Adaptive CHOKe: An algorithm to increase the fairness in Internet Routers

K.Chitra
Assistant Professor of Computer Science, D.J. Academy for Managerial Excellence
Coimbatore, Tamil Nadu, India.
Email: chitrakandaswamy@yahoo.com
Dr. G.Padamavathi
Professor & Head, Dept. of Computer Science
Avinashilingam University for Women, Coimbatore, Tamil Nadu, India.

ABSTRACT

Routers in Internet face the problem of congestion due to the increased use of Internet. AQM 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. There are various existing algorithms that have been evolved over the past few years to solve these problems of congestion in routers. RED based algorithms use queue length as congestion indicator while some of them use flow information for more accurate congestion indication. In this paper, we propose an AQM scheme that considers only the advantages of both these queue length based and flow based algorithm and satisfies the QOS requirements of the network. This proposed scheme aims to provide good service under heavy load and shields the responsive flows form unresponsive flows to offer a good QOS to all users.

Keywords : **Congestion, Drop Probability, Fairness, Queue length, Misbehaving flows**

Date of Submission: 01, April 2010 Date of acceptance: 11, June 2010

1. INTRODUCTION

A router in the Internet may receive thousands of flow at any given time due to heavy data traffic. Every flow should receive its fair share while sharing queue in the router. Another possibility is that the load tends to fluctuate in an Internet router resulting in a queue oscillation. Router must be able to handle the above problem. The buffer in the routers is to be used effectively by using an efficient Active Queue Management Mechanisms. Active Queue Management tries to prevent congestion and provides quality of service to all users. A router implementing RED AQM [1] maintains a single queue to be shared by all flow 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. The major disadvantage that exists in RED is parameter tuning problem. RED allows unfair bandwidth sharing when a mixture of traffic types share a link. In case a source sends too fast regardless of loss rate gains an unfair fraction of the bandwidth in RED AQM. In order to solve these problems, RED and its variant were proposed.

The RED based AQMs tried to solve the various problems existing with RED. Some of these AQMs tried to get rid of the parameter tuning problem in RED. While some of them tried to improve the performance compared to RED. The problem of unfairness was attempted by some of the AQMs. Adpative RED [2], PD-RED [3], AutoRED with RED and AdaptiveRED [4] tried to solve the parameter tuning problem. AQMs like MRED [5], DS-RED [6] were proposed to improve the performance measures of RED AQM. To improve fairness in case of different traffic types, some AQM started using flow based information as in Fair Random Early Detection (FRED) [7]. But FRED incurs extra implementation overhead because they collect information about the active flows. To overcome this, CHOKe [8] algorithm was proposed that simultaneously identifies and penalizes misbehaving flows. Both FRED and CHOKe were implemented in RED algorithm to calculate average queue size and packet drop probability with the additional flow manipulation to bring in fairness among the different traffic types available in the network that failed in RED.

RED based AQMs used Queue length as congestion indicator to detect congestion. Some of the AQM techniques besides using queue length as congestion indicator tried to use load and in some cases uses both queue length and load to detect congestion. AQMs like AVQ [9], Yellow [10] use load or input rate as their congestion metric to indicate congestion, whereas REM [11] calculates packet drop probability using both input rate and queue length. BLUE [12] uses link history and packet loss as congestion indicator to compute the packet drop probability.

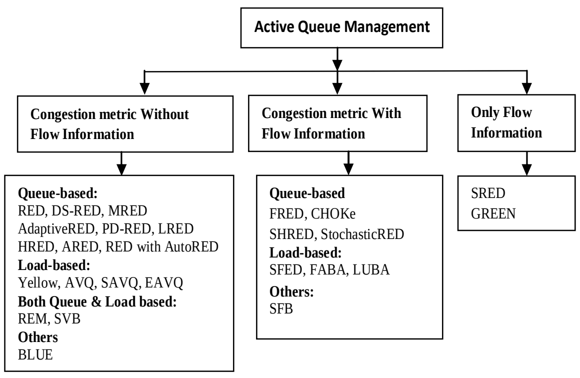
The objective of this paper is to propose an algorithm that considers the advantages of both queue based algorithm and flow based information. Because of the simplicity and easy implementation that exists in CHOKe, this paper takes AQM CHOKe as the base algorithm with certain modifications. To bring in the advantages of the queue based AQMs, the AQM CHOKe is implemented in the Queue based algorithm in this proposed algorithm. Specifically, we exhibit an AQM that removes the problems of RED like parameter tuning and unfairness among the different flows. This algorithm is simple to implement 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. Our conclusions are presented in section 4.

2. BACKGROUND

The Internet routers face the problem of congestion from the birth of Internet. Research activities are carried out with the origin of various congestion avoidance mechanisms in Internet to improve the performance of Internet traffic. The origin of each of these mechanisms has revealed the inefficiency of each of the AQMs in certain circumstances especially in heavy traffic network. Currently there are numerous algorithms to handle this problem. So research in this field has become a continuous process in identifying the best Active Queue Management algorithm. Congestion leads to high packet loss resulting in high cost that is minimized by the various existing AQM schemes. The various existing AQMs detect congestion based on different factors and calculate the packet dropping probability. Based on the various information used, the degree of congestion varies and is reduced differently. Taking this information into account, the existing AQMs can be categorized [13] as in Fig. 1.

Figure 1. Classification of AQM Schemes

RED AQM was the first proposed by Floyd et al in 1993



with the objective of preventing congestion occurring with reduced packet loss. This AQM alleviates congestion by detecting incipient congestion early and delivering congestion notification to the host to reduce the transmission rates avoiding overflow from occurring. The success of RED depends to a large extent on the appropriate selection of the RED parameters. So this becomes a major drawback for the RED AQM implemented in the network link. RED AQM also leads to global synchronization and lock-out problem 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, it maintains a virtual queue whose capacity is less than the actual capacity of the link. In the presence of the long-lived flows AVQ achieves low loss with high utilization. However it is difficult to achieve faster response time and high link utilisation using a constant value γ. In improving the method for setting the value for γ, SAVQ [14] is proposed. SAVQ stabilizes the dynamics of queue maintaining high link utilization.

In REM, both queue length and load is used as congestion indicators. REM shows poor performance when compared with AVQ. The BLUE algorithm resolves some of the problems of RED by employing two factors: packet loss from queue congestion and link utilization. So BLUE performs queue management based on packet loss 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. The parameters of BLUE are δ_1, δ_2 and freeze time. The freeze time determines the minimum time period between two consecutive updates of p_m. BLUE maintains minimum packet loss rates and marking probability over varying queue size and number of connections compared to RED. In case of large queue, RED has continuous packet loss followed by lower load that leads to reduced link utilization.

Regardless of the congestion indicators like queue length or input rate, some of the AQM requires additional flow information to calculate the packet drop probability or to detect congestion. To overcome the problem of unfairness among the flows in RED, FRED was proposed. FRED provides fairness between varying traffic. So during times of congestion, the number of flows uses the equal share of the queue. FRED calculates the average queue length as in RED, but uses the flow information to handle non-adaptive flows and robust flows. Considering its overhead of keeping track of flow information, CHOKe algorithm was proposed. It is a stateless algorithm that does not require any special data structure. This algorithm improves performance by penalizing misbehaving flows and implementing fair bandwidth allocation. Two shortcomings were faced by this algorithm which penalises UDP flows as well as TCP flows resulting in worsening TCP performance. Secondly it does not work well incase of only few packets from unresponsive flows in the queue. SFB [15] is a FIFO queueing algorithm that identifies and rate-limits non-responsive flows based on accounting mechanisms. The accounting bins are used to keep track of queue occupancy statistics of packets belonging to a particular bin. Each bin keeps a dropping probability p_m which is updated based on bin occupancy. As a packet arrives at the queue, it is hashed into one of the N bins in each of the levels. If the number of packets mapped to a bin goes above a certain threshold, p_m for the bin is increased. If the number of packets drops to zero, p_m is decreased. SFB is highly scalable and enforces fairness using an extremely amount of state and a small amount of buffer space.

SRED in [16] 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 [17] 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. As discussed in Introduction, some of the algorithms arrive at a stable queue size and some of them bring in fairness when the shared link has n flows. The unstable queue size results in high queue oscillation due to the parameter tuning problem in queue based AQMs. We are motivated to identify a scheme that penalizes the unresponsive flows with the stable queue size.

As mentioned in the Introduction, the proposed algorithm - AdaptiveCHOKe enforces the concept of queue-based and flow information. It is desirable for AQM schemes to act without storing a lot of information otherwise it becomes a overhead and non-scalable. We propose an algorithm that modifies the CHOKe algorithm to remove its drawback. This algorithm also 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}. As in Fig. 2, average queue size is compared with these thresholds for every arriving packet.

```
For every packet arrival {
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 ($Q_{ave} \geq max_{th}$) {
                 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	:current queue size
p_b	: temporary marking or dropping probability
w_q	: queue weight
max_p	: maximum dropping probability

Fixed parameters:

min_{th}	: minimum threshold for queue
max_{th}	: maximum threshold for queue

Figure 2 Pseudo code of AdaptiveCHOKe algorithm

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 the randomly selected packet is placed in the same position in the buffer.
In this proposed algorithm, the arriving packet is dropped with a probability depending on the average queue size. This algorithm is embedded in AdaptiveRED with AutoRED rather than RED. 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 as in AutoRED with AdaptiveRED. In such a case, the proposed algorithm reduces the problem of parameter tuning.

To achieve good throughput and reasonable average queue length with RED based algorithm requires careful tuning of both w_q and max_p. Adapting max_p controls the relationship between the average queue size and the packet drop probability and helps in maintains a steady average queue size in the presence of varying traffic. 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.

AdaptiveCHOKe as in Fig. 3 achieves the average target queue length and does not overshoot max_{th}. So this reduces both the packet loss rate and the variance in queuing delay. The dynamic value of w_q adapts itself to the varying nature of the congestion and traffic. The w_q is redefined and results in reduced instantaneous queue oscillation.

```
To calculate max_p:
Every interval seconds:
if ($Q_{ave}$ > target  and $max_p \le 0.5$)
        increase $max_p$
        $max_p = max_p + \alpha$;

elseif ($Q_{ave}$ < target  and $max_p \ge 0.01$)
        decrease $max_p$
        $max_p = max_p * \beta$;

Fixed parameters:
interval   :time; 0.5 seconds
target        :target for $Q_{ave}$
        [$min_{th} + 0.4 * (max_{th} - min_{th})$,
        $min_{th} + 0.6 * (max_{th} - min_{th})$]

α      : increment; min (0.01, $max_p/4$)
β      : drecrease factor :0.9
```

Figure 3. Definition of max_p

AdaptiveCHOKe also calculates the average queue size that uses w_q to calculate packet dropping probability. In Fig. 4,

$$Q_{avg,t} = (1 - w_q) * Q_{avg,t-1} + w_q Q_t$$

Where	Q_t	= instantaneous queue size at time t
	w_q	= small constant - queue weight
	$Q_{avg,t}$	= Average queue size at time t
	$Q_{avg,t-1}$	= Average queue size at time t-1

Figure 4 EWMA model

RED scheme uses the EWMA model to calculate Q_{avg},.

In AdaptiveCHOKe technique, the EWMA model is redefined as:

$$Q_{avg,t} - Q_{avg,t-1} = w_{q,t} (Q_t - Q_{avg,t-1})$$

The AdaptiveCHOKe models the weight w_q as a fine dependent parameter reflecting the dynamics of three types of network characteristics: 1) dynamics of congestion characteristics which include changes in transient congestion, changes between congestion and no congestion status, ii) dynamics of traffic characteristics which include changes in burst of traffic and steady state iii) queue normalization. So, the calculation of w_q in Fig. 5 includes the above stated network characteristics that derives a meaningful increase in the average queue size and eliminates the slowly varying nature of the average queue size that leads to chaos.

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

$w_{q,t}$	= 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

Figure 5 Definition of the weighing parameter $w_{q,t}$

The definition of weighting parameter w_q is written as a product of three functions 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 function T_t represents the dynamics of congestion characteristics of the network. It represents the probability that the system can lead to congestion in two steps according to the information available at time t and is a time dependent function.

The second function J_t represents the dynamics of the traffic characteristics of the network and is also time dependent function.

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

As mentioned previously, the AdaptiveCHOKe also calculates packet drop probability as given in Fig. 6.
The dynamic varying nature of w_q and max_p takes care of the network characteristics while the flow based information takes care of the unresponsive flow and misbehaving flows and brings in fair queuing at a minimum implementation overhead. This is implemented

$$p_b = max_p(Q_{avg} - min_{th}) / (max_{th} - min_{th})$$
$$p_a = p_b / (1 - count \cdot p_b)$$

Figure 6 To calculate p_a, p_b

by simple comparison of the packet from incoming traffic and the queued packets. Thus in a simple fashion the packets of the misbehaving flows are penalized. This is done because incase of packets belonging to non-adaptive or misbehaving flows are more likely to be chosen for comparisons. These packets are dropped more often than the adaptive flows and well behaved flows. The dynamic varying nature of the parameters and the flow information will try to keep the router congestion controlled.

4. CONCLUSION

This paper proposes an AQM scheme called AdaptiveCHOKe which aims to protect well-behaved flows from misbehaving flow and adaptive flows from non-adaptive flows. It also obtains high utilisation, low queuing delay and low packet loss with well adaptively tuned parameters. This packet dropping scheme discriminates against the flows resulting in fair congestion indication. This proposed AQM scheme inherits the advantages of the queue length based and flow based algorithm satisfies the QOS requirements of the network. Further work involves studying the performance of good service under heavy load and protecting the responsive flows form unresponsive flows to achieve a good QOS to all users by simulation.

REFERENCES

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

[2] S.Floyd., R.Gummadi,S.Shenkar 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

[3] Jinsheng Sun. King-Tim Ko.,Guanrong Chen., Sammy Chan.,Moshe sukerman., "PD –RED : To Improve Performance of RED", IEEE COMMUNICATIONS LETTER, August 2003

[4] Shan Suthaharan, "Reduction of queue oscillation in the next generation Internet routers", Science Direct, Computer Communication, 2007

[5] Jahoon Koo., Byunghun Song., Kwangsue Chung., Hyukjoon Lee., Hyunkook Kahng.,"MRED: A New Approach To Random Early Detection" 15th International Conference on Information Networking, February 2001

[6] Bing Zheng ,Mogammed Atiquzzaman, "DSRED: An Active Queue Management Scheme for Next Generation Networks" Proceedings of 25th IEEE conference on Local Computer Networks LCN 2000,November 2000

[7] D.Lin, R.Morris, "Dynamics of Random Early Detection". Proceedings of ACM SIGCOMM October 1997.

[8] Pan R., Prabhakar.B., and Psounix.k, "CHOKe, a Stateless Active Queue Management Scheme for Approximating Fair Bandwidth Allocation", IEEE INFOCOMM, Feb 2000

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

[10] Chengnian Long., Bin Zhao., Xinping Guan., Jun Yang., "The Yellow active queue management algorithm", Computer Networks, November 2004

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

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

[13] K.Chitra, G.Padamavathi, "Classification and Performance of AQM-Based Schemes for Congestion Avoidance " , IJSCIS, Vol. 8 No. 1, April 2010

[14] Cheng-Nian long., Bin Zhao., Xin-Ping Guan., "SAVQ: Stabilized Adaptive Virtual Queue Management Algorithm" ., IEEE Communications Letters ., January 2005

[15] Wu-chang Feng, Dilip D. Kandlur, Debanjan Saha, Kang G. Shin, **"**Stochastic Fair Blue: A Queue Management Algorithm for Enforcing Fairness", IEEE INFOCOM 2001

[16] T.J.Ott,T.V.Lakshman,and L.Wong, "SRED: Stablised RED", IEEE INFOCOMM, March 99

[17] Wu-chun Feng, Apu Kapadia , Sunil Thulasidasan,, "GREEN: Proactive Queue Management over a Best-Effort Network", IEEE GlobeCom, Taipei, Taiwan, November 2002

Authors Biography

K.Chitra received her B.Sc (C.Sc) from Women Christian College, Chennai and M.Sc from Avinashilingam University for Women, Coimbatore in 1991 and 1993 respectively. And, she received her M.Phil degree in Computer Science from Bharathiar University, Coimbatore in 2005. She is pursuing her PhD at Avinashilingam University for Women. She is currently working as Assistant Professor in the Department of Computer Science, D.J.Academy for Managerial Excellence, Coimbatore. She has 12 years of teaching experience. Her research interests are Congestion Control in Networks and Network Security.

Dr. Padmavathi Ganapathi is the Professor and Head of the Department of Computer Science, Avinashilingam University for Women, Coimbatore. She has 21 years of teaching experience and one year Industrial experience. Her areas of interest include Network security and Cryptography and real time communication. She has more than 80 publications at national and International level. She is a life member of many professional organizations like CSI, ISTE, AACE, WSEAS, ISCA, and UWA. She is currently the Principal Investigator of 5 major projects under UGC and DRDO