Dynamic QoS Control of Multimedia Applications based on RTP

Ingo Busse, Bernd Deffner, Henning Schulzrinne*
GMD-Fokus, Hardenbergplatz 2, D-10623 Berlin
{busse,deffner,schulzrinne}@fokus.gmd.de

May 30, 1995

Abstract

We describe a mechanism for dynamic adjustment of the bandwidth requirements of multimedia applications. The sending application uses RTP receiver reports to compute packet loss. Based on this metric the congestion state seen by the receivers is determined and the bandwidth is adjusted by a linear regulator with dead zone.

The suggested mechanism has been implemented and controls the bandwidth of the `vic` video conferencing system. We ran several experiments on the Internet and on our local ATM network in order to tune and evaluate the algorithm. The results indicate that the mechanism can be applied to both environments. Since an ATM network has a different loss characteristic than the Internet different controller parameters must be set.

1 Introduction

In "traditional" design of integrated services networks, multimedia applications negotiate a desired quality of service during the connection setup phase and then the network guarantees this quality, barring catastrophic failures. However, this mechanism is suboptimal for a number of reasons. First, network load may change significantly during the duration of the connection. In particular, many multimedia conferencing applications have call holding times measured in hours rather than the typical three-minute narrowband phone call. If the call arrives during a busy period, the network will permit only a lower quality, even though the network may become idle during later parts of the call. Alternatively, if the network is generous with the newly arriving call, it may have to turn away other customers as the day wears on. Also, in order to maximize customer satisfaction, a network operator should strive to give the network users the best available media quality and thus maximum use of the available bandwidth.

Unlike telephones or circuit-switched videoconferencing systems, workstation-based multimedia applications are often capable of adjusting their media quality and data rates over a wide range, particularly if software codecs are used. For example, the video frame rate can be varied from less than one frame a second to full-motion frame rate for conferencing applications. Video spatial resolution and quantization can be adjusted as well. For audio, most encodings have a single, constant rate, so that only encodings can be changed. We refer to such applications as "controllable". (Adaptive applications adjust their playout delay to the current network conditions.)

Even controllable multimedia applications will likely require a minimum guaranteed bandwidth (e.g., 32 kb/s for audio or 15 frames/second for entertainment video). If they cannot obtain that, users may prefer deferring a conversation, or using an alternate means of communication (say, email instead of a voice call). In such situations, the telephone model of preferring existing calls over newly arriving ones is probably appropriate in the majority of cases. Pricing will likely be based on the minimum (guaranteed) rate, as this

*This work was partially supported by the Commission of European Communities (CEC) under project R2116 TOMQAT.

rate represents opportunity costs to the provider. (A guaranteed minimum cannot be reassigned to somebody else.) Therefore, application control complements rather than replaces admission control. Some networks that have sufficient capacity or a large fraction of controllable applications, either traditional data applications or continuous media, can avoid explicit reservation.

Application control for multimedia extends the notion used successfully for data applications in packet-switched networks, with two major differences. First, data applications can usually usefully be reduced to near-zero throughput, particularly temporarily, meaning that most users would rather get 100 bytes/minute than none at all[1]. Most multimedia applications require a minimum bandwidth commitment for useful service. Secondly, while data services can change their rate instantaneously, continuous-media services have adaptation periods measured in tens of seconds to avoid abrupt and annoying changes in perceived quality.

We can distinguish two kinds of in-call QoS adaptation: end-system initiated and network-initiated. End-system initiated applications [1] use signaling to request additional bandwidth (or release bandwidth) from the network when either the media characteristics or quality requirements change. Network-initiated control, the variety investigated here, bases the application target data rate on network feedback. Low losses lead the application to slowly increase its bandwidth, while high packet losses lead to the bandwidth decrease.

The algorithm presented in this paper is based on the feedback information delivered in the receiver reports of the Real-Time Transport Protocol (RTP). This feedback information allows the source to estimate the loss rates experienced by the receivers and to adjust its bandwidth accordingly. Measurements with a controlled video application have been performed on two different network technologies, the Internet and our in-house ATM network. These measurements showed that parameters suitable for both network technologies can be found.

The remainder of this paper is structured as follows. The following section gives a short introduction to RTP. In section 3 we describe our feedback control mechanism. In section 4 we describe our experiments and discuss the results. Related work is presented in section 5. Section 6 summarizes our results.

2 The Real-Time Transport Protocol RTP

RTP has been designed within the Internet Engineering Task Force (IETF) [2, 3]. Note that the moniker "transport protocol" could be misleading, as it is currently mostly used together with UDP, also designated as a transport protocol. The name emphasizes, however, that RTP is an end-to-end protocol. To avoid misunderstandings, it may help to clear up some of the things that RTP does *not* attempt to do. RTP has no notion of a connection; it may operate over either connection-oriented or connectionless lower-layer protocols. It has no dependencies on particular address formats and only requires that framing and segmentation is taken care of by lower layers. RTP offers no reliability mechanisms. It is typically implemented as part of the application and not of the operating system kernel. RTP consists of two parts, a data part and a control part. Continuous media data like audio and video is carried in RTP data packets. The functionality of the control packets is described below. If RTP packets are carried in UDP datagrams, data and control packets use two consecutive ports, with the data port always being the lower, even numbered one. If other protocols serve underneath RTP (e.g., RTP directly over ATM AAL5), it is possible to carry both in a single lower-layer protocol data unit, with control followed by data.

2.1 RTP Data Packets

RTP data packets consist of a 12-byte header followed by the payload, e.g., a video frame or a sequence of audio samples. The payload may be wrapped again into an encoding-specific layer. The header contains the following information:

[1] There may be data applications that require minimum usable throughput.

Payload type: A one-byte payload type identifies the kind of payload contained in the packet, for example JPEG video or GSM audio.

Timestamp: A 32-bit timestamp describes the generation instant of the data contained in the packet. The timestamp frequency depends on the payload type.

Sequence number: A 16-bit packet sequence number allows loss detection and sequencing within a series of packets with the same timestamp.

Marker bit: The interpretation of a marker bit depends on the payload type. For video, it marks the end of a frame, for audio the beginning of a talkspurt.

Synchronization source (SSRC) identifier: A randomly generated 32-bit scalar that uniquely identifies the source within a session.

Some additional bit fields are not described here in the interest of brevity.

2.2 RTP Control Functionality

RTP offers a control protocol called RTCP that supports the protocol functionality. An RTCP message consists of a number of "stackable" packets, each with its own type code and length indication. Their format is fairly similar to data packets; in particular, the type indication is at the same location. RTCP packets are multicast periodically to the same multicast group as data packets. Thus, they also serve as a liveness indicator of session members, even in the absence of transmitting media data. The functionality of RTCP is described briefly below:

- **QoS monitoring and congestion control**. RTCP packets contain the necessary information for quality-of-service monitoring. Since they are multicast, all session members can survey how the other participants are faring.
  Applications that have recently sent audio or video data generate a sender report. It contains information useful for intermedia synchronization (see below) as well as cumulative counters for packets and bytes sent. These allow receivers to estimate the actual data rate.
  Session members issue receiver reports for all video or audio sources they have heard from recently. They contain information on the highest sequence number received, the number of packets lost, a measure of the interarrival jitter and timestamps needed to compute an estimate of the round-trip delay between sender and the receiver issuing the report.

- **Intermedia synchronization**. The RTCP sender reports contain an indication of real time (wallclock time) and a corresponding RTP timestamp. These two values allow the synchronization of different media, for example, lip-syncing of audio and video.

- **Identification**. RTP data packets identify their origin only through a randomly generated 32-bit identifier. For conferencing application, a bit more context is often desirable. RTCP messages contain an SDES (source description) packet, in turn containing a number of pieces of information, usually textual. One such piece of information is the so-called canonical name, a globally unique identifier of the session participant. Other possible SDES items include the user's name, email address, telephone number, application information and alert messages.

- **Session size estimation and scaling**. RTCP packets are sent periodically by each session member. The desire for up-to-date control information has to be balanced against the desire to limit control traffic to a small percentage of data traffic even with sessions consisting of several hundred members. The control traffic load is scaled with the data traffic load so that it makes up a certain percentage of the nominal data rate (5%).

3 End-to-End Application Control Mechanism

Our feedback control schemes uses RTP as described in the previous section. The receiving end applications deliver receiver reports to the source. These reports include information that enables the calculation of packet losses and packet delay jitter.
There are two reasons for packet loss: packets get lost due to buffer overflow or due to bit errors. The probability of bit errors is very low on most networks, therefore we assume that loss is induced by congestion rather than by bit errors, just as it is done within TCP [4]. Buffer overflow can happen on a congested link or at the network interface of the workstation. To avoid losses at the network interface we used the workstations for the multimedia application exclusively. Our QoS feedback control scheme is depicted in figure 1.

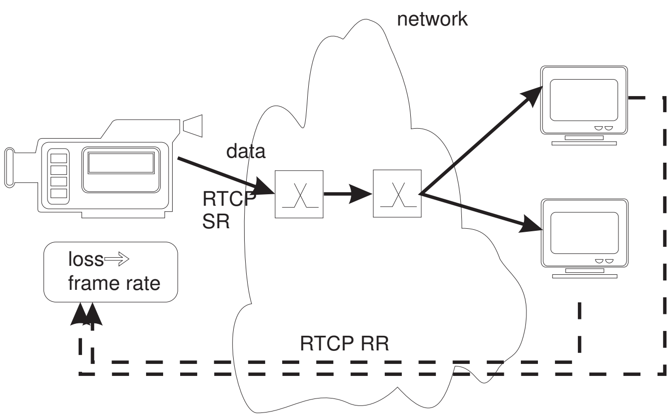

Figure 1: The End-to-End Application Control Mechanism

On receiving an RTCP receiver report (RR), a video source performs the following steps :

- **RTCP analysis**. The receiver reports of all receivers are analyzed and statistics of packet loss, packet delay jitter and roundtrip time are computed.
- **Network state estimation**. The actual network congestion state seen by every receiver is determined as unloaded, loaded or congested. This is used to decide whether to increase, hold or decrease the bandwidth requirements of the sender.
- **Bandwidth adjustment**. The bandwidth of the multimedia application is adjusted according to the decision of the network state analysis. The user can set the range of adjustable bandwidth, i.e., specify the minimum and maximum bandwidth.

All steps except the adjustment are independent of the characteristics of the multimedia application. The steps are discussed in the following sections.
Besides loss, delay jitter, also reported by RTCP, might be used to determine a forthcoming congestion. Due to the related QoS degradation it is desirable to detect congestion before packet loss occurs. In this case the delay will increase due to increased buffering within the network elements. A quick reduction of the bandwidth might completely avoid packet loss. The use of jitter as congestion indicator is only touched in this paper and will be subject to future research.

3.1 RTCP Analysis

The source provides a record for each receiver containing the most recent receiver reports, the information of the session description packets, the loss rate and packet delay jitter. In the current scheme only the loss rate is used as congestion indicator. We use a filter to smooth the statistics and to avoid QoS oscillations. The smoothed loss rate λ is computed by the low-pass filter $\lambda \leftarrow (1 - \alpha)\lambda + \alpha b$, where b is the new value and $0 \leq \alpha \leq 1$. Increasing α increases the influence of the new value while decreasing α results in a higher influence of the previous values.

3.2 Network State Estimation based on Loss

As a measure of congestion we use the smoothed value of the loss rates observed by the receivers. The network congestion state is determined and used to make the decision of increasing, holding or decreasing the bandwidth. We have chosen a linear regulator with dead zone.

3.2.1 Receiver Classification

We define two thresholds to determine the network state seen by each receiver as UNLOADED, LOADED or CONGESTED according to the distinction in figure 2:

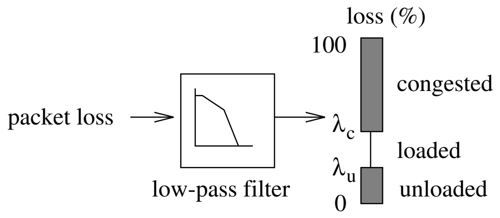

Figure 2: Receiver Classification

The upper threshold λ_c should be chosen so that the data transmission may suffer from the losses but is still acceptable. The deadzone must be large enough, i.e. we have to set λ_u low enough, to avoid QoS oscillations. The choice is more or less arbitrary and has to be justified by experimental results. Suitable values for our video source are $\lambda_u = 2\%$ and $\lambda_c = 4\%$ as the results of section 4 indicate.

3.2.2 Making an Adjustment Decision

In the unicast case the network congestion state can be directly mapped to decrease, hold and increase, respectively. In the case of a point-to-multipoint connection a large number of receivers, possibly spread all over the world, are receiving multicast real-time data and are sending back receiver reports. Should we decrease the bandwidth of a video session only because of one link on the other end of the world suffers from high packet loss?

To solve this problem we examine the following two algorithms. The first one looks for the receiver with the highest average loss rate. Based on the classification of this receiver the bandwidth will be adjusted. In this case any congestion will be avoided. The disadvantage is that a receiver which is connected via a low bandwidth link forces the sender to provide low quality also to the other receivers.

This algorithm is a special case of the second more general algorithm. The second algorithm counts the network congestion states as seen by the receivers and derives the adjustment decision from the proportion of unloaded, loaded and congested receivers. First all receivers are classified as described above. We get the number of receivers in the unloaded state n_u, in the loaded state n_l, and in the congested state n_c as well as the total number of receivers n. Then the decision is made according to the algorithm in figure 3.2.2.

if	$\frac{n_c}{n} \geq N_d$	then	$d \leftarrow$DECREASE
else if	$\frac{n_l}{n} \geq N_h$	then	$d \leftarrow$HOLD
else			$d \leftarrow$ INCREASE

Figure 3: Adjustment Decision

As a starting point we have chosen $N_d = 0.1$ and $N_h = 0.1$. This algorithm then gives the decrease decision priority and increases only if at least 80% of the receivers network states are unloaded.

3.3 Bandwidth Adjustment

Another important aspect is how the multimedia application reacts to the decisions of the network state estimate. This reaction depends on the characteristics of the application. In the case of congestion the application should rapidly reduce its bandwidth. Therefore we use a multiplicative decrease μ if we get a DECREASE message and an additive increase ν if the decision is INCREASE. If the decision is HOLD no changes take place. This well-known scheme is used in TCP [4] and ATM rate-control of ABR traffic [5]. It was also successfully applied in the context of application control by Bolot et al. [6]. As proposed therein we make sure that the bandwidth is always larger than a minimum bandwidth b_{min} to guarantee a minimal quality of service at the receivers. A maximum bandwidth b_{max} can also be set. The bandwidth adjustment algorithm is depicted in figure 4.

if	$d =$DECREASE	then	$b_a \leftarrow \max\{b_r * \mu, b_{min}\}$
else if	$d =$ INCREASE	then	$b_a \leftarrow \min\{b_r + \nu, b_{max}\}$

Figure 4: Bandwidth Adjustment

We distinguish between the reported bandwidth b_r and the allowed bandwidth b_a with $b_a, b_r \in [b_{min}, b_{max}]$. The former is the actual bandwidth as reported in the most recent RTCP sender report, the latter is the allowed bandwidth that can be used by the multimedia application.

4 Experiments

In order to test our control algorithm we ran several experiments on two different network infrastructures, the Internet and our in-house ATM network. As controlled application we took the `vic`[2] video conferencing system that is widely used for MBONE transmissions of conferences and workshops. `vic` has already an RTP implementation and allows to set the maximum bandwith of the transmitted video stream. This parameter is manipulated by our control module. The chosen video encoding was nv [7].
The measurement scenario was as follows. Two video sources at GMD Fokus were sending two different previously recorded CNN news shows. This allowed us to use the same video sequence for each experiment. We set the minimum and maximum bandwidth to 50 kbit/s and 1000 kbit/s.

[2]`vic` has been developed at the Lawrence Berkeley Laboratory, Berkeley, CA.

4.1 Internet Experiments

For the Internet experiments the receiving host was a workstation at TU-Berlin. GMD Fokus and TU-Berlin are connected by a 2 Mbit/s IP-over X.25 link. The distance between sender and receiver were 5 hops. We made our measurements in the morning between 8 am and 10 am when the link was lightly loaded. As motivation for bandwidth control we performed one experiment without changing the bandwidth of the video source. Figure 6 shows that heavy losses ranging from 20% to 50% percent occur. The rest of the experiments has been performed with bandwidth control.

Because we have a great number of parameters we decided to vary in this first set of experiments only the three parameters α, λ_u and λ_c. The bandwidth adjustment parameters were set to $\mu = 0.875$ and $\nu = 50$. The initial bandwidth of the sources was 50 kbit/s.

With the first experiments we tried to find a suitable value for the filter parameter α. Figures 7, 8 and 9[3] show the bandwidth evolution, loss and smoothed loss of one source over 300 seconds with $\alpha = 0.1$, $\alpha = 0.3$, and $\alpha = 0.5$. The threshold parameter were $\lambda_u = 5\%$ and $\lambda_c = 10\%$. In figure 7 and 9 can be observed that the bandwidth is hardly reduced although heavy losses over 10% occur. An adjustment can only be seen in figure 8 where the overall bandwidth is lower than in the other two experiments. This showed that the right choice of the filter parameter α is crucial. Remember that the control algorithm is triggered by the reception of RTP receiver reports. If α is set too low, e.g., $\alpha = 0.1$, then a new loss value has not enough influence to raise the smoothed loss value λ over the congestion threshold λ_c and no adjustment takes place. The loss value is forgotten immediately. If α is set too high, e.g., $\alpha = 0.5$, then a new value has high influence and raises the smoothed value. But this influence vanishes immediately with the advent of the next values if they indicate no or only few losses. Because $\alpha = 0.3$ seems to work well we made all the following experiments with this value.

The following experiments focus on the impact of the thresholds λ_u and λ_c. Figures 10, 11 and 12 depict the bandwidth evolution and the smoothed loss of the two sources. From the three figures can be seen that the adjustment process works well, i.e., the bandwidth is reduced significantly when the smoothed loss exceeds the threshold λ_c. A threshold $\lambda_u = 1\%$ is too low, i.e., the scheme is then not aggressive enough and available bandwidth is not exploited. Figure 10 shows that the bandwidth increase is very slow although the smoothed loss tends to be low in the first 150 seconds. With $\lambda_u = 2\%$ the increase after losses is much faster. The adjustment works also well with $\lambda_u = 5\%$ and $\lambda_c = 10\%$; it can be observed that the loss varies most of the time between 5% and 10% but a look at the received video stream reveals that an ongoing loss in this range is hardly acceptable[4]. In these three experiments no heavy QoS oscillations can be noticed, the dead-zone $[\lambda_u, \lambda_c]$ is big enough. Another interesting result is that the bandwidth is shared nearly equally between the two sources. This is evident because the shapes of the two smoothed loss curves are nearly identically. We will see later that this need not be the case.

We discussed in section 3 the idea of using packet delay jitter as a predictor for packet loss. In none of our experiments a significant increase in jitter before packet loss could be observed (this is why we did not depict jitter in the experiment graphs). It remains to be studied if jitter works as a loss predictor on longer distances when network packets have to traverse more hops than in our case.

4.2 ATM Experiments

We tested our control algorithm also on our in-house ATM network in order to see how it performs on a different network technology. Figure 5 shows the measurement scenario. The Sun workstations have FORE SBA 200 ATM interfaces and are connected to the switches with TAXI 100 Mbit/s interfaces. `vic` was run

[3] The measurements were not performed at the same time, therefore we have to deal with different background load. Due to the control of the sources it is not possible to test different parameters in one experiment.

[4] We note that the impact of loss on the video quality depends strongly on the used video encoding.

via UDP/IP on top of AAL5. The HP Broadband Series Test System E1401A (HP Analyser) connected with a TAXI 140 MBit/s, generates the background load that is necessary to lead the link interconnecting the two FORE ASX 200 switches in a congested state. This link has a capacity of 140 Mbit/s. Cells are lost when the output buffer of the first switch overflows. This buffer has a capacity of 256 cells. As a first step we chose a Poisson distributed cell traffic with a mean load of 131 Mbit/s as traffic model. This is quite a simple traffic scenario, future experiments will be performed with more elaborate models where the regulated traffic has a larger percentage of the overall traffic.

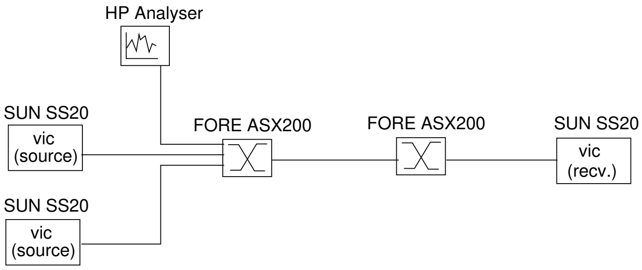

Figure 5: The Measurement Scenario on the GMD Fokus ATM Network

Figure 13, 14 and 15 depict the ATM experiments with the parameters $\lambda_u = 2\%$, $\lambda_c = 4\%$ and $\alpha = 0.3$. These three figures show that the smoothed loss remains most of the time within the given deadzone $[\lambda_u, \lambda_c]$. In principle this is correct but shows a weakness of the chosen parameters. In figure 13 it can be seen that the bandwidth is equally shared between the two sources if they have the same initial bandwidth. This is not the case in figure 14. There the sources keep their initial bandwidth since only few adjustment takes place. Using very low values of λ_u and λ_c might solve this problem.

A weakness of the algorithm is revealed in figure 15. We note that this figure shows only one source. Remember that the algorithm takes the actual bandwidth as reported in the RTP sender reports as a base for increase or decrease. Thus the actual bandwidth depends on the contents of the video stream. The steep flank occured due to a low-bandwidth video scene showing a black screen over several seconds. The allowed bandwidth was adjusted accordingly. After that sequence it took approximately 100 seconds to increase the bandwidth again. It seems to be a better choice to compute the allowed bandwidth independently of the actual bandwidth.

5 Related Work

A scalable feedback control mechanism for video sources has also been proposed in [6]. This mechanism is also end-to-end and defines network states according to feedback information from the receivers. This work differs from our approach in the sense that a probabilistic polling mechanism with increasing search scope and a randomly delayed reply scheme is used to supply the source with feedback information. With RTP the receiver reports are multicast periodically so that an explicit probing mechanism is not required. Another feedback controller is presented in [8]. However, this controller requires that the switches send their buffer occupancies and service rates back to the source. It was not designed to scale for multicast distributions.

The control of the application by a network manager that monitors the network elements via standardized management infomation bases is presented in [9]. The network manager notifies applications of network congestion based on feedback received from switches and routers. This approach has serious scaling problems.

6 Conclusions and Future Work

In this paper we propose to use a feedback mechanism based on RTP to control the bandwidth of real-time multimedia applications according to network load. This feedback controller has been implemented to control the bandwidth of a video conferencing system. Like TCP it is based on packet loss. We performed several experiments with this controlled application. These experiments supplied us with a set of appropriate parameters to tune the algorithm and proved that it can be sucessfully applied to the Internet and to the ATM environment. Packet loss can be held within certain bounds. Bandwidth is shared equally among the sources as long as an adjustment takes place. If the losses stay within the deadzone of the regulator, no adjustment occurs and the sources maintain their initial bandwidth. This can be solved by setting the congestion threshold much lower as in the described experiments or by using another value for the multiplicative decrease, e.g., to cut bandwidth to half. However, a quicker adjustment with a smaller deadzone must be balanced against QoS oscillations. In the Internet environment loss is not an exception but occurs rather often therefore the adjustment algorithm is invoked frequently and the bandwidth is shared equally among the video sources in our experiments. Thus it is necessary to choose the parameters according to the network technology.

We investigated the use of the jitter value reported by the RTP receiver reports. It was expected that the jitter increases just before losses occur due to increased buffering at the switches. In our experiments no significant change of the jitter value was observed. The main reason might be that our video data traversed only a small number of hops. Experiments over larger distances are called for.

We also addressed the problem of scalability in multicast distributions. Two solutions were offered, the first one is to adjust bandwidth according to the reports of the worst-positioned receiving site, the second one is to allow only a certain fraction of the receivers to be in a congested state. Both solutions have their limitations. With the second solution some of the receivers will have a low video quality due to high packet loss on their links. With the first solution the bulk of participants receive low-quality video. It is not clear which solution should be chosen in practice. A more powerful approach are video gateways or layered coding schemes as proposed in [10] to solve this problem. However, this affects both video encoding and switch design, issues we wanted to avoid in our approach to make it applicable in actual network environments.

In our ATM experiments we used a Poisson distributed cell traffic as background traffic which covered 93.4% of the bandwidth of the bottleneck link. We plan to reduce this background load and set up additional workstations that create TCP traffic to increase the fraction of controlled traffic on the link. We expect interesting results of how TCP traffic interacts with the controlled traffic of our video sources.

References

[1] M. Grossglauser, S. Keshav, and D. Tse, "The case against variable bit rate service," in *Proc. 5th Intl. Workshop on Network and Operating System Support for Digital Audio and Video*, (Durham, New Hampshire), pp. 307–310, Apr. 1995.

[2] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, "RTP: A transport protocol for real-time applications." Internet draft (work-in-progress) *draft-ietf-avt-rtp-*.txt*, Mar. 1995.

[3] H. Schulzrinne, "Internet services: from electronic mail to real-time multimedia," in *Proc. of KIVS (Kommunikation in Verteilten Systemen)* (K. Franke, U. Hübner, and W. Kalfa, eds.), Informatik aktuell, (Chemnitz, Germany), pp. 21–34, Gesellschaft für Informatik, Springer Verlag, Feb. 1995.

[4] V. Jacobson, "Congestion avoidance and control," *ACM Computer Communication Review*, vol. 18, pp. 314–329, Aug. 1988. Proceedings of the Sigcomm '88 Symposium in Stanford, CA, August, 1988.

[5] S. Sathaye, "DRAFT ATM Forum traffic management specification version 4.0," ATM Forum 95-0013, Dec. 1994.

[6] J.-C. Bolot, T. Turletti, and I. Wakeman, "Scalable feedback control for multicast video distribution in the internet," in *SIGCOMM Symposium on Communications Architectures and Protocols*, (London, England), pp. 58–67, ACM, Aug. 1994.

[7] S. McCanne and V. Jacobson, "The vic video conferencing tool." Manual page, Nov. 1994.

[8] H. Kanakia, P. Mishra, and A. Reibman, "An adaptive congestion control scheme for real-time packet video transport," in *SIGCOMM Symposium on Communications Architectures and Protocols*, (San Francisco, California), pp. 20–31, ACM/IEEE, Sept. 1993.

[9] I. Busse, D. Cochrane, B. Deffner, P. Demestichas, B. Evans, W. Grupp, N. Karatzas, K. Kassapakis, P. Legand, Z. Lioupas, Y. Manolessos, K. Nagel, T. Pecquet, B. Rabere, G. Seehase, G. Stamoulis, M. Scholz, H. Schulzrinne, and S. Vaisenberg, "Quality module specification (D7)," Deliverable R2116/ISR/WP2/DS/S/007/b1, RACE Project 2116 (TOMQAT), Dec. 1994.

[10] T. Turletti and J.-C. Bolot, "Issues with multicast video distribution in heterogeneous packet networks," in *Proc. 6th International Workshop on Packet Video, Portland, Orgegon*, 1994.

[11] S. Deering, "Reservations or no reservations." INFOCOM '95 Panel Session, Apr. 1995.

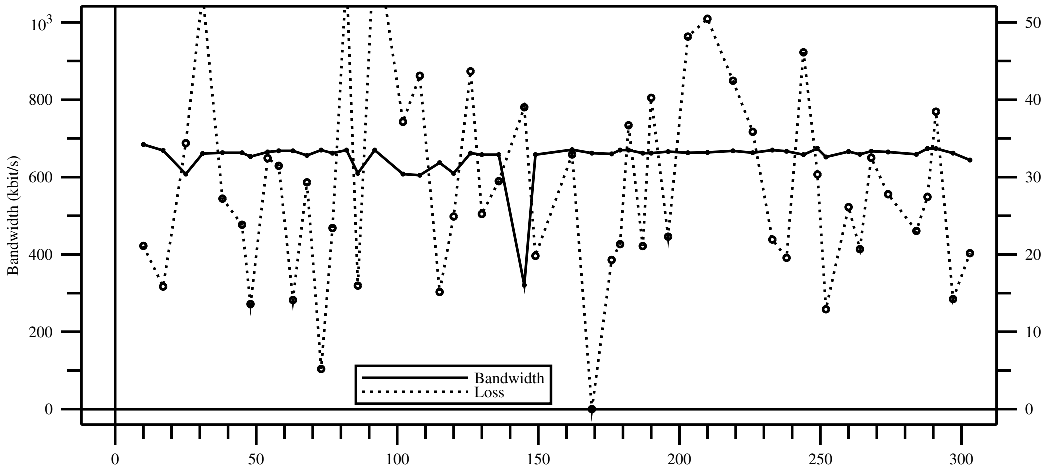

Figure 6: Internet experiment with constant bandwidth of 746 kbit/s

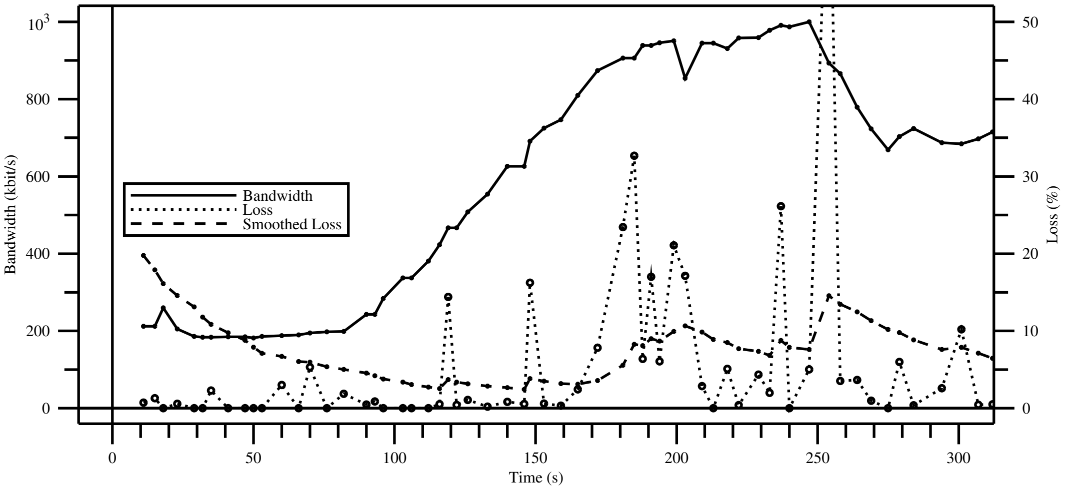

Figure 7: Internet experiment with $\lambda_u = 5, \lambda_c = 10$ and $\alpha = 0.1$

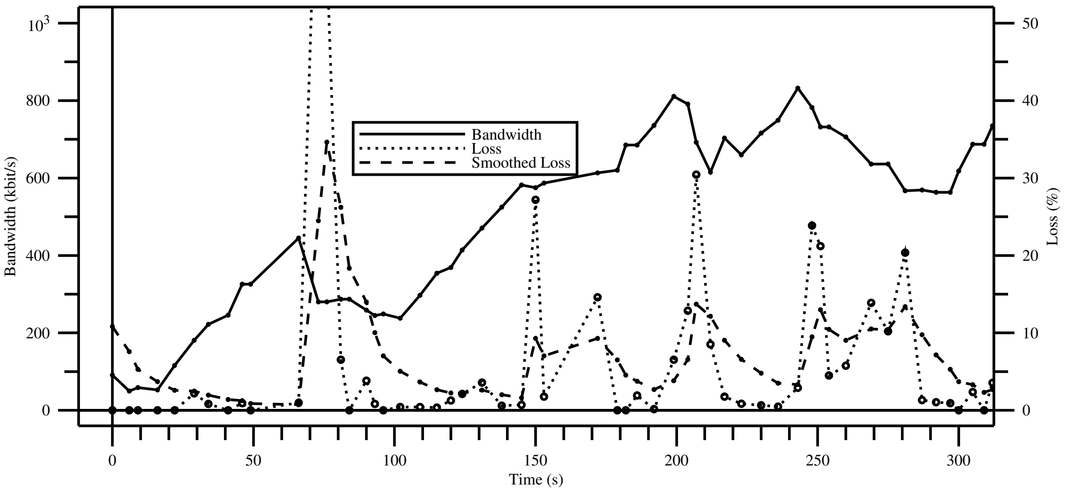

Figure 8: Internet experiment with $\lambda_u = 5, \lambda_c = 10$ and $\alpha = 0.3$

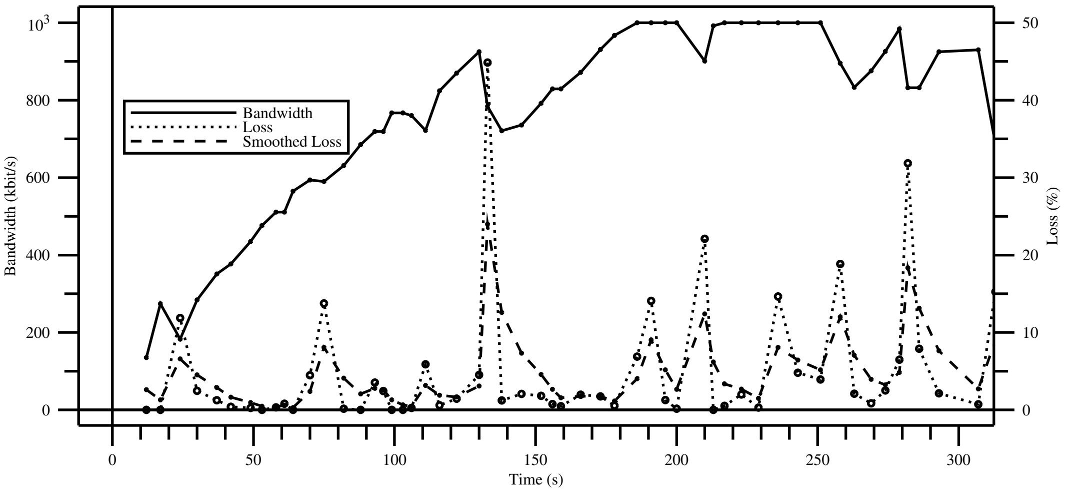

Figure 9: Internet experiment with $\lambda_u = 5, \lambda_c = 10$ and $\alpha = 0.5$

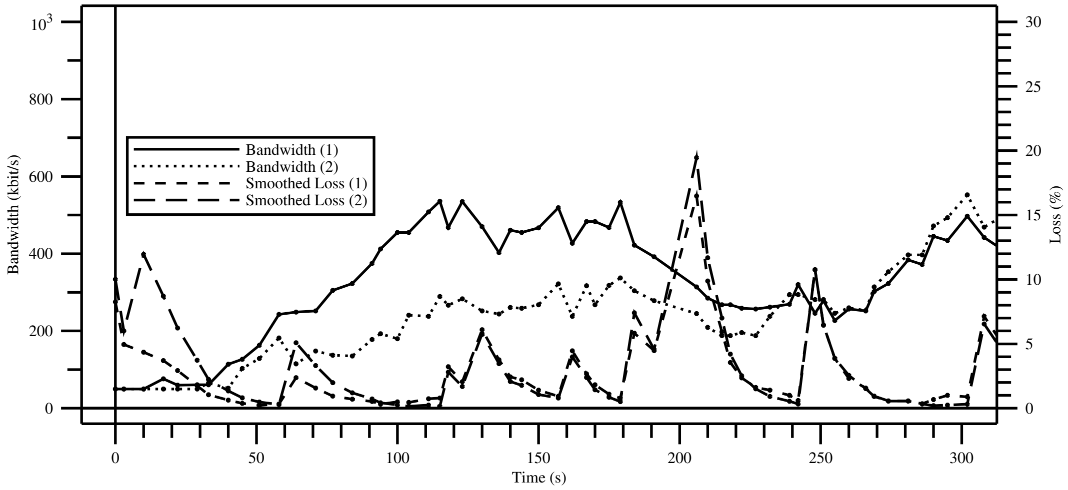

Figure 10: Internet experiment with $\lambda_u = 1, \lambda_c = 3$ and $\alpha = 0.3$

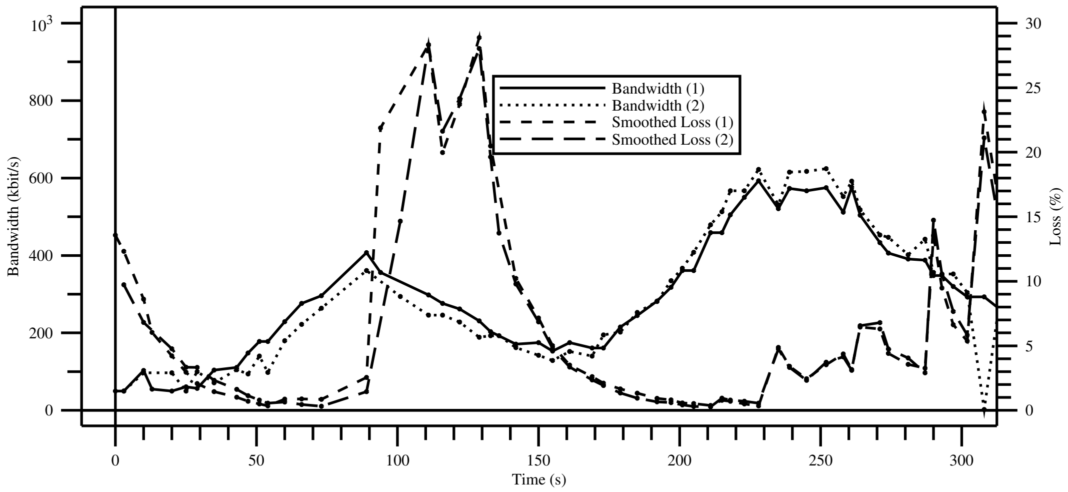

Figure 11: Internet experiment with $\lambda_u = 2, \lambda_c = 4$ and $\alpha = 0.3$

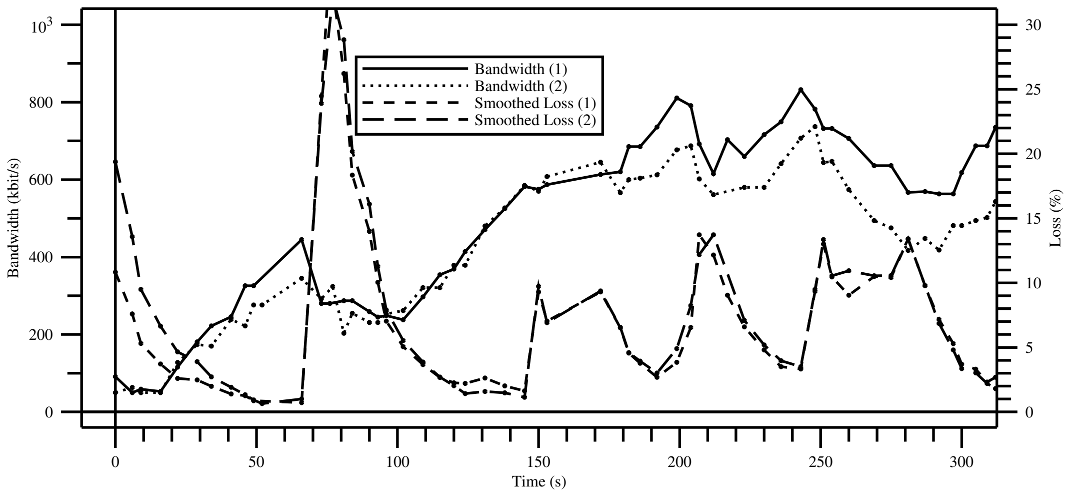

Figure 12: Internet experiment with $\lambda_u = 5, \lambda_c = 10$ and $\alpha = 0.3$

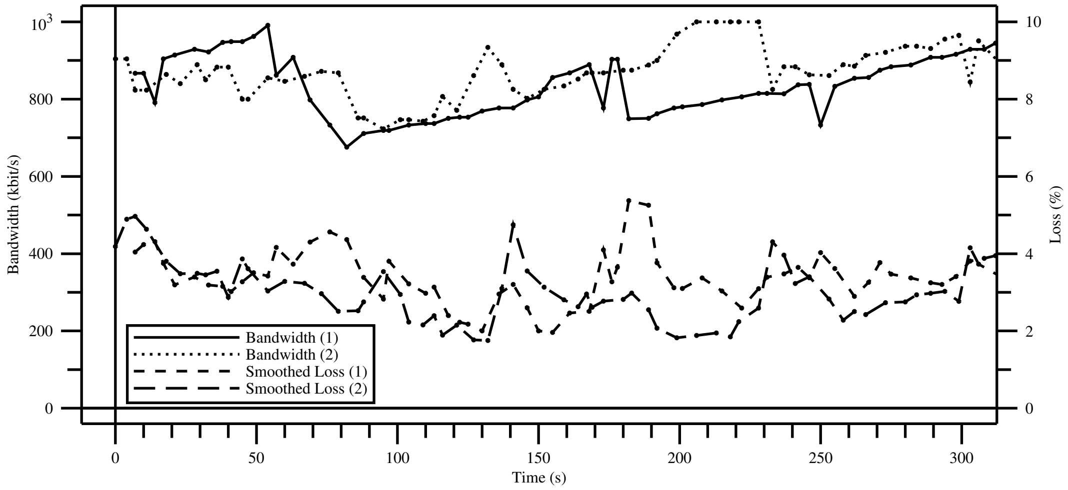

Figure 13: ATM experiment with $\lambda_u = 2, \lambda_c = 4$ and $\alpha = 0.3$

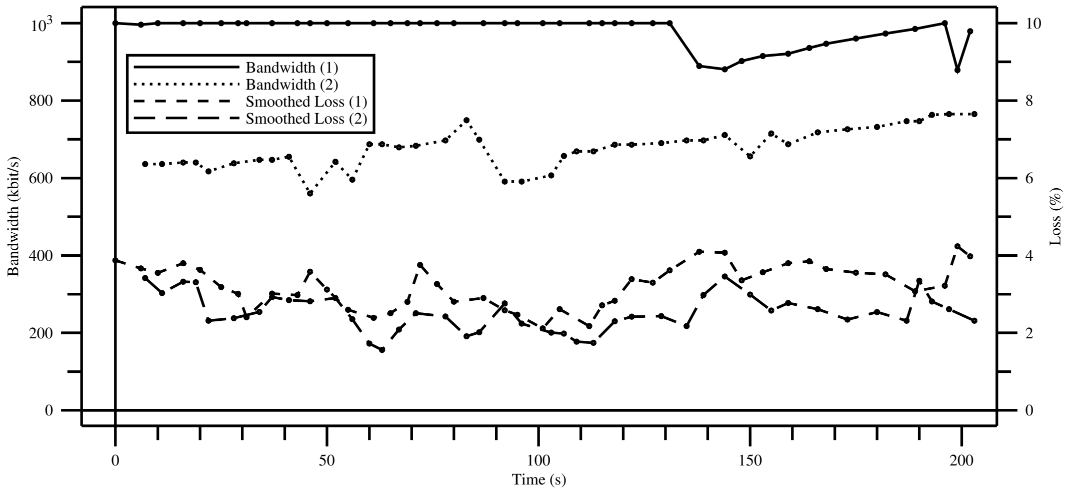

Figure 14: ATM experiment with $\lambda_u = 2, \lambda_c = 4$ and $\alpha = 0.3$

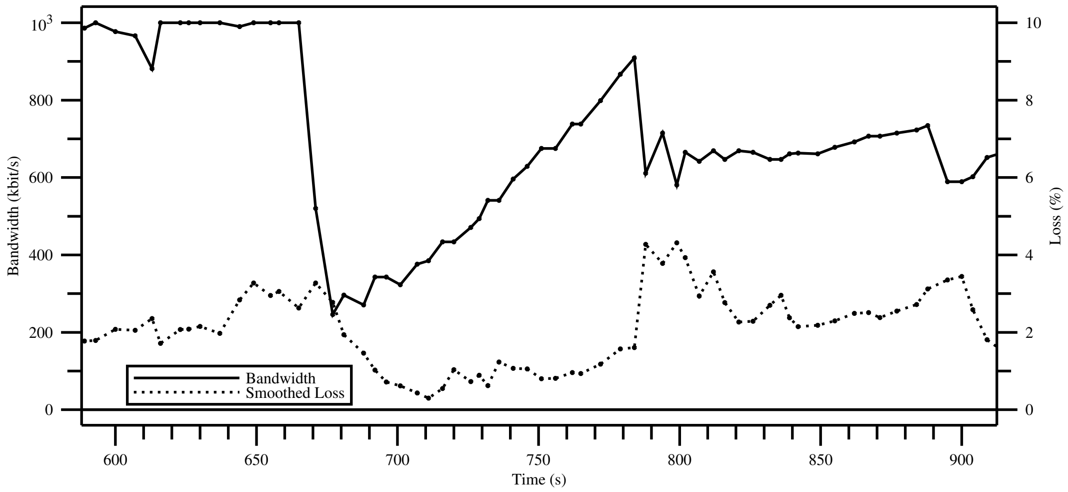

Figure 15: ATM experiment with $\lambda_u = 2, \lambda_c = 4$ and $\alpha = 0.3$