A Study on an Adaptive AQM Using Queue Length Variation

Jeong-Hwan Seol, and Ki Young Lee, *Member, KIMICS*

Abstract— The AQM (Active Queue Management) starts dropping packets earlier to notify traffic sources about the incipient stage of congestion. The AQM improves fairness between response flow (like TCP) and non-response flow (like UDP), and it can provide high throughput and link efficiency. In this paper, we suggest the QVARED (Queue Variation Adaptive RED) algorithm to respond to bursty traffic more actively. It is possible to provide more smoothness of average queue length and the maximum packet drop probability compared to RED and ARED (Adaptive RED). Therefore, it is highly adaptable to new congestion condition. Our simulation results show that the drop rate of QVARED is decreased by 80% and 40% compare to those of RED and ARED, respectively. This results in shorter end-to-end delay by decreasing the number of retransmitted packets. Also, the QVARED reduces a bias effect over 18% than that of drop-tail method; therefore packets are transmitted stably in the bursty traffic condition.

Index Terms— AQM, RED, ARED, Queue Length Variation

I. INTRODUCTION

In the recent years, user's requests are increasing daily as well as the field is very wide in the sphere of internet application services. Current internet offers best-effort service for users, however, existing high-speed network environment have a low delay, broad bandwidth and sensibility for instantaneous congestion. Therefore, it is difficult to provide QoS (Quality of Service) at user's desire. To control the congestion, many approaches have been proposed, they can divide into two branches that are host-based (sender and receiver) and router-based schemes. Such host-based schemes as slow start and congestion avoidance in the TCP have a problem that these can't accommodate bursty traffic because the methods are performed after detecting network congestion. Therefore, a congestion control based on router that is the middle node is necessary.

AQM (Active Queue Management), is a router-based mechanism, provides high-throughput without queuing delay at a router.[1] RED (Random Early Detection), as representative scheme of AQM, is the buffer management algorithm that proposed by S.Floyd. However, because RED algorithm uses fixed parameters, it has a demerit that overall network performance is affected by setting parameters. There are a lot of variants of RED to solve this problem.[2][3]

FRED (Fair RED) uses per-active-flow accounting to impose, on each flow a loss rate that depends on the flow's use of buffer space. SRED (Stabilized RED) proposed to stabilize queue using the step drop function which used activity flow. WRED (Weighted RED) classifies traffic into several groups and prioritizes as packet drop probability and weight are applied to each group. It uses the priority to drop packets. ARED (Adaptive RED) automatically sets several other RED parameters; operators need only set the desired target average queue length [4][5][6]. Among them, ARED retains RED's basic structure and merely adjusts the parameter max_p to keep the average queue size between min_{th} and max_{th}. Moreover ARED reduces both the packet loss rate and the variance in queuing delay. However, ARED does not come up to expect to solve the problem, susceptibility to parameters.[5]

In this paper, we propose QVARED which applied variant of queue, as distinct from previous AQM. This algorithm enhanced link efficiency and dynamically dropped packets on applications due to have high throughput in case of bursty traffic.

II. RELATED WORK

A. RED (Random Early Detection)

The method of drop-tail is currently used for management in router. When the buffer is full, incoming packets are abandoned by the process. However, it could have long delay as a bottleneck state if the congestion is lasted for a long time. To make up for the defect, a mechanism of AQM is developed and the method let the source detect congestion early and drop packets before buffer overflow. RED is representative method from among AQM mechanism.[7]

The goals of RED algorithm are to raise link occupancy rates as it reduces both packet loss and queuing delay and to avoid global synchronization. In RED, average queue length (*avg*) is calculated using EWMA (Exponential Weighted Moving Average) as follow formula (1).[1]

Manuscript received December 13, 2007.

Jeong-Hwan Seol and Ki Young Lee are with the Department of Info & Telecom Engineering, University of Incheon, Incheon, 402-749, Korea (Tel: +82-32-762-5754, Email: kylee@incjheon.ac.kr)

$$avg_{(t)} = (1 - w_q)avg_{(t-1)} + w_q q_{(t)} \quad (1)$$

In (1), $avg_{(t)}$ is current average queue length and it shows that queue length at present is affected by former length. w_q is queue weight; the optimized value is 0.002.

for each packet arrival
calculate the average queue size *avg*
if $min_{th} \le avg \le max_{th}$
calculate probability P_a
with probability P_a:
mark the arriving packet
else if $max_{th} \le avg$
mark the arriving packet

Fig. 1. RED's general algorithm

Fig. 1 describes typical RED algorithm, considered avg which calculated by (1), in RED. When avg gets the value between min_{th} (minimum threshold for queue) and max_{th} (maximum threshold for queue), packets are dropped randomly with the probability of P_a which computed by (2).

$$P_a = \frac{avg - \min_{th}}{\max_{th} - \min_{th}} \cdot \max_p \quad (2)$$

p_a is packet-marking probability and max_p is the maximum value for p_a. When the avg is less than the min_{th}, no packets are marked. When the avg is greater than the max_{th} - maximum threshold, every arriving packet is marked. However, when the link is lightly congested max_p is high, the average queue size is near min_{th}; when the link is more heavily congested max_p is low, the average queue size is closer to, or even above, max_{th}. As a result, the average queuing delay from RED is sensitive to the traffic load and to parameters, and is therefore not predictable in advance. RED's main weakness is that the average queue size varies with the level of congestion and with the parameter settings. That is, RED cannot be active because performance is influenced according fixed parameters. In addition, RED's another weakness is that the throughput is also sensitive to the traffic load and to RED parameters. ARED is one of the suggested algorithm to resolve the problems of RED.

B. ARED (Adaptive RED) Algorithm

The performance of RED is controlled by not only traffic load, but also setting parameters. If it is inadequate to set parameters, the efficiency could not come up to drop-tail. ARED (Adaptive-RED), which was suggested by S.Floyd[5], can alleviate the problems of variable delay and parameter sensitivity; the max_p is restricted within the limit of [0.01, 0.5]. The average queue length is kept as avg is within target range.[8][9]

ARED does self-tuning parameters according to the present state of traffic. ARED aims to anticipate mean delay with limit of avg in target range. When current avg be over target range, packets are dropped up to be less than average queue size. In other hands, when avg is below the target range, it is put off to drop packets in order to increase avg.[4] In addition, max_{th} is set three times of min_{th} so that pertinent value of min_{th} is set at initial configuration. However, adaptability could reduce if congestion persists on mass real-time traffic. After all, impertinent configuration of target range causes a problem thus it is necessary to consider other approaches besides setting parameters.

III. PROPOSED QVARED ALGORITHM

A. Consideration based on the variation of queue length per unit time

Whenever a new packet comes in the router, as for the RED, it is calculated average queue length. It sets up a new weight parameter about current queue length, and it is calculated new queue length [1].

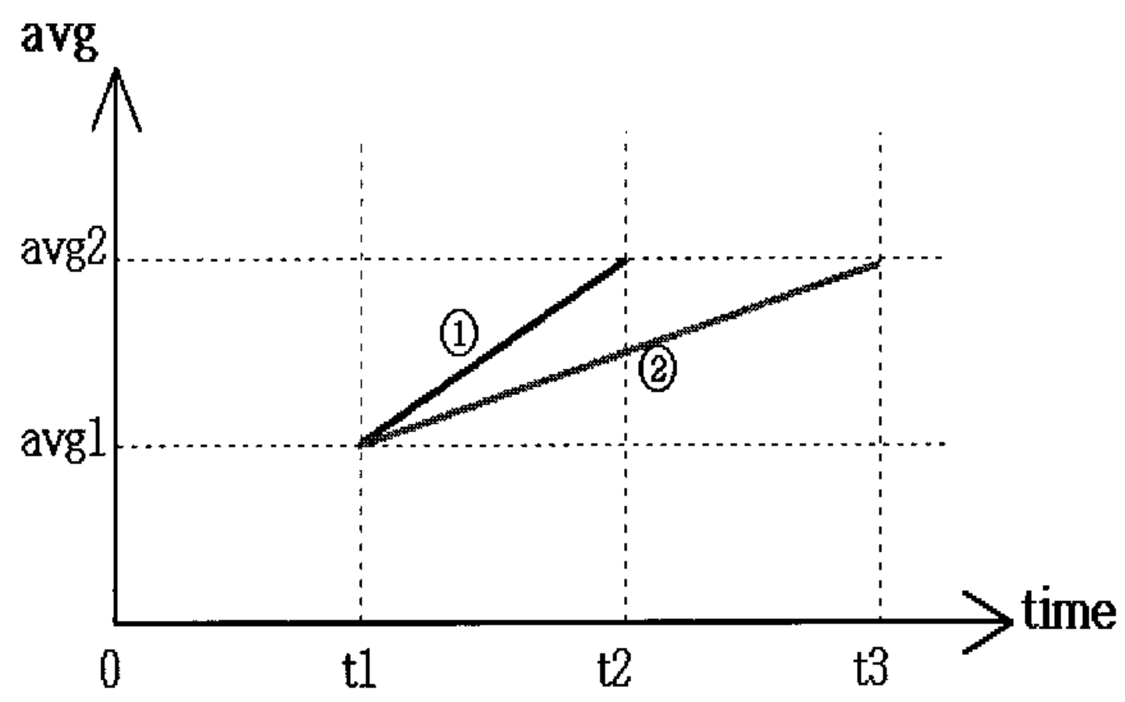


Fig. 2 The variation of queue length per time.

For Fig. 2, the previous RED recognizes ① and ② to be the same event. A value of the avg which was newly calculated is same in current time t1 whether time of t3-t1 passes and becomes avg2 whether time of t2-t1 passes, and avg2 works when queue length is avg1. This is the result that did not consider time. We considered it by a part about time, and calculation did packet marking probability. While variation of queue length has a positive value per hour, and if the price is large, it can be called bursty traffic. If this price is a positive value, it raises packet marking probability, and it reduces packet marking probability if it is a negative value. In this paper, we drop of a packet with half probability of the time when queue length was increased with probability.

B. QVARED (Queue Variation ARED)

Previous RED and ARED drop packets without considering change of current queue length. In this paper, we suggest QVARED that is applied basic structure of ARED: to calculate max_p, ARED sets up target range and dose self-tuning. When there is a queue during decrease, QVARED dispose of a packet with half probability of the time that queue length is increased. Fig. 3 shows the drop probability of proposed algorithm and detailed algorithm is in Fig. 4.

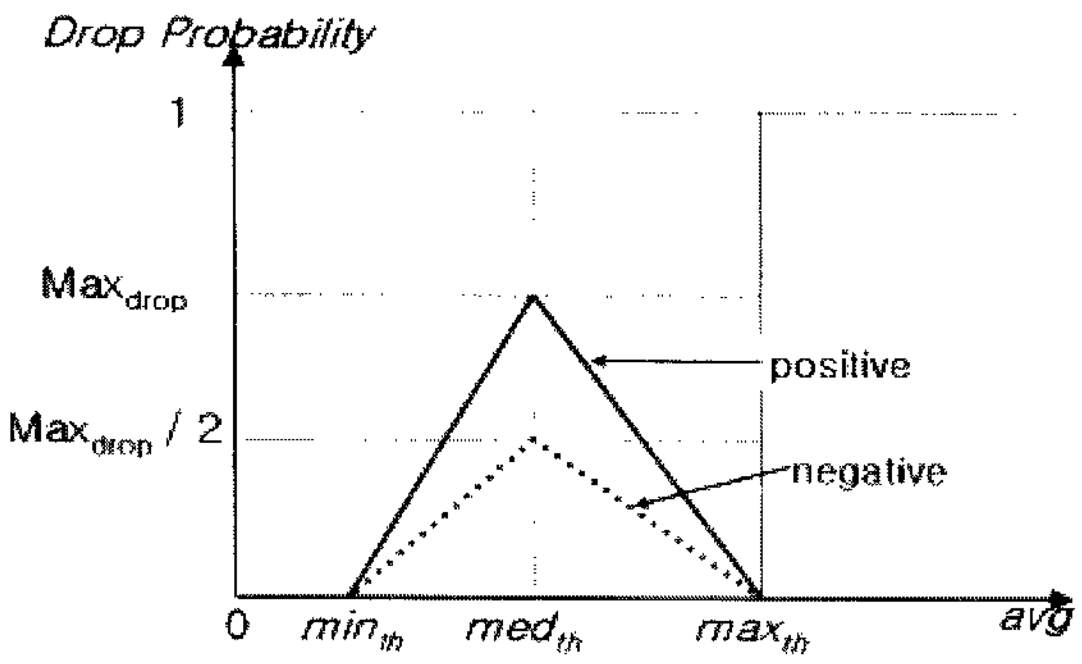


Fig. 3 QVARED's drop function

$q_t \leftarrow 0$, $q_{tmax} \leftarrow 0$
$avg_{prev} \leftarrow 0$, $time_{prev} \leftarrow 0$
calculate $q_t \leftarrow (avg - avg_{prev}) / (time - time_{prev})$
$q_{tmax} \leftarrow max(q_t, q_{tmax})$
$avg_{prev} \leftarrow$ avg, $time_{prev} \leftarrow$ time
if $min_{th} < avg < max_{th}$
increment count
calculate probability Pa:
if $avg < med_{th}$
$Pb \leftarrow max_p (avg - min_{th}) / (med_{th} - min_{th})$
else
$Pb \leftarrow max_p (max_{th} - avg) / (max_{th} - med_{th})$
if $q_t < 0$ $Pb \leftarrow Pb / 2$
if $q_{tmax} > 0$ $Pb \leftarrow Pb + Pb * q_t / q_{tmax}$
$Pa \leftarrow Pb / (1 - \text{count} * Pb)$

Fig. 4 QVARED's algorithm

IV. SIMULATION RESULTS

We conducted out simulation based on the ns-2 network simulator [10] and network topology is shown in Fig. 5.

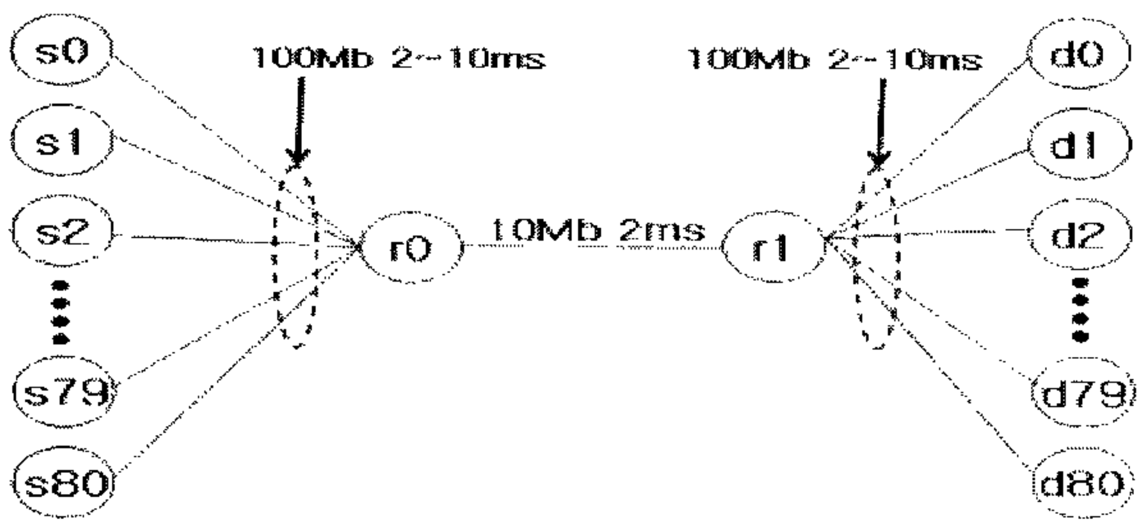


Fig. 5 Simulation topology

The substitution bandwidth between S1, S2... S80 and R0, R1 and D1, D2... D80 held the 100 Mbps, the delay with 2~10ms, and the substitution width of R0 and R1 did the 10 Mbps, the delay with 2ms and run for 100 seconds. We used a single bottleneck topology in each simulation and compared with the RED, ARED, and QVARED. In RED, we fixed max_{th} and min_{th} at 80 and 20 packets respectively, med_{th} (medium threshold for queue) at 50 packets, max_p at 0.02. In ARED, we fixed target range at 50 packets. In this study, we did not consider change of packet size and fixed average packet size at 1000 bytes at TCP nodes.

In first simulation, throughput of RED, ARED and QVARED measured by (3) separately and the results are shown in Fig. 6.

$$Throughput = \frac{Packet\ time\ to\ have\ sent\ successfully}{Total\ simulation\ time} \quad (3)$$

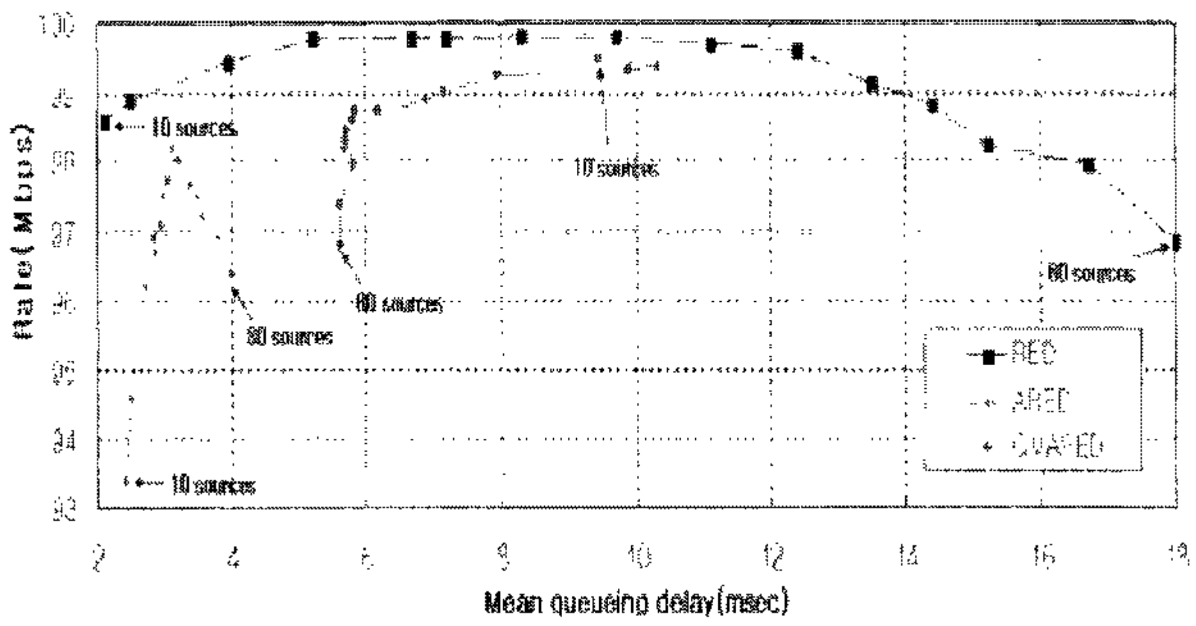
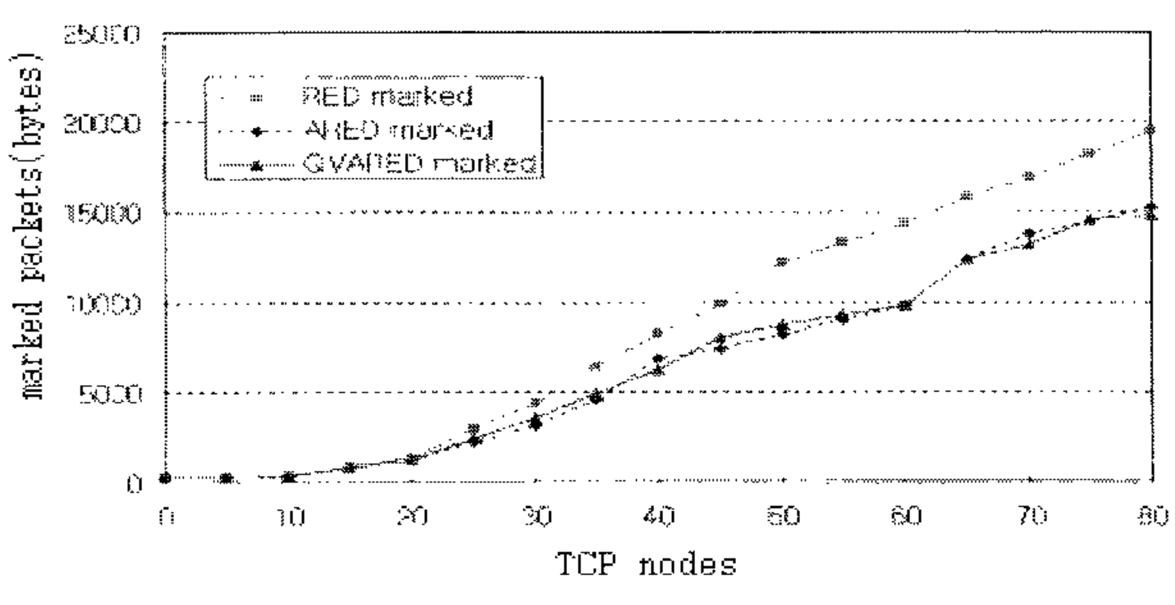


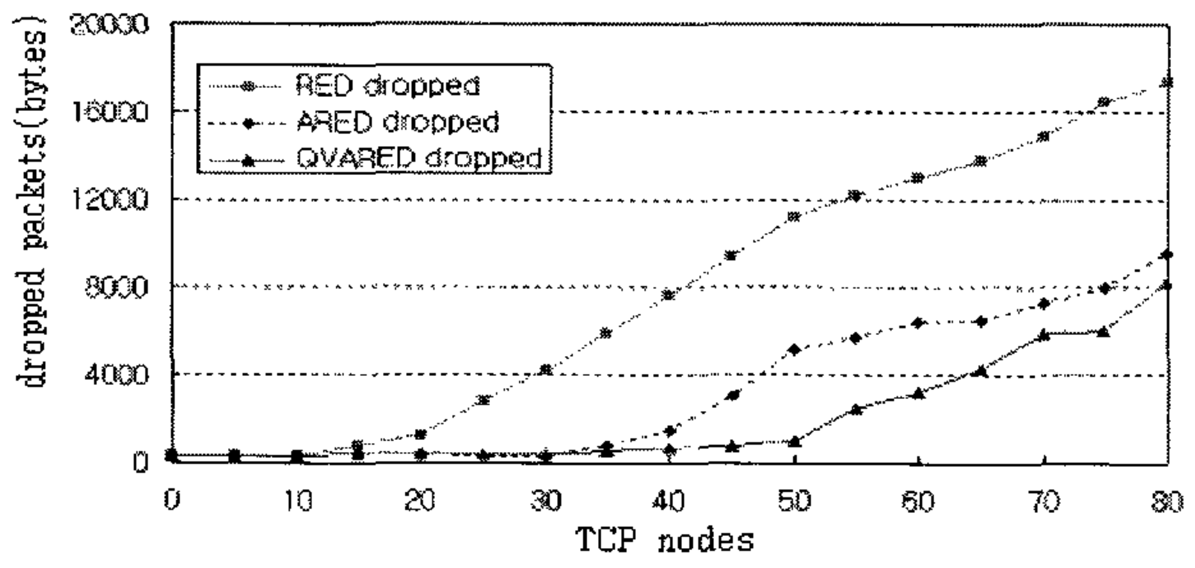
Fig. 6 The result of throughput

Although link and processing rate are superior in RED, queuing delay increases linearly by TCP sources. Fig. 6 shows that queuing delay of proposed QVARED algorithm is much less than one of RED and that QVARED is more efficient than the other. QVARED works so as to reduce queuing delay account for TCP nodes growth because the algorithm tries to keep link stability for bursty traffic how to decrease queuing delay account for variation of queue.

In next simulation, we counted marked and dropped packets among incoming packets at a router in order to find congestion and the result is shown in Fig. 7.



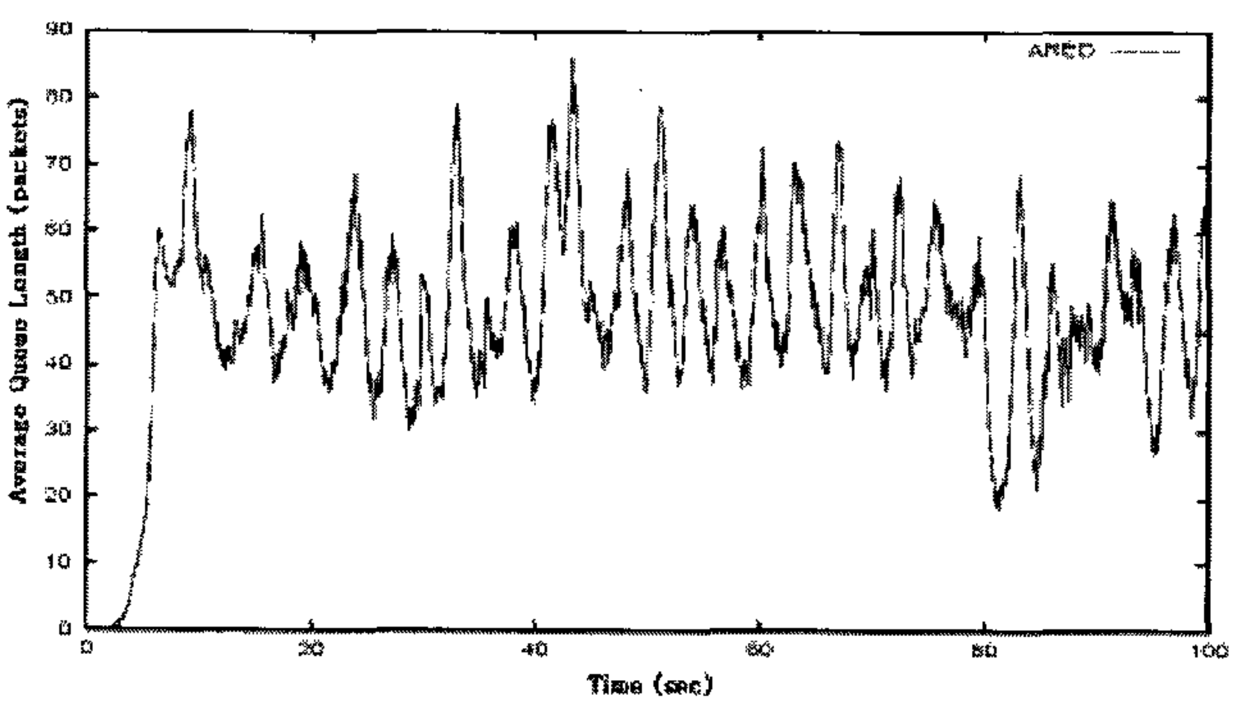
(a) The number of marked packets



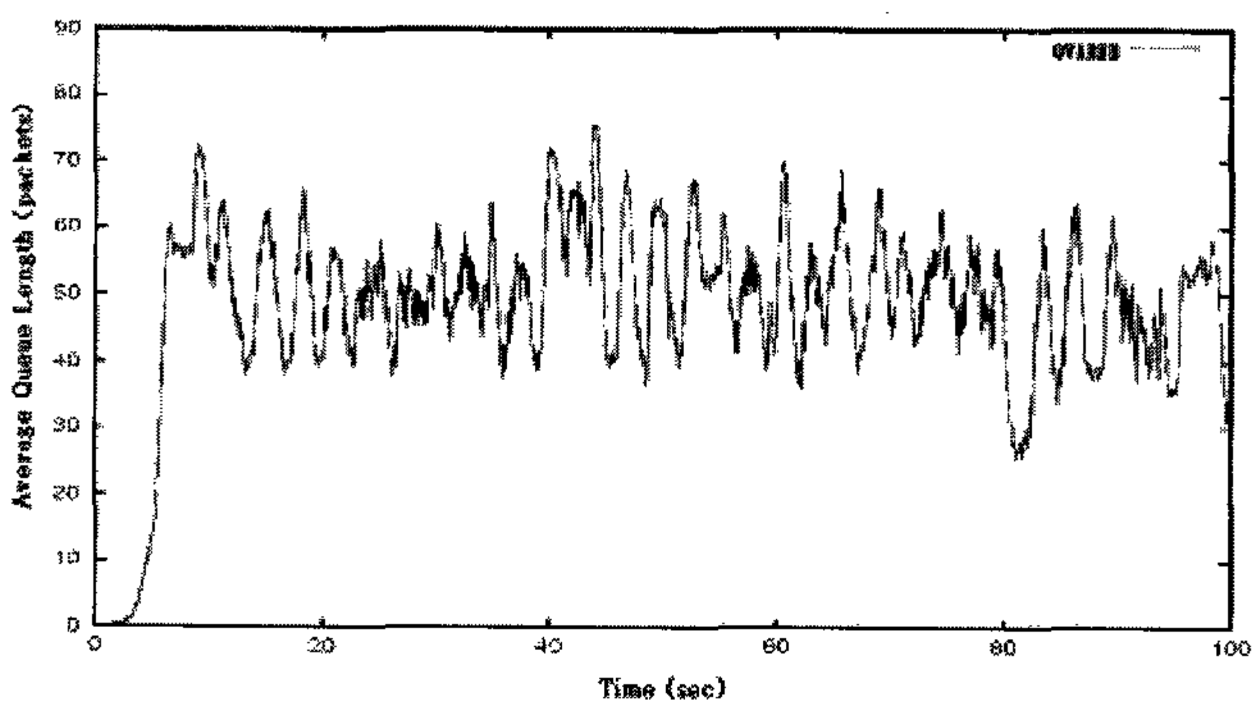
(b) The number of dropped packets

Fig. 7 The number of marked / dropped packets

While TCP nodes increase, dropped packets also increase because of the bottleneck by incoming packets at a router and it is difficult to deal with bursty traffic. In Fig. 7(b), QVARED reduce about 40%, 80% of the drop rate than RED and ARED respectively. The less packets drop, the better because packets are retransmitted less. 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. We compared variation of avg with ARED and QVARED; the result is displayed in Figure 8.



(a) ARED



(b) QVARED

Fig. 8 The average queue length(*avg*) of ARED and QVARED

In ARED, average queue length changes close to target range, fixed 50 packets. As shown in Fig. 8(b), but the wave of QVARED is similar to ARED, it is more smoothness at a change of avg by reason of considering variation of queue. That is, it means improvement in stability and in adaptability to congestion

Fig. 9 shows how bias against bursty traffic is in drop-tail, RED, ARED and QVARED. We measured marking rate (actual rate) and idle rate (fair rate) and represented the proportion of them. The more this ratio increase, the more stable packets are transmitted against bursty traffic.

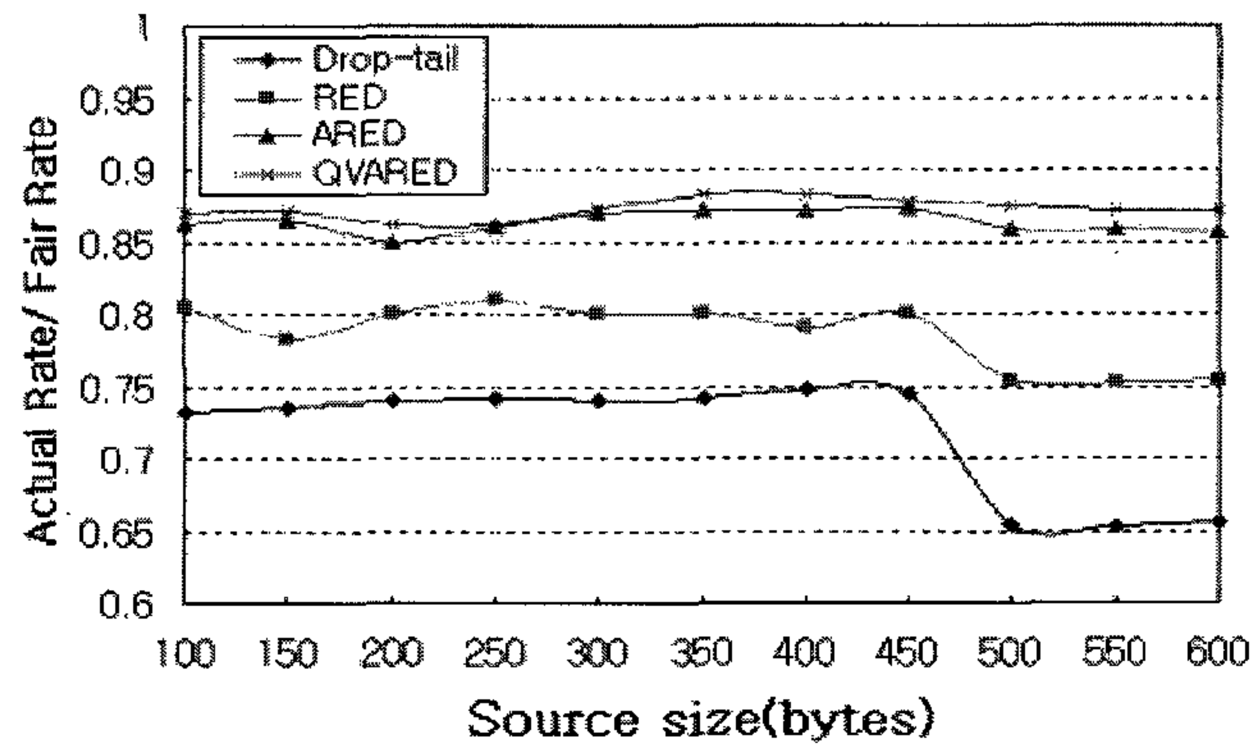


Fig. 9 Bias against bursty traffic

The average of QVARED is about 0.871 and it indicates that the algorithm is more stable than the other even if buffer size is over 500 bytes.

IV. CONCLUSIONS

In this paper, we propose QVARED algorithm which applied variant of queue, as distinct from previous AQM. We applied the variation of queue length which is not considered in current RED and ARED algorithm, and as the result the dropping process at the router can make more actively. From our simulation result, we are able to confirm that QVARED provide good performance on link efficiency and reducing bias. We realized that the existing RED and ARED algorithms still need to be improved even if they are effective and very simple to implement.

Our simulation results show that the drop rate of QVARED is decreased by 80% and 40% compare to those of RED and ARED, respectively. This results in shorter end-to-end delay by decreasing the number of retransmitted packets. Also, the QVARED reduces a bias effect over 18% than that of drop-tail method; therefore packets are transmitted stably in the bursty traffic condition

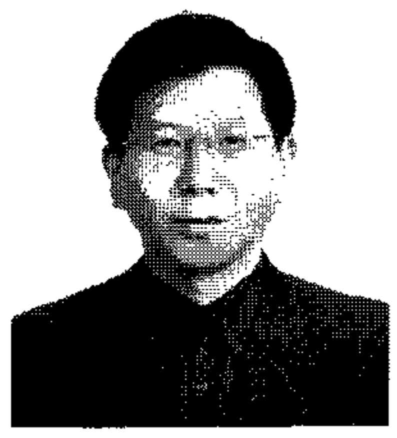
ACKNOWLEDGMENT

This work was supported by the **University of Incheon** Research Grant in 2005.

REFERENCES

[1] S. Floyd, and M. Jacobson, "Random Early Detection Gateways for Congestion Avoidance", IEEE/ACM Transactions on Networking,1(4):397-413, Aug. 1993.
[2] S. Floyd. RED: Discussions of Setting Parameters, November 1997.
[3] http://www.aciri.org/floyd/REDparameters.txt.
[4] Seungwan Ryu and Sung Hee Kim, "Recent Advances in TCP Congestion Control", Telecommunications Review, vol. 13, no.5, October 2003.
[5] S. Floyd, R. Gummadi, and S. Shenker, "Adaptive RED: An Algorithm for Increasing the Robustness of RED's Active Queue Management", http://www.irir.org/floyd/paers /adaptiveRed.pdf.
[6] V. Firoiu, and M. Borden, "A Study of Active Queue Management for Congestion Control", 2000, IEEE INFOCOM.
[7] Mahbub Hassan, and Raj Jain, "High Performance TCP/IP Networking, Prentice Hall", pp.287-298, 2003.12.
[8] V.Paxson M, Allman and W.Stevens, "TCP Congestion Control", RFC 2581, April, 1999.
[9] W. Feng D. Kandlur, D. Saha, and K. Shin, "A Self-Configuring RED Gateway", in Proc. INFOCOM 99, pp.1320-1328, vol. 3, 1999.
[10] The Network Simulator – NS-2. http://www.isi.edu/nsnam/ns.

Jeong-Hwan Seol
Received the B.S degree in Information and Telecom Engineering at University of Incheon, Korea in 2006. Since 2006, he has been an M.S. student in University of Incheon. (Communication and Network Security Lab) The research areas of interest include Internet protocols and sensor network security.



Ki Young Lee
Received the B.S. and M.Eng. degrees in Electrical Engineering from Yonsei University, Seoul, Korea in 1982 and 1984, respectively. And he received M.S. (1987) from the University of Colorado, Boulder and the Ph.D. (1993) from the University of Alabama in Electrical & Computer Engineering. Since 1994, he has been a professor in the Department of Information and Telecommunication Engineering at the University of Incheon. His research interests include Internet traffic control and protocols, ubiquitous sensor networks and network security system.