Analysis and Improvement of fairness among many TCP connections sharing Tail–Drop and RED Routers

Go Hasegawa, Kouichi Tokuda and Masayuki Murata

Cybermedia Center, Osaka University
1-3, Machikaneyama, Toyonaka, Osaka 560-8531, Japan
E-mail: {hasegawa, kouichit, murata}@ics.es.osaka-u.ac.jp

Abstract— **Appropriate control parameters are important for the successful deployment of RED (Random Early Detection) routers, especially when many TCP connections share the bottleneck link. In this paper, we first describe a new simple analysis method for determining the window size distribution of many TCP connections sharing a bottleneck router. We consider two kinds of buffering disciplines: TD (Tail Drop) and RED. We model the window size evolution of TCP connections by using a Markov process whose state is represented by a set of the current window size and the ssthreth value. The state transition matrix is then calculated by considering the characteristics of TD and RED routers. We show numerical results demonstrating the accuracy of our analysis and we discuss the fairness of TD and RED. We confirm that RED does not help improve the router's throughput even when appropriate control parameters are chosen but that it is still useful to provide the fairness among many competing TCP connections. We also describe an improved version of RED that adjusts its control parameters dynamically according to the changes in network congestion.**

Keywords— **TCP (Transmission Control Protocol), Window Size, TD (Tail Drop), RED (Random Early Detection), Fairness**

I. INTRODUCTION

The future development of the Internet requires a better understanding of the behavior of TCP (Transmission Control Protocol) [1] widely used in the current Internet, and many research efforts have been devoted to revealing the characteristics of the TCP connection. For example, equations for calculating the throughput of the TCP connection from several parameters (packet loss probability, round trip time, maximum window size, etc.) have been shown in [2], [3]. Those equations, however, estimate only the long-term throughput of the TCP connection, and cannot be used to examine instantaneous behavior. It is reported in [4] that the average size of Web documents at several Web servers is about 10 [KBytes], and the instantaneous TCP throughput is important to the estimation of the transfer time of such small documents. More important, the equations estimating throughput are based on the long–term averages of packet loss probabilities and RTT (Round Trip Time) values, which implies that the interaction among active TCP connections at the router cannot be investigated.

An approach for examining the instantaneous throughput of the TCP connection can be found in [5], [6], [7], where the distribution of the congestion window size of the TCP connection is obtained. Then the TCP window size is directly related to the short–term throughput of the TCP connection. This approach is based on a stochastic modeling of the TCP window size behavior. Those works, however, assume the probabilistic packet loss model, in which each packet sent from the TCP sender is dropped with a constant probability p. In a sense, it can be considered that RED (Random Early Detection) [8] is used at the bottleneck router. When this is assumed, however, it is impossible to examine the effects of misconfigured control parameters of the RED routers because the packets tend to be lost in bursts by such misconfigured RED.

Because we wanted to investigate the effect of the RED routers when many TCP connections share the bottleneck link, we developed a simple method for evaluating the TCP window size distribution by using Markov analysis. In the analysis, we allow two kinds of the packet loss models: the probabilistic loss model and the bursty loss model, which respectively correspond to RED routers and Tail-Drop (TD) routers. In our approach, the Markov model is used to explain the evolution of the TCP window size, which is done by explicitly considering packet queuing at the router's buffer. Steady state probabilities are then calculated and used to derive the distribution of TCP window size. One of our main contributions in this paper is the derivation of the TCP window size evolution in the case where packet loss occurs in a bursty fashion at RED routers. This allows to analyze the case in which the RED parameters are inappropriately configured to be considered. Such a case cannot be examined when using the analytic approaches assuming a constant packet loss probability adopted in, since the existing approaches [5], [6], [7] implicitly assume that RED routers always work ef-

fectively. Another contribution is the analysis of the window size behavior of TCP connections under conditions in which bursty packet losses occur at TD routers. The traditional TD routers tend to drop the incoming packets in a bursty fashion [8], and our approach allows to evaluate the effect of those routers. It then becomes possible to compare TD and RED routers in a unified way.

Our analysis allows us to investigate the routers shared by many active TCP connections. When the number of active TCP connections is large, as it is in the current backbone routers, the packet buffer that each TCP connection can utilize becomes small and throughput degradation becomes obvious [9]. Our analysis can treat such a case and determine the packet buffer size sufficient for, say, thousands of active TCP connections. Furthermore, we use the analysis results to evaluate the fairness of the TD and RED routers and that the fluctuation of the window size is much smaller for RED routers. We provide numerical examples showing that RED routers can only provide as high throughput as TD routers can even when the configuration parameters of RED are determined appropriately [10], [11] but that they can greatly improve the fairness among many TCP connections. Our analysis can be used to determine, for a given buffer size and given number of active TCP connections, the appropriate control parameter set for the RED routers.

The problem is that the appropriate parameter setting varies according to the network congestion status, and we could not find one parameter set to make the RED algorithm work effectively in all cases. We therefore propose *dt-RED*, by which we improved the RED algorithm by adding a simple mechanism that sets the threshold values dynamically. We evaluate the effectiveness of this improved algorithm through simulation and show that it can provide good fairness in various network congestion states by using one parameter set.

The rest of this paper is organized as follows. Section II briefly explains the TD and RED disciplines and introduces the analysis and simulation model. Section III presents our analysis of the window size distribution and some numerical results used to evaluate fairness. Section IV describes our improved RED algorithm, called dt-RED, and shows some simulation results demonstrating its effectiveness. Section V concludes the paper with a brief summary and some future research topics.

II. Model Description

A. TD and RED Routers

Historically, Internet routers have used a TD (Tail Drop) discipline as a buffer management mechanism: the TD router serves incoming packets in order of their arrival and simply discards newly arriving packets when the buffer is full. The problem with this mechanism is that routers tend to discard packets in bursts [12], which results in packets from the same connection being likely to be discarded. As a result, the fast retransmit algorithm does not help avoid timeout expirations, and this leads to the global synchronization problem [13]. Furthermore, since the duration of bursty packet losses depends on many factors (network configurations, the number of active connections, and so on), it is difficult to determine the packet loss rate of the TD router. As will be shown in Section III, however, we can analyze TD router if we make some reasonable assumptions about the network.

The RED (Random Early Detection) algorithm [8] is designed to cooperate with the congestion control mechanism of the TCP. It detects the beginning of congestion by monitoring the average queue size at the router (the average number of packets in the router buffer) and notifies TCP senders that congestion has occurred by intentionally dropping packets at a certain probability. The RED algorithm sets the packet dropping probability as a function of the average queue size. By keeping the average queue size low, burst packet dropping can be avoided even when packets from the same connection arrive continuously. That is, the algorithm has no bias against bursty traffic. More specifically, it uses a low–pass filter with an exponentially weighted moving average when calculating the average queue size avg, which is maintained and compared with two thresholds: a minimum threshold (min_{th}) and a maximum threshold (max_{th}). The packet dropping probability is determined in different ways according to the queue size avg:

- If $avg < min_{th}$, all arriving packets are accepted.
- If $min_{th} < avg < max_{th}$, arriving packets are dropped with probability $p_{\text{red}}(avg)$, which is defined as follows:

$$p_{\text{red}}(avg) = \frac{avg - max_{th}}{max_{th} - min_{th}} max_p \qquad (1)$$

- If $max_{th} < avg$, all arriving packets are dropped.

The RED router helps prevent TCP's retransmission timeouts, and most lost packets are thus retransmitted by the fast retransmit algorithm. It also helps avoid the phase effect [12] causing all connections to exhibit the similar window changes.

In recent works [10], [11], however, the authors have pointed out that it is difficult to choose the control parameters of RED ($max_{th}, min_{th}, max_p$) to work well, and even when those are appropriately configured, the RED routers cannot provide good performance compared with the TD routers. As opposed to these results, we present a new ob-

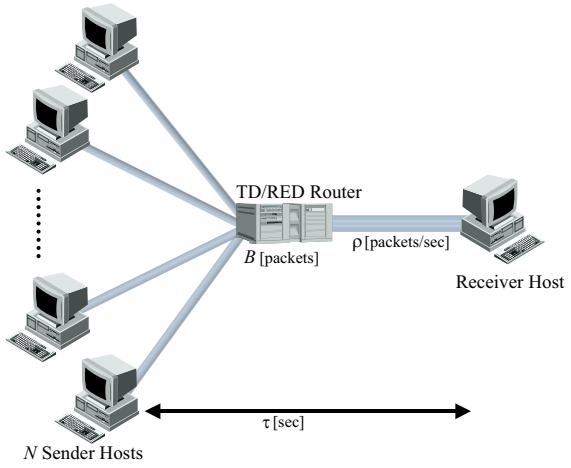


Fig. 1. Network model.

servation in this paper that RED is still useful, especially from a viewpoint of the fairness among many TCP connections.

B. Network Model

Fig. 1 depicts a network model used in the following analysis and simulation. It consists of sender hosts, a receiver host, a router, and links interconnecting the hosts and the router. N sender hosts share a bottleneck link of ρ [packets/sec], and sends data packets to the receiver host by TCP Reno. The propagation delay between the sender hosts and the receiver host is τ [msec]. The intermediate router has a buffer of the TD or RED discipline. The buffer size is represented by B [packets].

III. Fairness Analysis and Evaluation

A. Notations and Settings

We assume that a TCP connection changes its state at every RTT (Round Trip Time) and we call the interval between state changes a *round*. To describe the Markov process, we define the state of the TCP connection i by the window size w_i [packets] and a ssthreth value t_i [packets] of the current round. That is, (w_i, t_i) represents the state of the current round of connection i. By assuming that the maximum window size of the TCP connection is W_{max} [packets], w_i is ranged from 1 to W_{max}, and t_i is from 1 to $W_{max}/2$, since t_i is reset to half of w_i when packet loss is detected [1]. Then, the number of the states of this Markov process becomes $W_{max}^2/2$.

We define $P_{(w_i,t_i)(w_i',t_i')}$ as the state transition probability that the state of the connection i changes from (w_i, t_i) to (w_i', t_i'). The goal of the analysis in this section is to derive the state transition matrix of $P_{(w_i,t_i)(w_i',t_i')}$ with consideration of the congestion control algorithm of TCP and the characteristics of TD and RED routers. The TCP connection changes its state by increasing the window size when no packet loss occurs, or by decreasing the window size when packet loss is detected [1]. That is, we must consider the following cases about the state transitions from (w_i, t_i).

1. When no packet loss occurs:
   1.a During a slow start phase, the window size is increased and the state is changed to $(\max(2w_i, t_i), t_i)$.
   1.b During a congestion avoidance phase, the window size is increased and the state is changed to $(\max(w_i + 1, W_{max}), t_i)$.
2. When packet loss occurs:
   2.a If the packet loss is detected by timeout, the state is changed to $(1, \lfloor w_i/2 \rfloor)$.
   2.b If the packet loss is detected by fast retransmit, the state is changed to $(\lfloor w_i/2 \rfloor, \lfloor w_i/2 \rfloor)$.

To derive the state transition matrix, we have to obtain the probability that each case in the above takes place. The packet loss probability in each state is affected by the number of connections and the queue size. For a meanwhile, we assume that the packet loss probability of the state (w_i, t_i), which is denoted by $p(w_i, t_i)$, is known. $p(w_i, t_i)$ in the TD and RED disciplines are derived in the following subsections in turn.

The probability that no packet loss occurs in the current state (w_i, t_i) is obtained by collecting the probabilities that w_i packets of connections are not lost. It is given by

$$p_{\text{noloss}} = (1 - p(w_i, t_i))^{w_i} \quad (2)$$

Since the window size is simply increased when the no packet loss occurs, the cases 1.a and 1.b takes place with the following probabilities:

$$P_{(w_i,t_i)(\max(2w_i,t_i),t_i)} = p_{\text{noloss}}, \text{ if } w_i < t_i \quad (3)$$

$$P_{(w_i,t_i)(\max(w_i+1,W_{max}),t_i)} = p_{\text{noloss}}, \text{ if } w_i \geq t_i \quad (4)$$

When packet loss occurs, on the other hand, we must consider whether the lost packet is retransmitted by timeout or fast retransmit. We as denote $p_{(w_i,t_i),j,\text{TO}}$ as the probability that j packets are lost in state (w_i, t_i), and the timeout takes place to retransmit the lost packets. When one of w_i packets is lost and the window size is too small (i.e., smaller than three), the packet loss is detected by timeout [14]. That is, $p_{(w_i,t_i),j,\text{TO}}$ for j = 1 is given by

$$p_{(w_i,t_i),1,\text{TO}} = \begin{cases} \binom{w_i}{1} p(w_i, t_i)(1 - p(w_i, t_i))^{w_i - 1}, & \text{if } w_i \leq 3 \\ 0, & \text{if } w_i > 3 \end{cases}$$

When more than one packets are lost – specifically, when j of w_i packets are lost – the first lost packet is retransmitted by timeout or fast retransmit with probabilities

of $p_{(w_i,t_i),1,\text{TO}}$ or $1 - p_{(w_i,t_i),1,\text{TO}}$, respectively. When timeout occurs, all of lost packets are retransmitted and the window size becomes one. When fast retransmit occurs, however, the window size is halved. If the halved window size is larger than three, the next lost packet is again detected by the fast retransmit algorithm, and the window size is further halved. Otherwise, the timeout occurs. That is, all of the j packet losses can be detected by the fast retransmits if the window size is large enough to be kept larger than three when it is halved j times. We therefore have

$$p_{(w_i,t_i),j,\text{TO}} = \begin{cases} \binom{w_i}{j} p(w_i,t_i)^j (1-p(w_i,t_i))^{w_i-j}, & \text{if } w_i/2^j \le 3 \\ 0, & \text{if } w_i/2^j > 3 \end{cases}$$

From the above two equations, we can derive $p_{(w_i,t_i),\text{TO}}$, the probability that the timeout occurs in state (w_i, t_i), as:

$$p_{(w_i,t_i),\text{TO}} = \sum_{k=\lfloor \log_2(w_i/4) \rfloor + 1}^{w_i} p_{(w_i,t_i),k,\text{TO}}$$

Then, we can obtain the probabilities that cases of 2.a and 2.b take place:

$$\begin{aligned} & P_{(w_i,t_i)(1,\lfloor w_i/2 \rfloor)} = p_{(w_i,t_i),\text{TO}} \\ & P_{(w_i,t_i)(\lfloor w_i/2^j \rfloor, \lfloor w_i/2^j \rfloor)} \\ & \quad = \binom{w_i}{j} p(w_i,t_i)^j (1-p(w_i,t_i))^{w_i-j}, \\ & \qquad 1 \le j \le \lfloor \log_2(w_i/4) \rfloor \end{aligned} \tag{5}$$

We have now to determine, for the TD and RED disciplines, the value of $p(w_i, t_i)$ at the bottleneck router.

B. Analysis for TD router

In the TD router, packet loss occurs in bursts when the router buffer is fully occupied. To calculate $p(w_i, t_i)$ we need to take into account the following factors:

- the total number of lost packets as a result of buffer overflow
- the frequency of buffer overflow
- the number of lost packets from each TCP connection

B.1 How many packets are lost as a result of buffer overflow?

Here we denote by W the total window size of N TCP connections. A buffer overflows when W exceeds the sum the buffer size B and the bandwidth-delay product of the bottleneck link. That is, when the buffer is fully occupied, W reaches $W_f = 2\tau\rho + B$. Suppose that all of the TCP connections are in the congestion avoidance phase, W is increased by N [packets] in every RTT, since each TCP connection increases its window size by one packet [1]. Therefore, when the total window size reaches W_f, it will be increased to $(W_f + N)$ [packets] in the next round. That is, N [packets] are discarded at the TD router buffer when buffer overflow occurs.

B.2 How frequently do buffer overflows occur?

When buffer overflow occurs, some of N connections experience packet losses and decrease their window sizes. As shown in Subsection III-A, the TCP connection with window size w_i decreases its window size to $w_i/2^j$ when j packets belonging to the same connection are lost by the fast retransmit algorithm [1]. When no packet loss occurs, on the other hand, the window size increases by one. Assume that the mean window size of each connection is $\overline{w} = W_f/N$ when the buffer is fully occupied. Then, the mean window size of each connection after the buffer overflow occurs, denoted by $\overline{w'}$, is given by

$$\begin{aligned} \overline{w'} &= \sum_{j=1}^{\overline{w}} \binom{\overline{w}}{j} p_f^j (1-p_f)^{\overline{w}-j} \lfloor \frac{1}{2^j} \overline{w} \rfloor \\ &\quad + (1-p_f)^{\overline{w}} (\overline{w}+1) \end{aligned} \tag{6}$$

where $p_f = N/W_f$, which is the packet loss probability when the buffer overflow occurs. Note that the above equation includes the case where the TCP connection experiences the timeout, but in that case, $\lfloor \frac{1}{2^j} \overline{w} \rfloor$ is zero. Then, the mean of the total window size just after the buffer overflow, $\overline{W'}$, becomes $N \cdot \overline{w'}$. Since the total window size is increased by N packets per RTT, the probability that the buffer overflows occur in the current round (denoted by p_{overflow}) can be calculated as follows:

$$p_{\text{overflow}} = \frac{1}{N(W - N \cdot \overline{w'})} \tag{7}$$

B.3 How many packets are lost from each TCP connection?

We have shown in Subsection III-B.1 that the total number of lost packets in the event of buffer overflow is given by N. We assume that the number of lost packets belonging to each connection is proportional to the size of that connection's window. Then l_i, the number of lost packets of connection i, is given by

$$l_i = \frac{w_i}{W} \cdot N \tag{8}$$

B.4 Derivation of $p(w_i, t_i)$

Using Eqs. (7) and (8), we can obtain the following equation for $p(w_i, t_i)$:

$$p(w_i, t_i) = \min\left(1, p_{\text{overflow}} \cdot \frac{l_i}{w_i}\right) \tag{9}$$

C. Analysis for RED router case

In RED, the packet loss probability is determined from the average queue size and control parameters by Eq. (1). But because our analysis is based on the Markov process model, we use instead of Eq. (1) the following function for the packet discarding probability of RED. That is, we use the instantaneous queue length (q) instead of the average queue length (avg). This makes little difference with regard to the packet discarding probability of RED because q always fluctuates around avg.

$$p'_{\text{red}}(q) = \begin{cases} 0, & \text{if } q \le min_{th} \\ \dfrac{q - max_{th}}{max_{th} - min_{th}} max_p, & \text{if } min_{th} \le q \le max_{th} \\ \dfrac{q \cdot max_p + (q - max_{th})}{q}, & \text{if } q \ge max_{th} \end{cases}$$

The third form of this equation corresponds to the case where the queue length equals or exceeds max_{th} and all of incoming packets are dropped. Note that the previous work [5], [6], [7] did not consider this condition. Although the above equation is an approximation for the behavior of the RED router, in Section III-D it will be shown to a good estimate of the packet discarding probability of RED.

To calculate $p(w_i, t_i)$ for the RED router, we first derive the average packet loss probability of the RED router by using a steady–state analysis. Note that the appropriateness of using the average queue length of a RED router to derive the window size distribution of a TCP connection has been already validated in [5]. Here we use W^* and q^* to respectively represent the steady-state values of the total window size of N TCP connections and the queue size. i.e.,

$$W^* = 2\tau\rho + q^* \tag{10}$$

The mean window size of each connection becomes $\overline{w^*} = W^*/N$. Since RED discards the incoming packets with probability $p_{\text{red}}(q^*)$, the following equation should – from Eq. (6) and considerations similar to those that in Subsection III-B.2 – hold in the steady state:

$$W^* = N\left\{\sum_{j=1}^{\overline{w^*}} \binom{\overline{w^*}}{j} p'_{\text{red}}(q^*)^j (1 - p'_{\text{red}}(q^*))^{\overline{w}-j} \frac{1}{2^j} \overline{w^*} + (1 - p'_{\text{red}}(q^*))^{\overline{w^*}} (\overline{w^*} + 1)\right\} \tag{11}$$

We obtain q^* and W^* by solving Eqs. (10) and (11). If the window sizes of the other connections except connection i are equal to W^*/N, the current queue size (q) is given by

$$q = \frac{N-1}{N} W^* + w_i - 2\tau\rho$$

where w_i is the current window size of connection i. Then $p(w_i, t_i)$ is given by

$$p(w_i, t_i) = p'_{\text{red}}(q) \tag{12}$$

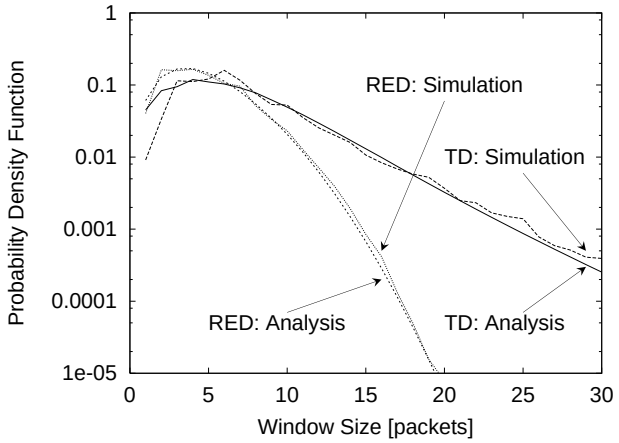
D. Numerical Results and Discussions

In this subsection, we examine the accuracy of our analysis approach by comparing its results with the results of simulation experiments, and we discuss the fairness of TD and RED routers. The simulation results were obtained by using ns-2 simulator [15].

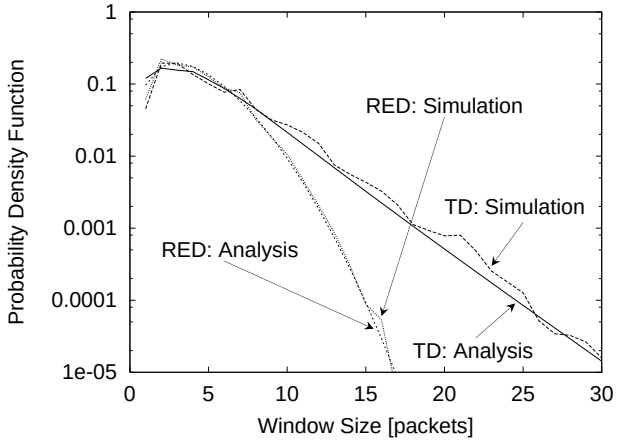
We set as network parameters $\rho = 187.5$ [packets/sec] ($\simeq 1.5$ [Mbps]), $\tau = 2$ [msec] and $N = 1,000$ (refer to Fig. 1). The analysis and simulation results for TD and RED router cases are shown in Fig. 2, where probability density is plotted as a function of the window size of the TCP connection. The buffer size B for TD and the max_{th} for RED are both set to 8,000 [packets] (Fig. 2(a)), to 5,000 [packets] (Fig. 2(b)), and to 3,000 [packets] (Fig. 2(c)). These results show that the analysis gives a good estimate of the window size distribution regardless of the buffer size. Even for a buffer size of 3,000 [packets], which is too small for 1,000 TCP connections in the TD router (only three for each connection) and many buffer overflows take place at the router buffer[9], the analysis results are very close to the simulation results and our analysis well follows the behavior of TD router.

Furthermore, comparing the results of TD and RED routers in Fig. 2, we can make the following observations about their fairness. When the TD's buffer size is equal to the RED's max_{th}, the window size varies more widely for TCP connections at the TD router. This can be understood more clearly by looking at Fig. 3, which shows the analysis results of 99.99% and 99.9999% values of the window size distribution. We can see that TCP connections at the TD router tend to have much larger window sizes than those at the RED router. That is, the RED router results in a smaller dispersion of the window sizes, which implies that the RED router improves fairness among many TCP connections.

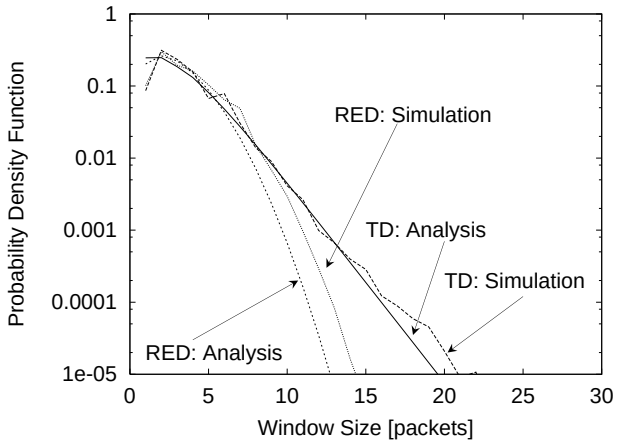
When the max_{th} of the RED router is set too small, however, the fairness of the RED router is degraded. This can be seen in Fig. 2(c), where the probability of a very small window size (< 5) is much higher than in Figs. 2(a) and 2(b). When max_{th} is too small, RED's probabilistic packet dropping does not work well, and the incoming



(a) $B = max_{th} = 8,000$



(b) $B = max_{th} = 5,000$



(c) $B = max_{th} = 3,000$

Fig. 2. Analysis and simulation results.

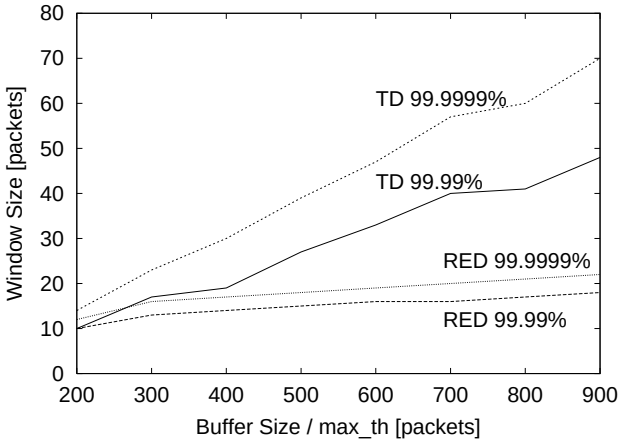


Fig. 3. 99.99/99.9999 percentile values of the window size.

TABLE I
EFFECT OF max_{th} ON THE THROUGHPUT AND FAIRNESS.

Router Discipline	Throughput	Fairness
TD	1.38 [Mbps]	0.80
RED ($max_{th} = 8,000$)	1.41 [Mbps]	0.94
RED ($max_{th} = 5,000$)	1.41 [Mbps]	0.95
RED ($max_{th} = 3,000$)	1.42 [Mbps]	0.81

packets are discarded in bursts. This situation corresponds to the third form of Eq. (10): when the queue length exceeds max_{th} all incoming packets are dropped.

The total throughput of the bottleneck link and the fairness values through simulation experiments are shown in Table I. We use a *fairness index* defined in [16] to evaluate fairness. When that the throughput of the TCP connection i is denoted x_i and the number of TCP connections is denoted n, the fairness index f is defined as follows:

$$f = \frac{(\sum_{i=1}^{n} x = i)^2}{n \sum_{i=1}^{n} x_i^2} \quad (13)$$

Note that the value of f ranges from 0 to 1, and a larger value shows a better fairness.

Table I clearly shows that while the throughput of RED remains almost same as that of TD regardless of the max_{th} value, RED can provide better fairness than TD if max_{th} is set appropriately. However, it does not help improve the fairness when a too small value of max_{th} is used. Therefore, max_{th} for the RED router should be large enough for the number of connections accommodated at the router. In [10], [11], the authors have concluded that there is little reason to deploy RED to Internet routers since RED routers cannot provide not so much throughput improvement of the bottleneck router and that it is sometimes lower than that of TD routers. We make the opposite conclusion.

RED is still valuable to apply to the router since it can improve the fairness among connections under the condition that max_{th} is appropriately chosen.

IV. DT-RED

A. Motivation

In the previous section, we have concluded that a RED router is fairer than a TD router although both kinds of routers show almost the same throughput. We have also pointed out that a RED provide better fairness only when the control parameters, especially the threshold values, are set appropriately. Specifically, its packet-discarding algorithm works well when the average queue length is between max_{th} and min_{th}. Therefore, if RED router is to be effective, its control parameters must be set to values appropriate to the condition of the network (link bandwidth, the number of active connections, congestion level, and so on). As pointed out by other researchers, however, it is very difficult to set them to cope with the dynamically changing network condition [10], [11].

Several algorithms that dynamically change the control parameters according to queue length, the number of active connections, and so on have therefore been proposed [17], [18], [19]. They focus on the dynamic control of the packet discarding probability p, but a rapid change of p degrades the performance of TCP connections, since the throughput of the TCP connection is directly affected by the packet loss ratio [3]. More important, the previous algorithms do not consider the case where many TCP connections share the bottleneck link. The level of congestion at a bottleneck router that has more than 1,000 connections active at the same time varies widely because each connection reacts to the network congestion independently. Therefore, when the above algorithms are applied, p is likely to fluctuate over a wide range, making the router unstable. Those algorithms thus do not resolve the problems pointed out in [10], [11], and p should be changed as smoothly as possible.

We therefore describe an improved algorithm we call *dt-RED*. It is very simple compared with the existing approaches but works well, especially with regard to fairness among connections. It does not change p directly, but instead controls the threshold values (max_{th} and min_{th}) to change p more smoothly.

B. Algorithm

To set the threshold values (max_{th} and min_{th}) appropriately according to the network condition, we developed an algorithm that monitors avg and controls the threshold values dynamically according to the change of avg, so that avg is always between max_{th} and min_{th}. The threshold values are tuned at a regular interval, which in this paper is called an "observation interval." The router monitors avg during every observation interval and sets the threshold values at the end of the interval.

- **Control of** max_{th}

When avg reaches $M_{max} \cdot max_{th}$ more than once during the observation interval, max_{th} is updated as follows:

$$max_{th} \leftarrow \min(max_{th} + \alpha_{max} \cdot B, M_B \cdot B) \qquad (14)$$

where M_{max} is a margin ratio for avoiding avg from reaching max_{th}, and M_B is a control parameter that determines the maximum values of max_{th}.
Otherwise, when avg is enough low and satisfies

$$avg - min_{th} < M_{max} \cdot (max_{th} - min_{th}), \qquad (15)$$

max_{th} is decreased as follows:

$$max_{th} \leftarrow \max(max_{th} - \beta_{max} \cdot (max_{th} - min_{th}), min_{th}) \qquad (16)$$

Note that α_{max} and β_{max} respectively determine the increase and decrease ratios of max_{th}.

- **Control of** min_{th}

When avg decreases to less than min_{th} during the observation interval, dt-RED changes min_{th} as follows:

$$min_{th} \leftarrow \max(min_{th} + \alpha_{min} \cdot (max_{th} - min_{th}), max_{th}) \qquad (17)$$

Conversely, when avg is always larger than min_{th}, min_{th} is decreased in order to shorten the queue length at the router buffer:

$$min_{th} \leftarrow \max(min_{th} - \beta_{min} \cdot (max_{th} - min_{th}), 0) \qquad (18)$$

where α_{min} and β_{min} are respectively the control parameters determining the increase and decrease ratios of min_{th}.

The key issue in dt-RED is how to set M_{max}. That is, when is max_{th} increased or decreased according to the change of avg? It should be set appropriately by considering the fluctuation of avg because the change of avg is greatly affected by the number of active TCP connections passing through the dt-RED router. Therefore, the value of M_{max} should be related to the observation interval. Suppose that the observation interval is set to the time during which the router receives N_p packets, where N_p is a fixed value. Then, M_{max} should also be fixed at a certain value since the number of packets arriving at the router during the observation interval is constant. When the observation

interval is fixed, on the other hand, M_{max} should become proportional to the number of connections since the number of arriving packets is proportional to the number of connections. For simplicity, we set the observation interval to be the time during which 100 packets arrive at the router, and we also set M_{max} to 0.9 in the following simulation results. When we want to use the fixed duration of the observation interval and set M_{max} proportional to the number of connections, we have to estimate the number of active TCP connections at the router. One way to do that might be by using a *Zombie List* [20], where the number of active connections can be estimated in $O(1)$ complexity, but we need further research to assess the applicability of the Zombie List.

Dt-RED uses six new control parameters to tune the two parameters of RED, max_{th} and min_{th} dynamically. Dt-RED differs from RED, however, in that the parameter setting of dt-RED has little effect on the fairness of dt-RED. We use only *one* parameter set to deal with various network configurations in dt-RED.

C. Evaluation Results

In this subsection, we show several simulation results demonstrating the fairness of dt-RED. We again use the network model shown in Fig. 1 and we set the bandwidth and the propagation delay of the link between the sender hosts and the router to 100 [Mbps] and 2 [msec]. We compare three kinds of buffering disciplines at the bottleneck router; Tail-Drop (TD), RED, and dt-RED. For RED, we use $min_{th} = 5$ and $max_p = 0.1$, $w_q = 0.002$, which are the values recommended in [8], and we vary max_{th} to find the best setting of RED. For dt-RED, we use $N_p = 100$ [packets], $M_{max} = 0.9$, $M_B = 0.8$, $\alpha_{max} = \alpha_{min} = \beta_{max} = \beta_{min} = 0.05$.

In the simulations, each sender host sends an infinite size of the document by FTP (File Transfer Protocol). Note that we have also examined the case where each sender host transfers WWW documents, the size distribution of which follows the one shown in [21]. Due to space limitation, we omit the results, but we confirmed the same results as shown in the below. Also note that we have evaluated dt-RED when we use the fixed duration of an observation interval and set M_{max} proportional to the number of connections, and we have also obtained the same results as what follows.

We first show the dynamical change of the threshold values of dt-RED, according to the change of the network congestion status. We set $\rho = 1.5$ [Mbps], $\tau = 4$ [msec], and $B = 10,000$ [packets], and change the number of TCP connections, N, as follows; $N = 500$ from 0 [sec] to 200 [sec] of the simulation time, $N = 1,000$ from 200 [sec] to 300 [sec], $N = 500$ from 300 [sec] to 400 [sec], and $N = 10$ from 400 [sec] to 500 [sec]. The changes of the instantaneous queue length, the average queue length (avg) and max_{th} are shown in Fig. 4 as functions of the simulation time. We can see that dt-RED changes max_{th} as the queue length at the router buffer increases and decreases.

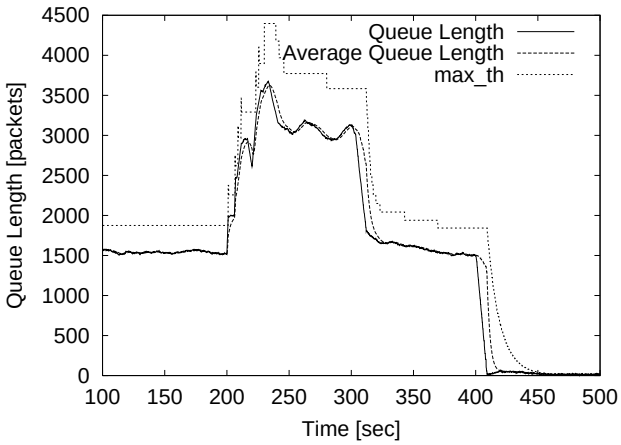
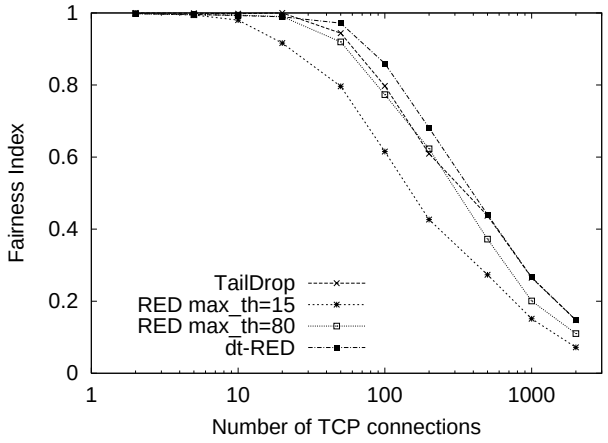


Fig. 4. Dynamical change of max_{th} in dt-RED.

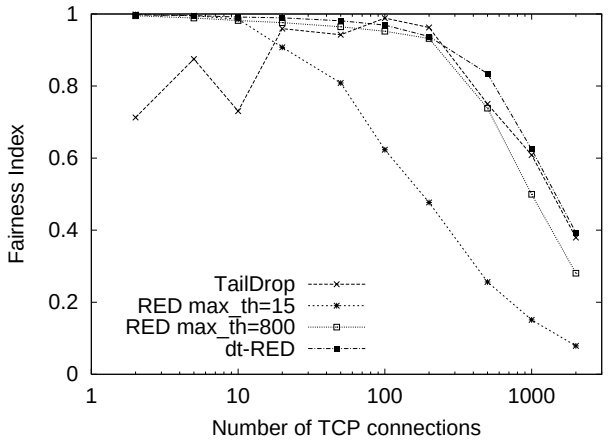
We next show the fairness aspect of the three mechanisms (TD, RED and dt-RED). We first set $\rho = 1.5$ [Mbps] and $\tau = 4$ [msec], and set B to 100, 1,000, and 10,000 [packets]. The results are shown in Fig. 5, where the fairness index values of the three mechanisms are plotted against the number of TCP connections. For RED, we show two cases: one with $max_{th} = 15$ (the value recommended in [8]), and the other with the best value that we found through trial-and-error. From these results, we can make several observations.

First, the TD discipline cannot provide fairness among connections at all, especially when the number of connections is large, because the TD discipline has no mechanism for fairness enhancement. Furthermore, the fairness of TD is also degraded when the buffer size is too large compared with the number of connections (Fig. 5(c)). This shows another problem of the TD discipline: when the buffer size per each TCP connection is too large, the buffer occupation becomes different among connections, which leads to differences of the throughput values of TCP connections. That is, larger buffers at TD routers do not always assure fairness among connections.

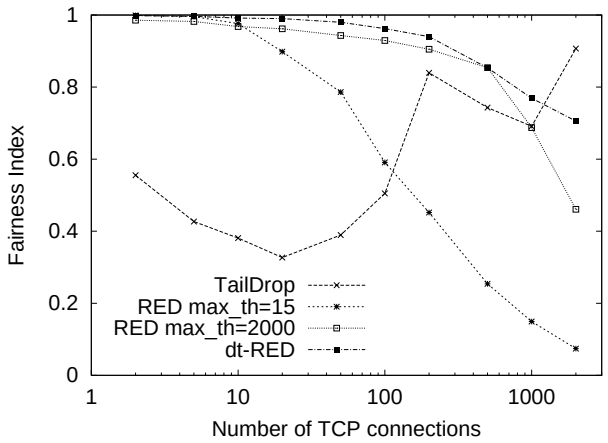
For the RED discipline, we can see that the recommended values of the control parameters cannot provide the good fairness. In the worst case, RED shows a lower fairness than TD does. The main reason is that the max_{th} in this case is too small for the number of connections and the buffer size. When max_{th} is set appropriately, on the other hand, RED provides better fairness than TD in



(a) B = 100 [packets]



(b) B = 1,000 [packets]



(c) B = 10,000 [packets]

Fig. 5. Fairness evaluation result for ρ = 1.5 [Mbps] and τ = 4 [msec].

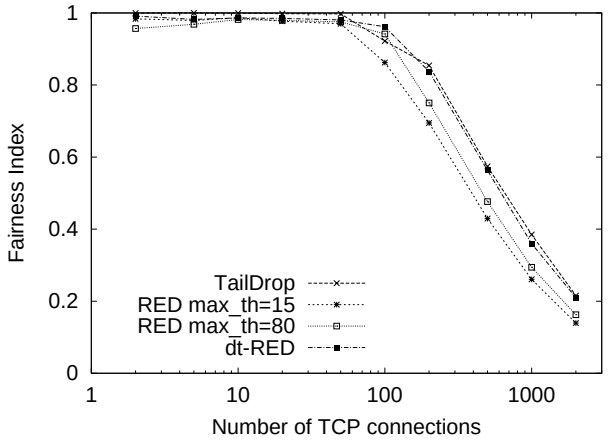
all cases. That is, RED can work better than TD if the control parameters are properly configured according to the change of the network congestion status. Of course, as shown in Section III, it is very difficult to find the best value of max_{th} (max_{th} = 80 for Fig. 5(a), 800 for Fig. 5(b), and 2,000 for Fig. 5(c)). An appropriate value cannot be found without many trials.

The fairness index values of dt-RED are almost same as the ones obtained by RED with the best parameter setting. This is a very important characteristics of dt-RED because dt-RED can provide this fairness by using a *single parameter set*; we need not to tune the control parameters of dt-RED according to the change of the network congestion status.

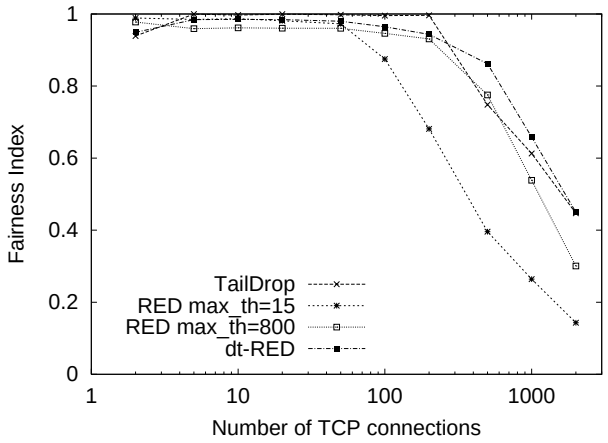
We next change the propagation delay of the bottleneck link, τ, from 4 [msec] to 400 [msec]. The simulation results are shown in Fig. 6, where we can again see that the RED algorithm with the control parameter values recommended in [8] cannot provide the fairness that the TD router can, especially when the number of connections is large. The fundamental reason is that the recommended values are not based on a situation in which there are more than 100 connections at the bottleneck router. The appropriate control parameters set for a given number of connections could of course be found, but only by trial-end error. dt-RED, on the other hand, shows very good fairness regardless of the number of connections and the size of the router buffer. It provides fairness almost same or even better than that provided by RED with the best parameter set. One problem is that when $B = 10,000$ [packets] (Fig. 6(c)) and the number of connections is large, TD is the fairest of the three disciplines. In this case, TD results in no packet loss at the buffer. Since RED and dt-RED routers intentionally discard incoming packets when only a few packets are stored at the router buffer, they are not as fair as a TD router. For further fairness improvement of dt-RED, we should treat this case more effectively, which is left to be a future work.

V. Conclusion

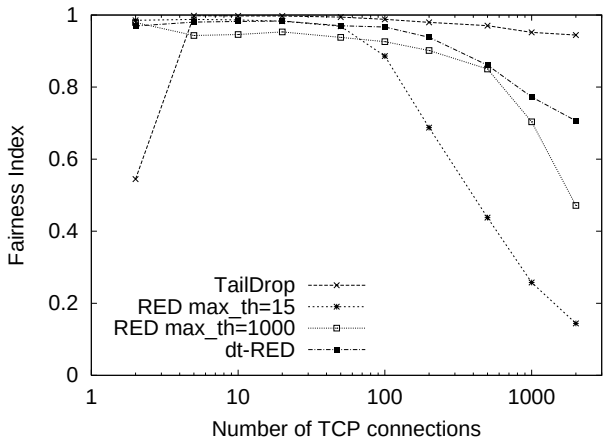
This paper has described a new analysis method using the Markov process model to determine the window size distribution of many TCP connections sharing the bottleneck router. We considered the RED router and the TD router, which drop incoming packets probabilistically and in bursts, respectively. Through numerical examples, we have shown that our analysis can give good estimates of the window size distributions at TD and RED routers. We have also discussed the fairness of TD and RED routers by using the analysis results and have concluded that RED is very effective in improving the fairness among many TCP



(a) B = 100 [packets]



(b) B = 1,000 [packets]



(c) B = 10,000 [packets]

Fig. 6. Fairness evaluation result for ρ = 1.5 [Mbps] and τ = 400 [msec].

connections. We have then described the dt-RED algorithm, which changes the RED's threshold values (max_{th} and min_{th}) dynamically according to the network congestion status. Dt-RED adds a very simple mechanism to the original RED but it can provide greatly improved fairness among many TCP connections without any parameter tunings.

In the past literature on stochastic/Markov modeling of TCP behavior, as well as in this paper, the homogeneous network model, where all TCP connections have the same propagation delays, has been used. The fundamental characteristics of the TCP has been revealed through these studies, but in the future we will extend the analysis in this paper and treat the network model in which each TCP connection has a different propagation delay and bandwidth. This will make clear the effect of the propagation delay on the window size distribution of each TCP connection. We also plan to use the analysis in this paper to estimate the throughput of TCP connections.

ACKNOWLEDGEMENTS

This work was partly supported by the Research for the Future Program of the Japan Society for the Promotion of Science under the Project "Integrated Network Architecture for Advanced Multimedia Application Systems," by a Grant-in-Aid for Scientific Research (B) 13450158 from the Japanese Ministry of Education, Culture, Sports and Science and Technology of Japan, and by the "Research on High-performance WWW server for the Next-Generation Internet" program of from the Telecommunications Advancement Foundation.

REFERENCES

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

[2] Matthew Mathis, Jeffrey Semke, Jamshid Mahdavi, and Teunis Ott, "The macroscopic behavior of the TCP congestion avoidance algorithm," *ACM SIGCOMM Computer Communication Review*, vol. 27, pp. 67–82, July 1997.

[3] Jitendra Padhye, Victor Firoiu, Don Towsley, and Jim Kurose, "Modeling TCP throughput: a simple model and its empirical validation," in *Proceedings of ACM SIGCOMM'98*, Aug. 1998, pp. 303–314.

[4] Masahiko Nabe, Masayuki Murata, and Hideo Miyahara, "Analysis and modeling of World Wide Web traffic for capacity dimensioning of Internet access lines," *Performance Evaluation*, vol. 34, no. 4, pp. 249–271, Dec. 1999.

[5] Archan Misra, Teunis Ott, and John Baras, "The window distribution of multiple TCPs with random loss queues," in *Proceedings of IEEE GLOBECOM '99*, Nov. 1999.

[6] Jitendra Padhye, Victor Firoiu, and Don Towsley, "A stochastic model for TCP reno congestion avoidance and control," Tech. Rep. 99-02, CMPSCI Technical Report, Oct. 1999.

[7] Claudio Casetti and Michela Meo, "A new approach to model the

stationary behavior of TCP connections," in *Proceedings of IEEE INFOCOM 2000*, Mar. 2000.

[8] Sally Floyd and Van Jacobson, "Random early detection gateways for congestion avoidance," *IEEE/ACM Transactions on Networking*, vol. 1, no. 4, pp. 397–413, Aug. 1993.

[9] Robert Morris, "TCP behavior with many flows," in *Proceedings of IEEE International Conference on Network Protocols*, October 1997.

[10] Martin May, Jean Bolot, Christophe Diot, and Bryan Lyles, "Reasons not to deploy RED," in *Proceedings of IWQoS'99*, June 1999.

[11] Mikkel Christiansen, Kevin Jeffay, David Ott, F. Donelson Smith, "Tuning RED for web traffic," in *Proceedings of ACM SIGCOMM 2000*, August 2000.

[12] Sally Floyd and Van Jacobson, "On traffic phase effects in packet-switched gateways," *Internetworking: Research and Experience*, vol. 3, no. 3, pp. 397–413, Aug. 1992.

[13] Lixia Zhang and David D. Clark, "Oscillating behavior of network traffic: A case study simulation," *Internetwoking: Research and Experience*, vol. 1, pp. 101–112, 1990.

[14] Kevin Fall and Sally Floyd, "Simulation-based comparisons of Tahoe, Reno, and SACK TCP," *ACM SIGCOMM Computer Communication Review*, vol. 26, no. 2, pp. 5–21, July 1996.

[15] "Network simulator - `ns` (version 2).," available from `http://www.isi.edu/nsnam/ns/`.

[16] Raj Jain, "Throughput fairness index: An explanation," *ATM Forum Contribution 99-0045*, February 1999.

[17] Haining Wang and Kang G. Shin, "Refined design of random early detection gateways," in *Proceedings of Globecom'99*, December 1999, pp. 769–775.

[18] Wu-chang Feng and Dilip D.Kandlur and Debanjan Saha and Kang G. Shin, "A self-configuring RED gateway," in *Proceedings of IEEE INFOCOM'99*, March 1999.

[19] Wu-chang Feng and Dilip D.Kandlur and Debanjan Saha and Kang G. Shin, "BLUE: A new class of active queue management algorithms," Tech. Rep. CSE-TR-387-99, U. Michigan, March 1999.

[20] Teunis J. Ott, T. V. Lakshman, and Larry Wong, "SRED: Stabilized RED," in *Proceedings of IEEE INFOCOM'99*, Mar. 1999.

[21] Paul Barford and Mark Crovella, "Generating representative Web workloads for network and server performance evaluation," in *Proceedings of ACM SIGMETRICS '98*, 1998.