A Queue Policy for Multimedia Communications

Seong-Ho Jeong
Department of Information and Communications Engineering,
Hankuk University of Foreign Studies, Korea
E-mail: shjeong@hufs.ac.kr

Abstract: To support UDP-based real-time multimedia applications over the Internet, it is necessary to provide a certain amount of bandwidth within the network so that the performance of the applications will not be seriously affected during periods of congestion. Since the flow rates of some of these applications do not back off during periods of congestion, it is also necessary to protect flow-controlled TCP flows from unresponsive or aggressive UDP flows. To achieve these goals, we propose a simple queue policy to support multimedia applications, called threshold-based queue management (TBQM). TBQM isolates UDP flows efficiently from TCP flows to protect TCP flows while supporting bandwidth requirements of UDP applications that require QoS. In addition, TBQM supports drop fairness between TCP flows without maintaining per-flow state. We also present some experimental results to show that the proposed queue policy can work well.

1. Introduction

With the emergence of real-time multimedia applications such as packet voice and packet video, UDP traffic has been increasing over the past few years [9]. In order to support these real-time applications on the Internet, it is necessary to provide a certain amount of bandwidth to the applications within the network so that the performance of the real-time applications will not be seriously affected within the network even during periods of congestion.

The flows of these real-time applications do not typically back off when they encounter congestion, thus they are called unresponsive or aggressive flows. As a result, they aggressively use up more bandwidth than other TCP "friendly" flows that utilize congestion control. This mix of congestion-controlled and congestion-uncontrolled traffic could cause an Internet Meltdown [1]. Therefore, while it is important to have router algorithms support UDP flows that require QoS by assigning appropriate bandwidth, it is also necessary to protect responsive TCP flows from unresponsive or aggressive UDP flows and to provide reasonable QoS to all users.

Basically, there are two types of router-based algorithms for achieving a certain QoS: scheduling algorithms and queue management algorithms. Scheduling algorithms can provide sophisticated bandwidth control, but they are often too complex for high-speed implementations and do not scale well to a large number of users. On the other hand, queue management algorithms have had a simple design from the beginning. One of the key examples of queue management algorithms is Random Early Detection (RED) [2]. A router implementing RED maintains a single FIFO that is shared by all the flows, and drops an arriving packet randomly during periods of congestion. The drop probability increases as the level of congestion increases. Since RED behaves in anticipation of congestion, it does not suffer from the lock-out and full-queue problems [2] inherent in the widely deployed Drop Tail (FIFO) mechanism. However, like Drop Tail, RED is not able to penalize unresponsive flows. The resulting percentage of packets dropped from each flow over a period of time is almost the same. As a result, misbehaving flows can continue to use up a large fraction of the link bandwidth and have a serious impact on responsive TCP flows.

To better distinguish unresponsive flows, a few variants of RED such as RED with penalty box [3] and Flow Random Early Drop (FRED) [5] have been proposed. However, these algorithms incur extra implementation overhead since they need to collect certain types of state information. RED with penalty box keeps information about unfriendly flows while FRED needs information about active flows. Furthermore, FRED does not consider provisioning of specific bandwidth for real-time UDP-based applications. Recent work in [6] proposes an algorithm called Stabilized RED (SRED), which stabilizes the occupancy of the FIFO buffer, independently of the number of active flows. SRED estimates the number of active flows and finds candidates for misbehaving flows. It does this by maintaining a data structure that serves as a proxy for information about recently seen flows. Although SRED identifies misbehaving flows, it does not propose a simple router mechanism for penalizing misbehaving flows. In addition, it does not consider provisioning of specific bandwidth for real-time UDP traffic.

The objective of this paper is to present a simple queue policy algorithm that provides a certain amount of bandwidth to UDP flows that require QoS and also protects TCP friendly flows from unresponsive UDP flows. Our approach also supports drop fairness between TCP flows without maintaining per-flow state. The rest of the paper is organized as follows. Section 2 describes the key features and operations of the proposed queue policy, called threshold-based queue management (TBQM). In Section 3, we provide some simulation results showing that it is possible to meet our goals using the proposed approach. Finally, Section 4 presents a summary of the paper.

2. Queue Policy Algorithm

The maximum length of a camera-ready manuscript is 4 pages, In this section, we describe our proposed queue policy, called threshold-based queue management (TBQM) in further detail. The network model is similar to that used in the Differentiated Services architecture where a network consists of edge routers and core routers as shown in Figure 1. The edge routers perform packet classification and encode certain state in packet headers, and the core routers use the state encoded in the packet headers for queue

management. The following sections describe the key features of TBQM.

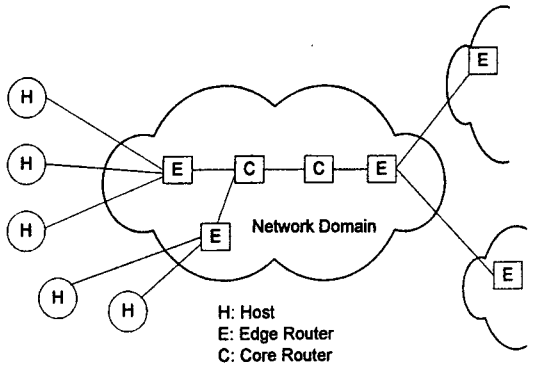


Figure 1. Network architecture

2.1 Traffic classification

Since we are focusing on UDP applications that require QoS and TCP applications, TBQM basically supports three different traffic classes: a UDP class that requires QoS (hereafter "QoS-UDP class"), better-than-best-effort-service (BBES) class for TCP traffic, and best-effort-service (BES) class for other traffic. We classify incoming traffic into QoS-UDP class, BBES class, or BES using packet state information inserted in the IP packet header. Note that carrying state in packets is explained in [8]. Initial packet classification is performed at the edge routers which may contact a policy server. Core routers simply check the packet state information to classify incoming packets. Although packets are classified, there is still only one queue of data packets in the core router, shared by all traffic classes.

2.2 QoS support for UDP flows and protection of TCP flows

In order to provide high-quality service to the QoS-UDP class, we allocate a certain amount of bandwidth to the QoS-UDP class by assigning a maximum allowable buffer amount for the QoS-UDP class in edge routers and core routers. The necessary buffer size is determined by the agreed upon traffic profile of the specific QoS-UDP traffic which can be determined based upon an admission control procedure or negotiation between a user and a service provider. Suppose that two flows share a finite buffer of size B and are multiplexed onto a link of capacity C using a FIFO scheduler. Flow 1 (e.g., QoS-UDP flow) has peak rate specified in its traffic profile while flow 2 is potentially aggressive, and could swamp the first flow if its arrival into the buffer is unregulated. We logically partition the buffer into two portions that correspond to the maximum occupancy levels allowed for flows 1 and 2, respectively, so as to ensure that flow 1 does not lose packets. In this case, flow 1's share of the buffer should be at least as large as its share of the bandwidth, i.e.,

$$B_1 / B \geq \rho_1 / C$$

where B_1 is the flow 1's share of the buffer. It is also shown in [4] that if flow i requires a guaranteed service rate ρ_i and is peak-rate conformant, a buffer occupancy threshold of $B\rho_i / C$ is sufficient to guarantee lossless service.

Based on these observations, it is possible to provide a certain amount of bandwidth to QoS-UDP traffic class using a specific buffer occupancy threshold for QoS-UDP traffic class. This also makes it possible to isolate QoS-UDP traffic from TCP traffic by managing each traffic class separately. It should be noted that all packets are still placed in the same buffer using a FIFO approach. The proposed algorithm keeps track and limits how much of the buffer is being used by each of the traffic classes, thus accomplishing the desired goals.

2.3 A simple admission control for UDP traffic

To provide appropriate service to QoS-UDP class traffic, an admission control algorithm is needed to control the admission of QoS-UDP flows based on the availability of resources within the network. We use a simple admission control scheme to decide whether a new QoS-UDP traffic flow is accepted or not. The admission control is based upon the rate information of QoS-UDP flows. The QoS-UDP flows may be sent at constant bit rate (CBR) or variable bit rate (VBR). In the case of CBR QoS-UDP flows, CBR applications simply put the "peak rate" information into the IP header of outgoing packets. In the case of VBR QoS-UDP flows, the edge router estimates the arrival rate of the flows based on the exponential averaging method as in [7]. The calculated rate information is inserted into the IP header of outgoing packets. Note that reference [7] computes flow arrival rates and inserts these rates into packet labels. We utilize this same technique here.

Core routers maintain two variables: the aggregate arrival rate and aggregate accepted rate for QoS-UDP flows. When a special packet, a rate-request packet from an edge router arrives, the core router compares the requested rate with available capacity. If the requested rate is less than or equal to the available capacity, the request will be accepted and the core router forwards the request packet downstream. Otherwise, the request will be rejected, and a reject message will be sent back to the source edge router. Upon receiving the reject message, the edge router does not accept incoming packets that belong to the rejected flow, and it also sends a reject message to the sending host. The sending host would then be made aware that the network is unable to meet its request for bandwidth.

Note that the aggregate accepted rate should be updated because some QoS-UDP flows may have terminated. To do this, the core router calculates an aggregate arrival rate for QoS-UDP flows when a packet arrives. If the aggregate arrival rate is less than the aggregate accepted rate over a certain time interval, the aggregate accepted rate is updated using the largest value of the aggregate arrival rate over the time interval. The aggregate arrival rate A and aggregate accepted rate F are updated using the exponential averaging method [7]. That is,

$$A_{new} = (1 - e^{-T/K})\frac{l}{T} + e^{-T/K} A_{old}$$

$$F_{new} = (1 - e^{-T/K})\frac{l}{T} + e^{-T/K} F_{old}$$

where l is the length of arriving packet, T is the inter-arrival time between the current packet and the previous packet, K

is a constant, A_{old} and F_{old} are the values of A and F, respectively before the updating.

2.4 Metering at edge routers

In order to minimize the impact of QoS-UDP traffic on well-behaved TCP traffic, we use profile meters for the QoS-UDP class traffic. If a QoS-UDP flow is admitted by the admission control procedure described above, we assume that the edge router maintains a traffic profile for the admitted QoS-UDP flow. A traffic profile contains an agreed upon rate between a user and a service provider. The edge router continuously monitors incoming QoS-UDP traffic to check whether or not the incoming traffic violates the traffic profile. Whenever QoS-UDP traffic exceeds the traffic profile, the exceeding traffic is discarded so that TCP traffic will not be affected by the excess traffic.

2.5 Congestion avoidance and fairness control

We use a simple congestion notification mechanism for TCP flows to avoid congestion. Edge routers insert a special packet, called choke, every N packets of each TCP flow. The core router maintains a separate special queue for keeping choke packets for TCP flows. The congestion detection function is performed using an average threshold of the logical TCP queue. The core router maintains statistics about the average queue length using the exponential weighted moving average (EWMA) method. When the average queue length exceeds the threshold, the core router randomly selects a choke packet from the choke queue and send it back to the source edge router so that the edge router can discard incoming TCP packets based on the received number of choke packets. Since each choke packet contains its own flow ID, it is easy for the edge router to decide which packets to discard. In this manner, packets are discarded only at edge routers, and TCP flows will be dropped fairly based on their rates since the drop rate will be based on the number of received choke packets. This is basically similar to the technique in [10], but our proposed approach maintains only a single data queue rather than multiple queues.

3. Simulation Results

In order to demonstrate the effectiveness of TBQM, we used simulations to compare the performance of TBQM to simple drop-tail (FIFO) queuing, RED, and FRED. Since the focus of our simulations was to show how well TBQM behaves to support UDP traffic as well as TCP traffic, we only considered the QoS-UDP class and BBES class. Note that our approach can be extended to support more classes by using multiple buffer occupancy thresholds.

The simulated network topology is shown in Figure 2, where there are four TCP sources and five UDP sources, three routers, and a sink. The TCP traffic originates from a collection of ftp sources, and the UDP traffic originates from a set of CBR sources. The edge router maintains traffic profiles for UDP sources after the admission control procedure is complete as described in section 2.1.3. Each UDP source is allowed to send 1 Mbps UDP traffic. The edge router continuously monitors incoming UDP traffic to check if the input traffic violates the traffic profile.

In our simulations (we used the network simulator NS), we considered a single congested link between two core routers, which is shared by all flows. Specifically, a core router is configured with a 10 Mbps outbound link that is the bottleneck link in the network. When there is no congestion on the link, TCP users are capable of utilizing about 62 % of the 10 Mbps link, in aggregate.

It is assumed that the minimum guaranteed rate for each UDP user is 1 Mbps. Since there are five UDP users, it is necessary to reserve 5 Mbps, in aggregate, in the network. The total buffer size of FIFO queue is 30 KB. Based on the discussion in section 2, the buffer occupancy threshold for QoS-UDP traffic is calculated as 15 KB. Accordingly, the available buffer size for other traffic is 15 KB. RED and FRED as well as a FIFO queue management scheme were simulated for comparative purposes. The values of minth, maxth, wq, maxp for RED and FRED are 15 KB, 30KB, 0.002, and 0.02, respectively.

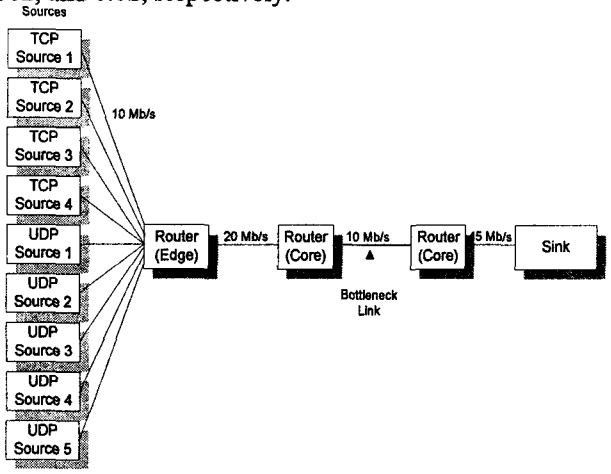


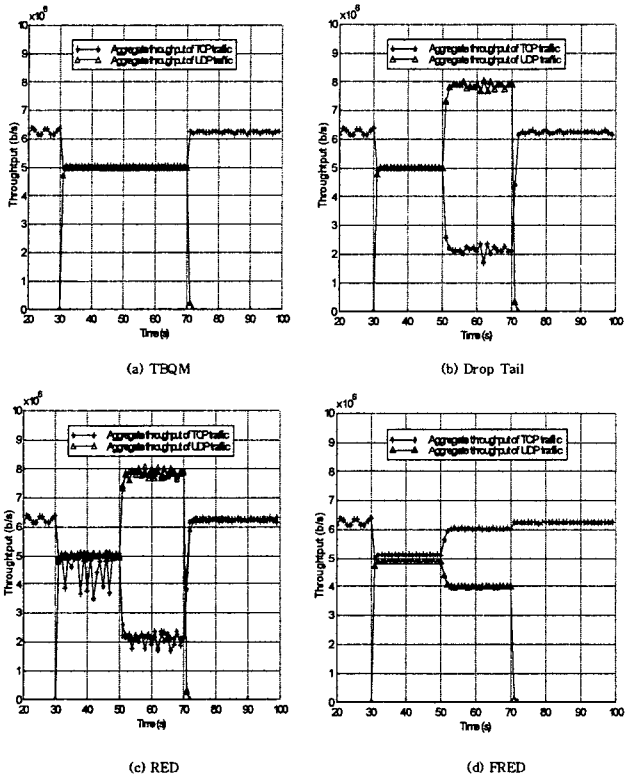
Figure 2. Network topology for simulations

Figure 3 illustrates the TCP throughput measurements on the outbound link of the congested router. In each experiment, a different queue management scheme is employed in all routers including edge and core routers. We ran infinite TCP sources during simulations. At time 30s, all UDP sources begin to generate unresponsive QoS-UDP traffic at the rate of 1 Mbps each. Thus, the total aggregate input rate of UDP sources is 5 Mbps from time 30s to 50s. At time 50s, UDP source 5 additionally generates 3 Mbps unresponsive QoS-UDP traffic until time 70s. As a result, the total aggregate input rate of UDP sources is 8 Mbps from time 50s to time 70s. All UDP sources stop generating traffic at time 70s.

As shown in Figure 3, the aggregate throughput of TCP traffic is approximately 6.2 Mbps for each of the four methods until time 30s. After five UDP sources are turned on at time 30s, the aggregate throughput of TCP traffic in all scenarios is reduced to approximately 5 Mbps. Congestion begins at time 30s since the total input rate (5+6.2=11.2 Mbps) exceeds the link capacity of 10 Mbps. In our simulations, choke packets were not used to reduce congestion so that we could show what happens when there is congestion. Note that RED shows worse TCP performance during this time interval since early discarding

of TCP packets reduces the TCP input rate, and RED does not protect TCP flows from unresponsive QoS-UDP flows.

At time 50s, congestion is more severe since the total aggregate input rate is increased to 14.2 Mbps. As shown in Figures 3-(b), (c), FIFO and RED show the worst TCP performance since unresponsive QoS-UDP traffic is not punished and consumes most of the link capacity. The results in Figure 3-(d) show that FRED protects TCP flows from QoS-UDP flows, however FRED does not guarantee the minimum rate (5 Mbps) for QoS-UDP traffic. On the other hand, as shown in Figure 3-(a), TBQM shows better performance than other schemes in terms of TCP flow protection and minimum rate guarantee for QoS-UDP flows. That is, from time 50s to 70s, TCP traffic is not affected by excess QoS-UDP traffic, and TCP gets approximately 5 Mbps continuously. Furthermore, QoS-UDP traffic gets the desired minimum rate (5 Mbps) during the 50 to 70 second simulation time period.



(a) TBQM (b) Drop Tail

(c) RED (d) FRED

Figure 3. Aggregate throughput achieved by UDP and TCP flows sharing a bottleneck link of capacity 10 Mbps in the network

4. Concluding Remarks

The volume of multimedia traffic in the Internet is dramatically increasing. The current best-effort forwarding model of the Internet is frequently insufficient for supporting multimedia traffic requirements. For example, the current Internet does not efficiently support UDP-based real-time applications such as voice over IP and video conferencing.

To satisfy the performance requirements of these ever more common applications, it is necessary to provide a certain amount of bandwidth within the network so that the performance of the applications will not be seriously affected during network congestion. Since the flow rates of some of these applications do not back off during periods of congestion, it is also necessary to protect responsive TCP flows from unresponsive UDP flows. To achieve these goals, we proposed a queue policy, called threshold-based queue management (TBQM). TBQM efficiently isolates UDP flows from TCP flows to protect TCP flows. TBQM does this by using logical buffer partitioning so as to support bandwidth requirements of UDP applications by reserving buffer space for UDP traffic.

To support our queue policy, we also proposed a simple admission control procedure for UDP traffic and a drop fairness control scheme for TCP traffic during periods of congestion. To demonstrate the effectiveness of TBQM, we compared the proposed approach with Drop-Tail (FIFO), RED, and FRED using simulations. TBQM showed better performance than other schemes in terms of TCP flow protection and minimum rate guarantee for UDP flows.

References

[1] Braden, B., Clark, D., Crowcroft, J., Davie, B., Deering, S., Estrin, D., Floyd, S., Jacobson, V., Minshall, G., Partridge, C., Peterson, L., Ramakrishnan, K., Shenker, S., Wroclawski, J., Zhang, L., "Recommendations on queue management and congestion avoidance in the internet," IETF RFC (Informational) 2309, April 1998.

[2] Floyd, S. and Jacobson, V., "Random Early Detection Gateways for Congestion Avoidance," *IEEE/ACM Transaction on Networking*, vol. 1, no. 4, pp. 397-413. August 1993.

[3] Floyd, S., and Fall, K., "Router Mechanisms to Support End-to-End Congestion Control", LBL Technical report, February 1997.

[4] Guerin, R., Kamat, S., Peris, V., and Rajan, R., "Scalable QoS Provision Through Buffer Management," *Proceedings of ACM SIGCOMM'98*, September 1998.

[5] Lin, D., and Morris, R., "Dynamics of random early detection", *Proceedings of ACM SIGCOMM'97*, p. 127-137, October 1997.

[6] Ott, T., Lakshman, T. and Wong, L., "SRED: Stabilized RED", *Proceedings of INFOCOM'99*, March 1999, p. 1346-1355.

[7] Stoica, I., Shenker, S., Zhang, H., "Core-Stateless Fair Queueing: Achieving Approximately Fair Bandwidth Allocations in High Speed Networks," *Proceedings of ACM SIGCOMM'98*, September 1998.

[8] Stoica, I. and Zhang, H., "Providing Guaranteed Services Without Per Flow Management," *Proceedings of ACM SIGCOMM'98*, September 1999

[9] Thompson, K., Miller, G., Wilder, R., "Wide-Area Internet Traffic Patterns and Characteristics," *IEEE Network*, vol. 11, no. 6, November/December 1997.

[10] Venkitaraman, N., Sivakumar, R., Kim, T., Lu, S., Bharghavan, V., "The Corelite QoS Architecture: Providing a Flexible Service Model with a Stateless Core," Working Draft of U