A priority queuing policy for better performance of multimedia networking

Heba El-Khobby [a], Gamal Attiya [b], Salah Khames [a] and Mostafa Nofal [b]
[a] *Electrical Engg. Dept., Faculty of Engg., Tanta University, Tanta Egypt*
[b] *Faculty of Electronic Engg., Minufiya University, Minufiya, Egypt*

Active Queue Management (AQM) in routers has been a very active research area in the Internet community to support congestion control. The most well-known AQM is Random Early Detection (RED). This approach, while necessary and powerful with the Transmission Control Protocol (TCP) flows, is not sufficient to provide good service in all circumstances specially with non-TCP flows such as multimedia traffic. In this paper, we propose a queuing discipline to the RED AQM so as to improve the performance of multimedia applications on the Internet. In the proposed strategy, multiple queues are used in the internet router and the arriving packets are queued according to their class type. Additionally, the queued packets are scheduled according to different priority classes; low and high priorities.

يهتم البحث بإدارة منهجية انتظار حزم المعلومات في مخزن محدد المسار (router) و عملية جدولة لتلك الحزم في طابور شبكة المعلومات. يعتبر منهج الكشف العشوائي المبكر (RED) من اشهر تلك المنهجيات المستخدمة لتجنب حدوث الاختناق في شبكة المعلومات وقد اثبت استخدامه كفاءة عالية بالنسبة ل "TCP Flows" إلا انه لم يحقق الخدمة الجيدة بالنسبة لبقية الأنواع الأخرى. وهنا نقترح نموذج معتمد على نظام (RED AQM) وذلك لتحسين أداء الشبكة بالنسبة لحزم الوسائط المتعددة ، وتم التركيز على بعض المعاملات مثل كمية الفقد في الحزم المرسلة و مقدار التأخير في وصول تلك الحزم عند مخزن محدد المسار وكذلك "Delay"، "Throughput"، "Jitter" و غيرها. وفى هذه الدراسة تم عمل تصنيف للحزم الواصلة إلى مخزن محدد المسار حسب نوع الحزم، حيث تم وضع حزم الوسائط المتعددة في طابور منفصل عن باقي الحزم المعلوماتية وبعد ذلك تم عمل جدولة لتلك الحزم مع إعطاء أولوية لها على حسب تصنيفها. وقد أثبتت النتائج التي حصلنا عليها أن النظام المقترح، يحسن من أداء الشبكة مقارنة بالنظام الشائع استخدامه.

Keywords: Queue Management, Packet Scheduling, Multimedia, QoS

1. Introduction

Active Queue Management (AQM) in routers has been a very active research area in the internet community to support the end-to-end congestion control [1-5]. Clearly, Internet routers must buffer incoming packets when the outgoing or forwarding interface link is busy. The size and management of the router buffer(s) are therefore crucial: too small a buffer, too many packets must be dropped in times of high congestion; too large a buffer, the delay experienced by packets may become excessive. Therefore, internet routers should implement some mechanisms to manage queue, reduce end-to-end latency and reduce packet dropping within the Internet.

The AQM comprises (i) queue management and (ii) packet scheduling. It is useful to distinguish between these two classes of router algorithms. The purpose of queue management is to manage the length of queue in the router by dropping packets from the queue when necessary or appropriate so as to avoid congestion, while the purpose of packet scheduling is to determine which packet to send next and is used primarily to manage the allocation of bandwidth among packets. Packet scheduling and queue management should be seen as complementary, not as replacements for each other. While these two router mechanisms are closely related, they address rather different performance issues. The default mechanism for managing queue lengths is Random Early Detection (RED). Indeed, routers still implement First-In First-Out (FIFO) (also called, First-Come First-Serve (FCFS)) scheduling with RED buffer management.

Active queue management requires that end-hosts recognize dropped packets and respond by reducing their rate of packet transmission. In the Internet, TCP recognizes packet loss as an indicator of network congestion, and reduces transmission rate [6]. To date, active queue management appears promising since the predominant transport protocol on the Internet is TCP. Unfortunately, most non-TCP flows (often termed misbehaved flows) use UDP and get an unfair share of network bandwidth when there is congestion [7]. 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 [8]. However, multimedia applications have different performance constraints than those of traditional applications. Traditional applications are very sensitive to lost packets; hence use TCP to guarantee that lost packets are retransmitted. 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 playout. However, in the presence of jitter, inter-arrival times will vary. In this case, 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. Detecting and retransmitting lost packets causes considerable jitter, making TCP unattractive to multimedia applications.

In this paper, we propose a queuing discipline to the RED AQM so as to improve the performance of multimedia applications on the Internet while ensuring fairness to TCP applications. In the proposed strategy, multiple queues are used in the internet router and the arriving packets are queued according to their class type. Additionally, the queued packets are scheduled according to different priority classes; low and high priorities. The proposed strategy is investigated and compared with REQ AQM by using the network simulator NS-2. 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 [9-11]. 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.

The rest of this paper is organized as follows: Section 2 briefly describes the RED AQM algorithm. Section 3 presents the FIFO packet scheduling algorithms. Section 4 introduces a priority queuing strategy is proposed to be implemented with the RED AQM algorithm so as to improve the performance of multimedia networking. In Section 5, simulation results are shown by examining the behavior of RED AQM when considering our approach in comparing with the behavior of RED AQM only towards multimedia traffic. Finally, section 6 summarizes our conclusions and lists some possible future work.

2. Random Early Detection (RED)

The default mechanism for managing queue/buffer lengths in the internet router is Random Early Detection (RED) [12]. It is a method for discarding packets to prevent and avoid network congestion. The RED uses a weighted-average queue size and thresholds to detect impending congestion, and randomly drops incoming packets as the average queue size exceeds a minimum threshold. The basic idea of RED is to start packet discarding before the queues are full; not as in other packet discard algorithms; packets are dropped when congestion is already in act. RED uses statistical methods to drop packets before the router queue overflows. The RED algorithm calculates the average queue size with an exponential weighted moving average. The average queue size is compared to two thresholds: a minimum and a maximum threshold. When the average queue size is less than the minimum threshold, no packets are dropped. When the average queue size is greater than the maximum threshold, every arriving packet is dropped. When the average queue size is between the minimum and maximum thresholds, the arriving packet is dropped with certain probability varies linearly

from 0 to max_p. The dropping probability P is a function of the average queue length avg, as shown in fig. 1. P is zero while the average queue length is less then the minimum threshold and it grows up to a max_p value as the average queue length increases.

3. Packet scheduling

The First-In First-Out (FIFO) is the scheduling discipline that is currently widely used in the Internet routers. The FIFO discipline is extremely simple when compared with other queue disciplines. The first packet come in the queue is the first packet that is served; hence this packet scheduling is also called First-Come First-Served (FCFS). In FIFO packet scheduling, an arriving packet to a queue joins at the tail of the queue and has to wait for all packets it sees in the queue upon its arrival to be serviced before it can leave the queue. In FIFO, all packets are treated in the same manner as they are placed in a single queue and are served in the same order they were placed. When the queue becomes full, congestion occurs and the new incoming packets are dropped. Thus, the FIFO don not discriminate arriving packets to a full buffer according to the sizes of their flows. Therefore, short-lived flows (interactive audio-video) can experience packet losses under FIFO which significantly affects their transfer times. Thus, FIFO scheduling penalizes short-lived flows. As a consequence, the overall performance under FIFO is reduced since short flows make up a very large fraction of all flows in the Internet today. On the other hand, FIFO has some of drawbacks. One of its drawbacks is that there is no way to handle different traffic classes or priorities. The goal of this paper is to develop/refine alternative packet scheduling scheme in order to improve the overall performance of short flows without penalizing the performance of long flows too much.

4. Priority queuing policy/strategy

Our approach consists of two separate parts. The first part concerns with how the arriving packets are organized and queued within the internet router. The second part deals with how the queued packets are scheduled or served.

In priority queuing strategy, multiple queues (two queues) are used in the internet router and the arriving packets are queued according to their class type: multimedia class (UDP flows) and text-based class (TCP flows). Different types of packets are classified via a certain field in the packet header. Initially, the router has to distinguish packets belonging to different class of services. By inspecting the packet header, the packet is classified and consequently is handled accordingly. Additionally, the queued packets are scheduled according to different priority classes; low and high priorities. This implies that, all packets belonging to the same flow or the same class of service are ruled in a predefined manner and are processed in a corresponding way by the router. For example,

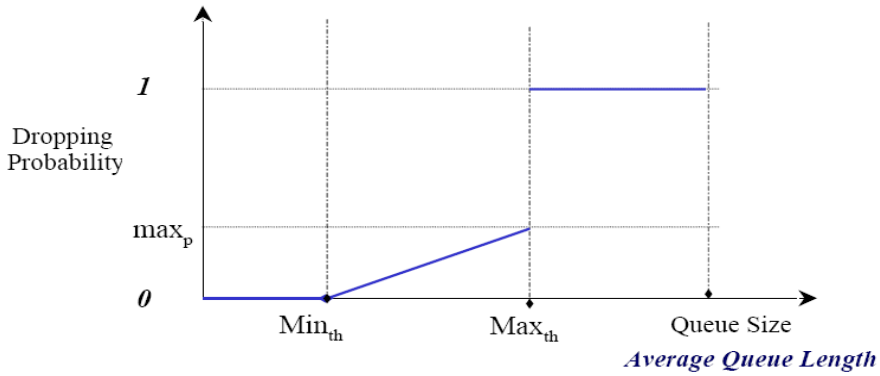

Fig. 1. Dropping probability vs. Average queue length.

all packets belonging to a certain application or related to a specific protocol may be defined to form a traffic flow. The main advantage of this strategy is that an overflow in one flow has no effect of other ones. Indeed, a critical time applications can be served with higher priority so as to overcome the end-to-end latency.

It is important to note that, the priority queuing policy should be coded with the RED AQM algorithm so as to improve the performance of multimedia applications on the Internet while ensuring fairness to TCP applications. Note also that, in the internet, all packets are treated in the same manner without discrimination between them. Here, packet classification indicates the process of categorizing packets into flows in a router. All packets belonging to the same flow or the same class of service are ruled in a predefined manner and are processed in a corresponding way by the router. Classification of packets is needed for those services that require the distinction between different kinds of traffic [13]. As the allocation of network resources to different users is an important concept, these resources have to be shared among many different traffic flows in an efficient way. This implies packets to schedule them into multiple queues to allocate the bandwidth to each class of traffic.

5. Simulation results

As a simulation environment, we have chosen the network simulator Ns-2 [10]. It is a discrete event simulator which provides substantial support for simulation of TCP, routing and multicast over wired and wireless networks. NS-2 is a commonly used tool for network simulation and modeling.

Fig. 2 shows the network topology and the application flows that used for the simulation study. It represents a simple network bottleneck configuration. Each link that connected a source has link capacity of 10 Mb/s and delay of 5 ms, the other links connected a destination node have 10 Mb/s bandwidth and 3 ms delay. The bottleneck link has 6 Mb/s bandwidth and 20 ms delay. The traffic sources are: 6 FTP, 3 MM_APP and 2 CBR, where FTP uses TCP Reno and the other uses UDP 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 1000 bytes. The UDP agents also have a packet size of 1000 bytes and a rate of 1Mbps. Network router used RED queue management, were assigned a 20 packets long queue. The parameter settings were, (minimum threshold = 5 packets, maximum threshold = 15 packets, average queue weight = 0.002 and max_p =0.1). We take simulation time of 50 sec.

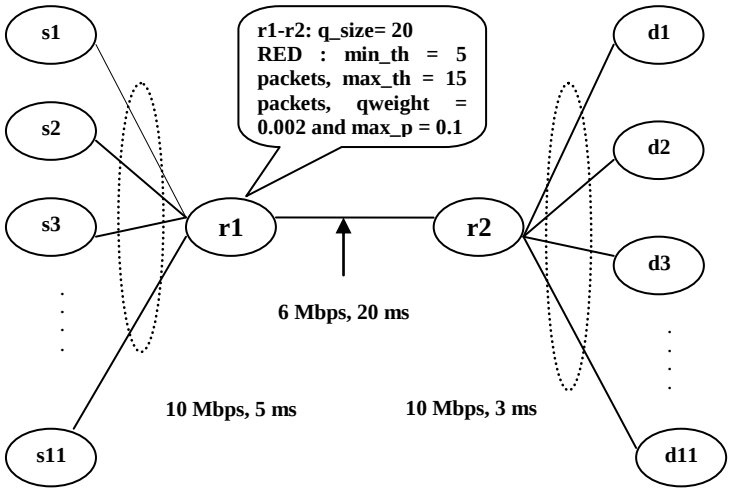

Fig. 2. Network TOPOLOGY and Flows.

One of the main objectives of RED algorithm is to accommodate short burst that may be sensitive in delay, but not to allow average queue size to increase. As shown in fig. 3, that the proposed algorithms; algorithm 2 (the RED algorithm with packet classification) and algorithm 3 (the RED algorithm with packet classification and priority) keep the average queue size is much lower than that of the RED algorithm (algorithm 1). Decreasing the average queue size means decreasing the instantaneous one and result in a decreasing of the arriving packet dropping/marking probability. As the average queue size is initially zero, then increased for each arriving packet according to the relation $(1-\omega_q)*avg + \omega_q*q$. Where, q is the actual queue size. If the queue size becomes empty; the number of packets that could have sent during the idle time is estimated.

Controlling the average queue size means controlling the average queuing delay; as shown in fig. 4, the average delay for using the proposed algorithms is much less than that of the RED algorithm (algorithm 1). In addition, the algorithm 3 provides a better significant performance than algorithm 2. However, this improvement at the expense of increasing the end-to-end delay. The end-to-end delay for the RED algorithm has a range from 0.028 sec to 0.048 sec and the delay for the algorithm 2, algorithm 3 is increased from 0.03 sec to 0.09 sec (as shown in fig. 5).

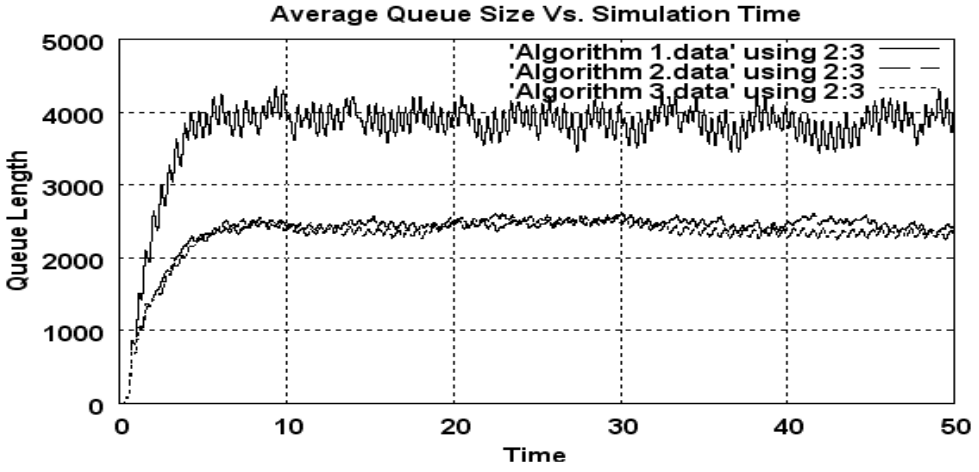

Fig. 3. Average queue size vs. time.

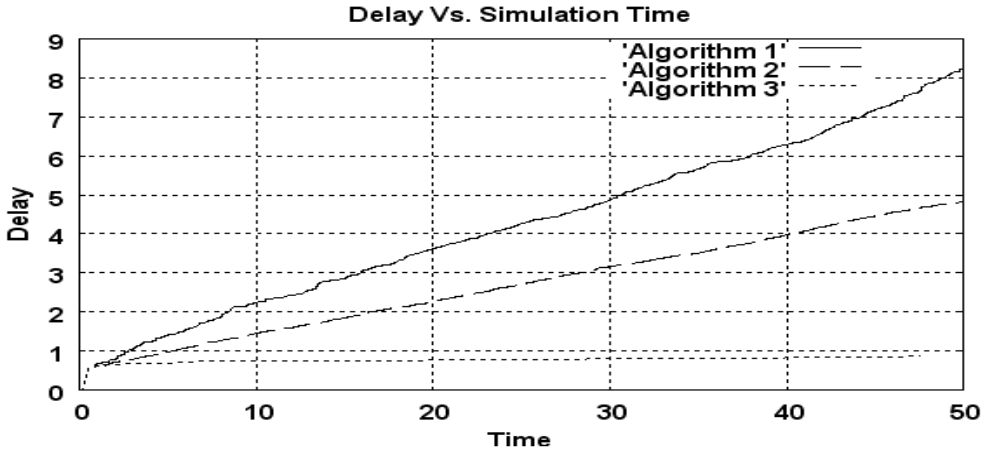

Fig. 4. Average queue delay vs. time.

The per flow quantities such as packet losses, number of received and dropped packets are shown in the figs. 6 – 8. It is observed that the proposed algorithms introduce a performance improvement in the traffic as the arrived packets increased; the packet drops and packet loss are decreased. It is observed that algorithm 3 provide a significant decrease in packet loss.

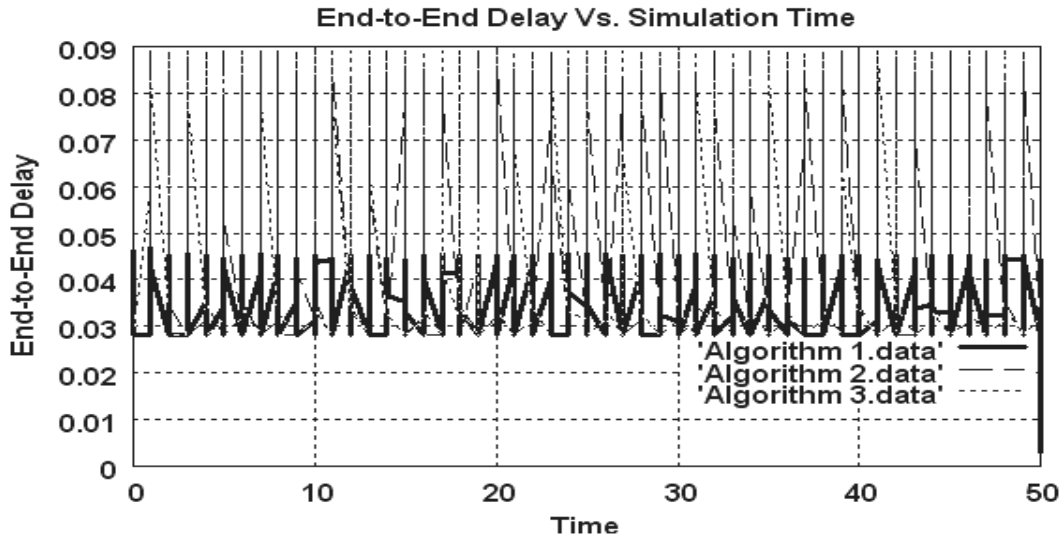

Fig. 5. Average end to end delay.

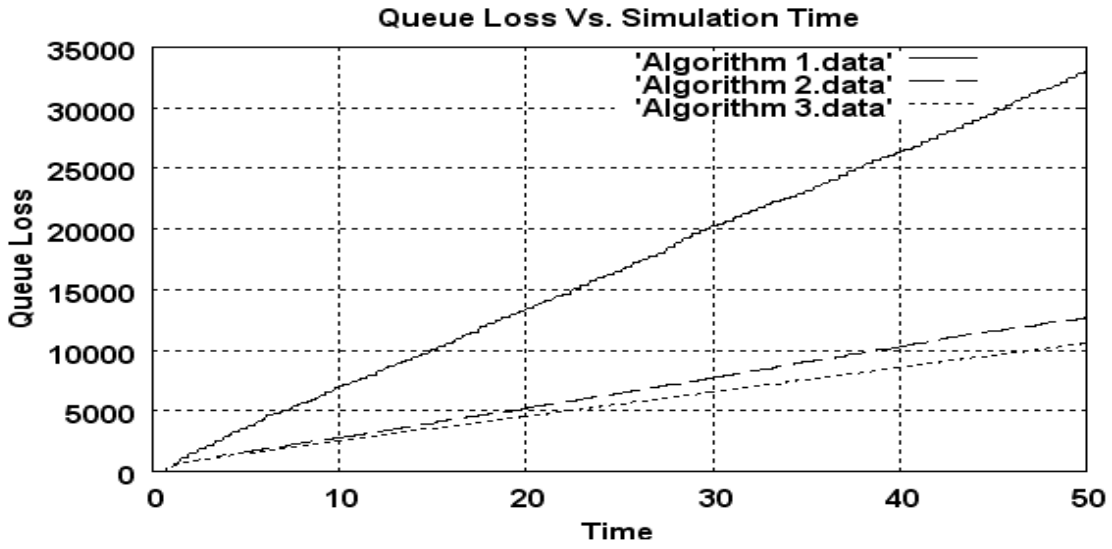

Fig. 6. Loss in packets vs. time.

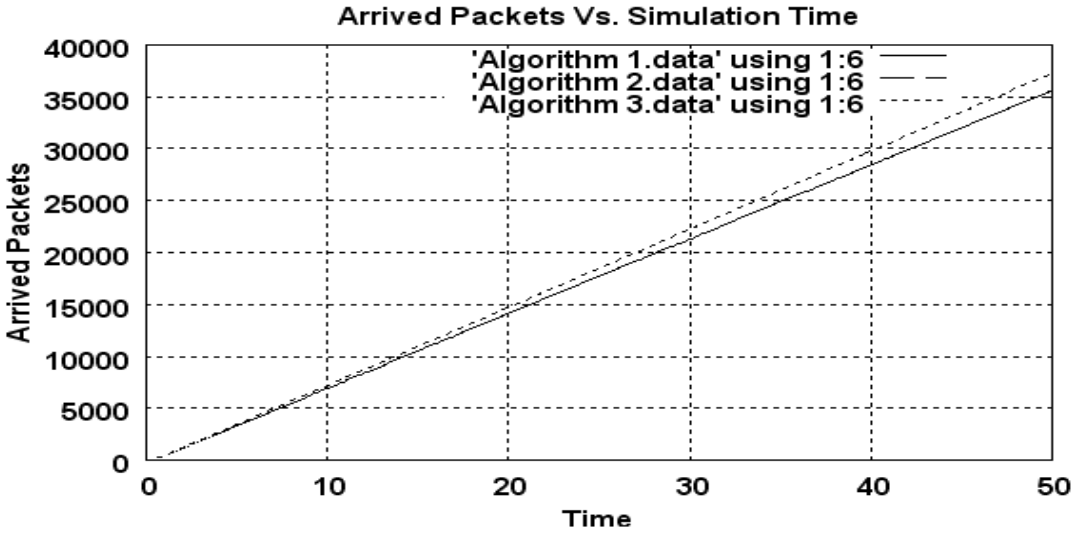

Fig. 7. Total number of arrived packets vs. time.

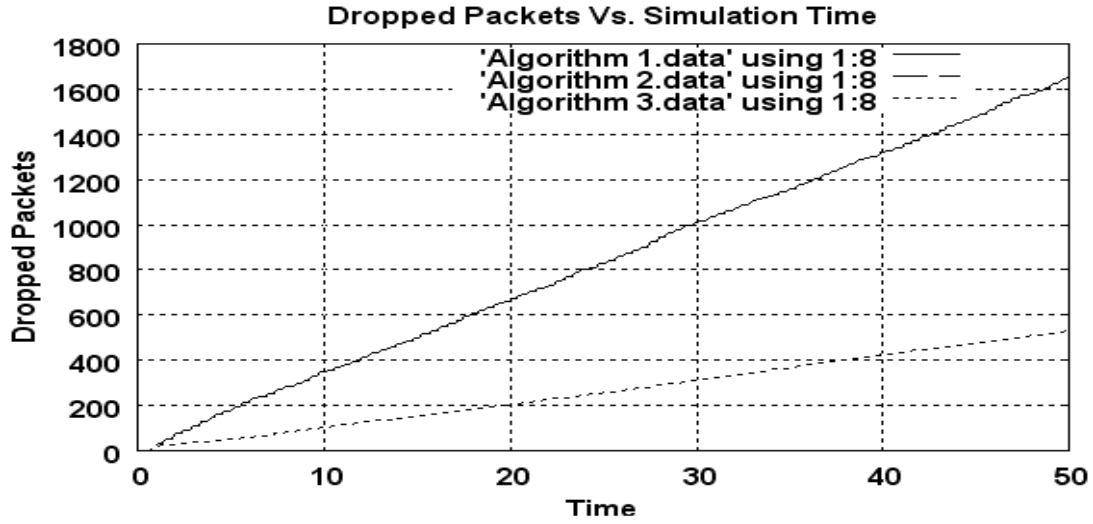

Fig. 8. Total number of dropped packets vs. time.

It is shown that the results for the algorithm 2 and 3 are identical for the received and dropped packets. As the RED AQM algorithm is designed to avoid synchronization, since different links may be received a congestion signal at the same time leading to oscillations in the throughputs [10]. These oscillations result in high jitter and lower the average throughput. The proposed algorithms increase the total throughput at the expense of decreasing the average throughput as a result of increasing the jitter as shown in fig. 9 and 10.

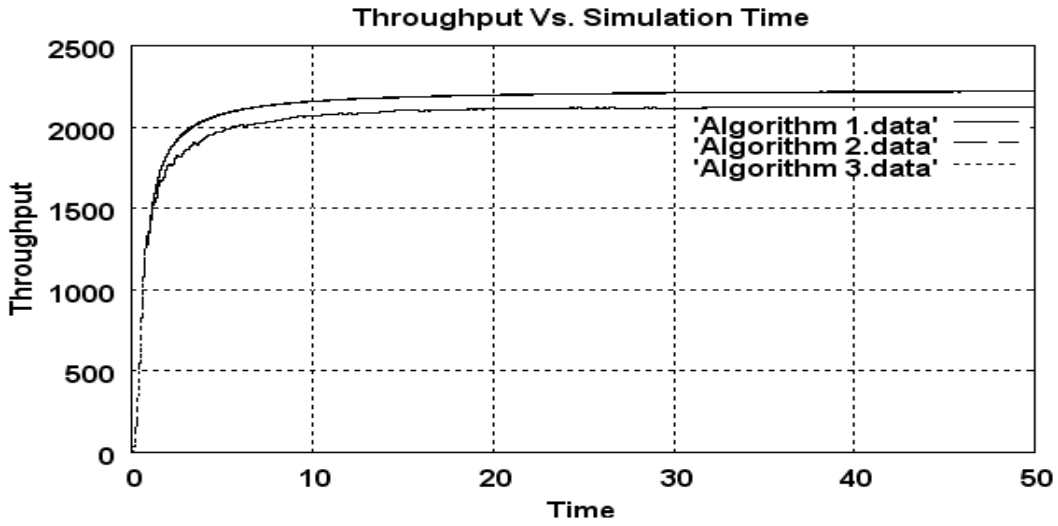

Fig. 9. Throughput vs. time.

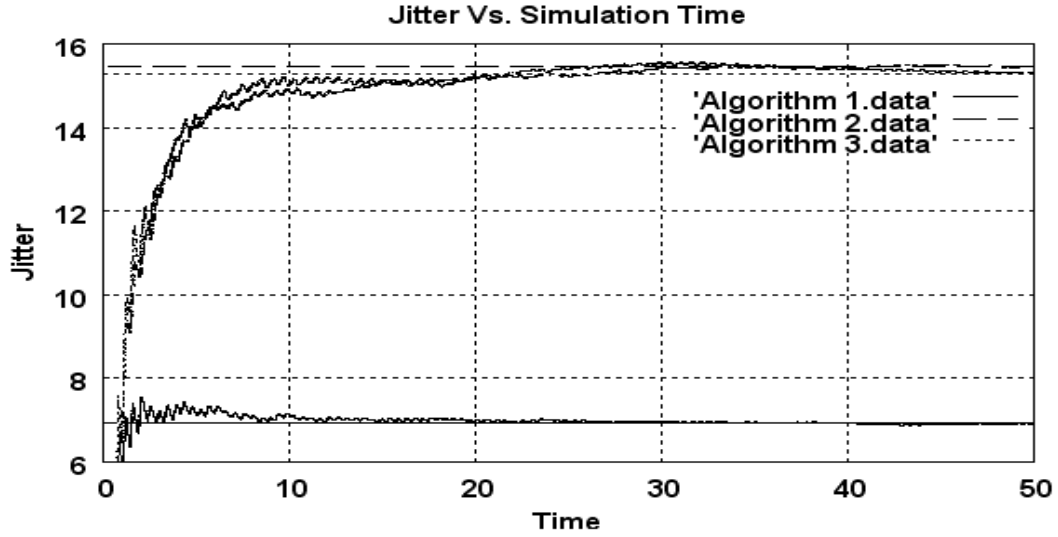

Fig. 10. Jitter vs. time.

6. Conclusions

In this paper, a priority queuing strategy to the RED AQM is developed to deal with the multimedia flows. The proposed algorithm improves the performance of multimedia on the network. The main objective is to reduce the packet loss, minimize the average packet delay and increasing the throughput. In the proposed strategy, multiple queues are used in the router and the arriving packets are queued according to its type. Additionally, the arriving packets are differentiated into different priority classes. The proposed strategy is coded with the RED AQM and investigated in comparing with the RED AQM algorithm using Ns-2. The results showed that using the priority queuing strategy with the RED AQM decreases the average queue size, the packet loss and the average queuing delay.

References

[1] Hanaa A. Torkey, Gamal M. Attiya and I.Z. Morsi, "Performance Evaluation of End-to-End Congestion Control Protocols",

Menoufia Journal of Electronic Engineering Research (MJEER), Vol. 18 (2) (2008).

[2] J. Aweya, Ouellette and D.Y. Montuno, "A Control Theoretic Approach to Active Queue Management", Computer Networks, Vol. 36, issue 2-3, pp. 203-35 (2001).

[3] C.V. Hollot, V. Misra, D. Towsley and W. Gong, "A Control Theoretic Analysis of RED", Proceedings of the 2001 IEEE Infocom, Vol. 3 (2001).

[4] C.V. Hollot, V. Misra, D. Towsley and W. Gong, "On Designing Improved Controllers for AQM Routers Supporting TCP Flows", Proceedings of the 2001 IEEE Infocom, Vol. 3 (2001).

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

[6] Bitorika, M. Robin and M. Huggard, "An Evaluation Framework for Active Queue Management Schemes", Proc. MASCOTS '03, IEEE (2003).

[7] J. Chung and M. Claypool, "Better-Behaved, Better-Performing Multimedia Networking", SCS Euromedia, May pp. 8-10 (2000).

[8] M. Parris, K. Jeffay and F.D. Smith, "Lighweight Active Router Queue Management for Multimedia Networking", Multimedia Computing and Networking (San Jose, CA), SPIE proceedings Series, Vol. 3020 (1999).

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

[10] 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/

[11] L. Breslau, et al., "Advanced in Network Simulation", IEEE Computer, Vol. 33 (5), pp. 59-67 (2000).

[12] K.K. Ramakrishnan and S. Floyd, "A Proposal to Add Explicit Congestion Notification (ECN) to IP", RFC 2481 (1999).

[13] K.K. Leung and J.K. Muppala, "Packet Marking Strategies for Explicit Congestion Notification (ECN)", Proceedings of 20th IEEE Performance, Computing and Communication Conference (IPCCC 2001), Pheonix, AZ, USA, pp. 17-23 (2001).

Received February 9, 2009
Accepted February 24, 2009