

# The Direct Adjustment Algorithm: A TCP-Friendly Adaptation Scheme

Dorgham Sisalem<sup>1</sup> and Henning Schulzrinne<sup>2</sup>

<sup>1</sup> GMD FOKUS, Berlin, Germany ,  
sisalem@fokus.gmd.de

<sup>2</sup> Columbia University, New York, USA  
schulzrinne@cs.columbia.edu

**Abstract.** Many distributed multimedia applications have the ability to adapt to fluctuations in the network conditions. By adjusting temporal and spatial quality to available bandwidth, or manipulating the playout time of continuous media in response to variations in delay, multimedia flows can keep an acceptable QoS level at the end systems. In this study, we present a scheme for adapting the transmission rate of multimedia applications to the congestion level of the network. The scheme called the direct adjustment algorithm (DAA), is based on the TCP congestion control mechanisms and relies on the end-to-end Real Time transport Protocol (RTP) for feedback information. Our investigations of the DAA scheme suggest that simply relying on the the TCP-throughput model might result under certain circumstances in large oscillations and low throughput. However, DAA achieves, in general, high network utilization network and low losses. Also, the scheme is shown to be fair towards competing TCP traffic. However, with no support from the network, long distance connections receive less than their fair share.

## 1 Introduction

While congestion controlled TCP connections carrying time insensitive FTP or WWW traffic still constitute the major share of the Internet traffic today [1], recently proposed real-time multimedia services such as IP-telephony and group communication will be based on the UDP protocol. While UDP does not offer any reliability or congestion control mechanisms, it has the advantage of not introducing additional delays to the carried data due to retransmissions as is the case with TCP. Additionally, as UDP does not require the receivers to send acknowledgments for received data, UDP is well suited for multicast communication. However, deploying non-congestion controlled UDP in the Internet on a large scale might result in extreme unfairness towards competing TCP connections as TCP senders react to congestion situations by reducing their bandwidth consumption and UDP senders do not. Therefore, UDP flows need to be enhanced with control mechanisms that not only aim at avoiding network overload but are also fair towards competing TCP connections, i.e. be *TCP-friendly*. TCP-friendliness indicates here, that if a TCP connection and an adaptive flow with

similar transmission behaviors have similar round trip delays and losses both connections should receive similar bandwidth shares. As an oscillative perceived QoS is rather annoying to the user, multimedia flows require stable bandwidth shares that do not change on the scale of a round trip time as is the case of TCP connections. It is, thus, expected that a TCP-friendly flow would acquire the same bandwidth share as a TCP connection only averaged over time intervals of several seconds or even over the entire life time of the flow and not at every time point [2].

In this paper, we propose an end-to-end rate adaptation scheme called the direct adjustment algorithm (DAA) for adjusting the transmission rate of multimedia applications to the congestion level of the network. DAA is based on a combination of two approaches described in the literature, namely: additive increase/multiplicative decrease (AIMD) schemes proposed in [3, 4] and an enhancement of the TCP-throughput model described in [5].

In Sec. 2, we present a general overlook on some of the TCP-friendly schemes currently proposed in the literature.

Sections 3 presents an approach for adapting the transmission rate of end systems to the network congestion state using RTP based on the AIMD approach and discusses unfairness of such schemes towards competing TCP connections. The direct adjustment algorithm is presented in Sec. 4. The performance of the scheme in terms of bandwidth utilization as well as the behavior of TCP connections traversing the same congested links is then investigated in Sec. 5 and Sec. 6.

## 2 Background and Related Work

Recently, there has been several proposals for TCP-friendly adaptation schemes that either use control mechanisms similar to those of TCP or base the adaptation behavior on an analytical model of TCP.

Rejaie et al. present in [6] an adaptation scheme called the rate adaptation protocol (RAP). Just as with TCP, sent packets are acknowledged by the receivers with losses indicated either by gaps in the sequence numbers of the acknowledged packets or timeouts. The sender estimates the round trip delay using the acknowledgment packets. If no losses were detected, the sender periodically increases its transmission rate additively as a function of the estimated round trip delay. After detecting a loss the rate is multiplicatively reduced by half in a similar manner to TCP.

Jacobs [7] presents a scheme called the Internet-friendly protocol that uses the congestion control mechanisms of TCP, however, without retransmitting lost packets. In this scheme, the sender maintains a transmission window that is advanced based on the acknowledgments of the receiver which sends an acknowledgment packet for each received data packet. Based on the size of the window the sender estimates then the appropriate transmission rate.

Padhye et al. [5] present an analytical model for the average bandwidth share of a TCP connection ( $r_{\text{TCP}}$ )

$$r_{\text{TCP}} = \frac{M}{t_{\text{RTT}} \sqrt{\frac{2Dl}{3}} + t_{\text{out}} \min\left(1, 3\sqrt{\frac{3Dl}{8}}\right) l(1 + 32l^2)} \quad (1)$$

with  $M$  as the packet size,  $l$  as the loss fraction,  $t_{\text{out}}$  as the TCP retransmission timeout value,  $t_{\text{RTT}}$  as the round trip delay and  $D$  as the number of acknowledged TCP packets by each acknowledgment packet.

Using this model Padhye et al. [8] present a scheme in which the sender estimates the round trip delay and losses based on the receiver's acknowledgments. In case of losses, the sender restricts its transmission rate to the equivalent TCP rate calculated using Eqn. 1 otherwise the rate is doubled.

Additionally, various schemes have been proposed for the case of multicast communication such as [9–11] that aim at using a TCP-friendly bandwidth share on all links traversed by the multicast stream.

### 3 Additive Increase and Multiplicative Decrease Adaptation Using RTP

When designing an adaptive control scheme, following goals need to be considered:

- to operate with a low packet loss ratio
- achieve high overall bandwidth utilization
- fairly distribute bandwidth between competing connections

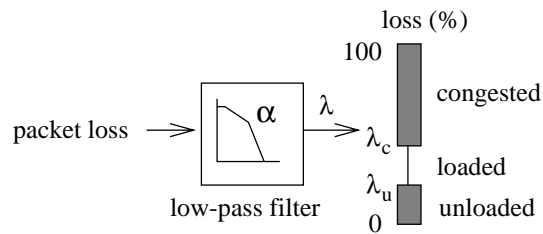
Whereas fairness in this context does not only mean equal distribution of bandwidth among the adaptive connections, but being friendly to competing TCP traffic as well. Various adaptation algorithms proposed in the literature [3, 4, 12] were shown to be efficient in terms of the first two goals of low losses and high utilization but neglected the fairness issue. As those schemes do not consider the bandwidth share of TCP traffic traversing the same congested links, such algorithms might lead to the starvation of competing TCP connections.

With most of the adaptation schemes presented in the literature [6, 5] the sender adapts its transmission behavior based on feedback messages from the receiver sent in short intervals in the range of one or a few round trip delays. This is particularly important for the case of reliable transport where the sender needs to retransmit lost packets. Additionally, with frequent feedback messages the sender can obtain up-to-date information about the round trip delay and, hence, use an increase in the round trip delay as an early indication of possible congestion.

On the contrary, in this paper we investigate adaptation schemes that use the real time transport protocol (RTP) [13] for exchanging feedback information about the round trip time and the losses at the receiver. As RTP is currently

being proposed as an application-level protocol for multimedia services over the Internet, using RTP would ease the introduction of adaptation schemes in the context of such services. RTP defines a data and a control part. For the data part RTP specifies an additional header to be added to the data stream to identify the sender and type of data. With the control part called RTCP, each member of a communication session periodically sends control reports to all other members containing information about sent and received data. However, with RTP, the interval between sending two RTCP messages is usually around five seconds. The in-frequency of the RTCP feedback messages dictates that an RTP sender can not benefit fast enough from rapid changes in the network conditions. Thus, the goal of RTCP-based adaptation is to adjust the sender's transmission rate to the average available bandwidth and not react to rapid changes in buffer lengths of the routers for example. This might be actually more appropriate in some cases than rapidly changing the transmission rate at a high frequency.

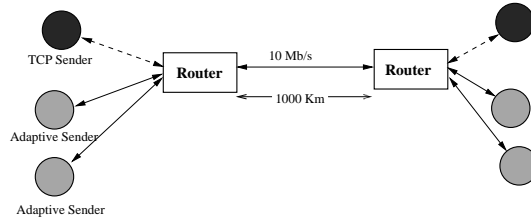
As an example for this behavior, we tested the approach described in [4]. This scheme has great resemblance to the schemes described in [3, 12, 14] and is based on the same AIMD approach. With this approach the sender reduces its transmission rate by a multiplicative decrease factor after receiving feedback from the receiver indicating losses above a certain threshold called the upper loss threshold. With losses below a second threshold called the lower loss threshold the sender can increase its transmission rate additively. For the case that the feedback information indicates losses in between the two thresholds the sender can maintain its current transmission level, see Fig. 3. Reducing the rate multiplicatively allows for a fairer reaction to congestion. That is, connections utilizing a disproportionately large bandwidth share are forced to reduce their transmission rate by a larger amount.



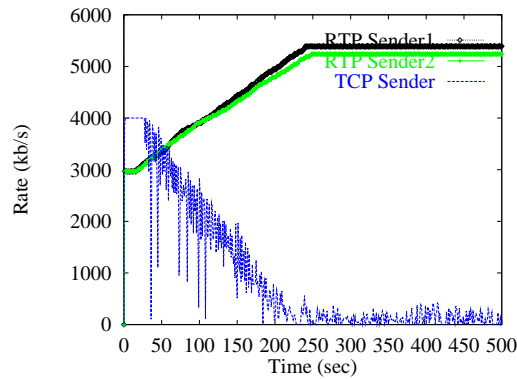
**Fig. 1.** AIMD algorithm

To test the behavior of TCP connections sharing the same congested links with connections using this scheme, we simulated the topology depicted in Fig. 2. One TCP connection is sharing a bottleneck router with two connections using the adaptation approach just described. The connection has a round trip time of 10 msec and a bandwidth of 10 Mb/s. The TCP source is based on Reno-TCP.

That is, the TCP source reduces its transmission window by half whenever loss is indicated. The adaptive connections have a lower loss threshold of 5% and a higher one of 10%. The additive increase factor was set to 50 kb and the multiplicative decrease factor to 0.875. Those values were suggested in [4] to be most appropriate based on measurements. The router is a random early drop (RED) gateway as was proposed by Floyd and Jacobson [15]. A RED gateway detects incipient congestion by computing the average queue size. When the average queue size exceeds a preset minimum threshold the router drops each incoming packet with some probability. Exceeding a second maximum threshold leads to dropping all arriving packets. This approach not only keeps the average queue length low but ensures fairness and avoids synchronization effects. In all of our simulations presented in this paper the minimum drop threshold of the router was set to 0.5 and the maximum one to 0.95 based on results achieved in [2].



**Fig. 2.** Test configuration for the interaction of the adaptive schemes and TCP



**Fig. 3.** Bandwidth distribution with an AIMD adaptation scheme

As Fig. 3 shows, the adaptive connections share the available bandwidth among themselves, leaving less than 5% for the TCP connection. This, however, was to be anticipated. With the acknowledgment scheme of TCP, senders are usually informed about packet losses within a time period much smaller than the minimal time between two RTCP packets, i.e., 5 seconds. With each loss notification, TCP reduces its transmission window by half. Older TCP versions, e.g., Tahoe-TCP reduce it even down to one packet [16]. So, while the adaptive source keeps on sending with the high transmission rate until an RTCP packet with loss indication arrives, the TCP source reduces its transmission window and thereby its rate. That is, the adaptive source can for a longer time period send data with the rate that might have actually caused the congestion. Also, as TCP reduces its transmission rate the congestion level will be decreased, so that the adaptive source will finally have to react to a reduced congestion level. Finally, the adaptive scheme will only start reacting to congestion if the losses are larger than the loss threshold. TCP, on the contrary, reacts to any lost packets. Therefore, in Fig. 3 we can notice that during the first 200 seconds, the TCP connection reduces its rate whereas the adaptive connection actually increases its transmission rate. The adaptive connection only starts reacting to congestion after measuring a loss ratio higher than the loss threshold that was set here to 5%. However, as the loss remains below the 10% upper threshold, the adaptive connections can keep their high rate.

#### 4 The Direct Adjustment Algorithm

The direct adjustment algorithm is based on both the AIMD approach as well as directly using the bandwidth share a TCP connection would be using under the same round trip time, packet size and loss ratio.

During an RTP session the receiver reports in its control packets the percentage of lost data noticed since sending the last control packet. At the sender site, the RTCP packets are processed and depending on the loss values reported within the RTCP packets, the sender can increase or decrease its sending rate. With the reception of each RTCP packet the sender needs to do the following:

- Calculate the round trip time ( $\tau$ ) and determine the propagation delay. The RTCP receiver reports include fields describing the timestamp of the last received report from this sender  $T_{LSR}$  and the time elapsed since receiving this report and sending the corresponding receiver report  $T_{DLSR}$ . Knowing the arrival time ( $T$ ) of the RTCP packet the end system can calculate the round trip time.

$$\tau = T - T_{DLSR} - T_{LSR} \quad (2)$$

- The RTCP receiver reports contain the value of the average packet loss ( $l$ ) measured for this sender in the time between sending two consecutive RTCP packets at the reporting receiver. To avoid reactions to sudden loss peaks in the network the sender determines a smoothed loss ratio  $l_s$

$$l_s = (1 - \sigma) \times l_s + \sigma \times l \quad (3)$$

with  $l$  as the loss value reported in the RTCP packet and  $\sigma$  as a smoothing factor set here to 0.3. This value was used in [4] and various simulations and measurements done in [17] suggested its suitability as well.

- Using  $l_s$  and  $\tau$  the sender calculates  $r_{\text{tcp}}$  as in Eqn. 1.
- For the case of ( $l_s > 0$ ) the sender sets its transmission rate to  $(\min[r_{\text{tcp}}, r_{\text{add}}])$  with

$$r_{\text{add}} = r + A \quad (4)$$

with  $A$  as the additive increase factor and  $r$  as the current transmission rate.

## 5 Performance of the Direct Adjustment Algorithm

As a first performance test of the direct adjustment algorithm we used the topology depicted in Fig. 4. The model consists of 15 connections sharing a bottleneck router. All connections use the direct adjustment algorithm and are persistent sources. That is, they always have data to send with the maximum allowed rate. They all have the same round trip times and are similar in their requirements and characteristics. The routers used RED for buffer management. In all of our simulations, we set the packet size to 1 kbyte which is a typical video packet size. Tab. 1 depicts the average utilization results achieved for different round trip

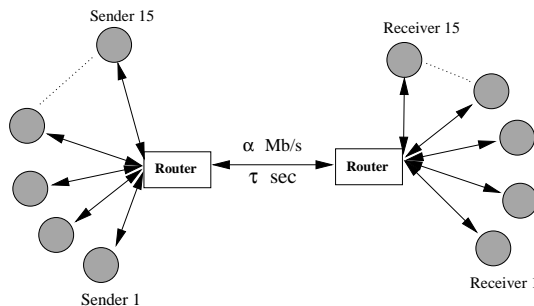
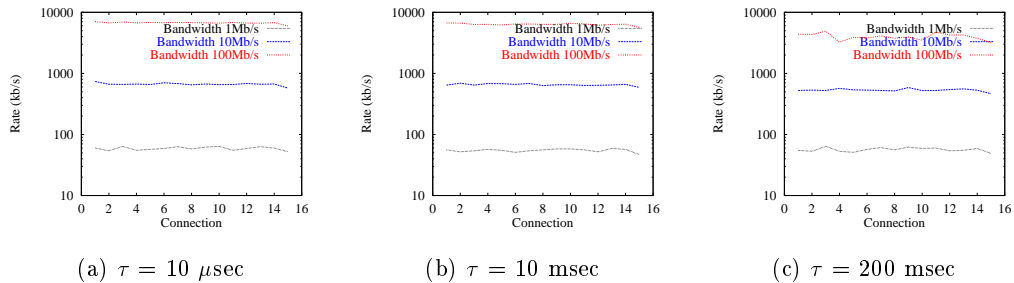


Fig. 4. DAA performance testing topology

times and different link bandwidths. Fig. 6 shows the average bandwidths the single connections achieved for simulation runs of 1000 seconds. Fig. 5 describe the average rates achieved for each flow under different simulation parameters. The similar values of the bandwidth shares indicate the fairness of DAA among competind DAA flows. The results shown in Tab. 1 as well as Fig. 5 reveal that using the direct adjustment algorithm bandwidth utilization between 60% and 99% is possible and that the bandwidth is equally distributed among all connections sharing the same link. Note also in Fig. 6, that while connection number

Propagation Delay( $\tau$ )	$\alpha=1$ Mb/s	$\alpha=10$ Mb/s	$\alpha=100$ Mb/s
5 $\mu$ sec	0.89	0.99	0.99
5 msec	0.82	0.97	0.95
100 msec	0.85	0.8	0.60

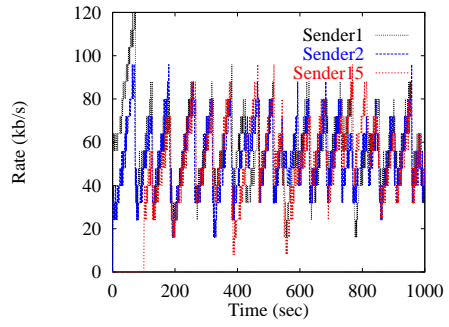
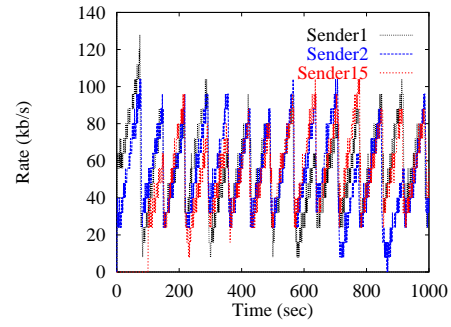
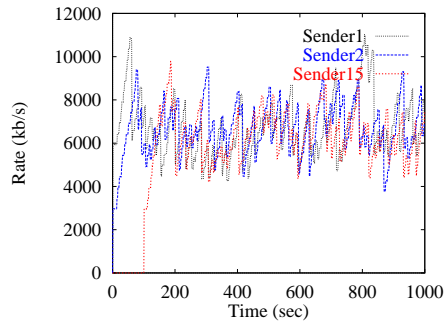
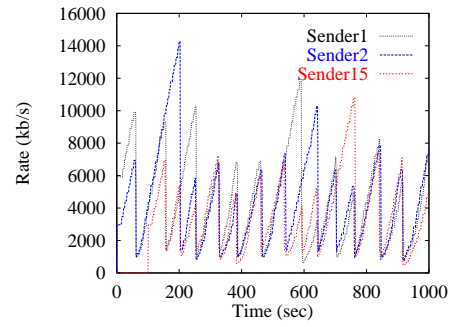
**Table 1.** Link utilization with the direct adjustment algorithm for different propagation delays ( $\tau$ ) and link rates ( $\alpha$ )



**Fig. 5.** Rate of the single connections of DAA under different round trip times ( $\tau$ ) and link bandwidths ( $\alpha$ )

15 starts 100 seconds later than the other connections it soon reaches the same bandwidth level.

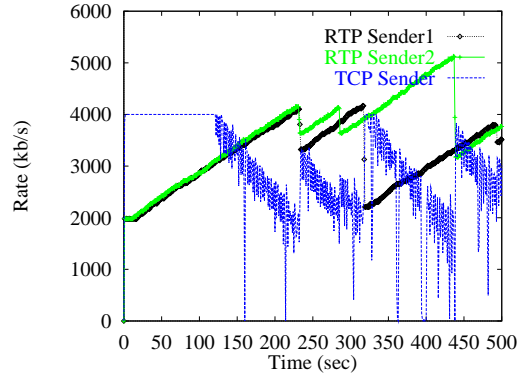
The variance in the achieved utilization results from the oscillatory behavior of the scheme. As Fig. 6 shows, the transmission rate of the senders varies in the range of  $\pm 30\%$  of the average rate. These oscillations result in part from the AIMD approach and in part from the partial incorrectness of the TCP model. AIMD schemes are inherently oscillatory. That is, such systems do not converge to an equilibrium but oscillate around the optimal state [18]. Also, the TCP-model we are using for determining the maximum allowed transmission rate does not consider the bandwidth available on the network or the number of connections sharing a link. Thereby, it might result in throughput values much higher than available on a link, and in other case might underestimate the appropriate bandwidth shares each connection should be using. This is especially evident in Fig. 6(d). Due to the high propagation delay, loss values as low as 0.1% result in a TCP throughput of only 1.5 Mb/s, whereas each connection should have been using a share of 6 Mb/s. So, as the TCP model does not take the available bandwidth into account these oscillations can not be prevented. In addition, note that the adjustment decisions are made here based on the loss and delay values reported in the RTCP packets. The TCP-throughput model is, however, based on considering the loss and delay values measured throughout the lifetime of a connection.

(a)  $\tau=10$  Mb/s,  $\alpha=5$  msec(b)  $\tau=10$  Mb/s,  $\alpha=200$  msec(c)  $\tau=10$  Mb/s,  $\alpha=5$  msec(d)  $\tau=100$  Mb/s,  $\alpha=200$  msec

**Fig. 6.** Temporal behavior of the direct adjustment algorithm under different round trip times ( $\alpha$ ) and link bandwidths ( $\tau$ )

## 6 TCP and the Direct Adjustment Algorithm

When designing the direct adjustment algorithm we aimed not only at achieving high utilization and low losses but also wanted the adaptation to be TCP-friendly.



**Fig. 7.** Bandwidth distribution with the direct adjustment algorithm and TCP

Fig. 7 shows bandwidth distribution achieved with the direct adjustment algorithm using the topology described in Fig. 4. While using the adaptation scheme presented in [4] leads to the starvation of the TCP connections, see Fig. 3, using the direct adjustment algorithm results in a near equal bandwidth distribution: the TCP connection gets around 30% of the available bandwidth and the two adaptive connections each get 35%. This slight unfairness might be explained with the long control intervals of the RTCP protocol. As the adaptive senders update their transmission rates only every few seconds they might keep on sending with a high rate during congestion periods until a control message indicating losses arrives. During this time, the TCP connection reduces its transmission window and thereby leads to a congestion reduction. Hence, the RTCP messages indicate a reduced congestion state.

## 7 Summary and Future Work

In this paper, we presented a new approach for dynamically adjusting the sending rate of applications to the congestion level observed in the network. This adjustment is done in a TCP-friendly way based on an enhanced TCP-throughput model. The senders can increase their sending rate during underload situations and then reduce it during overload periods. We have run various simulations investigating the performance of the scheme, its effects on TCP connections sharing the same bottleneck and its fairness.

In terms of TCP-friendliness DAA performs better than schemes based solely on the AIMD approach [4]. However, the results presented here suggest that the adaptation approach used for DAA describes more of “*how not to*” realize congestion control for multimedia communication than “*how to*”. Using the TCP-model to periodically set the transmission rate results in a very oscillative transmission behaviour which is not suitable for multimedia streams. Due to rapid variations in the network congestion state observing losses and delays for short periods of time and using those values to estimate the TCP-friendly bandwidth share using the TCP-model results in very oscillative values that do not often resemble the actual resource availability in the network. To achieve a more stable behavior the network congestion state over moving windows of several seconds need to be observed. Work done based on such an approach [19] confirms these observations.

Another major point that needs to be considered here, is the value to use for the additive increase factor ( $A$ ). In our simulations, we set the  $A$  based on running different simulations and choosing the most appropriate results. In order to cover a wide range of possible communication scenarios, the increase rate needs to be set dynamically in a way that is suited to the number of competing flows, the available resources and network load. Better tuned additive increase factors might also lead to a more stable bandwidth distribution.

## 8 Acknowledgments

The RTP/RTCP simulation models were mainly implemented by Timur Friedman and improved by Frank Emanuel. The comments of Adam Wolisz are gratefully acknowledged and were the basis for various aspects of this work.

## References

1. K. Thompson, G. J. Miller, and R. Wilder, “Wide-area Internet traffic patterns and characteristics,” *IEEE Network*, vol. 11, no. 6, pp. –, November/December 1997.
2. Sally Floyd and Fall Kevin, “Router mechanisms to support end-to-end congestion control,” Tech. Rep., LBL-Berkeley, Feb. 1997.
3. Jean-Chrysostome Bolot, Thierry Turletti, and Ian Wakeman, “Scalable feedback control for multicast video distribution in the internet,” in *SIGCOMM Symposium on Communications Architectures and Protocols*, London, England, Aug. 1994, ACM, pp. 58–67.
4. Ingo Busse, Bernd Deffner, and Henning Schulzrinne, “Dynamic QoS control of multimedia applications based on RTP,” *Computer Communications*, vol. 19, no. 1, pp. 49–58, Jan. 1996.
5. J. Padhye, V. Firoiu, D. Towsley, and J. Kurose, “Modeling TCP throughput: A simple model and its empirical validation,” in *ACM SIGCOMM '98*, Vancouver, Oct 1998.
6. Reza Rejaie, Mark Handley, and Deborah Estrin, “An end-to-end rate-based congestion control mechanism for realtime streams in the Internet,” in *Infocom'99*, New York, March 1999, IEEE.

7. Stephen Jacobs and Alexandros Eleftheriadis, "Providing video services over networks without quality of service guarantees," in *RTMW'96*, Sophia Antipolis, France, Oct. 1996.
8. J. Padhye, J. Kurose, D. Towsley, and R. Koodli, "A model based TCP-friendly rate control protocol," in *Proc. International Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV)*, Basking Ridge, NJ, June 1999.
9. Lorenzo Vicisano, Luigi Rizzo, and Jon Crowcroft, "TCP-like congestion control for layered multicast data transfer," in *Proceedings of the Conference on Computer Communications (IEEE Infocom)*, San Francisco, USA, Mar. 1998.
10. Thierry Turetli, Sacha Fosse Prisis, and Jean-Chrysostome Bolot, "Experiments with a layered transmission scheme over the Internet," Rapport de recherche 3296, INRIA, Nov. 1997.
11. Dorgham Sisalem and Adam Wolisz, "MLDA: A TCP-friendly congestion control framework for heterogeneous multicast environments," in *Eighth International Workshop on Quality of Service (IWQoS 2000)*, Pittsburgh, PA, June 2000.
12. Dorgham Sisalem, "End-to-end quality of service control using adaptive applications," in *IFIP Fifth International Workshop on Quality of Service (IWQOS '97)*, New York, May 1997.
13. H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, "RTP: a transport protocol for real-time applications," Tech. Rep. RFC 1889, Internet Engineering Task Force, Jan. 1996.
14. R. El-Marakby and D. Hutchison, "Towards managed real-time communications in the Internet environment," in *The Fourth IEEE Workshop on the Architecture and Implementation of High Performance Communication Systems (HPCS'97)*, Chalkidiki, Greece, June 1997.
15. Sally Floyd and Van Jacobson, "Random early detection gateways for congestion avoidance," *IEEE/ACM Transactions on Networking*, vol. 1, no. 4, pp. 397-413, Aug. 1993.
16. W. Richard Stevens, *TCP/IP illustrated: the protocols*, vol. 1, Addison-Wesley, Reading, Massachusetts, 1994.
17. Frank Emanuel, "Investigation of scalability and fairness of adaptive multimedia applications," Studienarbeit, School of Engineering, TU Berlin, Berlin, May 1997, Work in progress.
18. D. Chiu and R. Jain, "Analysis of the increase/decrease algorithms for congestion avoidance in computer networks," *Journal of Computer Networks and ISDN*, vol. 17, no. 1, pp. 1-14, June 1989.
19. Sally Floyd, Mark Handley, Jitendra Padhye, and Joerg Widmer, "Equation-based congestion control for unicast applications," in *SIGCOMM Symposium on Communications Architectures and Protocols*, Stockholm, Sweden, Aug. 2000.