A Real-Time Video Multicast Architecture for Assured Forwarding Services

Ashraf Matrawy, *Member, IEEE,* and Ioannis Lambadaris

***Abstract*—This paper presents our work on developing an architecture for multicasting real-time MPEG4 over IP networks that provide service differentiation. In particular, this work is targeted at assured forwarding (AF) style services. This work is an attempt to find a simple solution to the problem of multicast congestion control of real-time traffic by exploiting the service differentiation capabilities of AF networks. Our architecture assumes loss differentiation in the network and assumes the network's ability to provide explicit congestion notification messages to the sender. We do not consider policing/shaping at the edge routers. Rather, we consider a more general case where packet marking and flow control are provided at the senders. For this network model, we built an end-to-end architecture and developed a rate-adaptation algorithm that can operate in both unicast and multicast applications with a minor modification. The simulation results show how the rate-adaptation algorithm accommodates different receivers with different networking capabilities and provides receivers with different levels of quality by taking advantage of the queue management capabilities of the AF service. We test how the architecture scales to a large number of receivers, how multiple multicast sessions interact, and how it interacts with TCP.**

***Index Terms*—Assured forwarding, differentiated services (diffserv), MPEG, multicast, multimedia, real-time, RED, video.**

I. Background

A LARGE number of group communication applications has emerged recently on the Internet. These applications have been the driving force behind the efforts of research and development of the IP Multicast technology. IP Multicast enables multipoint communications over IP networks while preserving bandwidth in a manner that can be extremely significant in cases of large groups. However, deployment of IP multicast services over the Internet has not been as rapid as needed. One of the main factors that contribute to this slow deployment is the lack of a multicast congestion control scheme that can be as robust as its unicast counterparts (e.g., TCP congestion control for reliable unicast).

The major challenges in the area of multicast congestion control are as follows.

1) The heterogeneity of receivers' networking capabilities as well as the heterogeneity of their QoS-requirements.
2) Maintaining the scalability of the multicast congestion control technique is a difficult task as the number of receivers is unknown to the sender and may grow significantly.

From the literature, multicast congestion control techniques can be classified into two categories: the sender-based (single-rate) techniques and the receiver-based (multirate, layered) techniques. All the examples presented here of both categories are pure end-to-end.

A. Receiver-Based Techniques

Receiver-based techniques are based on the ability to generate the source data in a layered format and sending the layers as different multicast groups. Each layer contains a subset of the information being sent. Receivers decide on how many layers (or equivalently, multicast groups) they can join using some bandwidth inference technique. Layers should be joined in a *cumulative* manner which means joining layers in order of their relevance. Basic layers will contain minimum information necessary to get basic quality and they should be joined first. Different approaches exist for organizing the layers and for bandwidth inference [1]–[5]. A *noncumulative* approach was proposed in [6] in which receivers can get any subset of the layers. This is based on a special encoding technique presented in [7]. Although the receiver-based techniques are a good solution to the heterogeneity problem, they have a number of other problems that are common to most of these techniques.

1) In a best-effort IP network, which drops packets uniformly at congestion time, packets from the basic layer may be lost which makes receiving higher layers useless.
2) Layered techniques assume that all layers (multicast groups) will follow the same multicast tree even when they are sent separately. This can not be guaranteed in IP networks.
3) Most of these techniques have fairness problems due to the way they react to congestion and the distribution of data across the layers [8]–[10]. This results in starving TCP of bandwidth when these protocols compete with TCP on the same link.

B. Sender-Based Techniques

In sender-based techniques, a single-rate one stream is sent to all receivers. Scalable Feedback Control [11] is one of the earliest works in this area. It uses feedback messages from receivers with information on packet loss to estimate the "group" reception status. Scalability is a potential problem with this approach because receiving feedback from all receivers simply overwhelms

Manuscript received September 17, 2002; revised November 29, 2002. This work was supported by the Natural Sciences and Engineering Research Council of Canada and the Communications and Information Technology Ontario. The associate editor coordinating the review of this paper and approving it for publication was Dr. Anna Hac.

The authors are with the Broadband Networks Laboratory, Department of Systems and Computer Engineering, Carleton University, Ottawa, ON K1S 5B6, Canada (e-mail: amatrawy@sce.carleton.ca; ioannis@sce.carleton.ca).

Digital Object Identifier 10.1109/TMM.2005.846778

the network. A proposal to use representatives of receivers groups was introduced in [12] and presented mechanisms to select representatives. However, changing representatives is major overhead for this approach. PGMCC [13] is a TCP-friendly protocol which is suitable for applications that can cope with larger variation in the sending rate. However, selection of the *acker* is very crucial to the performance of PGMCC [14]. An extension for equation-based congestion control to multicast applications was recently presented in [14] where a calculation of the round trip time is needed. From these proposals, we can identify two major problems with sender-based techniques.

1) A single slow receiver may drag down the data rate for the whole group.
2) Feedback from *all* receivers is not scalable. Solutions that are based on selecting an agent or a representative of the group presents the overhead of selecting this agent and reselecting another when network conditions change.

C. Overview of Our Work

Both the receiver-based and the sender-based techniques presented here are pure end-to-end. This motivated a number of researchers to adopt network support in their work on multicast congestion control [15]–[18]. Our goal for this work is to develop a multicast congestion control scheme that relies on the IETF proposed assured gorwarding (AF) [19] architecture. We considered AF because it helps us build a simple end-to-end architecture that avoids the problems of earlier approaches to multicast congestion control. It is also expected to be deployed soon in Internet routers as opposed to other proposed support mechanisms [15]–[18] that require major changes in current router's functionality. For the sake of simpler experiments, we do not completely implement the IETF proposed AF service. In particular, we do not consider marking/policing at the edge routers and instead marked the packets at the sender. Although our main focus is multicast, the architecture we propose here along with our rate-adaptation algorithm operates equally well in both unicast and multicast applications with a minor modification. In this paper, we report only the multicast case as it proved to be more challenging than unicast.

The rest of this paper is organized as follows. The network model and the end-to-end architecture are presented in Sections II and III respectively. We discuss the rate-adaptation algorithm in detail in Section IV. Simulation setup is explained in Section V and results are discussed in Section VI. In Section VII we discuss some of the design issues and limitations. Section VIII concludes the paper.

II. Network Model

In this section, we describe the network model we consider for our work. The choice of this model is motivated by the design goal of providing different levels of quality to receivers and guaranteeing (with a high probability) a minimum quality during congestion. We consider IP networks that support priority-dropping as a means of providing different levels of services to its users. Priority-dropping is the process of packet dropping during congestion by routers based on a priority level assigned to the packet at the sender or at an edge router. AF [19] is a good candidate to meet these requirements. AF-compliant routers achieve that by employing active queue management (AQM) techniques that recognizes packet priorities and enqueue, dequeue, and drop packets based on this priority.

We also assume that routers can provide Congestion Notification messages upstream to the sender with information about the router's congestion status. These messages are sent based on the specifications of backward explicit congestion notification (BECN) [20].

A. Queue Management for AF

AF provides assurance of packet delivery with a high probability. It also encourages exceeding user's pre-assigned profiles with the understanding that the excess traffic is not expected to receive the same high probability of delivery. IETF RFC2597 [19] recommends providing four different classes of AF and allocation a certain amount of resources to each class at each AF-compliant router. In the same RFC, three drop precedence levels (within each AF class) are also recommended. As the name suggests "drop precedence" is used in priority-dropping of packet within each AF class. It determines which packets should be dropped first within each class when needed.

Implementing priority-dropping in AF is based on AQM techniques. In particular, AF is usually based on variants of random early detection (RED) [21]. RED is a buffer management technique that is used for congestion avoidance in IP networks. RED routers try to early detect upcoming congestion by computing an average of the queue size in the router. A sustained long queue is a sign of network congestion. When a packet arrives, a RED router checks the average queue size against specified $\min_{\text{th}}$ and $\max_{\text{th}}$ thresholds. Based on this check, one of three actions is taken.

```
IF (QueueAverageLength < min_th)
  THEN no action is taken
IF (min_th < QueueAverageLength < max_th)
  THEN with probability, the packet is
dropped
IF (QueueAverageLength > max_th)
  THEN packet is dropped
```

We assume that AF routers support one of RED's extensions for service differentiation. We namely consider both RIO (RED with In/Out bits) [22] and WRED (Weighted RED) [23]. Both RIO and WRED maintain a single queue for each AF class. Then, within a certain AF class, they maintain a different set of parameters for each drop precedence level and treat each of these levels as a different virtual queue. The difference between RIO and WRED is that WRED uses *one* average queue length to make dropping decisions while RIO uses *two*[1] averages to make dropping decisions. WRED calculates its average queue length based on the total number of packets in the queue. RIO maintains one average for the number of *in* (most important) packets and another average for the *out* packets. The number of RIO *out* packets can be calculated based on the total number of

[1]two or more according to the number of drop precedence levels

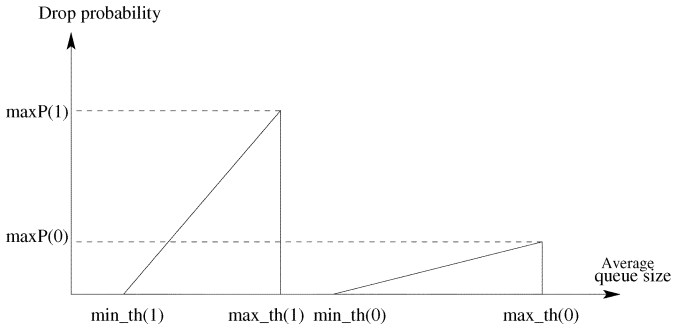


Fig. 1. Staggered parameter configuration of a two-priority RED queue.

packets in the whole queue (coupled mode or RIO-C) or based on the number of *out* packets only (de-coupled mode or RIO-D).

In this paper, we present the results of using a queue model with two drop precedence levels. However our architecture along with the simulation models we developed can support more priority levels without any change. We use the staggered configuration of parameters. This is setting $\min_{\rm th}(n-1) > \max_{\rm th}(n)$, where level $n-1$ is higher in priority than level n within a certain class. Check Fig. 1 for the staggered parameters setting of a RED queue with two priorities. *This configuration offers the maximum sheltering and isolation of classes* for both RIO and WRED. For RIO, in addition to staggered parameters, the choice of coupled versus de-coupled mode determines which class is more sheltered. The coupled mode (RIO-C) drops the lower priority class in favor of the higher priority class which is more protected. RIO-C also allows bandwidth borrowing among classes when the load is light. In the de-coupled mode (RIO-D), lower priority is more protected than RIO-C because in this case its average queue length is based solely on the number of lower priority packets in the queue.

To summarize, in this paper we will be using

- a single RIO or WRED queue for a single class AF service;
- two drop precedence levels within this class;
- the staggered setting of parameters is used to set the drop precedence levels as in Fig. 1;
- we do not use policing/shaping at network ingress points. Instead, we do flow control and packet marking at the sender.

B. Backward Explicit Congestion Notification

As we mentioned above, in RED buffer management, if the queue size is between its $\min_{\rm th}$ and $\max_{\rm th}$ thresholds, the packet is dropped with a probability. Explicit congestion notification (ECN) [24] was proposed for use with TCP With ECN, instead of dropping with probability while between the two threshold $\min_{\rm th}$ and $\max_{\rm th}$, the packet is marked and sent to the receiver. The receiver in this case, marks a flag in the TCP header of an ACK message and sends it back to the sender. Based on the information in this ACK the sender reacts by reducing its congestion window as well as its slow start threshold. The sender then sends some notification to the receivers that it did that to stop the receiver from sending more marked ACKs back. This mechanism forces the TCP sender to react early before congestion develops without the need to drop packets. However, this method has some limitations.

1) This approach is coupled with TCP.
2) It takes a RTT before the sender reacts.

In [20], the authors proposed using feedback at the IP layer that should result in the same sender reaction. This feedback message is sent if the queue size is between its $\min_{\rm th}$ and $\max_{\rm th}$ thresholds or if it is greater than $\max_{\rm th}$ threshold. The packet is still marked to prevent other routers from sending more feedback messages for the same packets. They called it BECN. This is done using the existing IP signaling mechanism, ICMP. Sending an ICMP source quench (ICMP SQ) message to the sender from the router has an advantage over ECN which is the lower time it takes before the sender can react. Also, because it is an IP-level mechanism, it can work with transport protocols other than TCP.

ECN [24] is not suitable for our real-time architecture as it takes a round-trip time (RTT) before the sender can react to congestion. We use BECN-style feedback from the router to the flow controller of the multicast sender that works on top of UDP. We send these messages back to the sender based on the status of every virtual queue, thus sending back information on which precedence level is experiencing problems. This information is used by our rate-adaptation algorithm to change the sending rate and to decide on the proportion of packets marked with each drop precedence level. It is noteworthy to mention that each feedback message is 40 bytes long.

III. End-to-End Architecture

We build an end-to-end architecture on top of the network model described in Section II. The results presented in this paper are based on testing the architecture in the context of multicasting adaptively-encoded MPEG4 video. For the multicast architecture, the follwoing are true.

- We do not define an active role for the receivers in our architecture other than deciding to join a multicast session.
- The architecture is based on the encoder's ability to produce video packets in two (or more) groups of packets that have different relevance to the decoding process at the receiver. In other words, some packets are very important to receive in order to be able to decode basic quality of the video, other packets (which are less important to receive) improve the video if they are included in the decoding process.
- The sender marks the packets with two (or more) different priorities[2] with basic information marked with high priority and enhancement information is marked with lower priority. Thus, during congestion basic quality can still be received. This results in emulating the layered approach (refer to Section I-A) within one stream. This removes the burden of dealing with different layers (multicast groups) at the receiver and ensuring that all packets will follow the same multicast tree.

[2]In the rest of this paper, we refer to "packet drop precedence levels" as "packet priorities" and we use the term "layer" to denote the packets marked with a certain "priority."

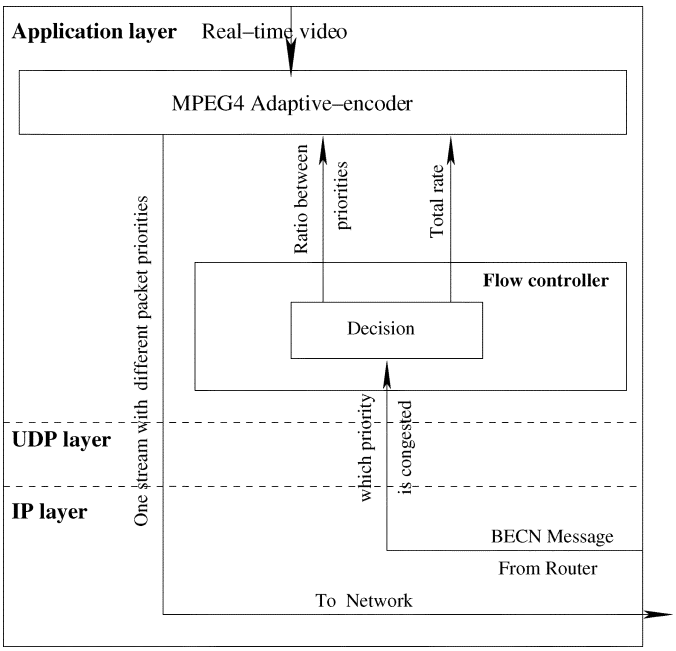


Fig. 2. Protocol stack at the sender.

- While congestion is developing, routers that run RIO (or WRED) will send back BECN messages to the sender with information on the priority level that caused the problem (i.e., which virtual queue).
- The sender runs a rate-adaptation algorithm to search for an operating point (total sending rate and ratio between priority levels) that will reduce this feedback messages rate. The protocol stack at the sender is shown in Fig. 2 where we only show the parts that we added or modified. The functionality of the rate-adaptation algorithm is represented by the *Decision* box of Fig. 2. Discussion of the algorithm follows in detail in Section IV.
- The major contribution of our architecture is how it deals with heterogeneity of receivers. The sender will try to lower the sending rate for the high priority to a rate suitable for the slowest receiver. This will allow the receiver to get useful information in the time of persistent congestion while marking excess packets with lower priority to allow faster receivers higher utilization of their bandwidth.

The motivations/advantages of this work can be summarized in the following points.

1) Sending packets as one stream is much easier to handle than multiple streams and ensures that they all follow the same multicast tree.
2) Sending all layers within one stream with different priorities over a network that supports priority dropping ensures that a minimum quality will be received in the time of network congestion.
3) Scalability issues is minimized when the feedback to the sender is provided from the routers rather than from receivers.
4) The required router functionality are already in use in today's routers or at least have been proposed for use. So this architecture will not present the network with added complexity.

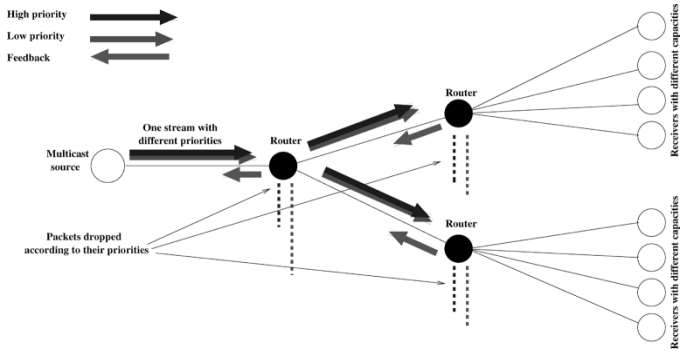


Fig. 3. End-to-end architecture.

An illustration of the end-to-end architecture is shown in Fig. 3.

For this architecture to operate in unicast mode, the rate-adaptation algorithm behaves differently at the lower priority by lowering the rate in the same fashion of the high-priority level, because there is only one receiver to accommodate. In the rest of this paper, the multicast case is discussed as it is the more complex one. We will clarify the difference between unicast and multicast at any point where that difference exists and will summarize these differences in Section VII.

IV. The Rate-Adaptation Algorithm

A. The Rate-Adaptation Equation

Assume that MPEG4 traffic is generated at the source and divided into L layers marked with L different priorities of the same AF class. All layers of the same source sent in one multicast stream. Also, we assume that this is the number of different priorities (and hence virtual queues) recognized at the routers for this AF class. Let $R_i(t), 1 \le i \le L$, be the rate (in packets/s) of layer i at the source at time t. We also consider

$$P_i(t) = P_i^{\text{Max}}(t) + P_i^{Send}(t)\, P_i^{\text{MinMax}}(t)$$

where $P_i(t)$ is the probability that virtual queue i will generate a feedback message at time t. Also, at time t, we have

$$P_i^{\text{Max}}(t) = \text{Prob}\left\{\text{QueueSize}(i) \ge \max_{\text{th}}\right\}$$

$$P_i^{\text{MinMax}}(t) = \text{Prob}\left\{\min_{\text{th}} \le \text{QueueSize}(i) \le \max_{\text{th}}\right\}$$

$$P_i^{Send}(t) = \text{Prob}\{\text{Send feedback message} \mid \min_{\text{th}} \le \text{QueueSize}(i) \le \max_{\text{th}}\}.$$

We derived $P_i(t)$ from the specification of BECN [20].

Considering the changes from *old* to *new* values of $R_i(t)$ and $P_i(t)$ in a small interval Δt, we use the following equation to update the rate $R_i(t)$:

$$R_i^{\text{new}} = R_i^{\text{old}}(1 - \alpha_i \Delta P_i), \quad 0 < \alpha_i < 1 \tag{1}$$

where

$$\Delta P_i = P_i^{\text{new}} - P_i^{\text{old}}.$$

The rationale behind using this equation is to always change $R_i(t)$ in the opposite direction of change of $P_i(t)$ with a step α_i. We change α_i to control how much $R_i(t)$ changes in reaction to changing network conditions. $|\Delta P_i|$ can assume values between 0 and 1. At these extreme values, changes in R_i^{new} can be either no change at all (0%) or very high (100%). We select $\alpha_i = C_i\sqrt{|\Delta P_i|}$ where C_i is a constant for layer i and $0 < C_i < 1$. This sets the maximum rate change (when $\Delta P_i = 1$) to $C_i R_i^{\text{old}}$ at layer i. The choice of square-root function was motivated by our design goal of being able to react to very small changes of network conditions (when $|\Delta P_i| \leq 0.1$) as the square root of these small values is greater than the actual value ($\sqrt{x} > x;\ |x| \leq 1$). This helps to react to congestion while it is developing. We experimented with functions other than the square root but this one gives the smoothest rate changes. Through simulations, we found that operating with values of C_i ranging from 0.05 to 0.25 keeps the system stable. A value of 0.1 at the high-priority layer gave the best performance based on the criteria of: 1) minimum packet loss ratio in the high-priority layer and 2) matching of the source's rate to receivers' bandwidth capacities.

Equation (1) is subject to the constraints

$$R_i^{\min} \leq R_i \leq R_i^{\max}$$
$$R^{\min} \leq \sum_{i=1}^{L} R_i \leq R^{\max}$$

where $R_i^{\min}$ and $R_i^{\max}$ are the value limits of the rate at layer i, respectively, and $R^{\min}$ and $R^{\max}$ are the limits for the total source rate. These values depend on the limitations imposed by the video encoder, its outgoing link speed, and the minimum accepted video quality at the receiver.

B. Round-Trip Time (RTT)

Routers send feedback messages to the sender with values of P_i^{new} that indicate the congestion status of the routers. The sender will evaluate the feedback from all routers every Δt and decide on a new rate R_i^{new}. Routers calculation of P_i^{new} is presented in Section IV-D. The value of Δt will depend of the sender's estimation of the RTT from the routers that send the feedback information. We select the RTT value that corresponds to the router that has the worst situation at the high-priority layer. That is, the router with $\text{Max}(P_i^{\text{new}})$ in its feedback message.

C. Feedback Suppression

The value of $P_i(t)$ will be estimated by routers and sent back to the sender. To reduce feedback, routers will send feedback messages with a probability instead of sending a feedback message for every packet that causes a problem. From simulations, we found that sending 2%–5% of the feedback messages kept feedback volume reasonable and in the same time kept the sender responsive to changes in network conditions.

D. Calculation of Probabilities

At each router, the quantities $P_i^{\text{Max}}(t)$ and $P_i^{\text{MinMax}}(t)$ are calculated using real-time measurements from the network rather than being based on an analytical model. The reason for this is that in the general case where all kinds of traffic flows are coming into the routers queues, it is very hard to assume a certain model for the input traffic.

We bias the probability estimation by giving more weight to newer values to make the estimate a better representative of the current state of the network. Otherwise, after a long time of operation, these values will converge to a constant value. We used the scheme used in [14] for measuring loss intervals. The probability is observed at each virtual queue i in k subsequent time intervals and give these intervals different weights w_i, $1 \leq i \leq k$. The length of the time interval is based on the speed of the link(s) incoming into the router queues. To calculate $P_i(t)$ (whether $P_i^{\text{Max}}(t)$ or $P_i^{\text{MinMax}}(t)$) at the end of an interval m we use

$$P_i(m) = \frac{\sum_{j=1}^{k} w_j\, P_{m-j}}{\sum_{j=1}^{k} w_j}. \tag{2}$$

Values of w_i are chosen so that recent probabilities have the same weight, while weights for older probabilities are smaller. The choice of a value for k is important. Large value of k will result in a smooth estimation of $P_i(m)$ while very large values of k will reduce the sender's responsiveness to changes in network conditions. For most or our simulations, we used $k = 10$, and $w = \{4, 4, 4, 4, 4, 2, 2, 2, 1, 1\}$. $P_i^{Send}(t)$ is calculated using the approach in [21]. It depends on the average queue size and on the RED parameters.

E. Changing the Equation Parameters

Routers calculate values of P_i^{new} and send them back to the sender that should select one of them as a new value of P_1^{new} every Δt seconds. There are different criteria that we consider for the sender to change $P_i\text{new}$.

- *At the highest priority layer,* the sender selects $\text{MAX}(P_i^{\text{new}})$ received during Δt. This results in accommodating of the router with the worst congestion situation. This is done subject to the constraint $R_i(t) \geq R_i^{\min}$ to avoid the problem where a slow portion of the receivers drag the sending rate down dramatically[3].
- *At lower priority layers,* the sender selects $\text{MIN}(P_i^{\text{new}})$ received during Δt. This results in maximizing $R_i(t)$ at layer i subject to $R_i(t) \leq R_i^{\max}$. The reason behind this is to let receivers with extra available bandwidth utilize their links. Slow receivers will have these packets dropped by their routers.

F. The Algorithm for Changing R_i

Now we describe the algorithm used by the sender to calculate R_i^{new} every Δt (or RTT) seconds.

```
REPEAT every RTT
  REPEAT for every layer i
   If Nofeedback
    increase Ri by 1%
```

[3]This is known as the "drop to zero" problem.

```
    else If (ΔP_i > 0)
     reduce R_i using (1)
    else If (ΔP_i < 0)
     If (i NOT highest priority layer)
      increase R_i using (1)
    else If (ΔP_i = 0)
     If P_i^new > 0.9
      reduce R_i by 3%
  END REPEAT for every layer i
END REPEAT every RTT
```

Note that the values of 1% and 3% are chosen to conservatively increase/decrease the rate as at this point we can not be sure exactly which direction ΔP_i is moving. Note also that the rate of the highest is increased only when there is no feedback to make sure that it is not increased beyond slow receivers' capacities.

V. Simulation

A. Setup

We carried out simulations using the network simulator *ns* [25] version 2.1b8a. Only one AF class is assumed with two drop precedence levels (priorities). In one experiment (Section VI-B, we compare the performance of our proposal in the case of using RIO-C, RIO-D, or WRED (refer to Section II) while in all other experiments only one of those models is used. Parameters configuration ($\min_{\text{th}}$ *and* $\max_{\text{th}}$) at the routers for these priority levels are the same in all experiments. The results we show are for simulations that are 300 s long. We use $C_i = 0.1$ [of (1)] for both priority levels [i =1, 2] except for the TCP experiment where we use $C_i = 0.25$ (Section VI-E). The simulation starts with equal rates at the two priority levels ($R_1(t) = R_2(t)$).

The authors have tried to report as much results in this paper as they could. However, interested readers should refer to [26] for more simulation results.

B. Generating Adaptively-Encoded MPEG4 Using a Transform Expand Sample (TES) Model

Our work in this paper depends on MPEG4's ability of adaptive encoding [27]. We developed a traffic generator [28] that can be used for studying MPEG4 behavior and performance through simulation using the TES methodology [29]–[31]. The traffic we generate closely matches the statistical characteristics (in terms of marginal distribution and auto-correlation function) of an original real trace of an MPEG4-encoded video. MPEG4 encoders generate video in three different frame types (I, P, and B) that serve to encode different portions of the video signal in different levels of quality. We modeled the I, P, and B using three TES models and used multiplexing to generate the original sequence of frames for MPEG4. Using feedback messages from the network, we recalculate a new target rate for the MPEG4 encoder and generate video packets based on this rate while maintaining the statistical properties of the original MPEG4 trace. We implemented this generator in software and integrated it into the network simulator [32]. The characteristics of the MPEG4 traces we used can be found in [33].

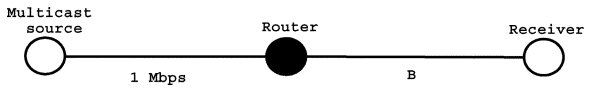


Fig. 4. Basic test topology.

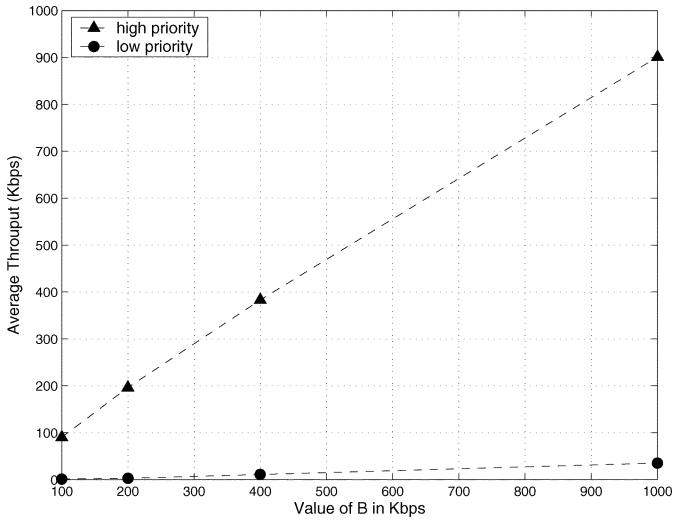


Fig. 5. Adapting to bandwidth B.

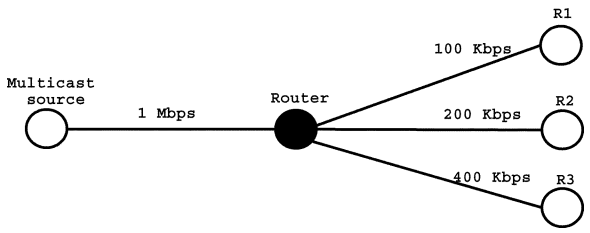


Fig. 6. Heterogeneity test topology.

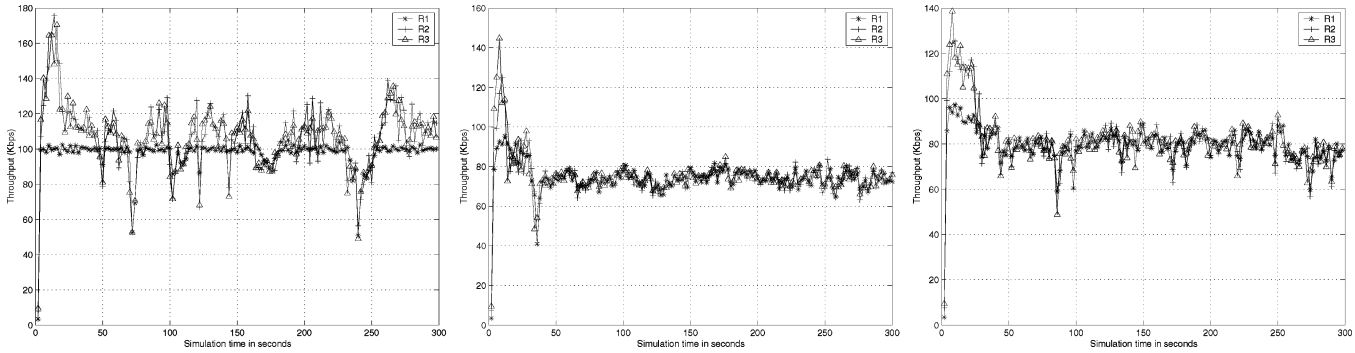
VI. Discussion of Results

A. Basic Test

In this basic test, the goal is to check how the sender will adjust its rate for the high-priority packets to closely match the bandwidth available at the receivers. In this experiment, we only have one sender and one receiver (Fig. 4), so the sender should match its rate of the high priority to the bandwidth B with low packet loss. For each of the following values of B, 100 kbps, 200 kbps, 400 kbps, and 1 Mbps, the simulation (of 300 s) is repeated and the average rate at each priority level is plotted against values of B in Fig. 5. We can see that the average rate at the high-priority level is very close to the available bandwidth (i.e., 100 kbps, 200 kbps, etc.) while the low-priority rate is a very small percentage of the available bandwidth. This experiment uses RIO-C for queueing.

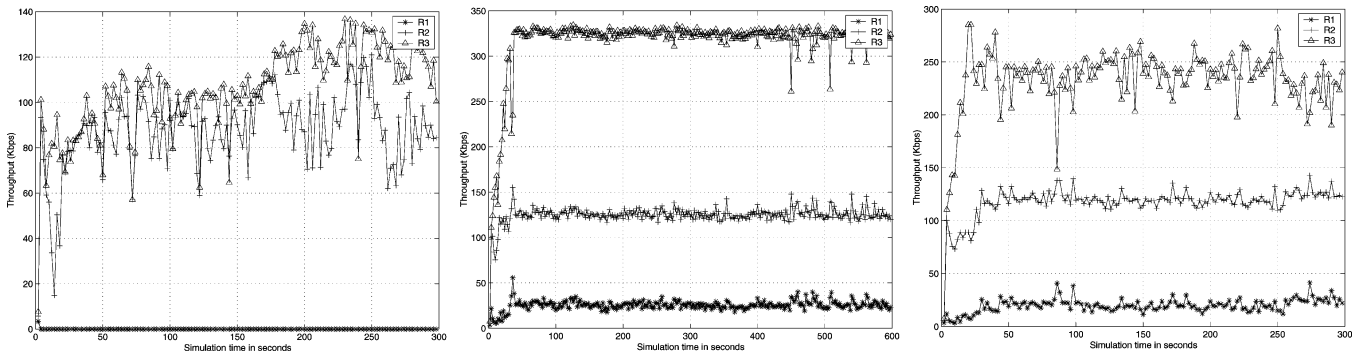
B. Heterogeneity Test

This experiment is a test of how the architecture (with its rate-adaptation algorithm) deals with heterogeneity. We simulate the topology of Fig. 6 where one sender is multicasting MPEG4 encoded video to three receivers who have different bandwidths (100 kbps, 200 kbps, and 400 kbps) through one



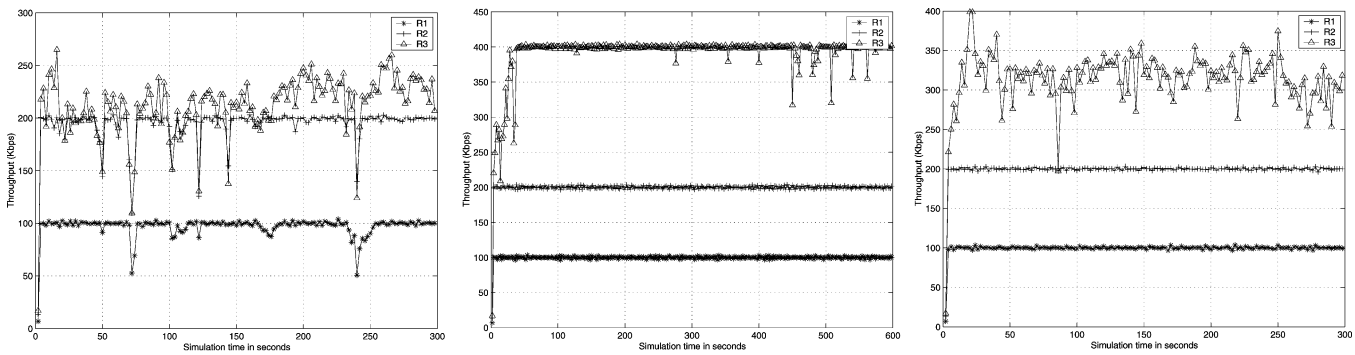
(a) (b) (c)

Fig. 7. High-priority throughput. (a) Using RIO-C. (b) Using RIO-D. (c) Using WRED.



(a) (b) (c)

Fig. 8. Low-priority throughput. (a) Using RIO-C. (b) Using RIO-D. (c) Using WRED.

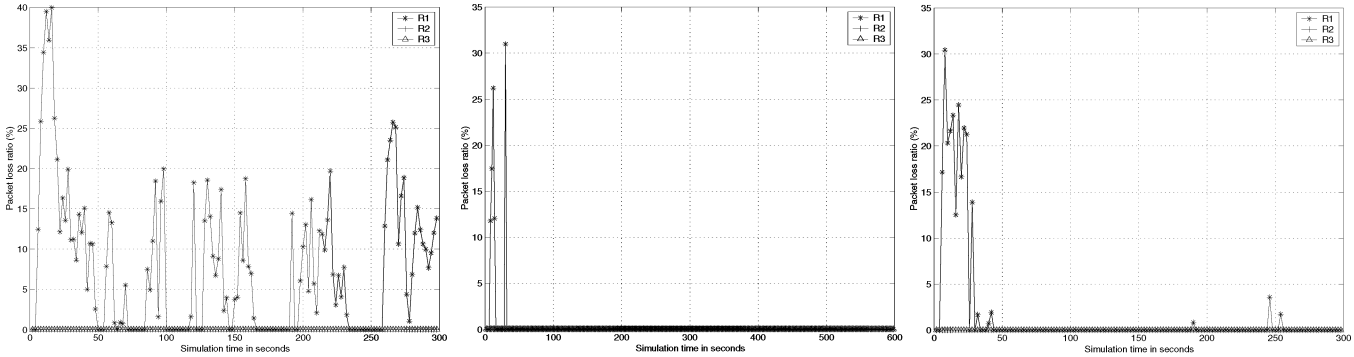


(a) (b) (c)

Fig. 9. Total throughput. (a) Using RIO-C. (b) Using RIO-D. (c) Using WRED.

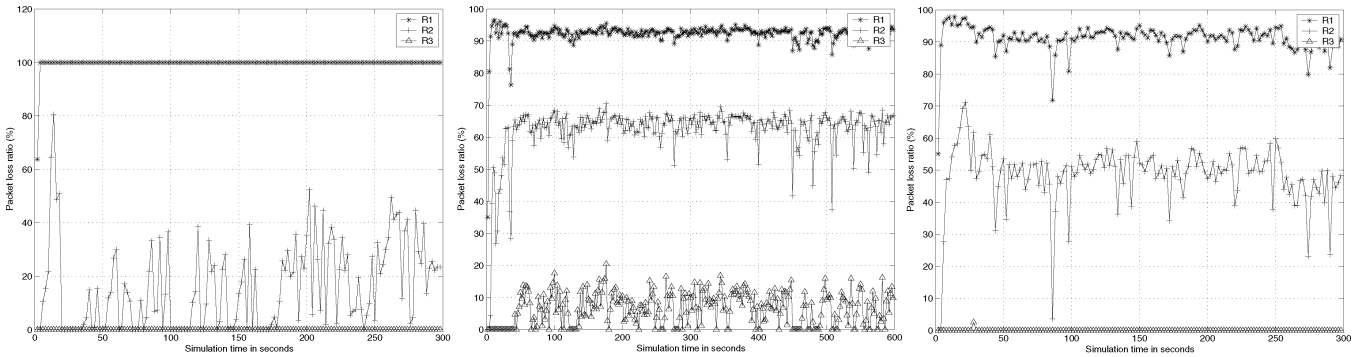
router. In addition to this goal, this experiment is repeated three times to compare the use of RIO-C, RIO-D, and WRED.

- *Throughput of high-priority layer:* The throughput is shown in Fig. 7. We can see in the figure that the algorithm tries to bring down the rate for the high-priority layer (for the three receivers) to a rate less than a 100 kbps, which is the rate of the slowest receiver. The difference between parts (a), (b), and (c) of Fig. 7 is due to the way each queuing mechanism protects its highest priority level. Note that in all simulations we used the same set of parameters for the two priority levels. In RIO-C, this layer gets all of the 100 kbps, while in RIO-D and WRED it gets a lower rate because the lower priority layer is more protected with these two mechanisms. Also in this figure, we can see that highest priority layer is more aggressive with RIO-C. Both RIO-D and WRED offer more isolation and protection and RIO-D offers best stability at this layer.
- *Throughput of lower priority layer:* This is shown in Fig. 8. In this layer, the algorithm allows receivers with higher capacities to better utilize their bandwidth. The best performance comes with RIO-D because of its isolation of classes. It allowed each receiver its maximum share at this layer. Although this is not the best option of R1 as it does not make use of the low priority in this case with a high loss rate as seen in Fig. 11. WRED is better than RIO-C but still not as good as RIO-D. The reason of the lower rates of RIO-C at this layer is that RIO-C protects the high-priority layer. This protection lets the high-priority layer be aggressive and eventually brings down the rate down for the lower priority layer.



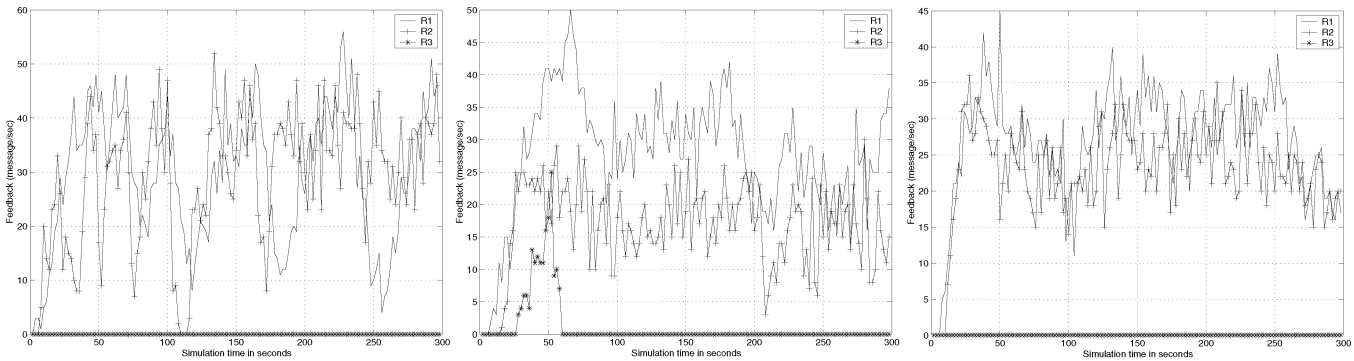
(a) (b) (c)

Fig. 10. Packet loss in high priority. (a) Using RIO-C. (b) Using RIO-D. (c) Using WRED.



(a) (b) (c)

Fig. 11. Packet loss in low priority. (a) Using RIO-C. (b) Using RIO-D. (c) Using WRED.



(a) (b) (c)

Fig. 12. Rate of feedback messages. (a) Using RIO-C. (b) Using RIO-D. (c) Using WRED.

- *Total throughput:* Fig. 9 shows the summation of the throughput at both layers for each receiver. It is clear that maximum utilization is achieved with RIO-D. This, however, should not be taken as an absolute metric for judging the performance, as we will comment in the Conclusion in Section VIII.
- *Packet loss at high-priority layer:* In Fig. 10, we can see that the loss ratio at the high layer for R2 and R3 is zero all the time. However, for R1, while the loss ratio is high only in transient phase in the beginning of the simulations for RIO-D and WRED, it has high values at different times during the simulation for RIO-C. The reason is as we mentioned earlier that RIO-C favors the high-priority layer and allows this layer to become

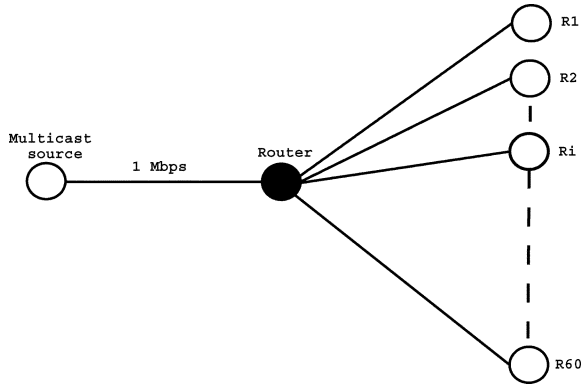


Fig. 13. Scalability test topology.

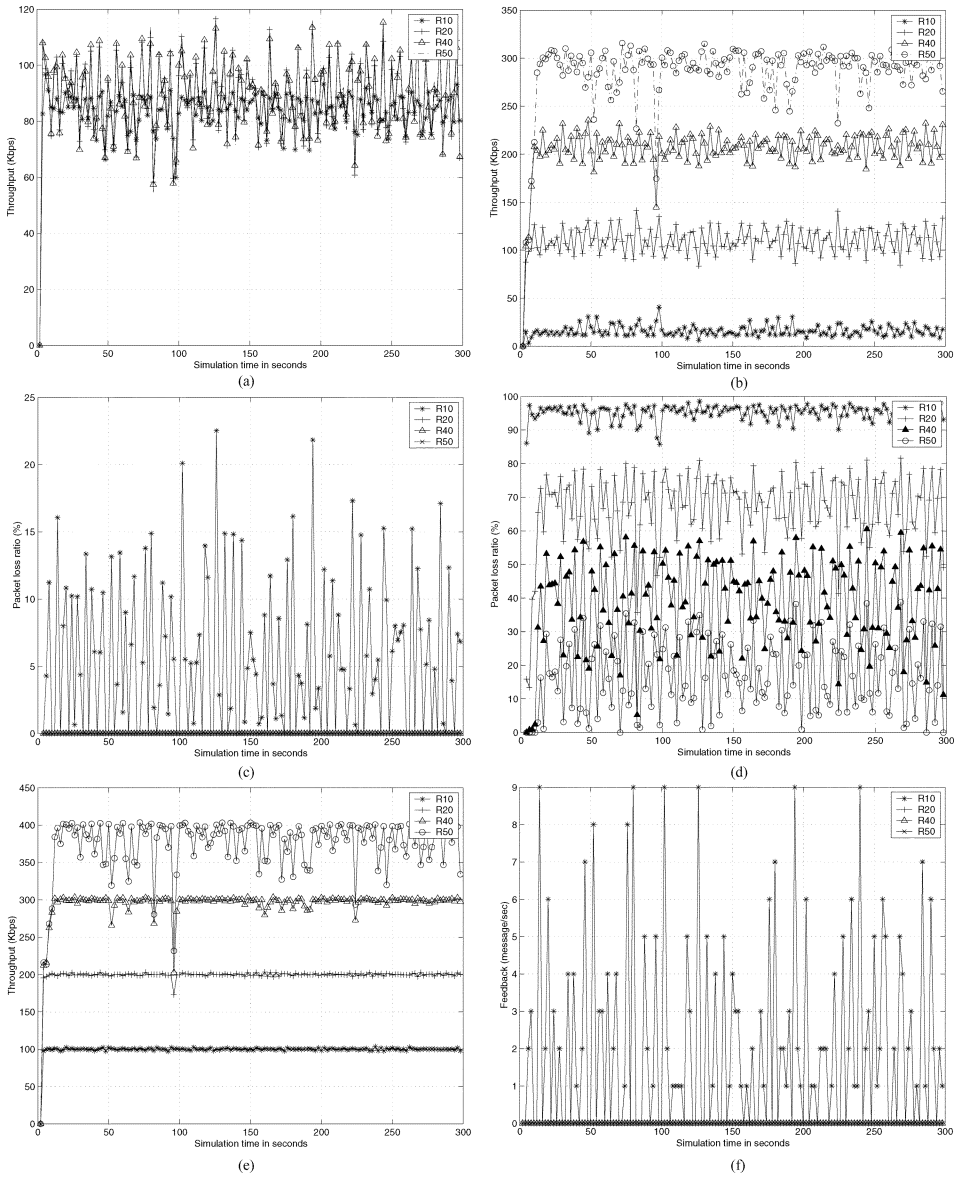


Fig. 14. Scalability test. (a) High-priority throughput. (b) Low-priority throughput. (c) Packet loss in high priority. (d) Packet loss in low priority. (e) Total throughput. (f) Feedback messages.

aggressive. This results in overshoots of the rate at this layer causing these bursts high of losses as in Fig. 11(a).

- *Packet loss at low-priority layer:* The reader should refer to Fig. 11. It can be seen that RIO-C has the lowest loss rate at this layer for the three receivers because it does not allow the rate at this layer to go as high as the cases of RIO-D and WRED.
- *Feedback:* In Fig. 12, the rate of feedback messages in message/s is shown. The results are consistent in the three cases where R3 generates the lowest feedback and R1 and R2 generate more feedback. Note that these messages have a very small size compared to the data sent (one message is 40 bytes). For most of our simulations, the feedback measured at the source upstream is around 5% of the data delivered downstream to the receivers.

C. Scalability Test

In this experiment, the topology of Fig. 13 is simulated for 300 s to test how the architecture handles a larger number of receivers. There is one sender and 60 receivers. Receivers 1 to 15 have a 100-kbps link each, receivers 16 to 30 have a 200-kbps link each, receivers 31 to 45 have a 300-kbps link each, and receivers 46 to 60 have a 400-kbps link each. The link between the sender and the router is 1 Mbps. Based on the results from Section VI-B, we use RIO-D.

Fig. 14 shows a representative receiver of each receiver group. We repeated the simulations several times and found out that receivers with the same bandwidth have identical performance in term of the metrics we present here. R10 represents the 100-kbps receivers, R20 represents the 200-kbps receivers, R40 represents the 300-kbps receivers, and R50 represents the 400-kbps receivers. Parts (a), (b), and (c) show the expected rates for the four receivers. They are compliant with results

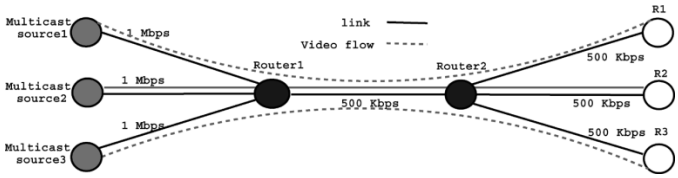


Fig. 15. Topology for testing multisession interaction.

from Section VI-B. Part (d) shows that all receivers have zero packet loss except for R10 that has an average loss of approximately 5%. Part (e) shows the differentiation between receivers in loss of the lower priority. Taking parts (a)-(e) into consideration, we come to the realization that receivers get the same rate at the high-priority layer while getting different rates with different packet loss ratios at the lower priority level. *Part (f) shows that the feedback from each receiver is much lower than the case of Fig. 12. The aggregate of these receivers (the 100-kbps group) is close to that of R1 in Fig. 12. This experiment shows that the architecture scales well with an increasing number of receivers.*

D. Multisession Interaction

This experiment shows how multiple multicast sessions using our architecture would share the bandwidth of a bottleneck link. In Fig. 15, three senders are multicasting to three receivers in three one-to-one sessions as shown in the figure. The three sessions compete for the link between Router1 and Router2 which is slower than the aggregate of the three incoming links between the senders and Router1.

Fig. 16 shows the allocation of bandwidth of the Router1-Router2 link between the three sessions. Part (a) shows that at the high-priority level, they share the bandwidth of the link fairly with an average rate of 170 kbps, each while at the lower priority level their rates tends quickly to zero.

E. Interaction With TCP

This final experiment tests the very important aspect of bandwidth sharing with TCP. Fig. 17 shows that we have one multicast session and one TCP session. We use Tahoe TCP and we assign TCP traffic to the high-priority level. Part (a) of Fig. 18 shows that the multicast session gets a higher rate but by calculating the averages we find out that the average rate for TCP is 210 kbps while that of the multicast session is 280 kbps. This is a ratio of 3:4, which is acceptable given the fact that the multicast traffic represents a UDP-based multimedia application. We also have to mention that in order to get this result, C_0 should be set to value near the high end of the range discussed in Section IV-A. Part (b) shows the difference of packet loss between the two competing flows. Finally part (c) shows that our architecture generates a relatively low feedback message rate given the fact that our feedback message has almost the same size as the TCP ACK. This is not provided for the sake of comparison, as TCP is expected to generate more feedback because it acknowledges every packet for 100% reliable transmission. Part (c) is intended to show that the feedback generated by our architecture can be acceptable in an operational network.

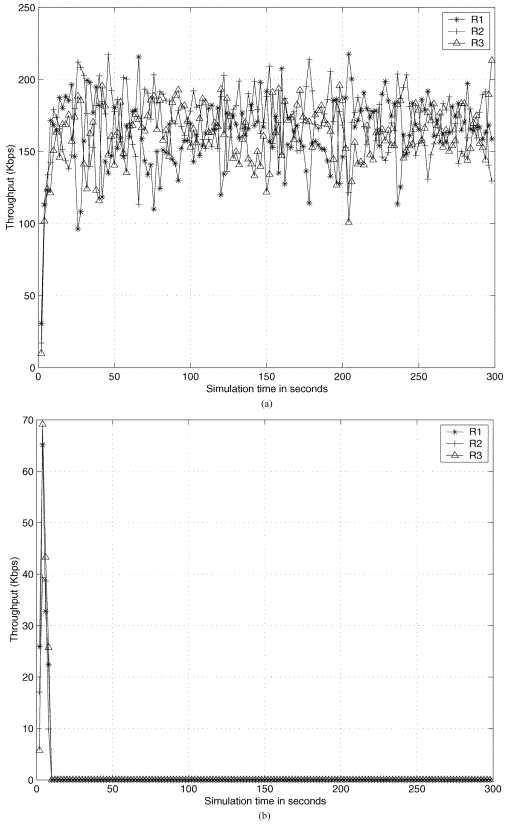


Fig. 16. Multisession interaction. (a) High-priority throughput. (b) Low-priority throughput.

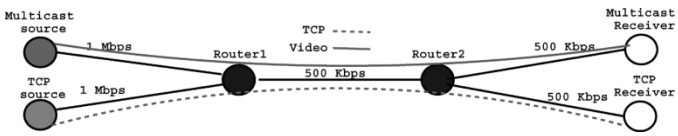


Fig. 17. Topology for testing interaction with TCP.

VII. Design Issues and Limitations

In this section, we present some notes on our design and point-out the limitations of our approach.

- The values of $R_i^{\min}$, $R_i^{\max}$, $R^{\min}$, and $R^{\max}$ are not enforced in our simulations. So the ratio of R_0 to R_1[4] may not be practical in our results, but we do that intentionally in our simulations to test the control mechanism we propose without imposing limitations on the rates. Applying these limits to the methodology presented in this paper will result in a better and more practical performance of our methodology.
- In our approach, we naturally assume that multicast receivers should at least have their bandwidth grater than $R_0^{\min}$ (minimum rate at the high-priority layer).
- To accommodate the higher heterogeneity of the receiver's bandwidth, a higher number of priority levels may be used. We note here that neither the architecture nor the rate-adaptation algorithm needs to be changed in order to support a higher number of layers.

[4]the high-priority layer is denoted by 0 and the lower layer is denoted by 1

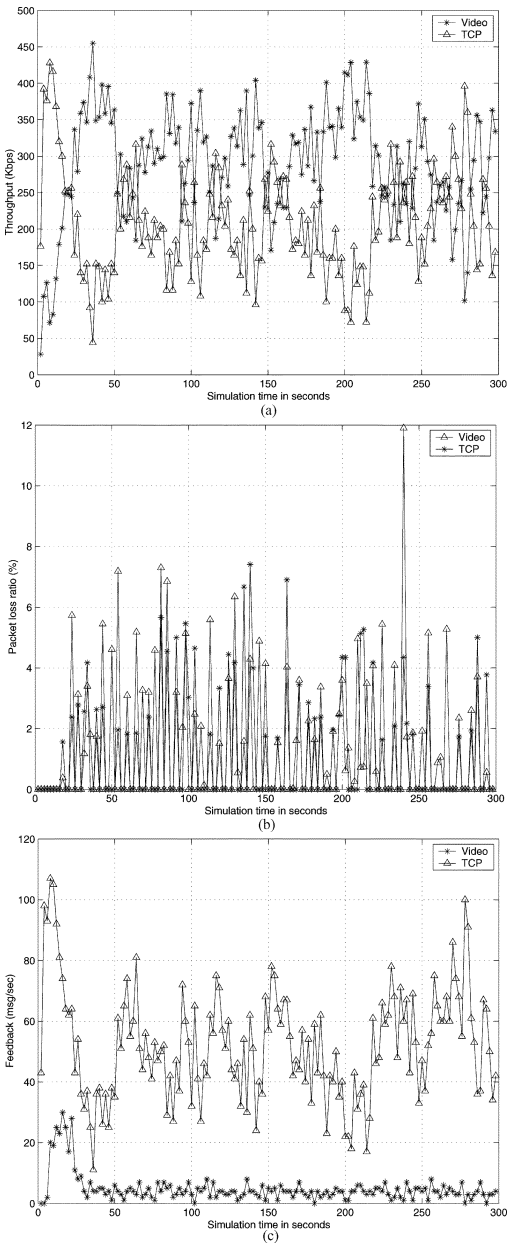


Fig. 18. Interaction with TCP. (a) Throughput of the high priority. (b) Packet loss in high priority. (c) Feedback of TCP and video.

- We drop a low-priority *multicast* packet at the input interface of a router if the virtual queue (of the output interface) that the packet is supposed to be forwarded to is experiencing a high loss rate. This keeps these queues operating with smaller average lengths and allows nonmulticast traffic to get a better share of the queues at the routers' output interfaces. Note that the packet loss ratio we show in these graphs are the total of those dropped at the input interface and in the virtual queues.
- Our work is also applicable to the *unicast* case although we do not present these results here. The difference in the unicast mode is the consideration of $\max(P_i^{\text{new}})$ at all priority levels so that the rate at each layer meets what is available at the single receiver of the unicast flow. Also in unicast, we do not apply the dropping at the input interface described above.
- The architecture assumes that there will be loss at lower priority levels and hence we recommend the use of some forward error correction (FEC) mechanism at the lower priority levels for applications that require a higher degree of reliability.

VIII. Conclusion

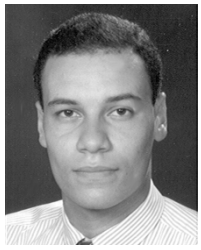
We have presented an architecture and a rate-adaptation algorithm for real-time video transport in AF networks. It can operate in either unicast or multicast modes and we presented the simulation results for the multicast case. We showed how it enables users with different bandwidth capabilities to receive the same video multicast in different qualities. This differentiation is based on encoding the video into two levels of priorities (more levels can be used without change in the methodology). We can see how the algorithm always attempts accommodating the slowest receiver at the high-priority level and suitably increases the rate at the lower priority level.

The comparison between the three queuing mechanisms we considered reveals that selecting one of them is not a trivial task. While RIO-D and WRED result in better utilization of bandwidth, they also result in high loss rate in the lower layer to a level that may render it undesirable (check loss rate of R2 at lower layer). On the other hand, RIO-C offers different qualities with lower loss rates at the expense of less bandwidth utilization.

References

[1] S. McCanne, "Scalable Compression and Transmission of Internet Multicast Video," Ph.D. dissertation, Univ. California, Berkeley, 1996.
[2] X. Li, S. Paul, and M. Ammar, "Layered video multicast with retransmission (LVMR): Evaluation of hierarchical rate control," in *Proc. INFOCOM*, 1998, pp. 1062–1072.
[3] A. Legout and E. Biersack, "PLM: Fast convergence for cumulative layered multicast transmission schemes," in *Proc. SIGMETRICS*, 2000, pp. 13–22.
[4] L. Vicisano, J. Crowcroft, and L. Rizzo, "TCP-like congestion control for layered multicast data transfer," in *Proc. INFOCOM*, San Francisco, CA, Mar. 1998.
[5] L. Wu, R. Sharma, and B. Smith, "Thin streams: An architecture for multicasting layered video," in *Proc. NOSSDAV*, St. Louis, MO, May 1997.
[6] J. Byers, M. Luby, and M. Mitzenmacher, "Fine-grained layered multicast," in *Proc. INFOCOM*, Anchorage, AK, Mar. 2001.
[7] J. Byers, M. Luby, M. Mitzenmacher, and A. Rege, "A digital fountain approach to reliable distribution of bulk data," in *Proc. SIGCOMM*, Vancouver, BC, Canada, Aug. 1998.
[8] A. Matrawy, I. Lambadaris, and C. Huang, "On layered video fairness on IP networks," in *Proc. IEEE GLOBECOM*, San Antonio, TX, Nov. 2001.
[9] A. Legout and E. Biersack, "Pathological behaviors for RLM and RLC," in *Proc. NOSSDAV*, Chapel Hill, NC, Jun. 2000.
[10] R. Gopalakrishnan *et al.*, "Stability and fairness issues in layered multicast," in *Proc. NOSSDAV*, Basking Ridge, NJ, Jun. 1999.
[11] J-C. Bolot, T. Turletti, and I. Wakeman, "Scalable feedback control for multicast video distribution in the internet," in *Proc. ACM SIGCOMM*, London, U.K., Aug. 1994.
[12] D. DeLucia and K. Obraczka, "Multicast feedback suppression using representatives," in *Proc. IEEE INFOCOM*, Kobe, Japan, Apr. 1997.
[13] L. Rizzo, "PGMCC: A TCP-friendly single-rate multicast congestion control scheme," in *Proc. SIGCOMM*, Stockholm, Sweden, Aug. 2000.
[14] J. Widmer and M. Handley, "Extending equation-based congestion control to multicast applications," in *Proc. SIGCOMM*, San Diego, CA, Aug. 2001.
[15] A. Clerget, "TUF: A Tag-Based UDP Multicast Flow Control Protocol," INRIA, Tech. Rep. 3728, 1999.
[16] M. Luby, L. Vicisano, and T. Speakman, "Heterogeneous multicast congestion control based on router packet filtering," in *Presentation at RMRG Meeting*, Pisa, Italy, Jun. 1999.

[17] K. Nakauchi, H. Morikawa, and T. Aoyama, "A network-supported approach to layered multicast," in *Proc. IEEE ICC*, Helsinki, Finland, Jun. 2001.
[18] Z. Zhang and V. O. K. Li, "Router-assisted layered multicast," in *Proc. IEEE ICC*, New York, NY, Apr. 2002.
[19] J. Heinanen, F. Baker, W. Weiss, and J. Wroclawski, "Assured Forwarding PHB Group,", RFC 2597, 1999.
[20] J. H. Salim, B. Nandy, and N. Seddigh. "A Proposal for Backward ECN for the Internet Protocol (IPv4/IPv6)," Internet Draft. [Online]. Available: draft-salim-jhsbnns-ecn-00.txt
[21] S. Floyd and V. Jacobson, "Random early detection gateways for congestion avoidance," *IEEE/ACM Trans. Networking*, vol. 1, no. 4, pp. 397–413, Aug. 1993.
[22] D. Clark and W. Fang, "Explicit allocation of best effort packet delivery service," *IEEE/ACM Trans. Networking*, vol. 6, no. 4, pp. 362–373, Aug. 1998.
[23] "Advanced QoS services for the intelligent internet," *Cisco White Paper*, 1997.
[24] S. Floyd, "TCP and explicit congestion notification," *ACM Comput. Commun. Rev.*, pp. 10–23, Oct. 1994.
[25] S. McCanne and S. Floyd. ns Network Simulator. [Online]. Available: http://www.isi.edu/nsnam/ns/
[26] A. Matrawy, "Network-Supported Congestion Control for Video Multicasting over IP Networks," Ph.D. dissertation, Carleton Univ., Ottawa, ON, Canada, 2002.
[27] R. Koenen, "MPEG4 overview," *IEEE Spectrum*, Feb. 1999.
[28] A. Matrawy, I. Lambadaris, and C. Huang, "MPEG4 traffic modeling using the transform expand sample methodology," in *Proc. 4th IEEE Int. Workshop on Networked Appliances*, Gaithersburg, MD, Jan. 2002.
[29] B. Melamed, "An overview of TES processes and modeling methodology," *Perf. Eval. Comput. Commun. Syst.*, 1993.
[30] D. Reininger *et al.*, "Variable bit rate MPEG video: Characteristics, modeling and multiplexing," in *Proc. 14th Int. Telegraffic Congr.—ITC 14*, Antibes Juan-les-Pins, France, Jun. 1994.
[31] M. R. Ismail, I. E. lambadaris, M. Devetsikiotis, and A. R. kaye, "Modeling prioritized MPEG video using TES and a frame spreading strategy for transmission in ATM networks," in *Proc. INFOCOM*, Boston, MA, Apr. 1995.
[32] A. Matrawy and I. Lambadaris. MPEG4 Generator Code for ns. [Online]. Available: http://www.isi.edu/nsnam/ns/ns-contributed.html
[33] R. M. Guirguis and S. Mahmoud, "Transmission of real-time multi-layered MPEG-4 video over ATM/ABR service," in *Proc. IEEE ICC*, New Orleans, LA, Jun. 2000.



Ashraf Matrawy (M'03) received the B.Sc. and M.Sc. degrees in computer science and automatic control from Alexandria University, Alexandria, Egypt, in 1993 and 1997, respectively, and the Ph.D. degree in electrical engineering from Carleton University, Ottawa, ON, Canada. He was the holder of the Ontario Graduate Scholarship for Science and Technology (OGSST) for four academic years (1998–2002).

Between December 1994 and December 1997, he was a Software Engineer with one of the USAID projects in Cairo, Egypt. He is currently an Assistant Professor at Carleton University. His research interests include reliable and secure computer networking, with emphasis on groups and multimedia applications.

Ioannis Lambadaris was born in Thessaloniki, Greece. He received the diploma in electrical engineering from the Polytechnic School, Aristotle University of Thessaloniki, in 1984, the M.Sc. degree in engineering from Brown University, Providence, RI, in 1985, and the Ph.D. degree in electrical engineering from the Department of Electrical Engineering and Systems Research Center (SRC), University of Maryland, College Park, in 1991 .

He was a Research Associate at Concordia University, Montreal, QC, Canada, during 1991–1992. Between September 1992 and July 1997, he was an Assistant Professor in the Department of Systems and Computer Engineering, Carleton University, Ottawa, ON, Canada, where he is now an Associate Professor. His research interests lie in the area of applied stochastic processes and their application for modeling and performance analysis of computer communication networks. His current research concentrates on quality of service control for IP and evolving optical networks architectures.

Dr. Lambadaris received the Premier's Research Excellence Award and the Carleton University Research Excellence Award (2000–2001) for his research achievements in the area of modeling and performance analysis of computer networks. He was a recipient at a Fulbright Fellowship (1984–1985) for graduate studies in the U.S. During his undergraduate studies, he received a fellowship from National Fellowship Foundation of Greece (1980–1984). He also received the Technical Chamber of Greece Award (ranked first in his graduating class).