Fuzzy Green: A Modified TCP Equation-Based Active Queue Management Using Fuzzy Logic Approach

Saman Taghavi Zargar[†] *and Mohammad Hossein Yaghmaee*[††]

[†]Department of Computer Engineering, Ferdowsi University of Mashad, Iran

[††]Institute for Studies in Theoretical Physics and Mathematics (I.P.M)

Abstract

Congestion control provides quality of service over the best effort networks. Congestion occurred when arrival rate to a router is greater than its departure rate. In this paper, using fuzzy logic approach, we have proposed a modified TCP equation-based active queue management mechanism which is based on traditional GREEN algorithm. Here we present our fuzzy GREEN controller which is expected to act as a congestion controller in the routers. We have simulated our fuzzy method both with and without background Pareto traffic and the results show the superiority of our fuzzy method compared to the non-fuzzy one in both long-lived as well as short-lived connections.

Key words:
Active Queue Management, Congestion Control, Fuzzy GREEN Algorithm, Quality of Services.

Introduction

Quality of Service (QoS) in current best effort Internet still is a critical issue. The number of Internet users is rapidly increasing and therefore the amount of data to be carried also increases. Furthermore to support new Internet applications such as voice over IP, it is necessary to design effective Quality of Service approaches. Congestion control provides quality of service over the best effort networks. Congestion occurred when arrival rate to a router is greater than its departure rate. Each router in the network uses queue management and scheduling as two classes of algorithms that are related to congestion control.

The queue management algorithms try to control the length of packet queues by dropping packets when appropriate. Scheduling algorithms on the other hand, determine which packet to drop next and which is to send and also used to manage the allocation of bandwidth among flows. Network congestion induced by traffic leads to wasting all the resources that the packet consumed on its way from source to destination.

Active Queue Management (AQM) techniques such as RED [1], BLUE [2], AVQ [3], REM [4], and PI [5] try actively to detect and react to the congestion that would otherwise fill the queue and cause a burst of packet drops.

REM, PI, and AVQ are techniques to increase the link utilization at a router while maintaining small queue sizes but not providing fairness for each flow. RED and BLUE were designed to stabilize queue sizes at low levels; Flow Random Early Drop (FRED) [6] and Stochastic Fair BLUE (SFB) [7] improve on their performance by operating at the flow level. FRED and SFB try to enhance throughput-fairness between flows by sacrificing flows of higher bandwidth, while Apu Kapadia [8] showed that these approaches do not perform well with flows that have been widely varying Round-Trip Time (RTT).

2. Related Work

In contrast with FRED and SFB that avoid congestion by reacting to congestion early before it becomes too problematic, GREEN (Generalized Random Early Evasion Network) [9] is a proactive queue management algorithm which regulates TCP flows over the same link to a fair sending rate, and hence, prevents them from including congestion, thus providing more stable QoS over a best-effort network. GREEN applies knowledge of steady-state behavior of TCP connections at the router to intelligently drop or mark packet for congestion notification [9]. By using this mechanism, a router can give each connection its fair share of bandwidth while preventing the build-up of packet queues. GREEN calculates the drop probability (P) of each packet regarding to the number of active flows (N), bandwidth of a bottleneck link (L), maximum segment size (Mss), each packet's RTT, and $\gamma(t)$ coefficient as parameters and a constant c, in order to prevent congestion.

The constant c, depends on the acknowledgment strategy we are using (i.e., delayed or every packet). $\gamma(t)$ is an adaptation parameter results in high utilization of the link while providing high fairness between flows at time t. When a packet is received at the GREEN router, GREEN

first obtains the packet's RTT and *p*, as a drop probability, is then calculated. The RTT can be achieved either by reading the TCP header (i.e., requires TCP senders to embed their current RTT estimates within the TCP header) or be estimated by using IDMaps [10]. The adaption parameter $\gamma(t)$ would be calculated upon current utilization and queue drops at time *t*.
The procedure for the GREEN algorithm is shown below [8]:

Algorithm 1: GREEN algorithm

```
Enqueue(Packet pkt)
$RTT \leftarrow obtainRTT(pkt)$
$p \leftarrow \left(\frac{N \times Mss \times c}{\gamma(t) \times L \times RTT}\right)^2$
$u \leftarrow UniformRand(0,1)$
    if (u ≤ p) then
        drop(pkt)
    else
        addToQueue(pkt)
    end if
    if (currentTime() – $lastUpdate \geq window$) then
        update(currentUtil, N, queueDrops)
        $lastUpdate \leftarrow currentTime()$
        if ($queueDrops > 0$) then
            $\gamma \leftarrow 0.95\gamma$
        else if ($currentUtil < 0.98$) then
            $\gamma \leftarrow \frac{1 + currentUtil}{2 \times currentUtil}\gamma$
        end if
    end if
```

2.1 Estimating RTT for Flows

Excellent performance of traditional GREEN comes with a tradeoff – the router must be able to infer a flow's RTT. Feng et al. [9] presented preliminary results for a GREEN router where the RTT was assumed to be known at the router. There are several ways to observe flows at a router and determine their RTTs, but these approaches require the use of per flow state. GREEN must not use any per flow state approach to provide any benefit over Fair Queuing. Therefore, Apu Kapadia presented two approaches for estimating RTTs without the need of per flow state called Embedded RTTs and IDMaps [8].

2.1.1 Embedded RTTs

The first approach to achieve RTT requires TCP senders to embed their current RTT estimates within the TCP header. This value of RTT is the ideal value since exact drop probabilities for a given bandwidth can be calculated for a flow. This requires that the TCP sender be modified. There are two issues involved with Embedded RTTs. In order to gain more information on these issues refer to [8].

2.1.2 Estimating RTT by Using IDMaps

The IDMaps is a proposed scalable Internet-wide service that aims to provide internet distance estimates. For example, the authors have suggested that IDMaps can be used by hosts for nearest mirror selection. Such a service is also well suited to GREEN, which can obtain RTT estimates for flows using IDMaps. Apu Kapadia proposed an architecture where GREEN routers are part of the IDMaps framework, and therefore, can perform fast lookups in a local IDMaps database. He also mentioned that IDMaps was a theoretical service and GREEN could not rely on IDMaps for actual deployment [8].

2.2 Using Fuzzy Logic in Congestion Control

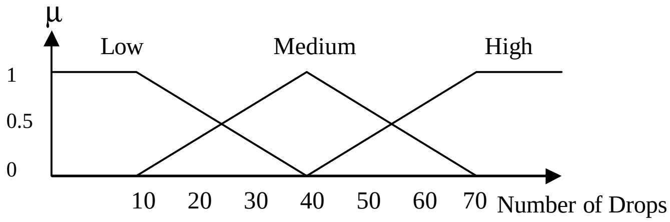
Fuzzy logic has been used in congestion control algorithms such as Fuzzy RED [11] and Fuzzy BLUE [12]. Each of which showed the superiority of fuzzy approach versus the non-fuzzy one. Our main contribution in this work is to provide a modified fuzzy GREEN algorithm which is more practical to deploy and maintains higher utilization and fairness, in the face of flows with widely varying RTTs and in the presence of short-lived connections, in contrast with the traditional GREEN.
The rest of this paper is organized as follows. In Section 3, the proposed fuzzy GREEN is presented. In section 4, describes our experimental setup and metrics for comparing the various queue management algorithms. In section 5, by using computer simulation, the performance of the proposed mechanism is compared with traditional GREEN algorithm. Section 6 intends to propose the future works and finally section 7, concludes the paper.

3. Proposed Fuzzy Approach

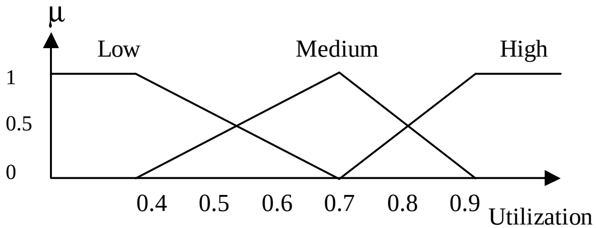
Apu Kapadia [8] proposed a simple negative feedback control algorithm mentioned in algorithm 1, to estimate the value of $\gamma(t)$ at time *t*, which scale (up or down) the sending rates of senders to adjust the overall link utilization and showed how GREEN performs extremely well by using this crisp value. We expected the fuzzy system to be appropriate for TCP active queue management technique, considering the uncertainty and adaptivity of the system. Our main goal in this work is to solve this problem by using concepts and techniques from *fuzzy logic theory* [14-15] as opposed to crisp set theory. Here we present our fuzzy GREEN controller which is expected to act as a congestion controller in the routers. The proposed controller uses a two-input-single-output fuzzy logic controller. The following steps are taken in order to design any fuzzy system [15].

3.1 Membership Functions and Linguistic Variables

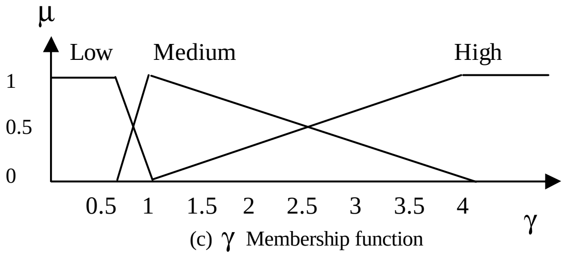
Definition of membership functions and linguistic variables are the first steps in fuzzy system designing. Each linguistic variable contains terms which are interpretation of technical figures. In our work we have used experimental triangular membership functions for coding and evaluation simplicity. The input linguistic variables are queue drops and current utilization and the output linguistic variable is the $\gamma(t)$ according to the time *t*. The membership functions of proposed fuzzy GREEN are plotted in Fig. 1.



(a) Queue drop's membership function



(b) Current utilization membership function



(c) γ Membership function

Fig. 1. Membership Functions

3.2 Fuzzy Rule Base

The second step in designing a fuzzy system is the creation of a fuzzy logic rule base which supplies the knowledge of the system [15]. To build the rule base, we need to present some standard methods. A fuzzy logic rule is an IF-THEN rule. The IF part is a fuzzy predicate defined in terms of linguistic values and fuzzy operators *Intersection* (*t-norm*) and *Union* (*s-norm*). The THEN part is called the *consequent*. In the proposed fuzzy system, we used product t-norm for aggregation [15]. The fuzzy rule base includes 9 experimental rules according to Gamma formula, which are shown in Fig. 2.

If Drop is low **and** Util is high **then** γ is high
If Drop is low **and** Util is medium **then** γ is high
If Drop is low **and** Util is low **then** γ is high
If Drop is medium **and** Util is high **then** γ is low
If Drop is medium **and** Util is medium **then** γ is medium
If Drop is medium **and** Util is low **then** γ is high
If Drop is high **and** Util is high **then** γ is low
If Drop is high **and** Util is medium **then** γ is low
If Drop is high **and** Util is low **then** γ is medium

Fig. 2. Fuzzy Rule Base

3.3 Defuzzification Method

The third step to design a fuzzy system is choosing an appropriate defuzzification method [15]. The objective of a defuzzification method is to derive the non-fuzzy (crisp) value that best represents the fuzzy value of the linguistic output variable. Center of area (CoA), center of maximum (CoM) and mean of maximum (MoM) are some of the defuzzification methods available. In the proposed fuzzy controller we decided to employ CoM for final γ assignment as the standard defuzzification is CoM which delivers the "best compromise" for the inference result.

$$\gamma = \frac{\sum_{i=1}^{n} \mu_i W_i}{\sum_{i=1}^{n} \mu_i}$$

, where μ_i is the T-norm of queue drops and utilization of the rule *i*, and W_i is the γ amount that results in μ_i.

3.4 Algorithm Design

Algorithm 2 shows our proposed method which consists of two major parts. In the first while loop the T-norm is calculated and then the crisp value W_i would be evaluated. The second loop generates CoM formula and finally calculates γ .

The procedure for the fuzzy GREEN algorithm is shown below:

Algorithm 2: Modified fuzzy GREEN algorithm

```
Enqueue(Packet pkt)
$RTT \leftarrow obtainRTT(pkt)$
$p \leftarrow \left( \frac{N \times Mss \times c}{\gamma(t) \times L \times RTT} \right)^2$
$u \leftarrow UniformRand(0,1)$
    if (u ≤ p) then
        drop(pkt)
    else
        addToQueue(pkt)
    end if
    if (currentTime() – lastUpdate ≥ window) then
        update(currentUtil, N, queueDrops)
        lastUpdate ← currentTime()
            i ← 1
            while (i<=9) do
                T-norm[i] ← μ(Q-Drop[i])*μ(Util[i])
                W[i] ← CrispValue(T-norm[i])
                i ← i + 1
            end while
            i ← 1
            while (i<=9) do
                Numerator = W[i] * T-norm[i]
                Denominator = T-norm[i]
            end while
            $\gamma$ ← Numerator / Denominator
    end if
```

4. Experimental Setup and Metrics

The proposed algorithm was implemented in C++ on NS2 [16] network simulator. Experiments were done under Linux SuSE 9.1 distribution operating system on a desktop computer with Pentium 4 CPU 2.5GHz. To compare the proposed mechanism with the traditional GREEN, we have simulated the following experiments in the same condition as Apu Kapadia did in his work [8] which based on two experiments. The first one does not include any background traffic to study fuzzy GREEN's behavior with long-lived connections, while the second one includes background Pareto traffic in order to study in case of short-lived connections such as web traffic.

In all experiments, we assume that a router knows the bandwidth (L) of the attached outgoing link. N is the number of active flows, i.e., flows that have had at least 1 packet to go through the router within a certain window of time. Since active flows not experiencing repeated timeouts send several packets per RTT, we use window = 1sec, which results in near-perfect estimates in our simulations. We leave more fine tuning of the window parameter for future work. We vary the number of foreground FTP flows from 50 to 400, and MSS equal to 1KB. We fix the value of constant c at 1.31, since we use a "random dropping, acknowledgment per packet" model according to [17]. We vary the RTT linearly from 72ms to 470ms. N sources and N sinks are connected to the routers over 10Mbps links. The bottleneck link has a bandwidth of 155Mbps and a delay of 30ms. We also set the buffer size in fuzzy GREEN router to be 600 packets. Fig. 3 shows the network topology used in the simulation.

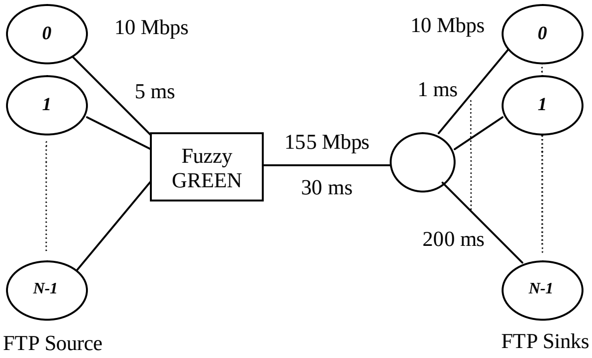


Fig. 3. Network topology

While the background traffic of each experiment varies, we start several FTP connections from the leftmost nodes to the rightmost nodes and run the simulation for 180 *sec* and evaluate the fairness between the FTP flows. We run the simulations for that long because unfairness arises from the long term effects of aggressive shorter RTT flows grabbing bandwidth from longer RTT flows. GREEN was implemented at the gateway, which is the bottleneck router in our simulation. All of the metrics such as link utilization, fairness, packet loss, average queue size, and bytes sent are measured at this gateway.

4.1 Fairness

We use Jain's Fairness Index [18] to access fuzzy GREEN's ability to maintain equal bandwidths between

TCP flows. The fairness always lies between 0 and 1 and thus a higher fairness index indicates better fairness between flows. Given a set of throughputs (x1,x2,…,xn), the fairness index is calculated as follows:

$$f(x_1, x_2, ..., x_n) = \frac{(\Sigma_{i=1}^{n} x_i)^2}{n\Sigma_{i=1}^{n} x_i^2}$$

4.2 Link Utilization

The overall link utilization we used in our simulation is computed using the following equation:

$$utilization = \frac{byte\_departures_t}{bandwidth \times t}$$

The numerator equals to the total number of bytes delivered by the link during the interval of *t sec*, and the dominator equals the total possible bytes that could have left the link in the same interval. The bandwidth is expressed as bytes/second.

4.3 Packet Loss

Both GREEN and fuzzy GREEN were compared with due regard to packet loss percentage. Since the overall packet loss stays below a few percent in our simulation, both GREEN and fuzzy GREEN are able to limit the rates of flows to their fair share of bandwidth.

4.4 Queue Size

The average queue size was calculated at the end of the simulation. Queue sizes are sampled at 20ms intervals. In fact, the more queue size an algorithm produces, the More buffer space it will need.

5. Performance Evaluation

We have simulated the following two experiments and measured the metrics that we have been mentioned above between GREEN and fuzzy GREEN.

5.1 Experiment 1

Fig. 4 to Fig. 8 show simulation results of the first experiment on GREEN and fuzzy GREEN for FTP flow without background traffic. Fig. 4 indicates that fuzzy GREEN behaves the same as GREEN in sending bytes per flows which means all flows send almost the same amount of data. At the end of our simulation with 200 FTP flows, we plot the number of packets sent by each flow. We observe that both GREEN and fuzzy GREEN have corrected TCP's inherent bias against larger RTT, and all flows achieve roughly the same average bandwidths at the end of the simulation.

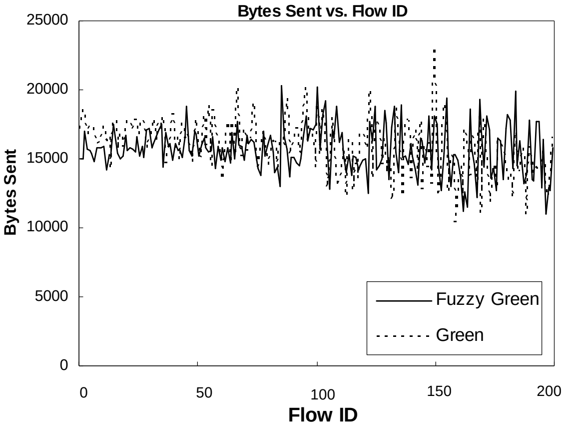


Fig. 4. Bytes Sent vs. Flow ID (RTT increase as Flow ID increases)

Fig. 5 shows that fuzzy GREEN provides significantly better bandwidth fairness than traditional one. Better fairness causes flows to use bandwidth fairly the same.

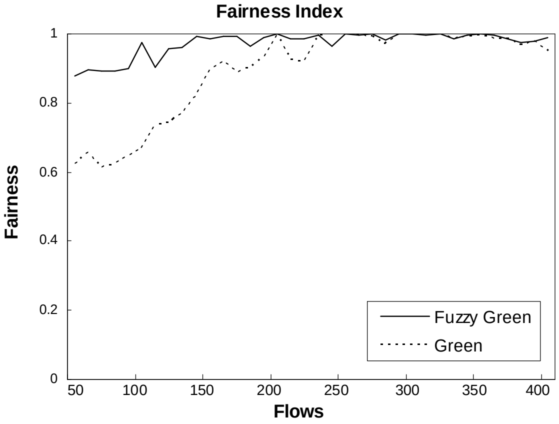


Fig. 5. Jain's Fairness Index vs. Number of Flows

Fig. 6 indicates that packet loss in fuzzy GREEN is mostly less than GREEN and always less than 2%, resulting in fewer overflows at the queue.

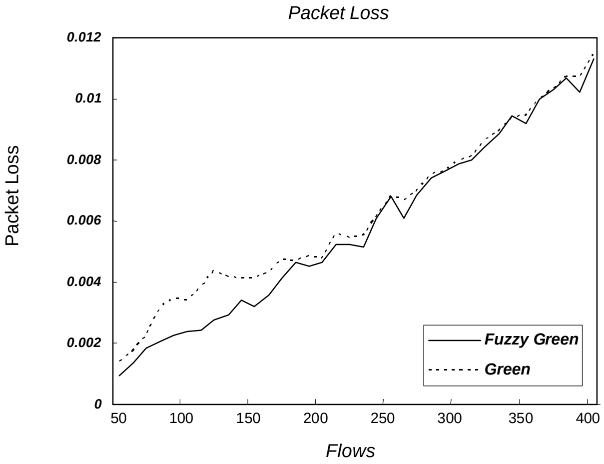


Fig. 6. Overall Packet Loss vs. Number of Flows

Fig. 7 shows that considering growth of flows, utilization of both approaches would remain over 85% and in high flow amount it would converge to 98%.

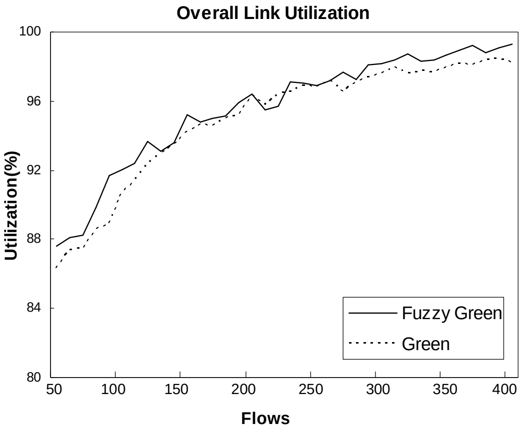


Fig. 7. Overall Link Utilization vs. Number of Flows

Fig. 8 indicates that queue buffer size is even less than traditional GREEN. As the number of flows increases (and hence the contention for bandwidth), fuzzy GREEN is able to more effectively control queue sizes.

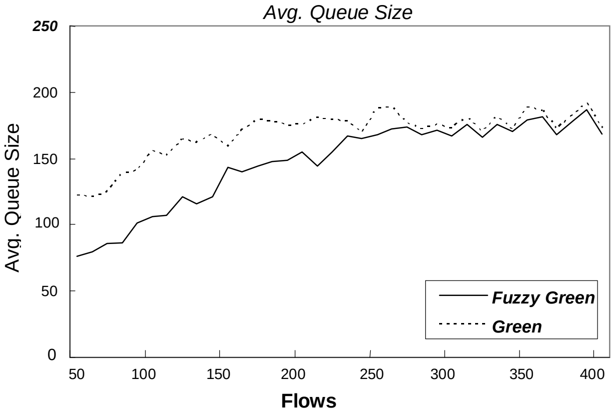


Fig. 8. Average Queue Size vs. Number of Flows

5.2 Experiment 2

In the second experiment, we used Pareto traffic to simulate the short lived connections. We pair each FTP flows with a Pareto flow with the same link characteristics. This is essentially two copies of the topology shown in Fig. 3, sharing the same bottleneck link and fuzzy GREEN router. One copy runs the FTP flows discussed in the previous experiment, and other copy runs Pareto flows with the following parameters: packet size: 1000bytes, burst time: 2sec, idle time: 5sec, and rate: 160Kbps.

Results in Fig. 9 to Fig. 13 show that fuzzy GREEN behaves as well as the previous experiment even in facing traffic background. Fig. 9 compares sending bytes per flows. Indicates that fuzzy GREEN behaves the same as GREEN in sending bytes per flows which means all flows send almost the same amount of data.

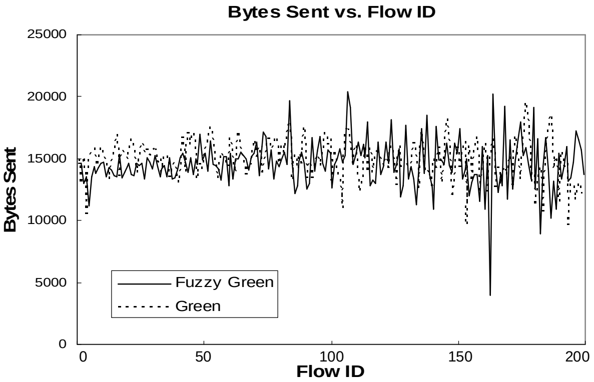


Fig. 9. Bytes Sent vs. Flow ID (RTT increase as Flow ID increases)

Fig. 10 shows superiority of our fuzzy method compared to the non-fuzzy one in fairness especially for less than 200 flows.

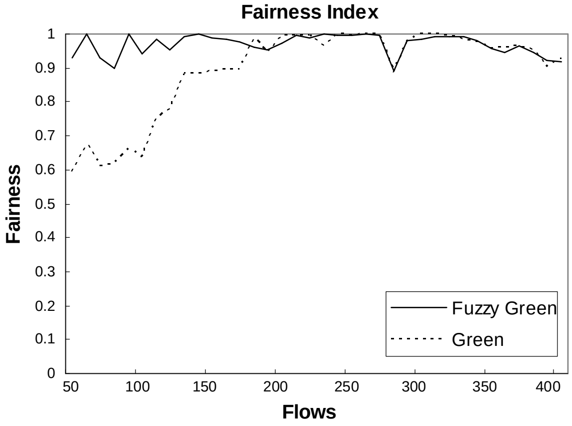


Fig. 10. Jain’s Fairness Index vs. Number of Flows

Fig. 11, like the previous experiment, indicates that packet loss in fuzzy GREEN is mostly less than GREEN and always less than 2%.

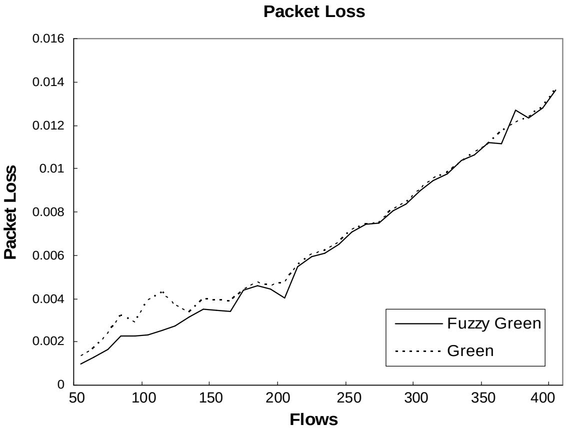


Fig. 11. Overall Packet Loss vs. Number of Flows

Fig. 12 shows that considering growth of flows, utilization of both approaches would remain over 88% and in high flow amount it would converge to 98%.

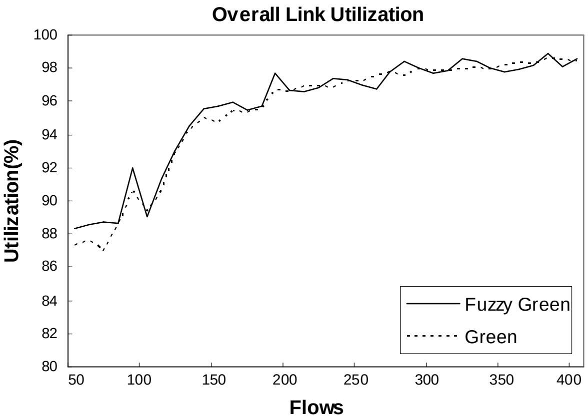


Fig. 12. Overall Link Utilization vs. Number of Flows

Fig. 13 shows that the queue buffer size is remarkably less than traditional GREEN. As the number of flows increases, fuzzy GREEN provides superior average queue size compared to traditional GREEN and keeps the average queue sizes low.

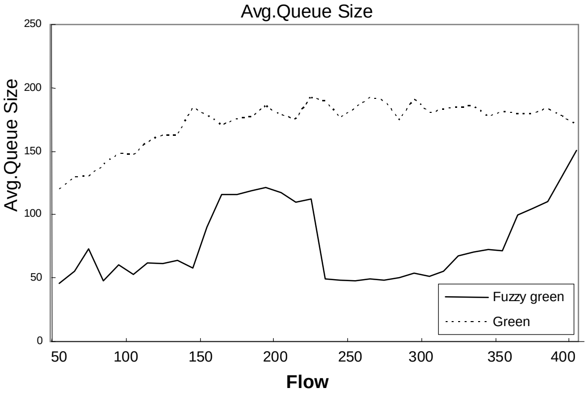


Fig. 13. Average Queue Size vs. Number of Flows

6. Future Works

More simulations are under process to reach better tuned fuzzy approach. We are also intends to design a more flexible controller by applying fuzzy approach not only on Gamma parameter, but also on drop probability and finally substitute all experimental formulas in the traditional GREEN algorithm with fuzzy rules. We are also planning to use genetic algorithm to best tune our membership functions and we will prove that fuzzy GREEN works better with tuned membership functions.

7. Conclusion

In this paper, we have devised a modified TCP equation-based active queue management using fuzzy logic approach based on traditional GREEN algorithm, and compared the efficiency of both fuzzy and non-fuzzy methods. Two experiments were done using ns2 network simulator and the outcome clearly showed the superiority of the fuzzy GREEN method versus the traditional GREEN one under simulation of bytes sent, fairness, packet loss, average queue size and link utilization.
Furthermore, unlike other stateless approaches that attempt to approximate Fair Queuing, our approach and also traditional GREEN [8] do not require coordination between routers, or significant modification to the TCP implementation. Currently routers do not apply fuzzy logic to control the congestion, however according to previous works on Fuzzy RED and Fuzzy BLUE and also our fuzzy GREEN; it is recommended router designers apply this method as a better congestion control algorithm.

References

[1] S.Floyd, V. Jacobson, “Random early detection gateways for congestion avoidance”, IEEE/ACM Transactions on Networking, 1(4):397-413, Aug 1993.

[2] W. Feng, et.al. “Blue: a new class of active queue management algorithm”, technical report, Apr 1999.

[3] S. Kunniyu, R. Srikant, “Analysis and design of an adaptive virtual queue (AVQ) algorithm for active queue management” ACM SIGCOMM, 2001.

[4] S. Athuraliya, et.al. “Rem: active queue management”, IEEE Network, June, 2001.

[5] C. V. Hollot, et.al., “On designing improved controllers for AQM routers supporting TCP flows”, In Proc. of IEEE INFOCOM, 2001.

[6] D. Lin, R. Morris, “Dynamics of random early detection” In Proc. Of ACM SIGCOMM, Sept 1997.

[7] W. Feng, et.al., ”Stochastic fair blue: a queue management algorithm for enforcing fairness”, In Proc. of IEEE INFOCOM, Apr 2001.

[8] Kapadia, W. Feng, R. H. Campbell, "Green: a TCP equation-based approach to active queue management", UIUC Technical Report: UIUCDCS-R-2004-2408/UILU-ENG-2004-1710, February 2004.

[9] W. Feng, A. Kapadia, S. Thulasidasan, "Green: proactive queue management over a best-effort network," In Proc. of IEEE Global Telecommunications Conference (GLOBECOM 2002).

[10] P. Francis, et.al., “Idmaps: a global internet host distance estimating service”. IEEE/ACM Transactions on Networking, Oct 2001.

[11] M. H. Yaghmaee, “A modified random early detection algorithm: fuzzy logic based approach”, JCN, Journal of Communication Network, vol.7, No. 3, September 2005, pp. 337-352.

[12] M. H. Yaghmaee, M. Menhaj, H. Amintoosi, “A fuzzy extension to the blue active queue management algorithm”, IAEEE, Journal of Iranian Association of Electrical and Electronics Engineers, vol. 1, No. 3, Winter 2005, pp. 3-14.

[13] Zadeh, L.A., “Outline of a new approach to the analysis of complex systems and decision processes,” IEEE Transaction Systems Man. Cybern, vol SMC-3, no. 1, pp. 28-44, 1973.

[14] Zadeh, L.A., “Fuzzy sets", Information and Control, 1965. 338-353

[15] Wang, L., A course in fuzzy systems and control, Prentice-Hall,1997.

[16] Ns-2. Network Simulator. http://www.isi.edu/nsnam/ns.

[17] M. Mathis, J. Semke, J. Mahdavi, and T. Ott., "The macroscopic behavior of the TCP congestion avoidance algorithm", Computer Communication Review, 27(3), July 1997.

[18] D. Chiu, R. Jain, “Analysis of the increase and decrease algorithms for congestion avoidance in computer networks”, Computer Networks and ISDN systems, 17:1-14, 1989.

Saman Taghavi Zargar was born on March 1980 in Mashad, Iran. He received his B.S. degree in Computer Engineering from Azad University of Mashad in 2004 and he is currently the M.S. degree student of Computer Software in Ferdowsi University of Mashad. He is an IEEE student member and also member of IEEE Computer society and International Association of Engineers (IAENG). During 2005-2006, he stayed in Computer and Communication Research Center, Ministry of Posts and Telecommunications of Iran to work on his M.S. project.

Mohammad Hossein Yaghmaee was born on July 1971 in Mashad, Iran. He received his B.S. degree in Communication Engineering from Sharif University of Technology, Tehran, Iran in 1993, and M.S. degree in communication engineering from Tehran Polytechnic (Amirkabir) University of Technology in 1995. He received his PhD degree in communication engineering from Tehran Polytechnic (Amirkabir) University of Technology in 2000. He has been a computer network engineer with several networking projects in Iran Telecommunication Research Center (ITRC) since 1992.

November 1998 to july1999, he was with Network Technology Group (NTG), C&C Media research labs., NEC Corporation, Tokyo, Japan, as visiting research scholar. He is author of one book: Computer Networks and Internet, in Persian, (Ferdowsi University of Mashad Publishing, Mashad, 2003). His research interests are in traffic and congestion control, high speed networks including ATM and MPLS, Quality of Services (QoS) and fuzzy logic control.