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

Dorgham Sisalem, Frank Emanuel
GMD-Fokus, Berlin
{sisalem,emanuel}@fokus.gmd.de

Henning Schulzrinne
Columbia University, New York
schulzrinne@cs.columbia.edu

August 6, 1997

Abstract

Many distributed multimedia applications have the ability to adapt to fluctuations in the network conditions. By adjusting temporal and spatial quality to available bandwidth, or manipulating the playout time of continuous media in response to variations in delay, multimedia flows can keep an acceptable QoS level at the end systems. In this study, we present a new scheme for adapting the transmission rate of multimedia applications to the congestion level of the network. The scheme called the direct adjustment algorithm (DAA), is based on the TCP congestion control mechanisms and relies on the end-to-end Real Time transport Protocol (RTP) for feedback information. Our investigations of the DAA scheme suggest the efficiency of the scheme in utilizing network resources and decreasing loss ratios. Also, the scheme is shown to be fair towards competing TCP traffic. However, with no support from the network, long distance connections receive less than their fair share.

1 Introduction

A large part of the multimedia applications used currently in the Internet, such as VIC [21], VAT [19], NEVOT [2] and NEVIT [30] are based on the UDP and RTP [24] transport protocols. However, both UDP and RTP offer no quality of service control mechanisms and can therefore not guarantee any level of QoS. Fluctuations of network conditions combined with the inability of those protocols to support QoS control often renders multimedia applications useless.

Two main approaches were introduced in the past to overcome this problem: over-provisioning through resource reservation [4] and application control [9]. Reserving enough resources for supporting a certain QoS level in advance guarantees this quality level and would be the most straightforward approach for handling the problem. However, as it is usually impossible to know the exact

*This work was funded in part by the BMBF (German Ministry of Education and Research) and the DFN (German Research Network).

characteristics of a stream in advance, one would tend to over-allocate resources to guarantee the requested QoS level and thus leading to network underutilization. This would get costly if one has to pay for the reserved but unused resources.

Many distributed multimedia applications exhibit flexibility to fluctuations in the network conditions [6]. By adjusting the media quality to available bandwidth or manipulating the playout time of continuous media in response to variations in delay, multimedia flows can keep an acceptable QoS level at the end systems without necessarily requiring any additional capabilities in the network. However, such an approach requires the exchange of state information between the source end system and the network as it is the case for the available bit rate service of ATM [25] or between the sender and the receivers as its is the case for TCP [31] and the scheme described here. As these state information have to traverse at least a part of the network, the source will not be able to change its sending rate immediately after a change in the congestion state of the network. So, as a source keeps on sending data with an outdated rate until a reduction request arrives data losses can not be ruled out. However, depending on the importance of the data, the context in which the applications are used and the used codecs losing a small percentage of the data is often a small price to pay in exchange for simple network architectures and low-cost communication.

A negative impact of deploying non-congestion-controlled traffic is the extreme unfairness towards competing TCP traffic. Sending best-effort traffic without any consideration of the network congestion state can easily result in packet losses. In response to that, TCP connections which still constitute the vast majority of the Internet traffic would reduce their transmission windows. However, without any rate reduction on behalf of the non-congestion-controlled traffic, the TCP connections would starve and receive much less than their fair bandwidth shares. Therefore, adaptive schemes should not only aim at reducing loss ratios and improve bandwidth utilization but also need to be fair towards competing TCP connections, i.e, be TCP-friendly.

In this paper, we propose a new end-to-end rate adaptation scheme called the direct adjustment algorithm (DAA) for adjusting the transmission rate of multimedia applications to the congestion level of the network. DAA is based on a combination of two approaches described in the literature, namely: additive increase/multiplicative decrease schemes proposed in [3, 5] and an enhancement of the TCP-throughput model described in [14].

While having applications adapt to network conditions has already been studied in other proposals [3, 5], little has been said about the fairness of this approach, its scalability and usability in heterogeneous environments. Therefore, in this paper we not only investigate the efficiency of DAA in terms of bandwidth utilization but also its fairness behavior towards connections with the same requirements but with different round trip times. Finally, as DAA aims to be TCP-friendly, we investigate the behavior of TCP connections sharing the same congested links with streams using DAA and compare the results with these obtained using another algorithm proposed in the literature [5].

In Sec. 2, we present a general introduction to adaptive schemes, briefly describe their advantages and drawbacks and the different mechanisms introduced in the literature for solving those drawbacks. As the adaptation scheme discussed in this paper is based on the Real Time transport Protocol (RTP), Sec. 3 shortly presents RTP and its control protocol (RTCP). The main emphasis in Sec. 3 is, however, on the capabilities RTP provides for adaptive applications such as feedback about losses and round trip times.

Sections 4 and 5 present two approaches for adapting the transmission rate of end systems to the network congestion state. Sec. 4 discusses the additive increase/multiplicative decrease algorithms and shows their unfairness towards competing TCP connections. A TCP-throughput model is presented in Sec. 5. Using this model an end system can determine the bandwidth share a TCP connection would be getting under the same circumstances. The direct adjustment algorithm is presented in Sec. 6. The performance of the scheme in terms of bandwidth utilization as well as the behavior of TCP connections traversing the same congested links is then investigated in Sec. 7 and Sec. 8. In Sec. 9, we take a brief look at the scalability of DAA and its behavior when sending data to multicast sessions of several hundred members.

Finally, in Sec. 10 we investigate the fairness of the presented adaptation scheme towards long distance connections and suggest a different approach for improving this aspect.

2 Bandwidth Adaptation Algorithms

As already mentioned in Sec. 1, two of the most discussed approaches for supporting QoS in the Internet are resource reservation and application adaptation. Due to the complexity of reservation systems, their scalability problems and the difficulty in estimating the resources to reserve, reservation seems to be more appropriate for real time services with stringent QoS requirements.

Ergonomic studies and the experience gained from the Internet demonstrates that people can use audio and video data as long as the information content is above a minimum level [32]. This level depends on media content and the task in hand. For instance, a foreign language is more difficult to understand than a native language when audio quality is reduced. So, at the expense of slight degradation in user satisfaction, it is possible to adjust the transmission rates of end systems in accordance with the network congestion state. Hence, deploying application control results in a better overall bandwidth utilization, as it avoids network congestion. A major advantage of application control schemes is that they require no or only minimal support from intermediate routers and are, thus, easy to deploy.

Adaptation algorithms can be classified as either sender-based and receiver-based. In Sender-based adaptation schemes [3, 5] including the one presented in this paper, receivers measure losses or delays and return the results to the senders. Based on the feedback information the senders can determine the appropriate traffic characteristics to use.

However, as IP-based networks such as the Internet traverse paths of widely ranging bandwidths and load factors, such an adaptation approach would perform poorly. In such a heterogeneous environment the conflicting bandwidth requirements cannot be satisfied with one transmission rate. Therefore, the rate is usually adapted to the worst receiver requirements, thereby reducing the perceived QoS at all receiving sites.

To avoid these problems, various proposals have been made for hierarchical data distribution [15, 22]. Those proposals are based on partitioning a data stream into a base layer, comprising the information needed to achieve the lowest quality representation and a number of enhancement layers. The different layers are then sent to different multicast sessions. In the receiver-based adaptation schemes, receivers determine how many sessions to join and thereby adjust their QoS in respect to their own requirements and capacities.

While layered data transmission solves the heterogeneity problems, it might cause additional delay at the receivers. As the different layers might use different paths and hence have different round trip times, the receivers need to resynchronize the arriving data. Additionally, data partitioning might lead to a drift problem caused by motion compensated coding if only the lower bit rate layer is received [16]. Another approach for layered transmission is to simply transmit the same data stream encoded with different quality levels on different multicast sessions [20]. As the streams contain all the necessary information for decompression, the receivers need only to join one multicast session. This approach avoids the resynchronization and drift problems at the cost of sending multiple replicated streams and thus possibly congesting the network.

3 The Real Time Transport Protocol (RTP)

The direct adjustment algorithm is based upon the Real Time transport Protocol (RTP) [24] designed within the Internet Engineering Task Force (IETF). Note that RTP is not a transport protocol in the common sense. That is, it offers no reliability mechanisms, has no notion of a connection and is usually implemented as part of the application and not the operating system kernel. Instead, RTP offers the application the capability of distinguishing between different media streams and keeping track of various statistics describing the quality of the session as seen by other members of the session.

RTP sessions consist of two lower-layer data streams, namely a data stream for, say, audio or video and a stream of control packets (using the sub-protocol called RTCP). Each session member periodically multicasts RTCP control reports to all other session members. Senders transmit reports describing the amount of data they have sent and include in their reports a timestamp indicating the time the report was generated. The receivers send for each incoming stream a receiver report indicating the fraction of lost data, the timestamp of the last received sender report for that stream and the time elapsed in between receiving the last sender report and sending the receiver report.

The desire for up-to-date control information has to be balanced against the desire to limit control traffic to a small percentage of data traffic even with sessions consisting of several hundreds of members. Therefore, the control traffic is scaled with the data traffic so that it makes up a certain percentage of the data rate (usually 5%). To stay within this limit the interval between two consecutive RTCP messages is adjusted as a function of the number of participating members, with a minimum interval of 5 seconds. This design implies that the number of RTCP packets sent by all members of a session should stay constant for sessions of different sizes. However, this is not completely true. As the minimum interval between two RTCP packets is limited to 5 seconds, it is possible that a certain number of members does not consume all the bandwidth available for RTCP. Hence, increasing the number of members results in an increase in the number of RTCP packets sent. Actually, the scaling schemes of RTCP start showing effect only for sessions with a higher number of members than a scaling threshold (th_{scale}) approximately specified as follows:

$$th_{\text{scale}} = \frac{\text{min. Interval} \times B_{\text{RTCP}}}{\text{RTCP Packet size}} \tag{1}$$

with the minimum interval set to 5 seconds and B_{RTCP} as the bandwidth dedicated for RTCP and is set to 5% of the bandwidth used by the session. Detailed analysis of this issue can be found in [11].

Up to the scaling threshold the number of RTCP packets received by the session members does not stay constant and actually increases linearly with the addition of each new member. Note that the equation presented here is only an approximation. The RTCP bandwidth is not distributed equally between senders and receivers. Also, with the addition of new senders the RTCP packet length increases as well.

4 Additive Increase and Multiplicative Decrease Adaptation Approach

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

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

Whereas fairness in this context does not only mean equal distribution of bandwidth among the adaptive connections, but being friendly to competing TCP traffic as well. Various adaptation algorithms proposed in the literature [3, 5, 27] were shown to be efficient in terms of the first two goals of low losses and high utilization but neglected the fairness issue. As those schemes do not consider the bandwidth share of TCP traffic traversing the same congested links, such algorithms might lead to the starvation of competing TCP connections.

As an example for this behavior, we tested the approach described in [5]. This scheme has great resemblance to the schemes described in [3, 27, 10] and is based on the same additive increase/multiplicative decrease approach. With this approach the sender reduces its transmission rate by a multiplicative decrease factor after receiving feedback from the receiver indicating losses above a certain threshold called the upper loss threshold. With losses below a second threshold called the lower loss threshold the sender can increase its transmission rate additively. For the case that the feedback information indicates losses in between the two thresholds the sender can maintain its current transmission level, see Fig. 1. Reducing the rate multiplicatively allows for a fairer reaction to congestion. That is, connections utilizing a disproportionately large bandwidth share are forced to reduce their transmission rate by a larger amount.

To test the behavior of TCP connections sharing the same congested links with connections using this scheme, we simulated the topology depicted in Fig. 2. One TCP connection is sharing a bottleneck router with two connections using the adaptation approach just described. The connection has a round trip time of 10 msec and a bandwidth of 10 Mb/s. The TCP source is based on Reno-TCP. That is, the TCP source reduces its transmission window by half whenever loss is indicated. The adaptive connections have a lower loss threshold of 5% and a higher one of 10%. The additive increase factor was set to 50 kb and the multiplicative decrease factor to 0.875. Those values were suggested in [5] to be most appropriate based on measurements. The router is a random

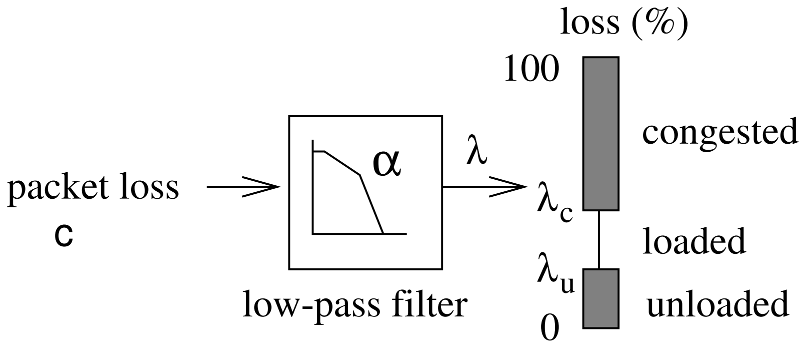


Figure 1: Additive increase/multiplicative decrease algorithm

early drop (RED) gateway as was proposed by Floyd and Jacobson [13]. A RED gateway detects incipient congestion by computing the average queue size. When the average queue size exceeds a preset minimum threshold the router drops each incoming packet with some probability. Exceeding a second maximum threshold leads to dropping all arriving packets. This approach not only keeps the average queue length low but ensures fairness and avoids synchronization effects. In all of our simulations presented in this paper the minimum drop threshold of the router was set to 0.5 and the maximum one to 0.95 based on results achieved in [14].

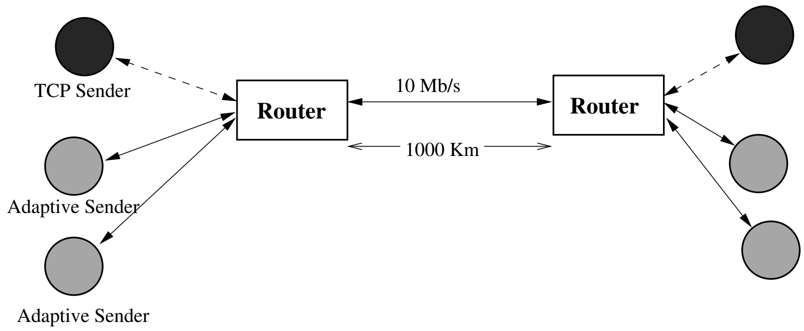


Figure 2: Test configuration for the interaction of the adaptive schemes and TCP

As Fig. 3 shows, the adaptive connections share the available bandwidth among themselves, leaving less than 5% for the TCP connection. This, however, was to be anticipated. With the acknowledgment scheme of TCP, senders are usually informed about packet losses within a time period much smaller than the minimal time between two RTCP packets, i.e., 5 seconds. With each loss notification, TCP reduces its transmission window by half. Older TCP versions, e.g., Tahoe-TCP reduce it even down to one packet [31]. So, while the adaptive source keeps on sending with the high transmission rate until an RTCP packet with loss indication arrives, the TCP source reduces

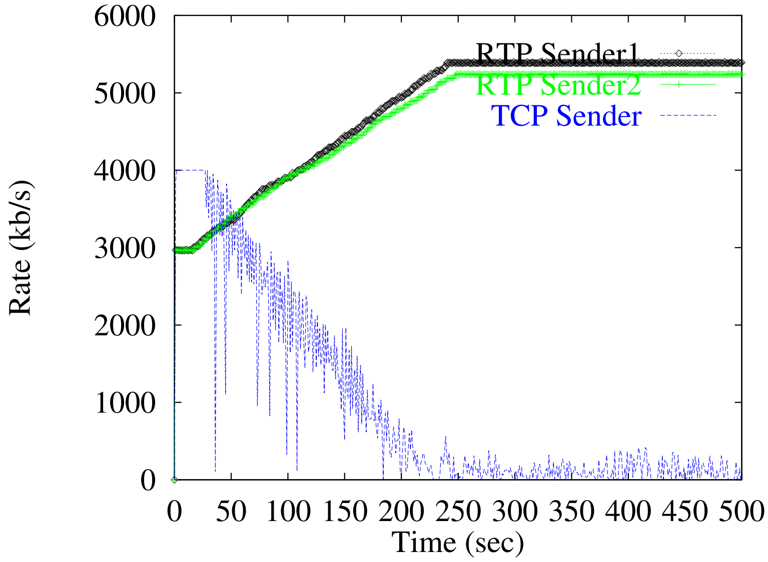


Figure 3: Bandwidth distribution with an additive increase/ multiplicative decrease adaptation scheme

its transmission window and thereby its rate. That is, the adaptive source can for a longer time period send data with the rate that might have actually caused the congestion. Also, as TCP reduces its transmission rate the congestion level will be decreased, so that the adaptive source will finally have to react to a reduced congestion level. Finally, the adaptive scheme will only start reacting to congestion if the losses are larger than the loss threshold. TCP, on the contrary, reacts to any lost packets. Therefore, in Fig. 3 we can notice that during the first 200 seconds, the TCP connection reduces its rate whereas the adaptive connection actually increases its transmission rate. The adaptive connection only starts reacting to congestion after measuring a loss ratio higher than the loss threshold that was set here to 5%. However, as the loss remains below the 10% upper threshold, the adaptive connections can keep their high rate.

5 TCP-Friendly Adaptation Algorithms

The use of the TCP congestion control mechanisms have been a crucial factor in the robustness of the Internet today. However, with the introduction of multimedia applications and multicast transmission that are based on UDP one can easily imagine situations in which a few high bandwidth applications could cause the network to collapse. To maintain this robustness and avoid starvation of TCP connections two different approaches could be used:

- Network-based schemes that deploy restriction mechanisms at the routers. With such schemes the user can send its best effort traffic without regard to the network congestion state. This approach is based on using per-flow scheduling mechanisms, that isolate the different flows [8] and would thereby prevent flows using a disproportionate bandwidth share from congesting the link. However, considering that a router needs to simultaneously support thousands of connections, such an approach would add considerable complexity to the routers. Another approach described in [14] is for the routers to deploy mechanisms to identify connections using disproportionate bandwidth shares and restrict these connections.

  One method for identifying connections using higher bandwidth shares than they should is to use the TCP bandwidth usage model. This model determines the maximum bandwidth share (R_{TCP}) a TCP connection having the same steady-state loss ratio, maximum packet size and minimum round trip time of the connection would be using. The relationship between throughput and loss was calculated in [14, 23] to be:

$$R_{\text{TCP}} = \frac{1.22 \times \text{MTU}}{\text{RTT} \times \sqrt{\text{Loss}}} \tag{2}$$

  with MTU as the maximum packet length and RTT as the round trip time of the connection. To derive Eqn. 2 consider a TCP connections with round trip time RTT. Further consider a steady-state model where the network drops a packet from the connection when the connection's window size increases to W packets. Just before the packet drop the connection would thereby be sending data at the rate of R.

$$R = \frac{W \times \text{MTU}}{\text{RTT}} \tag{3}$$

  After the packet drop the sender reduces its transmission window by half and it takes the sender $W/2$ round trip times until it reaches the window size of W again. Hence, the steady-state throughput the TCP connection receives (R_{TCP}) is on the average $(0.75 \times R)$ bytes/s.

$$R_{\text{TCP}} = \frac{0.75 \times W \times \text{MTU}}{\text{RTT}} \tag{4}$$

  Note that the loss ratio for the TCP connections is,

$$\text{Loss} = 1 \div (\frac{W}{2} + (\frac{W}{2} + 1) + \cdots + W) \tag{5}$$

$$\text{Loss} \sim 1 \div (\frac{3}{8} \times W^2) \tag{6}$$

Solving for W results in

$$W \sim \sqrt{\frac{3}{8 \times \text{Loss}}} \tag{7}$$

Substituting Eqn. 7 in Eqn. 4 results then in Eqn. 2.

This model is based on a TCP sender that interprets any packet drop as an indication of congestion and responds by reducing the congestion window and thereby the effective sending rate by half. Further, this sender should only increase its transmission window by at most one packet each round trip time. A TCP conformant adaptation scheme would thereby use this model for regulating the transmission rate.

While this model is rather simple, it is only partially correct. It does not consider timeout cases or delayed acknowledgments. Also, simulations run in [14] show that it applies only up to a loss ratio of 16%.

Finally, note that this model is based on a single TCP connection with a steady-loss ratio. The authors of [23] discuss that if part of the round trip time is caused by queuing delays, or if the bottleneck router is shared among competing connections, this model is only of limited value. Thereby, using only this model to identify aggressive connections could result in restricting the wrong connections.

- The second approach and the one we are presenting in this paper is based on avoiding network congestion by adjusting the transmission rate of the senders in a TCP-friendly way and in accordance with the network load state. This approach has the advantage of not requiring any additional capabilities in the network.

6 The Direct Adjustment Algorithm

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

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

- Calculate the round trip time (RTT_i) to the reporting receiver i and determine the propagation delay. The RTCP receiver reports include fields describing the timestamp of the last received report from this sender (LSR) and the time elapsed since receiving this report and sending the corresponding receiver report (DLSR). Knowing the arrival time (T) of the RTCP packet the end system can calculate the round trip time. The propagation delay can be assumed to be the smallest value seen for the round trip time. The difference between the propagation delay and the last measured round trip time can then be considered as the queuing delay in the routers.

$$\text{RTT}_i = T - \text{DLSR} - \text{LSR} \tag{8}$$

- As mentioned in Sec. 3, RTCP receiver reports contain the value of the average packet loss measured for this sender in the time between sending two consecutive RTCP packets at the reporting receiver. To avoid reactions to sudden loss peaks in the network the sender maintains for each receiver i a smoothed loss ratio $L_{\text{smooth\_i}}$

$$L_{\text{smooth\_i}} = (1 - \alpha) \times L_{\text{smooth\_i}} + \alpha \times L \tag{9}$$

  with L as the loss value reported in the RTCP packet and α as a smoothing factor set here to 0.3. This value was used in [5] and various simulations and measurements done in [11] suggested its suitability as well.

- Using $L_{\text{smooth\_i}}$ and (RTT_i) the sender calculates $R_{\text{TCP\_i}}$ for receiver i as in Eqn. 2.

- For the case of ($L_{\text{smooth\_i}} \leq 16\%$) the sender sets its transmission rate to (min $[R_{\text{TCP-min}}, R_{\text{Add}}]$) with

$$R_{\text{Add}} \;\; = R + \frac{\text{AIF}}{m} \tag{10}$$

  with AIF as the additive increase factor, R as the current transmission rate and m is set to (min $[\text{members}, th_{\text{scale}}]$). The number of members can be determined using the RTCP packets and the scaling threshold (th_{scale}) can be calculated as in Eqn. 1. $R_{\text{TCP-min}}$ is the minimal $R_{\text{TCP\_i}}$ calculated for all receivers.

  Note that in the TCP model presented in Sec. 5 the round trip time (RTT) is assumed to be constant. In fact, RTT increases during congestion periods due to the queuing delays at the routers. To account for this fact, we used a slightly different TCP model based on the assumption that just before the packet drop the connection has a round trip time consisting of the propagation delay plus the queuing delay. After reducing the transmission window by half and restarting the window increase cycle the connection has a round trip time of only the propagation delay. Hence, the average round trip time we should be considering for calculating $R_{\text{TCP\_i}}$ is

$$\text{RTT}_i = \text{propagation delay} + \frac{\text{queuing delay}}{2} \tag{11}$$

This is a very rough approximation that helps, in part, to introduce the effects of the queuing delay to the TCP model.

- With $(L_{\text{smooth}} \geq 16\%)$, after which the TCP-throughput model is no longer valid, the sending rate is reduced by half. To allow for the reduction to take effect, all RTCP packets are ignored for five seconds, which is approximately the minimum interval between the sending of two consecutive RTCP packets or until another RTCP packet from the receiver, who reported the loss is received.

Determining the propagation delay as the minimal observed round trip time value could result in underestimating $R_{\text{TCP\_i}}$. Consider the case of a connection starting transmission during a network overload period. Until reaching a non-congested state the minimal seen round trip time, i.e., the propagation delay, would include queuing delay and would thereby be probably higher than that of other connections traversing the same path. This would then lead to underestimating the equivalent TCP-throughput and the connection would be restricted to a smaller bandwidth than other connections. This unfairness is, however, only valid during the congestion period. After receiving a more precise measurement of the propagation delay this case becomes no longer of interest.

7 Performance of the Direct Adjustment Algorithm

As a first performance test of the direct adjustment algorithm we used the topology depicted in Fig. 4. The model consists of 15 connections sharing a bottleneck router. All connections deploy the direct adjustment algorithm and are persistent sources. That is, they always have data to send with the maximum allowed rate. They all have the same round trip times and are similar in their requirements and characteristics. The router is a RED router [13] just as in Sec. 5 with a buffer of 200 kbytes. In all of our simulations, we set the packet size to 1 kbyte which is a typical video packet size. Tab. 1 depicts the average utilization results achieved for different round trip times and different link bandwidths. Fig. 6 shows the average bandwidths the single connections achieved for

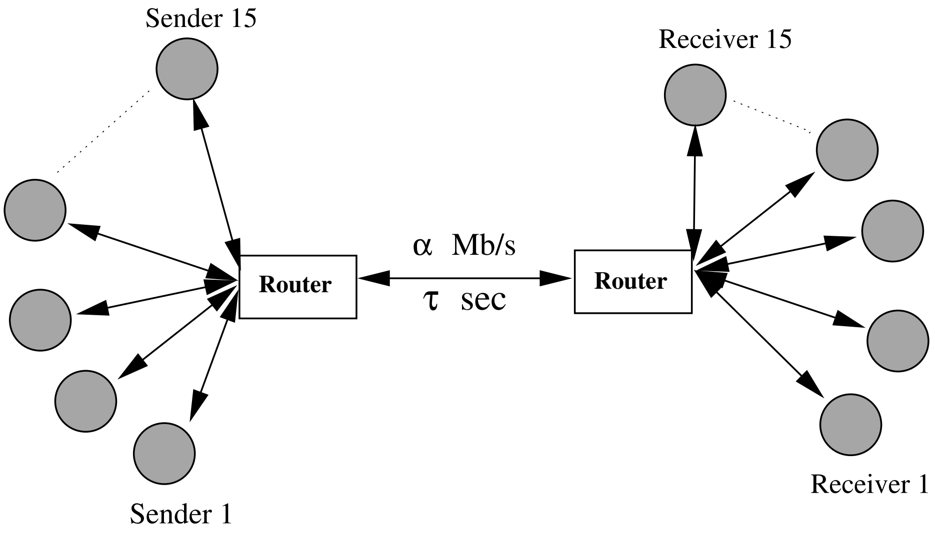


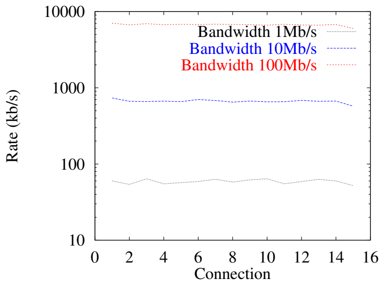
Figure 4: DAA performance testing topology

simulation runs of 1000 seconds. The results shown in Tab. 1 as well as Fig. 5 reveal that using the direct adjustment algorithm bandwidth utilization between 60% and 99% is possible and that the bandwidth is equally distributed among all connections sharing the same link. Note also in Fig. 6, that while connection number 15 starts 100 seconds later than the other connections it soon reaches the same bandwidth level.

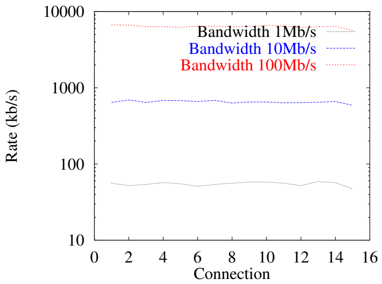
Propagation Delay(τ)	α=1 Mb/s	α=10 Mb/s	α=100 Mb/s
5 μsec	0.89	0.99	0.99
5 msec	0.82	0.97	0.95
100 msec	0.85	0.8	0.60

Table 1: Link utilization with the direct adjustment algorithm for different propagation delays (τ) and link rates (α)

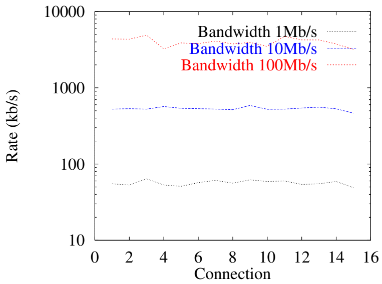
The variance in the achieved utilization results from the oscillatory behavior of the scheme. As Fig. 6 shows, the transmission rate of the senders varies in the range of $\pm$30% of the average rate. These oscillations result in part from the additive increase/multiplicative decrease approach and in part from the partial incorrectness of the TCP model. Additive increase/multiplicative decrease schemes are inherently oscillatory. That is, such systems do not converge to an equilibrium but oscillate around the optimal state [7]. Also, the TCP-model we are using for determining the maximum allowed transmission rate does not consider the bandwidth available on the network or the number of connections sharing a link. Thereby, it might result in throughput values much higher



(a) RTT = 10 μsec



(b) RTT = 10 msec



(c) RTT = 200 msec

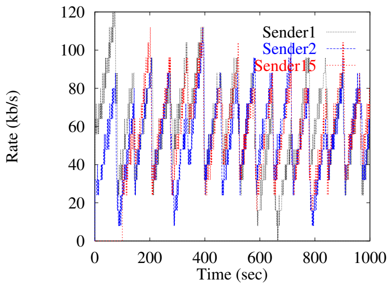
Figure 5: Rate of the single connections of DAA under different round trip times (τ) and link bandwidths (α)

than available on a link, and in other case might underestimate the appropriate bandwidth shares each connection should be using. This is especially evident in Fig. 6(i). Due to the high propagation delay, loss values as low as 0.1% result in a TCP throughput of only 1.5 Mb/s, whereas each connection could have been using a share of 6 Mb/s. So, as the TCP model does not take the available bandwidth into account these oscillations can not be prevented. Also, note that TCP connections are inherently oscillatory as well.

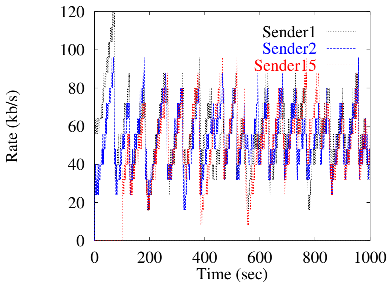
8 TCP and the Direct Adjustment Algorithm

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

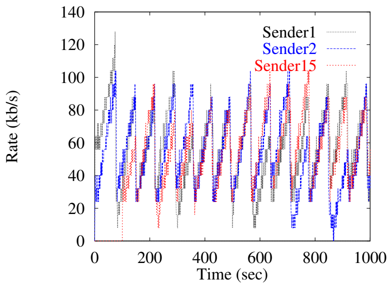
Fig. 7 shows bandwidth distribution achieved with the direct adjustment algorithm using the same topology as in Sec. 5. While using the adaptation scheme presented in [5] leads to the starvation of the TCP connections, see Fig. 3, using the direct adjustment algorithm results in a near equal bandwidth distribution: the TCP connection gets around 30% of the available bandwidth and the two adaptive connections each get 35%. This slight unfairness might be explained with the long control intervals of the RTCP protocol. As the adaptive senders update their transmission rates only every few seconds they might keep on sending with a high rate during congestion periods until a control message indicating losses arrives. During this time, the TCP connection reduces its transmission window and thereby leads to a congestion reduction. Hence, the RTCP messages indicate a reduced congestion state.



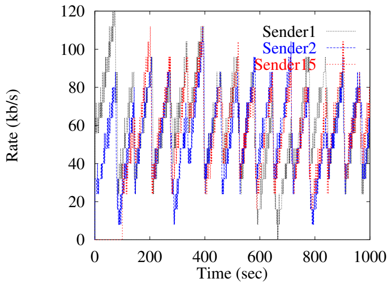
(a) B=1 Mb/s, D=5 μsec



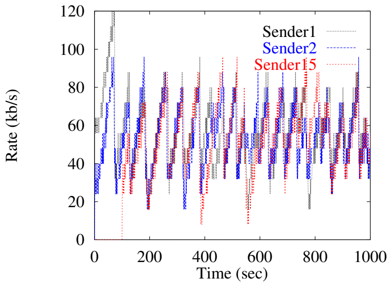
(b) B=1 Mb/s, D=5 msec



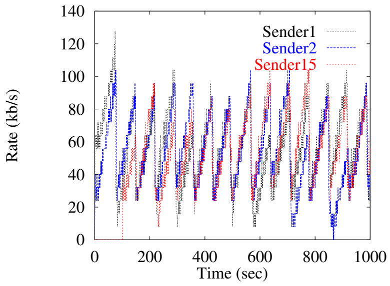
(c) B= 1 Mb/s, D= 100 mses



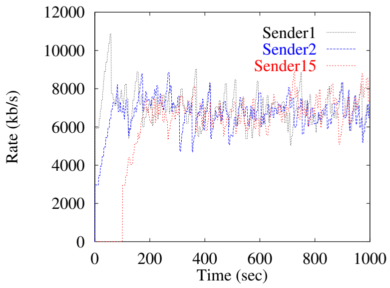
(d) B=10 Mb/s, D=5 μsec



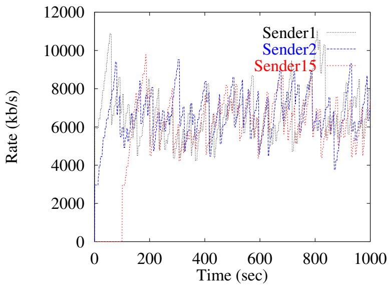
(e) B=10 Mb/s, D=5 msec



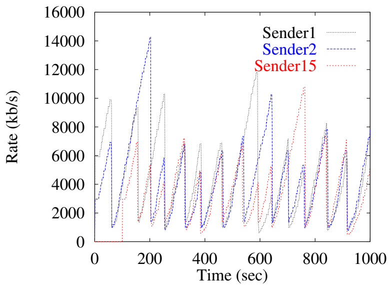
(f) B= 10 Mb/s, D= 200 msec



(g) B=100 Mb/s, D=5 μsec



(h) B=10 Mb/s, D=5 msec



(i) B= 100 Mb/s, D= 200 msec

Figure 6: Temporal behavior of the direct adjustment algorithm under different round trip times (D) and link bandwidths (B)

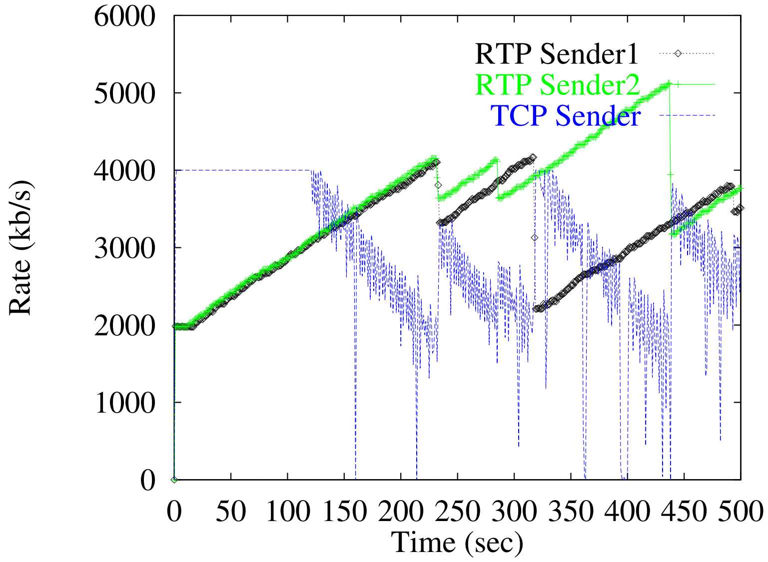


Figure 7: Bandwidth distribution with the direct adjustment algorithm and TCP

9 Scalability of the Direct Adjustment Algorithm

One of the main issues when considering a QoS control approach based on feedback messages is its scalability in multicast sessions with various numbers of participants. As we described in Sec. 3, increasing the number of session members increases the number of received RTCP packets at the receivers. Therefore, the additive increase factor needs to be scaled with the number of control packets. Another possible problem with the scalability of RTCP is that for large numbers of members the interval between two RTCP packets for a single source increases. If a receiver reported high losses and had only a low $R_{\text{TCP\_i}}$, that is throughput according to the TCP model, the sender would not be able to increase its transmission rate above this limit until another RTCP message indicating lower losses arrives from that receiver. For sessions having a large number of members such an update might take a long time ranging from several minutes to several hours. Therefore, to avoid this problem, DAA can be enhanced for the case of very large groups by adding a timeout value for $R_{\text{TCP\_i}}$. That is, each $R_{\text{TCP\_i}}$ entry in the sender's data base gets a timestamp indicating when it was generated. For determining the minimal value ($R_{\text{TCP-min}}$) only $R_{\text{TCP\_i}}$ entries generated after a specific time are considered.

In this section, we consider the behavior of the direct adjustment algorithm for multicast sessions with the number of participant ranging from 4 up to more than 600 members. As a test configuration we used one sender multicasting data through a RED-router to a different number of receivers, see Fig. 9. All receivers were connected through similar links resulting thereby in a ho-

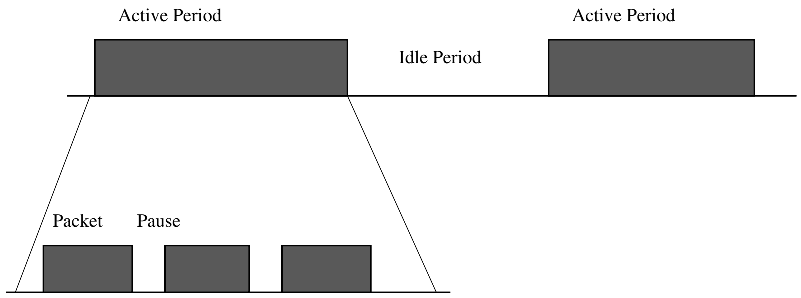


Figure 8: A three state active-idle source

mogeneous environment. To simplify the simulation we only considered the case of lossless RTCP transmission. That is, all RTCP messages were correctly delivered to all receivers. The link rate was set to 1 Mb/s, the propagation delay to 10 msec and the packet size to 1 kbytes. As background traffic we used an active-idle source as was described in [33], see Fig. 8. The number of packets generated during an active state is geometrically distributed with mean N_p. The pause between two packets during the active state is drawn from a negative exponential distribution with mean T_{off}. The idle states can have any general distribution with mean T_{idle}. Based on the traffic characteristics described in [33], we used an active-idle source with N_p set to 50, T_{off} set to 2 msec and T_{idle} set to 5 seconds.

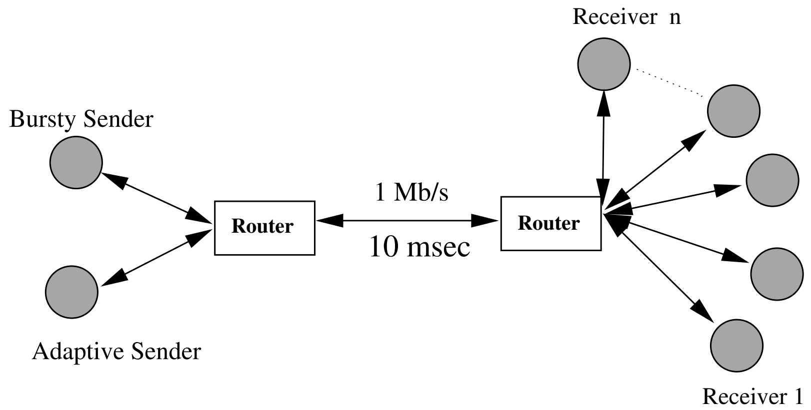


Figure 9: Test topology for the scalability of DAA

Fig. 10 shows the transmission bandwidth of the adaptive sender when multicasting data to a varying number of members. We note that the performance does not change considerably and that the algorithm remains stable even for multicast groups of several hundred members.

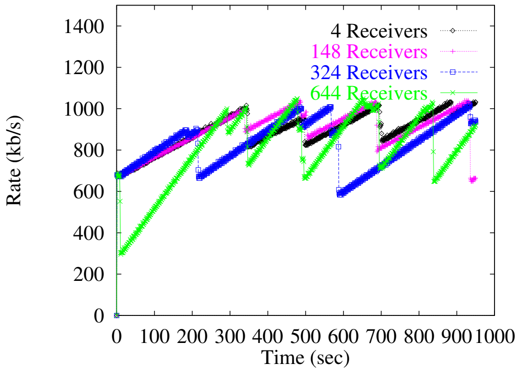


Figure 10: Performance of the DAA scheme with various numbers of receivers

10 Fairness of the Direct Adjustment Algorithm

Adaptive applications are from a certain point of view similar to the available bit rate (ABR) service proposed for ATM. Both schemes adapt the sending rate of the end systems to the network congestion state by increasing the sending rate during underload situations and decreasing it during congestion periods. However, while in the ABR case the network determines the appropriate rate the senders should use, in our case the senders determine the sending rate using the control messages sent by the receivers.

One of the main evaluation criteria of the various proposed ABR schemes [17, 26] is their fairness towards connections with the same parameters and requirements but with different round trip times and paths. As a fairness criterion we used the max-min fairness definition as it was described in the ATM Forum traffic management specification 4.0 [25]. With this definition, a connection's bandwidth request is either satisfied or set to the smallest fair share calculated at all traversed links. This fair share is determined as follows:

$$\text{Fair share} = \frac{\text{B} - \sum \text{bandwidth of constrained connections}}{\text{connections} - \text{constrained connections}} \tag{12}$$

with B as the capacity available for the adaptive traffic. A connection is considered to be constrained if the rate it is currently using is smaller than the fair bandwidth share calculated at this link.

As an example for a max-min distribution, consider the topology depicted in Fig. 11. The first switch has a capacity of 2 Mb/s and is being used by two connections. This would usually imply that each connection should have a fair share of 1 Mb/s. However, the connection from source 1 to destination 1 passes another switch which has a capacity of only 0.5 Mb/s. That is, this connection is bottlenecked at the second switch and can only use a maximum capacity of 0.5 Mb/s. With the max-min allocation, the connection from source 0 to destination 0 should then be getting 1.5 Mb/s.

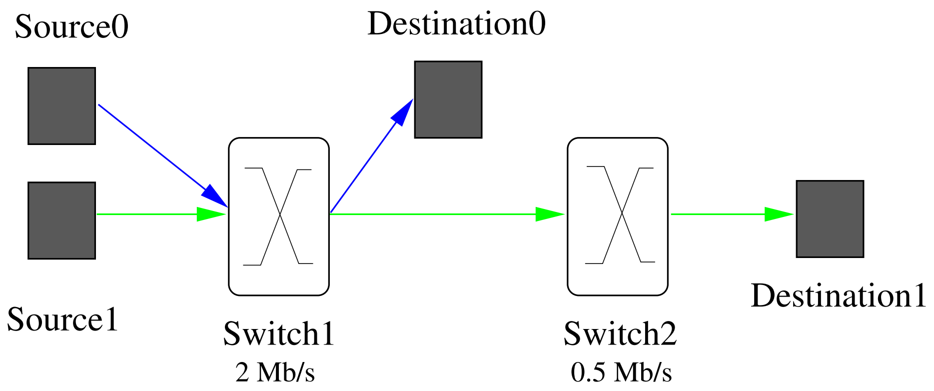


Figure 11: Example for the max-min fair bandwidth distribution

In Fig. 12, we use a generic fairness topology as was proposed in [33] to investigate the fairness of the algorithm when handling with connections that have the same requirements and parameters but differ in their round trip times and traversed paths. All the links have a link rate of 1 Mb/s and are 1000 km long. The connection from source 1 to destination 1 (transit source) traverses three links, all other connections traverse only one link (local connections). All sources have an initial sending rate of 700 kb/s and a bandwidth range of 10 kb/s to 1.5 Mb/s. Here, again, RED routers are used.

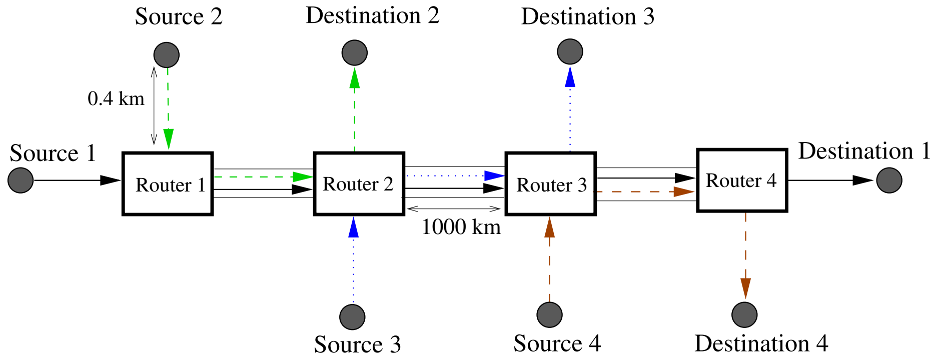
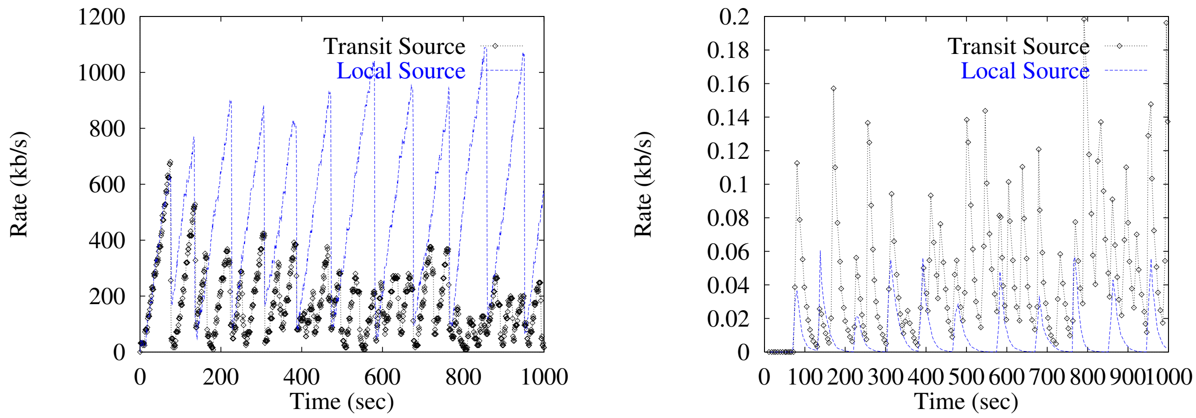


Figure 12: Network topology for investigating the max-min fairness behavior of the direct adjustment algorithm

As all links are shared only between two connections, we would expect every connection to get half the link capacity, i.e., 500 kb/s. However, Fig. 13(a) shows that the rate of the transit connection actually gets reduced to an average rate of around 150 kb/s while the local connections receive an average rate of 600 kb/s.

It is obvious that the direct adjustment algorithm is unfair towards the long distance connection.

This is due to a problem already noticed in TCP [12, 29] and early ABR proposals [1], namely the *beat-down problem*. The beat-down problem arises as a result of the increase in the loss probability with each additional traversed link, see also [28]. Hence, the long distance sender receives control messages with higher loss values than the loss values reported to the local senders. In addition to the higher loss ratio of the transient connection it also has a higher round trip time. This additionally results in lower throughput according to the TCP model. Up to our knowledge, no solution has been proposed for solving this problem in TCP until now. In the case of ABR, several solutions requiring the network switches to calculate the fair share and sending this value to the sources were introduced [18].



(a) Bandwidth distribution

(b) Loss

Figure 13: Bandwidth and loss values measured at the receivers with adaptation applied

A solution that would ensure fairness for both TCP connections as well as adaptive sources would be to choose an approach similar to that used in ABR. That is, introduce a new service class to the integrated service model that not only guarantees a minimal QoS level but dynamically informs the senders of the actual resources they could use. So, with such an approach the intermediate routers could calculate the fair bandwidth share each connection could use. Using RSVP as a QoS signaling protocol, end systems establish a closed control loop. The senders inform the network and receivers about their traffic characteristics by sending PATH packets. The actual reservation is triggered by the receivers who send their reservation wishes based on the traffic profiles announced. To use a service similar to ABR, senders could additionally inform the network about their current sending rate and the maximum rate they would like to use. Based on those parameters and the current overall traffic situation intermediate routers can calculate the fair rate shares the senders should be using to avoid congestion. If the calculated fair share is lower than the desired rate the source is announcing, the desired rate is decreased down to the fair share. After traversing different routers and arriving at the receivers, the path packets would contain the fair share the connection should be using. The receiver can now make its reservation based on this value and inform the sender of the fair share value it should be sending with. We can imagine a similar approach based on the RTCP protocol with the senders writing their current and desired rate in the sender reports, the routers writing the fair share in the RTCP sender reports and the receivers informing the senders about the fair share using the receiver reports. Naturally, such a service would require some kind of dynamic admission control and algorithms that can efficiently calculate the fair shares without adding considerable complexity to the routers.

11 Summary and Future Work

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

While the DAA scheme presented here achieves a high utilization, low loss and a fair distribution of the available bandwidth among connections with the same round trip times, the scheme is unfair towards connections with longer round trip times. Our belief is that the fairness problems are inherent to any adaptation scheme based on control information collected only at the end systems. Due to the beat-down problem, the loss ratio observed for a given connection will usually increase with the number of traversed links. So, with the higher loss ratio the sources will tend to decrease their sending rates more often and thereby introduce the here observed unfairness. Work done on other adaptation schemes such as [5] show the same unfair behavior noticed in this work [11]. A possible solution might be to introduce a service class for adaptive applications similar to the ABR service of ATM [25]. That is, let the network nodes determine the exact fair bandwidth share a connection should use to avoid congestion and notify the sender of this fair share.

A main drawback of the scheme is its oscillatory behavior. Multimedia application that could benefit from adaptation schemes require, at least partially, steady rate streams. Notifying end systems of the exact bandwidth share they should be using would erase the need for the end systems to discover the appropriate transmission rates by themselves and would reduce those oscillations considerably. Another approach might be to further enhance the TCP model in order to calculate more precise results [23].

Another major point that needs to be considered here, is the value to use for the additive increase factor (AIF). In our simulations, we set the AIF based on running different simulations and choosing the most appropriate results. We are currently working on enhancing the scheme with methods to dynamically discover the appropriate values. Better tuned additive increase factors might also lead to a more stable bandwidth distribution.

Finally, we are currently working on implementing the direct adjustment scheme in the NEVIT [30] video tool and will soon be running different tests to verify the behavior we noticed through the simulations.

12 Acknowledgments

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

References

[1] A. Arulambalam, X. Chen, and N. Ansari. Allocating fair rates for available bit rate service in atm networks. *IEEE Communications*, 34(11):92 – 100, Nov. 1996.

[2] S. Baker. Multicasting for sound and video. *Unix Review*, 12(2):23–29, Feb. 1994.

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

[4] R. Braden, L. Zhang, and S. Berson. Resource reservation protocol (RSVP) – version 1 functional specification. Internet Draft, Internet Engineering Task Force, Nov. 1995. Work in progress.

[5] I. Busse, B. Deffner, and H. Schulzrinne. Dynamic QoS control of multimedia applications based on RTP. *Computer Communications*, 19(1):49–58, Jan. 1996.

[6] A. Campbell, D. Hutchinson, and C. Aurrecoechea. Dynamic QoS management for scalable video flows. In *Proc. International Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV)*, Lecture Notes in Computer Science, pages 107–118, Durham, New Hampshire, Apr. 1995. Springer.

[7] 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.

[8] D. D. Clark, S. Shenker, and L. Zhang. Supporting real-time applications in an integrated services packet network: architecture and mechanism. In *SIGCOMM Symposium on Communications Architectures and Protocols*, pages 14–26, Baltimore, Maryland, Aug. 1992. ACM. *Computer Communication Review*, Volume 22, Number 4.

[9] C. Diot, C. Huitema, and T. Turletti. Multimedia application should be adaptive. In *HPCS*, Mystic, Connecticut, aug 1995.

[10] R. El-Marakby and D. Hutchison. Towards managed real-time communications in the internet environment. In *The Fourth IEEE Workshop on the Architecture and Implementation of High Performance Communication Systems (HPCS'97)*, Chalkidiki, Greece, June 1997.

[11] F. Emanuel. Investigation of scalability and fairness of adaptive multimedia applications. Studienarbeit, School of Engineering, TU Berlin, Berlin, May 1997. Work in progress.

[12] S. Floyd. Connections with multiple congested gateways in packet-switched networks, part 1–one-way traffic. *ACM Computer Communication Review*, 21(5), Oct. 1991.

[13] S. Floyd and V. Jacobson. Random early detection gateways for congestion avoidance. *IEEE/ACM Transactions on Networking*, 1(4):397–413, Aug. 1993.

[14] S. Floyd and F. Kevin. Router mechanisms to support end-to-end congestion control. Technical report, Feb. 1997.

[15] D. Hoffman and M. Speer. Hierarchical video distribution over internet-style networks. In *ICIP'96*, Lausanne, Switzerland, Sept. 1996.

[16] U. Horn and B. Girod. Scalable video coding for the internet,. In *8th Joint European Networking Conference*, Edinburgh, England, May 1997.

[17] D. Hunt, J. Scott, J. C. R. Bennet, A. Chapman, R. Nair, B. Simcoe, H. T. Kung, and K. Brinkerhoff. Credit-based FCVC proposal for ATM traffic management -revision R1. Technical Report 94-0168R1, ATM Forum, May 1994.

[18] R. Jain. Congestion control and traffic management in ATM networks: Recent advances and a survey. *Computer Networks and ISDN Systems*, Feb. 1995.

[19] I. Kouvelas, V. Hardman, and A. Watson. Lip synchronization for use over the internet: Analysis and implementation. In *GLOBECOM'96*, London, UK, Nov. 1996.

[20] X. Li and M. 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 Societey.

[21] S. McCanne and V. Jacobson. vic: A flexible framework for packet video. In *Proc. of ACM Multimedia '95*, Nov. 1995.

[22] S. McCanne, V. Jacobson, and M. Vetterli. Receiver-driven layered multicast. In *SIGCOMM Symposium on Communications Architectures and Protocols*, Palo Alto, California, Aug. 1996.

[23] T. Ott, J. Kemperman, and M. Mathis. Window size behavior in tcp/ip with constant loss probability. In *The Fourth IEEE Workshop on the Architecture and Implementation of Hi gh Performance Communication Systems (HPCS'97)*, Chalkidiki, Greece, June 1997.

[24] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson. RTP: a transport protocol for real-time applications. RFC 1889, Internet Engineering Task Force, Jan. 1996.

[25] S. S. Shirish. ATM Forum traffic management specification version 4.0. Technical Report 94-0013R6, ATM Forum, June 1995.

[26] D. Sisalem. Behavior of various switch mechanisms for the abr service in the presence of persistent and dynamic traffic. In *The Fourth IEEE Workshop on the Architecture and Implementation of High Performance Communication Systems (HPCS'97)*, Chalkidiki, Greece, June 1997.

[27] D. Sisalem. End-to-end quality of service control using adaptive applications. In *IFIP Fifth International Workshop on Quality of Service (IWQOS '97)*, New York, May 1997.

[28] D. Sisalem and H. Schulzrinne. End-to-end rate control in ABR. In *First Workshop on ATM Traffic Management (WATM'95)*, pages 281–287, Paris, France, Dec. 1995. IFIP WG 6.2.

[29] D. Sisalem and H. Schulzrinne. Congestion control in TCP: Performance of binary congestion notification enhanced TCP compared to Reno and Tahoe TCP. In *International Conference on Network Protocols (ICNP)*, pages 268–275, Columbus, Ohio, Oct. 1996.

[30] D. Sisalem, H. Schulzrinne, and C. Sieckmeyer. The network video terminal. 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.

[31] W. R. Stevens. *TCP/IP illustrated: the protocols*, volume 1. Addison-Wesley, Reading, Massachusetts, 1994.

[32] F. Wilson, I. Wakeman, and W. Smith. Quality of service parameters for commercial application of video telephony. In *Human Factors in Telecommunication Symposium*, Darmstadt, Germany, Mar. 1993.

[33] L. Wojnaroski. Baseline text for traffic management sub-working group. Technical Report 94-0394r5, ATM Forum, Oct. 1994.