Performance Improvement on RED Based Gateway in TCP Communication Network

Sumet Prabhavat* and Ruttikorn Varakulsiripunth **

Faculty of Information Technology* and Faculty of Engineering**
and Research Center for Communications and Information Technology (ReCCIT)*,**
King Mongkut's Institute of Technology Ladkrabang, Bangkok 10520, Thailand
sumet@it.kmitl.ac.th*, kvruttik@kmitl.ac.th**

Abstract: Internet Engineering Task Force (IETF) has been considering the deployment of the Random Early Detection (RED) in order to avoid the increasing of packet loss rates which caused by an exponential increase in network traffic and buffer overflow. Although RED mechanism can prevent buffer overflow and hence reduce an average values of packet loss rates, but this technique is ineffective in preventing the consecutive drop in the high traffic condition. Moreover, it increases a probability and average number of consecutive dropped packet in the low traffic condition (named as "uncritical condition"). RED mechanism effects to TCP congestion control that build up the consecutive of the unnecessary transmission rate reducing; lead to low utilization on the link and consequently degrade the network performance. To overcome these problems, we have proposed a new mechanism, named as Extended Drop slope RED (ExRED) mechanism, by modifying the traditional RED. The numerical and simulation results show that our proposed mechanism reduces a drop probability in the uncritical condition.

Keywords: Active queue management, Buffer queue management, RED, ExRED

1. INTRODUCTION

Recently, Internet Engineering Task Force (IETF) has been considering the deployment of the Random Early Detection (RED) [5] in order to avoid the increasing of packet loss rates caused by an exponential increment of network traffic and buffer overflow. RED is an active queue management technique [6] that can prevent buffer overflow and reduce an average of packet loss rates. But this technique is not only ineffective in preventing high rate of consecutive drop but also increases a probability and mean number of consecutive dropped packets [11]. The consecutive drop will cause consecutive loss and low utilization of a link and it consequently degrades link performance due to TCP congestion control.

Most commonly used TCP congestion control, i.e. TCP Reno / New Reno, assume that every packet loss is an indication of network congestion. This interaction between the error recovery and the congestion control procedures results in a low utilization of the link. In this paper, we have proposed new approach for modifying RED, in order to overcome the above-mentioned problems.

In section 2, we will describe a TCP congestion control to describe how the consecutive drops cause the low utilization on a link. A tradition queue management, i.e. Tail Drop and RED mechanism, will be described in the section 3. And we have shown the analytical and simulation results to compare a probability and a mean number of consecutive dropped packets between Tail Drop and RED. Our simulation results illustrates that RED reduces a packet loss rates by increasing a packet drop probability and a mean number of consecutive dropped packets to prevent buffer overflow. However, an increasing of average queue size results a higher increasing rate of mean number of consecutive dropped packets. It is an inefficiency of RED in preventing the consecutive drops. With the consecutive drops, TCP congestion control mechanism will assume that network congestion occurs and consequently TCP link utilization is reduced. It could be more critical and leads to the TCP global synchronization problem [11]. In section 4, we propose our algorithm to overcome the problems. Our key of the solution is the modification of drop probability characteristic line. The 2nd order polynomial function is applied to the drop probability characteristic line. We assume that the problem occurs because of inefficiently and unsuitable queue management. If we have a more applicable queue management, buffer will be more efficiently utilized and the consecutive drops will be decreased. We call our proposed algorithm as the Extended drop slope RED (ExRED) mechanism. Finally, we have compared ExRED to the prior mechanism to show that it can imporve the performance. The result of our research will be shown in the last section of the paper include the conclusion. We also present that our proposed mechanism improve throughput while keeps an advantage of RED mechanism in the issue of queuing delay.

2. TCP CONGESTION CONTROL

TCP was first introduced in early 1980s to provide reliable operation over a variety of transmission media. The efficiency of the TCP has been improved through a series of TCP reference implementations (e.g. Tahoe, Reno), which have refined TCP behavior. The congestion control (slow start [2]) and congestion avoidance (multiplicative decrease [2]) techniques introduced from Tahoe TCP use the principle of self-clocking.

On detection of congestion, slow start procedure implemented in Tahoe TCP 'drains the pipe' before transmission of more data [2]. This achieves network stability but is also unduly conservative. The fast recovery algorithm [3, 4] has therefore been introduced by the Reno TCP

implementation to drain only a half of the pipe and then recommence transmission, assuming the reception of each duplicate ACK is an indication of a packet leaving the network. Although most of the operating systems released in early 1990s have implemented Tahoe, the current implementations of TCP are based on the Reno reference implementation or have the same functions of Reno.

TCP congestion control mechanism assumes that every packet loss is an indication of network congestion and takes measures to avoid further congestion in the network by reducing the transmission rate. These also illustrate that a single loss of packet effect to transmission rate. Moreover, the consecutive packet losses result a very low utilization of the TCP link due to buffer overflow or buffer control mechanism action.

3. BUFFER QUEUE MANAGEMENT

3.1 Tail-Drop

An arrival packet will be allowed to accommodate the buffer queue only if a space of buffer is available. We can illustrate as the following:

$$q \leftarrow \begin{cases} q+1 & ; \quad q < BufferSize \\ q & ; \quad otherwise \end{cases} \tag{1}$$

Where q is buffer queue size and the BufferSize is a space in buffer. In our simulation, we assume that arrivals occur randomly according to a Poisson Process with rate λ and service times of single-server with buffer size K is exponentially distributed at rate μ. Thus, performance model of Tail Drop mechanism can be estimated by using M/M/1/K queuing model. Equation (2) expresses drop probability.

$$P_{drop} = \begin{cases} \dfrac{\left(1-\dfrac{\lambda}{\mu}\right)\left(\dfrac{\lambda}{\mu}\right)^{K}}{1-\left(\dfrac{\lambda}{\mu}\right)^{K+1}} & ;\lambda \neq \mu \\ \dfrac{1}{K+1} & ;\lambda = \mu \end{cases} \tag{2}$$

Equation (3) and (4) express consecutive drop probability and average number of consecutive drop, respectively.

$$P_{drop}^{cons}(N > n) = \left(\frac{\dfrac{\lambda}{\mu}}{1+\dfrac{\lambda}{\mu}}\right)^{n} \tag{3}$$

$$E(\mathrm{N}) = 1 + \frac{\lambda}{\mu} \tag{4}$$

Equation (5) expresses queuing delay.

$$D = \frac{E(Q)}{(1-P_{drop})\lambda} - \frac{1}{\mu}, \tag{5}$$

where $E(Q)$ is mean of queue size and given by (6).

$$E(Q) = \begin{cases} \dfrac{\dfrac{\lambda}{\mu}\left(1-(K+1)\left(\dfrac{\lambda}{\mu}\right)^{K}+K\left(\dfrac{\lambda}{\mu}\right)^{K+1}\right)}{\left(1-\left(\dfrac{\lambda}{\mu}\right)^{K+1}\right)\left(1-\dfrac{\lambda}{\mu}\right)} & ;\lambda \neq \mu \\ \dfrac{K}{2} & \lambda = \mu \end{cases} \tag{6}$$

3.2 RED Mechanism

RED [5] itself consists of two main parts, i.e., the estimation of an average queue size and the decision of whether or not to drop an incoming packet. An average queue size is calculated by a current queue size (or instantaneous queue size) using an exponentially weighted moving average (EWMA) [5] as shown below.

$$\hat{q} \leftarrow \begin{cases} (1-w)\hat{q} + wq & ; \quad q \neq 0 \\ (1-w)^{\frac{\mu}{\lambda}}\hat{q} & ; \quad otherwise \end{cases} \tag{7}$$

where w is an EWMA parameter which is a small constant value and defined by [5] as queue weight.

In the 2nd portion of RED algorithm, RED decides whether or not to drop an incoming packet. It is RED's particular algorithm for dropping that results in performance improvement for responsive flows. There are two thresholds figure prominently in this decision process. Minimum threshold k_l specifies the average queue size below which no packet will be dropped and maximum threshold k_h specifies the average queue size above which all packets will be dropped. When the average queue size varies from minimum to maximum threshold level, the packets will be dropped with probability that vary linearly from 0 to $\max_p$ where $\max_p$ is the maximum drop probability parameter. Then, the packet drop probability distribution function is defined as shown below [5]. The suitable value for parameters in this function were discussed in [5] and [11].

$$p_d(\hat{q}) = \begin{cases} 0 & ; \quad \hat{q} < k_l \\ \dfrac{\hat{q}-k_l}{k_h-k_l} \times \max_p & ; \quad k_l \le \hat{q} < k_h \\ 1 & ; \quad \hat{q} \ge k_h \end{cases} \tag{8}$$

By using the PASTA property [9] with arrival rate λ and service rate μ, RED performance can be presented in average packet drop probability, consecutive drop probability, and queuing delay of finite buffer size K based on Markovian model [7][10] as the following:

Drop Probability

Drop probability in a RED router can be approximated by

$$P_{drop} = \sum_{\hat{q}=k_h}^{K} \pi(\hat{q}) + \sum_{\hat{q}=k_l}^{k_h-1} \left(p_d(\hat{q}) \times \pi(\hat{q})\right). \tag{9}$$

Where $\pi(\hat{q})$ is stationary probability distribution of the average queue size and $p_d(\hat{q})$ is packet drop probability.

Consecutive Dropped Packet

$P_{drop}^{cons}(N)$ = Probability that N packets will be dropped consecutively and

$$\forall n \geq 0, \qquad P_{drop}^{cons}(\mathrm{N} > n) = \frac{\sum_{\hat{q}=0}^{K-1} \pi(\hat{q}) \left(p_d(\hat{q})\right)^{n+1}}{\sum_{\hat{q}=0}^{K-1} \pi(\hat{q}) p_d(\hat{q})}. \tag{10}$$

Equation (10) allows us in particular to evaluate the mean of the number of consecutive dropped packets, as shown below.

$$E(\mathrm{N}) = 1 + \frac{\sum_{\hat{q}=0}^{K-1} \pi(\hat{q}) \frac{\left(p_d(\hat{q})\right)^2}{1-p_d(\hat{q})}}{\sum_{\hat{q}=0}^{K-1} \pi(\hat{q}) p_d(\hat{q})}. \tag{11}$$

Average Queuing Delay

By using Little's theorem, average queuing delay is given by

$$D = \frac{E(\hat{Q})}{(1-P_{drop})\lambda} - \frac{1}{\mu}. \tag{12}$$

$E(\hat{Q})$ is mean of average queue size and given by

$$E(\hat{Q}) = \sum_{k=0}^{K} \frac{k\left(\frac{\lambda}{\mu}\right)^k \prod_{\hat{q}=0}^{k-1} \left(1-p_d(\hat{q})\right)}{\sum_{k=0}^{K} \left(\left(\frac{\lambda}{\mu}\right)^k \prod_{\hat{q}=0}^{k-1} \left(1-p_d(\hat{q})\right)\right)}. \tag{13}$$

4. ExRED MECHANISM

4.1 ExRED PRINCIPLE

The proposed scheme of our research is called Extended Drop Slope Random Early Detection (ExRED). The model of ExRED is shown in Fig.1,

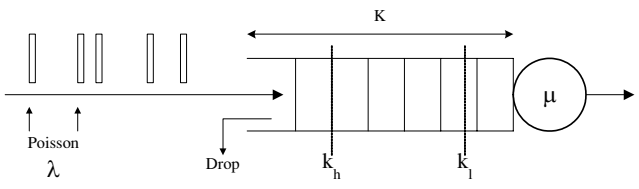

Fig. 1. Analytical model for ExRED

According to (8) to (13), we have observed that $p_d(\hat{q})$ influence all above performance issues especially drop probability. Packets will be dropped consecutively when $\hat{q} \geq k_h$ and a packet will be dropped randomly when $k_l \geq \hat{q} > k_h$. It is common sense that, under high load, most of packet drops occur when $\hat{q} \geq k_h$. In order to reduce a number of packet drop and consecutive drop, we modify drop distribution function and decrease drop probability in case of $\hat{q} \geq k_h$. To keep packet drop rate increasing smoothly but continue with a higher rate when queue size is more closed to limit of buffer size, $k_h \leq \hat{q} \leq K$, the function of p_d is modified to be a second order polynomial function of $\hat{q}$ and a new drop distribution must satisfy three conditions as below.

$$p_d(\hat{q} = k_h) = \max_p$$

$$p_d(\hat{q} = K) = 1$$

$$\frac{d}{d\hat{q}} p_d(\hat{q})\Big|_{\hat{q}=k_h} = \frac{\max_p}{k_h - k_l}$$

Thus, new packet drop distribution can be expressed as:

$$p_d(\hat{q}) = \begin{cases} 0 & ; \quad \hat{q} < k_l \\ \frac{\hat{q}-k_l}{k_h-k_l} \times \max_p & ; \quad k_l \leq \hat{q} < k_h \\ a_2\hat{q}^2 + a_1\hat{q} + a_0 & ; \quad k_h \leq \hat{q} \leq K \end{cases} \tag{14}$$

where

$$a_2 = \frac{(k_h - k_l) - (K - k_l)\max_p}{(k_h - k_l)(K - k_h)^2}$$

$$a_1 = \frac{(k_h^2 + K^2 - 2k_h k_l)\max_p - 2k_h^2 + 2k_h k_l}{(k_h - k_l)(K - k_h)^2}$$

$$a_0 = \frac{k_h^3 - k_h^2 k_l - (k_h^2 + 2k_h k_l + k_l K)K \max_p}{(k_h - k_l)(K - k_h)^2}$$

Packet drop probability will increase with higher rate for more seriously lack of available buffer space, the parameter setting must satisfy the following condition:

$$\max_p \leq \frac{k_h - k_l}{K - k_l}$$

Note that the recommended parameter setting in [5] and [11] is also satisfied above condition. When average queue size exceeds the maximum threshold, packet drop probability vs. average queue size by varying the difference of minimum and maximum threshold can be illustrated in Fig. 2.

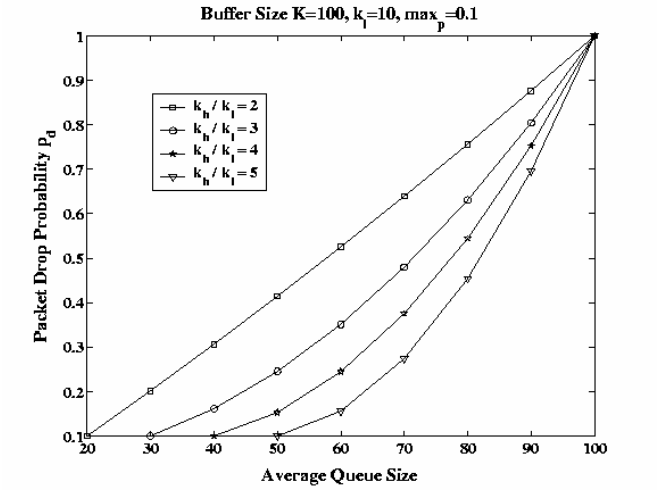

Fig. 2. Drop function of ExRED
when average queue size exceeds maximum threshold

4.3 Performance Analysis

We compare the drop function curve of ExRED to RED and illustrate as shown in Fig. 3. Performance will be presented in simulation result next section. However, in this section, we can roughly estimate the result using (9). When p_d is decreased, it follows then from (9) that the average packet drop is written as (15).

$$P_{drop} = \sum_{\hat{q}=k_h}^{K} \pi(\hat{q}) \times (a_2 \hat{q}^2 + a_1 \hat{q} + a_0) + \sum_{\hat{q}=k_l}^{k_h - 1} p_d(\hat{q}) \times \pi(\hat{q}) \tag{15}$$

If P_{drop}^{RED} and P_{drop}^{ExRED} are drop probability of RED obtained by (9) and ExRED obtained by (15), respectively. Fig. 4 illustrates $P_{drop}^{ExRED} \le P_{drop}^{RED}$ and therefore ExRED throughput gets better. Similarly, the decreasing of $p_d(\hat{q})$, the consecutive dropped packet probability $P_{drop}^{cons}(N)$ from (10) can be expressed by

$$\forall n, \quad 1 < n < K, \quad P_{drop}^{cons}(N_{ExRED} > n) < P_{drop}^{cons}(N_{RED} > n)$$

.

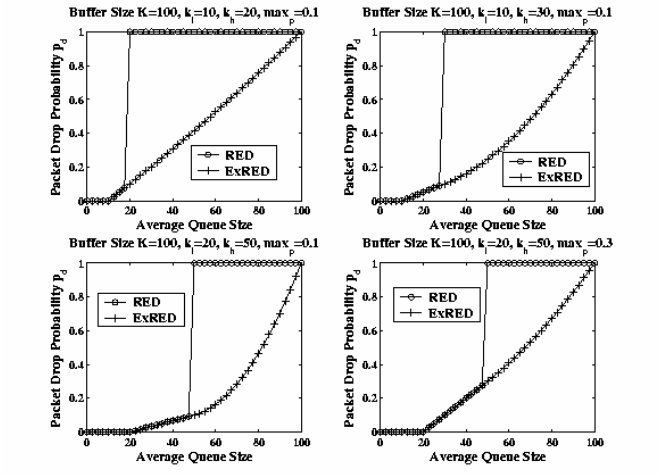

Fig. 3. Comparing of RED and ExRED Drop function

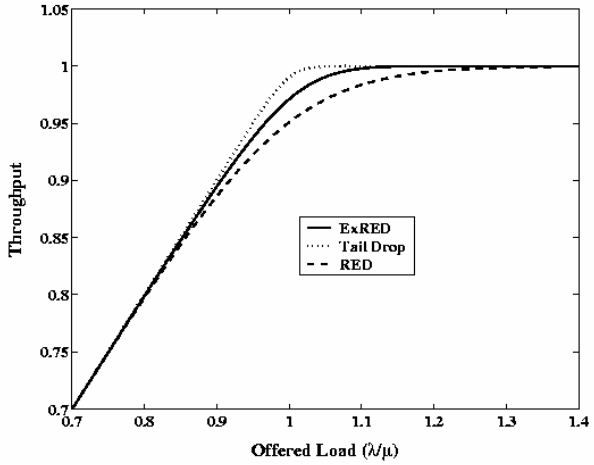

Fig. 4. Estimation of Throughput vs. Offered load

4.3 Simulation results

To compare the simulation result of our proposed mechanism, we model the discrete event simulation. We suppose that the arrivals occur randomly according to a Poisson Process with rate λ and service times of single-server with buffer size K is exponentially distributed at rate μ. We select the value of parameters setting in the simulation as the follows:

- μ = 1000 packets per second,
- K = 100 packets,
- w = 0.02,
- k_l =10, k_h =30: The values were chosen such that $k_l \le \frac{1}{4}\mathrm{K}$ and $k_h \ge 2k_l$ as suggested in [5].
- $\max_p$ = 0.1: The value was chosen as suggested in [11].

Throughput of Tail Drop, RED, and ExRED with varying offered load can be illustrated in Fig. 5.

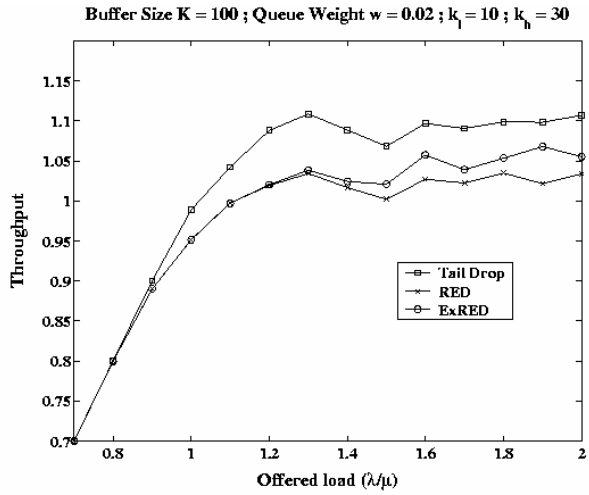

Fig. 5. Throughput of Tail Drop, RED, and ExRED vs. Offered load

Consecutive drop probability and its average of Tail Drop, RED, and ExRED can be illustrated in Fig. 6 and Fig. 7. Average queuing delay of Tail Drop, RED, and ExRED with varying offered load can be illustrated in Fig. 8,

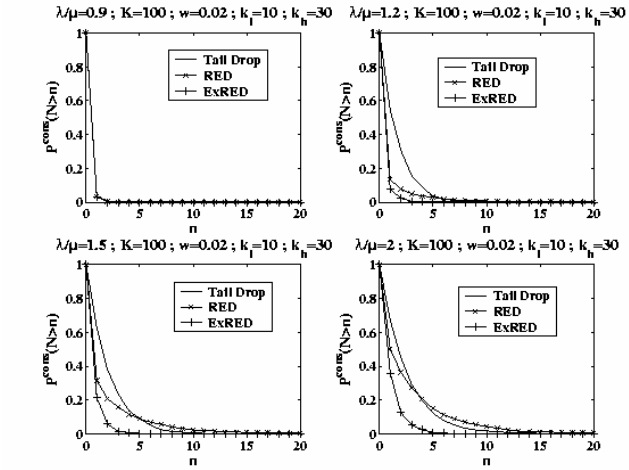

Fig. 6. Consecutive Drop Probability vs. Offered load at 0.9, 1.2, 1.5, and 2.0

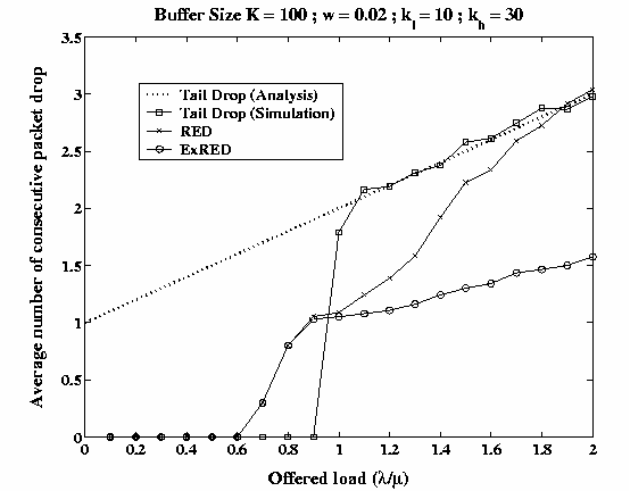

Fig. 7. Average number of Consecutive Drop vs. Offered load

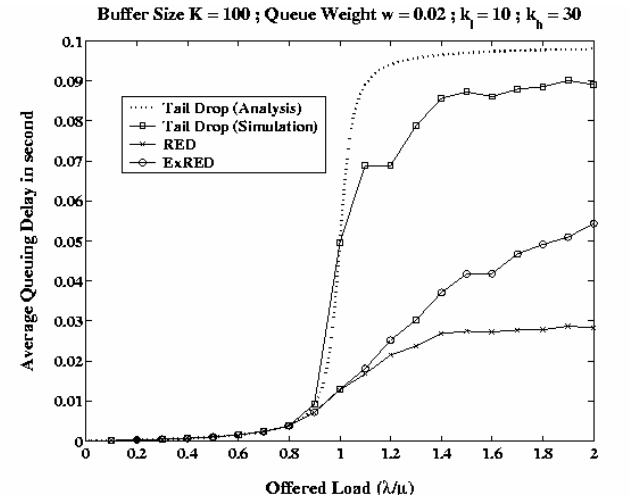

Fig. 8. Average Queuing Delay vs. Offered load

5. CONCLUSIONS

In order to relax an aggressive drop in case of the average queue size is over maximum threshold and not exceeds limit of buffer size, ExRED reduces packet drop probability. The 2nd order polynomial function shape of packet drop function provides a flexible increasing of drop probability. When average queue size exceeds maximum threshold, drop probability is set to the maximum drop probability parameter. The decreasing of available buffer space, the increasing rate of drop probability is accelerated higher. Not only the average packet drop is reduced, but also the consecutive drop. The simulation results in Fig. 5, Fig. 6, and Fig. 7, of section IV present that ExRED reduces packet drop and consecutive drop probability; hence throughput get better. Fig. 8 illustrates average queuing delay brought about by ExRED. It is a simple tradeoff between delay and throughput. However, we can conclude that ExRED performs a much lower delay when compare to Tail Drop while it performs a higher throughput when compare to RED. Moreover, end-to-end delay might be better if propagation and transmission delay of a retransmitted dropped packet is a long time.

Next, the complexity of implementation will be briefly discussed on this section. For average queue size exceeds the maximum threshold, packet drop probability will be computed for an incoming packet. This is unavoidably more complicate on ExRED than RED. Indeed, it does not add more difficulty. The algorithm in this part similar to what both RED and ExRED mechanism have done when average queue size is higher than minimum threshold but does not exceed maximum threshold. Packet drop probability could be computed on high performance processor or gotten from the table of previously calculated value stored in memory.

REFERENCES

[1] J. Postel, "Transmission control protocol," Information Sciences Institute, *RFC793*, 1981.

[2] V. Jacobson, "Congestion avoidance and control," *Proceeding of SIGCOMM '88*, (ACM), pp. 314-329, 1988.

[3] W. R. Steven, "TCP/IP illustrated", vol. 1, Addison Wesley, New York, 1994.

[4] J. C. Hoe, "Improving the start-up behaviour of a congestion control scheme for TCP", *Proceeding of SIGCOMM '96*, (ACM), pp. 314-329, 1996.

[5] S. Floyd and V. Jacobsan, "Random Early Detection Gateways for Congestion Avoidance", In IEEE/ACM Transactions on Networking, vol. 1, no. 4, pp. 397-413, August, 1993.

[6] "Recommendations on queue management and congestion avoidance in the Internet", RFC2309, April, 1999.

[7] T.Bonald, M. May, and J-C.Bolot, "Analytic Evaluation of RED Performance", In IEEE INFOCOM2000, Tel-Aviv, Israel, March, 26-30, 2000.

[8] "Assured Forwarding PHB Group", RFC 2597, June, 1999.

[9] R. W. Wolf, “Poisson Arrivals see Time Average”, Operations Research, 20, 223-231, 1982.
[10] R.Laalaoua, T.Czachorski, and T.Atmaca, “Markovian Model of RED Mechanism”, In Proceeding of First IEEE/ACM International Symposium on Cluster Computing and the Grid 2001, pp. 610-617, 2001.
[11] S. Floyd, RED: “Discussions of Setting Parameters”, http://www.aciri.org/floyd/REDparameters.txt, Nov, 1997