

# Constrained TCP-Friendly Congestion Control for Multimedia Communication

Dorgham Sisalem\*  
GMD-Fokus, Berlin  
sisalem@fokus.gmd.de

Adam Wolisz  
Technical University Berlin  
wolisz@ee.tu-berlin.de

March 7, 2000

## Abstract

With the lack of admission control and resource reservation mechanisms in the Internet, overload situations can only be avoided by having the end systems deploying congestion control schemes. That is, the end systems should adjust their transmission and reception behavior in accordance with the available network resources. This is generally achieved by adjusting the transmission behavior of a sender based on feedback information from the receiver about the network congestion situation. Current proposals for rate adaptation assume that the sender can adjust its transmission with arbitrary granularity. However, especially for the case of audio and video communication it is often the case, that the used compression style, application or the user might impose strict limitations as to the maximum and minimum transmission rates, the granularity with which the sender can change its transmission rate and the frequency with which such changes should occur.

In this paper, we present a general model for describing a generic constrained multimedia source. Additionally, we describe an control framework called constrained TCP-friendly adaptation framework (CTFAF). CTFAF incorporates the proposed generic multimedia model with TCP-friendly adaptation schemes and hence allows for adapting the transmission rate of multimedia senders in a TCP-friendly way and at the same time takes the restrictions imposed by the source into account. Using simulations we investigate the performance of this approach in terms of bandwidth utilization, stability and fairness.

## 1 Introduction and Motivation

The rapid growth, increasing bandwidth and the availability of low-cost multimedia end systems have made it possible to use the Internet for multimedia applications ranging from telephony to conferencing, distance learning, media-on demand and broadcast applications [17].

However, transmitting such data packets into the network without regard to the actual available resources can easily result in overload situations and data losses. To avoid such a situation some mechanisms are required to control the amount of data packets entering the network.

---

\*The author was partially supported by HRL LLC in Malibu, CA

In the literature there has been generally two approaches for controlling the amount of traffic transported in a network: admission control combined with QoS signaling protocols and data adaptation.

With the combination of admission control [10] and resource reservation [2] a user has to inform the network about the amount of resources he wishes to use. Based on the availability of resources the network decides then to accept or reject the user's data. While such a tight control can ensure that the flows admitted to the network actually receive an adequate amount of network resources, this approach requires considerable changes to the networking infrastructure by introducing protocols for signaling the users' requirements, collecting information about the availability of network resources and accounting and charging the user for the consumed resources. While deploying such an approach in the Internet might prove to be necessary and adequate in the long term, the currently proposed approaches [3] still face different problems in terms of complexity, scalability and applicability.

The second approach is to adapt the amount of data entering the network in accordance with the network congestion situation. This has been widely used in the Internet through the transport control protocol (TCP) [22] used for data applications such as WWW and FTP transports. With TCP, the sender increases its transmission rate by an additive amount while no losses are observed. After observing a loss the rate is reduced by a multiplicative factor. To avoid overloading the network, non-TCP flows, e.g., multimedia flows, should deploy similar congestion control mechanisms to TCP. In addition to avoiding congestion and reducing the possibility of losses adaptation schemes for multimedia flows need to be fair towards competing TCP traffic, i.e., be TCP-friendly [8]. 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. However, TCP shows a rather oscillatory adaptation behavior with fast and large changes in the transmission rate. Such a behavior results in an oscillative perceived QoS that can be rather annoying to the viewer of a video flow for example. Hence, 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 multimedia 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 [8].

Recently, there have been various proposals for realizing TCP-friendly adaptation [16, 13, 9, 20]. Those schemes are mainly based on collecting loss and delay information at the end systems and determining a TCP-friendly transmission rate to use based on these information. The TCP-friendly rate can be either determined by additively increasing the allowed transmission rate during underload situations and multiplicatively reducing it otherwise [16], by using an analytical model for determining the equivalent TCP bandwidth share under similar loss and delay values [13] or by a combination of both approaches [20, 21].

All of those approaches assume that the sender can adjust its transmission rate in accordance with the values determined by the adaptation scheme and in accordance with the dynamics of the TCP-friendly algorithm with arbitrary granularity and with no boundaries as to the maximum or minimum bandwidth values. However, in reality there are certain maximum and minimum bandwidth requirements for multimedia contents above and below which the sent data is meaningless for the receiver. Additionally, the used compression style for a multimedia content might dictate that the changes in the transmission rate of a video stream for example can only be realized in a granular way. Also, there might be some restrictions arising from the user himself with regard to

the acceptable variations in the perceived QoS over a time interval. Such restrictions depend primarily on the transferred content, the used compression style, the communication scenario and the user himself.

To investigate the effects of such restrictions on the adaptation process we present in Sec. 2 a simple model of a generic multimedia source and describe some constraints such a model might impose on the adaptation process. While this model does not represent a specific audio or video compression style, a communication scenario or particular user preferences, the model can be tuned through different parameters to describe a wide range of possible characterizations of multimedia sources and user preferences. Based on this model we describe in Sec. 3 a general framework called constrained TCP-friendly adaptation framework (CTFAF) to assist rate adaptation algorithms in incorporating various constraints in the adaptation process on the one hand and retain fair bandwidth distribution characteristics on the other. As an example of a TCP-friendly rate adaptation scheme we describe in Sec. 4.1 the loss-delay based adaptation algorithm presented earlier in [20, 21] called LDA+. With this approach, the adaptation scheme determines the actions of increasing or decreasing the transmission rate. The effects of enhancing an adaptation algorithm such as LDA+ to accommodate user and media constraints is then investigated in terms of bandwidth utilization, stability and TCP-friendliness in Sec. 5.

## 2 A Generic Model for Multimedia Communication

In designing an adaptation algorithm, we need to take the constraints imposed by the user and transmitted content into account. In this section, we present a generic model that describes possible constraints imposed by the user, compression algorithm and data content on an adaptation scheme. This model considers the following constraints:

**Fixed limits:** The quality of a multimedia stream can usually be improved by increasing the bandwidth share of the stream. However, above a certain limit ( $r_{\max}$ ) no noticeable improvements in the QoS will be observed anymore. Additionally, reducing the bandwidth share of the stream below some minimal rate ( $r_{\min}$ ) would redeem the transported data useless by the receiver. The values of those maximum and minimum values depend highly on the used compression styles, the users and the data content. While transmitting a talking heads kind of video conference at a frame rate of only a few frames per seconds might be acceptable, transmitting a live sports event requires obviously a high frame rate in order to be able to keep track of all instantaneous and rapid changes in the play field.

**Granular adaptation:** Depending on the used compression scheme and data content, a data source might only be able to change its transmission rate in steps of  $S_{\text{adaptation}}$ .

Taking the example of an MPEG video stream, changing the transmission rate can be achieved by either changing the transmission frame rate or the quantization level. In this case, the adaptation can only be realized in steps of ( $S_{\text{adaptation}}$ ) at the granularity of the discrete quantization steps or the size of a video frame. For the case of audio communication, adaptation can often only be realized by changing the used compression style which yields fixed adaptation steps as well.

**Stable presentation:** With TCP the transmission rate is changed in the order of once every round trip time in possibly large steps. Applying such an adaptation behavior to multimedia flows would result in a highly fluctuating perceived quality that can be annoying to the user. This can be observed for example when rapidly changing the frame rate for the case of video communication or continuously changing the used audio compression scheme.

To provide the user with a stable perceived quality, the adaptation scheme needs to limit the maximum changes ( $\delta_{\text{adaptation}}$ ) as well the rate of changes in the transmission rate of a multimedia stream. This allows for smooth changes in the transmission rate and a stable perceived QoS.

**Loss tolerance:** Depending on the transferred content, the user and the used compression scheme, some data losses might be tolerated during a multimedia communication. For example some compressions scheme already include loss recovery mechanisms for hiding data losses from the user. As another example, while users listening to an audio stream carrying music would require a very low loss rate, users involved in an interactive audio conference would be more tolerant to losses. Again here, the tolerated loss depends not only on the loss rate and distribution pattern but also on the fluency of the users in the used communication language for example.

Note, that this model only presents a subset of all the possible constraints that need to be considered when designing an adaptation scheme. Another example for such a constraint results from the case that changing some compression parameters must be followed by a recovery period to allow the senders and receivers to synchronize the encoding and decoding process in accordance with the new parameters.

### 3 A General Framework for Rate Adaptation for Constrained Multimedia Flows

When adapting the rate of a multimedia source in accordance with the network congestion state, on the one hand one has to take the constraints imposed by this source as described by the generic model of Sec. 2 into account and achieve TCP-friendly congestion control on the other.

In this section, we present a general framework for adapting the transmission rate of constrained multimedia sources. With this framework, called the constrained TCP-friendly adaptation framework (CTFAF), the source uses a TCP-friendly adaptation algorithm for determining the bandwidth share of the multimedia flow. However, instead of adjusting the transmission rate of the source solely based on the rate determined by the control algorithm, the constraints imposed by the generic multimedia source are taken into account as well. This means that at some time points, the sender has to reduce its transmission rate by an amount less than determined by the adaptation scheme. To take such divergence from the TCP-friendly behavior into account the sender sums the difference between the bandwidth change values as determined by the adaptation scheme and the changes allowed by the multimedia source into a variable called virtual bandwidth ( $B_{\text{virtual}}$ ). A negative value of  $B_{\text{virtual}}$  indicates that the source was too aggressive in the past and that the sender should reduce its rate by the maximum allowed value by the multimedia model every time a rate reduction is requested by the adaptation scheme. Finally, to avoid the case where an extreme value of  $B_{\text{virtual}}$  af-

ffects the adaptation behavior of the multimedia source over long time intervals,  $B_{\text{virtual}}$  is periodically reset to zero and the transmission rate is adjusted by an averaged value of  $B_{\text{virtual}}$ .

Considering all the possible constraints imposed by multimedia compression styles and contents as well as the users in the adaptation process requires deep knowledge of the compression style as well as the content to be transmitted. Additionally, the minimum and maximum acceptable transmission rates result from the content and the communication scenario. Finally, the acceptable rate of changing the QoS depends also on the content as well as the user himself.

As a simplification of this problem we consider in this part of the work a generic encoder with the following characteristics:

- A minimum bandwidth requirement of  $r_{\text{min}}$  and a maximum one of  $r_{\text{max}}$ .
- The adaptation itself can only be realized in steps of  $(m \times S_{\text{adaptation}})$ .
- To accommodate the requirement of a stable perceived QoS the changing rate ( $\delta_{\text{adaptation}} = \frac{\Delta r}{\Delta t}$ ) of the transmission rate of a sender should not be increased or decreased by more than  $(n \times S_{\text{adaptation}} \leq \delta_{\text{adaptation}})$  during a time interval ( $T_{\text{adaptation}}$ ), whereas  $T_{\text{adaptation}}$  can be considered as the time between two adaptation points, i.e., the time points at which the sender executes the adaptation algorithm and adapts its transmission rate.
- A sender does not need to adjust its transmission rate if losses were below  $l_{\text{allowed}}$  or its transmission rate would go beneath  $r_{\text{min}}$  or above  $r_{\text{max}}$ .

As a generic adaptation algorithm we consider here a scheme that reduces the transmission rate of the sender by an amount  $X_r$  during congestion periods and increases it by an amount of  $X_i$  during network underload situations.

Now imposing constraints on the adaptation process means that during some time intervals the transmission rate of some flow can not be reduced even though losses are measured or the rate can not be increased even though the network is underloaded. To be able to integrate these constraints with the adaptation algorithm and still allow for a fair bandwidth distribution we incorporate the concept of virtual bandwidth share ( $B_{\text{virtual}}$ ) with the adaptation process. A positive  $B_{\text{virtual}}$  represents the amount of bandwidth the flow should have acquired based on the calculations of some adaptation scheme ( $r_{\text{calculated}}$ ) but did not use due to the constraints of the multimedia source. As the unused resources were consumed by competing flows an average equal bandwidth distribution can then be achieved if the flow consumed a bandwidth share larger than the rate calculated by the adaptation algorithm ( $r_{\text{calculated}}$ ) in the future. A negative value indicates the amount of bandwidth used in excess of the amount calculated by the adaptation scheme and hence that the flow was too aggressive in the past and should use a transmission rate lower than  $r_{\text{calculated}}$  when possible. While this approach does not guarantee fair bandwidth distribution at all times it allows for a fair distribution over long time periods.

With such an enhancement to an adaptation algorithm, the increase and decrease behavior of the control algorithm is influenced by the value of  $B_{\text{virtual}}$  in addition to the limitations imposed by the user or the data source and the network congestion state.

$B_{\text{virtual}}$  is initialized to 0 at the beginning of the transmission process and then adjusted as follows:

- If the sender is allowed to increase its transmission rate by an amount  $\{X_i : X_i > \delta_{\text{adaptation}}\}$  then the sender increases its transmission rate by  $\delta_{\text{adaptation}}$  and  $B_{\text{virtual}}$  is set to

$$B_{\text{virtual}} = B_{\text{virtual}} + X_i - \delta_{\text{adaptation}} \quad (1)$$

- If the sender needs to reduce its transmission rate by  $\{X_r : X_r > \delta_{\text{adaptation}}\}$  then he only reduces its transmission rate by  $\delta_{\text{adaptation}}$  and  $B_{\text{virtual}}$  is set to

$$B_{\text{virtual}} = B_{\text{virtual}} - X_r + \delta_{\text{adaptation}} \quad (2)$$

- If the sender needs to reduce its transmission rate by an amount  $X_r$  due to a loss ( $l$ ) lower than the allowed loss value the flow can ignore or repair by himself  $\{l : l < l_{\text{allowed}}\}$  or the reduction would lead to a transmission rate  $\{r : r < r_{\text{min}}\}$  then he does not reduce its transmission rate and  $B_{\text{virtual}}$  is set to

$$B_{\text{virtual}} = B_{\text{virtual}} - X_r \quad (3)$$

An optimal adaptation behavior is realized with  $B_{\text{virtual}} \approx 0$ . To achieve such a behavior, in CTFAF, we additionally adjust the decrease and increase behavior of the adaptation algorithm to consider  $B_{\text{virtual}}$  and hence reduce the effects of temporarily unfair distribution as fast as possible.

**Overload situation:** For the case of network congestion the sender should reduce its current transmission rate by  $X_r$ .

Depending on the value of  $B_{\text{virtual}}$  the sender behavior is adjusted as follows:

$B_{\text{virtual}} > 0$ : A positive  $B_{\text{virtual}}$  indicates that this sender was too conservative in the past. As a compensation the sender can ignore the loss situation and maintain its current transmission rate.  $B_{\text{virtual}}$  is then reduced by the amount the rate should have been reduced by:

$$B_{\text{virtual}} = B_{\text{virtual}} - X_r \quad (4)$$

$B_{\text{virtual}} < 0$ : A negative virtual bandwidth indicates on the contrary that the sender has been too aggressive in the past. Therefore, for the case of  $X_r < \delta_{\text{adaptation}}$  the rate is reduced by  $\delta_{\text{adaptation}}$  and  $B_{\text{virtual}}$  can be increased as follows:

$$B_{\text{virtual}} = B_{\text{virtual}} + \delta_{\text{adaptation}} - X_r \quad (5)$$

**Underload situation:** For the case of network underload the sender would be allowed to increase its transmission rate by  $X_i$ . To compensate a negative virtual bandwidth the sender maintains in this case its current transmission rate. Thereby  $B_{\text{virtual}}$  is changed as follows:

$$B_{\text{virtual}} = B_{\text{virtual}} - X_i \quad (6)$$

Recent values of  $B_{\text{virtual}}$  bear more significance in terms of characterizing the friendliness or aggressiveness of a flow towards competing traffic. Hence, to avoid having some value of  $B_{\text{virtual}}$  propagating over long time periods and influencing the behavior of the adaptation scheme even though the network situation that has resulted in this value has changed considerably we refer to resetting  $B_{\text{virtual}}$  in periods of  $T_{\text{reset}}$ .  $T_{\text{reset}}$  is several times as large as the adaptation intervals ( $T_{\text{adaptation}}$ ), i.e., the time between two adaptation actions. At the end of each resetting period ( $T_{\text{reset}}$ ) the sender changes its transmission rate by an averaged  $B_{\text{virtual}}$  ( $\overline{B_{\text{virtual}}}$ ) and resets  $B_{\text{virtual}}$  down to zero. The averaged  $B_{\text{virtual}}$  is determined as

$$\overline{B_{\text{virtual}}} = B_{\text{virtual}} \times \frac{T_{\text{adaptation}}}{T_{\text{reset}}} \quad (7)$$

As  $B_{\text{virtual}}$  is changed at each adaptation point,  $\overline{B_{\text{virtual}}}$  represents an averaged value of all additions and subtractions done during  $T_{\text{reset}}$ .

Using a ( $\overline{B_{\text{virtual}}} : \overline{B_{\text{virtual}}} > \delta_{\text{adaptation}}$ ) contradicts the requirement of limiting the maximum changes in the transmission rate to  $\delta_{\text{adaptation}}$ . However, we found this approach to be necessary in order to adjust the performance of a flow in accordance with the available resources. Without this resetting behavior, a sender transmitting data at a rate higher than the capacity of some link on the path towards the receiver and specifying a small  $\delta_{\text{adaptation}}$  compared to the needed change in the transmission rate would cause high and long lasting congestion. For the case that the flow is using a bandwidth share close to the average share calculated with the adaptation scheme,  $\overline{B_{\text{virtual}}}$  would have a small value and would not have considerable effects on the adaptation behavior. Instead of using a large change of ( $\overline{B_{\text{virtual}}} > \delta_{\text{adaptation}}$ ) another approach would be for the sender to change its transmission rate by  $\delta_{\text{adaptation}}$  each  $T_{\text{adaptation}}$  until  $B_{\text{virtual}}$  reaches 0 again.

Without prior knowledge of the available resources a sender might start transmitting data at a rate much higher than its fair share. If the sender additionally specified a slow maximum changing rate  $\delta_{\text{adaptation}}$  the sent flow would lead to an overload situation lasting at least till the end of the first resetting period ( $T_{\text{reset}}$ ). As  $T_{\text{reset}}$  might be in the range of one or more minutes relying solely on the resetting actions would take too long. To overcome this situation we specify that for a transient phase lasting for a maximum period of  $T_{\text{init}}$  the sender applies the adaptation scheme restricted only by the encoders limitations and not the user specifications. That is, for the first  $T_{\text{init}}$  seconds the sender can adapt its transmission rate by any number of adaptation steps ( $S_{\text{adaptation}}$ ) as dictated by the adaptation scheme without regard to  $\delta_{\text{adaptation}}$ .

Note that the proposed guidelines here are only one option for accommodating user and media constraints into the adaptation process. For example, resetting  $B_{\text{virtual}}$  periodically is only one possible approach for preventing  $B_{\text{virtual}}$  accumulated during past periods of times to have long lasting effects on the adaptation process. Other options might be to use weighted moving averages or requiring that flows having a negative  $B_{\text{virtual}}$  value higher than some maximum value must reduce their rate more often and by larger amounts than indicated by the adaptation algorithm and user preferences. The exact strategies to use as well as their parameters will depend on the communication scenario.

## 4 Increase/Decrease Adaptation Schemes

In general, congestion control schemes are based on collecting information about the resource availability in the network and adjusting the transmission rate of the sender based on this information. With the lack of mechanisms for collecting and distributing explicit information about the resource shares the flows can utilize in the Internet, most of the control schemes found in the literature refer to heuristic approaches for adjusting the transmission rate based on information about losses and delays in the network as observed by the end systems. Such approaches are usually based on some variation of an increase/decrease mechanism (ID). With ID schemes, the bandwidth share of a flow is gradually increased by some amount during underload situations and reduced during overload situations. A popular version of ID schemes is the additive increase and multiple decrease mechanism (AIMD) which is also the basis for TCP. With this approach the sender can increase its resource share by an additive amount during underload periods and reduces it by a multiplicative factor during overload periods. The choice of additive increase and multiplicative decrease can be briefly justified as follows. If the network is operating below the optimal point designated by high utilization and low losses, all users go up equally. If the network is congested, the multiplicative decrease makes users with higher resource share go down more than those with smaller shares making the allocation more fair. However, with no information about the explicit share to use, ID schemes do not converge to a single steady state. Instead, the system reaches an equilibrium (steady state) in which it oscillates around the optimal state [5]. The time taken to reach this equilibrium (responsiveness or transient state) and the size of the oscillations (smoothness) jointly determine the convergence. Ideally, the time as well as the oscillation should be small, see Fig. 1 [5]. An an exam-

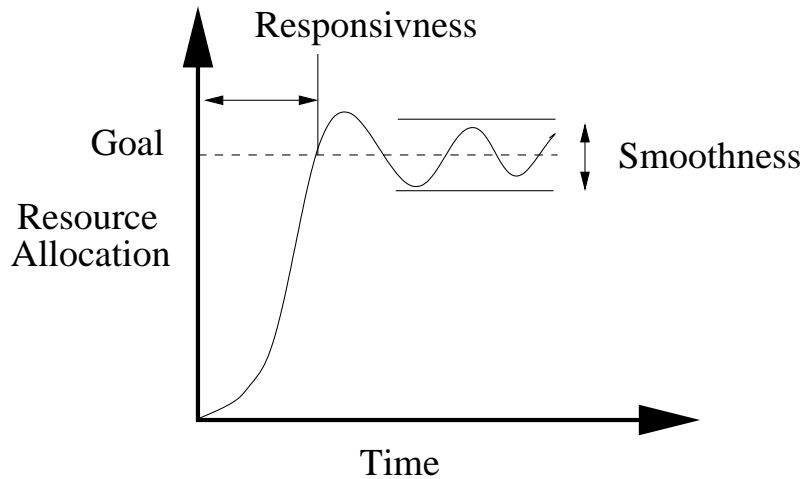


Figure 1: Responsiveness and smoothness of increase/decrease schemes

ple for a TCP-friendly adaptation scheme based on the increase/decrease concept, we describe here an adaptation algorithm that is based on previous work in [20, 21] called the enhanced loss-delay based adaptation scheme (LDA+).

## 4.1 The Enhanced Loss-Delay Based Adaptation Algorithm

With LDA+ a receiver collects loss and delay information about the incoming data and sends the collected data to the sender. Based on these information the sender increases its transmission rate by a dynamically determined additive increase rate ( $A$ ) during underload situations and reduces it otherwise.

### 4.1.1 Feedback Information

In contrast to other adaptation schemes such as [13, 16] that introduce a new protocol for establishing the flow of feedback messages from the receiver to the sender, the enhanced loss-delay based adaptation algorithm (LDA+) relies on the real time transport protocol (RTP) [18].

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 (RTCP), each member of a multicast session periodically sends control reports to all other members containing information about sent and received data. Additionally, the end systems might include in their reports an application specific part (APP) intended for experimental use. The RTCP traffic is scaled with the data traffic so that it makes up a certain percentage of the data rate (usually 5%) with a minimum interval of 5 seconds between sending two RTCP messages. RTCP messages include information about the losses and delays noticed in the network. In addition, RTCP was enhanced with the ability to estimate the bottleneck bandwidth of a flow based on the packet pair approach first used by Bolot [1] for estimating the characteristics of data networks. The essential idea behind this approach is: If two packets can be caused to travel together such that they are queued as a pair at the bottleneck, with no packets intervening between them, then the inter-packet spacing will be proportional to the time required for the bottleneck router to process the second packet of the pair. In this context, a bottleneck router indicates the router with the smallest capacity along the path traversed by the flow.

The RTCP sender reports were enhanced by an additional application defined part (APP) that includes the source sequence number, the sequence number (SEQ) of a data packet that will start a stream of probe packets and the number ( $n$ ) of probe packets that will be sent. Then,  $n$  data packets starting with packet numbered SEQ are sent at the access speed of the end system. At the receiver site, the bottleneck bandwidth ( $R$ ) is calculated as:

$$R = \frac{\text{probe packet size}}{\text{gap between 2 probe packets}} \quad (8)$$

Using data packets as probe packets restricts the additional bandwidth required for the bottleneck probing to the increased size of the RTCP packets. The choice of running the measurement of the bottleneck bandwidth after the sending of each RTCP sender report or every few ones should be left to the application.

Due to losses of probe packets, network congestion or interference of other traffic Eqn. 8 might result in wrong estimations of the bottleneck bandwidth. Hence, the receivers need to deploy mechanisms such as [11, 15, 4] to filter out wrong estimates. Which filtering mechanism to use in the framework of LDA+ is irrelevant, whereas schemes resulting in accurate estimations after short transient periods are to be favored.

## 4.2 Rate Adjustment with LDA+

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 consumed by a TCP connection under similar loss and delay values as described in [12]. 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.

In the detail, after receiving the  $m$ th. receiver report the sender estimates the bandwidth share ( $r_m$ ) it should be using as follows:

**No loss situation:** In this case, the sender can increase its estimation of its TCP-friendly bandwidth share by an additive increase rate ( $A$ ). To allow for a smooth increase of  $A$  and to allow flows of smaller bandwidth shares to faster increase their transmission rates than competing flows with higher shares,  $A$  is determined in dependence of the bandwidth share ( $r_{m-1}$ ) the sender is currently consuming relative to the bottleneck bandwidth ( $R$ ) of the path connecting the sender to the receiver. Thus with an initial transmission rate of ( $r_0$ ), an initial additive increase value of  $\dot{A}$ ,  $A$  would evolve as follows:

$$A_{\text{add}_1} = \dot{A} + \left(1 - \frac{r_0}{R_i}\right) \times \dot{A} \quad (9)$$

$$A_{\text{add}_m} = A_{m-1} + \left(1 - \frac{r_{m-1}}{R}\right) \times A_{m-1} \quad (10)$$

Both  $r_0$  and  $\dot{A}$  are set by the user but should be kept small relative to the bottleneck bandwidth. To limit the rate increase maximally to the bottleneck bandwidth a second value of  $A$  is determined, that converges to 0 as the bandwidth share of the flow converges to the bottleneck bandwidth. One function that fulfills this requirement is the exponential function in the form of

$$A_{\text{exp}_m} = \left(1 - \exp^{-\left(1 - \frac{r_{m-1}}{R}\right)}\right) \times r_{m-1} \quad (11)$$

Finally, an RTP flow should not increase its bandwidth share faster than a TCP connection sharing the same link. With an average value of  $T$  seconds between the reception of two receiver reports and a round trip delay of ( $\tau$ ) a TCP connection would increase its transmission window by  $P$  packets with  $P$  set to

$$P = \sum_{q=0}^{T/\tau} q = \frac{\left(\frac{T}{\tau} + 1\right) \times \frac{T}{\tau}}{2} \quad (12)$$

with the window size being increased by one packet each round trip delay. With a packet size of  $M$  and averaged over  $T$ , the RTP receiver should maximally increase its estimation of its bandwidth share by

$$A_{\text{TCP}_m} = M \times \frac{P}{T} \rightarrow M \times \frac{\frac{T}{\tau} + 1}{2 \times \tau} \quad (13)$$

The additive increase value ( $A_m$ ) is then set to

$$A_m = \min(A_{\text{add}_m}, A_{\text{exp}_m}, A_{\text{TCP}_m}) \quad (14)$$

Finally, the receiver determines  $r_m$  as

$$r_m = r_{m-1} + A_m \quad (15)$$

**Loss situation:** In this case, the sender needs to reduce its transmission rate minimally to the average rate of a TCP connection having the same loss and delay values. Padhye et al. present in [12] an analytical model for determining the average transmission rate of a TCP sender ( $r_{\text{TCP}}$ ) under known loss ( $l$ ) and round trip delays ( $t_{\text{RTT}}$ ).

$$r_{\text{TCP}} = \frac{M}{t_{\text{RTT}} \sqrt{\frac{2Dl}{3}} + t_{\text{out}} \min\left(1, 3\sqrt{\frac{3Dl}{8}}\right) l (1 + 32l^2)} \quad (16)$$

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

For the case of losses, this model suggests that TCP adjusts its transmission rate inversely proportional to the square root of the losses. Thus, if losses ( $l$ ) were indicated in the RTCP messages then the transmission rate ( $r_m$ ) is reduced to:

$$r_m = \max(r_{m-1} \times (1 - \sqrt{l}), r_{\text{TCP}}) \quad (17)$$

with  $r_{\text{TCP}}$  as the rate calculated using Eqn. 16. Additionally,  $A$  is reset to  $\dot{A}$ . In case the current transmission rate is already lower than  $r_{\text{TCP}}$  the sender is allowed to further increase its transmission rate up to  $r_{\text{TCP}}$ .

The work done in [20] as well as the extensive simulations and measurements reported in [19, 21] suggest LDA+ to be efficient in terms of bandwidth utilization and loss reduction as well as being friendly to competing TCP-traffic. A sample of those simulations describing the TCP-friendliness aspects of LDA+ are presented in appendix A.

## 5 Performance Investigation of CTFAF

In this part of the work, we investigate the effects of imposing user- or encoder-based restrictions on the adaptation process in terms of buffer occupancy and bandwidth utilization. For this purpose we compare the performance of LDA+ without any restrictions, i.e., ( $\delta_{\text{adaptation}} = \infty$ ) and its performance when the rate adjustments are controlled not only by network conditions but by the CTFAF guidelines described in Sec. 3 as well.

For investigating the performance of CTFAF we used the simulation topology depicted in Fig. 2 with FTP, WWW and multimedia flows competing for a bottleneck router of 10 Mb/s and ( $m = n = k = 27$ ). The FTP and multimedia flows were modeled as greedy sources that can always send data at the rate allowed by the congestion control scheme, i.e., the TCP mechanisms for the case of FTP and LDA+ for the case of multimedia flows. The WWW servers were modeled as on-off processes

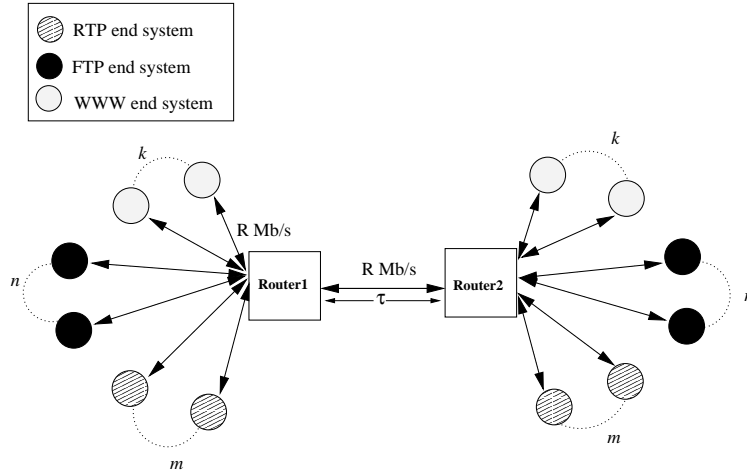


Figure 2: CTFAF performance testing topology

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 [14]. The routers used random early packet discard (RED) [7] buffer management. 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 connections receive the same loss ratio and avoids synchronization among the competing flows. This is of utmost importance in this study in order to properly evaluate the fairness aspects of the investigated adaptation schemes on connections having the same loss. However, the performance of the here presented solutions does not depend on the usage of RED routers. Based on results achieved in [8] the minimum drop threshold was set to 0.3 of the route buffer and the maximum to 0.8. Unless otherwise stated, the round trip propagation delay was set to 0.4 seconds. The initialization period ( $T_{init}$ ) was set to 60 seconds. We tested the performance of CTFAF in terms of TCP-friendliness and the standard deviation of the transmission rate of the single flows as an indication of the stability of the used transmission rate. The performance of CTFAF was tested in terms of achieved average rate ( $r$ ), fairness factor ( $F$ ), average standard deviation ( $\sigma$ ) and average utilization value ( $u$ ). The average rate ( $r$ ) was determined over the entire simulation time minus the transient time which was approximated in this case to the first 200 seconds. The fairness index ( $F_i$ ) was determined as the ratio of the bandwidth share of the CTFAF flows to the bandwidth share consumed by the TCP connections.  $\sigma$  indicates the average of the deviation value of the temporal rate values of each CTFAF flow from the average rate of that flow. The temporal values were measured over intervals of one second.

## 5.1 Effects of the Length of the Adaptation Steps ( $S_{\text{adaptation}}$ ) on the Performance of CTFAF

As an example for a multimedia flow we simulated the case of video flows with constant quantization steps of 2 kbits, packet sizes of 10 kbits and frame sizes of 20 kbits. We tested the case of coarse and fine granularity adaptation. Using ( $S_{\text{adaptation}} = 20$  kb/s) resembles the case of changing the transmission rate by varying the number of sent video frames when using JPEG compression for example. We set  $\delta_{\text{adaptation}}$  in this case to 1 and 2 frames/s, i.e., 20 and 40 kb/s. With ( $S_{\text{adaptation}} = 2$  kb/s) the adaptation is more granular and simulates the case of varying the quantization or quality factor.  $\delta_{\text{adaptation}}$  is set in this case to 4 kb/s. As a minimum transmission rate we use a value of 40 kb/s. The parameter settings for the video coder were chosen based on observations made during a talking heads conference with JPEG as the compression style. For another communication scenario or a different compression style we would most likely have to use a different set of parameters. Refining the video model and choosing the appropriate parameters requires detailed knowledge of the used compression style and communication scenario. Additionally, to account for the user preferences extensive subjective testing is needed. The resetting period ( $T_{\text{reset}}$ ) was set to 60 seconds. From Tab. 1 we can observe that while the standard deviation which indicates the

$\delta_{\text{adaptation}}$ (kb/s)	r (kb/s)	$\sigma$ (kb/s)	$F_i$	$u$
$\infty$	134	44	0.8	0.81
40	137	45	0.84	0.80
20	132	37	0.80	0.79
4	140	29	0.85	0.81

Table 1: Achieved average rate and fairness of CTFAF with different values of  $\delta_{\text{adaptation}}$

level of oscillations of the flows is reduced for finer setting of  $\delta_{\text{adaptation}}$  the utilization and fairness levels are hardly changed compared to the case of no restrictions on the adaptation ( $\delta_{\text{adaptation}} = \infty$ ).

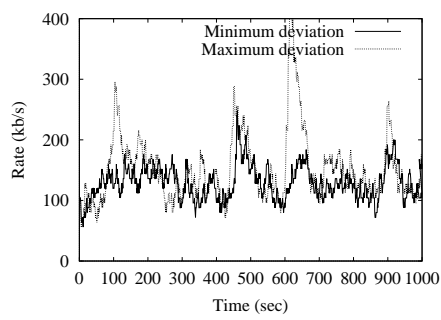
The reduction of oscillations is particularly evident in Fig. 3 which shows the temporal behavior of the flows with the largest and smallest standard deviation ( $\sigma$ ) values. For the case of ( $\delta_{\text{adaptation}} = 4$  kb/s) the flows have a much smoother shape compared to the other cases.

The fairness results presented in this section suggest that while CTFAF maintains the TCP-friendliness of LDA+, it can successfully take various constraints imposed by the multimedia flow into account.

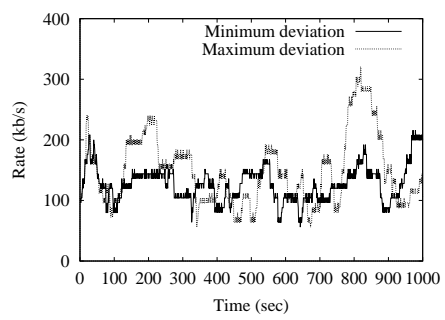
## 5.2 Effects of the Length of the Resetting Period ( $T_{\text{reset}}$ ) on the Performance of CTFAF

The length of the resetting period ( $T_{\text{reset}}$ ) determines the interval in which  $B_{\text{virtual}}$  is reset and the bandwidth share of the flow is corrected possibly faster than the user specified adaptation rate ( $\delta_{\text{adaptation}}$ ). Setting  $T_{\text{reset}}$  too small would mean that this correction is done rather often and would, thus, defy our goal of restricting the adaptation rate to  $\delta_{\text{adaptation}}$ . Using too large values of  $T_{\text{reset}}$  would on the other hand mean that the flow can not adapt rapidly enough to large changes in the network conditions.

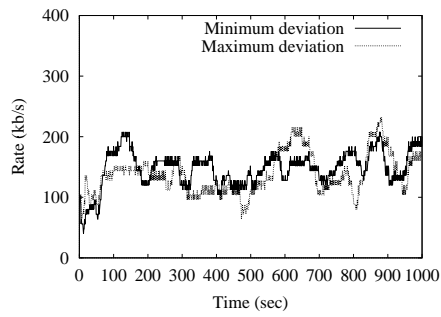
Here, we simulated the case of competing WWW, FTP and CTFAF flows over a link of 10 Mb/s with a round trip propagation delay of 0.4 seconds and a maximum queuing delay of 0.1 seconds.



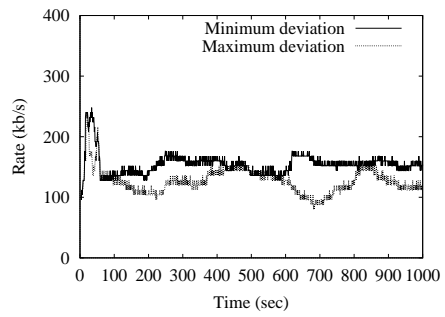
(a)  $\delta_{\text{adaptation}} = \infty \text{ kb/s}$



(b)  $\delta_{\text{adaptation}} = 40 \text{ kb/s}$



(c)  $\delta_{\text{adaptation}} = 20 \text{ kb/s}$



(d)  $\delta_{\text{adaptation}} = 4 \text{ kb/s}$

Figure 3: Temporal behavior of CTFAF flows with the maximum and minimum standard deviation values for different values of  $\delta_{\text{adaptation}}$

The CTFAF flows used adaptation steps ( $S_{\text{adaptation}}$ ) of 2 kb/s and the maximum possible rate change at each adaptation point ( $\delta_{\text{adaptation}}$ ) was set to 4 kb/s. These adaptation values impose rather strict constraints on the adaptation process and a better distinction to the case of adapting the transmission rate without any restrictions ( $\delta_{\text{adaptation}} = \infty$ ).

$T_{\text{reset}}$ (sec)	$r$ (kb/s)	$\sigma$ (kb/s)	$F_i$	$u$
30	137	30	0.81	0.82
60	140	29	0.85	0.81
120	134	26	0.83	0.81
300	142	25	0.86	0.83

Table 2: Achieved average rate and fairness of CTFAF with different values of  $T_{\text{reset}}$

The fairness and utilization results in Tab. 2 show little variance with varying values of  $T_{\text{reset}}$ . The average standard deviation values ( $\sigma$ ) presented in Tab. 2 as well as the temporal behavior of the flows with the best and worst deviation values in Fig. 4 suggest that with increased values of  $T_{\text{reset}}$  the flows display a lower degree of oscillations. However, with large values of  $T_{\text{reset}}$  the flows can not react to large and sudden changes in the network conditions such as when several new flows start sending data on the same link as the CTFAF flows. Hence, setting  $T_{\text{reset}}$  to 60 or 120 seconds presents a better compromise between stability and fast reaction than a value of 300 seconds for example.

### 5.3 Effects of Delays and Flow Numbers on the Performance of CTFAF

In this part, we investigate the performance of CTFAF in terms of bandwidth utilization ( $u$ ), average bandwidth share of single CTFAF flows ( $r$ ) and its fairness towards competing TCP connection ( $F_i$ ) under different test settings with varying numbers of competing flows and delays. As a simulation topology we used the one described in Fig. 2 with a varying number of flows competing for a bottleneck of 10 Mb/s. In addition to  $n$  FTP and  $m$  CTFAF flows, 27 WWW servers continuously generated short TCP connections.

Tab. 3 presents the simulation results for the case of varying number of competing flows ( $m = n = N$ ) and round trip propagation delays ( $\tau$ ). From Tab. 3 we can observe that the average standard deviation of the flows ( $\sigma$ ) is around than 20% of the average bandwidth of the flows. Additionally, the deviation ( $\bar{\sigma}$ ) in the average rate values of the single flows from the average bandwidth share of all the flows is rather negligible indicating a high degree of inter-protocol fairness with all the flows receiving similar bandwidth shares. Except for the case of large propagation delays of one second, the fairness index ( $F_i$ ) is in an acceptable range between 0.8 and 1.15 indicating similar bandwidth shares for the TCP and CTFAF flows.

Fig. 5 describes the bandwidth share obtained by the LDA+ flows with the maximum and minimum standard deviation ( $\sigma$ ). Comparing the behavior of the CTFAF flows to the LDA+ flows under the same conditions as depicted in Fig. 7 we can observe that using CTFAF shows a considerable improvement in terms of flow stability as requested by the small value of  $\delta_{\text{adaptation}}$  while maintaining similar performance in terms of TCP-friendly bandwidth distribution.

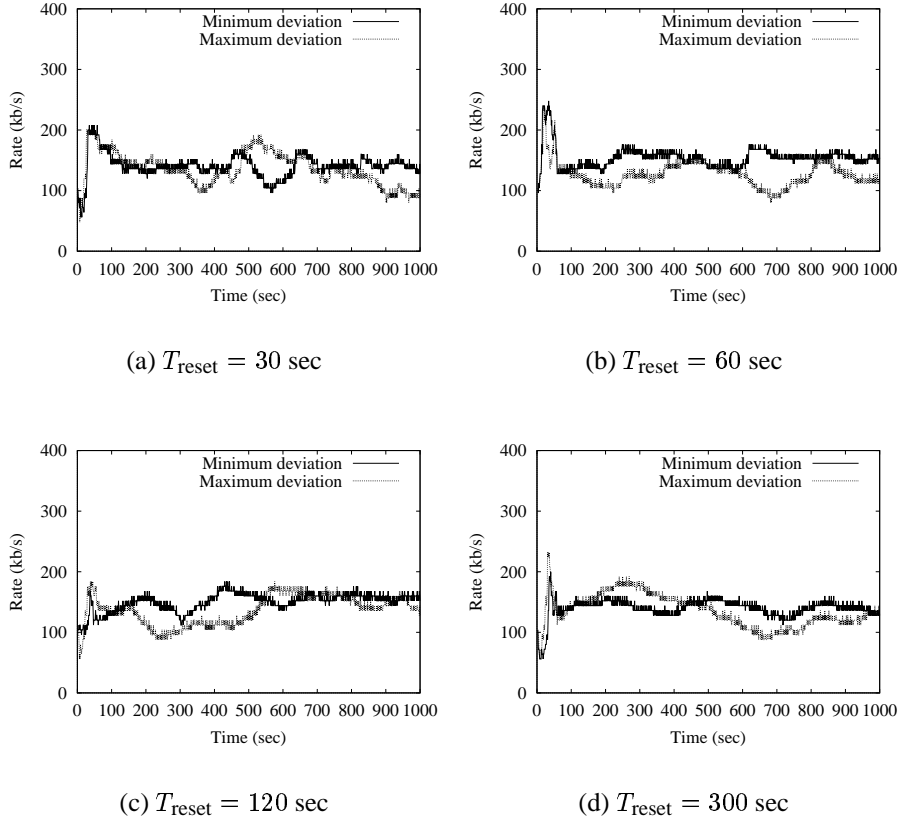


Figure 4: Temporal behavior of CTFAF flows with the maximum and minimum of  $\sigma$  for different values of  $T_{\text{reset}}$

$N$	27					54					81				
$\tau$ (sec)	$r$	$\sigma$	$\bar{\sigma}$	$F_i$	$u$	$r$	$\sigma$	$\bar{\sigma}$	$F_i$	$u$	$r$	$\sigma$	$\bar{\sigma}$	$F_i$	$u$
0.1	160	33	7.8	0.93	0.87	99	22	5.5	1.17	0.91	75	19	4	1.3	0.94
0.2	130	27	7.5	0.71	0.83	82	20	3.4	0.87	0.9	62	16.1	4.1	1	0.92
0.4	140	29	13	0.84	0.80	76	18	3.5	0.79	0.90	55	15	2.1	0.86	0.93
0.6	169	32	16	1.14	0.85	80	20	7	0.91	0.89	54	13.9	5	0.83	0.92
1.0	234	35	17	2.47	0.89	105	22	10	1.6	0.9	65	18	6.2	1.2	0.92

Table 3: Achieved average rate ( $r$ ) in kb/s, deviation ( $\sigma, \bar{\sigma}$ ) in kb/s, utilization ( $u$ ) and fairness of CTFAF with  $N$  TCP and  $N$  CTFAF competing flows,  $R$  set to 10 Mb/s and  $\tau_q$  set to 0.1 seconds

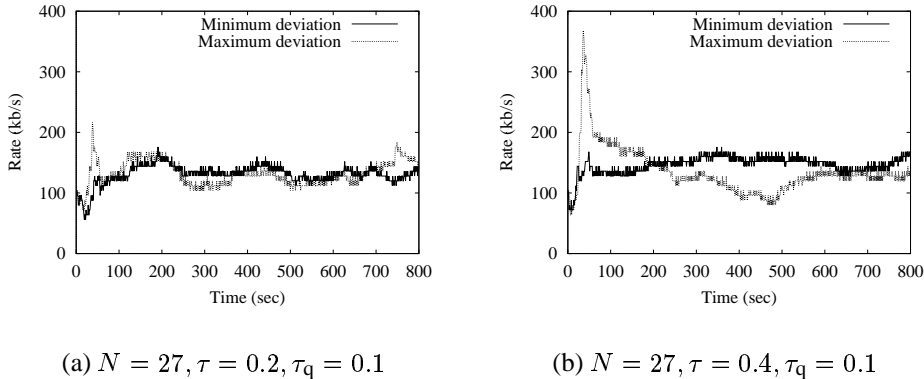


Figure 5: Temporal behavior of CTFAF

## 6 Summary and Future Work

In this paper, we presented a novel approach for incorporating user and coder imposed restrictions on the adaptation behavior with a TCP-friendly adaptation algorithm used for controlling the transmission of multimedia flows. The achieved simulation results suggest that the used model and adaptation approach maintain the TCP-friendliness characteristics of the original algorithm and achieve a stable adaptation behavior that corresponds to the parameters imposed by the coder and the user.

While the work presented here gives a possible approach for adapting video flows, there are still many open questions regarding the exact tuning of the adaptation parameters, incorporating different kinds of compression styles and the actual implications of such an adaptation behavior on the perceived quality. This work should be seen as work in progress that requires simulating a wider range of possible parameters as well as assessing the changes in perceived quality of service on real life multimedia streams using series of subjective tests. Further, the case of multicast with heterogeneous receiver requirements and capabilities needs to be studied.

## 7 Acknowledgments

The RTP/RTCP simulation models were mainly implemented by Timur Friedman and improved by Frank Emanuel.

## A Performance Tests of LDA+

For testing the performance of LDA+ we used the same test topology as in Fig. 2 with  $n$  TCP connections sharing a single bottleneck with  $m$  UDP flows. The round trip delay ( $\tau$ ) was set to 0.04 seconds and the packet size was set to 100 bytes. The router used FIFO buffer management with the buffer length set to  $(4 \times R \times \tau)$ .

Fig. 6(a) depicts the fairness results of a test setting with the bottleneck bandwidth ( $R$ ) set to  $(2 \times N \times 40 \text{ kb/s})$  for a varying number of flows. The fairness index is determined as the average

goodput of the LDA+ flows to the average goodput of the TCP connections. LDA+ achieves in this case a fairness index of around one indicating a high degree of TCP-friendliness.

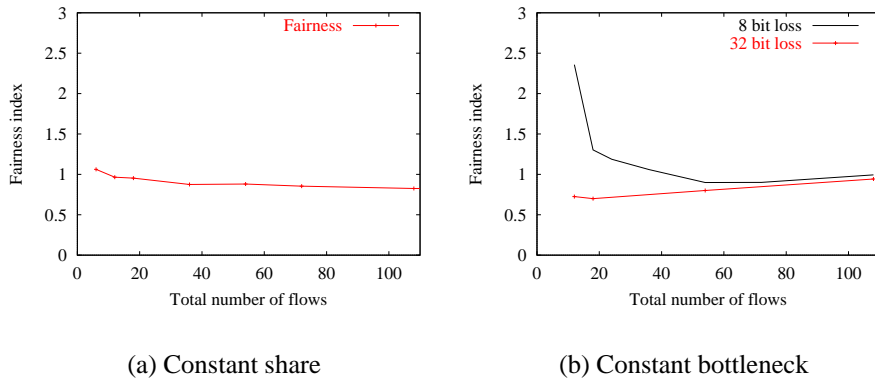


Figure 6: Fairness index with LDA+

Fig. 6(b) depicts the fairness results for the case of a constant bottleneck bandwidth of 1.5 Mb/s. For higher numbers of competing flows, LDA+ achieves a fairness index of around one indicating its TCP-friendliness in cases of higher network load. However, for the case of a lower number of competing flows a fairness index higher than one is achieved indicating that the LDA+ flows receive a higher bandwidth share than the TCP connections. This is especially evident for the case of six LDA+ flows competing with six TCP connections. In this case, a fairness index of 2.3 is achieved indicating that the LDA+ flows receive on the average twice as much bandwidth as the TCP connections. However, this is not caused by the adaptation algorithm itself but by the way loss information are transported in RTP. The receiver reports use an eight bit field for the loss information and hence the smallest loss information that can be indicated is approximately 0.004. However, under the used topology with 0.04 seconds round trip delay and a bottleneck of 1.5 Mb/s to be divided among 12 competing flows, using Eqn. 16 indicates that the average loss rate each flow needs to suffer in order for it to get a bandwidth share of  $(\frac{1.5}{12} = 0.125\text{Mb/s})$  is around 0.002. In this case, the receivers round the loss value down to 0 and the RTP flows can increase their rate in situations where they are supposed to reduce their transmission rate. Fig. 6(b) depicts the fairness results achieved when using a 32 bit field in the receiver reports for loss indications instead of eight bits. In this case, we can observe that LDA+ is rather conservative and achieves for the case of smaller numbers of competing flows a fairness index of less than one. For the case of higher numbers of competing flows the behavior of LDA+ is identical for both loss representations.

## A.1 Performance of LDA+ in the Presence of WWW Traffic

The simulations in the previous section assumed senders with infinite amount of data for transmission. However, a considerable amount of the traffic in the Internet currently consists of small burst WWW connections [6]. To test the effects of the presence of bursty traffic on the performance of LDA+ we used the topology depicted in Fig. 2 with  $(m = n = k = N = 27)$ , the packet size set to 1 kbyte and the bottleneck bandwidth set to 10 Mb/s. The router used RED buffer management.

Tab. 4 presents the simulation results with varying round trip delays ( $\tau$ ) and buffering delays ( $\tau_q$ ).

$\tau_q$ (sec)	0.10					0.50				
$\tau$ (sec)	r	$\bar{\sigma}$	$F_i$	l	u	r	$\bar{\sigma}$	$F_i$	l	u
0.1	155.95	4.07	0.85	0.046	0.87	148.59	7.30	0.82	0.019	0.87
0.2	134.74	4.91	0.72	0.033	0.84	156.35	10.34	0.88	0.015	0.88
0.4	134.09	8.56	0.78	0.021	0.80	168.66	15.12	1.02	0.012	0.88

Table 4: Achieved average rate and fairness for LDA+ with 27 TCP and 27 LDA+ competing flows and R set to 10 Mb/s

Fig. 7 describe the bandwidth share obtained by the LDA+ flows with the maximum and minimum standard deviation ( $\sigma$ ).  $\sigma$  indicates the deviation of the temporal rate values of those flows from the average rate of the flow during the steady state. The temporal values were measured over intervals of one second.

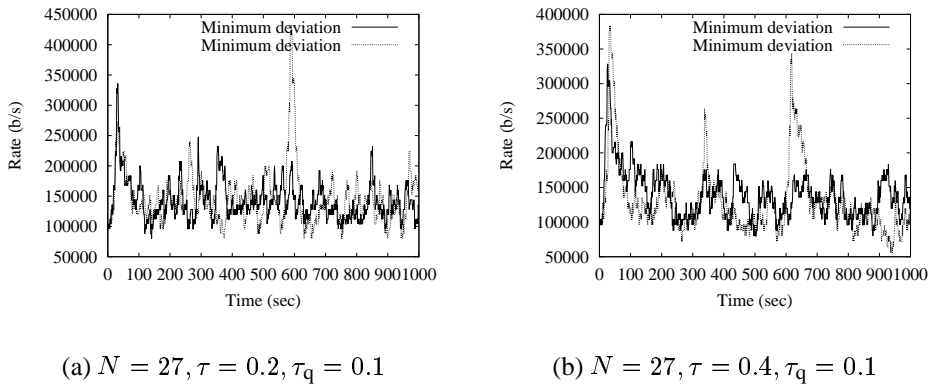


Figure 7: Temporal behavior of LDA+

## References

- [1] J.-C. Bolot. End-to-end packet delay and loss behavior in the Internet. In D. P. Sidhu, editor, *SIGCOMM Symposium on Communications Architectures and Protocols*, pages 289–298, San Francisco, California, Sept. 1993. ACM. also in *Computer Communication Review* 23 (4), Oct. 1992.
- [2] B. Braden and L. Zhang. Resource ReSerVation protocol (RSVP) – version 1 message processing rules. Request for Comments (Proposed Standard) 2209, Internet Engineering Task Force, Oct. 1997.
- [3] R. Braden, D. Clark, and S. Shenker. Integrated services in the internet architecture: an overview. Technical Report RFC 1633, Internet Engineering Task Force, June 1994.
- [4] R. L. Carter and M. E. Crovella. Measuring bottleneck link speed in packet-switched networks. Technical Report BU-CS-96006, Computer Science Department, Boston University, Mar. 1996.
- [5] D. Chiu and R. Jain. Analysis of the increase/decrease algorithms for congestion avoidance in computer networks. *Journal of Computer Networks and ISDN*, 17(1):1–14, June 1989.

- [6] K. Claffy and G. Miller. The nature of the beast: Recent traffic measurements from an Internet backbone. In *INET '98*, Palexpo Konferenz Zentrum, Genf, July 1998. Internet Society.
- [7] S. Floyd and V. Jacobson. Random early detection gateways for congestion avoidance. *IEEE/ACM Transactions on Networking*, 1(4):397–413, Aug. 1993.
- [8] S. Floyd and F. Kevin. Router mechanisms to support end-to-end congestion control. Technical report, LBL-Berkeley, Feb. 1997.
- [9] M. Handely and S. Floyd. Strawman specification for TCP friendly reliable multicast congestion control, 1998. Note to the Internet Reliable Multicast Group mailing list.
- [10] S. Jamin. *A measurement-based admission control algorithm for integrated services packet networks*. PhD thesis, Dept. of Computer Science, University of Southern California, Los Angeles, California, Aug. 1996.
- [11] K. Lai and M. Baker. Measuring bandwidth. In *Proceedings of the Conference on Computer Communications (IEEE Infocom)*, New York, USA, Mar. 1999.
- [12] 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.
- [13] 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.
- [14] K. Park, G. Kim, and M. 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.
- [15] V. Paxson. *Measurements and Analysis of End-to-End Internet Dynamics*. PhD thesis, Lawrence Berkeley National Laboratory, University of California, Berkeley, California, Apr. 1997.
- [16] R. Rejaie, M. Handley, and D. Estrin. An end-to-end rate-based congestion control mechanism for realtime streams in the Internet. In *Infocom'99*, New York, March 1999. IEEE.
- [17] H. Schulzrinne. Re-engineering the telephone system. In *Proc. of IEEE Singapore International Conference on Networks (SICON)*, Singapore, Apr. 1997.
- [18] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson. RTP: a transport protocol for real-time applications. Technical Report RFC 1889, Internet Engineering Task Force, Jan. 1996.
- [19] D. Sisalem. A TCP-friendly adaptation scheme based on RTP. Technical report, GMD, Fokus, Germany, Apr. 1999.
- [20] D. Sisalem and H. 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.
- [21] D. Sisalem and A. Wolisz. Towards TCP-friendly adaptive multimedia applications based on RTP. In *Fourth IEEE Symposium on Computers and Communications (ISCC'99)*, Egypt, July 1999.
- [22] W. R. Stevens. *TCP/IP Illustrated: The Protocols*, volume 1. Addison-Wesley, Reading, Massachusetts, USA, Dec. 1993.