg, 1989.

[14] “LBNL network simulator (ns).” available at `http://www-nrg.ee.lbl.gov/ns/`.