# Overview of Fuzzy-RED in Diff-Serv Networks

L. Rossides[1], C. Chrysostomou[1], A. Pitsillides[1], A. Sekercioglu[2]

[1] Department of Computer Science, University of Cyprus, 75 Kallipoleos Street, P.O. Box 20537, 1678 Nicosia, Cyprus,
Phone: +357 2 892230, Fax: +357 2 892240.
[2] Centre for Telecommunications and Information Engineering, Monash University, Melbourne, Australia,
Phone: +61 3 9905 3503, Fax: +61 3 9905 3454.

**Abstract.** The rapid growth of the Internet and increased demand to use the Internet for time-sensitive voice and video applications necessitate the design and utilization of new Internet architectures with effective congestion control algorithms. As a result the Diff-Serv architectures was proposed to deliver (aggregated) QoS in TCP/IP networks. Network congestion control remains a critical and high priority issue, even for the present Internet architecture. In this paper we present Fuzzy-RED, a novel approach to Diff-Serv congestion control, and compare it with a classical RIO implementation. We believe that with the support of fuzzy logic, we are able to achieve better differentiation for packet discarding behaviors for individual flows, and so provide better quality of service to different kinds of traffic, such as TCP/FTP traffic and TCP/Web-like traffic, whilst maintaining high utilization (goodput).

## 1 Introduction

The rapid growth of the Internet and increased demand to use the Internet for time-sensitive voice and video applications necessitate the design and utilization of new Internet architectures with effective congestion control algorithms. As a result the Diff-Serv architecture was proposed [1] to deliver (aggregated) QoS in TCP/IP networks. Network congestion control remains a critical and high priority issue, even for the present Internet architecture.

In this paper, we aim to use the reported strength of fuzzy logic (a Computational Intelligence technique) in controlling complex and highly nonlinear systems to address congestion control problems in Diff-Serv. We draw upon the vast experience, in both theoretical as well as practical terms, of Computational Intelligence Control (Fuzzy Control) in the design of the control algorithm [2]. Nowadays, we are faced with increasingly complex control problems, for which different (mathematical) modeling representations may be difficult to obtain. This difficulty has stimulated the development of alternative modeling and control techniques which include fuzzy logic based ones. Therefore, we aim to exploit the well known advantages of fuzzy logic control [2]:

- Ability to quickly express the control structure of a system using a priori knowledge.

D.W. Bustard, W. Liu, R. Sterritt (Eds.): Soft-Ware 2002, LNCS 2311, pp 2-14, 2002.

- Less dependence on the availability of a precise mathematical model.
- Easy handling of the inherent nonlinearities.
- Easy handling of multiple input signals.

Our approach will be to adopt the basic concepts of RED [14], which was proposed to alleviate a number of problems with the current Internet congestion control algorithms, has been widely studied, and has been adapted, in many variants, for use in the Diff Serv architecture. Despite the good characteristics shown by RED and its variants in many situations, and the clear improvement it presents against classical droptail queue management, it has a number of drawbacks, including problems with performance of RED under different scenarios of operation, parameter tuning, linearity of the dropping function, and need for other input signals.

We expect that Fuzzy-RED, the proposed strategy, will be robust with respect to traffic modeling uncertainties and system nonlinearities, yet provide tight control (and as a result offer good service). It is worth pointing out that there is increasing empirical knowledge gathered about RED and its variants, and several 'rules of thumb' have appeared in many papers. It will be beneficial to build the Fuzzy Control rule base using this knowledge. However, in this paper we only attempt to highlight the potential of the methodology, and choose a simple Rule Base and simulation examples, but with realistic scenarios.

## 2 Issues on TCP/IP Congestion Control

As the growth of the Internet increases it becomes clear that the existing congestion control solutions deployed in the Internet Transport Control Protocol (TCP) [3], [4] are increasingly becoming ineffective. It is also generally accepted that these solutions cannot easily scale up even with various proposed "fixes" [5], [6]. Also, it is worth pointing out that the User Datagram Protocol (UDP), the other transport service offered by IP Internet, offers no congestion control. However, more and more users employ UDP for the delivery of real time video and voice services. The newly developed (also largely ad-hoc) strategies [7], [8] are also not proven to be robust and effective. Since these schemes are designed with significant non-linearities (e.g. two-phase—slow start and congestion avoidance—dynamic windows, binary feedback, additive-increase multiplicative-decrease flow control etc), and they are based mostly on intuition, the analysis of their closed loop behaviour is difficult if at all possible, even for single control loop networks. Even worse, the interaction of additional non-linear feedback loops can produce unexpected and erratic behavior [9]. Empirical evidence demonstrates the poor performance and cyclic behavior of the TCP/IP Internet [10] (also confirmed analytically [11]). This is exacerbated as the link speed increases to satisfy demand (hence the bandwidth-delay product, and thus feedback delay, increases), and also as the demand on the network for better quality of service increases. Note that for wide area networks a multifractal behavior has been observed [12], and it is suggested that this behavior—cascade effect—may be related to existing network controls [13]. Based on all these facts it is becoming clear that new approaches for congestion control must be investigated.

## 3 The Inadequacy of RED

The most popular algorithm used for Diff-Serv implementation is RED (Random Early Discard) [14]. RED simply sets some min and max dropping thresholds for a number of predefined classes in the router queues. In case the buffer queue size exceeds the min threshold, RED starts randomly dropping packets based on a probability depending on the queue length. If the buffer queue size exceeds the max threshold then every packet is dropped, (i.e., drop probability is set to 1) or ECN (Explicit Congestion Notification) marked. The RED implementation for Diff-Serv defines that we have different thresholds for each class. Best effort packets have the lowest min and max thresholds and therefore they are dropped with greater probability than packets of AF (Assured Forwarding) or EF (Expedited Forwarding) class. Also, there is the option that if an AF class packet does not comply with the rate specified then it would be reclassified as a best-effort class packet. Apart from RED, many other mechanisms such as n-RED, adaptive RED [15], BLUE [16], [17] and Three Color marking schemes were proposed for Diff-Serv queue control.

In Figure 1 we can see a simple Diff-Serv scenario where RED is used for queue control. A leaky bucket traffic shaper is used to check if the packets comply with the SLA (Service Level Agreement). If EF packets do not comply with the SLA then they are dropped. For AF class packets, if they do not comply then they are remapped into Best Effort Class packets. Both AF and Best Effort packets share a RIO [18] Queue. RIO stands for RED In/Out queue, where “In” and “Out” means packets are in or out of the connection conformance agreement. For AF and Best Effort class we have different min and max thresholds. EF packets use a separate high priority FIFO queue.

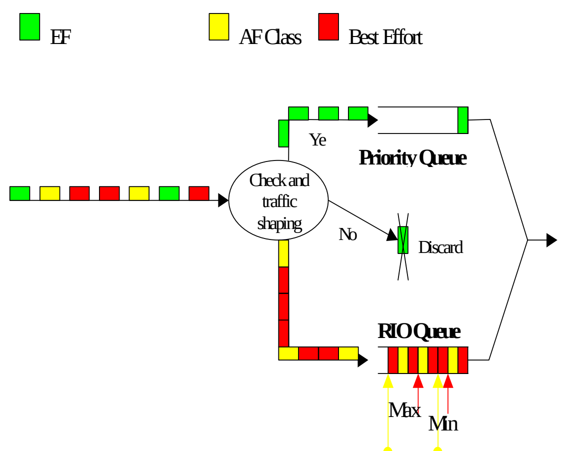


**Fig. 1.** Diff-Serv scenario with RED queue for control

Despite the good characteristics shown by RED in many situations and the clear improvement it presents against classical droptail queue management, it has a number of drawbacks. In cases with extremely bursty traffic sources, the active queue management techniques used by RED, unfortunately, are often defeated since queue lengths grow and shrink rapidly well before RED can react.

The inadequacy of RED can be understood more clearly by considering the operation of an ideal queue management algorithm. Consider an ideal traffic source sending packets to a sink through two routers connected via a link of capacity of L Mbps (see Figure 2).

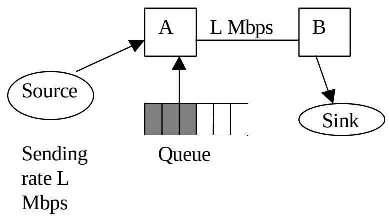


**Fig. 2.** Ideal Scenario

An ideal queue management algorithm should try to maintain the correct amount of packets in the queue to keep a sending rate of sources at L Mbps thus having a full 100% throughput utilization. While RED can achieve performance very close to this ideal scenario, it needs a large amount of buffer and, most importantly, correct parameterization to achieve it. The correct tuning of RED implies a "global" parameterization that is very difficult, if not impossible to achieve as it is shown in [16]. The results presented later in this article show that Fuzzy-RED can provide such desirable performance and queue management characteristics without any special parameterization or tuning.

## 4 Fuzzy Logic Controlled RED

A novel approach to the RED Diff-Serv implementation is Fuzzy-RED, a fuzzy logic controlled RED queue. To implement it, we removed the fixed max, min queue thresholds from the RED queue for each class, and replaced them with dynamic network state dependant thresholds calculated using a fuzzy inference engine (FIE) which can be considered as a lightweight expert system. As reported in [16], classical RED implementations with fixed thresholds cannot provide good results in the presence of dynamic network state changes, for example, the number of active sources. The FIE dynamically calculates the drop probability behavior based on two network-queue state inputs: the instantaneous queue size and the queue rate of change. In implementation we add an FIE for each Diff-Serv class of service. The FIE uses separate linguistic rules for each class to calculate the drop probability based on the input from the queue length and queue length growth rate. Usually two input FIEs can offer better ability to linguistically describe the system dynamics. Therefore, we can expect that we can tune the system better, and improve the behavior of the RED queue according to our class of service policy. The dynamic way of calculating the drop probability by the FIE comes from the fact that according to the rate of change of the queue length, the current buffer size, and the class the packet belongs, a different set of fuzzy rules, and so inference apply. Based on these rules and inferences, the drop probability is calculated more dynamically than the classical RED approach. This point can be illustrated through a visualization of the decision surfaces of the FIEs used in the Fuzzy-RED scheme. An inspection of these surfaces and the associated linguistic rules provides hints on the operation of Fuzzy-RED. The rules for the "assured" class are more aggressive about decreasing the probability of packet drop than increasing it sharply. There is only one rule that results in increasing drop probability, whereas two rules set the drop probability to zero. If we contrast this with the linguistic rules of the "best effort" class packets, we see that more rules lead to an increase in drop probability, and so more packet drops than the assured traffic class. These rules reflect the particular views and experiences of the designer, and are easy to relate to human reasoning processes.

We expect the whole procedure to be independent of the number of active sources and thus avoid the problems of fixed thresholds employed by other RED schemes [16]. With Fuzzy-RED, not only do we expect to avoid such situations but also to generally provide better congestion control and better utilization of the network.

## 5 Simulation Results

In this section we evaluate using simulation the performance of Fuzzy-RED and compare with other published results. The implementation of the traffic sources is based on the most recent version of ns simulator (Version 2.1b8a).

Three simulation scenarios are presented. Scenario 1 was a simple scenario used to make an initial evaluation and test of Fuzzy-RED. Scenario 2 compares the behavior of RED and Fuzzy-RED using only TCP/FTP traffic, and finally in Scenario 3 we introduce web traffic, reported to test the ability of RED [19] to evaluate a simple Diff-Serv implementation using RIO and Fuzzy-RED.

### 5.1 Scenario 1

We have done an initial testing of the performance of the fuzzy-RED queue management using the ns simulation tool with the simple network topology shown in Figure 3 (also used by other researchers for RED performance and evaluation [14]). The buffer size was set to 70 packets (max packet size 1000 bytes), the min threshold ($min_{th}$) for RED was 23 packets and the max threshold was 69 packets. The link between the two routers was set to 40 Mbps and the simulation lasted for 100 seconds. The scripts used for simulation Scenario 1 (and in all other simulation scenarios presented here) were based on the original scripts written in [14]. The rule files used for Scenario 1 were written without any previous study on how they can affect the performance of Fuzzy-RED, so they can been seen as a random pickup of sets and rules for evaluating Fuzzy-RED. After an extensive series of simulations based on Scenario 2 topology, and analysis of their results a new set of rule base files was created. This set of files was used in all the rest simulation scenarios (Scenarios 2 and 3) without any change in order to show the capabilities of Fuzzy-RED in various scenarios using different parameters.

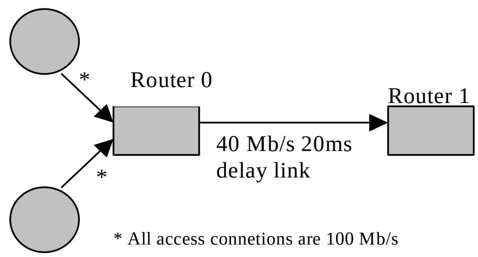


**Fig. 3.** Simple network topology used for the initial simulations

From the simulation results shown in Figure 4 and Figure 5 Fuzzy-RED achieves more than 99% utilization while RED and droptail fail to achieve more than 90%. As one can see from Figure 4 and Figure 5, Fuzzy-RED presents results very close to the ideal (as presented in Figure 2).

The throughput goes up to the 99.7% of the total link capacity (40 Mbps link) and the average queue size is around half the capacity of the buffer while maintaining a sufficient amount of packets in the queue for achieving this high throughput. While these are results from a very basic scenario (see Fig. 3) they demonstrate the dynamic abilities and capabilities of an FIE RED queue compared to a simple RED queue or a classical droptail queue. The results presented in the following simulation scenarios shows that these characteristics and abilities are maintained under all conditions without changing any parameters of Fuzzy-RED.

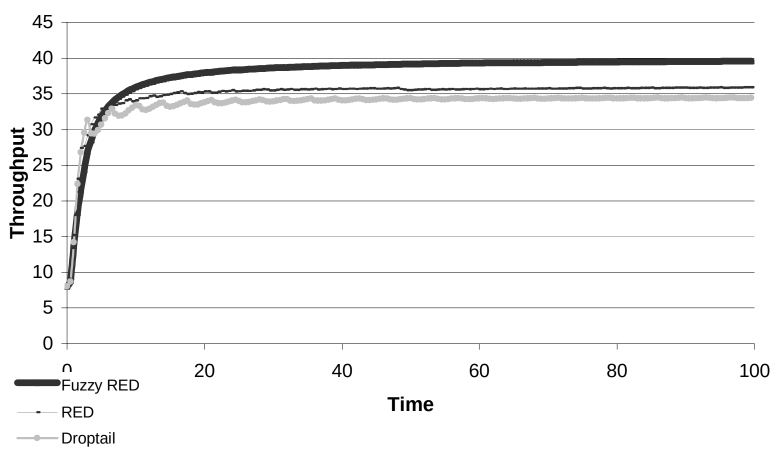


**Fig. 4.** Throughput vs. Time

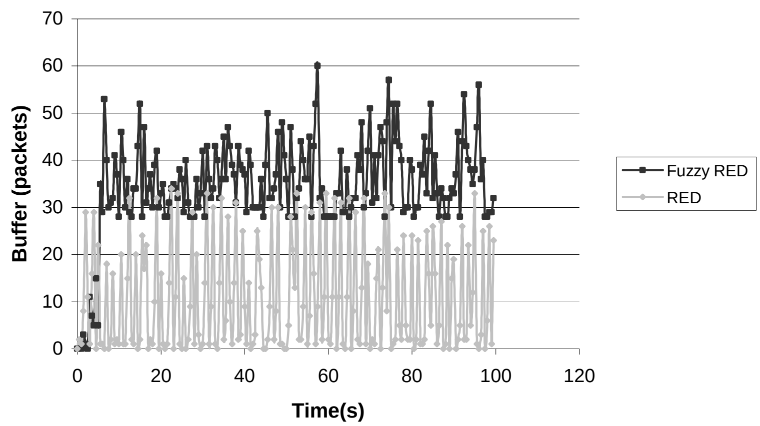


**Fig. 5.** *Buffer Size vs. Time*

### 5.2 Scenario 2

In order to show the dynamic abilities of Fuzzy-RED we use a new network topology. In this network topology we have five sources (each has different delay links varying from 1ms to 8 ms) sending traffic to five other sinks through a link connecting two routers. The link bandwidth is set to 45 Mbps in all cases. The propagation delay between the two routers is set to 10 ms. The TCP window is set to 100 packets. The buffer size is set to 140 packets. For RED the min threshold is set to 45 packets and the max threshold to 135 packets. The new network topology used in this series of simulation scenarios is shown in Figure 6.

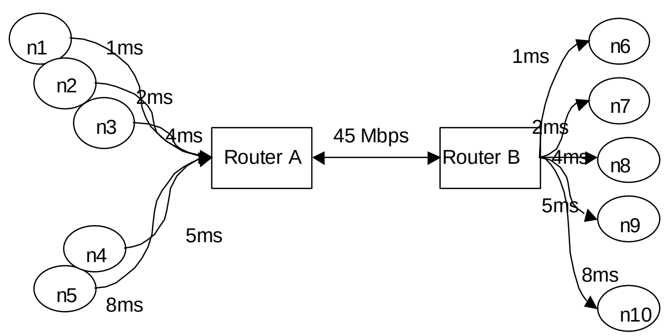


**Fig. 6.** Network Topology for Simulation Scenarios 2-5

From Figure 7 we see that although we have increased the number of sources sending data, Fuzzy-RED manages to keep the queue size in a close to constant value and significantly better than RED. What is more important is that Fuzzy-RED matches a near 100% throughput (99.57%), see Figure 8, while packet drops are kept at very low levels (Figure 10) compared to the number of packets sent (Figure 9).

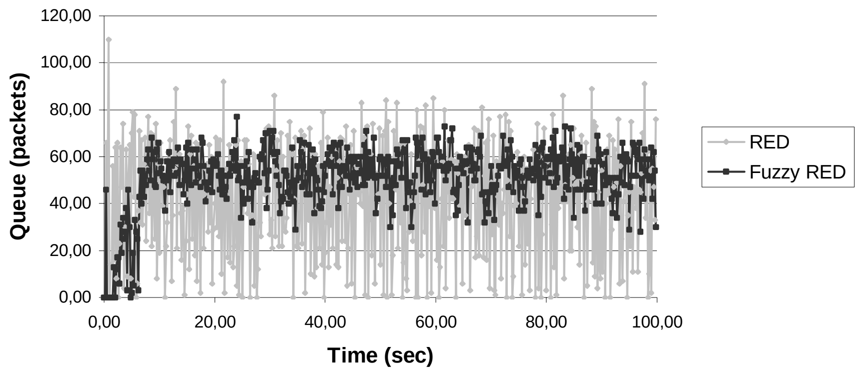


**Fig. 7.** Scenario 2 - Buffer size vs. time

These results show clearly that Fuzzy-RED manages to adequately control the queue size while keeping a higher than RED throughput (97.11% compared with 99.57% of Fuzzy-RED).

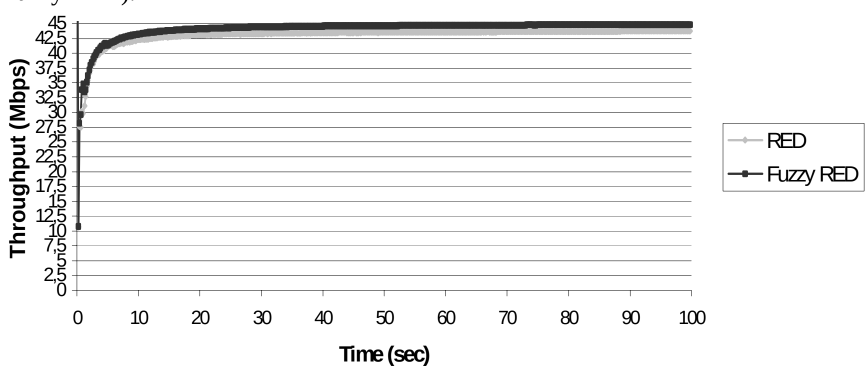


**Fig. 8.** Scenario 2- Throughput vs. time

### 5.4 Scenario 3

In Scenario 3 we introduce a new network topology used in [19] (Fig. 11). The purpose of this scenario is to investigate how Fuzzy-RED and RIO perform under a Diff-Serv scenario. To simulate a basic Diff-Serv environment we introduce a combination of web-like traffic sources and TCP/FTP sources. Half of the sources are TCP/FTP and the other half TCP/Web-like Traffic. Traffic from the Web-like sources is tagged as assured class traffic and the FTP traffic as best effort. We run the simulation three times for 5000 seconds each in order to enhance the validity of the results. The network topology is presented in Figure 11. We use TCP/SACK with a

TCP window of 100 packets. Each packet has a size of 1514 bytes. For the Droptail queue we define a buffer size of 226 packets. We use AQM (Fuzzy RED or RIO) in the queues of the bottleneck link between router 1 and router 2. All other links have a simple droptail queue. The importance of this scenario is that it compares and evaluates not simply the performance of an algorithm but the performance in implementing a new IP network architecture, Diff-Serv. This means that we want to check whether Fuzzy-RED can provide the necessary congestion control and differentiation and ensure acceptable QoS in a Diff-Serv network. We also attempt to compare Fuzzy-RED with RIO (a RED based implementation of Diff-Serv).

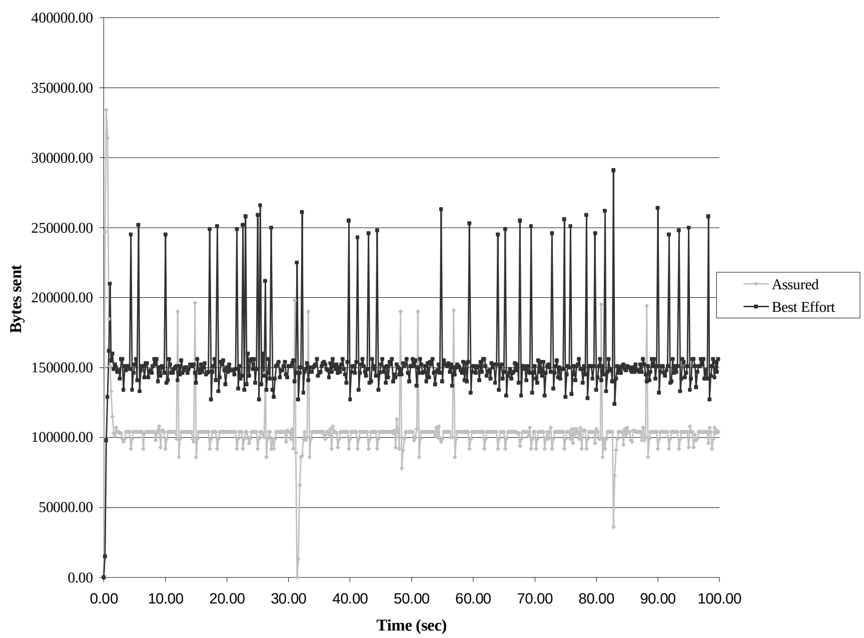


**Fig. 9.** Scenario 2 – Packets transmitted

The choice of distributions and parameters is based on [19] and [20] and is summarized in Table 1. The implementation of the traffic sources is based on the most recent version of ns simulator (version 2.1.8b). All results presented for this scenario were extracted from the ns simulation trace file

**Table 1.** Distributions and Parameters

| | Inter-page Time | Objects per page | InterObject Time | Object Size |
|---|---|---|---|---|
| **Distribution** | Pareto | Pareto | Pareto | Pareto |
| **Mean** | 50ms | 4ms | 0.5ms | 12KB |
| **Shape** | 2 | 1.2 | 1.5 | 1.2 |

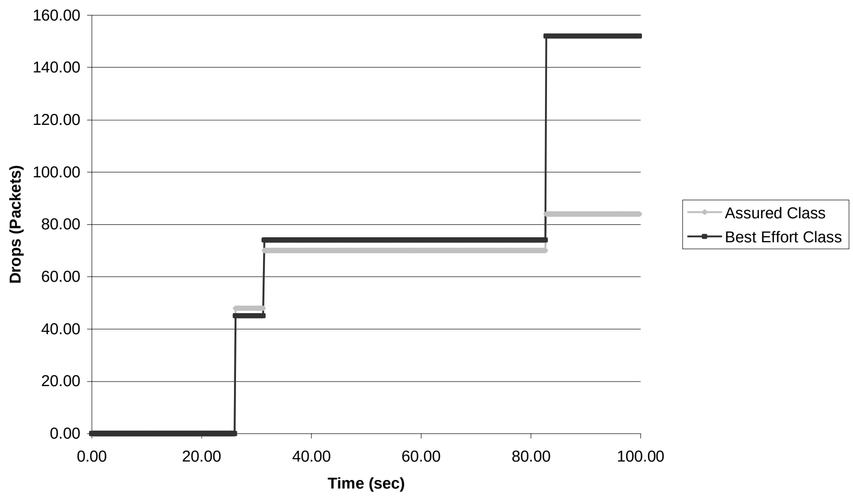


**Fig. 10.** Scenario 2- Packet Drops

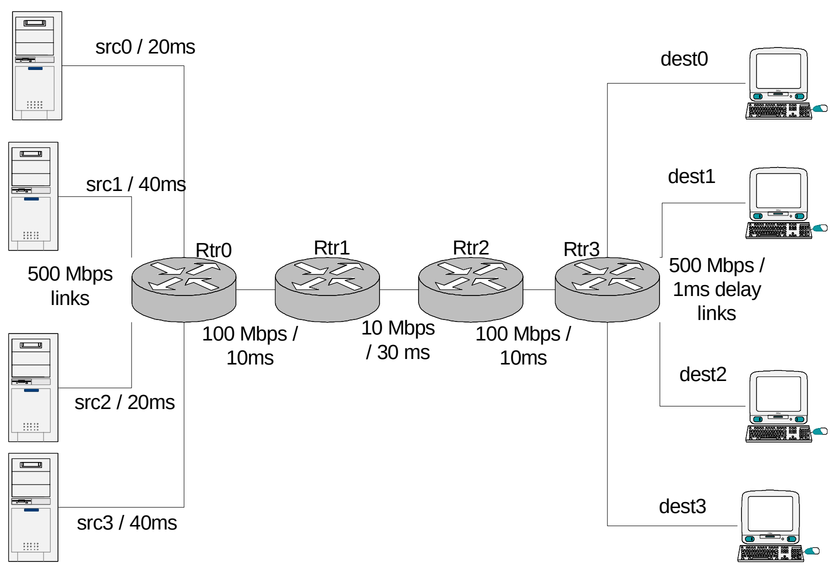


**Fig. 11.** Scenario 3 - Network Topology

Although this scenario is a simple and basic one it can be the starting point for investigating the abilities of Fuzzy-RED in a Diff-Serv network. The results presented here are limited to just two graphs since the purpose of this paper is to present the prospective of Fuzzy-RED in providing a comparable to RIO congestion control in Diff-Serv networks and not a detailed description of its performance.

From the results we see that although Fuzzy-RED and RIO show similar behaviour, Fuzzy-RED appears to control better the flow rate across the network. From Figure 12 we see the throughput behaviour. With throughput here we mean the rate at which traffic is coming to a link (in this case the link connecting router1 with router2) before entering any queue. So it is the total traffic arriving at router1 (both ftp and web traffic). From the graph RIO seems to stabilise its throughput around 10.25 Mbps (note that the link speed is limited to 10 Mbit/sec). This means that it can't effectively control the rate at which the sources are sending traffic (according to the ideal scenario shown in Figure 3). Around t=2500sec we see a small ascending step. At that point the traffic increases further from 10.1 Mbps to a 10.25 Mbps. This means that we have an increase of drops and therefore a decrease of goodput. We define goodput as the traffic rate traversing a link minus all dropped packets and all retransmitted packets.

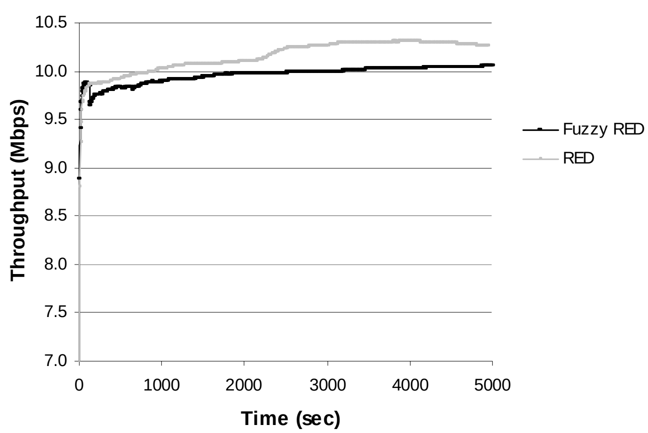


**Fig. 12.** Scenario 3 – Throughput vs. time

From Figure 13 we see that Fuzzy-RED delivers a steady goodput around 9.9 Mbps while RIO has a decrease from 9.9 to 9.8 due to dropped packets that create retransmissions. The difference is not as important as the fact that Fuzzy-RED seems to provide a more stable behaviour. This result along with the previous encourages us to proceed with further testing in the future.

## 6 Conclusions

Current TCP/IP congestion control algorithms cannot efficiently support new and emerging services needed by the Internet community. RED propose a solution, however in cases with extreme bursty sources (such a case is Internet) it fails to effectively control congestion. Diff-Serv using RIO (RED In-Out) was proposed to offer differentiation of services and control congestion. Our proposal in implementing Diff-Serv using a fuzzy logic controlled queue is a novel, effective, robust and flexible approach and avoids the necessity of any special parameterization or tuning, apart from linguistic interpretation of the system behavior.

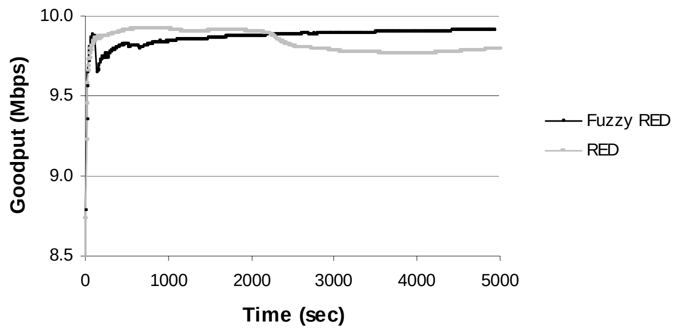


**Fig. 13.** Scenario 3- Goodput vs. time

It can provide similar or better performance compared to RIO without any retuning or parameterization. Specifically in scenario 3 we see that Fuzzy-RED, using the same rules and values in the fuzzy sets (i.e. no finer tuning), has achieved equal or better performance than RIO, in which we use the optimal parameterization discussed in paper [19]. From this scenario we see that Fuzzy-RED can perform equally well using homogeneous or heterogeneous traffic sources (in this case TCP/FTP traffic and TCP/Web-like traffic) without any change in the way we define it or any special tuning. We believe that with further refinement of the rule base through mathematical analysis, or self-tuning, our algorithm can achieve even better results.

In future work we will investigate further performance issues such as fairness among traffic classes, packet drops per class (a QoS parameter), utilization and goodput under more complex scenarios. We expect to see whether Fuzzy-RED can be used to provide the necessary QoS needed in a Diff-Serv network. From these results and based on our past experience with successful implementations of fuzzy logic control [21], [22], we are very optimistic that this proposal will offer significant improvements on controlling congestion in TCP/IP Diff-Serv networks.

## References

1. S.Blake et al. “An architecture for Differentiated Services”, December 1998, RFC 2475
2 W. Pedrycz, A. V. Vasilakos (Ed.), *Computational Intelligence in Telecommunications Networks*, CRC Press, ISBN: 0-8493-1075-X, September 2000.
3. V. Jacobson, Congestion Avoidance and Control, ACM SIGCOMM88, 1988.
4. W. Stevens, "TCP Slow Start, Congestion Avoidance, Fast Retransmit, and Fast Recovery Algorithms", RFC 2001, January 1997.
5. W. Stevens, TCP/IP Illustrated, Volume 1 The Protocols', Addison-Wesley, 1994.
6. V. Jacobson, R. Braden, D. Borman, TCP Extensions for High Performance, RFC 1323, May 1992.
7. K.K. Ramakrishnan, and S. Floyd, A proposal to add explicit congestion notification (ECN) to IP, draft-kksjf-ecn-03.txt, October 1998. (RFC2481, January 1999).

8. Braden et al, Recommendations on Queue Management and Congestion Avoidance in the Internet, RFC2309, April 1998.
9. C.E. Rohrs and R. A. Berry and S. J.O'Halek, A Control Engineer's Look at ATM Congestion Avoidance, IEEE Global Telecommunications Conference GLOBECOM'95, Singapore, 1995.
10. J. Martin, A. Nilsson, The evolution of congestion control in TCP/IP: from reactive windows to preventive flow control, CACC Technical Report TR-97/11, North Carolina State University, August 1997.
11. T.V. Lakshman and U. Madhow, The Performance of TCP/IP for Networks with High Bandwidth Delay Products and Random Loss, IEEE/ACM Transactions on Networking, vol. 5, pp. 336-350, June 1997.
12. A.Feldmann, A.C. Gilbert, W. Willinger, “Data Networks as cascades: Investigating the multifractal nature of the Internet WAN traffic”, SIGCOMM 98, Vancouver, 1998.
13. A. Feldmann, A. C. Gilbert, P. Huang, W. Willinger, “Dynamics of IP Traffic: A study of the role of variability and the impact of control”, Proceedings of ACM, SIGCOMM 2000.
14. S. Floyd and V. Jacobson, "Random Early Detection Gateways for Congestion Avoidance", ACM/IEEE Transaction on Networking, 1(4): pgs 397-413, August 1993
15. W. Feng, D. Kandlur, D. Saha, and K. Shin, “A self-configuring RED gateway,” IEEE INFOCOM’99, New York, Mar. 1999.
16. Wu-chang Feng, "Improving Internet Congestion Control and Queue Management Algorithms", PhD Dissertation, University of Michigan, 1999
17. W. Feng, D. Kandlur, D. Saha, and K. Shin, “Blue: A New Class of Active Queue Management Algorithms,” tech. rep., UM CSE-TR-387-99, 1999
18. Clark D. and Fang W. (1998), *Explicit allocation of besteffort packet delivery service*, IEEE/ACM Transactions on Networking, Volume 6, No. 4, pp. 362 – 373, August 1998.
19. G. Iannaccone, C. Brandauer, T. Ziegler, C. Diot, S. Fdida, M. May, Comparison of Tail Drop and Active Queue Management Performance for Bulk-data and Web-like Internet Traffic, 6th IEEE symposium and Computers and Communications. Hammamet. July 2001.
20. A. Feldmann, A. C. Gilbert, P. Huang, W. Willinger, “Dynamics of IP Traffic: A study of the role of variability and the impact of control”, Proceedings of ACM, SIGCOMM 2000
21. A. Pitsillides, A. Sekercioglu, G. Ramamurthy, “Effective Control of Traffic Flow in ATM Networks using Fuzzy Explicit Rate Marking (FERM)”, IEEE JSAC, Vol. 15, Issue 2, Feb 1997, pp.209-225.
22. A. Pitsillides, A. Sekercioglu, ‘Congestion Control’, in *Computational Intelligence in Telecommunications Networks*, (Ed. W. Pedrycz, A. V. Vasilakos), CRC Press, ISBN: 0-8493-1075-X, September 2000.