

# Fuzzy Logic Congestion Control in TCP/IP Networks: A Dual Explicit Notification Mechanism

*Clement N. Nyirenda*

Research Centre for Radio Access Technologies  
School of Electrical, Electronic and Computer Engineering  
University of KwaZulu Natal  
Durban  
South Africa  
nyirendac@ukzn.ac.za

**Abstract**—In this paper, a Fuzzy Logic Congestion Control algorithm which combines the merits of the Explicit Congestion Notification (ECN) and the Backward Explicit Congestion Notification (BECN) mechanisms in TCP/IP Networks is proposed. The performance evaluation of this algorithm was carried out by using both long-lived TCP (FTP) and short lived TCP (web) flows. Simulation results show that this algorithm reduces packet loss rates significantly while ensuring that the transmission rate of BECN Internet Control Message Protocol (ICMP) Source Quenches on the reverse path remains low. This algorithm exhibits minimum average transfer delay for web traffic compared to the Random Early Detection (RED) based approach.

**Index Terms**— Backward Explicit Congestion Notification (BECN), Explicit Congestion Notification, Fuzzy Logic Controller (FLC), Internet Control Message Protocol (ICMP), Random Early Detection (RED)

## I. INTRODUCTION

THE Internet has experienced an explosive growth over the past two decades and with that growth have come severe congestion problems. The need for Internet congestion control originally became apparent during several periods of 1986, when the Internet experienced the "congestion collapse" condition predicted by Nagle [1]. During these periods, a large number of widely dispersed Internet sites experienced simultaneous slowdown or cessation of networking services for prolonged periods. In response, Jacobson [2] proposed the congestion avoidance mechanisms that are now required in TCP implementations. These mechanisms operate in the hosts to enable TCP to reduce the sending rates in times of congestion. These mechanisms are responsible for preventing a meltdown of today's Internet. Considerable research on endpoint congestion avoidance mechanisms has been going on since 1988 [3, 4].

The effective deployment of endpoint congestion mechanisms triggered research in router (gateway) based congestion avoidance mechanisms in order to replace the traditional Drop-Tail policy, which discards arriving packets based on the overflow of the output buffer. The *Random Early Detection* (RED), proposed by Floyd and Jacobson in 1993 [5], marks one of the greatest milestones in router based congestion avoidance such that the Internet Engineering Task Force (IETF) recommended its

deployment for congestion avoidance in 1998[6]. Whenever, a packet arrives, RED will drop it with a certain probability ( $p_b$ ), which is calculated according to the average queue length ( $aql$ ) at each time when packets arrive. RED effectively prevents "TCP synchronization" and "full queues" problems that are associated with the Drop-Tail policy. However, RED still has the following weaknesses: 1) it is hard to configure its four operational parameters; 2) it becomes unstable if traffic changes suddenly; 3) it only uses queue length as an estimator of congestion. While the presence of a persistent queue indicates congestion indicates congestion, its length gives very little information as to the severity of congestion.

As a result, there have been many proposals for router based congestion avoidance [7, 8]. They are also known as Active Queue Management (AQM) algorithms. In [9], it is shown that, as capacity or delay increases, AQM mechanisms, including RED, all eventually become oscillatory and prone to instability. In [10], it is further pointed out that traditional AQM schemes demonstrate instability with the introduction of high bandwidth-delay links because they still require a careful configuration of non-intuitive control parameters. As a result, they are non-robust to dynamic network changes. They exhibit greater delays than the target mean queuing delay with a large delay variation, and large buffer fluctuations, and consequently cannot control the router queue. In [11], it is stressed that in practical queuing systems, the mean arrival rate and the mean service rate are frequently fuzzy ie. they cannot be expressed in exact terms. The European Network for Intelligent Technologies (EUNITE) Roadmap [12] points out that the application of fuzzy control techniques to the problem of congestion control in IP-based networks is suitable due to the difficulties in obtaining a precise mathematical model using conventional analytical methods.

### A. Explicit Congestion Notification Mechanisms

Explicit Congestion Notification was proposed for TCP/IP networks as a means of explicitly notifying end-hosts of network congestion by marking, instead of dropping packets. This is done during the packet's path from sender to receiver. Upon receipt of a congestion marked packet, the TCP receiver informs the sender (in the subsequent ACK) about incipient network congestion. In response, the sender invokes the congestion avoidance algorithm. Studies [13] have shown that the use of ECN mechanism for the notification of congestion to the end nodes prevents unnecessary packet drops retransmissions and time-outs. In terms of implementation, the ECN mechanism does not require generation of additional traffic at the router and can

be easily implemented in the datapath of routers. It requires the marking of one bit.

Recent studies [14,15] have reported the use of the Backward Explicit Congestion Notification (BECN) for congestion control in IP networks. The scheme relies on sending Internet Control Message Protocol (ICMP) Source Quenches as a means of reverse notification in TCP/IP networks. Studies have shown that BECN reduces transfer delay for short-lived flows and improves the goodput of long-lived flows under heavy congestion [14]. BECN has also shown significant improvement for TCP flows that traverse paths with a high bandwidth-delay product. Finally, its key benefit is that it is independent of the transport protocol and thus, can be used as a congestion notifier for multiple protocols. e.g. UDP or SCTP.

### B. Concerns with BECN and ECN

In [15], concerns have been raised over the use of BECN congestion notification in TCP/IP networks. Firstly, there is a concern that the performance of Internet routers could be adversely affected due to the overhead required for the generation of ICMP Source Quenches (ISQ). This is because ISQ generation would likely be performed by the control plane of routers as opposed to the data plane which is much faster. As a result, it is generally desirable to reduce the transfer of packets between the control and data plane in high speed routers.

Secondly, the issue of sending ISQs introduces a certain number of small packets in the reverse direction of the packets experiencing congestion. Therefore it is preferable to minimize such control traffic as much as possible.

Thirdly, the issue of reliable congestion notification is of concern. ECN can be more reliably delivered compared to ISQ notifications. ACK loss is not as much of a concern since ECN-Echo ACKs are continuously generated until the sender notifies the receiver that congestion notification has been received. This is achieved using a Congestion Window Reduced (CWR) mechanism. In the case of BECN, ISQ loss will not be detected by the sender nor can the congestion notification be reliably delivered.

The major concern with ECN relates to the long delay in notification of congestion. Under heavy load, the congestion will persist for a longer time. This leads to higher queue variance, reduced throughput and longer transfer delays for short lived flows.

### C. Motivation for a Fuzzy Logic Based Dual Explicit Congestion Notification Mechanism

In [15] a combined BECN/ECN mechanism is proposed and evaluated. The mechanism invokes ECN for low to medium level congestion and BECN for medium to high level congestion based on RED as the congestion detection algorithm. A new threshold, *becnthresh* is introduced between RED's minimum (*minth*) and maximum (*maxth*) threshold, in order to identify the point at which BECN is triggered. ECN is invoked when the average queue length is between *minth* and *becnthresh* while BECN is invoked when the average queue length is between *becnthresh* and *maxth*. Results show that this mechanism combines the merits of BECN and ECN algorithms and leads to improved performance compared to individual ECN and BECN schemes. It also results in significant reduction in ISQ reverse traffic compared to the BECN mechanism. However,

its weakness is that it is based on RED. As a result, it is bound to suffer from all the inherent shortcomings of RED. It also requires the configuration of a threshold between RED's *minth* and *maxth* for identifying the point at which BECN is triggered. The configuration of this threshold for optimum performance has been pointed out in [15] as an area of future research.

In this paper we extend the Fuzzy Logic Congestion (FLC) control principles proposed in [10,16] by implementing a simple fuzzy classifier that classifies the state of congestion at the router and invokes ECN for all cases of congestion while BECN is invoked only when the congestion status has been classified as high. The reception of multiple congestion notification signals, in succession, at the sending TCP does not lead to multiple reductions in the sending rate as TCP reduces its sending rate only once each roundtrip time (RTT).

The rest of this paper is organized as follows: Section II gives a brief Overview of Fuzzy Logic Control Theory. Section III describes our proposed Fuzzy Logic Based Dual Explicit Congestion Notification algorithm. In Section IV and V, we describe the simulation model and provide a discussion of results. The conclusion of this paper is presented in Section VI.

## II. OVERVIEW OF FUZZY LOGIC CONTROL THEORY

Fuzzy logic is a generalization of classical logic, in which there is a smooth transition between true and false. The basics of fuzzy logic are derived from the fuzzy set theory [12]. In conventional (crisp) sets, members are always fully categorized and there is no ambiguity about membership while in fuzzy sets, the transition from membership to non-membership is gradual rather than abrupt. A fuzzy set  $A$  in  $X$  is characterized by a membership function,  $\mu_A(x)$ , which associates each element in  $X$  with a real number in the interval  $[0, 1]$ .  $\mu_A(x)$  is known as the grade of membership. Hence the fuzzy set on the universe of discourse  $X$  is defined as:

$$A = \{(x, \mu_A(x)) \mid x \in X\} \quad (1)$$

Fuzzy rules are the backbone of a fuzzy logic system. A simple fuzzy rule can be written as follows:

$$\text{if } x \text{ is } \textit{HIGH} \text{ then } y \text{ is } \textit{POSITIVE} \quad (2)$$

where HIGH and POSITIVE are linguistic values defined by fuzzy sets on the universe of discourse  $X$  and  $Y$  respectively. The if-part, *if x is HIGH*, is known as *the antecedent* and the then-part, *y is POSITIVE*, is known as *the consequent*. A set of linguistic rules used to map fuzzy inputs to outputs is known as *the rule base*. The functional components of a fuzzy controller are shown in Fig 1.

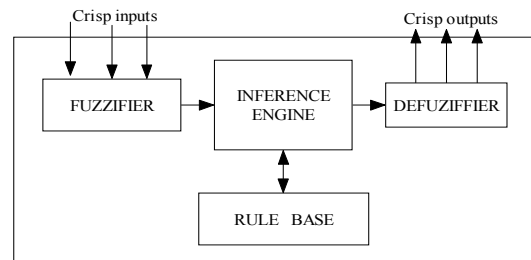


Fig. 1. Fuzzy Logic Controller Scheme

The Fuzzifier calculates suitable sets of degree of membership, called fuzzy sets for the crisp (discrete) inputs. The Inference Engine evaluates output fuzzy sets from input

sets using the predefined fuzzy rules contained in the rule base. The Defuzzifier transforms the output fuzzy set into a crisp number to be useful to the real world. The Inference Engine calculates *the degree of activation* for every rule in the rule base. If the antecedent for *rule j* contains one variable, the rule's degree of activation is equal to the degree of membership of that single variable:

$$\mu(r_n) = \mu_1(x_1) \quad (3)$$

where  $\mu(r_n)$  is the degree of activation of *rule n* and  $\mu_1(x_1)$  is the degree of membership of input,  $x_1$ .

If the antecedent for *rule j* contains more than one variable in the form:

$$\text{Rule } j: \text{ if } A_j^1 \text{ and } A_j^2 \text{ and } \dots \text{ and } A_j^n \text{ then } b_j \quad (4)$$

where  $A_j^k$  is a fuzzy set with membership function,  $\mu_j^k: \mathfrak{R} \rightarrow [0,1]$ ,  $j=1, \dots, m$ ,  $k=1, \dots, n$ ,  $b_j \in \mathfrak{R}$ . In this case, *the degree of activation* for *rule j*,  $\mu(r_j)$  is determined using the minimum t-norm as follows:

$$\mu(r_j) = \mu_j^1(x_1) \otimes \mu_j^2(x_2) \otimes \dots \otimes \mu_j^n(x_k) \quad (5)$$

Where  $\mu(r_j)$  is the degree of activation of *rule j*.  $\mu_j^n(x_k)$  is the degree of membership of input,  $x_k$ . The degree of activation is inferred as the degree of membership of the output variable upon its fuzzy set, which is defined in the corresponding consequence. The output of inferring  $m$  rules upon the  $i^{\text{th}}$  output variable is the aggregation of the individual rule outputs upon that output variable. The implied output sets are combined to formulate a crisp output through a routine known as defuzzification. The widely applied defuzzification method is the Centre Of Gravity (COG) technique, which computes the weighted-average of the centre of gravity of each membership function. The COG of the system with  $m$  rules is as follows:

$$y(x) = \frac{\sum_{j=1}^m b_j \mu(r_j)}{\sum_{j=1}^m \mu(r_j)} \quad (6)$$

where  $b_j$  is the centre of the membership function recommended by the consequent of rule  $j$ . The membership functions for the fuzzy controller are initialized by the user based on a priori knowledge.

### III. THE FUZZY LOGIC BASED DUAL EXPLICIT CONGESTION NOTIFICATION ALGORITHM

A single FIFO queue in which all packets are treated equally is assumed. Based on the guidelines in [16], the queue status is sampled at 160Hz in order to obtain the queue-occupation size (backlog), denoted by  $q(t)$  and the traffic incoming rate, denoted by  $r(t)$ , which are the two inputs to our controller. The backlog,  $q(t)$ , is translated into a ratio (with respect to the Buffer Size):

$$BBR = q(t) / BS \quad (7)$$

where BBR denotes the backlog to buffer size ratio. BS is the Buffer Size.

The traffic incoming rate is a function of the variation of the queue,  $q_{\text{var}}$ . Therefore it is determined by monitoring the variation of the queue-occupation size. This is captured by using the following equation:

$$q_{\text{var}} = q(t) - q(t - \tau) \quad (8)$$

A very low (very negative)  $q_{\text{var}}$  denotes decreasing traffic incoming rates while a very high (very positive)  $q_{\text{var}}$  denotes increasing traffic incoming rates. In order to limit our fuzzy operations in the positive plane, the equation in (8) is scaled up by  $0.5 \cdot BS$ . It is further normalized as a ratio, *var* with respect to the Buffer Size, BS as follows:

$$\text{var} = (q_{\text{var}} + 0.5 \cdot BS) / BS \quad (9)$$

Scaling factors,  $SG_1$  and  $SG_2$  are applied to both inputs:

$$SG_1 = SG_2 = 100 \quad (10)$$

These scaling factors help to ensure that the crisp inputs to the fuzzy system fall between 0 and 100 for ease of computation. The diagram of our controller and classifier mechanism is shown in Fig 2. The membership functions of the two input variables, Backlog to Buffer Ratio (BBR) and the normalized queue variation (*var*) are shown in Fig 3 and 4 respectively. The membership function of the fuzzy controller's output, the change in packet marking probability,  $\Delta p_b$ , is shown in Fig 5. The rule base for the fuzzy controller is shown in Fig 6. The new packet marking probability,  $p_b$  is calculated as follows:

$$p_b(t) = p_b(t - \tau) + \Delta p_b(t) \quad (11)$$

The probability,  $p_b$  is fed to the fuzzy classifier which generates three output values,  $\beta_1$ ,  $\beta_2$  and  $\beta_3$  based on the membership function in Fig 7. These values denote congestion status levels 1, 2 and 3 respectively. Level 1 denotes "the no congestion status". Level 2 denotes "the general congestion status". Level 3 denotes "the high congestion status".  $\beta_1$ ,  $\beta_2$  and  $\beta_3$  are degrees of membership for fuzzy sets, *no congestion*, *general congestion* and *high congestion* respectively.

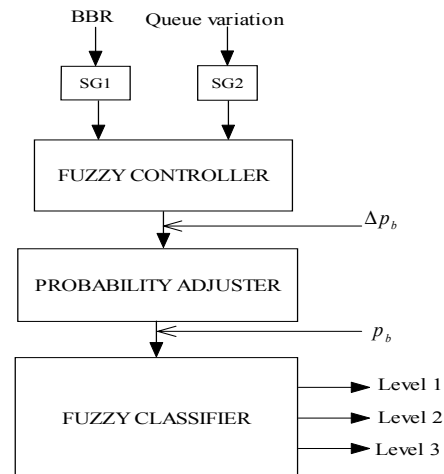


Fig. 2. The proposed fuzzy controller and classifier

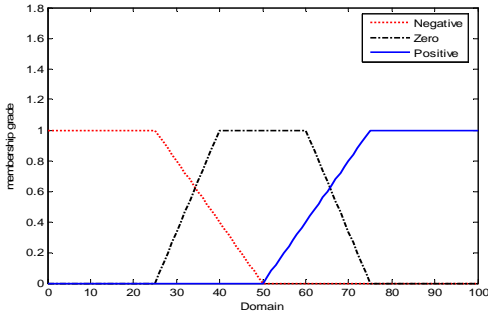


Fig. 3. Membership function of the Backlog to Buffer Ratio (BBR)

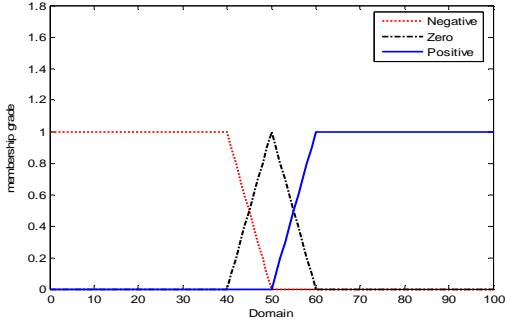


Fig. 4. Membership function for the queue variation, var

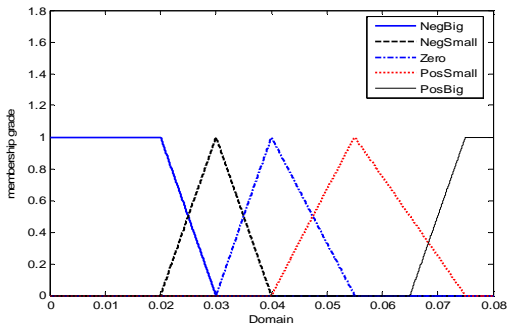


Fig. 5. Membership functions for the change in packet marking probability,  $\Delta p_b$ .

```

/* Linguistic rules of FBC */
if BBR is low and queue variation is negative then  $\Delta p_b$  is negative big (NB);
if BBR is low and queue variation is zero then  $\Delta p_b$  is negative small (NS);
if BBR is low and queue variation is positive then  $\Delta p_b$  is zero (Z);
if BBR is medium and queue variation is negative then  $\Delta p_b$  is negative big (NB);
if BBR is medium and queue variation is zero then  $\Delta p_b$  is zero (Z);
if BBR is medium and queue variation is positive then  $\Delta p_b$  is negative small (NS);
if BBR is high and queue variation is negative then  $\Delta p_b$  is negative small (NS);
if BBR is high and queue variation is zero then  $\Delta p_b$  is positive small (PS);
if BBR is high and queue variation is positive then  $\Delta p_b$  is positive big (PB);

```

Fig. 6. The Rule base for the Fuzzy Logic Controller

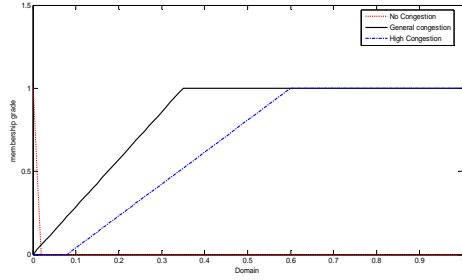


Fig. 7. Membership function for the change in packet marking probability,  $\Delta p_b$ .

In Fig. 7, the high congestion status is a subset of the general congestion status. ECN is invoked in all congestion cases while BECN is invoked only when the congestion level is high. ECN marking is invoked in the deque routine in order to reduce the delay incurred in delivery of congestion information to the sender. BECN marking is invoked in the enqueue routine in order to inform the sender about incipient congestion as soon as it is detected. The higher the value of  $\beta_2$ , the more the ECN marks. The higher the value of  $\beta_3$ , the more the BECN marks.

#### IV. SIMULATION MODEL

In order to study our scheme, simulations were conducted using the dumb-bell shaped topology shown in Fig. 8. We have used the Network Simulator [NS-2.28] maintained by the MASH Research Group (University of California, Berkeley) as the simulation platform. The bottleneck bandwidth is 10Mbps with a propagation delay of 40ms. All other links have 100Mbps with a propagation delay of 2ms. Traffic flow is from sources,  $S(1) \dots S(n)$  through routers, Router1 and Router2 to receivers,  $R(1) \dots R(n)$ . The FTP traffic model in the Network Simulator is used with an infinite amount of traffic to send. TCP type is New Reno with a data packet size of 1000 bytes and ACK packet size of 40 bytes. The TCP clock granularity is set to 100ms. The maximum TCP congestion window is set to 100KB to ensure that in all experiments the TCP transfers were in the order of the bottleneck bandwidth delay product of the link. Delayed ACKs are not used in these simulations. We follow the guidelines by Floyd [5] in setting RED queue parameters as follows:  $minth = 15KB$ ,  $maxth = 3 * minth$ ,  $buffer\ size = 2 * maxth$ ,  $becn\ thresh = 20KB$  and  $40KB$ ,  $maxp = 0.1$ ,  $wq = 0.002$ ,  $mean\ packet\ size = 1000\ bytes$ . Packet-based marking for RED is used. For RED, ECN marking is employed at the deque routine when the average queue is above  $maxth$ . This is done for purposes of fairness in comparison because the fuzzy scheme employs packet-based ECN marking in the deque routine.

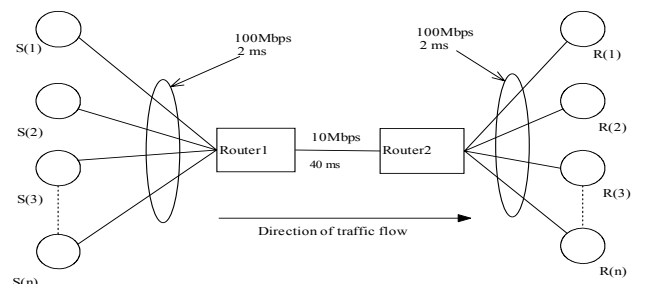


Fig. 8. Simulation Topology

The following performance metrics are used for evaluation:

1) *Bottleneck utilization (U)*: Utilization is computed using the following formula:

$$U = \frac{(8 * 1000 * n) + (8 * 40 * m) + (8 * 40 * p)}{c} \quad (12)$$

Where n, m and p are the number of data, ACKs and ICMP packets respectively that successfully traverse the bottleneck link and are received by the receiver. C is the capacity of the bottleneck link.

2) *Percentage loss*: Measures the ratio of packets dropped at the bottleneck link to the total number of packets injected into the bottleneck link for a particular flow or set of flows.

3) *Percentage ISQ reverse traffic*: Computed as the number of BECN ISQs generated in the reverse direction of data flow as a ratio of the total number of packets in the reverse direction.

4) *The mean queue length for short lived flows (web traffic)*: This helps in determining the mean queuing delay for web traffic.

5) *The mean queue variance for short lived flows (web traffic)*: This helps in determining the average transfer delay for web traffic.

## V. EXPERIMENTS AND RESULTS

In this section we describe two sets of experiments and present explanations for observed behavior of our algorithm. In this experiments, we compare the performance of our algorithm with two scenarios of the RED based approach (with  $becn_{thres} = 20$  and  $40$  packets). Apart from the individual operational settings for each of these algorithms all other settings are the same. The simulation period for each experiment was set at 200s.

### A. Homogeneous long-lived flows

In this experiment, FTP flows are run on RED ( $becn_{thres}=20$ ), RED ( $becn_{thres}=40$ ) and on the Fuzzy queue. The FTP flows start randomly within the first 5s of simulation. The number of FTP connections is varied using 10, 20, 30, 40, 50, 60, 70, 80, 90 and 100 flows. The percentage loss rates, percentage reverse traffic, and utilization of the bottleneck link, are measured. Fig. 9–Fig. 11 show the results.

Fig. 9 shows that the fuzzy approach exhibits a lower packet loss rate. It reaches a maximum of 1% for 100 FTP flows while the RED based approaches have maximum loss rates of 1.5% at 70 and 80 FTP flows respectively. The loss rate in RED begins to decrease as the number of FTP flows goes beyond 80. As the number of flows increases the RED queue operates permanently above the maximum threshold of 45 such that a large number of packets are ECN marked because the packet marking probability which is a function of the average queue length is predominately high. In terms of ISQ reverse traffic, the fuzzy approach achieves an average rate of 0.595% in contrast to 1.259% and 1.888% for RED40<sup>1</sup> and RED20<sup>2</sup> respectively [Fig. 10]. In Fig.11, the fuzzy approach achieves higher and more stable link utilization. RED20 has a higher initial link utilization than

RED40 because it thrives on BECN as the congestion notification mechanism but as the number of FTP flows increases, both of them undergo a slight decrease in link utilization due to transition in packet marking from BECN to ECN. RED40 is the worst affected because it has a shorter BECN marking interval (40 – 45) such that its period for BECN marking is shorter than that of RED20 which has a longer BECN marking interval (20-45). ECN marking is generally characterized by low link utilization compared to BECN marking [15]. As the number of FTP flows increases beyond 60, link utilization increases in both RED20 and RED40 and at 100 FTP flows all the three approaches register almost the same level of utilization.

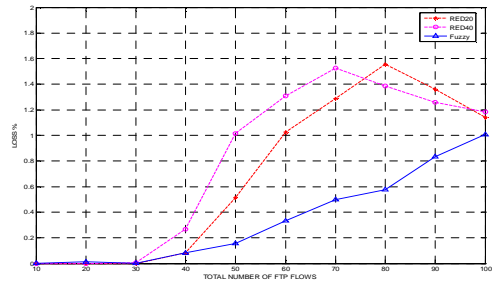


Fig. 9 Packet Loss rates

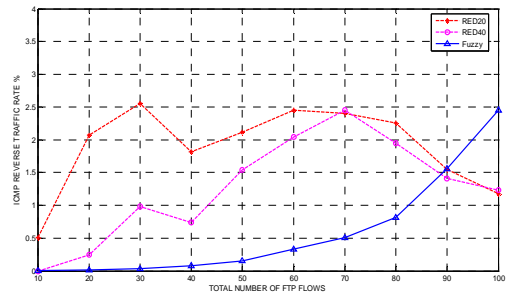


Fig. 10 ISQ Reverse traffic

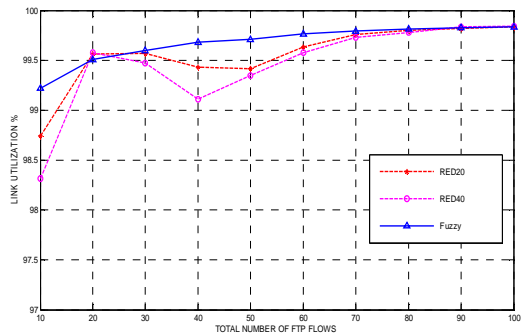


Fig. 11 Bottleneck link utilization

### B. Combination of long-lived flows and short lived flows

In this experiment, short lived TCP flows start randomly between 50s and 100s. The number of FTP flows is still varied using 10, 20, 30, 40, 50, 60, 70, 80, 90 and 100 flows. They start randomly within the first 5s. The short lived flows are generated by using the Pareto Traffic generator. The aim is to simulate web traffic. The short lived traffic is configured with the following parameters:  $packet\_size = 1000$ ,  $burst\_time = 200s$ ,  $idle\_time = 200s$ ,  $shape = 1.5$ ,  $rate = 1.5$ . We measure the mean of the queue at the congested router. Fig. 12 shows the observed results. The mean queue length for the fuzzy approach is generally the lowest ranging from 32 to 48 while for the RED approaches it ranges from 22 to 60. This means that the fuzzy approach exhibits the

<sup>1</sup> RED40 denotes RED with the BECN threshold set to 40

<sup>2</sup> RED20 denotes RED with the BECN threshold set to 20

lowest queuing delay. This happens because the Fuzzy algorithm uses backlog and queue variation as inputs while RED uses only backlog. A low queuing delay denotes that web object transfer delay is generally the lowest in the Fuzzy approach. This is a good characteristic for the HTTP end user because he can be assured of faster response times.

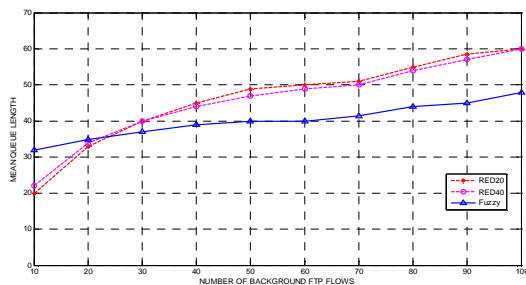


Fig. 12 Mean Queue for short lived flows

## VI. CONCLUSION

In this study, a Fuzzy Logic Based Dual Explicit Congestion Notification Mechanism has been proposed. This mechanism effectively combines the merits of ECN and BECN without the need for the configuration of thresholds. It reduces packet loss rates while ensuring that ISQ reverse traffic rates remain low. This algorithm also exhibits minimum queuing delay for web traffic compared to RED.

As part of future research, a particle swarm optimizer will be used for tuning the membership functions in order to achieve optimum performance and avoid the “trial and error” tuning of the membership functions. We also intend to evaluate the performance of this scheme in wireless networks.

## VII. ACKNOWLEDGEMENT

The author gratefully acknowledges the support from the African Network of Science and Technological Institutions (ANSTI) towards his studies.

## VIII. REFERENCES

- [1] John Nagle “Congestion Control in IP/TCP Internetworks”. RFC 896, FACC, 6 January 1984.
- [2] Van Jacobson and Michael Karels. Congestion Avoidance and Control. *Proceedings ACM SIGCOMM '88*, 1988, pp. 314-29.
- [3] Van Jacobson, Robert Braden and Lixia Zhang. TCP Extensions for High-Speed Paths, *RFC 1185*, October 1990.
- [4] Sally Floyd and Tom Henderson. The New Reno modification to TCP’s fast recovery algorithm. *RFC 2582*, Experimental, April 1999.
- [5] Sally Floyd and Van Jacobson. Random early detection gateways for congestion avoidance. *IEEE/ACM Transactions in Networking*, 1(4):397-413, August 1993.
- [6] B. Braden *et al.* Recommendations on Active Queue Management and Congestion Avoidance in the Internet. IETF RFC2309, April 1998.
- [7] W. Feng *et al.*, "Blue: A New Class of Active Queue Management Algorithms," Technical Report CSE-TR-387-99, University of Michigan, 1999.
- [8] C. V. Hollot, V. Misra, D Towsley, W.B. Gong. A Control theoretic analysis of RED. *In Proceedings of*

*IEEE INFOCOM*, vol.3, pp. 1510-1519, Anchorage, Alaska, 2001.

- [9] S.H. Low, F. Paganini, J. Wang, S. Adlakha, and J.C. Doyle. Dynamics of TCP/AQM and a scalable control. *In Proc. of IEEE INFOCOM*, June 2002.
- [10] C. Chrysostomou *et al.* Fuzzy Logic Congestion Control in TCP/IP Best-Effort Networks, *2003 Australian Telecommunications Networks and Applications Conference (ATNAC 2003)*, Melbourne, Australia, 8 - 10 December 2003 (CD ROM - ISBN: 0-646-42229-4).
- [11] Li, R.J. and E.S. Lee (1987) Analysis of fuzzy queues. *Comput. Math Applications*, 17(1989), 1143-1147.
- [12] E.H. Mamdani, Application of fuzzy algorithm for control of simple dynamic plant. *Proc. IEEE* 121 (1974), p. 12.
- [13] Long Le *et al*, The Effects of Active Queue Management on Web Performance. *SIGCOMM'03*, August 25-29, Karlsruhe, Germany
- [14] Frank Akujobi *et al*, BECN for Congestion Control in TCP/IP Networks: Study and Comparative Evaluation , *Globecom 2002*
- [15] Frank Akujobi *et al* "Congestion control in TCP/IP networks: A combined ECN and BECN approach", *MILCOM 2003 - IEEE Military Communications Conference*, vol. 22, no. 1, Oct 2003 pp. 248-254
- [16] Ren Fengyuar *et al*, “Design of a fuzzy controller for active queue management” *Computer Communications* 25 (2002) 874-883

**Clement Nthambazale Nyirenda** received a B.Sc. degree in Electrical Engineering at the University of Malawi in 2000. From 2000 to August 2004, he worked as an Assistant Lecturer in Computer Engineering at the same University. He is currently researching in areas of Internet Congestion Control using Computational Intelligence techniques for his MSc degree at the University of KwaZulu Natal.