Available at
www.ElsevierComputerScience.com
POWERED BY SCIENCE DIRECT•

Computer Communications 27 (2004) 345–354

computer communications

www.elsevier.com/locate/comcom

How to increase the efficiency of receiver-driven adaptive mechanisms in a new generation of IP networks

Paulo Mendes[a,b,*], Henning Schulzrinne[a], Edmundo Monteiro[b]

[a]Department of Computer Science, Columbia University, New York 10027-7003, USA
[b]Department of Informatics Engineering, University of Coimbra 3030 Coimbra, Portugal

Received 8 August 2003; accepted 8 August 2003

Abstract

Receiver-driven adaptation allows streaming of multicast multimedia content to different receivers across networks with heterogeneous characteristics, such as different resource availability. However, receivers are only encouraged to adapt if network providers guarantee a fair distribution of bandwidth and also the punishment of receivers that do not adjust their rate in case of congestion. Therefore, we define a receiver-driven adaptive mechanism based on a new fairness protocol that provides the required guarantees for adaptation in heterogeneous networks. We use simulations to evaluate the proposed mechanism and to compare its performance with other receiver-driven mechanisms. Our work contributes to show that the problems of receiver-driven adaptation, such as the induction of losses, can be solved even with simple receiver-driven adaptive mechanisms, when a fair allocation of resources in provided by the network.
© 2003 Elsevier B.V. All rights reserved.

Keywords: Receiver-driven adaptation; Streaming; SAPRA; Fairness; TCP

1. Introduction

The increasingly higher number of Internet users constitutes an encouragement for multimedia providers, and network providers to create new services in the Internet. This will allow them to continue generating revenues and to reduce their operational cost. These new services include support for long-lived sessions that reach large audiences, such as streaming. But, even without showing a 'killer application' behavior, since consumers normally take on the order of a decade to embrace a new technology [26], streaming applications can cause short-term periodic disruption of the Internet. This can happen since users can generate high transmission rates, which are sometimes destined for large audiences. To limit quality degradation, the adaptiveness of video can be used by congestion control mechanisms to perform graceful adjustments to the perceptual quality of the displayed video stream in response to quality fluctuations.

One possible congestion control mechanism can be performed by the source. However, source-based rate adaptation performs poorly in a heterogeneous multicast environment, where there is no single target rate: the different bandwidth requirements of receivers cannot be simultaneously satisfied with one transmission rate. This problem can be solved if streaming sources use scalable encoding and the adaptation task is performed by receivers. Scalable encoding [16,32] divides a video stream into cumulative layers with different rates and importance. Layers are then sent to different multicast groups. The rate of each stream is obtained by adding the rates of all its layers. When a receiver-driven adaptive approach is combined with scalable encoding, receivers can adapt to the best quality the network offers, by joining the multicast group of a subset of layers of a stream. Besides the adaptation to heterogeneous environments, receiver-driven adaptation mechanisms have other advantages over sender-based ones namely, in the former the burden of adaptation is distributed among receivers resulting in enhanced system scalability. Receiver-driven adaptation also avoids possible congestion near the sender, which can occur in sender-based

[*] Corresponding author. Department of Computer Science, Columbia University, New York 10027-7003, USA. Tel.: +1-212-939-7000; fax: +1-212-666-0140.
E-mail addresses: mendes@cs.columbia.edu (P. Mendes), pmendes@dei.uc.pt (P. Mendes), schulzrinne@cs.columbia.edu (H. Schulzrinne), edmundo@dei.uc.pt (E. Monteiro).

0140-3664/$ - see front matter © 2003 Elsevier B.V. All rights reserved.
doi:10.1016/j.comcom.2003.08.007

mechanisms due to synchronized notifications sent by receivers.

However, receiver-driven approaches normally have low performance, namely due to latency when leaving a layer. An enhancement of receiver-driven adaptive mechanisms can be done by collecting more accurate information from the network in order to increase the adaptation efficiency. Beside this, it is not clear how motivated receivers are to adapt to a congestion situation, since they do not have any guarantees that receivers that are inducing congestion will also reduce their perceptual quality in order to reduce the network traffic. Therefore, to motivate receivers to adapt, the network has to have three fairness properties. The first is *inter-session* fairness, where a session is composed by the group of layers of a stream. Inter-session fairness is the ability to guarantee a fair distribution of bandwidth between sessions sharing a service. The second is *intra-session* fairness, the ability to respect the importance of each layer of a session. The third is the ability to punish *high-rate* sessions, i.e. sessions with a rate higher than their fair share of bandwidth. Sessions become high-rate sessions when their receivers do not reduce the reception rate when packets are lost. This is, even after detecting packet losses, receivers do not leave layers of their session.

One major consequence of the network support required by enhanced services, such as streaming, is that it enters in conflict with some principles of the original Internet architecture, namely the end-to-end argument [31]. This principle is broken by many new developments, such as the need to police different types of traffic, and the debut of new end hosts: appliances and not 'intelligent' personal computers. Therefore, some reflection is required to adjust the end-to-end principle to a new Internet architecture. An example is the view described by D. Clark et al. [4] of an Internet composed by regions managed by different policies, and maintaining its heterogeneity principle. The functionality of the regions can be managed by the *Differentiated Services* (DiffServ) model [3].

The current DiffServ model aggregates traffic into services with different priorities at the boundaries of each network domain. Among the available services, the ones based on the *Assured Forwarding* (AF) [7] PHB are ideal to transport scalable sessions, since layers are assigned different drop precedences. These type of services reduces the price of communications, and allows an easier management of resources for multi-receiver sessions, which cannot be achieved by virtual leased line service type of services based on the *Expedited Forwarding* (EF) [12] PHB. Although AF services provide intra-session fairness, the DiffServ model lacks the other two required properties. Therefore, we proposed a protocol named *Session-Aware Popularity-based Resource Allocation* (SAPRA) [23–25] that allows a fair allocation of resources in each DiffServ service. SAPRA provides inter-session fairness by assigning more bandwidth to sessions with higher audience size, and intra-session fairness by assigning to each layer a drop precedence that matches its importance. SAPRA has a punishment function and a resource utilization maximization function. The former increases the drop percentage of high-rate sessions during periods of congestion. The latter avoids waste of resources when sessions are not using their whole fair share: the remaining bandwidth is equally distributed among other sessions. SAPRA is implemented in edge routers to handle individual traffic aggregated in each service: interior routers are not changed.

In this paper, we propose a simple receiver-driven adaptive mechanism named SAPRA *Adaptive Mechanism* (SAM), which uses the network support provided by the SAPRA protocol. Simulation results show that when network resources are fairly distributed using SAPRA, simple receiver-driven adaptive mechanisms such as SAM have good performance. These simple adaptive mechanisms are in accordance with the debut of new appliances, which are not as intelligent as personal computers. Therefore, we contribute to the creation of new generation of IP networks, by proposing a simple receiver-driven adaptive mechanism that allows clients of multicast streaming to adapt the perceptual quality of the displayed video, based on a signaling protocol that ensures a fair allocation of shared resources in DiffServ networks. The adaptation is done considering the quality requirements of receivers, and periodic information collected from SAPRA. The goal of our work is to provide an answer to efficiently multicast a stream to different users throughout a heterogeneous network, and to encourage users to use network services coherently in order not to disrupt the Internet. The design of SAM, and the support that it receives from SAPRA is consistent with the Internet architecture.

The remainder of this paper is organized as follows. In Section 2, we describe related adaptive mechanisms. Section 3 describes SAPRA support to receiver-driven adaptive mechanisms and characterizes SAM operation. In Section 4 we evaluate SAM using simulations. Section 5 presents some conclusions.

2. Related work

McCanne et al. [22] developed the *Receiver-driven Layered Multicast* (RLM) mechanism, the first receiver-driven adaptive mechanism for scalable sessions, which performs well over a broad range of conditions. However, RLM has high instability with bursty traffic such as *Variable Bit Rate* (VBR), poor fairness and low bandwidth utilization [28]. Vicisano et al. [33] described a protocol called *Receiver-driven Layered Congestion control* (RLC) that complements a TCP-friendly version of RLM. RLC is based on the generation of periodic bursts that are used for bandwidth inference, and on synchronization points that indicate when a receiver can join a layer. However, RLC does not solve issues such as slow convergence and losses provoked by the adaptive mechanism on other flows.

An analysis of the pathological behavior of RLM and RLC [18] showed that their bandwidth inference mechanism is responsible for transient periods of congestion, instability and periodic losses. In RLM, there is no explicit notifications of what is the current congestion level, so join-experiments are carried out to find if the receiver is able of joining another layer. However, a failed join-experiment can increase the congestion in the network, resulting in a degradation of the video quality both in the receiver who initiated the experiment, and possible on others receivers that share the congested link. In the case of RLC, the inference mechanism leads to instability, since the current layer is dropped when receivers experience losses during a burst, whereas, according to Vicisano et al. [33], RLC should stay at the current layer and just infer that it cannot join an upper layer. Moreover, the analysis of Legout et al. [18] shows that the bandwidth inference of RLC depends on the judicious choice of the size of network queues. Hence, since the bandwidth inference is not always successful, receivers can be periodically induced to join a layer when there is not enough bandwidth available, which leads to congestion.

Legout et al. [20] developed a receiver-driven protocol for scalable sessions named *packet Pair receiver-driven cumulative Layered Multicast* (PLM) that uses *Packet-pair Probing* (PP) [17] and *Packet-level Generalized Processor Sharing* (PGPS) scheduling [27] to infer the available bandwidth in the path. With this, PLM avoids the instability and losses induced by the bandwidth inference mechanism of RLM and RLC. However, PLM has four major disadvantages. First, PP measurements can have large oscillations, since PP depends on packet size and the burstiness of traffic. Second, PLM requires all packets from all layers to be sent back-to-back and adds an extra bit to the packet header to identify the start of a PP burst. Third, PLM is not suitable for DiffServ scenarios, since all routers have to implement PGPS. Fourth, PGPS uses the max–min fairness definition [1]. However, this fairness definition cannot be applied to discrete sets of rates [30], does not take into account the audience size of sessions, not increasing the number of receivers with good reception quality, and does not punish high-rate sessions.

A different approach is presented by Jiang et al. [15], who developed a protocol to control the rate of a multicast session, with the goal of maximizing the inter-receiver fairness. The proposed protocol requires the source to transmit the multimedia content in two different multicast groups: the main group has a variable rate, and the alternative group has the lowest possible rate that is acceptable to the sender. Receivers always join the main multicast group first, but if losses start to be experienced, they notify the sender with the goal of potentially influencing it to reduce the rate of the main multicast group. If losses continue despite of this, receivers leave the main multicast group and join the alternative one. This approach has several drawbacks: first, it requires the simultaneous transmission of the same multimedia content to two different multicast groups. Second, it does not cope with heterogeneity, since all receivers get the same rate for the session. Moreover, the authors refer that different techniques are required to aggregate and process notifications sent by receivers, but do not clarify how these techniques can avoid congestion near the sender.

3. SAM description

In this section we describe how receivers use SAM to adapt to different network conditions. We also explain briefly how SAPRA support simple receiver-driven adaptive mechanisms, such as SAM. SAPRA is described and evaluated in Ref. [25] and a detailed study of its fairness policy can be found in Refs. [23,33].

3.1. SAPRA support for receiver-driven adaptation

SAPRA is most suited for long-lived scalable streams with large audiences such as Internet TV, and *Near Video-on-Demand* (NVoD) systems. Each stream is sent to all receivers of the audience by using multicast, being each layer of the stream identified by a different *Source-Specific Multicast* (SSM) channel [2].

Each scalable stream is mapped to one *SAPRA session*, which consists of all SAPRA messages that refer to the same state, traverse the same multicast path or part of it, and share the same session identifier, independently of the direction in which messages travel. The state of a SAPRA session is defined as a group of *SAPRA layers*, where each SAPRA layer corresponds to one layer of a scalable stream. The importance relationship among SAPRA layers is the same as the one among scalable layers.

Although perfectly adjusted to scalable multicast streams, SAPRA integrates also non-scalable streams and unicast traffic: the former are treated as SAPRA sessions with one layer, and the latter as SAPRA sessions where the destination address of each layer is the unicast address of the unique receiver.

DiffServ edge routers that implement SAPRA are called *SAPRA nodes*. SAPRA nodes are placed only at the edges of DiffServ domains, since they only need to access the individual traffic of each service. We call *SAPRA path* to the path that include all SAPRA nodes visited by a SAPRA session from its source to one of its receivers. A *SAPRA tree* is therefore the group of all SAPRA paths rooted at the sender. A SAPRA tree can be a unicast of multicast tree. In the former case, the tree has only one branch, while in the latter case, the tree can have from one or several branches, depending on the number of receivers and their location. SAPRA does not restrict the location of multicast branch points, which can be positioned in edge or interior routers of DiffServ domains. In either case, receivers are attached only to edge routers: the DiffServ model mentions that an end host may act as an edge router for applications running on that host. If a host does not act as an edge router, then the closest DiffServ node acts as the edge router for the applications on that host.

Each SAPRA node implements one *SAPRA agent* and one *SAPRA traffic conditioner* per downstream link. By downstream we mean the direction from the sender to receivers and upstream we mean the opposite direction. Agents implement the core of SAPRA functionality and have responsibilities only in the control plane. Traffic conditioners enhance markers and droppers of DiffServ, to make the connection between SAPRA agents and the data plane, by policing the traffic of each session.

SAPRA is designed to fit a new architecture based on domains. However, not all domains have to implement SAPRA. SAPRA messages cross transparently all non-SAPRA domains in a path, since they are only exchanged between SAPRA agents. This allows a progressive deployment of SAPRA. However, receivers in non-SAPRA domains do not count toward the audience size of their SAPRA session and thus sessions with a large audience in non-SAPRA domains are left with a smaller share of resources.

Since SAPRA aims to distribute resources reserved for network services, and not to perform the reservation of such resources, we assume that the capacity of each service is dimensioned by the domain administrator to avoid congestion in interior routers. The development of inter-domain and intra-domain reservation mechanisms responsible for the provisioning of services is beyond the scope of this paper.

Each agent is responsible to compute the fair share that each session is entitled to use in each link downstream the node where the agent is located. The *fair rate* represents a percentage of the link bandwidth, given by the ratio between the session audience size and the total population of the link. Eq. (1) gives the fair rate F_{ui} of a session S_u in a link i, where n_{ui} is the audience size of S_u and C_i is the bandwidth of the service shared by m_i sessions.

$$F_{ui} = \left(\frac{n_{ui}}{\sum_{x=1}^{m_i} n_{xi}} \right) C_i \tag{1}$$

The *sustainable rate* of a session is also computed for each downstream link. The sustainable rate is the larger of the fair rate of the session and the sum of the session rate plus the bandwidth not being used in the link. Eq. (2) gives the sustainable rate U_{ui} of a session S_u in a link i, where r_{ui} is the rate of S_u, and b is the bandwidth not being used in that link.

$$U_{ui} = \max(F_{ui}, (r_{ui} + b)) \tag{2}$$

The traffic conditioner marks and drops packets in each SAPRA node based on the fair rate of each session. Besides controlling the traffic with fairness in each SAPRA node, SAPRA also provides receivers with periodic reports about the minimum sustainable rate in the path from their session source. Reports are updated with a minimum interval of 1s. In case the sustainable rate does not change significantly (25% or more in our experiments), reports are suppressed.

Receivers use SAM to adjust the perceptual quality of the displayed video based on the session sustainable rate and on the existence of congestion in the session path. The sustainable rate can increase if the session has a higher audience, other sessions have a lower audience, or there is bandwidth not being used in the path of the session.

3.2. Overview of SAM operation

Receivers can join sessions by, for instance, listening to *Session Announcement Protocol* (SAP) [6] messages, which may include information about the address and rate of each layer. Receivers join first the multicast group for the most important (lowest) layer and then the SAM algorithm controls the reception quality by joining and dropping additional (higher) layers.

The decision to join or drop layers depends on the session sustainable rate and on the existence of congestion in the session path. The sustainable rate, provided by SAPRA, gives SAM an indication of the maximum number of layers that receivers can join. Packet loss is a sign of penalization for high-rate sessions in a congested path. Packets start to be dropped from the less important layer, since SAPRA protects the most important ones. When losses happen in any layer, SAM is triggered to leave layers. In this paper we assume a loss limit of 2.5% as the maximum quality degradation allowed by receivers. We chose this value based on the study made by Kimura et al. [10], which shows that in MPEG-2 layering with Signal to Noise Ratio scalability, 5% of losses in the most important layer in addition to 100% of losses in all other layers, lead to a decrease of the quality of sessions from good to bad accordingly to the ITU-500-R rating [11].

SAM operation is divided into three states as shown in Fig. 1: steady state, join state and drop state. Receivers remain in the steady state as long as they do not receive a report and while losses are lower than 2.5%. Upon receiving a report, receivers enter the join state. If the new sustainable rate is higher than the previous one, receivers increase their reception quality, by adding layers. Since receivers know in advance the average rate of each layer, they immediately

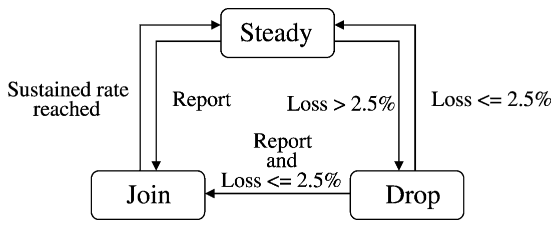


Fig. 1. SAM states.

join as much layers as possible. The number of layers they can join is upper bounded by the sustainable rate of their session. After this, receivers return to the steady state. Receivers react to a lower sustainable rate considering the percentage of lost packets and not the report. If losses rise above 2.5%, receivers enter the drop state. In the drop state, receivers drop a layer every 500 ms, the *non-reacting* period, while losses are higher than 2.5%. The non-reacting period avoids over-reacting to losses. With losses equal or below 2.5%, a receiver enters the join state if it receives a new report while in the drop state. Otherwise, it returns to the steady state.

Since we compare SAM with PLM in Section 4, we use the same non-reacting period as PLM. Although we use a loss limit of 2.5% and a non-reacting period of 500 ms, these values can be changed to suit other configurations.

4. SAM evaluation

In this section we present simulations (using NS) that aim to show that SAM has small convergence time, remains stable in the optimal quality level, is fair towards TCP, and allows receivers to use a rate proportional to the audience size of their session and to the amount of bandwidth not being used. We use the three scenarios shown in Fig. 2.

As metrics, we use the rate measured at receivers, presented with a precision of 10 kb/s. As system parameters, we use the type of traffic and the granularity of layers, since these factors affect the adaptation performance. We use layers with exponential rates, which are common in scalable codecs [29], and thin layers as diagnostic tool, since they can identify pathological behaviors.

The first simulation uses the topology *Top1* of Fig. 2. We aim to show how the sustainable rate computed by SAPRA allows SAM to reach an optimal quality level. We use four sessions: S_1 spans seconds 30–130, and S_2, S_3 and S_4 span seconds 10–240. Sessions S_1, S_2 and S_3 have one receiver each, and S_4 has one receiver until second 170 and five receivers after that. Each session has six layers: the most important layer, l_1, has 32 kb/s and each layer l_i has a rate equal to twice the rate of l_{i-1}. Sessions S_1 and S_2 share the link between routers r_1 and r_2, (r_1, r_2), and S_2, S_3 and S_4 share (r_2, r_3). R_i represents receivers of S_i. We assume that queues have a size of 64 packets, which is the default value in Cisco IOS 12.2, and data packets have 1000 bytes, a middle value between the 576 bytes MTU of dial-up connections and the 1500 bytes MTU of ethernet and high speed connections.

Fig. 3 (left) shows the rate that receivers get when sessions have Constant Bit Rate (CBR) sources. Until $t = 30$ s, S_2 is the only session in (r_1, r_2), so it has a sustainable rate of 1 Mb/s in that link. In (r_2, r_3), there are three sessions, S_2, S_3 and S_4. Since these three sessions have one receiver each, SAPRA distributes the link bandwidth equally between them, giving each a sustainable rate of 1 Mb/s. Therefore, R_2, R_3 and R_4 receive a sustainable rate of 1 Mb/s in the first report. This allows them to join four more layers, reaching a rate of 992 kb/s, as show in Fig. 3 (left). At $t = 30$ s, S_2 starts sharing (r_1, r_2) with S_1. Therefore, the sustainable rate of S_2 is diminished by half, becoming 500 kb/s, the same value of the sustainable rate of S_1. As a consequence, R_1 joins four layers, reaching 480 kb/s, and R_2 leaves one layer, decreasing its rate from 992 to 480 kb/s. R_3 and R_4 get a new report since the sustainable rate of their sessions increase more than 25% due to the decrease of the sustainable rate of S_2. However, R_3 and R_4 do not join l_6, maintaining their rate of 992 kb/s. This happens because the new sustainable rate is lower than 2.016 Mb/s, the total rate of the six layers. At $t = 130$ s, R_1 leaves and the sustainable rate of S_2 increases from 500 kb/s to 1 Mb/s. Therefore, R_2 gets a new report and grabs the bandwidth not being used by S_1 in (r_1, r_2), reaching again a rate of 992 kb/s. At $t = 170$ s, four more receivers join S_4, increasing the sustainable rate of the session from 1 Mb/s to 2.142 Mb/s. Therefore, the five receivers of S_4 receive a new report, which allows the four new receivers to join six layers and the previous receiver to join one more layer, reaching each of them a rate of 2.016 Mb/s. This shows that SAM allows late-join receivers to get the same quality as previous members of an existing session. Due to the higher audience size of S_4, the sustainable rates of S_2 and S_3 decrease to 428 kb/s each, which is insufficient to maintain their quality level. R_2 is the first to react to losses leaving l_5 with 2.54% of losses and leaving l_4 500 ms after that, with 9.68% of losses. Due to the reaction of R_2, the sustainable rate of S_3 increases to 760 kb/s, which is sufficient to maintain l_4. This shows that with SAM, competition for bandwidth does not increase quality oscillations.

Fig. 3 (right) shows SAM behavior when sessions have a VBR source. Each layer has a mean rate equal to its rate with CBR, a maximum and minimum rate 1.5 times higher

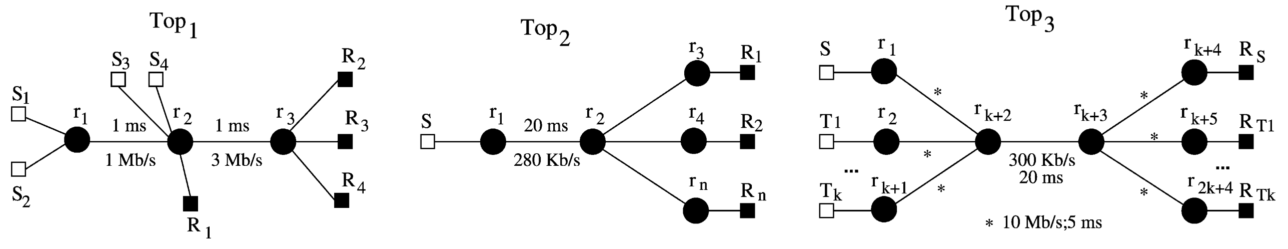


Fig. 2. Topologies.

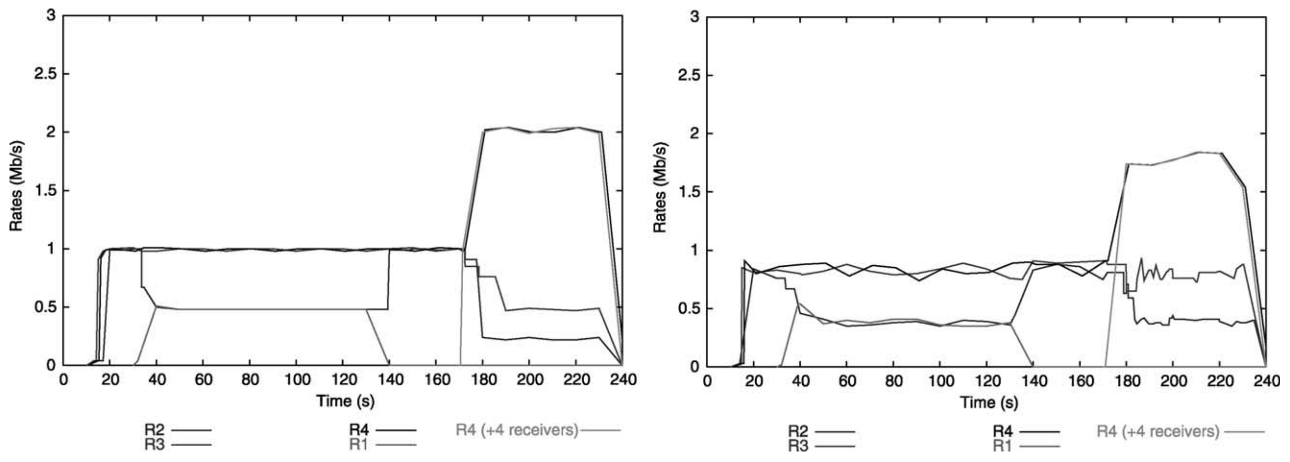


Fig. 3. Rate that receivers get when sessions have CBR (left) and VBR sources (right).

and lower than the mean rate, respectively, and a burst time of 2 s with a deviation of 0.5 s. Results show that SAM is able to adjust fairly the reception quality even when sessions have oscillatory rates. We can observe that S_2 and S_3 have a higher rate with VBR than with CBR after $t = 170$ s. This happens because they have a higher sustainable rate, since S_4 has a rate lower with VBR than with CBR.

The simulation with topology *Top1* of Fig. 2 evaluates the operation of SAM. Results shows that reports provided by SAPRA allow a simple adaptive mechanism, such as SAM, to keep receivers rate close to the sustainable rate of their sessions, guaranteeing inter-session fairness and increasing bandwidth utilization. Results also show that a loss threshold of 2.5% does not lead to quality oscillations.

In what concerns the convergence time, stability and fairness with TCP, Legout et al. [20] show that PLM performs better than RLM and RLC. Hence, we compare SAM with the results presented by Legout et al. for PLM. For that, we use the same scenarios used by Legout et al. These simulations use the topologies *Top2* and *Top3*, shown in Fig. 2.

We use the topology *Top2* shown in Fig. 2 to evaluate the time SAM takes to convergence to an optimal quality level, its accuracy and stability. We also show the performance that SAM would have, if receivers did not know in advance the rate of each layer. Links (r_2, r_n) have a bandwidth uniformly chosen between [500,1000] kb/s and a delay uniformly chosen between [5,150] ms. We use one session, S, and layers with a thin granularity of 50 kb/s. At $t = 5$ s, 20 receivers join S. From $t = 30$ to 50 s, a receiver joins S every 5 s. At $t = 80$ s, five more receivers join this session. Each receiver is positioned in a different leaf router. We use the packet and queue size used for PLM, i.e. packets with 500 bytes and queues with 20 packets.

Fig. 4 (left) shows that the first 20 receivers start to converge to their optimal rate at $t = 10$ s, 5 s after joining S, while late-join receivers wait a little less (3 s) to converge. This happens because receivers only start to converge after receiving the first report. First receivers wait longer, since the fair rate of S has to be computed for the entire path, and so the first report is originated by the node nearest to the source. Nevertheless, all receivers converge immediately to an optimal rate, which is maintained without losses.

Fig. 4 (right) shows that the convergence time would be slower if receivers did not know in advance the average rate of each layer. This happens because receivers would have to

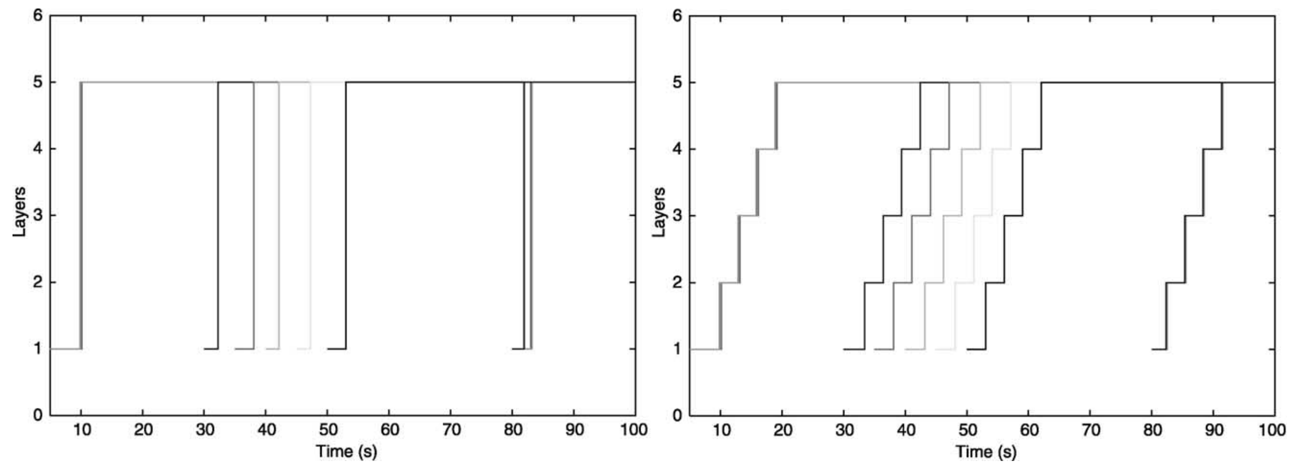


Fig. 4. Convergence time with (left) and without (right) knowledge of layers rates.

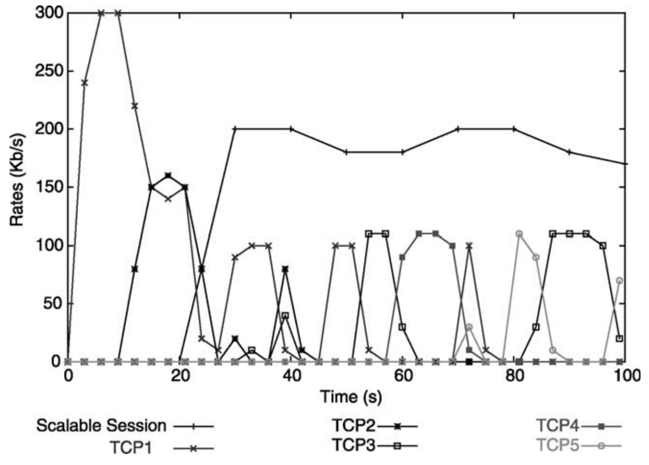


Fig. 5. Five Reno flows and one non-adaptive scalable session.

wait before joining each layer, in order to estimate the current rate of the session and predict the rate of the next higher layer. In these experiments, receivers use an exponential equation to estimate the average rate of each received layer, and predict that the next layer has a rate equal to the last joined layer.

This simulation shows that neither the audience size or late-joins influence the convergence time and stability of SAM. The results shown in Fig. 4 (left) are similar to the ones presented by Legout et al. for PLM [19], except that with PLM receivers start to converge 2 s after joining a session. This happens because with PLM receivers are notified only about the available bandwidth in the path and not about the sustainable rate of their session.

We use the topology *Top3* of Fig. 2 to evaluate the behavior of SAM in the presence of TCP flows. The two most common reference implementations for TCP are TCP Tahoe [13] and TCP Reno [14]. The first refers to TCP with the slow start, congestion avoidance, and fast re-transmit algorithms, first implemented in 4.3 BSD in 1988. The second refers to TCP with the earlier algorithms plus fast recovery, as implemented in 4.3 BSD in 1990. Although TCP Reno uses a fast recovery algorithm, it has performance problems when multiple packets are dropped from a window of data. These problems result from the need to await re-transmission timer expiration before re-initiating data flow. To overcome these performance problems, TCP Reno was extended to create the versions, TCP New-Reno and TCP Sack. With TCP New-Reno [8,9], partial ACKs, which acknowledges some but not all packets that were outstanding at the beginning of the fast recovery period, do not take TCP out of fast recovery. With TCP Sack [21], TCP Reno was extended with selective acknowledgments and selective retransmissions. Kevin Fall et al. [5] show that selective acknowledgments are not required to solve the TCP Reno performance problems when multiple packets are dropped. However, Kevin Fall et al. also show that the absence of selective acknowledgments limits the performance of TCP, namely when a large number of packets are dropped from a window of data. Without selective acknowledgments, TCP implementations are constrained to either retransmit at most one dropped packet per round-trip time, or to retransmit packets that might have already been successful delivered. TCP Reno and New-Reno use the first strategy, and Tahoe used the second one.

Fig. 5–7 show the results of our simulations using TCP Reno, New-Reno and Sack, respectively. These simulations are designed to highlight the behavior of SAM in the presence of several TCP connections with and without selective acknowledgment. In our simulations, SAPRA handles TCP flows as scalable sessions with one layer and one receiver. The topology *Top3* of Fig. 2 has one scalable session, S, and multiple TCP flows, T_1–T_5. S has one receiver, R_S, and 10 layers with granularity of 20 kb/s. S at $t = 20$ s, and the TCP flows start one every 60 s, starting at $t = 0$ s. The size of each packet is set to 500 bytes.

Fig. 5 shows what happens when the network do not provide any kind of fairness, and when non-TCP receivers, is this case a receiver of a scalable multimedia session, do

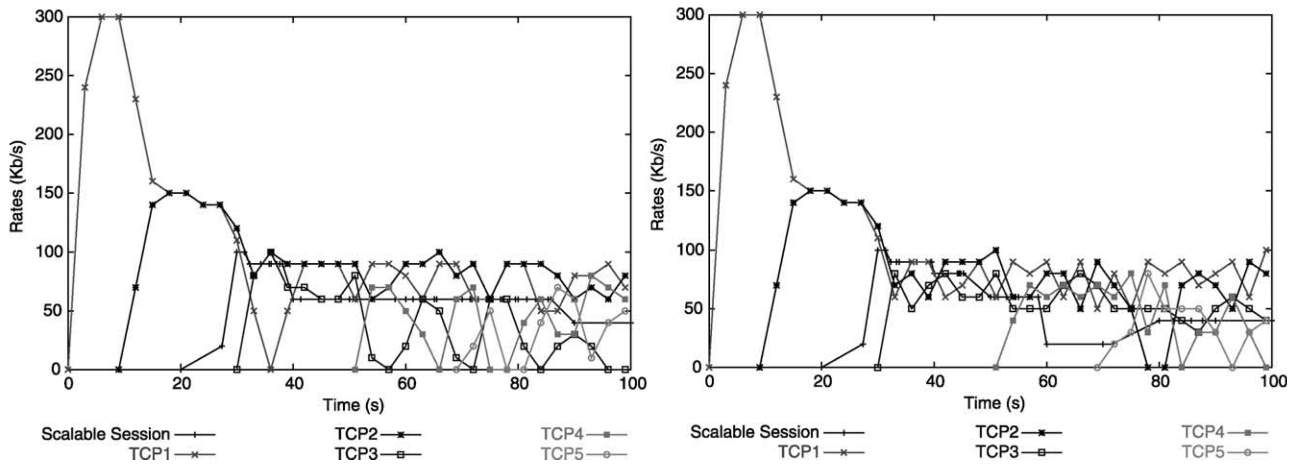
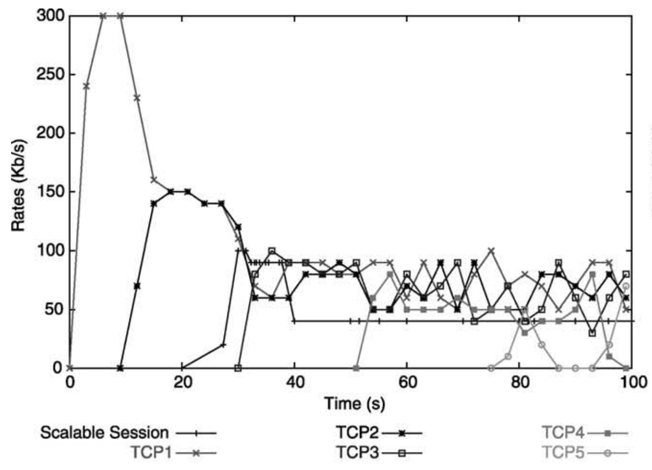


Fig. 6. Five Reno (left) and new-Reno (right) flows with one SAM session.



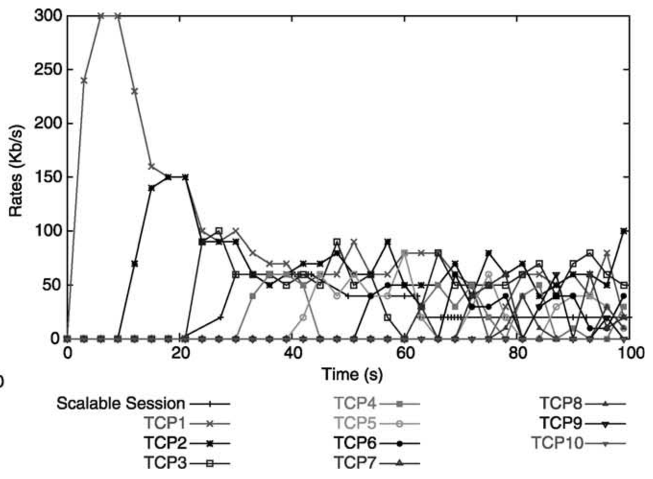
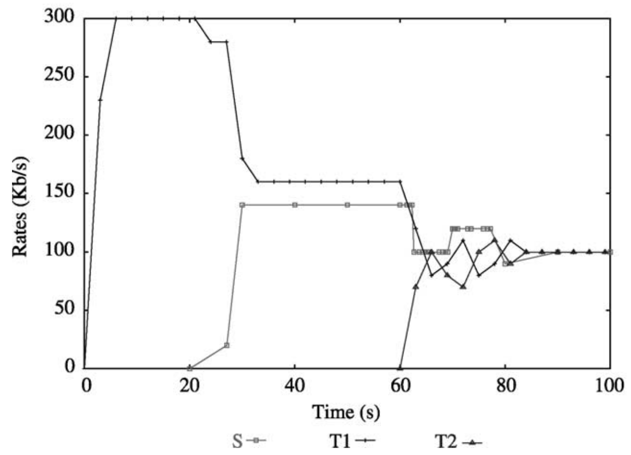


Fig. 7. Five (left) and ten (right) SACK flows with one SAM session.

not adapt when congestion occurs. As expected, the scalable session consumes all the network resources required by its sender maximum rate. In this simulation the sender maximum rate is 200 kb/s. In this scenario there is no fairness with TCP flows, which reduce their rate when congestion occurs, ending up sharing half of the link capacity between them.

Figs. 6 and 7 illustrate that SAM is fair with TCP, independently of the implemented TCP version. In these simulations, SAPRA equally distributes the service capacity on the shared link, since the scalable session has the same population of each TCP flow, i.e. one receiver. Besides this, SAPRA guarantees that each flow only uses its fair share of resources, allowing only the use of extra resources, if there are flows that are not using their entire share. These four figures show that R_S reacts to the congestion induced when each TCP flow starts, leaving the number of layers required to maintain the reception rate below the sustainable rate of its session. With a layer granularity of 20 kb/s, R_S has normally only to leave one layer for each new TCP flow. However, in some situations R_S can leave more than one layer when its sustainable rate is decreased by the presence of a new TCP flow, since we configured SAM with a short non-overreaction period and a high loss sensibility. This happens for instance with TCP Reno and TCP Sack, where R_S goes from five to three layers and five to two layers, respectively, when the third TCP connection starts. With TCP Reno, after the fifth TCP flow, SAM leaves another layer since the new sustainable rate of 50 kb/s is only enough for two layers. With the New-Reno TCP version, R_S only leaves one layer after the third TCP, but has a sensitive reaction to losses when the fourth TCP starts, leaving two layers and reducing its receiving rate to 20 kb/s. However, the new sustainable rate of 50 kb/s, that R_S gets after the beginning of the fifth TCP flow, allows the scalable receiver to re-adjust its reception rate by adding one more layer. The TCP fairness property of SAM still exists when we double the number of TCP flows, as shown in Fig. 7 (right).

These results show that the network-based adaptation mechanism used by SAM does not induce losses in other flows and increases the system stability. This property is due to the fact that the network information provided by SAPRA allows SAM to attain a good perceptual quality of the displayed video stream without the aggressiveness of



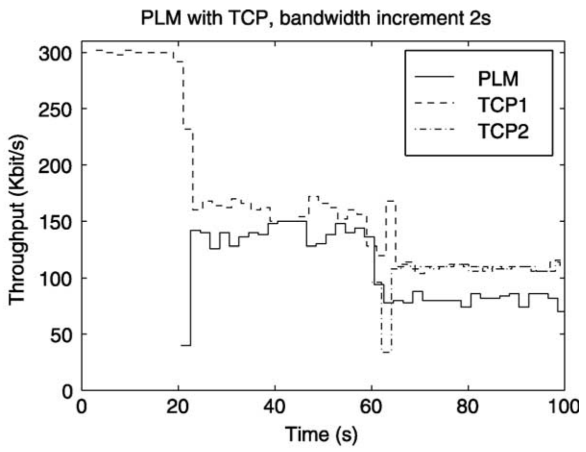


Fig. 8. Fairness with TCP: comparison of SAM (left) and PLM (right) behavior.

Table 1
Link (r_4, r_5) utilization in kb/s

	60 (s)	63 (s)	66 (s)	69 (s)	72 (s)	75 (s)	78 (s)	81 (s)	84 (s)
S	140	100	100	120	120	120	120	90	100
T_1	160	120	80	90	110	80	70	110	100
T_2	0	70	100	80	70	100	110	90	100
Total	300	290	280	290	300	300	300	290	300

a probing mechanism. This result also show that SAM is very sensitive to losses, but this is due to the conservative configuration with a short non-overreaction period and a small percentage of allowed losses. As said before, these configuration parameter can be adjusted in each implementation of SAM.

In order to study in more detail the behavior of SAM and to compare it to PLM, we use the topology *Top3* of Fig. 2 with one scalable session, S, and two TCP flows, T_1 and T_2. S has one receiver, R_S, and layers with granularity of 20 kb/s. T_1 starts at $t = 0$ s, S at $t = 20$ s and T_2 at $t = 60$ s. We use only two TCP flows, since this was the scenario used by Legout et al. to analyze the behavior of PLM.

Fig. 8 (left)[1] confirm SAM fairness in the presence of TCP. R_S joins the lowest layer at $t = 20$ s, reaching a rate of 20 kb/s. At $t = 27$ s, it increases its rate after receiving the first report. Since S has only one receiver, the bandwidth of link (r_4, r_5) is equally divided between S and T_1. Therefore, after $t = 27$ s, S and T_1 have a fair rate of 150 kb/s, and so R_S joins seven layers, reaching a rate of 140 kb/s. Since S does not use 10 kb/s of its fair share, the rate of T_1 reaches 160 kb/s. When T_2 starts, the fair rates of S, T_1 and T_2 reach 100 kb/s. R_S starts to experience losses and decreases its rate to 100 kb/s, but it maintains seven layers since losses are lower than 2.5%.

Due to T_1 and T_2 oscillations until $t = 84$ s, R_S grabs the bandwidth not being used by the TCP flows, increasing its rate to 120 kb/s. From then until the end of the simulation, the rate of S, T_1 and T_2 stabilizes at 100 kb/s. In the meantime, R_S leaves layer seven with losses of 2.91%, but maintains the rate of 120 kb/s, which is the maximum possible rate with six layers. At $t = 80$ s, R_S leaves layer six, since losses reach 3.28%. Table 1 shows that the bandwidth of link (r_4, r_5) is completely used, except for the interval from $t = 63$ to 81 s, where the utilization rate decreases to 98.1% due to T_1 and T_2 oscillations.

By comparing the results shown in Fig. 8, we can see that SAM and PLM are fair in the presence of TCP, but SAM presents smaller quality oscillations.

5. Conclusion

Receiver-driven adaptive mechanisms can accommodate heterogeneity when combined with scalable encoding. However, receivers are only motivated to adapt if the network guarantees a fair distribution of bandwidth and also punishes receivers that do not adjust their rate in case of congestion.

This paper describes and evaluates SAM, a receiver-driven adaptive mechanism based upon SAPRA. SAPRA is a signaling protocol that has the required punishment and fairness properties. SAM controls the perceptual quality of the displayed video stream by joining and dropping layers. The sustainable rate, provided by SAPRA, indicates to SAM the maximum number of layers that receivers can join, while the measured packet losses triggers SAM to drop layers.

When analyzing SAM behavior, simulation results show that receivers get always a rate near the sustainable rate of their session, independently of the number of sessions and their audience size. Results also show that SAM has small convergence time and remains stable even in the presence of bursty traffic, such as VBR.

In the presence of TCP flows, results confirm SAM fairness with TCP, independently of the TCP version implemented and the number of TCP flows. Simulations show that SAM is not aggressive when increasing the reception rate, contributing to the stability of the system. Compared to PLM, SAM has less quality oscillations and requires few changes in the network structure.

As major improvement, SAM motivates receivers to adapt, since SAPRA guarantees a fair distribution of bandwidth and the punishment of high-rate sessions.

As future work, we intend to study the behavior of SAPRA and SAM in mobile environments, namely to analyze the effect of hand-offs on the efficiency of the proposed fairness protocol and receiver-driven adaptive mechanism.

Acknowledgements

This work is supported by POSI—Programa Operacional Sociedade de Informação of Portuguese Fundação para a Ciência e Tecnologia and European Union FEDER.

References

[1] D. Bertsekas, R. Gallager, Data Networks, Prentice-Hall, Englewood Cliffs, NJ, 1987.

[1] Fig. 8 (right) was taken from [19] with the author's permission.

[2] S. Bhattacharyya, C. Diot, L. Giuliano, R. Rockell, J. Meylor, D. Meyer, G. Shepherd, B. Haberman, An Overview of Source-Specific Multicast (SSM), Internet Draft, Internet Engineering Task Force, Work in progress, November 2002.

[3] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, W. Weiss, An Architecture for Differentiated Service, Request for Comments 2475, Internet Engineering Task Force, December, 1998.

[4] D.D. Clark, M.S. Blumenthal, Rethinking the design of the Internet: the end to end arguments vs. the brave new world, In Proceedings of 28th Research Conference on Communication, Information and Internet Policy, Alexandria, Virginia, September 2000.

[5] K. Fall, S. Floyd, Simulation-based Comparisons of Tahoe, Reno, and SACK TCP, ACM Computer Communication Review 26 (3) (1996) 5–21.

[6] M. Handley, C. Perkins, E. Whelan, Session Announcement Protocol, Request for Comments 2974, Internet Engineering Task Force, October 2000.

[7] J. Heinanen, F. Baker, W. Weiss, J. Wroclawski, Assured Forwarding PHB Group, Request for Comments 2597, Internet Engineering Task Force, June 1999.

[8] J.C. Hoe, Start-up Dynamics of TCP's Congestion Control and Avoidance Schemes, Master Thesis, Department of Electrical Engineering and Computer Science, MIT, Cambridge, MA, May 1995.

[9] J.C. Hoe, Improving the Start-Up Behavior of a Congestion Control Scheme for TCP, In SIGCOMM Symposium on Communications Architectures and Protocols, Stanford, CA, USA, August 1996, pp. 270–280.

[10] J. Kimura, F.A. Tobagi, J.-M. Pulido, P.J. Emstad, Perceived quality and bandwidth characterization of layered MPEG-2 video encoding, In Proceedings of SPIE International Symposium, Boston, MA, USA, September 1999.

[11] ITU-500-R, Methodology for the Subjective Assessment of Quality of Television Pictures, ITU-500-R Recommendation bt.500-8, ITU, 1998.

[12] V. Jacobson, K. Nichols, K. Poduri, An Expedited Forwarding PHB, RFC 2598, Internet Engineering Task Force, June 1999.

[13] V. Jacobson, Congestion avoidance and control, ACM Computer Communication Review 18 (4) (1988) 314–329. Proceedings of the Sigcomm '88 Symposium in Stanford, CA, August, 1988.

[14] V. Jacobson, Modified TCP Congestion Avoidance Algorithm, LBL, Technical Report, LBL, April 1990.

[15] T. Jiang, E. Zegura, M. Ammar, Inter-Receiver Fair Multicast Communication over the Internet, Basking Ridge, New Jersey, USA, June 1999.

[16] M. Johanson, Scalable video conferencing using subband transform coding and layered multicast transmission, In Proceedings of International Conference on Signal Processing Applications and Technology (ICSPAT), Orlando, Florida, USA, November 1999.

[17] S. Keshav, A control-theoretic approach to flow control, In Proceedings of SIGCOMM Symposium on Communications Architectures and Protocols, Zürich, Switzerland, September 1991. ACM also in Computer Communication Review, 21(4), September 1991.

[18] A. Legout, E. Biersack, Pathological behaviors for RLM and RLC, In Proceedings of the International Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV), Chapel Hill, NC, USA, June 2000.

[19] A. Legout, E.W. Biersack, PLM: fast convergence for cumulative layered multicast transmission schemes, Technical Report, Institut Eurém, Sophia-Antipolis, France, November, 1999.

[20] A. Legout, E.W. Biersack, PLM: fast convergence for cumulative layered multicast transmission schemes, In Proceedings of ACM SIGMETRICS performance evaluation review, Santa Clara, CA, USA, June 2000.

[21] M. Mathis, J. Mahdavi, S. Floyd, A. Romanow, TCP selective acknowledgement options, RFC 2018, Internet Engineering Task Force, October 1996.

[22] S. McCanne, V. Jacobson, M. Vetterli, Receiver-driven layered multicast, In Proceedings of SIGCOMM Symposium on Communications Architectures and Protocols, Palo Alto, CA, USA, August, 1996, pp. 117–130.

[23] P. Mendes, H. Schulzrinne, E. Monteiro, Session-aware popularity resource allocation for assured differentiated services, In Proceedings of the Second IFIP-TC6 Networking Conference, Pisa, Italy, May 2002.

[24] P. Mendes, H. Schulzrinne, E. Monteiro, Session-aware popularity resource allocation for assured differentiated services, IEEE Communications Magazine 40 (9) (2002) 104–111.

[25] P. Mendes, H. Schulzrinne, E. Monteiro, Signaling protocol for session-aware popularity-based resource allocation, In Proceedings of the IFIP/IEEE International Conference on Management of Multimedia Networks and Services (MMNS), Santa Barbara, CA, USA, October 2002.

[26] A.M. Odlyzko, The slow evolution of electronic publishing, in: A.J. Meadows, F. Rowland (Eds.), Electronic Publishing '97: New Models and Opportunities, ICCC Press, 1997, pp. 4–18.

[27] A.K. Parekh, R.G. Gallager, A generalized processor sharing approach to flow control in integrated services networks: the single node case, Journal of IEEE/ACM Transactions on Networking 1 (3) (1993) 344–357.

[28] G. Raman, J. Griffioen, G. Hjalmtysson, C. Sreenan, Stability and fairness issues in layered multicast, In Proceedings of the International Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV), Basking Ridge, NJ, USA, June 1999.

[29] K. Rose, S.L. Regunathan, Toward optimality in scalable predictive coding, Journal of IEEE Transactions on Image Processing 10 (7) (2001) 965–976.

[30] D. Rubenstein, J. Kurose, D. Towsley. The impact of multicast layering on network fairness, In Proceedings of SIGCOMM Symposium on Communications Architectures and Protocols, Cambridge, MA, USA, September 1999.

[31] J.H. Saltzer, D.P. Reed, D.D. Clark, End-to-end arguments in system design, ACM Transactions on Computer Systems 2 (4) (1984) 277–288.

[32] D. Taubman, A. Zakhor, Multirate 3-D subband coding of video, Journal of IEEE Transactions on Image Processing 3 (5) (1994) 572–588.

[33] L. Vicisano, L. Rizzo, J. Crowcroft. TCP-like congestion control for layered multicast data transfer, In Proceedings of the Conference on Computer Communications (IEEE Infocom), San Francisco, CA, USA, March/April 1998.