

# MLDA: A TCP-friendly Congestion Control Framework for Heterogeneous Multicast Environments

Dorgham Sisalem  
GMD FOKUS  
Berlin, Germany  
sisalem@fokus.gmd.de

Adam Wolisz\*  
GMD Fokus/TU Berlin  
Berlin, Germany  
wolisz@ee.tu-berlin.de

*Abstract*—To avoid overloading the Internet and starving TCP connections, multimedia flows using non-congestion controlled UDP need to be enhanced with congestion control mechanisms. In this paper, we present a general framework for achieving TCP-friendly congestion control called MLDA. Using MLDA, multimedia senders adjust their transmission rate in accordance with the network congestion state. For taking the heterogeneity of the Internet and the end systems into account, MLDA supports layered data transmission where the shape and number of the layers is determined dynamically based on feedback information generated by the receivers. Performance tests of MLDA as well as comparisons to other congestion control schemes suggest that MLDA achieves TCP-friendly congestion control on the one hand and is able to accommodate the needs of heterogeneous receivers on the other.

## I. 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 adding 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 it is well suited for multicast communication. However, deploying UDP in the Internet on a large scale might result in extreme unfairness towards competing TCP traffic. In response to losses in the network, TCP connections sharing the same congested links with UDP flows reduce their transmission rates. However, without any rate reduction on behalf of the non-congestion-controlled traffic, the TCP connections would starve and receive a much smaller bandwidth share than the competing UDP flows [2]. 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 with 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 only over the entire life time of the flow and not at every time point [2].

Several aspects need to be considered when designing congestion control mechanisms for multicast communication:

- **Rate Adaptation:** Similar to the case of unicast congestion control, multicast congestion control should be TCP-friendly. Here, TCP-friendliness indicates that the bandwidth share consumed by the multicast flow on any traversed path resembles the bandwidth share of a TCP connection traversing the same path. As the bandwidth share of a TCP connection depends on its round trip delay and losses, calculating a TCP-friendly bandwidth share involves determining losses and round trip times on all paths traversed by the multicast session. Hence, multicast congestion control schemes need to support scalable and accurate loss and delay measurement approaches.
- **Scalability:** The performance of the control scheme should not deteriorate with increasing numbers of receivers. Additionally, the amount of data gathered at the end systems or transmitted between the end systems should be sustainable within the available resources.
- **Heterogeneity:** Internet links as well as the end systems connected to the Internet vary greatly in their capabilities and resources. Multicast congestion control schemes need to take this heterogeneity into account and aim at satisfying the requirements of a large part if not all possible receivers. So, for the case of  $n$  receivers one might actually determine a set of  $[r_1, r_2, \dots, r_n]$  rates the sender needs to adjust its transmission rate simultaneously to, to satisfy the needs of all receivers. This might be achieved by simulcasting the same content at the different rates [3] or referring to hierarchical data transmission [4], [5] by dividing a data stream into a base layer representing the transmitted data content in a basic quality and several enhancement layers that combined with the base layer improve the overall perceived quality of the data stream. Each layer is then sent on a different multicast group. As multicast routers only forward data requested by the receivers, the receivers can determine how many sessions to join and thereby adjust their QoS in respect to their own requirements and capacities. For the case of  $n$  receivers with  $[r_1, r_2, \dots, r_n]$  possible rates with  $r_i$  as an increasing

\* The work was partially done during the author's stay in ICSI Berkeley

value, the sender might set the rate of the lower layer to  $r_1$ , the first enhancement layer to  $r_2 - r_1$  and so on. Each receiver would then join up to  $n$  layers based on its capacity.

In this paper, we describe a general TCP-friendly congestion control approach for handling the case of heterogeneous multicast groups called multicast enhanced loss-delay based adaptation algorithm (MLDA). With MLDA the receivers collect information about the congestion state on the paths connecting them to the sender and determine a bandwidth share that would be utilized by a competing TCP connection traversing the same path under the same loss and delay conditions. To satisfy the needs and capabilities of heterogeneous receivers, MLDA incorporates the concept of layered data transmission in its architecture. Adaptive schemes using layered data transmission usually assume statically set layer sizes. However, to better accommodate the actual heterogeneity of the receivers, MLDA sends periodically collect the information about the determined bandwidth shares at the receivers, adjust the sizes of the different layers they are transmitting based on these information and inform the receivers about the sizes and addresses of the different layers. The receivers can then determine the number of layers to join based on the information announced by the sender and their own estimation of the TCP-friendly rate they could be using.

MLDA is a hybrid sender and receiver-based adaptation scheme that combines on the one hand various well known concepts for multicast congestion control such as receiver-based rate calculation presented by Handley et al. [6], layered transmission [4] and dynamic layering presented by Sisalem et al. [7] into a unified congestion control architecture. On the other hand, MLDA provides novel solutions for round trip delay measurements, scalable feedback collection and a general framework for the cooperation between the sender and receivers that enables a seamless integration of these different concepts.

To allow for scalable feedback and yet provide the sender with enough information about the actual heterogeneity of the receivers we introduce a novel feedback approach called “partial suppression” that allows the exchange of control messages in a timely manner and to suppress the transmission of receiver information of similar content.

In Sec. II we present a brief overview of some related work in the area of TCP-friendly congestion control. In Sec. III the general behavior of receivers and senders in the MLDA framework is presented as well as different schemes for realizing scalable feedback and synchronization among the receivers. In Sec. IV, we investigate through simulations the performance of MLDA in a variety of settings and its ability to accommodate the needs of heterogeneous multicast receivers in a TCP-friendly manner.

## II. BACKGROUND AND RELATED WORK

Recently, there has been several proposals for TCP-friendly adaptation schemes that vary in their complexity and efficiency.

Various congestion control schemes for UDP-based communication deploy an analytical model [8] of TCP for estimating the TCP-friendly bandwidth share of the UDP flow. With this model, the average bandwidth share of a TCP connection ( $r_{TCP}$ )

is determined as

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

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

As an example for an approach that uses Eqn. 1 for congestion control, Handley et al. present in [6] a scheme in which the receivers estimate using Eqn. 1 the rate the sender should be using and inform the sender about this value. As this approach relies solely on the TCP-model, the receivers always need to estimate a loss value to be able to use Eqn. 1. This is achieved by making loss measurements over long time intervals.

Vicisano et al. [5] present a control scheme supporting layered streams. Here, the receivers join or leave a layer based on their measured loss value. Using specially flagged packets the sender indicates synchronization points at which receivers might join or leave a specific layer. The scheme does not consider the round trip delay in its adaptation behavior and only supports static layering of data.

The packet pair receiver-driven layered multicast (PLM) [9] is based on layered transmission and on the use of the packet pair approach [10] to infer the bandwidth available at the bottleneck to decide which are the appropriate layers to join. To apply the packet pair approach for reliably estimating a flow’s bandwidth share, the authors of PLM assume that all routers in the network deploy some kind of a fair queuing mechanism that allocates each flow a fair bandwidth share.

Sisalem et al. present in [7] a first approach for incorporating sender-based adaptation with dynamic shaping of data layers.

Jiang et al. [11] present a scheme in which the sender transmits its data in a fixed base layer and a variable enhancement layer. Based on their measured losses, the receivers estimate their bandwidth share and report this to the sender. The transmission rate of the variable layer is then determined as to increase the satisfaction of most of the receivers.

## III. MULTICAST ENHANCED LOSS-DELAY ADAPTATION ALGORITHM (MLDA)

MLDA provides for a general framework for realizing congestion control in heterogeneous environments. The functionality of MLDA is realized on an end-to-end basis without requiring any support from the network routers beyond their capability of forwarding and routing multicast traffic.

The basic message exchange of MLDA is as follows:

1. The sender periodically transmits reports containing information about the sent layers.
2. After receiving a sender report each receiver ( $j : j = 0, 1, \dots, n$ ) measures the loss and delay of the incoming data for a period of time and determines a TCP-friendly bandwidth share ( $r_j$ ) the sender could utilize on the path connecting the sender and receiver.

3. Based on the calculated share and the rates of the layers as reported in the sender reports the receivers decide to join a higher layer, stay at or leave the current one.
4. Further, the receivers schedule the transmission of reports indicating their calculated bandwidth share after a random period of  $T_{\text{wait}}$ . Finally, if a report from another receiver with rate indication similar to  $r_j$  was seen before  $T_{\text{wait}}$  expires the receiver suppresses the transmission of its report.
5. Based on the receiver reports, the sender adjusts the sizes of the different layers.

In the following subsections, see Sec. III-A and Sec. III-B, the behavior of the sender and receivers is presented. Reliability and synchronization issues are then described in Sec. III-D and Sec. III-E.

#### A. Sender Behavior

With MLDA the sender is responsible for adjusting its transmission behavior in accordance with the bandwidth available for the multicast session as estimated and reported by the receivers. The sender periodically polls feedback information from the receivers by sending a sender report at intervals of  $(T_{\text{control}} + p)$ .  $p$  is a uniformly distributed random variable that assures that the reports from senders that start at the same time do not get synchronized. The sender includes in its reports information about the number of layers ( $y$ ) it is transmitting, the rate ( $R_{L_k} : k = 1, 2, \dots, y$ ) of each layer ( $L_k$ ) and the address on which each layer can be received.

The receiver reports indicate the TCP-friendly bandwidth share estimated by the receivers to be available on the paths connecting the sender to them. The collected reports in between the sending of two sender reports are then used to adapt the sizes of the transmitted layers before sending the next sender report. To allow for fast reactions to changes in the network conditions the sender reduces the rate of the base layer whenever it receives a feedback message indicating a rate request lower than that used for the base layer.

##### A.1 Data Layering

Ideally, the sender would partition its data stream into a number of layers that satisfies the needs of all receivers. In a communication scenario with  $n$  receivers reporting transmission rates of  $(r_1, \dots, r_n : r_{i-1} < r_i < r_{i+1})$  the sender would then need to partition its data stream into substreams of  $(r_2 - r_1, r_3 - r_2, \dots, r_n - r_{n-1})$ . Each receiver determining  $r_j$  as its available bandwidth share would then join  $x$  layers with  $(r_j = \sum_{i=1}^x R_{L_i})$  for  $(R_{L_1} = r_1 \text{ and } R_{L_i} = r_i - r_{i-1})$ .

However, using a large number of layers might result in drift problems [12], increased delays due to the need to synchronize the different layers and higher complexity at the receivers. As an approximation usually only a few layers are used.

Note, that any approach for dividing the data in different layers could be used. The layers could be chosen based on the coding used or the number of receivers requesting a certain rate if such information are available [13]. The only precondition for using such a dynamic layering approach is the availability of

coders that can dynamically shape the number and sizes of layers [14].

When dividing data into layers, each layer should use its own ranges of sequence numbers. That is, while packets belonging to the same layer should have consecutive sequence numbers, this is not needed among packets belonging to different layers. This would allow the receivers to determine the loss rates of each layer. However, the receivers still need some means for resynchronizing the different layers into one data stream. This can be achieved by adding timestamps in the data packets headers indicating the generation time of the packets as is done with the real time transport protocols (RTP) [15].

#### B. Receiver Behavior

Within the MLDA architecture, the receivers measure the characteristics of the paths connecting the sender to them, estimate the TCP-friendly bandwidth share they could consume, join the appropriate number of layers in accordance with their estimated bandwidth share and inform the sender about their estimated shares. In this section, the rules governing the join and leave actions of MLDA receivers are presented.

The join and leave actions of the receivers are triggered by the sender reports which are generated periodically in intervals of  $T_{\text{control}}$  by the sender.

After receiving a sender report the receivers measure the losses of the incoming data stream for a period of  $(T_o \times x)$  with  $T_o$  as the minimum measurement time needed to obtain usable loss information.  $x$  indicates the number of layers the receiver is already tuned to. After the measurement period each receiver ( $j$ ) calculates the rate ( $r_j$ ) that would be appropriate to use on the link connecting the sender to him.

With  $R_{L_k}$  as the transmission rate on layer  $L_k$  the receiver can now take one of the following actions:

- $r_j \geq \sum_{k=0}^{x+1} R_{L_k}$ : The determined TCP-friendly rate is higher than the rate of the incoming data stream in addition to the rate of the next higher layer. The receiver joins the next higher layer and starts an observation time of  $T_o$  to measure the loss values of the now increased incoming stream. After the measurement period a new transmission rate is determined and the receiver can again join a higher layer, leave the just joined layer or stay at the higher layer.
- $r_j < \sum_{k=0}^x R_{L_k}$ : In this case, the rate of the incoming data stream is higher than the theoretically TCP-friendly share determined by some adaptation algorithm. The receiver must, thus, leave the highest layers it is currently receiving until the inequality  $(r_j \geq \sum_{k=0}^x R_{L_k})$  is satisfied.
- $\sum_{k=0}^x R_{L_k} < r_j < \sum_{k=0}^{x+1} R_{L_k}$ : The receiver stays at the current level.

Finally, the receiver schedules the transmission of a report carrying the value of  $r_i$ .

The time passing between issuing a join request for a multicast layer and receiving the data for that layer is usually only the time needed for extending the multicast distribution tree towards the receiver. Hence, joining a multicast session can be thought of as increasing the bandwidth consumed by the receiver instantously

through the addition of a new layer. However, leaving a session only results in a reduction in the rate as measured at a receiver after a period of time. After receiving a leave request, the network routers that forward the multicast data to the receiver that has issued the leave request need to make sure that the session is no longer requested by any other receiver before stopping forwarding the data on the links towards that receiver.

MLDA receivers directly determine the number of layers they can listen to. Hence, there is no need for observing the network situation after dropping a layer and taking the leave latency into account. Note also, that losses caused by a failed join operation by a receiver do not have any effects on the loss estimation of other receivers. That is, if some receivers are listening to  $x$  layers and share a part of the multicast tree they would have an observation period of  $(T_o \times x)$ . Only after the period of  $(T_o \times x)$  would the receivers be allowed to join a higher layer. In case one of the receivers tried to join the next higher layer  $(x + 1)$  and failed, the losses caused by the failed operation would not be included in the loss estimations of the other receivers who have already completed their observation period.

### C. Scalable Feedback

In order to take the heterogeneity of the network and receivers into account in its adaptation decision the sender needs to collect feedback information representing all receivers.

Therefore, we propose a mechanism we call *partial suppression* with which all possible rates a receiver can calculate are divided into  $S$  intervals. That is, if the possible rates a receiver could calculate might vary between  $R_{\min}$  and  $R_{\max}$  we divide this range into subintervals  $[R_{\min}, R_1), [R_1, R_2), \dots, [R_{S-1}, R_{\max})$  with  $R_{\min}$  and  $R_{\max}$  as pre-defined constants. After finishing the observation period each receiver ( $j$ ) calculates a theoretically TCP-friendly rate ( $r_j$ ) and determines in which subinterval this rate is found in. The receiver schedules now the transmission of the receiver report after some time period ( $T_{\text{wait}}$ ). If a report from another receiver indicating a rate in the same subinterval was seen during this period the receiver suppresses the transmission of its own report.

For realizing efficient suppression, Nonnenmacher et al. [16] suggest using a truncated exponentially distributed timer in the interval  $[0, T_{\text{rand}}]$  with the density of

$$T_{\text{wait}} = \begin{cases} \frac{1}{\exp^\lambda - 1} \times \frac{\lambda}{T_{\text{rand}}} \exp \frac{\lambda}{T_{\text{rand}}} z & 0 \leq z < T_{\text{rand}} \\ 0 & \text{otherwise} \end{cases} \quad (2)$$

For  $(\lambda = 10)$  and  $(T_{\text{rand}} = 5c)$  with  $c$  as the delay between two receivers [16] shows analytically that for 10000 receivers less than 10 feedback messages are generated for each event the receivers are reporting on.

For dividing the possible rates into  $S$  subintervals we suggest using the following equation:

$$R_1 = R_{\min} \times (1 + \rho) \quad (3)$$

$$R_s = R_{s-1} \times (1 + \rho) \quad (4)$$

with  $\rho$  as the difference in percentage between two subsequent subintervals and  $(s = 1, 2, \dots, S)$ .

The total number of subintervals ( $S$ ) can then be determined as follows:

$$R_{\max} = R_{\min} \times (1 + \rho)^S \quad (5)$$

Finally, while the sender reports are important for all receivers, the receiver reports are only of meaning to the sender and receivers of similar capacities, i.e., receivers that determine similar theoretical rates. Hence, the sender reports are sent on the base layer. The receivers, however, should send their reports only on the highest layer they are currently listening to. This avoids overloading receivers listening to the lower layers with the reports of the receivers listening also to higher layers.

### D. Synchronous Join and Leave Actions

For the case of multicast, a data stream traverses a network node as long as at least one receiver behind this node is listening to this layer. Thus, the action of leaving a layer results in an overall rate reduction at this node only if all receivers behind this node leave this layer as well. Additionally, if two receivers joined different layers at the same time, the receiver joining the lower layer might observe losses that were not caused by the rate increase due to his join action but by the other receiver joining a higher layer.

To coordinate the actions of the different receivers we use the end of the observation periods as implicit synchronization points. That is, receivers can try to join the first enhancement layer after the end of the observation period of the basic layer ( $T_{o_0}$ ). Receivers listening to the first enhancement layer as well as the base one can join the second one or leave the first enhancement layer at the end of the observation period for the first enhancement layer, i.e.,  $(T_{o_0} + T_{o_1})$  after receiving the sender report.

### E. Reliability Issues

The dependency on the sender reports for initiating the adaptation actions of the receivers might lead to a deadlock situation. During overload periods the sender reports might get lost. However, without these reports the receivers do not start the observation periods and then leave the higher layers in case of losses. Hence, in this case the congestion situation prevails and more sender reports might get lost. To avoid this situation, the receivers schedule a new observation period after a timeout of  $(\sigma \times T_{\text{control}} : \sigma > 1)$  with  $T_{\text{control}}$  as the time period between sending two reports at the sender. Thus, if the sender report was lost, the receiver would start an observation period maximally  $(T_{\text{control}} \times \sigma)$  seconds after receiving the last sender report. After the observation period, the receiver can join or leave a layer and schedule a receiver report as was described above. In case a sender report was received before the timeout expires the scheduled actions are cancelled and new ones are scheduled.

## F. Parameter Settings

In the description of MLDA we have used several parameters that control the temporal behavior of the algorithm. The sender transmits a report every  $(T_{\text{control}} + p)$ , the receivers measure the behavior of a layer for  $T_o$  and schedule the transmission of a report in a period of  $[0, T_{\text{rand}}]$ .

Here we need to consider observation periods for different layers and accommodate possible delays in the leave actions. We therefore set  $T_{\text{control}}$  to 10 seconds and  $p$  arbitrarily to 0.5 seconds. To allow for a good degree of suppression we set  $T_{\text{rand}}$  to 1.5 seconds which is 5 times a typical half of the round trip delay between Europe and the States as measured between Berlin and Berkeley. Finally, note that for taking the adaptation decision based on the rates determined by all receivers, the receiver reports triggered by a sender report need to arrive at the sender before sending the next report. Therefore, the time left for the observation periods for all layers ( $T_{o_{\text{all}}}$ ) can be determined as:

$$T_{o_{\text{all}}} = T_{\text{control}} - p - T_{\text{rand}} - T_{\text{return}} - T_{\text{sync}}$$

with  $T_{\text{return}}$  as the time it takes the receiver reports to arrive at the sender and should be set at least to half the maximum round trip delay. With  $T_{\text{return}}$  and  $T_{\text{sync}}$  set arbitrarily to 0.5 seconds  $T_{o_{\text{all}}}$  has a value of 7 seconds. After several simulations we decided to use an observation period of at least 1.5 seconds. Using smaller values resulted in inaccurate loss values and a highly oscillative adaptation behavior. This means that the scheme supports only up to around 5 layers. While this naturally presents a limitation, using a larger number of layers would, however, increase the complexity of the receivers as they need to resynchronize the data from different layers. As the different layers might actually take different routes to the receiver, they might suffer from different delays. To reconstruct a data segment consisting of data packets sent over different layers the receiver would need to wait until all the different parts are received. This incurs additional delay and thus might reduce the overall perceived quality.

## IV. PERFORMANCE INVESTIGATION OF MLDA

In this part of the work, we study the behavior of MLDA using simulations. Here, we mainly concentrate on the TCP-friendliness of MLDA and its ability to accommodate the needs of heterogeneous receivers.

### A. Estimation of a TCP-Friendly Bandwidth share

As an example for an algorithm for calculating the TCP-friendly bandwidth share to utilize by non-TCP-controlled flows we use here an algorithm called the enhanced loss-delay based adaptation algorithm (LDA+) [17]. While one could use some other scheme, we had already investigated LDA+ in various other studies and achieved good results in terms of stability and TCP-friendliness [17].

In contrast to previous work in [18] and [17], with MLDA the bandwidth share a flow can obtain between the sender and a receiver is estimated at the receiver instead of the sender.

LDA+ is an additive increase and multiplicative decrease algorithm with the addition and reduction values determined dynamically based on the current network situation and the bandwidth share a flow is already utilizing. During loss situations LDA+ estimates a flow's bandwidth share to be minimally the bandwidth share determined with Eqn. 1, i.e., the theoretical TCP-friendly bandwidth share determined using the TCP model. For the case of no losses, the flow's share can be increased by a value that does not exceed the increase of the bandwidth share of a TCP connection with the same round trip delay and packet size.

From Eqn. 1 we can see that for determining a TCP-friendly bandwidth share we need to take losses as well as delays on the links between the sender and receiver into account.

Packet losses can be estimated by including sequence numbers in the packet headers and counting the gaps in the sequence numbers. For estimating the round trip delay from all the receivers to the sender we introduce in [19] a scalable measurement approach based on a combination of one-way delay estimation using timestamps of non-synchronized hosts connected by possibly asymmetrical links and end-to-end measurement of the delay.

### A.1 Simulation Model of MLDA

For the rate estimation part of MLDA we use in this study the loss-delay adaptation algorithm (LDA+) [17]. For dividing the data into  $K$  layers we refer here to a simple approach. The sender determines the minimum ( $r_{\text{min}}$ ) and maximum ( $r_{\text{max}}$ ) reported rates and sets the rate for the basic layer to  $r_{\text{min}}$ . The enhancement layers are then set to  $(\frac{(\alpha \times r_{\text{max}}) - r_{\text{min}}}{K-1})$ . ( $\alpha : \alpha \leq 1$ ) is a dampening factor that reduces the effects of a possible overestimation of the available resources by the receiver reporting  $r_{\text{max}}$ . To avoid establishing layers with a negligible content, the minimum size of a layer is set to  $L_{\text{min}}$ . In the simulations presented here, we set  $\alpha$  to 0.9 and  $K$  to 3. Note that while the chosen layering approach might have some impact on the fairness and stability of MLDA, the actual performance of MLDA does not depend on using a particular layering scheme.

As a simplification, it is assumed that the clocks of the sender and receivers are synchronized and that the round trip delay is measured accurately. This allows us to better investigate the aspects of TCP-friendliness and stability of the scheme without having to consider possible side effects of measurements errors.

Finally, we assume that the time passing between sending a leave request and this leave action actually taking effect is constant and is set to one second.

### A.2 Performance of MLDA in Heterogeneous Multicast Environments

To test the performance of MLDA in heterogeneous multicast environments we use the topology depicted in Fig. 1 with a multicast session consisting of a sender and 6 receivers connected to the sender over routers with different capacities. Each router is shared between an MLDA stream and  $m$  TCP connections that have the same end-to-end propagation delay as that of the MLDA sender/receiver pair. The TCP connections are modeled as FTP

flows that always have data to send and last for the entire simulation time. A TCP-Reno [20] model was used for simulating the congestion control behavior of TCP. The sender transmits packets of 1 kbytes and each router is a random early drop (RED) [21] router. 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 that all flows receive the same loss ratio and avoids synchronization among the flows. Actually, the measurements conducted in [19] and other simulation studies [22] suggest that the performance of MLDA is not affected by the used buffer management scheme. Here we use a maximum queuing delay of 0.15 seconds and set the maximum and minimum thresholds to 0.8 and 0.3 of the maximum buffer size. The router R0 works only as a distributor of data and incurs no losses or delays to the data. In our simulations we set the round trip propagation delay between the sender and the receivers to 200 msec.

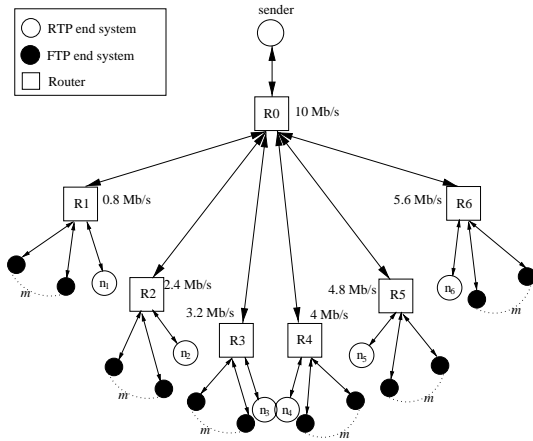
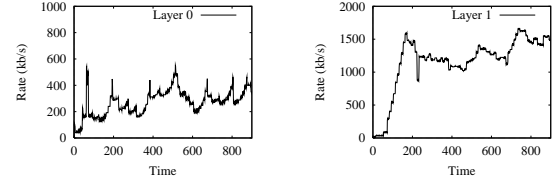


Fig. 1. Multicast testing topology

Fig. 2 shows the distribution of bandwidth among the different layers for the case when each router is shared between an MLDA flow and a TCP connection. We can observe that the rate of the basic layer, see Fig. 2(a), is adjusted in accordance with the capacity of receiver  $n_0$ .

Fig. 3 displays the bandwidth distribution between the competing MLDA and TCP flows at the different routers.

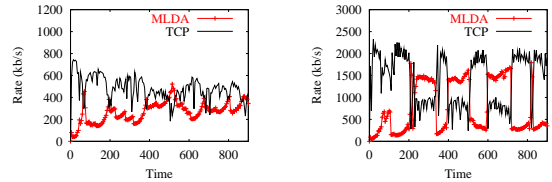
For the case of different numbers of competing TCP connections ( $m$ ) Tab. I indicates that the ratio of the average bandwidth share of the MLDA receivers to the average bandwidth share of the competing TCP connections at each router ( $R_n$ ) varies around one. Both the results in Fig. 3 and Tab. I indicate that MLDA is in general fair towards the competing TCP connections and manages to satisfy the capabilities of the heterogeneous receivers. While the bandwidth shares of the receivers with the lower bandwidth ( $n_1$  and  $n_2$ ) is restricted by the adapta-



(a) Layer 0

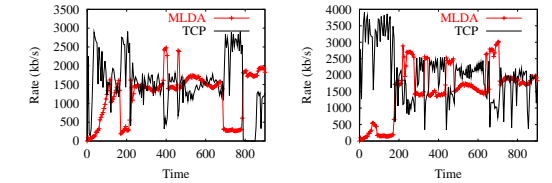
(b) Layer 1, 2

Fig. 2. Bandwidth distribution of the layers



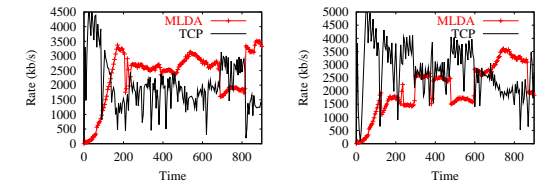
(a) Router 1

(b) Router 2



(c) Router 3

(d) Router 4



(e) Router 5

(f) Router 6

Fig. 3. Bandwidth distribution between TCP and MLDA at the different routers

tion scheme, receivers that have a TCP-friendly bandwidth share that is larger than the sum of the first  $x$  layers but below the sum of the  $x + 1$  layers will oscillate between layer  $x$  and  $x + 1$ . Depending on the chosen layering approach and number of layers this might result in temporary unfair bandwidth distribution. Improved fairness and smaller oscillations can only be reached using a larger number of layers [23] which, however, increases the complexity of the scheme as the end systems need to manage and resynchronize a larger number of layers.

Flows ( $m$ )	$R_1$	$R_2$	$R_3$	$R_4$	$R_5$	$R_6$
1	0.66	0.70	1.03	1.04	1.5	0.95
8	0.50	0.90	0.85	1.09	0.92	1.02

TABLE I

RATIO OF AVERAGE BANDWIDTH SHARE OF THE MLDA FLOW TO THE AVERAGE SHARE OF COMPETING TCP CONNECTIONS

### A.3 Performance of MLDA in Dynamical Environments

In this section, we investigate the behavior of MLDA in a dynamical receiver setting, i.e., a setting with the receivers joining and leaving the multicast session during the simulation time. For this purpose, we repeat the simulations of Sec. IV-A.2 but with receiver ( $n_1$ ) joining the multicast session at time 300 seconds and leaving it at time 600 seconds. Each router is shared between the MLDA flow and uncorrelated background traffic which consumes maximally half of the router's capacity. The background traffic is modulated as the aggregate of 100 on-off processes with the on period lasting for the time needed to carry a number of packets drawn from a Pareto distribution with the factor of 1.1 and a mean of 20 packets and the off period lasting for a time drawn from a Pareto distribution with a factor of 1.8 and a mean of 0.5 seconds [24].

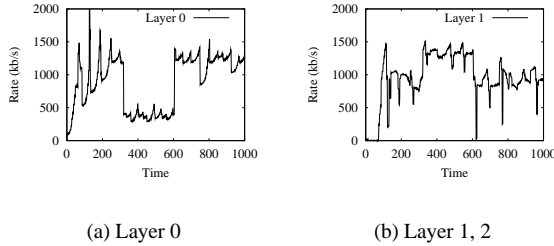


Fig. 4. Bandwidth distribution of the layers

As Fig. 4 depicts, during the first 300 seconds the bandwidth share of the lowest layer is increased up to around 1.2 Mb/s which is the bandwidth share of receiver  $n_2$  which constitute the worst receiver in this case. After receiver  $n_1$  joins the session the size of the base layer is reduced down to the appropriate level of the new worst receiver, i.e., 400 kb/s. The sizes of the upper layers is increased to compensate the reduction in the base layer. Thus the receivers listening to the higher layers are not affected by the reduction in the base layer. The share of receiver ( $n_6$ ) as depicted in Fig. 5(f) is not changed due to the join and leave actions and is comparable to the results achieved in Sec. IV-A.2, see Fig. 3(f). After receiver  $n_1$  leaves the session at 600 seconds the size of the base layer is increased again and, hence, improving the bandwidth share of the receivers listening only to the lower layer.

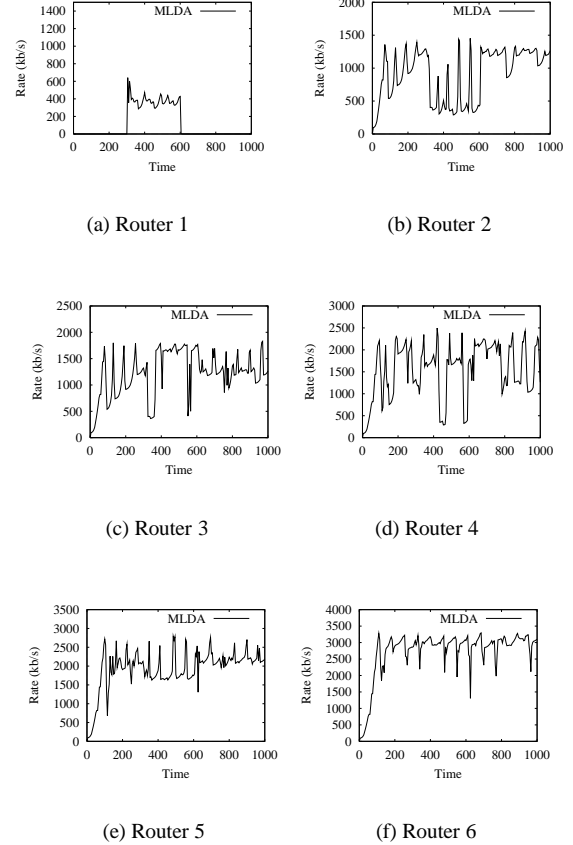


Fig. 5. Bandwidth share of MLDA at the different routers

### A.4 Performance of MLDA in Multicast Environments with Shared Links

In the previous section, see Sec. IV-A.2 and Sec. IV-A.3, we only considered multicast distribution trees without any shared links among the receivers. This was beneficial for investigating the TCP-friendliness of MLDA and its ability to accommodate heterogeneous receivers without having to consider the effects of interactions between different receivers, traversing multiple routers and different round trip delays among the receivers.

Fig. 6 depicts another configuration of a multicast tree that is shared among different receivers with each link of the tree having a different capacity. Similar to Fig. 1, RED routers are used. Each router is shared between the MLDA flow and uncorrelated background traffic which consumes maximally half of the router's capacity.

The bandwidth distribution among the layers, see Fig. 7, accommodates in this case the capacities of the receivers such that receiver  $n_5$  manages to get a bandwidth share of around 170 kb/s and the receivers connected to the first router ( $R_1$ ) get a bandwidth share of 540 kb/s.

As Fig. 8 suggests that receivers connected to the same router over links with similar bandwidth capacities but with different round trip delays receive identical bandwidth shares and stay

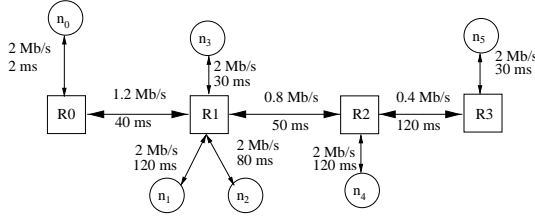
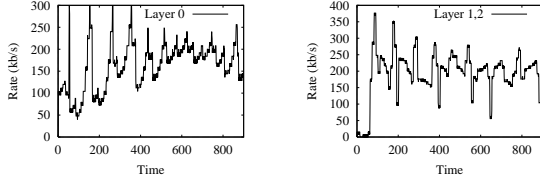


Fig. 6. Testing topology of shared multicast trees



(a) Layer 0

(b) Layer 1,2

Fig. 7. Bandwidth distribution of the layers

synchronized throughout the simulation time. Also, notice that receiver ( $n_4$ ) oscillates between the first two layers which might cause losses in the stream forwarded further towards receiver  $n_5$ . However, due to the synchronization mechanisms of MLDA those losses are ignored at receiver  $n_5$  which does not include the losses caused of the failed join operations of receiver  $n_4$  into its loss estimations.

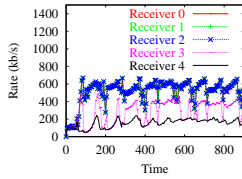


Fig. 8. Bandwidth distribution among the receivers

## V. COMPARISON OF THE PERFORMANCE OF MLDA TO OTHER CONGESTION CONTROL SCHEMES

In this part of the work, we compare between the performance of MLDA and a row of recent proposals for TCP-friendly congestion control schemes for multicast communication. Due to the large number of such proposals we restrict our comparisons to only a few that combine TCP-friendly adaptation with layered data transmission. For comparing the schemes, we picked out some representative test cases as were described in the papers presenting the to be compared algorithms. We re-simulated those cases in our simulation environment and compared the achieved results using MLDA with the results achieved by the other schemes as were reported by their authors. This approach reduces possible errors in the comparisons due to misinterpretations or wrong implementations of the algorithms.

### A. Comparison of MLDA and PLM

The packet pair receiver-driven layered multicast (PLM) [9] is based on layered transmission and on the use of the packet pair approach [10] to infer the bandwidth available at the bottleneck to decide which are the appropriate layers to join. PLM is receiver driven, i.e., congestion control is achieved through the join and leave actions of the receivers. The packet pair approach is used usually to estimate the bandwidth of the bottleneck router on the path between the sender and receiver. To apply the packet pair approach for reliably estimating a flow's bandwidth share, the authors of PLM assume that all routers in the network deploy some kind of a fair queuing mechanism that allocates each flow a fair bandwidth share. Only under this assumption, which is currently invalid in the Internet, is it possible to use PLM for congestion control.

With PLM the sender periodically sends a pair of its data packets as a burst to infer the bandwidth share of the flow. The receivers use the gaps between the specially marked packet pairs to estimate their bandwidth share. Based on the estimated share they determine the number of data layers they can receive.

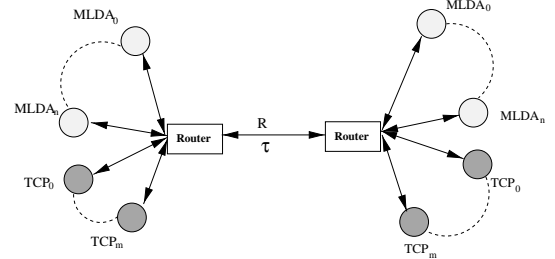


Fig. 9. Testing topology

For comparing the TCP-friendliness of PLM and MLDA we use the topology depicted in Fig. 9 with a bottleneck link shared between two TCP ( $m = 2$ ) connections and an MLDA flow ( $n = 1$ ). The bottleneck link has a bandwidth of 300 kb/s and a delay of 0.02 seconds. For reducing the effects of synchronization and ensuring a fair distribution of the losses we use a RED router with a maximum buffering delay of 0.2 seconds and the maximum and minimum thresholds set to 0.3 and 0.8. The initial transmission rate of the MLDA flow was set to 100 kb/s and the packet size was set to 500 bytes. The first TCP connection starts at time 0 seconds and the second one 60 seconds later. The adaptive flow starts at time 20 seconds. For PLM, a packet pair was sent each second and each flow had a queue size of 20 packets. Additionally, the sender transmitted 17 layers each of 20 kb/s. This topology was taken from [9] Fig.[2] whereas in [9] a router with fair queuing was used.

The results presented in Fig. 10 describe the bandwidth distribution between the TCP and MLDA flows when using PLM and MLDA. Measured over the entire simulation time, both approaches achieve similar average bandwidth distributions with the adaptive flow receiving around 80 kb/s and the TCP flows having a share of 110 kb/s. However, looking at the temporal behavior of the flows in Fig. 10 it can be observed that with PLM

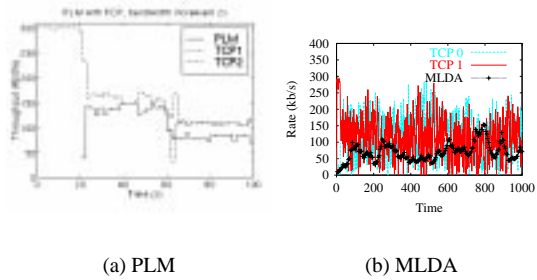


Fig. 10. Bandwidth distribution with MLDA and PLM

the flows show hardly any oscillations and have rather constant bandwidth shares. With MLDA on the contrary, the TCP flows oscillate between 0 kb/s and 250 kb/s. While the MLDA flow itself shows a less oscillatory behavior than TCP it still shows a variance of  $\pm 50\%$  of its average bandwidth share. With PLM the receivers always know their exact bandwidth share and as the number of flows in the network is constant the bandwidth shares of the flows is constant as well. Hence, this stable behavior of the flows with PLM was to be expected.

Note that PLM not only assumes a fair queuing network but also that the packet pair approach always results in correct estimations of the bandwidth share. However, even in a fair queuing network, the packet pair approach might result in under or over-estimation of the bandwidth share. That is, due to losses of probe packets, network congestion or interference of other traffic this approach might result in wrong estimations of the bandwidth. Hence, the receivers need to deploy mechanisms such as [25], [10] to filter out wrong estimates. As such effects are difficult to simulate, the results presented in [9] for PLM are to be considered rather optimistic. Due to the effects of losses and traffic interference a higher degree of oscillations can be expected in a more realistic environment.

### B. Comparison of MLDA and RLC

Vicisano et al. present in [5] a receiver-driven layered control scheme (RLC) for realizing TCP-friendly congestion control. With RLC, the sender divides its data into layers and sends them on different multicast sessions. To test the availability of resources, the sender periodically generates short bursts of packets followed by an equally long relaxation period in which no packets are sent. For the duration of the bursts the consumed bandwidth by the flow is doubled. Hence, if the packets of the burst are not lost then this indicates the availability of resources. After receiving a packet burst, the receivers can join a higher layer if the burst was lossless otherwise they remain at their current subscription level. The receivers might leave a layer at any time if losses were measured.

In [5] a simulation topology is used similar to Fig. 9 but with the round trip delay ( $\tau$ ) set to 0.42 seconds and the bottleneck bandwidth ( $R$ ) set to 1.5 Mb/s. The link is shared between 8 TCP connections and 8 adaptive flows and the packet size is set to

1024 bytes. Fig. 11(a) suggests that RLC shows a very conservative behavior under these conditions and the RLC flow receives only around 60% of the bandwidth share consumed by the TCP connection. The behavior of MLDA in this case resembles the behavior of TCP to a greater extent with the MLDA flow receiving around 90% of the bandwidth consumed by the TCP connection, see Fig. 11(b).

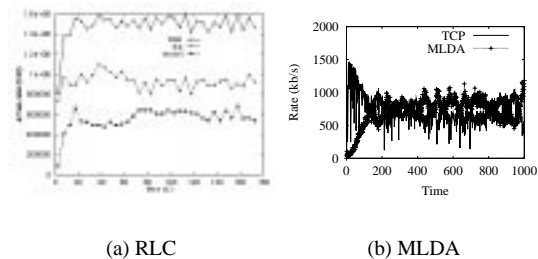


Fig. 11. Bandwidth distribution with RLC and MLDA

In a second test, we compare the behavior of RLC and MLDA in a multicast tree with shared links, see Fig. 12. This topology is similar to that described in Sec. IV-A.4 but with different delay and bandwidth parameters. Also in this case, the routers are shared between the adaptive flow and uncorrelated traffic.

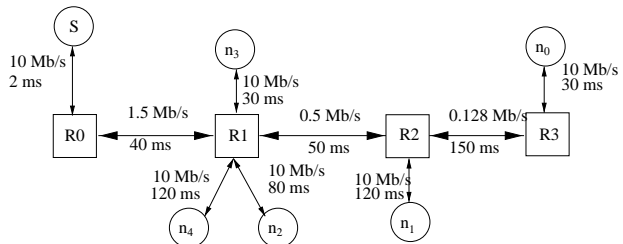


Fig. 12. Testing topology of shared multicast trees

As Fig. 13 describes, receivers connected to the same bottleneck receive identical shares and stay synchronized throughout the simulation time with both RLC and MLDA. However, with RLC the maximum bandwidth share of the receivers ( $n_2, n_3, n_4$ ) is restricted by the static nature of the transmitted layers. That is while the background traffic only consumes half of the 1.5 Mb/s, the receivers ( $n_2, n_3, n_4$ ) can only utilize up to 500 kb/s instead of 750 kb/s. With MLDA the sizes of the layers is shaped in accordance with the available resources and the receivers ( $n_2, n_3, n_4$ ) receive on the average a bandwidth share of around 750 kb/s. Hence, while MLDA introduces a higher complexity due to the exchange of control messages between the sender and receivers it is more capable of accommodating the needs of heterogeneous receivers than RLC.

## VI. SUMMARY AND FUTURE WORK

In this paper, we presented an adaptive rate control mechanism for multicast communication called MLDA. We have tested

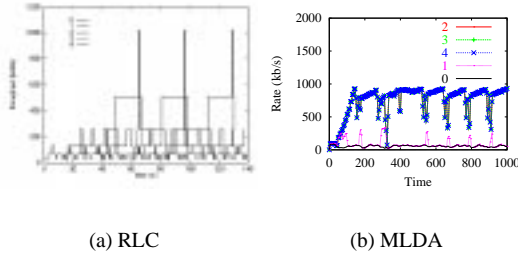


Fig. 13. Behavior of RLC and MLDA in a shared multicast tree

MLDA under various topologies with various parameters and compared the behavior of MLDA to a range of recently proposed TCP-friendly congestion control schemes. Our simulations and measurements, see [19] for more details on the measurement results, suggest the efficiency of the scheme and its friendliness towards competing TCP traffic.

The results of comparing MLDA to other congestion control schemes, suggest that while the bandwidth distribution with MLDA is not as stable as that achieved with network-supported scheme such as [9] it is nevertheless still TCP-friendly. Compared to simple schemes that do not require the exchange of control information or the measurement of round trip delays, MLDA achieves a more TCP-friendly bandwidth distribution and is better suited to accommodate the needs of heterogeneous receivers.

While MLDA was presented as a complete scheme it could be considered as a set of mechanisms that can be easily combined with other approaches. For example, we could replace the rate calculation at the receivers (LDA+) with some other such as [6]. The layering approach can be replaced by another one that might take better into account the heterogeneity of the network and the constraints imposed by the used coder or content. The here proposed mechanisms for providing the sender with information about the heterogeneity of the network (partial suppression) could also be used separately in some other congestion control approaches.

## VII. ACKNOWLEDGMENTS

The RTP/RTCP simulation models were mainly implemented by Timur Friedman and improved by Frank Emanuel. The comments of Henning Sanneck and Henning Schulzrinne are gratefully acknowledged.

## 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 Kevin Fall, "Promoting the use of end-to-end congestion control in the internet," *IEEE/ACM Transactions on Networking*, Aug. 1999.
- [3] Xue Li and Mostafa H. Ammar, "Bandwidth control for replicated-stream multicast video," in *HPDC Focus Workshop on Multimedia and Collaborative Environments (Fifth IEEE International Symposium on High Performance Distributed Computing)*, Syracuse, New York, Aug. 1996, IEEE Computer Society.

- [4] Steven McCanne, Van Jacobson, and Martin Vetterli, "Receiver-driven layered multicast," in *SIGCOMM Symposium on Communications Architectures and Protocols*, Palo Alto, California, Aug. 1996.
- [5] 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.
- [6] Mark Handley and Sally Floyd, "Strawman specification for TCP friendly reliable multicast congestion control," 1998, Note to the Internet Reliable Multicast Group mailing list.
- [7] Dorgham Sisalem and Frank Emanuel, "QoS control using adaptive layered data transmission," in *IEEE Multimedia Systems Conference'98*, Austin, TX, USA, June 1998.
- [8] 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.
- [9] A. Legout and E. W. Biersack, "Fast convergence for cumulative layered multicast transmission scheme," Tech. Rep., Eurecom, Sophia-Antipolis, France, Oct. 1999, under submission.
- [10] Vern Paxson, *Measurements and Analysis of End-to-End Internet Dynamics*, Ph.D. thesis, Lawrence Berkeley National Laboratory, University of California, Berkeley, California, Apr. 1997.
- [11] Tianji Jiang, Ellen Zegura, and Mostafa Ammar, "Inter-receiver fair multicast communication over the internet," in *Proc. International Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV)*, Basking Ridge, New Jersey, June 1999.
- [12] Uwe Horn and Bernd Girod, "Scalable video coding for the internet," in *8th Joint European Networking Conference*, Edinburgh, England, May 1997.
- [13] Celio Albuquerque, Brett Vickers, and Tatsuya Suda, "An end-to-end source-adaptive multi-layered multicast (SAMM) algorithm," in *9th International Packet Video Workshop*, New York, USA, Apr. 1999.
- [14] E. Miloslavsky and A. Zakhor, "Constant PSNR rate control for layered video compressing using matching pursuits," in *Proceedings of the International Conference on Image Processing*, Kobe, Japan, Oct. 1999.
- [15] 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.
- [16] J. Nonnenmacher and Ernst W. Biersack, "Optimal multicast feedback," in *Proceedings of the Conference on Computer Communications (IEEE Infocom)*, San Francisco, USA, Mar. 1998.
- [17] Dorgham Sisalem and Adam Wolisz, "Towards TCP-friendly adaptive multimedia applications based on RTP," in *Fourth IEEE Symposium on Computers and Communications (ISCC'99)*, Egypt, July 1999.
- [18] Dorgham Sisalem and Henning Schulzrinne, "The loss-delay based adjustment algorithm: A TCP-friendly adaptation scheme," in *Proc. International Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV)*, Cambridge, England, July 1998.
- [19] Dorgham Sisalem and Adam Wolisz, "MLDA: A TCP-friendly congestion control framework for heterogeneous multicast environments -extended version," Tech. Rep., GMD Fokus, Berlin, Germany, May 2000.
- [20] W. Richard Stevens, *TCP/IP illustrated: the protocols*, vol. 1, Addison-Wesley, Reading, Massachusetts, 1994.
- [21] 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.
- [22] Dorgham Sisalem, "TCP-friendly congestion control for multimedia communication," Technical report, GMD, Fokus, Germany, Apr. 2000.
- [23] Dan Rubenstein, Jim Kurose, and Don Towsley, "The impact of multicast layering on network fairness," in *Special Interest Group on Data Communication (SIGCOMM'99)*, Cambridge, USA, Aug. 1999.
- [24] Kihong Park, Gitae Kim, and Mark Corvella, "On the relationship between file sizes, transport protocols and self-similar network traffic," in *International Conference on Network Protocols (ICNP)*, Columbus, Ohio, Oct 1996.
- [25] Kevin Lai and Mary Baker, "Measuring bandwidth," in *Proceedings of the Conference on Computer Communications (IEEE Infocom)*, New York, USA, Mar. 1999.