FDynamicAutoRED: An Algorithm to Stabilize the Queue in Internet Routers

K.Chitra
D.J.Academy for Managerial Excellence
Coimbatore, Tamil Nadu, India

Dr.G.Padmavathi
Avinashilingam University for Women
Coimbatore, Tamil Nadu, India

ABSTRACT
Internet Routers face the problem of congestion due to the increased use of Internet. Active Queue Management algorithm is a solution to the problem of congestion control in the Internet routers. As data traffic is bursty in routers, burstiness must be handled without comprising the high link utilization and low queuing delay. Congested link causes many problems such as large delay, unfairness among flows, underutilization of the link and packet drops in burst. RED based AQMS use only queue length as congestion indicator to indicate congestion. An AQM scheme is proposed that considers the advantages of this Queue length based AQMs and uses the flow information to satisfy the QOS requirements of the network. This proposed scheme aims to provide good service even under unresponsive load, offers stabilised queue with reduced queue oscillation and controlled packet drop rate.

General Terms
Packet Switched Networks, Congestion Control et. al.

Keywords
Packet Drop Probability, Fairness, Average Queue Size

1. INTRODUCTION
Research activities have become an on going process with the origin of various congestion avoidance mechanisms in Internet to improve the performance of Internet traffic. The introduction of these mechanisms has indicated the inefficiency of each of the AQMs in heavy traffic network. The various existing AQMs detect congestion based on different factors and calculate the packet dropping probability. Based on the various factors used in AQMs, the degree of congestion varies and its performance also varies.

Floyd et al proposed the first RED [1] AQM in 1993 with the objective of preventing congestion with reduced packet loss. This AQM alleviates congestion by detecting incipient congestion early and delivering congestion notification to the source to reduce its transmission rates avoiding overflow from occurring. But the appropriate selection of the RED parameters defines the success of RED. So RED AQM faces a major drawback implemented in the network link. Incase of heavy traffic, RED AQM also leads to global synchronization, lock-out problem and unstable queue size if parameters not properly tuned. In order to overcome parameter tuning problem in RED, AdaptiveRED was proposed. It adaptively tuned the values of max_p to keep the target queue length within a target range between min_{th} and max_{th}. The AQM automatically set w_q based on the link speed, and max_p in response to measured queue length. This reduced the RED's parameter sensitivity. Further in improving RED and AdaptiveRED AQMs, AutoRED technique was implemented in them. The AutoRED technique uses the concept of dynamic w_q which varies based on multiple characteristics of the network.

The queue-based AQM schemes use average queue-length or instantaneous queue length as a congestion indicator to calculate packet drop probability. The YELLOW AQM [2] proves that the packet drop probability just does not depend only on the queue length rather can be calculated using the congestion indicator like input rate that helps in identifying the real congestion in the queue. In case of the rate-based AQM AVQ, it maintains a virtual queue whose capacity is less than the actual capacity of the link. However it is difficult to achieve faster response time and high link utilisation using a constant value γ. In improving this method for setting the value for γ, SAVQ [3] is proposed. SAVQ stabilizes the dynamics of queue maintaining high link utilization.

In REM, both queue length and load is used as congestion indicators. The BLUE algorithm resolves some of the problems of RED by employing two factors: packet loss from queue congestion and link utilization. It maintains a single probability p_m to mark or drop packets. If the buffer overflows, BLUE increases p_m to increase the congestion notification and is decreased to reduce the congestion notification rate in case of buffer emptiness. This scheme uses link history to control the congestion. In case of large queue, RED has continuous packet loss followed by lower load that leads to reduced link utilization.

Another possibility is that the heavy load tends to vary in an Internet router resulting in a queue oscillation. Router must take care of the above problem. The buffer in the routers is to be used effectively by using an efficient Active Queue Management Mechanisms. Active Queue Management prevents congestion and provides quality of service to all users. A router implementing RED AQM maintains a single queue that drops an arriving packet at random during periods of congestion. RED suffers from lockout and global synchronization problems when parameters are not tuned properly.

Adpative RED [4], AutoRED with RED [5] tries to improve the parameter tuning problem in RED. While AQMS like PD-RED [6], MRED [7], DS-RED [8] tries to improve the performance compared to RED. The problem of unfairness and Queue oscillation still existed in these AQMs.

RED based AQMs use only Queue Length as congestion metircs to detect congestion. Some AQMs techniques were introduced that used Input Rate as congestion indicators besides using average queue size. REM AQM [9] was proposed that used both congestion indicators queue size and input rate to detect congestion where as BLUE [10] uses link history and packet loss as congestion indicator to compute the packet drop probability.

The objective of this paper is to propose an algorithm that reduces queue oscillation in Queue based algorithm. The proposed AQM is implemented in AutoREDwithRED of its simplicity, easy implementation and to bring in the advantages of Queue based algorithm. The proposed AQM is implemented in this to bring in advantages of Queue based algorithm and to remove the parameter tuning problem. This algorithm is simple to implement, stabilises the queue size and improves the performance of the routers. The rest of the paper is organized as follows: Section 2 explains the background study that includes the various AQM algorithms and its drawback. In Section 3, the concepts regarding the proposed algorithm are discussed. In section 4, the simulation is carried out with discussion of results. Our conclusions are presented in section 5.

2. BACKGROUND

Research activities have become an on going process with the origin of various congestion avoidance mechanisms in Internet to improve the performance of Internet traffic. The introduction of these mechanisms has indicated the inefficiency of each of the AQMs in heavy traffic network. The various existing AQMs detect congestion based on different factors and calculate the packet dropping probability. Based on the various factors used in AQMs, the degree of congestion varies and its performance also varies.

Floyd et al proposed the first RED AQM in 1993 with the objective of preventing congestion with reduced packet loss. This AQM alleviates congestion by detecting incipient congestion early and delivering congestion notification to the source to reduce its transmission rates avoiding overflow from occurring. But the appropriate selection of the RED parameters defines the success of RED. So RED AQM faces a major drawback implemented in the network link. Incase of heavy traffic, RED AQM also leads to global synchronization, lock-out problem and unstable queue size if parameters not properly tuned. In order to overcome parameter tuning problem in RED, AdaptiveRED was proposed. It adaptively tuned the values of max_p to keep the target queue length within a target range between min_{th} and max_{th}. The AQM automatically set w_q based on the link speed, and max_p in response to measured queue length. This reduced the RED's parameter sensitivity. Further in improving RED and AdaptiveRED AQMs, AutoRED technique was implemented in them. The AutoRED technique uses the concept of dynamic w_q which varies based on multiple characteristics of the network.

The queue-based AQM schemes use average queue-length or instantaneous queue length as a congestion indicator to calculate packet drop probability. The YELLOW AQM proves that the packet drop probability just does not depend only on the queue length rather can be calculated using the congestion indicator like input rate that helps in identifying the real congestion in the queue. In case of the rate-based AQM AVQ [11], it maintains a virtual queue whose capacity is less than the actual capacity of the link. However it is difficult to achieve faster response time and high link utilisation using a constant value γ. In improving this method for setting the value for γ, SAVQ is proposed. SAVQ stabilizes the dynamics of queue maintaining high link utilization.

In REM, both queue length and load is used as congestion indicators. The BLUE algorithm resolves some of the problems of RED by employing two factors: packet loss from queue congestion and link utilization. It maintains a single probability p_m to mark or drop packets. If the buffer overflows, BLUE increases p_m to increase the congestion notification and is decreased to reduce the congestion notification rate in case of buffer emptiness. This scheme uses link history to control the congestion.

SRED in [12] pre-emptively discards packets with a load-dependent probability when a buffer in a router is congested. It stabilizes its buffer occupancy at a level independent of the number of the active connections. SRED does this by estimating the number of active connections. It obtains the estimate without collecting or analysing state information. SRED keeps the buffer occupancy close to a specific target and away from overflow or underflow. In SRED the buffer occupancy is independent of the number of connections while in RED the buffer occupancy increases with the number of connections. The hit mechanism is used to identify misbehaving flows without keeping per-flow state. Stabilised RED overcomes the scalability problem but suffers from low throughput. GREEN [13] algorithm uses flow parameters and the knowledge of TCP end-host behavior to intelligently mark packets to prevent queue build up, and prevent congestion from occurring. It offers a high utilization and a low packet loss. An improvement of this algorithm is that there are no parameters that need to be tuned to achieve optimal performance in a given scenario. In this algorithm, both the number of flows and the Round Trip Time of each flow are taken into consideration to calculate the congestion-notification probabilities. The marking probability in GREEN is generally different for each flow because it depends on characteristics that are flow specific.

3. PROPOSED ALGORITHM

The proposed algorithm is motivated by the need for a stable operating point for the queue size and fair bandwidth allocation irrespective of the dynamic traffic and congestion characteristics of the n flows. The unstable queue size results in high queue oscillation due to the parameter tuning problem in queue based AQMs. Motivation is to identify a scheme that penalizes the unresponsive flows with the stable queue size with controlled packet drop rate.

The proposed algorithm - FDynamicAutoRED enforces the concept of queue-based and uses the flow information. It is desirable for AQM schemes to act without storing a lot of information otherwise it becomes an overhead and non-scalable. An algorithm is proposed that modifies the Queue based AutoREDwithRED algorithm to penalize the unresponsive flows. As in Fig. 1, this algorithm calculates the average queue size of the buffer for every packet arrival. It also indicates two thresholds on the buffer, a minimum threshold min_{th} and a maximum threshold max_{th}. Average queue size Q_{ave} is compared with these thresholds for every arriving packet.

If average queue size is less than min_{th}, every arriving packet is queued. If average queue size is greater than max_{th}, every arriving packet is dropped. This results in queue size below max_{th}. When the average queue size is greater than min_{th}, every arriving packet is compared with a randomly selected packet from the queue for their flow id. If they have the same flow id, both are dropped otherwise if average queue size is less than

max_{th} then the packet is dropped with the probability otherwise the new packet is dropped.

In this proposed algorithm, the arriving packet is dropped with a probability depending on the average queue size. This algorithm uses AutoRED technique where w_q is dynamic in nature compared to a constant w_q in RED. To achieve good throughput and reasonable average queue length with RED based algorithm requires careful tuning of w_q. Incase if too small a value of w_q, performance is in terms of queuing delay and too large a value leads to decreased throughput. The dynamic value of w_q adapts itself to the varying nature of the congestion and traffic.

```
Initially I_maxth= Cur_maxth = max_th
For every packet arrival {
Calculate P_t
Calculate w_q
If (P_t < 0.550) && (Q_t < I_maxth) && (Cur_maxth > I_maxth)
        Reinitialise Cur_maxth to I_maxth
Else if (P_t > 0.550) && (P_t < 0.880) && (Q_t < I_maxthresh)
&& (Cur_maxthresh> I_maxthresh)
                Increment Cur_maxth
Calculate Q_ave
if (Q_ave < min_th)
    Forward the new packet
Else
    Select randomly a packet from the queue for their
    flow id
    Compare arriving packet with a randomly selected
    packet.
    If they have the same flow id
                Drop both the packets
    Else
        if (Q_ave <= max_th)
                Calculate the dropping probability p_a
                Drop the packet with probability p_a
        Else
                Drop the new packet
}
Variables:
Q_ave       : average queue size
p_a         :current packet-marking probability
Q_t         : Instantaneous queue size at time t
p_b         : temporary marking or dropping probability
w_q         : queue weight
max_th      : maximum threshold for queue
P_t         : Probability that the network can lead to
congestion at time t
max_p       :maximum dropping probability
Fixed parameters:
min_th      : minimum threshold for queue
```

Figure 1 Pseudocode of FDynamicAutoRED

w_q is redefined as in [5].and results in reduced instantaneous queue oscillation. The definition of weighting parameter w_q is written as a product of three characteristics as follows:

$$T_t = p_t(1-p_t)$$

$$J_t = \frac{2(5.923+(Q_t - Q_{avg,t-1}))}{\ln(5.923+(Q_t - Q_{avg,t-1}))}$$

$$K_t = \frac{1}{bs}$$

The first characteristic T_t represents the dynamic status of the congestion in the network. It signifies the probability with which the system can lead to congestion with the information available at time t and is a time dependent function.

$$w_{q,t} = p_t(1-p_t) * \frac{2(5.923+(Q_t - Q_{avg,t}))}{\ln(5.923+(Q_t - Q_{avg,t-1}))} * \frac{1}{bs}$$

where

$w_{q,t}$ = Newly defined time dependent weighing function
Q_t =Instantaneous queue size at time t
$Q_{avg,t-1}$ = Average queue size at time t-1
$Q_{avg,t}$ = Average queue size at time t
P_t =Probability that the network can lead to congestion at time t
bs = Buffer size

The second characteristic J_t projects the current status of traffic in the network at time t and it is also a time dependent function.

The third characteristic K_t is time independent parameter and it allows normalization of instantaneous queue size changes with respect to the buffer size. Therefore these three characteristics are used to incorporate the dynamic changes in the congestion and traffic in the calculation of average queue size.

In this proposed AQM, max_{th} of the queue size is not constant and it varies depending on the congestion in a traffic. Based on the level of congestion of traffic in a network at a particular instant time, max_{th} of the queue is either incremented or decremented. As the probability of packet drop depends also on max_{th}, the value of max_{th} is kept dynamically varying based on the congestion level.

The level of congestion varies and is indicated depending on the following criteria:

- P_t (Probability of congestion)
- Q_t (Instantaneous queue size) and I_{maxth} (Initialmaxthresh)
- Cur_{maxth} (Currentmaxthresh) and I_{maxth} Initialmaxthresh)

If (P_t < 0.550) and (Q_t < I_{maxth}) and (Cur_{maxth} > I_{maxth}), then Reinitialise Cur_{maxth} to I_{maxth}. This indicates a low congestion or no congestion in the network so Cur_{maxth} can be reinitialized. The value of max_{th} will not affect the probability of packet drop in case of no congestion.

If (P_t between 0.550 and 0.880) and (Q_t >= I_{maxth}) and (Currentmax_{th} == Initialmax_{th}), then Increment Cur_{maxth}. This indicates initial stages of heavy congestion in the network so Cur_{maxth} is incremented to control the probability of packet drop. A higher value of max_{th} in the initial stages of heavy congestion will result in reduced probability of packet drop.

Incase of very heavy congestion, max_{th} will not affect the probability of packet drop because any value of max_{th} will lead

to packet drop. The value of max_{th} will be vital only in case of initial stages of heavy congestion and not during no congestion or very heavy congestion. The packet drop probability will be controlled based on the value of max_{th}. In this proposed algorithm, max_{th} is set based on the probability of congestion at time t in network and the occurrence of congestion at time (t-1).

So calculating average queue size and packet drop probability are not dependent on well tuning of the parameters as in RED. This algorithm will work fine as the parameters are well tuned automatically and parameterized. In such a case, the proposed algorithm reduces the problem of parameter tuning.

To calculate p_a, p_b:

$$p_b = (avg\text{-}min_{th}) / (max_{th} - min_{th})$$

$$p_a = p_b / (1 - count \cdot p_b)$$

Though the dynamic varying nature of w_q takes care of the network characteristics it keeps the average queue length high and in a unstable point in case of heavy traffic. This algorithm overcomes this problem with the help of the flow based information. So both w_q and the flow information take care of the unresponsive flows and misbehaving flows and brings in stable and fair queuing. This is implemented by simple comparison of the packet from incoming traffic and the packets in the queue. Thus in a simple manner the packets of the misbehaving flows are penalized. This is done as packets belonging to non-adaptive or misbehaving flows are more likely to be chosen for comparisons than the adaptive flows. Packets of unresponsive flows are dropped more often than the adaptive flows and well behaved flows. The value of max_{th} decides on the packet drop. The threshold value of max_{th} should be set sufficiently to increase the network power. In case of low threshold value of max_{th}, the average queue size will remain too low. If the average queue size is too low, then the output link will be underutilized. When the network traffic is bursty then average queue size can also be made bursty to improve link utilization. Nextly increasing $max_{th} - min_{th}$ sufficiently large avoids global synchronization. If the $max_{th} - min_{th}$ is too small, then the computed average queue size will be regularly around max_{th}. The dynamic varying nature of the parameters w_q, max_{th} and the flow information will try to keep the router congestion controlled

4. EXPERIMENTATION

In this section, the packet-simulator ns-2 is used to simulate the FDynamicAutoRED algorithm. In this simulation the network topology in Figure 2 is with a single link of capacity 1Mbps that drops packet according to the AQM algorithm. The congestion link is in between the two routers R1 and R2. The link is shared by n TCP flows and n UDP flows. End hosts are connected to the routers using a 10Mbps link. All links have a small propagation delay of 1ms so that the delay introduced is by the buffer delay rather than the transmission delay. The maximum window size of TCP is set to 300 such that it is not a limiting factor for the flow's throughput. The TCP flows are derived from FTP sessions which transmit large size files. The UDP hosts send packets at a constant bit rate of 0.08 Mbps. In the simulation setup 32 TCP flows and 1 UDP flow is considered in the network.

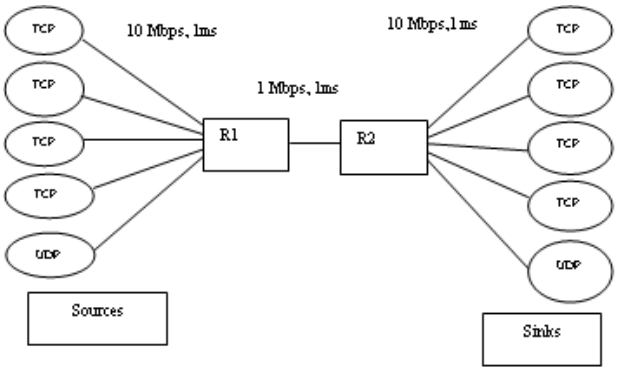

Figure 2 Network Topology

The minimum threshold min_{th} in the FDynamicAutoRED scheme is set to 100 and the maximum threshold max_{th} to be twice the min_{th} and the physical queue size is fixed at 300 packets.

In a dynamic varying mixture of traffic, the control parameter w_q alone does not help in achieving the stable operating point for the queue size. As shown in Figure 4 dynamic varying parameter w_q maintains the average queue size and instability at a higher level in case of AutoREDwithRED. Though AutoREDwithRED keeps the average queue size at a stable point for the traffic consisting of adaptive flows, but for different conditions that included non-adaptive flows the average queue size projects an oscillating behaviour. But this algorithm shows a stable and a moderate average queue size compared to other AQMs as in Figure 3. The average queue size is neither too low nor high in this proposed AQM as compared to other AQMs. A very high average queue size increases the queuing delay. The average queue size should not be too low which results in poor link utilization. This proposed algorithm keeps the average queue size controlled at a moderate level compared to other AQMs.

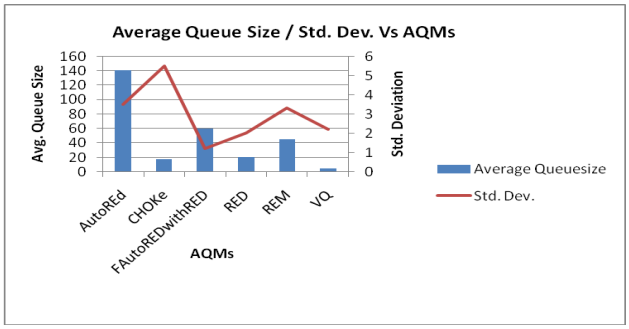

Figure 3 Average Queue Size / Std. Dev of other AQMs with FDynamicAutoRED

RED and other AQMs are unable to penalize unresponsive flows. As the packets dropped from each flow over a period of time is almost the same. Consequently the misbehaving traffic like UDP can take up a large % of the link bandwidth and starve out TCP friendly flows as in Figure 4**.** FDynamicAutoRED identifies and penalizes misbehaving flows effectively compared to the existing AQMs as in Table 1.

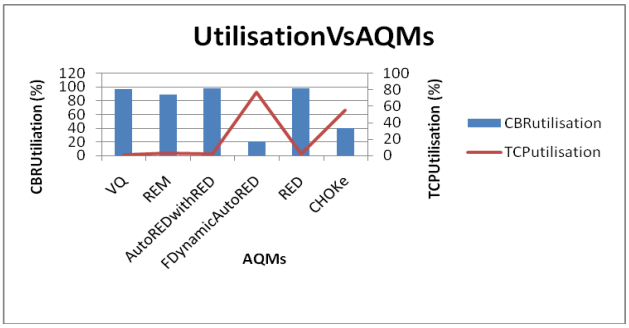

Figure 4 CBR and TCP Utilisation of other AQMs with FDynamicAutoRED

The Figure 5 and Table 2 indicates that other AQMs almost take up the entire bandwidth for UDP flow though its actual CBR fair share is very minimum. While CBR throughput is only 20% of the bandwidth using this algorithm. So TCP utilization is almost very minimum in case of other AQMs and when compared to this algorithm shows a good fair utilization of 76%.

In case of packet drop rate shown in Table 3 and Figure 6, the FDynamicAutoRED has a controlled drop compared to the other AQMs due to the drop of the UDP. It tries to give a fair share by

Table 1 CBR and TCP Utilization of other AQMs with FDynamicAutoRED

	In %	
	CBR utilization	**TCP utilization**
VQ	97.053714	0.646857
REM	88.578286	2.395429
AutoREDwithRED	98.299429	1.702857
FDynamicAutoRED	20.254857	76.78514
RED	98.166857	1.830857

Table 2 Comparison of CBR Fair share and Throughput of other AQMs with FDynamicAutoRED

	In Mbps	
	CBRFair share	**CBR Throughput**
VQ	0.03030303	0.970537
REM	0.03030303	0.885783
AutoREDwithRED	0.03030303	0.982994
FDynamicAutoRED	0.03030303	0.202548
RED	0.03030303	0.981669

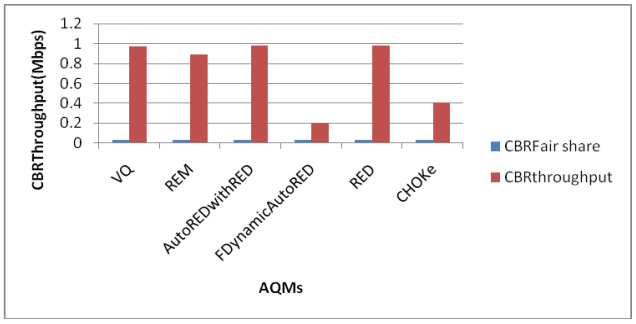

Figure 5 Comparison of CBR Fair share and Throughput of other AQMs with FDynamicAutoRED

dropping the UDP packets otherwise the UDP utilises the link to the maximum without allowing the TCP packets. The packet drop rate is controlled due to the increment/decrement in the max_{th} that depends on the congestion characteristics. The queue oscillation in queues does not prove as good performance of AQMs in Internet routers. But this proposed AQM proves good as it gives reduced queue oscillation. The queue stability of this proposed algorithm is high compared to other AQMs.

Table 3 Comparison of Packet Drop Rate of other AQMs with FDynamicAutoRED

AQMs	**Packet Drop Rate**
FDynamicAutoRED	9.49
RED	9.87
AutoREDwithRED	9.77
REM	9.74
VQ	9.79

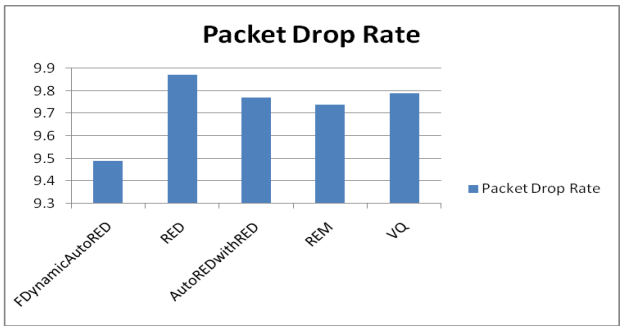

Figure 6 Comparison of Packet Drop Rate of other AQMs with FDynamicAutoRED

5. CONCLUSIONS

This paper proposes an AQM scheme called FDynamicAutoRED which aims to reduce queue oscillation with controlled packet drop rate in case of mixture of traffic. The adaptive flows are protected from non-adaptive flows in this scheme. It is obtained without comprising high utilisation, low queuing delay and controlled packet loss. It is achieved with well dynamically tuned parameters and flow information. This packet dropping scheme discriminates against the unresponsive flows resulting in fair congestion indication. This proposed AQM scheme inherits the advantages of these queue length based AQM and uses flow information to satisfy the QOS requirements of the network.

6. REFERENCES

[1] Floyd, S. and Jacobson, V. 1993. Random early detection gateways for congestion avoidance. IEEE/ACM Trans. Networking, vol. 1, pp. 397–413, Aug. 1993.

[2] Chengnian, L. Zhao, Guan, B. Yang, J. 2004. The Yellow active queue management algorithm. Computer Networks, November 2004

[3] Cheng-Nian, L. Zhao, B. Guan, X. 2005. SAVQ: Stabilized Adaptive Virtual Queue Management Algorithm. IEEE Communications Letters. January 2005

[4] Floyd, S. Gummadi, S. Shenkar, S. and ICSI. Adaptive RED: An algorithm for Increasing the robustness of RED's active Queue Management. Berkely,CA [online] http:www.icir.org/floyd/red.html

[5] Suthaharan, S. 2007. Reduction of queue oscillation in the next generation Internet routers. Science Direct, Computer Communication. 2007

[6] Jinsheng, S. King-Tim, K. Guanrong, C. Sukerman, S. M. S. 2003. PD – RED: To Improve Performance of RED. IEEE COMMUNICATIONS LETTER. August 2003

[7] Jahoon, K. Byunghun, S. Kwangsue, C. Hyukjoon, L. Hyunkook, K. 2001. MRED: A New Approach To Random Early Detection. In 15th International Conference on Information Networking. February 2001.

[8] Bing, Z. Mogammed, A. 2000 DSRED: An Active Queue Management Scheme for Next Generation Networks. In Proceedings of 25th IEEE conference on Local Computer Networks LCN 2000. November 2000

[9] Athuraliya. S. Li, V. H. Low, S. H. and Yin, Q. 2001. REM: Active queue management. IEEE Network Mag. vol. 15. pp. 48–53. 2001.

[11] Kunniyur, S. Srikant, R. 2001. Analysis and design of an adaptive virtual queue (AVQ) algorithm for active queue management. In Proceedings of ACM SIGCOMM. San Diego. 2001

[10] Feng, W. Kandlur, D. D. Saha D. 2005. The Blue active queue management algorithms. IEEE/ACM Transactions on Networking 2002.

[12] Ott, T. J. Lakshman, T. V. and Wong, L. 1999. SRED: Stablised RED. IEEE INFOCOMM. March 1999

[13] Feng, W. Kapadia, A. Thulasidasan, S. 2002. GREEN: Proactive Queue Management over a Best-Effort Network. IEEE GlobeCom. Taipei. Taiwan. November 2002