Improving Internet Quality of Service through Active Queue Management in Routers

Gamal Attiya[1] and Heba El-Khobby [2]

[1] **Dept. of Computer Science and Engineering,
Faculty of Electronic Engineering,
Minoufiya University, Egypt**

[2] **Dept. of Electronics and Electrical Communications Engineering,
Faculty of Engineering, Tanta University, Egypt**

Abstract

The traffic characteristics of real-time and non real-time applications require a certain Quality of Service (QoS) from the Internet in terms of bandwidth, delay, packet loss, fairness and jitter. However, most of the current Active Queue Management (AQM) algorithms at the internet routers do not guarantee QoS for real time traffics such as video and audio. This is because; most of the algorithms handle different packets of different traffics by the same strategy. In this paper, we propose a new AQM strategy to guarantee QoS for real time traffics. The proposed strategy uses three queues at the internet routers, each of which handles a single class of traffic. Where, the arriving packets are queued according to their class type. Additionally, the queued packets are scheduled according to a predefined weight. The proposed algorithm is evaluated and compared with the most recent algorithms by using the Network simulator NS-2.

Keywords: *Active Queue Management, Packet Scheduling, QoS, Multimedia, Congestion Control*

1. Introduction

In the past few years, the Internet is moving from traditional data communication network for transferring text-based messages such as file transfer (FTP), email and web browsing, to an underlying communication network for multimedia applications such as IP telephony, interactive video conferencing, video on demand and online games. However, with the increasing volume of traffic from both traditional and multimedia applications, a serious problem, called congestion collapse, arises [1]. This problem leads to performance degradation particularly for real-time applications.

Over years, continuous efforts are being done and several congestion control mechanisms are being developed to avoid this problem. The congestion control methodologies can be categorized into End-to-End protocols and Active Queue Management (AQM) algorithms. The End-to-End congestion control protocols are implemented at the end systems based on the Transmission Control Protocol (TCP) [2]. The end system's TCP protocol adjusts its sending rate by adjusting its window size using Additive Increase and Multiplicative Decrease (AIMD) method according to the signaled congestion (e.g. packet loss). Several algorithms to the End-to-End congestion control are investigated and evaluated in [3].

On the other hand, the Active Queue Management (AQM) algorithms are implemented at the internet routers, wherein a router signals impending congestion by dropping packets. Queues are used in routers to absorb transient bursts in incoming packet rates, allowing the router sufficient time for packet transmission. When the incoming packet rate is consistently higher than the router’s outgoing packet rate, the queue size will increase, eventually exceeding available buffer space. When the buffer is full, some packets must be dropped. The most well-known active queue management mechanism is Random Early Detection (RED) [4]. RED operates based on an average queue length that is calculated using an exponential weighted average of the instantaneous queue length. It uses the weighted-average queue size and thresholds to detect impending congestion, and randomly drops incoming packets as the average queue size exceeds a minimum threshold. RED drops packets with a probability depending on the average length of the queue. The drop probability increases from 0 to a maximum drop probability as the average queue size increases from a minimum threshold to the maximum threshold. If the average queue size goes above the maximum threshold, all packets are dropped.

The RED was developed to keep the average queue size small, increase throughput, reduce burst loss and global synchronization. However, it is known that RED’s effectiveness is heavily dependent on the setting of its control parameters. Moreover, the average queue length of

RED in steady state depends on the number of active TCP connections. Some of the difficulties in RED configuration are explained in [5]. Several variants of RED have been proposed to address the performance problems of RED. These variants are; Fair Random Early Drop (FRED) [6], Stabilized RED (SRED) [7], Gentle RED (GRED) [8], DS-RED [9], CHOCKe [10], Dynamic-RED (DRED) [11], Adaptive RED [12], MRED [13], Random Early Mark (REM) [14], Green [15], Blue [16], PD-RED [17], LRED [18], HRED [19], ARED [20] and AutoRED [21] and Adaptive CHOKe [22]. Classification and performance evaluation of the different variants of the AQM schemes are presented in [23].

Although many variants were developed, the current Internet only provides best-effort service for non-real-time traffics. But, it does not guarantee quality of service (QoS) for multimedia or real-time traffics [24-26]. This paper presents a new strategy to the RED AQM in order to guarantee quality of service (QoS) for the real-time traffics. The approach uses three queues each of which handles a single class of traffic: video, audio or data. The queued packets are then scheduled according to a predefined static weight. The proposed strategy is investigated and compared with the RED AQM by using the network simulator NS-2 [27-29].

The rest of this paper is organized as follows. Section 2 describes the problem in some details. A description of the proposed algorithm is presented in Section 3. Section 4 presents the simulation result that showing the effect of the proposed approach on the throughput, delay and packet loss of different traffics. Section 5 summarizes our conclusions and future work.

2. Problem Definition

Multimedia applications have different performance constraints than do traditional applications. Traditional applications are very sensitive to lost packets. Multimedia applications, on the other hand, can tolerate some data loss, but are very sensitive to variance in packet delivery, called jitter. In the absence of jitter and packet loss, video frames can be played as they are received, resulting in a smooth play-out. However, in the presence of jitter, the user would see the frozen image of the most recently delivered frame until the tardy frame arrived. The tardy frame would then be played only briefly in order to preserve the timing for the subsequent frame.

The Active Queue Management (AQM) algorithms control packet loss by Early Congestion Notification (ECN) marks to signal congestion. It requires that the end-hosts recognize dropped packets and respond by retransmitting the dropped packets and reducing their rate of transmission. In the Internet, TCP recognizes packet loss as an indicator of network congestion, and reduces transmission rate. Hence, to date, active queue management appears promising since by far the predominant transport protocol on the Internet is TCP.

Traditional applications use TCP to guarantee that lost packets are retransmitted. Unfortunately, detecting and retransmitting lost packets causes considerable jitter, making TCP unattractive to multimedia applications. So, most non-TCP flows (real time traffics) use User Datagram Protocol (UDP) without employing end-to-end flow and congestion control. But, UDP get an unfair share of network bandwidth when there is congestion. This unfairness occurs because many non-TCP flows do not reduce transmission rates while the TCP flows are forced to transmit data at their minimum rates. Even worse, typical active queue management policies apply the same drop rate to each flow. Thus, when UDP coexists with TCP, it induces not only a congestion collapse problem but also an unfairness problem that each flow cannot get the same treatment, causing an unstable Internet and lower link utilization. In this paper, we propose an approach to the RED-AQM to give equal importance to both the throughput and delay requirements of an application.

3. Proposed Algorithm

The main idea of the proposed algorithm is illustrated in Figure 1. As shown in the figure, three queues are used in the Internet router to queue packets that arrive from different sources, where each queue is assigned a single type of traffic. Packets from different sources are first arrived at a classifier which discriminates between different types of traffics. The classifier has to distinguish packets belonging to a traffic type via a certain field in the packet header. As shown in Figure 1, video traffic is classified as high priority flow and inserted into the first queue, audio traffic is classified as medium priority flow and inserted into the second queue, and other flows is classified as low priority flows and inserted into the third queue.

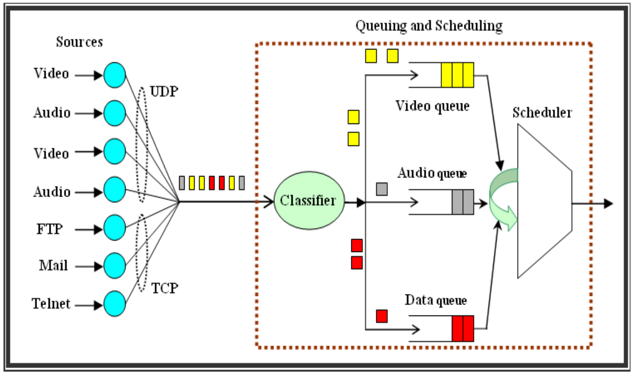


Figure 1: Illustration of Proposed Algorithm

The problem now is how to serve different packets that are queued in the different queues. The simplest solution is the priority based scheduling, where packets in different queues are served with different priorities. In other words, packets in higher priority queues are served first and other packets from a certain queue are scheduled only if the higher priority queues are empty. In addition, with each priority queue, packets are scheduled in FIFO strategy. This method would significantly reduce the delay of the high priority packets (delay sensitive or real-time traffics such as video and audio). However, if the amount of high priority traffic is excessive, the low priority queue may not get any service until the high priority traffic is completely served. In this case, the queues allocated to lower priority traffic (not delay sensitive or non real-time traffics such as FTP and Mail) may overflow. As a result, the low priority traffic may experience a large delay or, in the worst case, a complete resource starvation.

To prevent resource starvation, the service rate of the high priority traffic should be limited. In the proposed approach, each type of traffic is queued in a separate queue and the ratio of service for each type is determined. In other words, a weight is assigned to each queue and packets are scheduled according to the predefined weight. For example, the service ratio may be 3:2:1; this means that three packets are served from high priority video traffic, two packets are served from medium priority audio traffic and one packet is served from low priority traffic (traditional data). The main advantage of this strategy is to allow TCP sources to utilize the available bandwidth efficiently after the high priority traffics consume their available capacity. Indeed, critical time applications overcome the end-to-end latency because they can be served with higher priority. In the proposed approach the service ratio is 1:1:1.

4. Simulation Results

The proposed strategy is coded in C++ and incorporated into the Network Simulator NS-2 to be used as an AQM scheme in the router. Additionally, the performance of the proposed algorithm is evaluated and compared with the RED AQM for handling real time video and audio traffics.

4.1. Simulation Environment

Our design and implementation are carried out in NS-2 [27-29]. It is a popular Wide Area Network simulator developed at the University of California, Berkeley but used by many others for a wide variety of network research. NS supports most of the common IP network components, including TCP (Tahoe, Reno and Vegas) and UDP transport agents, and several queue management mechanisms, including RED. Unfortunately, NS-2 considerably lacks support for multimedia applications, only providing a basic mechanism to build Constant Bit Rate (CBR) media streams. NS does not support streaming Variable Bit Rate (VBR) multimedia, such as an MPEG or Real Video. VBR applications are required for responsive multimedia applications that must maintain their strict timing constraints. Thus, a further contribution of this work is the NS-compatible source code for two flow multimedia applications; video and audio traffics.

Figure 2 shows the network topology and the application flows that used for the simulation. The topology represents a simple network bottleneck configuration. It has two multimedia (MM) sources represent real time video and audio traffics, and six FTP sources represent non real time traffics. The network links are labeled with their bandwidth capacity and propagation delay. Each source as well as each destination is connected through a link has capacity 10 Mb/s and propagation delay 5 ms. The bottleneck link has 6 Mb/s bandwidth and 20 ms propagation delay. For transferring data, the FTP uses TCP connection while the video and audio traffics use UDP connection as the underlying transport agent. All the TCP agents were set to have a congestion window size of 20 packets and a packet size of 1500 bytes. The UDP agents also have a packet size of 1500 bytes and a rate of 1.5 Mbps for video traffic and 384 kbps for audio traffic.

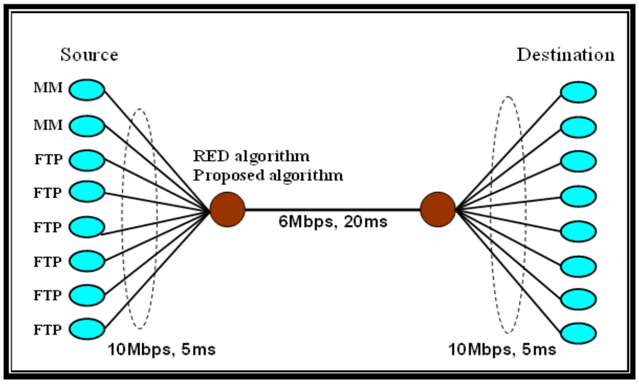
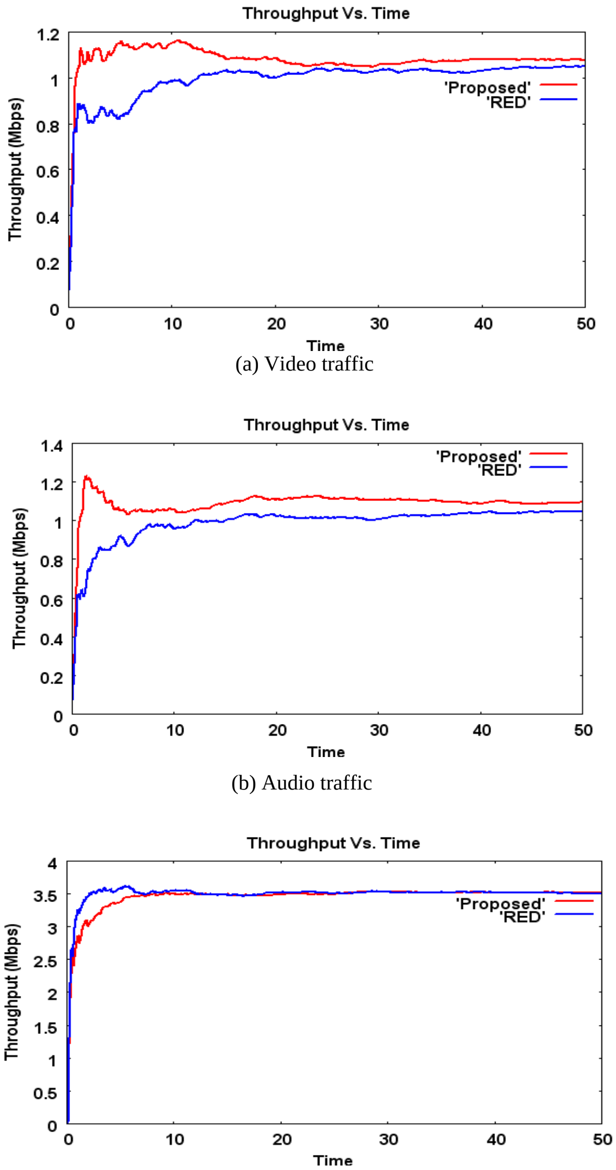


Figure 2: Network Topology and Flows

In the following, we study the effect of the proposed algorithm and the RED-AQM on the throughput, packet loss and the average delay. All measurements were occurred at the router, considering the simulation time 50 sec. The router uses the RED AQM with 20 packets long queue. The parameter settings of the RED are that the minimum threshold = 5 packets, maximum threshold = 15 packets, average queue weight = 0.002 and the probability max_p =0.1.

4.2. Effect of the Algorithms on Throughput

Figure 3 shows the throughput for each type of traffic when using the RED and the proposed algorithm with service ratio 1:1:1. From the figure, it is shown that, the proposed algorithm (*denoted by Proposed*) provides higher throughput than the RED algorithm (*denoted by RED*) for both video and audio traffics. For FTP flows, the throughput when using the proposed algorithm is identical to that of using the RED algorithm. This indicates that the proposed approach improves the throughput of real time traffics without affecting on the non real time traffics.



(a) Video traffic

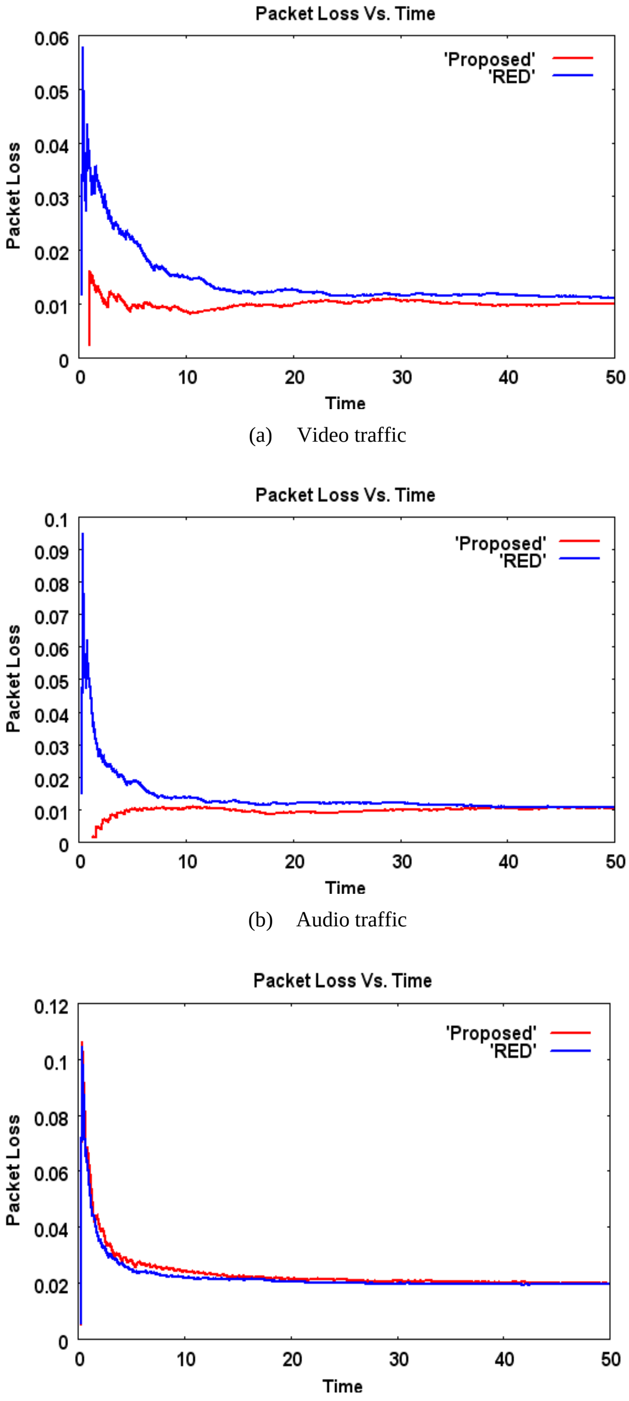
(b) Audio traffic

(c) FTP Data traffic

Figure 3: Throughput vs. time for the RED and the proposed algorithm.

4.3. Effect of the Algorithms on Packet Loss

Figure 4 shows the packet loss ratio for each type of traffic when using the RED and the proposed algorithm with service ratio 1:1:1.



(a) Video traffic

(b) Audio traffic

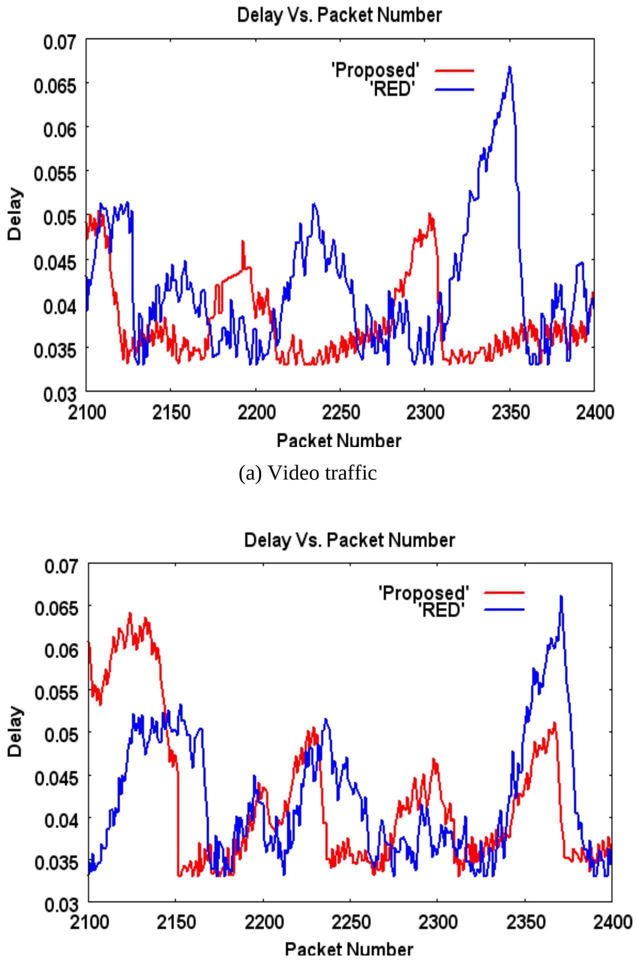
(c) FTP data traffic

Figure 4: Packet Loss vs. time for the RED and the proposed algorithm.

From the figure, it is shown that, the packet loss ratio when using the proposed algorithm (*denoted as Proposed*) is less than the ratio of the packet loss of the RED algorithm (*denoted as RED*) for both the real time video and audio traffics. For FTP flows, the packet loss ratio remains identical to the RED algorithm. This indicates that the proposed approach decrease the packet loss of real time traffics without affecting on the non real time traffics.

4.4. Effect of the Algorithms on Delay

Figure 5 shows the packet delay experienced by each type of traffic when using the RED and the proposed algorithm with service ratio 1:1:1. From the figure, it is shown that, the average delay experienced by the proposed algorithm (*denoted as Proposed*) for both Video and Audio traffics is less than that of the RED algorithm (*denoted as RED*).



(a) Video traffic

(b) Audio traffic

Figure 5: Delay vs. Packet Number for the RED and the proposed algorithm.

4.5. Discussion

An important question arises now, why the service ratio 1:1:1 improves performance although the behavior of the router with this ratio seems like using one queue with scheduling strategy FIFO?

The service ratio 1:1:1 improves performance because using one queue with FIFO scheduling strategy does not guarantee fairness between different traffics. Since traditional data such as FTP arrives at the internet router in burst, then many packets of the same type will be queued in the buffer. Thus, during service, one type of traffic will be served for long period of time based on the size of the burst. But, queuing packets into three different queues and using the scheduling strategy 1:1:1 give the chance of service to the different types of traffics, resulting in fairness between different traffics.

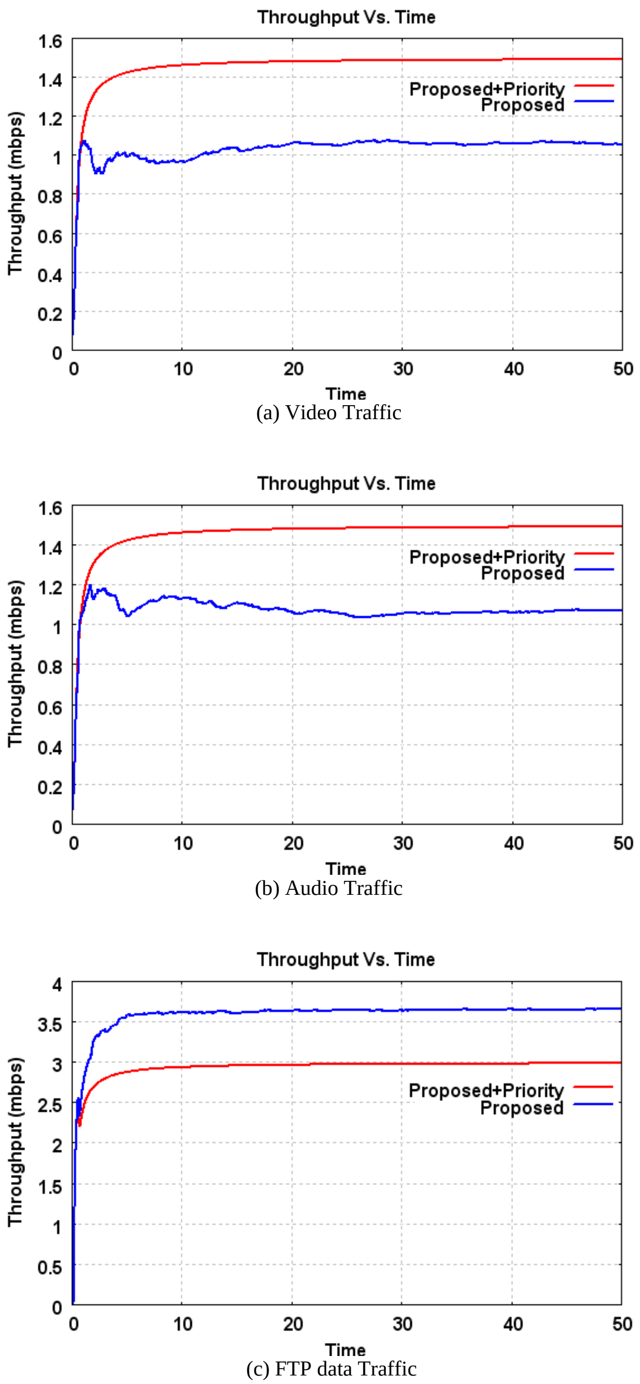
Additional improvement to the real time traffics may be achieved by serving different packets of different traffics by different weights. The problem is how to specify this weight. An approach is to predefine the weight statically with different priorities. For example, the service ratio may be 3:2:1. Another method is to define the weight dynamically based on the average queue size. That is, starting with the service ratio 1:1:1 and during each cycle test the average size of the queues and then calculate the new service ratio for the next cycle based on the current average queue size.

4.6 Effect of Static Priority Based Scheduling

In this section, we study the effect of assigning different priorities to different types of packets. The purpose is to compare the static priority based scheme with non priority based scheme and to study the effect of giving priority to interactive traffics over the ftp traffic. The service ratio is considered as 3:2:1, i.e., for every three packets of video, two audio packets and one data packet will be served.

Figure 6 shows the throughput for different traffics using the proposed algorithm with priority (*denoted by Proposed+Priority*) and without priority (*denoted by Proposed*). The figure shows that the proposed algorithm with priority provides higher throughput than the proposed algorithm without priority for the multimedia flows but the throughput decreases for FTP flow.

Figure 7 shows the delay of different traffics using the proposed algorithm with priority (denoted by *Proposed+Priority*) and without priority (denoted by *Proposed*). The figure shows a significant reduction in delay for both Video and Audio traffics when using the proposed algorithm with priority.



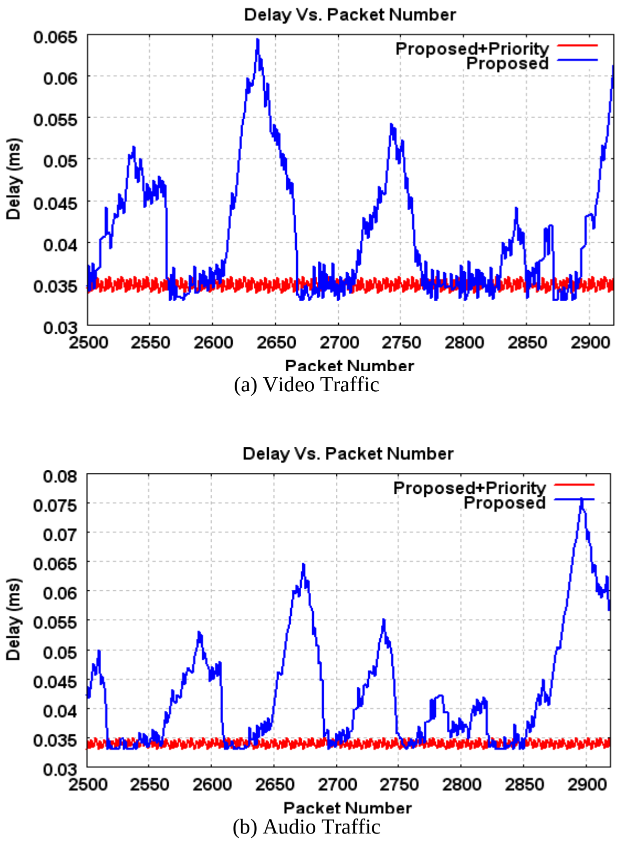
(a) Video Traffic

(b) Audio Traffic

(c) FTP data Traffic

Figure 6: Throughput vs. time for the proposed algorithm with/without priority.

Figure 8 shows the packet drop rate when using the proposed algorithm with priority (denoted by *Proposed+Priority*) and without priority (denoted by *Proposed*). The figure shows that the proposed algorithm with priority decreases the drop rate comparing to the proposed algorithm without priority.



(a) Video Traffic

(b) Audio Traffic

Figure 7: Delay vs. Packet Number for the proposed algorithm with/without priority.

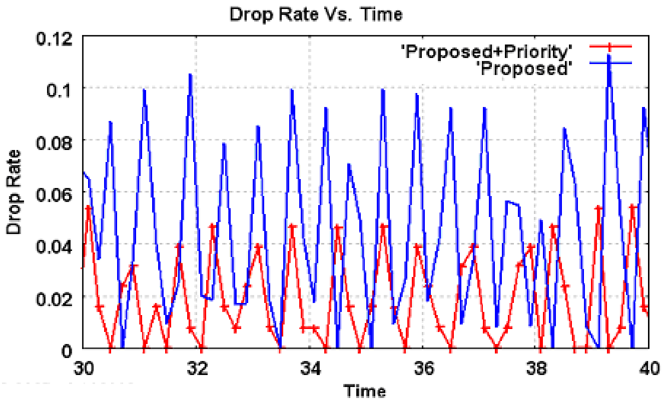


Figure 8: Packet drop rate vs. time for the proposed algorithm with/without priority.

5. Conclusions and Future Work

In this paper, a queuing discipline is proposed to the active queue management to deal with both the best-effort flows and multimedia flows. The proposed algorithm uses three priority queues each of which handles a single class of traffic. Additionally, the queued packets are scheduled according to predefined weights. The proposed approach prevents starvation of lower priority traffic (i.e., best effort traffic) while satisfying the QoS requirements for higher priority traffics. This would result in a significant decrease in the delay and loss of the real time packets. Our future work will be concerned on the dynamic weight adjustment of packet scheduling to achieve more improvement to the real time traffics.

References

[1] V. Jacobson, and M. J. Karels, ”Congestion Avoidance and Control,” Proceedings of ACM SIGCOMM, Vol.18, No. 4, pp. 314-329, August 1988.

[2] S. Floyd, and K. Fall, ”Promoting the use of End-to-End Congestion Control in the Internet,” IEEE/ACM Transactions on Networking, Vol.7, (4), pp. 458-472, August 1999.

[3] Hanaa A. Torkey, Gamal M. Attiya, and I. Z. Morsi, "Performance Evaluation of End-to-End Congestion Control Protocols," Minufiya Journal of Electronic Engineering Research (MJEER), Vol. 18, No. 2, July 2008.

[4] Sally Floyd and Van Jacobson, "Random Early Detection Gateways for Congestion Avoidance," IEEE/ACM Transactions on Networking, Vol.1, pp. 397-413, August 1993.

[5] V. Firoiu and M. Borden, "A Study of Active Queue Management for Congestion Control," Proceedings of IEEE INFOCM 2000, Vol. 3, pp. 1435-1444, March 2000.

[6] D. Lin and R. Morris, "Dynamics of Random Early Detection," Proceedings of ACM SIGCOMM Conference, Vol. 27, No. 4, pp. 127-137, 1997.

[7] T. J. Ott, T. V. Lakshman, and L. Wong, “SRED: Stabilized RED,” Proceedings of IEEE INFOCOM ’99, pp. 1346–1355, Mar. 1999.

[8] S. Floyd, “Recommendations on using the gentle variant of RED,” May 2000, available at http://www.aciri.org/floyd/red/gentle.html.

[9] B. Zheng, and Mohammed Atiquzzaman, ”DSRED: An Active Queue Management Scheme for Next Generation Networks” Proceedings of 25th IEEE conference on Local Computer Networks (LCN) 2000, November 2000

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

[11] J. Aweya, M. Ouellette, and D. Y. Montuno, "A control theoretic approach to active queue management," Computer Networks, Vol. 36, pp. 203–235, 2001.

[12] Sally Floyd, Ramakrishna Gummadi, and Scott Shenker, "Adaptive RED: An Algorithm for Increasing the Robustness of RED’s Active Queue Management," Under submission, http://www.icir.org/floyd/papers/adaptiveRed.pdf, 2001.

[13] J. Koo., Byunghun Song., Kwangsue Chung., Hyukjoon Lee., and Hyunkook Kahng, ”MRED: A New Approach to Random Early Detection” 15th International Conference on Information Networking, February 2001

[14] S. Athuraliya, V. H. Li, S. H. Low, and Q. Yin, “REM: Active Queue Management,” IEEE Network Magazine, Vol. 15, pp. 48–53, 2001.

[15. W. Feng, Apu Kapadia, and Sunil Thulasidasan,, “GREEN: Proactive Queue Management over a Best-Effort Network”, IEEE GlobeCom (GLOBECOM 2002), Taipei, Taiwan, November 2002.

[16] W. Feng, D.D. Kandlur, D. Saha, and D. Saha, “The Blue active queue management algorithms,” IEEE/ACM Transactions on Networking, 2002.

[17] J. Sun, K. Ko, G. Chen, S. Chan, and M. Zukerman, "PD-RED: To Improve the Performance of RED", IEEE Communication Letters, Vo. 7, No. 8, pp. 406-408, August 2003.

[18] C. Wag, Bin Liu, Y.Thomas Hou., and Kazem Sobraby., “LRED: A Robust Active Queue Management Scheme Based on Packet Loss Ratio”, 23rd Annual Joint Conference on Performance, Computing and Communication 2004.

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

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

[21] Shan Suthaharan, “Reduction of queue oscillation in the next generation Internet routers”, Science Direct, Computer Communication, 2007.

[22] K. Chitra and G. Padamavathi, "Adaptive CHOKe: An Algorithm to Increase the fairness in Internet Routers," International Journal Advanced Networking and Applications, Vol. 01, Issue 06, pp. 282-386, 2010.

[23] K.Chitra and G.Padamavathi, "Classification and Performance of AQM-Based Schemes for Congestion Avoidance," International Journal of Computer Science and Information Security, Vol. 8, No. 1, 2010

[24] V. A. Reguera, F. F. Paliza, W. Godoy, and E. M. Fernandez, "On the impact of Active Queue management on VOIP quality of Service," Computer Communications, Vo. 31, pp. 73-87, 2008.

[25] I. Awan, S. Ahmed, and B. Ahmed, "Performance Analysis of Multimedia Based Web Traffic with QoS Constraints," Journal of Computer and System Sciences, Vol. 74, pp. 232-242, 2008.

[26] N. Selvam, and S. Radhakrishnan, "Processor Based Active Queue Management for Providing QoS in Multimedia Application," International Journal of Computer Science and Information Security, Vol. 7, No. 9, 2010.

[27] S. McCanne and S. Floyd, "ns Network Simulator", http://www.isi.edu/nsnam/ns.

[28] L. Breslau, et al., "Advanced in network simulation," IEEE Computer, Vol. 33, No. 5, pp. 59-67, May 2000.

[29] K. Fall and K. Varadhan, “The ns Manual,” UC Berkeley, LBL, USC/ISI, and Xerox PARC, June 2003: http://www.isi.edu/nsnam/ns/doc/

Gamal Attiya graduated in 1993 and obtained his MSc degree in computer science and engineering from the Menufiya University, Egypt, in 1999. He received PhD degree in computer engineering from the University of Marne-La-Vallée, Paris-France, in 2004. He is currently Lecturer at the department of Computer Science and Engineering, Faculty of Electronic Engineering, Minoufiya University, Egypt. His main research interests include distributed computing, task allocation and scheduling, computer networks and protocols, congestion control, QoS, and multimedia networking.

Heba A. El-Khobby graduated in 1998 and received her M.Sc and PhD degree from the Tanta University, Egypt, in 2003 and 2009 respectively. She is currently a Lecturer at the Department of Electronics and Electrical Communications Engineering, Tanta University, Egypt. Her research interests are in the field of computer networks including congestion control, routing, mechanisms for resource management and QoS.