Improving The Loss Performance of Random Early Detection Gateway Using Fuzzy Logic Control.*

Mohammad Hossein Yaghmaee

Computer Department, Ferdowsi University of Mashhad, Faculty of Engineering, Mashad, Iran
E-mail: hyaghmae@ferdowsi.um.ac.ir

Hadi Safari Farmad

Khorasan Electrical Distribution Company Vakil-Abad Blwd., Farhang Blwd., Mashad, IRAN

Abstract

In this paper we propose a fuzzy logic implementation of Random Early Detection (RED) mechanism [1]. The main objective of the proposed fuzzy controller is to reduce the loss probability of the RED mechanism without any change in channel utilization. Using fuzzy logic capabilities, we try to dynamically tune the loss probability of RED gateway. To achieve this goal, we use a two-input-single-output fuzzy controller. The inputs of fuzzy controllers are 1) the error signal e1 which represents the difference between average queue size and a target point. 2) the error signal e2 which is calculated as the difference between the estimated value of incoming data rate and the target link capacity. The output of fuzzy controller is used to dynamically tune the max_p parameter of RED gateway. All simulation results show that the proposed fuzzy mechanism has better performance than traditional RED and Adaptive RED (ARED) mechanisms [3-5].

1. Introduction

The number of Internet users is rapidly grown. By increasing the number of users, the amount of traffic is also increased. For a network's router, when packets arrive faster than the output link capacity, then congestion can be occurred. To prevent the computer network from becoming a bottleneck, traffic management is necessary. In circuit switch networks, i.e. Public Switch Telephone Network (PSTN), each connection uses a fixed amount of bandwidth and data is transmitted at a constant bit rate. So call acceptance procedure is very simple in circuit switch networks. In packet switch networks due to random nature of input traffic and fault condition within the network, the traffic control function is a very complex task. The Internet is increasingly facing packet loss and queuing delays due to its rapid growth. This can lead to congestion collapse, which will reduce the quality of Internet applications [6]. It is important for the current Internet to support various traffic classes with variable bit rates. To support new Internet applications such as voice over IP, video on demand, multimedia and electronic commerce, it is necessary to design of effective congestion control and queue management algorithms. However, such a design is known to be difficult, because of variety of services supported in the Internet and their various demands for Quality of Service (QoS). As the most current Internet traffic is bursty, routers are provisioned with fairly large buffers to absorb this burstiness and maintain high link utilization. Active Queue Management (AQM) techniques try to detect and react to the congestion before its consequences such as packet loss or queuing delays. In reaction to suspected congestion, AQM algorithms drop packets early, or ECN-mark them to inform the congestion to the traffic sources. The most

* This research was in part supported by a grant from Institute for Studies in Theoretical Physics and Mathematics (I.P.M) (NO.CS1382-4-09)

Authorized licensed use limited to: Ferdowsi University of Mashhad Trial User. Downloaded on December 8, 2008 at 02:22 from IEEE Xplore. Restrictions apply.

important difference among AQM schemes is that when they guess congestion and how do they select the packets to be marked/dropped. AQM is the pro-active approach of informing the sender about incipient congestion before a buffer overflow happens. By using AQM mechanisms, the senders are informed early about congestion and can react accordingly. Random Early Detection (RED) [1] is the most important AQM mechanism which was proposed in order to solve problems caused by Drop Tail (DT) queue management mechanism. RED uses randomization to solve both the lockout and full queue problems in an efficient manner, without requiring any changes at the end hosts. RED simply sets minimum and maximum dropping thresholds. If the average buffer size exceeds the minimum threshold, RED starts randomly dropping packets based on a probability depending on the queue length. If buffer size exceeds the maximum threshold, then every packet is dropped. As expressed completely in [7,8], RED contains severe problems. The fundamental one is that it uses queue length as a congestion indicator. This indicator cannot completely show the severity of congestion. On the other hand, average queue length varies with the level of congestion and with the parameter settings. As a result, the performance of RED is too sensitive to traffic load and parameter settings [8]. Different variants of RED such as SRED [9] and Adaptive RED (ARED) [3-5] have been proposed which could fix some of its shortcomings. In [10-12] some AQM mechanisms were proposed which use flow based congestion indicator.

In this paper we use a fuzzy logic controller to reduce the loss probability of the RED mechanism. The main objective of our proposed model is to tune the loss probability of RED mechanism so that its average queue size remains nearly constant. Our proposed fuzzy controller has two inputs and single output. The inputs linguistic variables are:

1- The error signal e1 which is calculated as the difference between average queue size *(avg)* and a target point.

2- The error signal e2 which is calculated as the difference between the estimated incoming data rate and the target link capacity.

The output of fuzzy controller is used to calculate the new value of *max_p*. The main objective of our proposed fuzzy controller is to control the average queue size near a target point. When the *avg* is less than the target and the incoming data rate is less than the target link capacity, the *max_p* is decreased which decreases the loss rate. On the other hand, when *avg* is greater than the target and also the incoming data rate is greater than the target link capacity, the *max_p* is increased which increases the loss rate. By controlling the *max_p* dynamically, the proposed fuzzy mechanism can achieve a low loss rate.

The remainder of this paper is as bellow. The proposed fuzzy logic controller is explained in section 2. In section 3 by using computer simulation, we compare the performance of our proposed fuzzy controller with that of original RED algorithm. In section 4 we introduce two adaptive RED algorithms and then compare the performance of the proposed fuzzy controller with those of adaptive RED algorithms. Finally section 5 concludes the paper.

2. Proposed Fuzzy Controller

As we already mentioned, the RED algorithm has two main problems. The first problem is that it uses queue length as a congestion indicator which cannot completely show the severity of congestion. The second problem of RED mechanism is its sensitivity to variation in its parameters. Based on the simulation results given in [5], The RED mechanism is very sensitive to its parameter. To achieve to a good throughput and reasonable average queue lengths, we use a fuzzy logic controller to dynamically tune *max_p*. In response to measured queue length and measured incoming data rate, the proposed fuzzy controller, dynamically tune the value of *max_p*. The fuzzy controller uses two input parameters:

1. The error signal e1 which is calculated as: e1 = *avg* – TARGET, where *avg* is the average buffer size and TARGET is the target queue size which is calculated as bellow:

$$TARGET = \frac{\max_{th} + \min_{th}}{2}$$

2. The error signal e2 which is calculated as the difference between the estimated incoming data rate (Cest) and the target link capacity (Ct). Usually the target link capacity, Ct, is set to 0.97*C where C is the link capacity.

To estimate the incoming data rate, we use exponential averaging as bellow:

$$Cest = \frac{(1 - e^{\frac{-Del}{K}}) * B}{Del} + Cest * e^{\frac{-Del}{K}}$$

where Del is the inter-packet delay, B the packet size and K is the time constant (usually K is set to 0.9). We believe that our proposed fuzzy controller can solve the following problems of original RED algorithm:

Authorized licensed use limited to: Ferdowsi University of Mashhad Trial User. Downloaded on December 8, 2008 at 02:22 from IEEE Xplore. Restrictions apply.

1. RED uses the average queue size as a congestion indicator. The queue size is not a good indicator of the severity of the congestion, and the level of congestion notifications issued may be too bursty, leading to excessive packet loss. The proposed fuzzy controller adjusts the rate of congestion notification in response to the flow based congestion measure.
2. RED is very sensitive to traffic load and its parameters setting. To solve this problem, the proposed fuzzy controller can successfully auto-tune the *max_p* parameter to achieve reliably good results. The major goal of fuzzy controller is to tune *max_p* so that the average buffer size varies in the neighborhood of a target queue size. The output of fuzzy controller, *delta_max_p*, demonstrates the change value in the *max_p*. At the end of each periodic time interval (typically 0.2 sec.), the *max_p*, is updated.

It is clear that in the proposed fuzzy controller, the loss probability of RED mechanism is dynamically tuned based on the level of congestion. Figure 1 shows the membership functions of the proposed fuzzy controller.

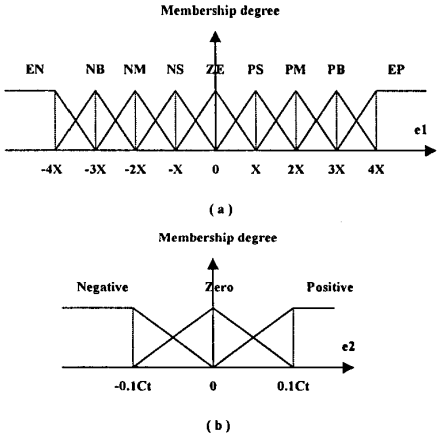


Figure 1. The membership functions of the proposed fuzzy logic controller a) membership functions of e1 b) membership functions of e2

In figure 1, the value of X is equal to 0.125*(max_th – min_th). As shown in figure 1, the term set of linguistic variables e1 and e2 are defined as bellow:
T(e1) ={ Extremely Negative (EN), Negative Big (NB),Negative Medium(NM), Negative Small (NS), ZEro (ZE), Positive Small (PS), Positive Medium (PM), Positive Big (PB), Extremely Positive(EP)}
T(e2) ={Negative, Zero, Positive}

The output term set of fuzzy logic controller is also defined as T(*delta_max_p*) ={ Decrease Very Fast (DVF), Decrease Fast (DF), Decrease Slow (DS), Decrease Very Slow (DVS), No Change (NC), Increase Very Slow (IVS), Increase Slow (IS), Increase Fast (IF), Increase Very Fast (IVF)}.

Since triangular and trapezoidal shaped functions offer more computational simplicity, to define the linguistic rules of the fuzzy logic controller we have used trapezoidal shaped functions. To obtain the best fuzzy rule base, we have used the conventional trial and error approach.

3. Simulation

In this section, by using the ns-2 simulator, we compare the performance of our proposed fuzzy controller with that of traditional RED mechanism. For this purpose we added our proposed fuzzy controller as a new queue management algorithm to ns-2 and we called it Fuzzy. To show the sensitivity of RED algorithm to *max_p* parameter, we consider two different values 1 and 0.01 for *max_p*. In the following figures, RED1 and RED0.01 refer the RED algorithm with *max_p*=1 and *max_p* =0.01, respectively. The network topology used for simulation is a single congested link in a dumbbell topology shown in figure 2. As shown in this figure, some TCP traffic sources are directly connected to a network router. To evaluate the performance of fuzzy controller we consider both FTP and bursty traffics. For FTP traffics, all traffic sources always have a packet to send and always send a 1000-bytes packet as soon as the congestion control window allows them to do so. The receiver immediately sends an acknowledge (ACK) packet when it receives a data packet. For both RED and Fuzzy mechanism, the max_th, min_th and wq were set to 80%buffer size, 20%buffer size and 0.002, respectively.

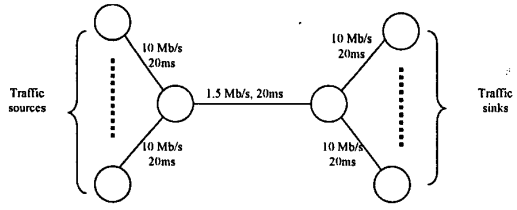


Figure 2. The simulation topology

Authorized licensed use limited to: Ferdowsi University of Mashhad Trial User. Downloaded on December 8, 2008 at 02:22 from IEEE Xplore. Restrictions apply.

3.1. FTP traffics

In figure 3, the packet loss probability of all mechanisms is plotted versus simulation time. The number of traffic sources was set to 100. All FTP traffic sources start to send packets at the start of simulation. The buffer size is equal to 50 packets. This figure shows the packet loss of our proposed fuzzy controller is 2% less than those of RED1 and RED0.01.

To evaluate the performance of proposed fuzzy controller under different congestion density, we performed more simulations. In this case, four traffic sources start at time 0 and at time 50, sixteen new traffic sources start to send the packets.

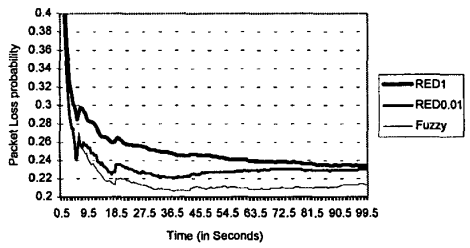


Figure 3. Packet loss rate of RED1, RED0.01 and Fuzzy

The packet loss rate of all mechanisms is shown in figure 4. This figure shows that for all mechanisms at time 50, the packet loss rate is increased. It can be seen that the packet loss rate of the fuzzy controller is less than those of RED1 and RED0.01.

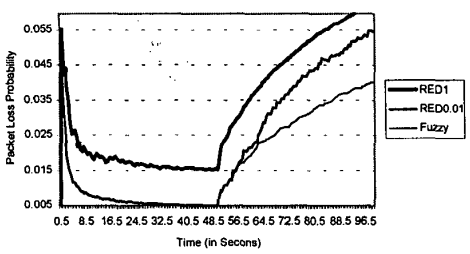


Figure 4. Packet loss rate of RED1, RED0.01 and Fuzzy (with an increase in congestion)

In the next simulation, we evaluate the performance of all mechanisms with a decrease in congestion. In this case, twenty traffic sources start at time 0. At time 50, sixteen traffic sources stop the sending of packets. Figure 5 shows the packet loss rate of RED1, RED0.01 and Fuzzy. As can be seen in this figure, for all mechanisms at time 50, the packet loss rate is decreased. It is clear that the packet loss rate of the proposed fuzzy controller is less than those of RED1 and RED0.01.

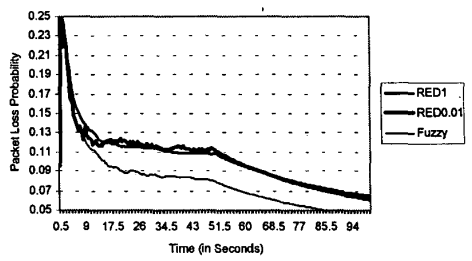


Figure 5. Packet loss rate of RED1, RED0.01 and Fuzzy (with a decrease in congestion)

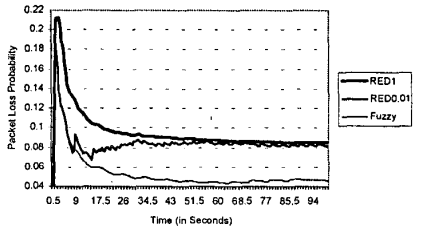
3.2. Bursty sources

In this section we evaluate the performance of fuzzy controller under bursty traffics. For this purpose, we simulated an exponentially bursty traffic in ns-2. The peak bit rate, mean burst size, mean silence size and the packet size of bursty traffic sources were set to 1Mb/s, 0.01 s, 0.1 s and 1000 bytes, respectively. In figure 6, for different values of traffic load, the packet loss rate of all mechanisms is plotted versus simulation time. The buffer size was set to 100 packets. This figures confirm that at different traffic load, the proposed fuzzy controller has better packet loss rate than RED1 and RED0.01.

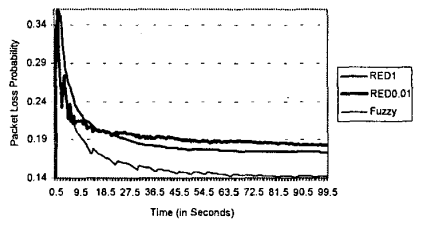
4. Related Works

In [3,4], Feng proposed the original Adaptive RED mechanism. This mechanism retains RED's basic structure and merely adjusts the parameter *max_p* to keep the average queue size between min_th and max_th. In [5], a new implementation of original adaptive RED mechanism was proposed by Floyd which could fix some of its shortcomings. As described in [5], the main objective of ARED is to keep the average queue size within a target range half way between max_th and min_th. To achieve this goal, the max_p is adaptived slowly in such away that stay within the range [0.01,0.5].

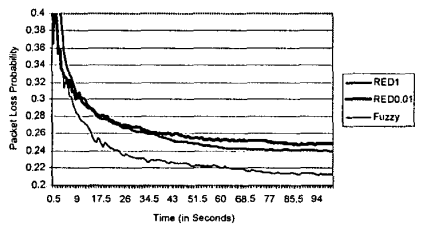
Authorized licensed use limited to: Ferdowsi University of Mashhad Trial User. Downloaded on December 8, 2008 at 02:22 from IEEE Xplore. Restrictions apply.



(a)



(b)



(c)

Figure 6. Packet loss rate of RED1, RED0.01 and Fuzzy under different traffic load a) low load (20 traffic sources) b) moderate load (50 traffic sources) c) high load (100 traffic sources)

The ARED uses Additive-Increase Multiplicative Decrease (AIMD) Policy. In ARED mechanism, at each periodic time intervals (typically 0.5 seconds), the average queue size is compared with the target value. When average queue size is greater than the target and also max_p is less than 0.5, then maxp is increased. When average queue size is less than the target and also max_p is greater than 0.01, then max_p is decreased. The robustness of ARED comes from its slow and infrequent adjustments of max_p. For ARED mechanism, when congestion density changes sharply, it could take some time, to adapt to this new value. As in ARED mechanism the value of max_p stays within the range [0.01, 0.5], then its performance will not be degraded during the transition period.

As we combined both flow based and queue based congestion indicator with the capabilities of fuzzy logic controllers, we believe that the performance of our proposed fuzzy controller is better than those of Feng's and Floyd's methods. To prove this, we performed new trials in ns-2 simulator. In figure 7, for bursty traffic, The packet loss rate of all mechanisms is plotted versus number of traffic sources. The buffer size was set to 500 packets. Based on this figure it is clear that the performance of the proposed fuzzy controller is better than those of Feng's and Floyd's mechanisms, especially when the congestion is heavy. For example when the number of traffic sources is 180, the packet loss rate of Feng's method, Floyed's method and Fuzzy is equal to 0.20, 0.18 and 0.15, respectively. In figure 8, for different values of the buffer size, the packet loss rate of all mechanisms is plotted versus simulation time. In this case, twenty bursty sources start to send their packets when the simulation begins. As can be seen in figure 8, for all value of buffer size, the proposed fuzzy controller shows better performance than Feng's and Floyd's mechanisms.

5. Conclusion

Based on previous research activities, the traditional RED algorithm contains severe problems. The performance of RED is too sensitive to traffic load and its parameter settings. In this paper we presented a fuzzy logic control approach for active queue management. In our proposed model, we use a fuzzy logic controller to dynamically tune the dropping probability of the RED mechanism. The main objective of our proposed model is to tune max_p parameter of RED mechanism so that its average queue size remains nearly constant. The performance of our proposed fuzzy controller was measured for two types of traffic, including FTP and bursty sources. It was observed that the proposed fuzzy controller can protect the QoS of TCP connections while it simultaneously utilizes the network resources well.

References

[1] S.Floyd, V.Jacobson, "Random Early Detection Gateways for Congestion Avoidance", IEEE/ACM Trans. On Networking, vol.1, Number 4, pp. 397-413, August 1993

Authorized licensed use limited to: Ferdowsi University of Mashhad Trial User. Downloaded on December 8, 2008 at 02:22 from IEEE Xplore. Restrictions apply.

2] The Network Simulator - ns-2 homepage, ttp://www.isi.edu/nsnam/ns/

3] W.Feng, D.Kandlur, D.Saha, and K.Shin, " Techniques or Eliminating Packet Loss in Congested TCP/IP Network", U.Michigan CSE-TR-349-97, November 1997.

[4] W.Feng, D.Kandlur, D.Saha, and K.Shin, " A self Configuring RED Gareway", Infocom, Mar 1999.

[5] S.Floyd, R.Gummadi, S.Shenker, "Adaptive RED: An Algorithm for Increasing the Robustness of RED", Technical Report, 2001

[6] V.Jacobson, " Congestion Avoidance and Control ", In Proceeding of ACM SIGCOMM, pages 314-329, Aguest 1998.

[7] M.May, J.Bolot, C.Diot, B.Lyles, "Reasons Not to Deploy RED", Proc. of 7th International Workshop on Quality of Service (IWQoS'99), pp. 260-262, June 1999

[8] W.Feng, D.Kandlur, D.Saha, K.Shin, "BLUE: A New Class of Active Queue Management Algorithms" U.Michigan CSE-TR-387-99, April 1999

[9] T.Ott, T.Lakshman, L.Wong, "SRED: Stabilized RED", in proceeding IEEE INFOCOM 1999

[10] Sanjeewa Athuraliya, Victor H. Li, Steven H. Low, Qinghe Yin, "REM: Active Queue Management",IEEE Network, May/June 2001.

[11] Bartek Wydrowski, Moshe Zukerman, " GREEN: An Active Queue Management Algorithm for a Self Managed Internet"

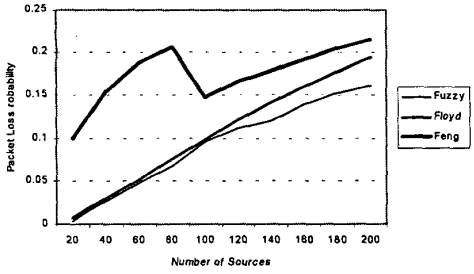
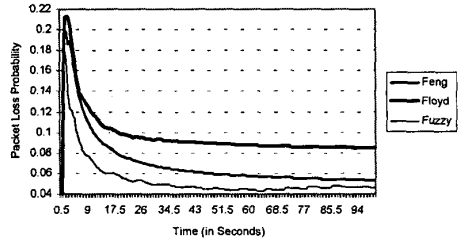
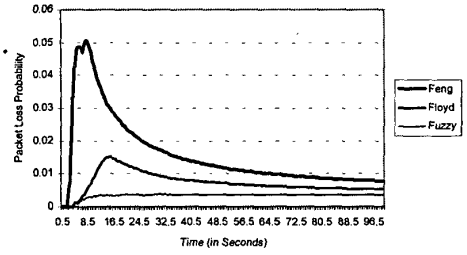


Figure 7. Packet loss rate of Feng's method, Floyd's method and Fuzzy at different number of traffic sources

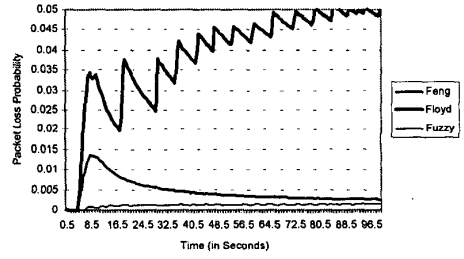
[12] R.Pan, V.Prabhakar, K.Psounis, " Choke: A Stateless Active Queue Management Scheme for Approximating Fair Bandwidth Allocation", Stanford University 1999.



(a)



(b)



(c)

Figure 8. Packet loss rate of Feng's method, Floyed's method and Fuzzy at different values of buffer size a) buffer size =100 packets b) buffer size =500 packets c) buffer size =1000 packets

Authorized licensed use limited to: Ferdowsi University of Mashhad Trial User. Downloaded on December 8, 2008 at 02:22 from IEEE Xplore. Restrictions apply.