A Survey On Active Queue Management Mechanisms

G.Thiruchelvi[1] and J.Raja[2],

[1]Periyar Maniammai University,Thanjavur,Tamilnadu,India [2] SSN College of Engineering,Kalavakkam,Tamilnadu,India

Summary
Congestion control mechanism is one of the key that keeps any network efficient and reliable for the users. Many mechanisms were proposed in the literature over theses years for the efficient control of congestion that occur in the network. Active Queue Management (AQM) is one such mechanism which provides better control in the recent years. It works at the router for controlling the number of packets in the router's buffer by actively discarding an arriving packet. Many schemes were proposed which give better delay performance and high throughput over different traffic conditions. In this paper an exhaustive survey is made on the AQM techniques that are proposed and the merits and short falls is presented

Key words:
Active Queue Management, Congestion Control, Queue length, Link utilization, TCP, Non-TCP

1. Introduction

Congestion in Internet occurs when the link bandwidth exceeds the capacity of available routers. This results in long delay in data delivery and wasting of resources due to lost or dropped packets. The primary role of a router is to switch packets from the input links to output links through buffer. Apart from forwarding the packets, routers are involved for controlling the congestion in the network. It is known from [1] that routing algorithms focus on two main concepts namely queue management and scheduling. Queue management algorithms manage the length of packet queues by dropping packets whenever necessary whereas scheduling algorithms determine which packets to be sent next. These algorithms are used primarily to manage the allocations of bandwidth among various flows.

New trends in communication, especially the deployment of multicast and real time audio/ video streaming applications, are likely to increase the percentage of non –TCP traffic in the internet [2]. They don't share the available bandwidth fairly with applications built on TCP, such as Web browsers, FTP or e-mail clients. The Internet community strongly fears that the current evolution could lead to congestion collapse and starvation of TCP traffic.

TCP can detect packet drops and interpret them as indications of congestion in the network. TCP sender will react to these packet drops by reducing their sending rates. This reduction in sending rate translates into a decrease in the incoming packet rate at the router, which effectively allows the router to clear up its queue. When the incoming packet rate is higher than the router's outgoing packet rate, the queue size will gradually increase and queue becomes full at one stage.

The traditional technique for managing queue lengths is to set a threshold (in terms of packets) for each queue, accepts packets for the queue until the threshold is reached, then reject (drop) subsequent incoming packets until the queue decreases below the value of threshold. This technique is known as "tail drop", since packet that arrived most recently is dropped when the queue is full. Even though this method has served the Internet well for years, Baren B.et al pointed out two important drawbacks namely, Lock-Out and Full Queues. In some situations tail drop allows a single connection or a few flows to monopolize queue space, preventing other connections from getting room in the queue. This "Lock-Out" phenomenon is often due to the result of synchronization or other timing effects. The tail drop discipline allows queue to maintain a full status for long periods of time, since tail drop signals congestion only when the queue has become full. It is important to reduce the steady- state queue size and this is perhaps the queue management's most important goal.

In routing the packets there is a need to tradeoff between delay and throughput. If the queue is full or almost full, an arriving burst will cause multiple packets to be dropped. This can result in global synchronization of flows throttling back, a sustained period of lowered link utilization, reducing overall throughput. The point of buffering in the network is to absorb data bursts and to transmit them during the ensuing bursts of silence. This is essential to permit the transmission of bursty data. Maintaining small queues can result in higher throughput

Manuscript received December 5, 2008
Manuscript revised December 20, 2008

as well as lower end-to-end delay. In short, queue limits should not reflect the steady state queues; instead they have to reflect the size of the bursts needs to be absorbed.

Besides tail drop, B.Braden et al.. Considered two alternative queue disciplines that can be applied when the queue becomes full. They are "random drop on full "[3] or "drop front on full"[4]. Both disciplines solve lock out problem but neither of them solves full queues problem.

In Internet, dropped packets serve as a critical mechanism of congestion notification to end nodes. The solution to the full queues problem is for routers to drop packets before a queue becomes full, so that end nodes can respond to congestion before buffers overflow. Such approach is called as "Active Queue Management (AQM)" which is discussed elaborately in RFC2309.By dropping packets before buffers overflow, AQM allows routers to control packet drops.

By keeping the average queue size small, queue management will reduce the delays seen by flows. This is particularly important for interactive applications whose subjective (and objective) performance is better when the end-to-end delay is low. Active queue management can prevent lock-out behavior by ensuring that there will almost always be a buffer available for an incoming packet. It can also prevent a router bias against low bandwidth for highly bursty flows.

This paper is organized as follows. In section 2 we introduce the concept of AQM and classification of AQM schemes based on metrics of congestion measure. In section 3 some of the AQM schemes based on queue length metrics was discussed. AQM schemes based on load metrics were discussed in section 4. In Section 5 AQM schemes based on both queue length and load metrics were discussed. A detailed discussion was carried out in section 6 and the paper is concluded with section 7

2. Active Queue Management

The essence of Internet congestion control is that a sender adjusts its transmission rate according to the congestion measure of the underline networks. There are two approaches to accomplish this. One is a source algorithm that dynamically adjusts the transmission rate in response to the congestion along its path; the other one is a link algorithm that implicitly or explicitly conveys information about the current congestion measure of the network to sources using that link. In the current Internet, the source algorithm is carried out by TCP, and the link algorithm is carried out by active queue management (AQM) schemes at routers

According to the metrics used to measure congestion, AQM schemes can be classified into three catalogs: queue-based, rate based, and schemes based on concurrent queue and rate metrics. In queue-based schemes, congestion is observed by average or instantaneous queue length and the control aim is to stabilize the queue length. The drawback of queue-based schemes is that a backlog is inherently necessitated. Rate- based schemes accurately predict the utilization of the link, and determine congestion and take actions based on the packet arrival rate. Rate-based schemes can provide Early feedback for congestion Other AQM schemes deploy a combination of queue length and input rate to measure congestion and achieve a tradeoff between queues stability and responsiveness

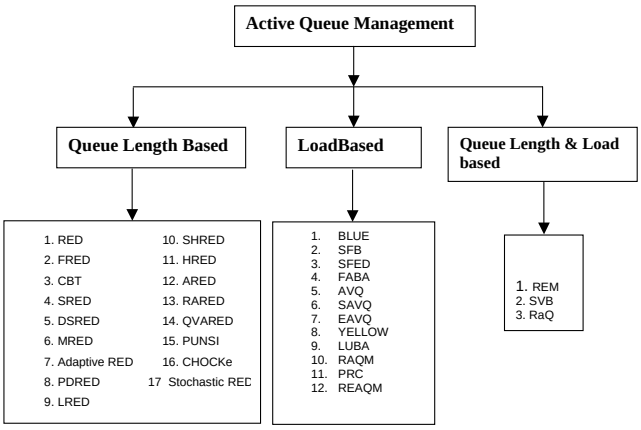

Fig 1.Classification of AQM Schemes

3. AQMs Based On Queue Length Merit

3.1 Random Early Detection (RED)

An active queue management scheme namely Random Early Deduction (RED) [5] alleviates congestion by detecting incipient congestion early and delivering congestion notification to the end source allowing them to reduce the transmission rates before overflow occurs. Since RED acts in anticipation of congestion, it does not surfer from the "Lockout" and "Full queue" problems inherent in the widely deployed drop tail mechanism. By keeping the average queue size small, RED reduces the delays experienced by most flows. The effectiveness of RED depends to a large extent, on the appropriate selection of the RED parameters. Self Configuring RED was later proposed by Feng [6] that self parameterizes itself based on the traffic mix. However adaptive

determination of the RED parameters complicates buffer management of high speed routers/gateways

3.2 Flow RED (FRED)

RED is vulnerable to unresponsive flows dominating a router's queue. Lin and Morris recognize this shortcoming of RED and proposed a scheme, called Flow Random Early Detection (FRED)[7], to promote fair buffer allocation between flows. FRED attempts to provide fair buffer allocation between flows, isolating each flow from the effects of misbehaving or non-responsive flows. FRED's approach is to impose uniformity during times of congestion by constraining all flows to occupying loosely equal shares of the queue's capacity (and hence receiving loosely equal shares of the outbound link's capacity). Moreover, flows that repeatedly exceed an average fair share of the queue's capacity are tightly constrained to consume no more than their fair share. This uniformity comes at a cost, however. Statistics must be maintained for every flow that currently has packets in the outbound queue of the router. These so-called "active flows" are allocated an equal share of the queue, which is determined by dividing the current queue size by the number of active flows. The number of packets a flow has enqueued is compared to the product of the flow's share value and a constant multiplier. This multiplier allows for non-uniform (bursty) arrival patterns among flows. A flow that exceeds the threshold including the multiplier is considered unresponsive and is constrained to its share (without the multiplier) until it has no more packets in the queue. FRED's major weakness, however, is the overhead associated with tracking active flows and keeping statistics (packet counts) for each active flow.

3.3 Class Based Thresholds(CBT)

Most multimedia applications choose UDP an unreliable transport mechanism as their underlying transport mechanism because they are concerned with throughput and latency rather than reliable delivery. CBT is an active queue management scheme that will maintain the positive features of RED, limit the impact of unresponsive flows, but still allow UDP flows access to a configurable share of the link bandwidth. Moreover, it does this without having to maintain per flow state in the router. CBT builds upon the drop thresholds of RED and the buffer allocations of FRED to provide a queue management policy that efficiently meets these goals[8]. The approach is to isolate TCP flows from the effects of all other flows by constraining the average number of non-TCP packets that may reside simultaneously in the queue. Classes of non-TCP traffic are also isolated from one another, specifically isolating continuous media traffic from all other traffic. Continuous media streams are tagged before they reach the router so that they can be classified appropriately. These flows are either self-identified at the end-system or identified by network administrators. Statistics are maintained for these classes of traffic and their throughput is constrained during times of congestion by limiting the average number of packets they can have enqueued. Untagged packets are likewise constrained by a different threshold on the average number of untagged packets enqueued. These thresholds only determine the ratios between the classes when all classes are operating at capacity (and maintaining a full queue). When one class is operating below capacity, other classes can borrow that class's unused bandwidth.

In CBT, the classification packets into one of the three classes i.e TCP, tagged or untagged is an operation that takes constant time. In FRED the packet are classified by which flow it is associated with. This classification is conceptually O(N) where N is the number of active flows. In the case of CBT the number of statistics involved is constant, one set for each of the three classes. Whereas in the case of FRED there are packet counts, strike counts, and a 5-tuple to identify the associated flow for every active flow (O(n)).

3.4 Stabilized RED (SRED)

Like RED, Stabilized Random Early Drop (SRED) [9] preemptively discards packets with a load dependent probability when buffer in a router in the Internet or an intranet seems congested. SRED has an additional feature that over a wide range of load levels helps it stabilize its buffer occupation at a level independent of the number of active connections. SRED does this by estimating of the number of active connections or flows. This estimate is obtained without collecting or analyzing state information on individual flows. The same mechanism can be used to identify flows that may be misbehaving. In SRED there is no computation of average queue length. The packet loss probability depends only on the instantaneous buffer occupation and on the estimated number of active flows.

T.J.Ott et al showed that, while buffer occupancy in SRED is independent of the number of connections when the number of connections are less ,it increases only slightly if the number of connections increased to 1000.In RED the buffer occupancy increases in proportion with the number of connections. For SRED the buffer occupancy is almost always at least B/3 where B is the buffer occupancy. Next observation in SRED is, for TCP

flows, the impact of packet drop is very high when the bottleneck buffer occupancy is dominated by a few active flows with large windows and is very little when the bottleneck buffer occupancy is caused by a large number of connections with small windows. Hence, mechanisms like RED, which tries to control buffer occupancy, can be benefited by adjusting their drop probabilities, using the estimates of the number of active connections. As SRED can stabilize over a wide range of load levels, the buffer occupancy at a level, which is independent of the number of active connections, therefore overcomes the scalability issues but suffers from low through put.

3.4 Double Slope RED (DSRED)

The low through put of the RED is the most important problem which needs to be carefully addressed. A new active queue management scheme called "Double Slope Random Early deduction (DSRED)" was proposed in [10] that use a combination of two different drop probability distributions to achieve higher performance than RED. It resembles RED in two aspects; first, both of them use linear drop functions to give a smooth increase in drop action based on average queue length. Secondly they calculate the average queue length using the same definition to account for the effect of long-term congestion. Therefore DSRED inherits the advantages of RED. When congestion increases, drop will increase with higher rate instead of constant rate. This will give an early warning to hosts to back off, preventing congestion from getting worse. As a consequence, congestion will be relieved and throughput will increase. The two segments of the drop function can be adjusted by the parameter γ where γ represents mode selector for adjusting drop function slopes. Therefore the operating mode of DSRED can be easily adjusted by a single parameter. i.e., by adjusting γ , one can get high drop rate first followed by a low drop rate, or vice versa. This is more effective than RED in handling complicated network congestion situations. Bing Zheng et al concluded that, under heavy load ,with γ = 0.96 and max $_{drop}$= 0.1 DSRED gateway always has lower queuing delay, smaller queue size and lower packet drops than the RED gate way queue .

Table 1 Performance comparison between DSRED and RED **(Reprinted from Ref10)**

	DSRED		**RED**	
Parameter	H Load	L load	H load	L load
Norm Thrupt	0.525	0.372	0.445	0.355
Avg QD(s)	0.0081	0.0081	0.013	0.00118
Avg QS(Pkt)	2.6	2.7	7.3	6.8
Avg PD (Pkt/s)	2.5	2.5	13.1	11.25

3.5 MRED

A new concept of active queue management called MRED [11] was developed which controls queue by using packet loss information and link utilization history information with small queue size. MRED estimate average queue size either using a simple EWMA in the forwarding path or using a similar mechanism in the background. Therefore MRED has two separate algorithms. One is for computing the average queue size to determine the degree of burstiness that will be allowed in the router queue and the other is the one for calculating the packet drop probability to determine how frequently the router drops packets at the current level of congestions. Since it calculates a drop probability value based on link utilization history information, MRED efficiently controls the congestion caused by retransmission of dropped packet. From Fig 2 it is found that the through put of MRED is slightly better than that of RED as shown below.

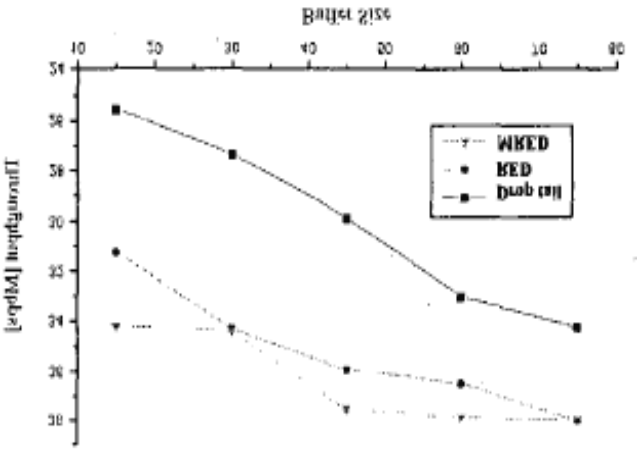

Fig -2 Throughput Vs Buffer size (Reprinted from Ref 11)

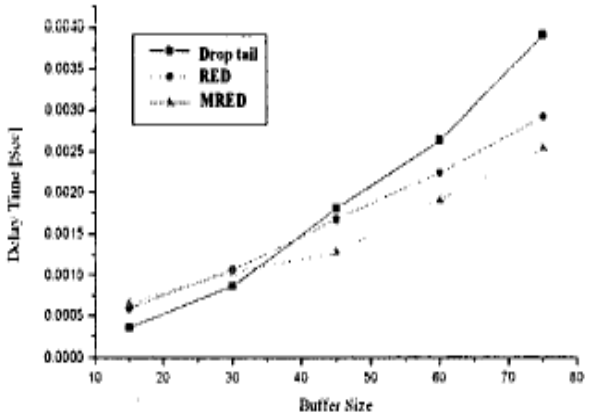

Fig-3 Delay time vs Buffer Size (Reprinted from Ref 11)

3.6 Adaptive RED

Floyed et al. proposed another AQM method called adaptive RED [12].In this the parameter max_p is adapted using an additive increase multiplicative decrease policy to keep the average queue length within a target range and is adapted not just to keep the average queue size between min_{th} and max_{th}, but to keep the average queue size within a target range halfway between min_{th} and max_{th}. max_p is adapted slowly, over time scales greater than a typical round trip time and in small steps. As adaptive RED can stabilize the queue length at a given target, its performance can still be improved if we adapt the drop probability max_p in a more methodological manner. Despite significant control theory advancement most industrial processes use proportional integral derivative (PID) controller. The new PDRED AQM [13] solution is composed of two parts (i) a new PD controller (ii) The original RED AQM. Jinsheng Sun et al. showed that the fluctuation in amplitude of PD-RED queue length is smaller and the variance of the drop probability is much smaller than those of adaptive RED.

3.7 Loss Ratio Based RED (LRED)

Another AQM scheme with fast response time was introduced by [14] called Loss Ratio based RED (LRED).It measures the latest packet loss ratio and uses it as a compliment to queue length in order to dynamically adjust packet drop probability. Employing the closed form relationship between packet loss ratio and the number of TCP flows this scheme is responsive if the number of TCP flows varies significantly. An increasing packet loss ration is a clear indication that severe congestion occurs and that aggressive packet dropping is needed. On the other hand a decrease in packet loss ratio can serve as a signal that congestion is receding and consequently, that packet drop action can change from aggressive to moderate. Therefore it is possible to use the packet loss ratio to design more adaptive and robust AQM scheme. LRED uses instantaneous queue length to calculate the packet drop probability each time packets arrive while dynamically adjusting the packet drop probability according to the measured packet loss ratio over relatively large time scale. Such a combination enables fast response time and high robustness. LRED is most suitable for congested networks though its performance is comparable to PI & REM in a network of a light traffic load. Another version of AQM, which uses the same packet loss ratio, is proposed in AQMS_PLR. The arriving packet is marked as the early drop probability by the network loss ratio of the early network so that the probability can approach the loss ratio of current real network

3.7 Short Lived Flow Friendly RED (SHRED)

Short-lived flow friendly RED (SHRED) [15] targeted at providing better network performance for short web traffic. The basic idea is to use a lower drop probability for flows with small congestion window and to have the drop probability to increase linearly with a flow's increased relative congestion window size. Using an edge hint to indicate the congestion window size in each packet sent by the flows source or by an edge router, SHRED preferentially drops packets from short-lived web flows less often than packet from long lived flows. Mark Claypool et al.. concluded that for web only traffic SHRED performs slightly better than drop tail for low to moderate congestion level whereas RED performs worse than drop tail. RED always performs better than drop tail for mixed traffic and SHRED performs significantly better than RED and drop tail with mixed traffic and web only traffic for moderate to high levels of congestion.

3.8 Hyperbola RED (HRED)

A minimal adjustment to the RED algorithm was proposed by Hyperbola RED (HRED) that uses the hyperbola as the drop probability curve[16]. The control law of HRED can regulate the queue size, which can be set by the user. As the reference queue size is set by the user, HRED is no longer sensitive to the level of network load and it can achieve higher network utilization and result in predictable average queuing delays. It retains the ability to control the short congestion by absorbing bursts, because its still keeps the moving average queue size algorithm and maintain a non-full queue.

3.9 ARED

Another adaptive AQM mechanism ARED was proposed in [17] which can keep queue around the expected value through adjusting maximum drop rate adaptively using gradient descent method. Simulation results show that ARED can stabilize the average queue length and instantaneous queue length around the target value with different number of connections while original RED algorithm oscillates violent both under heavy and light traffic loads. The performance of Adaptive RED is further increased by tuning the max_p which in the Refined Adaptive RED (RARED) [18] constrained to remain in the range[0.01,0.5].The metrics used in RARED for analysis are TCP good put and TCP and UDP packet drop rate. From Figure 4 and table 2 it is observed that RARED has a slightly higher or similar to TCP good put than those with RED, ARED, and a slightly lower TCP/UDP packet drop rate. From Fig 5 it is clear that RARED maintains steady true average queue size than RED,ARED

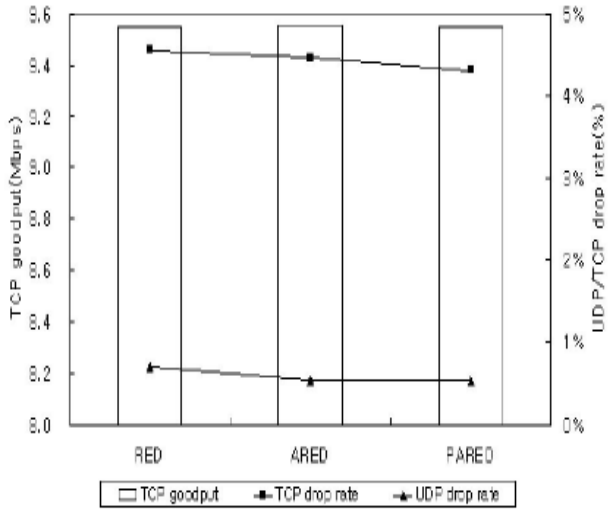

Fig - 4 TCP goodput(left) and TCP/UDP packet drop rate(right) for 25 long-lived TCP connections with 10% UDP traffic (Reprinted from Ref 18)

3.10 Queue Variation Adaptive RED (QVARED)

"Queue Variation Adaptive RED" [19] a variant of ARED responds to bursty traffic more actively. This is based on the variation of a queue per hour. As QVARED handles bursty traffic dynamically, dropped packets decrease significantly in comparison with RED and ARED. Thus end- to- end delay, like Web traffic gets short as a result. The study showed that the drop rate of QVARED is decreased by 80% and 40% compared to RED and ARED respectively. It reduces the bias effect over 18% than that of drop- tail method, therefore packets are transmitted stably with respect to bursty traffic

Table -2 TCP/UDP packet drop rate for 25 long-lived TCP connections.(Reprinted from Ref 18)

		RED	ARED	RARED
TCP packet drop rate	1 UDP	4.56%	4.46%	4.31%
	2 UDP	4.44%	4.95%	4.83%
	3 UDP	6.12%	6.13%	6.12%
	4 UDP	6.69%	6.84%	6.43%
UDP packet drop rate	1 UDP	0.70%	0.54%	0.54%
	2 UDP	0.66%	0.66%	0.53%
	3 UDP	0.79%	0.77%	0.76%
	4 UDP	0.83%	0.87%	0.71%

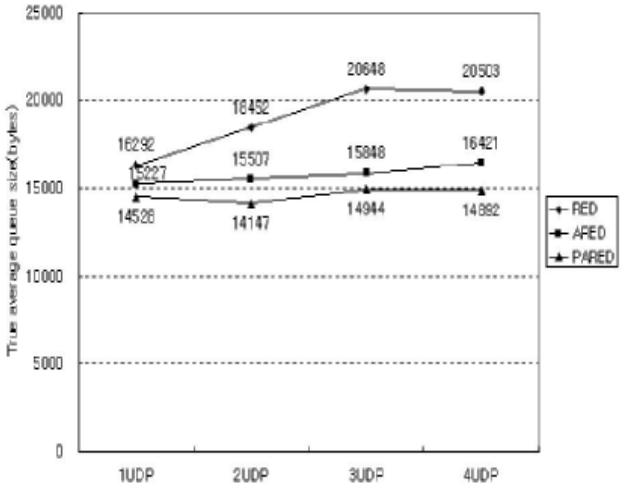

Fig – 5 True average queue size of RED, ARED and RARED, for 50 long-lived TCP connections with 1-4 UDP traffic.(Reprinted from Ref 18)

3.11 PUNSI

Another algorithm, which deals unresponsive flows, is the PUNSI algorithm [20]. It prevents unresponsive flows from dominating available bandwidth shared with responsive flows. This is done by penalizing packets from unresponsive flows with a higher probability than those from responsive flows. It is motivated by the observation that unresponsive flows tend to generate traffic of higher rates than responsive flows and that, when a packet is dropped due to buffer overflow, fellow packets from the same flow seem to be found in the buffer among those having joined recently. This algorithm first allocates good fair share of bandwidth among all flows passing through a router and achieves this without per flow information. Queuing algorithms with good fair sharing of bandwidths and stateless information are important since they reduce the complexity due to large overhead caused by more number of flows as against algorithms like Flow Random early Drop (FRED) which maintain per flow status. CHOKe algorithm [21] penalizes not only high bandwidth UDP flows but also TCP ones. Several packet losses in a short period worsen TCP performance significantly It doesn't work well if there are only a few packets from unresponsive flows in the queue. These two shortcomings of CHOKe are overcome by PUNSI algorithm that penalized UDP flows more effectively in accordance with its burstiness. The authors have shown using Figures 6 and 7, that PUNSI doesn't worsen the TCP performance In contrast to UDP flow, the TCP loss rate is consistently around 1% which means that PUNSI penalizes the bursty UDP flows but not TCP flows

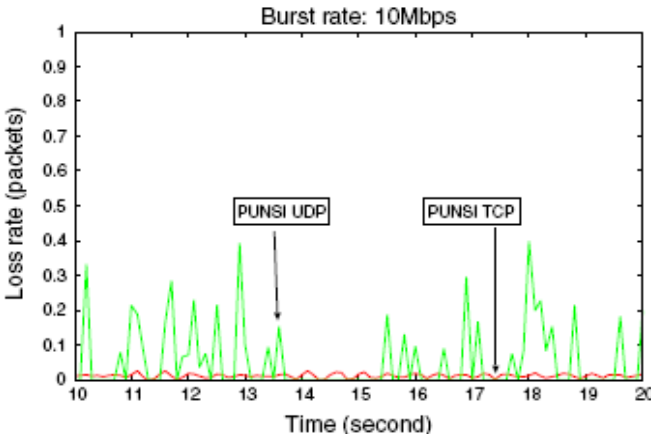

Fig -6 Loss Rate of On - Off CBR traffic with Burst rate of 10Mbps(Reprinted from Ref 20)

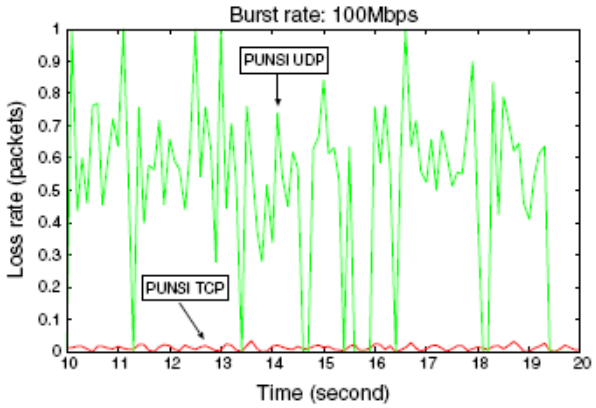

Fig -7 Loss Rate of On - Off CBR traffic with Burst rate of 100Mbps(Reprinted from Ref 20)

3.11 Stochastic RED

A scalable algorithm called Stochastic RED [22] was introduced later keeping in mind the tremendous growth of unresponsive traffic in Internet. A major difficulty in distinguishing individual flows without requiring per-flow state information at the routers is overcome by Stochastic RED. It is called stochastic because it does not really distinguish the flows accurately. The arriving traffic is divided by the router into a limited number of counting bins using a hashing algorithm. On the arrival of each packet at the queue, a hash function is used to assign the packet to one of the bins based on the flow information.

Stochastic Red dispatches the packets of the different flows to the set of bins. With a given hash function, packets of the same flow are mapped to the same bin. Therefore, when the flow is unresponsive, the bin load increases dramatically. Stochastic RED estimates the bin loads and uses these loads to penalize flows that map to each bin according to the load of the associated bin. Thus unresponsive flows experience a larger packet drop probability. However, because of the hashing and limited number of bins, multiple flows may end up associated with the same bin. Thus flows that share a bin with an unresponsive flow are punished unnecessarily. To prevent this situation, Stochastic RED changes it's hashing function often enough so that the time span for any two flows to collide into the same bin is within several seconds. In the long run, only misbehaving flows get significantly disciplined by Stochastic RED, making them TCP friendly and improving the response time of Web transfers.

From the above discussion on various queue length based AQMs we observe that the RED has problems such as low throughput, large delay/jitter, unfairness to connections and inducing instability in the network.

Tuning of parameters of RED is the main goal of almost all newly developed AQMs. Most of the simulation studies assume idealized traffic, which differs significantly from real bursty traffic. The queue based AQM schemes use average queue length (or instantaneous queue length) as a congestion indicator. However the window size and packet marking probability are relative to input traffic load directly. Therefore a new congestion indicator and control function are needed to provide adaptive control to the traffic characteristics such as the amount of traffic, fluctuation of traffic load and traffic nature

4. AQMs Based On Load Merit

Rate based AQMs determine congestion and take actions based on packet arrival rate. The goals of the rate based AQMs are to alleviate rate mismatch between enqueue and dequeue, and achieve low loss ,low delay and high link utilization. Since the queue length is a cumulative difference values of rate mismatch between enqueue and dequeue, queue merit is insensitive to current queue arrival and drain rates. This incurs the conservative /aggressive packet marking behavior when the queue length is small or large It explains in part the promising performance of rate based AQMs compared with queue based schemes under dynamic traffic scenarios.

4.1 BLUE

The inherent problem with the AQM algorithms is that they use queue length as the indicator of the severity of congestion. In the light of this observation, a fundamentally different AQM, called BLUE, [23] is proposed, implemented and evaluated. BLUE uses packet loss and link idle events to manage congestion. BLUE maintains a single probability, p_m, which it uses to mark (or drop) packets when they are enqueued. If the queue is continually dropping packets due to buffer overflow, BLUE increments p_m, thus increasing the rate at which it sends back the congestion notification. Conversely, if the queue becomes empty or if the link is idle, BLUE decreases its marking probability. This effectively allows BLUE to "learn" the correct rate it needs to send back congestion notification. BLUE uses two other parameters, which control how quickly the marking probability changes over time. The parameter freeze _time determines the minimum interval between two successive updates of p_m. The value of freeze _time should be randomized in order to avoid global synchronization. The other parameters used δ_1 and δ_2 determine the amount by which p_m is incremented when queue overflows or is decremented when the link is idle. In BLUE the value of δ_1 is set slightly larger than δ_2 . This is because the link under utilization can occur when congestion management is either too conservative or too aggressive, but packet loss occurs only when the congestion management is too conservative. The most important consequence of using BLUE is that congestion control can be performed with a minimal amount of buffer space. This reduces end-to-end delay over the network, which in turn improves the effectiveness of the congestion control algorithm. Using both simulation and controlled experiments the authors Wu –Chung Feng et.al. showed that, BLUE is performing significantly better than RED, both in terms of packet loss rates and buffer size requirements

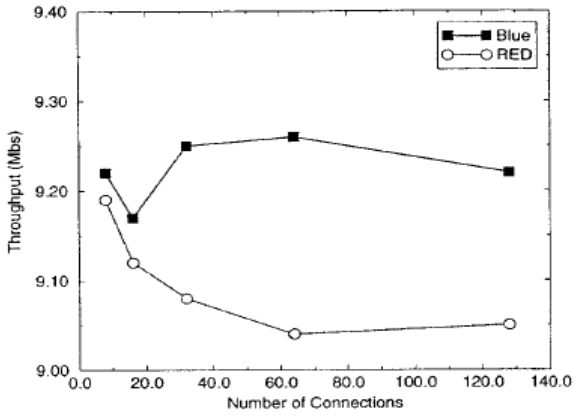

Fig -8 Queue management performance Throughput Vs Number of Connections (Reprinted fromRef - 23)

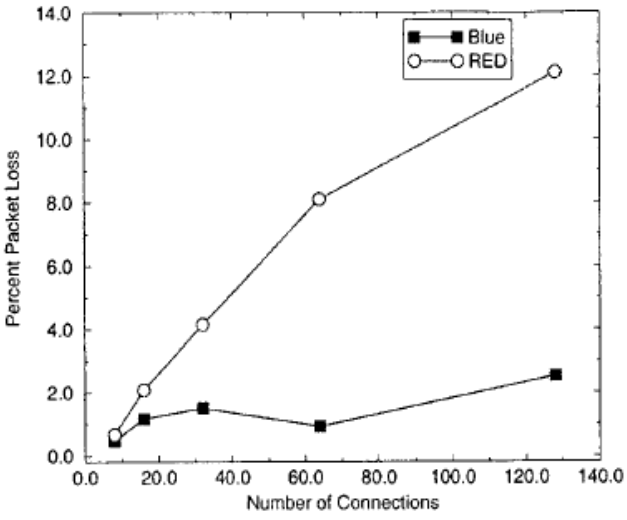

Fig - 9 Queue management performance - percent packet loss Vs Number of Connections (Reprinted from Ref-23)

4.2 Stochastic Fair BLUE

As an extension to BLUE, a Stochastic Fair Blue [23] algorithm based on Bloom filter is proposed that can identify and limit the rate of non-responsive flows using very small amount of state information. SFB maintains NxL accounting bins. The bins are organized in L levels with N bins in each level. In addition SFB maintains L independent hash functions ,each associated with one level of the accounting bins. Each hash function maps a flow, vis its connection ID(Source address, Destination address, Source port ,Destination port, Protocol) into one of the N accounting bins in that level. The accounting bins are used to keep track of queue occupancy statistics of packets belonging to a particular bin. This is in contrast to Stochastic Fair Queueing (SFQ) where the hash functions map flows into separate queues. SFQ is similar to SFB with one level of bins. The biggest difference is that instead of having separate queues SFB uses hash function for accounting purposes. Thus the SFB has two advantages over SFQ. The first is that it can make better use of its buffers. SFB gets some statistical multiplexing of buffer space as it is possible for the algorithm to overbook buffer space to individual bins in order to keep the buffer space fully utilized. Other is that SFB is a FIFO queueing discipline. As a result, it is possible to change the hash function on the fly without having to worry about packet reordering caused by mapping flows into a different set of bins. Each bin in SFB maintains a marking probability p_m , as in BLUE ,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 L levels, If the number of packets mapped to a bin goes above a certain threshold (i.e the size of the bin) p_m for the bin is increased and if the number of packets drops to zero, p_m is decreased. The observation which drives SFB is that a non responsive flow quickly drives pm to 1 in all of the L bins it is hashed into. Responsive flows may share one or two bins with non responsive flows, however, unless the number of non responsive flow is extremely large compared to the number of bins, a responsive flow is likely to be hashed into at least one bin that is not polluted with non-responsive flows and thus has normal p_m value.

4.3 SFED

SFED is an easy to implement rate control based AQM discipline,[24] which can be coupled with any scheduling discipline. It operates by maintaining a token bucket for every flow (or aggregates of flow). The token filling rates are in proportion to the permitted bandwidths. Whenever a packet is enqueued, tokens are removed from the corresponding bucket. The decision to enqueue or drop a packet of any flow depends on the occupancy of its bucket at that time. A sending rate higher than the permitted bandwidth results in low bucket occupancy and so a larger drop probability thus indicating the onset of congestion at the gateway. This ensures the adaptive flow to attain a steady state and prevents it from getting penalized severely. However non-adaptive flows will continue to send at the same rate and thus will suffer more losses. The rate at which the tokens are removed from bucket of a flow is equal to the rate of incoming packets of that flow, but the rate of addition of tokens in a bucket depends on its permitted share of bandwidth and not on the rate at which packets of that particular flow are dequeued. In this way token bucket controls the bandwidth consumed by a flow. This SFED takes O(N) operations for enqueue and dequeue .

4.4 FABA

The extension of SFED called FABA [25] is proposed which takes O(1) operations for both enqueue and dequeue. This extension makes FABA algorithm scalable, and hence, practical to implement as compared to the SFED algorithm. The observation from Fig-10 and Fig 11 is that the fairness index is the largest with FABA for even very large number of HTTP connections and for FTP and Telnet connections also, FABA performs consistently better than any other AQM mechanisms. With a traffic mix of fragile and non-fragile sources, FABA provides bandwidth allocation for fragile flow almost as good as in the ideal case. For a small maximum window size every algorithm is able to accommodate the bursts of the fragile flows without any drops, but with increasing maximum window size, packet drops result in drastic reduction of the fragile flow throughput. A packet drop is fatal for a fragile flow as it is slow in adapting to the state of the network. the throughput becomes constant after a while since the window size of the fragile source is not able to increase beyond a threshold. Therefore no matter how large the maximum window size is increased beyond this threshold, the throughput does not increase and approaches a constant value as shown in Fig 12. This constant is much less than its fair share due to less adaptive nature of fragile flows

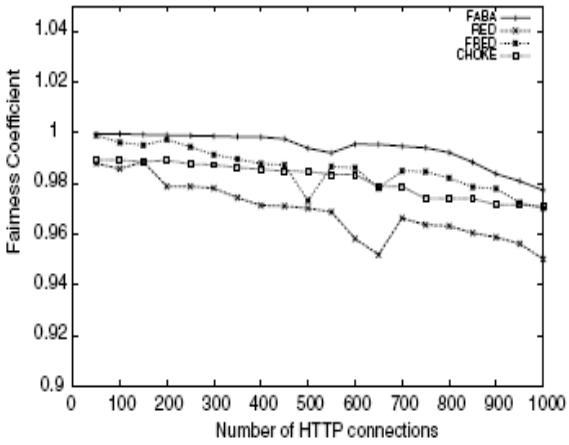

Fig- 10 Fairness Coefficient versus number of HTTP connections for different AQM mechanisams (Reprinted from Ref 25)

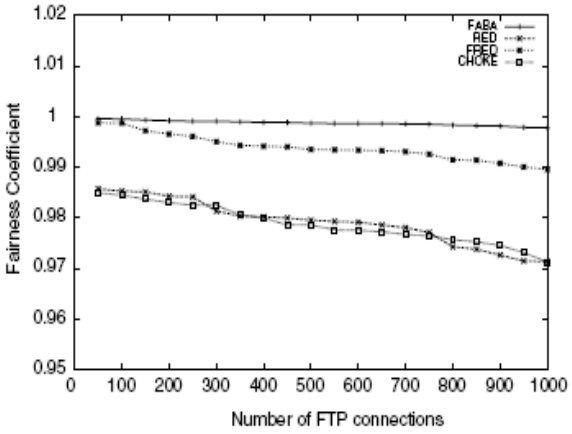

Fig-11Fairness Coefficient versus number of FTP connections for different AQM mechanisms (Reprinted from Ref 25)

4.5 Adaptive Virtual Queue (AVQ)

Another rate based AQM called Adaptive Virtual Queue algorithm for active Queue Management (AVQ) which maintains a virtual queue whose capacity is less than the actual capacity of the link [26]. When a packet arrives in a real queue, virtual queue is also updated to reflect the new arrival. Packets in the real queue are dropped/marked when the virtual buffer overflows. The virtual capacity at each link is then adapted to ensure that the total flow entering each link achieves desired utilization of the link. There are two parameters that have been chosen to implement AVQ. One is desired utilization γ and the other is damping factor α. The desired utilization determines the robustness to the presence of uncontrollable short flows. It

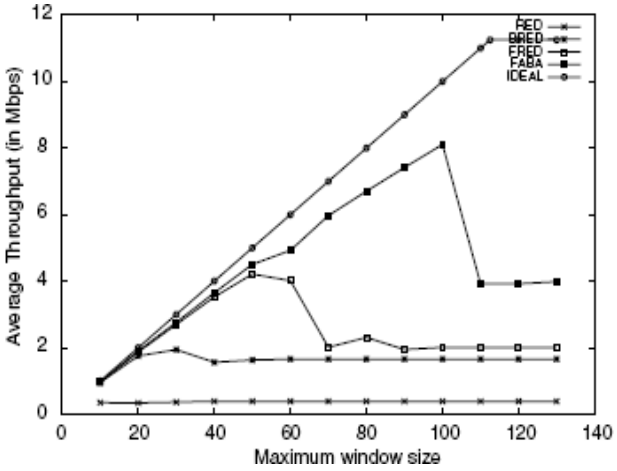

Fig-12 Performance of the fragile flow with increasing receiver window constraint (Reprinted from Ref 25)

allows an ISP to trade off between high levels of utilization and small queue length. Both the parameters determine the stability of the AVQ algorithm. Srisankar S.Kunniyur et al. showed that in the presence of long lived FTP flows alone AVQ achieves low loss with high utilization. They also showed that AVQ is responsive to changes in network load and is able to maintain a small queue length even when network load keeps increasing. When short flows are introduced along with FTP flows, the AVQ has lower drops compared to REM, PI and RED and higher drop than GKVQ. Even though the drop is higher than GKVQ, the utilization at the link for AVQ is significantly greater than GKVQ algorithm. When dropping (instead of marking) is employed at the routers, the AVQ performs better than other AQM schemes in terms of utilization and average queue length but the fairness can be improved using probabilistic AQM scheme like RED on AVQ. Probabilistic AQM scheme is required only when the link drops packets and not when the link marks packets, because multiple marks within a single window does not cause TCP to timeout or to go into slow start. However it is difficult to achieve a fast system response and high link utilization simultaneously using a constant value γ.

4.6 Stabilized Adaptive Virtual Queue(SAVQ)

An adaptive setting method for γ is proposed according to the instantaneous queue size and the given reference queue value in Stabilized Adaptive Virtual Queue Management algorithm (SAVQ) which stabilizes the dynamics of queue maintaining a high link utilization

[27].The table 3 shows the performance indices of AVQ (with γ=1.0, γ=0.98) and SAVQ.

Table –3 Performance indices of AVQ and SAVQ(Reprinted from Ref 27)

Criterion	AVQ		SAVQ
	γ =1.0	γ=0.98	
Average Queue Length	115.4	37.64	43.19
STD of Average Queue Length	72.52	52.53	38.09
Utilisation%	99.86	98.22	99.58
Loss Ratio%	0.339	0.129	0.092

In the case of AVQ with γ=1, the queue jitter is remarkable which deteriorates the transient performance and increases the average queue length and packet loss ratio. The link utilization reduces much using AVQ with γ=0.98.The above table shows that the transient response of SAVQ algorithm outperforms AVQ and the queue length of SAVQ converges fast to the desired length while maintaining a satisfactory utilization and packet loss ratio.

4.7 Stable Enhanced Adaptive Virtual Queue (EAVQ)

An enhancement to AVQ was proposed in rate-based Stable Enhanced Adaptive Virtual Queue (EAVQ) algorithm[28]. The concepts of the Principal and subordinate measures of congestion, as well as desired link utilization ratio were introduced into EAVQ. Arrival rate at the network link was maintained as a principal measure of congestion. The desired link utilization ratio was used as a subordinate measure and a rate-based adaptive mechanism of which was designed to resolve the problems, such as hardness of parameters setting, poor ability of anti-disturbance, and a little link capacity loss. EAVQ improved the transient performances of the system and ensured the entire utilization of link capacity. Qian Yanping et al , shown the excellent performances of EAVQ in terms of higher utilization, the lower link loss rate, the more stable queue length, and the faster system dynamic response than AVQ. Furthermore the performances of EAVQ are insensitive to the number of TCP connections

4.8 YELLOW

Yellow active queue management algorithm [29] uses the mismatch between the input rate and link capacity as the primary metric. Therefore the advantages of rate based AQMs are inherited. Furthermore, queue size is made as a secondary metric. Queue length affects the load factor using Queue Control function, which is computed by a non-linear hyperbola function of instantaneous queue length and reference queue size. Known from other rate based schemes Yellow provides an early controlling queuing delay maintaining the main load merit. The average queue length and Standard deviation of queue length of Yellow are little affected by the introduction of the UDP flows.

4.9 Link Utilization Based Approach (LUBA)

Another Link Utilization based Approach is LUBA [30]. In this approach, the malicious flows are identified which might be causing congestion at the router and assign them drop rates in proportion of their abuse of the network. If the overload factor U=λ/μ (where λ is the aggregate arrival rate at the router and μ is the outgoing link capacity of the router), is below the target link utilization ,the router is non-congested and packets are not marked or dropped. When it is greater, all arriving packets are monitored while assigning flow _id to each ingress flow at the router. A history table is maintained to monitor flows which take more than their fair share of bandwidth in a luba interval which is a byte count of total packets received by the congested router during an interval in which we measure whether a flow is hogging more than its fair share. For each incoming packet, if its flow _id is not listed in the history table, its flow _id is inserted in the table with drop probability that is assigned to all those flows that are not identified as malicious , to limit their aggregate share to the residual link capacity that is remaining after rate limiting the malicious flows. The packet is inserted in FIFO queue if it is not dropped as per the dropping probability. If the flow _id is present in the history table, the packet is dropped as per the drop probability and if the packet does not get dropped, it is inserted into the FIFO buffer. All packets in the buffer are serviced by the router at the μÛ per second where Û is the target link rate. The LUBA interval should not be very large as that would result in sluggish system and unresponsive behavior in the presence of short-lived flows. But it should be large enough to capture active flows. When λ increases, the time duration τ to get the Luba interval bytes decreases. It implies that during high congestion period, history table contains mostly malicious flows because during such a short time interval only those

flows can regularly figure in history table that are consistently sending at a very high rate. Jitter degrades the performance of both TCP and UDP flows. With the change in arrival rate, queue builds up at the rate proportional to log (λ) instead of λ. This has implications in FIFO queue buildup and jitter properties of the router. The end-to-end jitter introduced due to a series of routers, demonstrate that LUBA acts as jitter suppressor. Manoj K.Agarwal et al. showed that UDP sources with rate less than 1 Mbps do not figure in the drop table. And UDP sources with data rate above 1 Mbps are almost always present in the history table. In the presence of large number of UDP flows, the fairness index is much better than other AQMs (Fig 13). SFB gives results closest to LUBA. But SFB is computationally expensive and difficult to configure. LUBA is also able to maintain the high TCP throughput with increasing load on the bottleneck link compared with other algorithms(Fig 14)

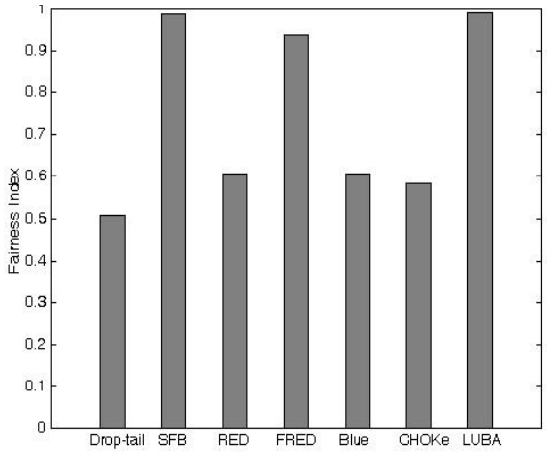

Fig -13 Fairness Comparison with other AQMs (Reprinted from Ref 30)

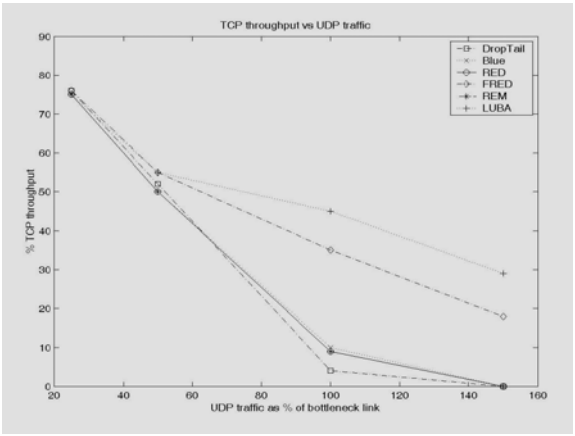

Fig -14 TCP throughput with varying load (Reprinted from Ref 30)

4.10 RAQM

Another new rate based AQM (RAQM) [31] uses the aggregated traffic input rate to calculate packet drop probability according to an exponential rule. This RAQM works in two modes. Although RAQM needs to measure traffic input rate, yet this operation is for the aggregated flows and there is no need to differentiate each micro flow: Queue independent mode and Queue dependent mode. In the queue independent mode it only relies on the aggregate traffic input rate to regulate the input rate to the expected link utility. In queue dependent mode, it also uses the instantaneous queue length to further adjust the packet drop probability and to regulate the queue length to the expected value. The rational for using the exponential rule is that it can achieve stability and leads to faster convergence of the packet drop probability to the expected value. RAQM also obtains higher Good put at the same cost of average queue length and better trade off between good put and queuing delay.

4.11 Proportional Rate Control (PRC)

In the proportional rate based control (PRC) two parameters i.e minimum threshold and maximum threshold are maintained to effectively control input rate under the desired level. virtual queue concept also introduced into system to regulate the queue size[32]. The attractive features include packet dropping to control the arrival rate between minimum and maximum desired rate and ECN marking to inform the source to reduce the transmission rate quickly when virtual buffer becomes full. It combines the advantages of quick responsiveness and persistent stability

4.11 Rate based Exponential AQM

Current AQMs and TCP are designed and tuned to work well for wired networks where packet loss is mainly due to network congestion. In wireless networks, however communication links suffer from transmission bit errors and handoff failures. As a result the performance of TCP flows is significantly degraded. To mitigate this problem a Rate based Exponential AQM (REAQM) is proposed[33] .It tries to stabilize the system and achieve low delay, low packet loss and high link utilization regardless of the dynamic of network conditions. Similar to REM REAQM maintains a variable, price, as a congestion measure and uses an exponential marking probability function. On the other hand, REAQM differs from REM in the definition of congestion measure. The

price at queue l at time t is denoted by $p_l(t)$.The marking probability function of REAQM is $m_l(t)=1-\Phi^{-p_l(t)}$ where Φ is a constant larger than 1.

Price is updated, periodically or asynchronously mainly based on rate mismatch. Rate mismatch is positive when the input rate exceeds the link capacity and negative otherwise. The price is incremented if the rate mismatch is positive and decremented otherwise. Queue length is used to compute the coefficient of rate mismatch and adjust the scale of rate mismatch. If queue length is small, the effects of rate mismatch decrease and REAQM is less aggressive. Otherwise, REAQM is more aggressive. The rational behind it is that smaller the queue size, the more rate mismatch is allowed. This makes the tradeoff between system stability and utilization. if the current input rate exceeds the link capacity, packet marking probability will increase. Otherwise it will decrease. The larger the current queue length, the larger the co efficient of rate mismatch is; thus the larger the price value increases or decreases as well. REAQM decouples packet loss and congestion measure. In wireless packets are lost mainly because of bit errors and intermittent connectivity. Simulation results demonstrate that REAQM is capable of performing well for TCP flows over both wired and wireless links.

Although the rate based AQM schemes can obtain good transient performance because of its fast responsiveness, large queuing delays jitter may occur by the reason of no explicit control mechanism of queue size under dynamic network scenarios especially

5. AQMs Based On Queue Length And Load Merits

5.1Random Exponential Marking (REM)

REM [34] is an active queue management mechanism that aims to achieve both high utilization and negligible loss and delay in a simple and scalable manner. The key idea is to decouple congestion measure from performance measure such as loss, queue length or delay. While congestion measure indicates excess demand for bandwidth and must track the number of users, performance measure should be stabilized around their targets independently of the number of users. Simulation results suggest that this goal seems achievable without sacrificing the simplicity and scalability of the original RED. This property can be exploited to improve the performance of TCP over wireless links.

5.2 Stabilized Virtual Buffer (SVB)

Like REM, Stabilized Virtual Buffer (SVB) [35] considers both the packet arrival rate and queue size to stabilize them around target value but unlike REM it maintains a virtual queue and responds to the traffic dynamics faster for better stability, especially in the presence of short flows. While the virtual concept queue is similar to AVQ, this SVB considers both arrival rate and queue length. Unlike AVQ where the service rate of the virtual queue is adaptable and packet is dropped/marked whenever the virtual queue overflows the physical buffer limit, in SVB the service rate is fixed as link capacity of the real queue and adapt the limit of the virtual buffer to the packet arrival rate. Another difference with respect to AVQ is that the incoming packets in SVB are then marked with a probability, which is calculated based on both the current virtual buffer limit and virtual queue occupancy. The performance of SVB is compared with other AQMs and the result is given in table 4 with the following parameters for various AQMs.

N = 200; Number of connections
RED : Min_{th} = 20 packets,
Max_{th} = 80 packets,
Q-weight = 0.002, Max_p= 0.5

REM : Φ=1.005, γ=0.001, α=0.1, T=8ms,
Target queue length =50packets

AVQ : γ = 1.0
SVB : b=45

Table 4 Performance comparison of SVB with other AQMs (Reprinted from Ref 35)

AQM	Loss rate	Good put	Avg q (packets)	STD q (packets)
Drop Tail	11.8%	1249.2	92.7	8.7
RED	13.1%	1245.8	42.1	26.0
REM	13.6%	1237.0	40.1	28.3
AVQ	14%	1218.0	10.5	11.8
SVB	13.2%	1249.2	51.4	5.9

5.3 Active queue management algorithm considering queue and load states

Jaesung Hong et.al suggest another AQM that predicts the average queue length and controls it to maintain a certain reference value to achieve high link utilization and low queueing delay by considering both the average queue length and the estimated packet arrival rate.[36] The algorithm uses two functions to calculate drop probability; One for averaging the queue length and the other for estimating the rate of queue change. The authors have showed that, with different values of Q_{max}, this algorithm keeps the average queue length constantly at the provided reference value Q_{ref}, regardless of traffic loading. It also has smaller deviation, which also gradually decreases with the traffic load and buffer size. It assures the queueing delay by reducing jitter under dynamic traffic load.

5.4 RaQ

Another mechanism called RaQ [37] uses the input rate and current queue length to calculate the packet dropping/marking probability. From the point of control theory, RaQ can be seen as dual loop feedback control. The inner loop is rate feedback and outer loop is queue length feedback control. Thus the rate feedback control enables RaQ to respond congestion quickly, so that it can decrease the packet loss due to buffer overflow, and queue length feedback control stabilizes RaQ's queue length around given target. So it can achieve predictable queueing delay and lower delay jitter.The simulation results show that RaQ is able to maintain queue length around the given target under different traffic loads, different RTPTs, and different bottleneck link capacities. Further simulation testing involving non-TCP traffic types and a multiple bottleneck topology have further confirmed the robustness of RaQ Comparison showed that the superiority of RaQ in low packet loss, achieving faster convergence to target queue length and then maintaining the queue length closest to the target .

From the analysis of the literature [38],the AQM schemes of this category can be viewed as a PID type controller, which essentially belongs to queue based category

6.**DISCUSSION**

Even though many mechanisms have been developed since 1999,a few of them are considered in this study. The goals of AQM are to maintain a stabilized queue, to achieve high resources utilization and lower queueing delay. The lock out and Full queue problems of tail drop mechanism are the issues that are being considered while developing any new AQM mechanisms. RED is the most widely employed AQM and tuning of parameters of RED is the main goal of almost all newly developed AQMs. Several AQMS like SRED, DSRED, MRED, Adaptive RED, ARED, RARED etc., were developed on this basis. The SRED does not calculate the average queue length. If further investigations on SRED has shown the way for improving the performance of SRED by the addition of computation of average queue size, then that would be beneficial. Even though the performance of MRED was slightly better than RED, it received only less attention. AVQ achieves good utilization while keeping queue length small. On the other hand CHOKe provides much better fairness but fails to keep the utilization as high as AVQ. The time varying link utilization factor of SAVQ improves the transient response of SAVQ, which outperforms AVQ, and the queue length of SAVQ converges fast to the desired length. Also the introduction ofarrival rate at the network as principal measure of congestion and the desired link utilization ratio as the subordinate measure of congestion in EAVQ utilize the link capacity hundred percent. Similar to EAVQ, Yellow also uses two measures to manage congestion. Very high link utilization is achieved keeping load factor as the main merit and queue control function as the secondary merit. AVQRED essentially combines AVQ and RED and enhances the way the virtual capacity is adjusted to adapt to dynamics of gateway resources. AVQRED's link utilization ARED, BLUE, GKVQ, AVQ and PI. AVQRED 's packet loss rate is upto 15% lowered and the standard deviation is 28 to 50% lower than the other AQM methods except for BLUE. The low drop standard deviation implies that the distance between drops is more uniform resulting in less consecutive drops. AVQRED's queueing delays are 25% lower than the other AQM methods while the delay jitters are about 5 to 15% higher than RED. Low delay jitters are not always good because it means that short lived and bursty traffic could be dropped. Overall performance of AVQRED is 24.2% higher than GKVQ and 24.1% higher than AVQ and 25% higher than PI. ARED, PI and REM all exhibit good network performance, however the differences observed are not significant enough to really distinguish these from each other. ARED has benefited from discussions in the literature on the settings of parameters for RED while others received much less attention. If ECN is not supported, ARED operating in byte mode was the best performing design, providing better response time performance than drop tail queuing at offered loads above 90% of link capacity. However ARED operating in Packet mode with or without ECN was the worst performing design, performing worse than drop-tail

queueing. ECN support is beneficial to PI and REM .With ECN, PI and REM were the best performing designs, providing significant improvement over ARED operating in byte mode. In the case of REM, the benefit of REM was dramatic. Without ECN, response time performance with REM was worse than drop-tail queueing at all loads considered. Whether or not the improvement in response times with AQM is significant, depends heavily on the range of round trip times (RTTs) experienced by flows. As the variation in flows' RTT increases, the impact of AQM and ECN on response time performance is reduced.

In many cases the simulation scenarios presented by the developers of a AQM mechanism concentrate on a few general scenarios and are often too simple to capture protocol behavior in non-standard situations. Claims about a mechanism that are based purely on simulation results should be taken with a grain of salt. Traffic conditions in the Internet are too complex to be modeled in all aspects in a network simulator, making it important to evaluate AQM mechanisms also under real–world conditions

6.1 The impact of Unresponsive flows on AQM performance

While Unresponsive flows contributing to about 70-80% of the Internet flows, account for only 10-20% of its byte volume. This small volume of Unresponsive flows can significantly impact on transient behavior of AQMs. Short lived TCP flows can dominate the dynamic of traffic increase when congestion is low and long

lived TCP flows dominate the dynamic of traffic decrease .The mean sending rate of unresponsive flows reduces the bandwidth available to long lived TCP traffic, which in turn makes the AQM more robust, but less responsive. Queue averaging is the issue that deals with an AQM's response to variation in Unresponsive traffic. It results in a tradeoff between AQM responsiveness, robustness and response to the uncontrolled flows. For robustness the queue averaging time constant should be chosen outside the range $(R,R^2C/N)$ where R is the round trip time is link capacity and N is the number of active TCP flows. AQM responsiveness is inversely related to the queue averaging time constant. It is also impossible via selection of the averaging time constant, to sufficiently and simultaneously smooth the variations in queue length and loss probability due to variations in Unresponsive flows. This is very important as it implies that while averaging results in a smooth or stable congestion feedback, it also introduces considerable jitter in the queueing delay. That trade off should be noted in AQM design

7. CONCLUSION

In this paper , we presented a survey on recent advances in the area of active queue management .The implementation of AQM is beneficial in a general network environment. Further we classified the mechanisms according to the type of metrics they used as congestion measure. From the survey we found that the performances of rate based AQM schemes are better than that of queue based schemes. The queue length of rate based scheme is less sensitive to the number of TCP connections than that of queue based schemes. Inclusion of more number of congestion measures in the existing rate based schemes such as AVQ,EAVQ may result in better performance in terms of , throughput, packet loss, link utilization.

References

[1] Braen,B., Clark,D.,et.al "Recommendations on queue management and congestion avoidance in the Internet" IETF RFC (Information)2309.April 1998

[2] Joerg Widmer., Robert Denda., Martin Mauve Prakitsche Informatik IV.,"A Survey of TCP Friendly Congestion Control"., IEEE Transactions on Network, May/June 2001

[3] E.Hashem,"Analysis of Random drop for gateway congestion Control"report LCS, TR-465,Laboratory for Computer Science, MIT ,Cambridge ,MA,1989,p 103

[4] T.V. Lakshman, A. Neidhardt, T. Ott, The Drop From Front Strategy in TCP Over ATM and Its Interworking with Other Control Features, Proc. Infocom 96, pp. 242- 1250

[5] S.Floyd,V.Jacobson,"Random Early Detection Gateways for congestion Avoidance" ,IEEE/ACM Transaction on Networking ,August 1993

[6] W.Feng, D.D Kandlur,D.Saha, K.G.Shin,"A Self Configuring RED Gateway", Proceedings of IEEE INFOCOMM, 1999,Vol 3 pp 1320-1328

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

[8] Mark Parris, Kevin Jeffay ,F. Donelson Smith "Lightweight Active Router-Queue Management for Multimedia Networking" Multimedia Computing and Networking *1999*, Proceedings, SPIE Proceedings Series, Volume 3654, San Jose, CA, January 1999, pages 162-174.

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

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

[11] 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

[12] 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

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

[14] Chonggang Wang ., Bin Liu., Y.Thomas Hou., Kazem Sohraby., Yu Lin.,"LRED: A Robust Active Queue Management Scheme Based On Packet Loss Ration" 23rd Annual Joint conference of IEEE Computer and Communication Societies INFOCOM,March 2004

[15] Mark Claypool., Robert Kinicki., Mathew Hartling.," Active Queue Management for Web Traffic"IEEE International Conference on Performance, Computing and Communication 2004

[16] Liujia Hu., Ajay D.Kshemkalyani., "HRED:A simple and Efficient Active Queue Management Algorithm" 13th International Conference on Computer Communications and Networking ICCCN 2004,October 2004

[17] Yue-Dong Xu., Zhen-Yu Wang., Hua Wang.,"ARED:A Novel Adaptive Congestion Controller" IEEE International Conference on Machine Learning and Cybernetics,August 2005.

[18] Tae-Hoon Kim., Kee-Hyun Lee" Refined Adaptive RED in TCP/IP Networks" ,IEEE ICASE ,October 2006

[19] Jeong-Hwan Seol,Ki Young Lee., Yoon Sik Hong " Performance Improvement of Adaptive AQM Using the Variation of Queue Length" IEEE Region 10 Conference TENCON, November 2006

[20] Tetsuji Yamaguchi.,Yutaka Takahashi ., " A queue Management algorithm for fair bandwidth allocation", Computer Communications ,April 2007

[21] Pan,R., Parbhakar,B., and Psounis,K.,"CHOKe, a Stateless Active Queue Management Scheme for Approximating Fair Bandwidth Allocation", IEEE INFOCOMM,Feb 2000

[22] Shan Chen, Zhen Zhou ., Brahim Bensaou., "Stochastic RED and its applications" ICC 2007

[23] Wu –Chung Feng., Kang G.Shin., Dilip D.Kandlur., Debanjan Saha .,"The BLUE Active Queue Management Algorithms" IEEE ACM Transactions on Networking, August 2002

[24]]A.Kamra, S.Kapila,V.Khurana, V.Yadhav, H.Saran,"SFED: a rate control based active queue management discipline,IBM India research laboratory Research Report ,available online fromhttp://www.cse.iitd.ernet.in/srajeev/publications.htm

[25] Abinav Kamra., Huzur Saran., Sandeep Sen., Rajeev Shorey " Fair Adaptive Bandwidth allocation: a rate control based active queue management discipline" ,Computer Networks ,July 2003

[26] Srisankar S.Kunniyur., R.Srikant " An Adaptive Virtual Queue (AVQ) for Active Queue Management"., IEEE/ACM Transactions on Networking, April 2004

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

[28] Qian Yanping, Li Qi, Lin Xiangze, Ji Wei," A Stable Enhanced Adaptive Virtual Queue Management Algorithm for TCP networks" May30 to June 1,2007, IEEE International Conference on Control and Automation

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

[30] Manoj K.Agarwal., Rajeev Gupta., Vivekanad Kargaonkar., " Link Utilsation Based AQM and its Performance", IEEE Communications Society ,Globecom 2004,December 2004

[31] Chonggang Wang., Bo Li., Thomous Hou., Kazem Sohraby., Keping Long.," A stable rate-based algorithm for active queue management" Computer Communications, January 2005

[32] Chin-Ling Chen., Jia-Chun Yu.,"A Proportional Rate-based Control Scheme for Active Queue Management" IEEE International Conference on Electro Information Technology, May 2005

[33] Jun Wang., Min Song., Houjun Yang " Rate- Based Active Queue Management for Congestion Control over Wired and Wireless Links"..,(Invited Paper) IEEE 2006

[34] Athuraliya., D.E Lapsley., S.H Low" Random Exponential Marking for internet congestion control" IEEE Transactions on Network, June 2001

[35] Xidong Deng., Sungwon Yi., George Kesidis., Chita R.Das., " Stabilised Virtual Buffer (SVB)-An Active Queue Management Scheme for Internet Quality of Service",IEEE Globecom November 2002.

[36] Jaesung Hong., Changhee Joo., Saewoong Bahk .," Active queue management algorithm considering queue and load states" , Computer Communications, November 2006

[37] Jinsheng Sun., Moshe Zukerman ., " RaQ: a robust active queue management scheme based on rate and queue length", Computer Communications, February 2007