Signaling Protocol for Session-Aware Popularity-Based Resource Allocation*

Paulo Mendes[1,2], Henning Schulzrinne[1], and Edmundo Monteiro[2]

[1] Department of Computer Science, Columbia University
New York, NY 10027, USA
{mendes,schulzrinne}@cs.columbia.edu
[2] CISUC, Department of Informatics Engineering, University of Coimbra
3030 Coimbra, Portugal,
{pmendes,edmundo}@dei.uc.pt

Abstract. The Differentiated Services model maps traffic into services with different quality levels. However, flows are treated unfairly inside each service, since the Differentiated Services model lacks a policy to distribute bandwidth between flows that form the same service aggregate traffic. Therefore, we present a signaling protocol that fairly distributes the bandwidth assigned to each service, among scalable multimedia sessions in a multicast environment. Fairness is achieved allocating bandwidth based upon the audience size of each session. We evaluate the efficiency of the proposed protocol using theoretical analysis and simulation.

Keywords: QoS; multicast; differentiated services; scalable sessions; fairness.

1 Introduction

Current popular Internet multimedia applications, such as RealNetworks SureStream and Microsoft Windows Media, stream the same content at different rates, depending on receiver capabilities. However, even if they use multicast, each stream in this multi-rate encoding contains a complete encoding, thus wasting bandwidth. This problem can be solved by using scalable encoding [17,9]. Scalable encoding divides a video stream into cumulative layers with different rates and importance. Thus, the stream rate is then the sum of its layers. Sources send only one stream to all receivers, mapping each layer to a different multicast group. All layers belonging to the same stream form a *session*.

Due to their real-time characteristics, scalable sessions need quality guarantees from networks, namely: *inter-session* fairness, the ability to guarantee a fair distribution of bandwidth between sessions sharing a service; *intra-session* fairness, the ability to respect the importance of each layer of a session; and punishment of *high-rate* sessions, i.e., sessions with a rate higher than their fair share

The original version of this chapter was revised: The copyright line was incorrect.This has been corrected. The Erratum to this chapter is available at DOI: 10.1007/978-3-540-45812-8_28

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

K.C. Almeroth and M. Hasan (Eds.): MMNS 2002, LNCS 2496, pp. 101–113, 2002.

of bandwidth. The current Differentiated Services (DS) model [3] aggregates traffic into services with different priorities at the boundaries of each network domain. Among the services DS can provide, the Assured Forwarding PHB (AF) [5] is ideal for transporting scalable sessions, since flows are assigned different drop precedences. Although AF services provide intra-session fairness, the DS model lacks the other two properties. Therefore, we propose a protocol named *Session-Aware Popularity-based Resource Allocation* (SAPRA) that allows a fair allocation of resources in each DS 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. To achieve its goal, SAPRA adds agents and markers to edge routers. Agents manage sessions to provide inter-session fairness; markers deal with layers, providing intra-session fairness. Only edge routers are changed, since SAPRA handles individual traffic aggregated in each service. The behavior of agents and markers is described and evaluated in [13]. In this paper, we describe and study the SAPRA signaling protocol used by agents to exchange information about sessions, allowing a fair distribution of resources in the path of each session.

The remainder of the paper is organized as follows. Section 2 briefly describes work related to SAPRA. In Section 3, we describe the SAPRA protocol. Section 4 presents a theoretical analysis and simulation of SAPRA. Finally, Section 5 presents some conclusions and future work.

2 Related Work

The amount of resources reserved for each DS service depends on service-level agreements (SLAs) between domains. The management of multicast traffic can be done with static SLAs for flows with more than one egress router. However, since multicast traffic is both heterogeneous and dynamic, static SLAs needs over-provisioning of resources in the domain. As alternative, dynamic SLAs can be used. These can be managed by bandwidth brokers (BB) or intra-domain signaling protocols to reserve resources where they are needed. However, these approaches do not distribute reserved service resources across several domains, among sessions that constitute the service aggregate traffic. To solve this issue some proposals present inter-session fairness mechanisms based on the max-min fairness definition [2], but this definition cannot exist with discrete set of rates [15]. Sarkar et al. present a fairness algorithm [16] that considers discrete set of rates, but does not consider the population of sessions, and its scalability and efficiency is not proved for Internet-like scenarios. Audience of sessions is pondered by Legout et al. [10], but their proposal does not consider intra-session fairness for scalable sources, requires changes in all routers, does not maximize resources,

and does not punish high rate flows. Li et al. [11] present another inter-session fairness mechanism, but it is also based on the max-min fairness definition, assumes only one shared link, and does not consider audience of sessions and the importance of their layers.

3 SAPRA Protocol

In this section we describe SAPRA. We assume that edge routers have an accurate notion about resources reserved for each service, being the mechanism implemented in each domain a policy issue beyond the scope of this paper. Multicast branch points can be located in edge and interior routers, but receivers are attached only to leaf edge routers.

SAPRA is most suited for long-lived scalable sessions with large audience size, such as Internet TV and periodic near-video-on-demand systems [1]. Each layer is identified by a source-specific multicast (SSM) channel [6]. SAPRA also integrates non-scalable sessions and unicast traffic. The former are treated like scalable sessions with one layer, and the latter as sessions with one layer and one receiver.

Not all DS domains have to implement SAPRA, because SAPRA messages are exchanged between SAPRA-speaking agents. However, receivers in non-SAPRA domains do not count toward the session population and thus sessions with a big audience outside of SAPRA domains have a small share of resources.

Receivers can join sessions by, for instance, listening to Session Announcement Protocol (SAP) [4] messages. Receivers first join the multicast group for the most important (lowest) layer and then increase their reception quality by joining additional (higher) layers. Each layer requires all layers below it. Agents in leaf edge routers use IGMPv3 "State-Changes" reports, issued when receivers join a layer, to measure the local audience size of each session.

SAPRA allocates resources to sessions, rather than managing multicast groups without concern about their relationship. If the structure of sessions would be kept hidden, agents would not be able to implement inter-session fairness, markers would fail to provide intra-session fairness, and sessions would have lower and less stable quality levels. We propose two methods for agents in leaf routers to collect information about the composition of sessions, using consecutive address ranges and IGMP extensions. With SSM, multicast addresses are locally allocated by each source, from a pool of 2^{24} addresses in IPv4 and 2^{32} addresses in IPv6. For the first method, each sender allocates consecutive multicast addresses to all layers inside a session and keeps a gap of at least one address between sessions. Since receivers join layers sequentially, an agent can detect a new session if its multicast address differs by at least two from any other session address. Alternatively, if the number of layers per session can be bounded, the address bits can be divided into high-order session and low-order layer bits. For the second method, we add the address of the most important layer to IGMPv3 [7], using the "auxiliary" data field. The address of the most important layer then identifies the session. For both methods, agents deduce the

relationship between layers in a session by the order receivers join them. Details can be found in [13].

We define downstream as the direction from sources to receivers, and the opposite as upstream. We also define the *Downstream Fair Rate* (DFR), the *Local Fair Rate* (LFR) and the *Used Fair Rate* (UFR) of sessions in a link. DFR is the fair rate that a session has downstream that link, LFR is the fair rate of the session in the link and UFR is the lesser of LFR and DFR. By using UFR instead of LFR, sessions that do not use downstream resources do not waste local bandwidth.

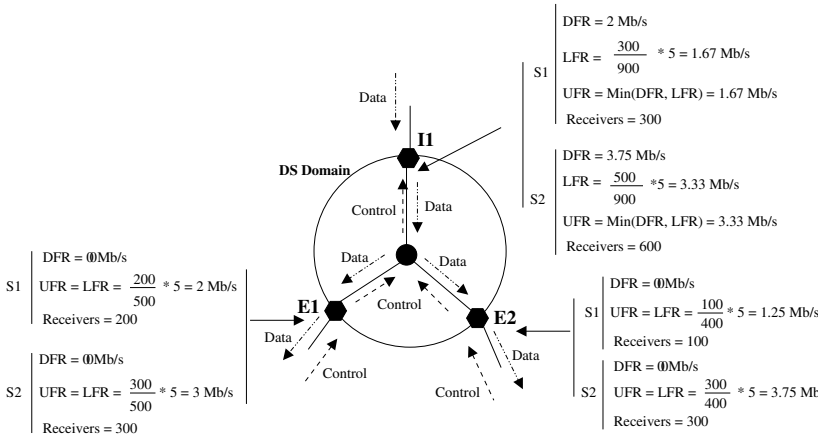


Figure 1. Fair rate computation in edge routers

Fig. 1 shows how agents compute the UFR of two sessions, S_1 and S_2, in a domain with three edge routers. This computation requires information about sessions population and DFR, which agents collect from downstream neighbors using the SAPRA signalling protocol. As we described in [13], agents compute the LFR dividing the resources of the shared service among sessions by assigning more resources to sessions with bigger audience size. For instance, assuming that the service has 5 Mb/s in each link, the LFR of S_1 is 1.25 Mb/s in E_2, since S_1 has 100 receivers and the total population is 400. If most of the sessions have few receivers, SAPRA allocates a small share of resources to each session, which can be useless to assure the minimum quality level that the source wants to provide to receivers. If agents are informed about this requirement, for instance by BB, they can compute a zero LFR for new sessions, assuring a minimum share to existing sessions. This can be an extension to SAPRA, considering pricing issues, which is beyond the scope of this paper.

In this example, the UFR of each session is equal to its LFR at egress routers, since we assume an infinite DFR. At the ingress router, the population of a session is the sum of its receivers in each egress router and its DFR is the maximum value of its UFR in each egress router. This maximum value satisfies

the heterogeneous requirements of receivers downstream different egress routers without congesting links downstream the domain, since packets will be dropped at egress routers in conformity with the local UFR.

In each downstream link of edge routers, UFRs are passed to the marker. Packets are marked in or out of profile depending on the relationship between the estimated rate and the UFR of sessions, being marked out of profile starting by the least important layer [13].

3.1 Overview of Protocol Operation

Agents use two messages, UPDATE and SYNC, to propagate information. UPDATE messages deliver the UFR and population size of each session to upstream neighbors, allowing agents along the path to compute fair rates. SYNC messages propagate information about the quality of sessions to downstream neighbors. When session listeners get a SYNC message, they can adapt to quality changes.

Messages are exchanged only between neighboring agents, increasing scalability. For robustness, SAPRA runs over TCP, eliminating the need to implement fragmentation and re-transmission. UPDATEs are forwarded using routing information of the Border Gateway Protocol (BGP) [14] and SYNCs follow the path of the UPDATEs in reverse, similar to RSVP PATH and RESV messages.

Agents send messages periodically. Messages include information about non-stationary sessions, i.e., sessions whose state changed significantly since the last time they were included in a message. So, agents do not send any message if there are only stationary sessions. The minimum state variation is set to 25%, which appears to be a good compromise between the number of messages and the propagation of accurate information. This restriction allows the use of 1 s intervals, improving the protocol accuracy: since sessions are long and their population normally varies significantly only when they begin and end, intervals as short as 1 s are enough to propagate significant state changes. To avoid synchronized messages, intervals are varied by a small random of 10%.

As an alternative, agents could send messages, not periodically, but only after receiving a message of the same type, a SYNC from the upstream neighbor in the path to the session source, or an UPDATE from each downstream neighbor where that session is present. However, this approach increases UPDATEs propagation delay since agents can take a long time to gather information about each session from all downstream neighbors. This happens, for example, when there is a different number of routers between the agent and each neighbor, or when neighbors send information about the same session at different times.

UPDATE messages are triggered when an agent receives the first join from local receivers or the first UPDATE. Stationary sessions are identified by the variation of their population or UFR, considering the total population and total UFR in the link where the UPDATE is going to be sent. When a session ends, its information is always included in the next UPDATE allowing the immediate release of unnecessary state in upstream agents. Sessions have hard-state, since state starts and ends with the reception of UPDATEs. However state is also refreshed by periodic complete UPDATEs that include information about all sessions. This

soft-state property makes SAPRA cope with routing changes and link failures, so the default values for the interval of complete UPDATEs and for the time agents wait before deleting state were chosen based upon BGP variables: the former is 30 s, the value of the BGP `keep_alive_time` variable, and the latter 90 s, the value of the BGP `hold_time` variable. When agents receive complete UPDATEs, they only update the state of those sessions whose DFR varies more than 25% to avoid computations that do not affect the local distribution of resources. The population size is always updated, to avoid cumulative rounding errors.

SYNC messages are triggered when an agent with local sources receives the first UPDATE, or when an agent without local sources receives the first SYNC. Stationary sessions are identified by the variation of their quality. In this paper, we consider that the quality information carried by SYNCs is the UFR of sessions. If the source of a session is local, that information is the UFR of that session in the downstream link, otherwise it is the minimum value between the local and upstream UFR.

3.2 Example of Protocol Operation

Fig. 2 illustrates how SAPRA works, by using two sessions, S_1 and S_2. We label edge router i as r_i, the agent on that router as a_i, and the link between r_i and r_j as (r_i, r_j). The notation $U_n@t$ indicates that the nth UPDATE is being sent at time t; $Y_n@t$ has a similar meaning for SYNCs. We assume that intra-domain links have 10 Mb/s of bandwidth and inter-domain links (r_1, r_2), (r_5, r_8) and (r_4, r_6) have 5 Mb/s, 3 ,Mb/s, and 4 Mb/s of bandwidth, respectively.

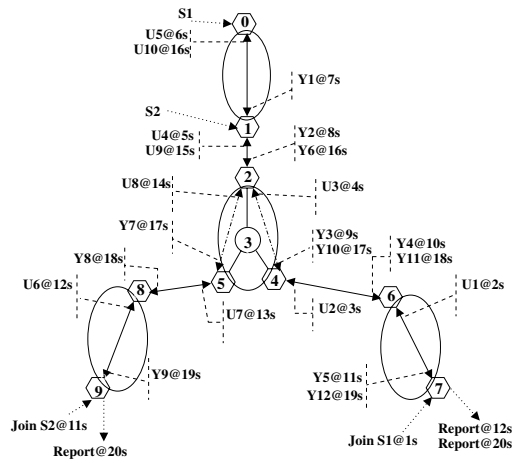


Figure 2. Protocol operation

When receivers join S_1, a_7 sends U_1 toward a_0. The reception of U_4 at a_1 triggers the sending of SYNCs, since a_1 has a local source. However, since the

source does not belong to S_1, a_1 does not send a SYNC at second 6. The first SYNC is sent by a_0 toward a_7 at second 7, with 4 Mb/s as minimum UFR of S_1 on the tree, the UFR on (r_4, r_6). Receivers on r_7 get a reports 11 s after having joined S_1. When receivers join S_2, a_1 receives U_9 at second 15. At that moment, S_1 and S_2 have an UFR of 2.5 Mb/s on (r_1, r_2). Since the minimum UFR of S_1 decreased more than 25%, a_1 sends U_{10} to a_0 at second 16. At the same time, a_1 sends Y_6 with the UFR of S_2. Since the UFR of S_1 differs more than 25% from the value sent in the previous SYNC, Y_6 also includes the UFR of S_1. The agent a_2 sends Y_7 to a_5 with the UFR of S_2 and Y_{10} to a_4 with the UFR of S_1. Hence, a_9 receives the UFR of S_2 8 s after receives joined the session and S_1 receivers on r_7 get a second report at second 20. After second 20, messages are suppressed, because there are no changes in population or in fair rates. However, agents send a complete UPDATE at second 30, the default interval.

4 Protocol Evaluation

4.1 Theoretical Analysis

To evaluate SAPRA, we analyze the memory required to store state, the bandwidth overhead, the time to update UFR of sessions in all agents, and the time to notify receivers about the quality variation of their session.

For N sessions, agents need to store $O(N)$ of state. The stored state does not depend on the number of receivers, increasing SAPRA scalability.

The bandwidth overhead is reduced, because some messages can be suppressed and they are only exchanged between neighbors. This means that the overhead is independent from the network size, being dependent only on the number of sessions and the size of intervals. We analyze the protocol for a worst-case scenario with short intervals (1 s) and high number of sessions (1,000). We assume that all sessions are present on all links, that each session has three layers with total rate of 1.8 Mb/s, and that their population is very dynamic, with significant changes every second. Sessions are decribed by their Id, source address and fair rate (12 bytes), and layers by their Id, group address and number of receivers (12 bytes). Since UPDATEs carry 48 bytes (session plus three layers) and SYNCs 12 bytes (only session) per session, the control rate between two agents is 480 kb/s. Also because the total data rate in a link is 1,800 Mb/s, the ratio between control and data rate is of 0.03%. Even with low-rate sessions the overhead is insignificant: if we assume a data rate of 64 kb/s, the protocol overhead is of 0.75%.

The time taken to update UFR of sessions in all agents depends on the number of agents in each session path and if agents were already triggered to start sending messages. Time required to notify receivers depends on one additional factor: if the minimum UFR of the session, in the path to its source, increases or decreases. The maximum time to update UFR of a session is given by $(n_a - 1)\, a_s$, and the maximum time to deliver a report to its receivers by $(2\,(n_a - 1) + 1)\, a_s$, where n_a is the number of agents in the session path and a_s the sending

interval. This maximum time occurs only during the first update, because the first UPDATE triggers all agents to start sending messages being delayed by the total a_s in each agent. After this, the delay is lower than a_s. So, for a path with six domains and 12 agents, the maximum time required to update UFR is 11 s and receivers get their first report 23 s after joining their session. We assume that there is only one DS domain per Autonomous Systems (AS) and that a path can have a maximum length of six AS, since the percentage of longer paths is very low. For instance, a November 2001 study of 60,978 different Autonomous Systems (AS) [18] showed that there are only 0.5% of paths with size of seven AS and 0.1% with size of eight AS.

When the UFR increases, if it remains higher than the upstream fair rate in all agents, the minimum UFR is only established by the agent closest to the source, and so the time required to notify receivers is proportional to the path length. Hence, notifications are faster when the minimum UFR is decreased by an agent that is not the agent closest to the source, because that agent immediately sends a SYNC and so n_a is lower than the number of agents in the path. This is possible because in each stabilized network branch the upstream fair rate in each link is equal to the minimum UFR on the branch. If changes happen in different branches at the same time, stabilization of fair rates is achieved first in each branch and only afterwards in the entire network. On the one hand, slower notification of higher minimum UFR leads to a higher stability, since receivers only join layers when resources of their session are updated in the entire path. On the other hand, faster notification of lower minimum UFR optimize resource utilization, since receivers leave layers more quickly, reducing network traffic.

4.2 Simulations

In this section we present simulations (using NS) that show SAPRA ability to allocate resources across several domains. We use a scenario with ten domains, one domain per AS as shown in Fig. 3, where the longest path has six domains.

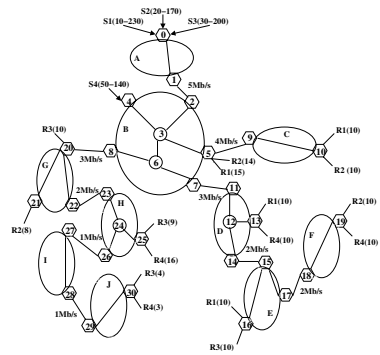


Figure 3. Simulation scenario

The topology has four sessions, S_1 to S_4, in different domains. Domains have multicast branches either in ingress or interior routers, and receivers either in ingress or egress routers. Due to lack of information about traffic distribution among ASs, we use the distribution of addresses by AS distance [18] to decide upon receiver locations. Hence, we place 26%, 40%, 26%, 9%, and 2.5% of receivers in domains at a distance of two, three, four, five, and six domains from their source, respectively. In Fig.3, $R_x(Y)$ denotes Y receivers of S_x.

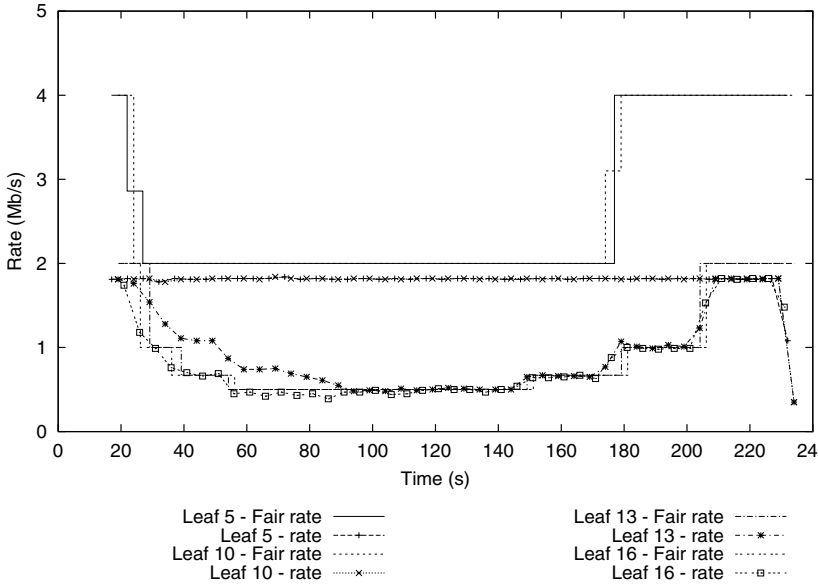
We assume that service capacity in inter-domain links is equal to the link bandwidth and that intra-domain links have enough bandwidth to avoid congestion. We also assume that each queue has a size of 64 packets, which is the default value in Cisco IOS 12.2, and that interior routers have FIFO queues, since these are the simplest ones. We use data packets with 1,000 bytes since this is a value between the MTU of dial-up connections, 576 bytes, and the MTU of ethernet and high speed connections, 1,500 bytes. Since control messages are exchanged between routers, UPDATE and SYNC packets have a size of 1,500 bytes, which is the default MTU in Cisco routers.

Session S_1 spans seconds 10 through 230, S_2 20 to 170, S_3 30 to 200, and S_4 50 to 150. Each session has three layers with total rate of 1.8 Mb/s. The most important layer has 303 kb/s, the medium 606 kb/s, and the least important one 909 kb/s. Although SAPRA supports any number of layers, three layers provide a good quality/bandwidth trade-off and additional layers provide only marginal improvements [8]. The simulation covers 240 s, which is enough to identify transient state when sessions start and end. For simplicity, we assume that receivers of the same session join and leave simultaneously. For longer simulations, sessions remain stable for a longer time and the time needed for their state to stabilize does not change. Results for 600 s simulations are available in [12].

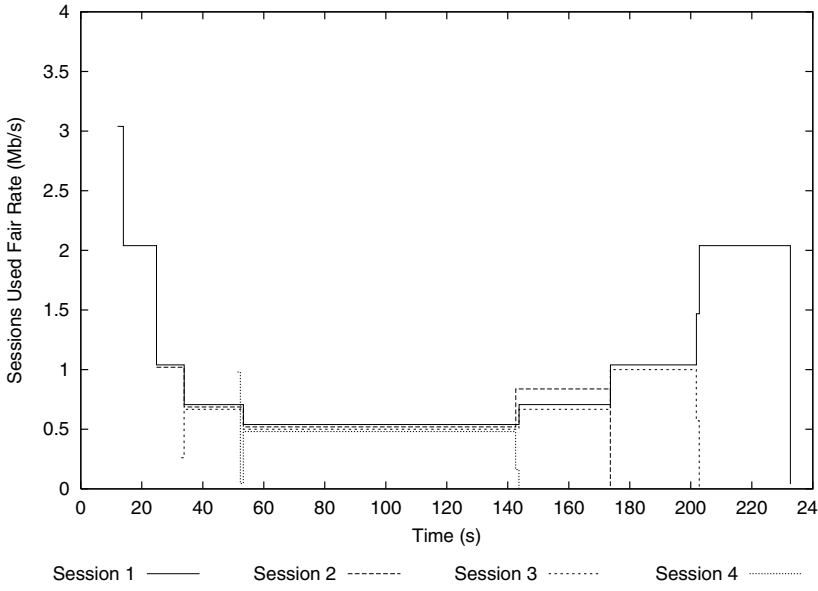
We analyze the protocol efficiency based on the session rate monitored by S_1 receivers and the minimum UFR they get from SAPRA, Fig.4 a), and based on the UFR of sessions on (r_7, r_{11}), Fig.4 b). Results for the remainder sessions and inter-domain links are available in [12].

Fig.4 a) shows that the rate of S_1 is always kept below its minimum UFR, with the exception of the rate monitored by r_{13} receivers from seconds 29 to 89. This happens because the UFR of all sessions on (r_7, r_{11}) is always lower than the service capacity on that link, as is shown in Fig.4 b), and so a percentage of out of profile packets from all sessions pass through r_7 allowing r_{13} receivers to get a rate higher than the minimum UFR. Nevertheless, Fig.4 a) also shows that S_1 rate on r_{13} decreases in that interval. This is due to the punishment mechanism that filters S_1 when this session is identified as high-rate session during the congestion of (r_7, r_{11}).

Fig. 4 b) shows that SAPRA is able to distribute resources in each link, decreasing UFR of sessions to release resources for new sessions or increasing them to grab resources that were used by sessions that ended. The only irregular behavior is the initial variation of the UFR of S_4. This happens because in this simulation a_7 gets information about receivers of S_4 on r_{13} and r_{19} at different moments: at second 51, a_7 has only knowledge about receivers on r_{13}, because



(a)



(b)

Figure 4. a) rate and minimum UFR of S_1; b) UFR of all sessions on (r_7, r_{11})

the UPDATE from a_{13} arrives to a_7 first than the one from a_{19}. At that time, the LFR of S_2, S_3 and S_4 is 0.6 Mb/s and the one of S_1 is 1.2 Mb/s, since S_1 has 20 receivers, and S_2, S_3 and S_4 have 10 receivers each. However, since S_1 has a DFR of 0.6 Mb/s, it releases 0.6 Mb/s from its LFR, which are used to increase the UFR of S_2 and S_3 until their DFR is reached (0.667 Mb/s). The remaining released bandwidth is used to increase the UFR of S_4 to 1.06 Mb/s. Although ten receivers joined S_4 on r_{19}, this happens at the time (second 50) a_{19} is sending an UPDATE. Thus the agent might not have information about some receivers when the message is sent (in this case, a_{19} has only information about the first receiver). Therefore, at second 52, a_7 gets information about only one receiver on r_{19}, and thus, the UFR of S_4 is reduced to 0.06 Mb/s (its UFR on link (r_{14}, r_{15})). After second 53, a_7 already knows about all S_4 receivers on r_{19} and so the UFR of S_4 increases to 0.5 Mb/s.

Next, we analyze the efficiency of SAPRA updating the UFR of sessions in all agents, and notifying receivers when S_1 and S_2 start and end. When receivers join S_1, the ones closer to the source take relatively more time to get their first report than the ones closer to the leaves. This time is proportional to the distance to the source, since it is their first UPDATE that triggers upstream agents. When receivers join and leave S_2, the time that SAPRA takes to notify S_1 receivers also depends on the variation of the minimum UFR of S_1, in addition to the session path length to their leaf router. Tab. 1 shows more clearly these times than Fig 4 a).

Table 1. Notifications of S_1 receivers

	router 5	router 10	router 13	router 16
S2 starts at second 20	27 s	24 s	29 s	25 s
S2 ends at second 170	177 s	179 s	179 s	181 s

Although r_5 receivers are notified about the minimum UFR of S_1 at second 27, Fig. 4 shows that they get a transient UFR of 2.8 Mb/s at second 21. This happens because changes occur at the same time in the branches from the source to r_5 and from r_5 to r_{10}, when receivers joined S_2 on r_5 and r_{10}, which means that the stabilization of UFR of S_1 is achieved first in each branch and only after in the entire path to r_{10}. This transient value is the minimum UFR in the branch upstream r_5 before second 21, but it is not the minimum UFR in the entire path to r_{10}, since the UFR of S_1 on the branch from r_5 to r_{10} was not considered yet upstream of r_5. Fig. 5 shows how the minimum UFR of S_1 is stabilized in the entire path to r_{10} after being computed in each branch.

When a_5 gets a SYNC at second 21 with an UFR of 2.8 Mb/s, it sends a notification to receivers on r_5, since this is the current minimum UFR of S_1 in the path from the source. At that time, S_1 has an UFR of 2 Mb/s on the branch from r_5 to r_{10}, which correspond to the UFR computed by a_5 on (r_5, r_9) after received an UPDATE from a_{10}. At second 22, a SYNC is sent toward a_{10} with an UFR of 2 Mb/s, since this value is lower than the upstream fair rate on a_5

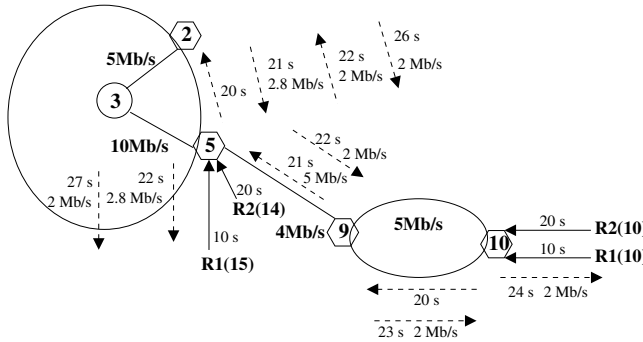


Figure 5. Messages after joining S_2

(2.8 Mb/s) and it is different from the fair rate previously sent (4 Mb/s). At the same time, a_5 also sends an UPDATE with an UFR of 2 Mb/s toward the source and so r_5 receivers get, at second 27, a report with the final minimum UFR for S_1 (2 Mb/s). When a_5 gets this SYNC, it does not sent another SYNC toward a_{10}, because this value is equal to the one sent in the previous SYNC.

When S_2 ends, the UFR of S_1 becomes higher than its upstream fair rate in all links, and so, only the agent closest to the source includes the new UFR of S_1 in a SYNC. So, receivers closer to the source are notified first, as shown on Tab.1, and the notification time increases downstream by $O(n_a x a_s)$, where n_a is the number of agents and a_s is 1 s.

5 Conclusions and Future Work

This paper describes and evaluates SAPRA, a protocol that distributes resources among sessions proportionally to their audience size. SAPRA also notifies receivers about the quality of their sessions, allowing them to adapt to quality variations. Theoretical analysis and simulation results show that SAPRA requires a small amount of storage state, has small bandwidth overhead, and is efficient keeping the rate of sessions below their fair share, updating the fair share of sessions and notifying receivers. Although SAPRA aims mainly to distribute services resources with fairness, it also provides information about the quality of sessions and their population, which can be used to settle SLAs in each domain or define cost allocation policies.

As future work, we will define a receiver-driven adaptation mechanism, where receivers adapt when they receive network reports, a cost allocation scheme based on SAPRA and also study SAPRA in mobile environments, where the location of receivers change frequently.

References

1. Charu C. Aggarwal, Joel L. Wolf, and Philip S. Yu. Design and analysis of permutation-based pyramid broadcasting. *Multimedia Systems*, 7(6):439–448, 1999. 103
2. Dimitri Bertsekas and Robert Gallager. *Data Networks*. Prentice-Hall, Englewood Cliffs, New Jersey, 1987. 102
3. S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss. An architecture for differentiated service. Request for Comments 2475, Internet Engineering Task Force, December 1998. 102
4. M. Handley, C. Perkins, and E. Whelan. Session announcement protocol. Request for Comments 2974, Internet Engineering Task Force, October 2000. 103
5. J. Heinanen, F. Baker, W. Weiss, and J. Wroclawski. Assured forwarding PHB group. Request for Comments 2597, Internet Engineering Task Force, June 1999. 102
6. H. Holbrook and B. Cain. Source-specific multicast for IP. Internet Draft, Internet Engineering Task Force, March 2001. Work in progress. 103
7. H. Holbrook and B. Cain. Using IGMPv3 for source-specific multicast. Internet Draft, Internet Engineering Task Force, March 2001. Work in progress. 103
8. Jun ichi Kimura, Fouad A. Tobagi, Jose-Miguel Pulido, and Peder J. Emstad. Perceived quality and bandwidth characterization of layered MPEG-2 video encoding. In *Proc. of SPIE International Symposium*, Boston, Massachussetes, USA, September 1999. 109
9. Mathias Johanson. Scalable video conferencing using subband transform coding and layered multicast transmission. In *Proc. of International Conference on Signal Processing Applications and Technology (ICSPAT)*, Orlando, Florida, USA, November 1999. 101
10. Arnaud Legout, Joerg Nonnenmacher, and Ernst Biersack. Bandwidth allocation policies for unicast and multicast flows. In *Proc. of the Conference on Computer Communications (IEEE Infocom)*, New York, New York, USA, March 1999. 102
11. Xue Li, Sanjoy Paul, and Mostafa Ammar. Multi-session rate control for layered video multicast. In *Proc. of Multimedia Computing and Networking*, San Jose, California, USA, January 1999. 103
12. Paulo Mendes. "SAPRA: Session-Aware Popurality-based Resource Allocation fairness protocol". http://www.cs.columbia.edu/~mendes/sapra.html. 109, 109
13. Paulo Mendes, Henning Schulzrinne, and Edmundo Monteiro. Session-aware popularity resource allocation for assured differentiated services. In *Proc. of the Second IFIP-TC6 Networking Conference*, Pisa, Italy, May 2002. 102, 104, 104, 105
14. Y. Rekhter and T. Li. A border gateway protocol 4 (BGP-4). Request for Comments 1771, Internet Engineering Task Force, March 1995. 105
15. Dan Rubenstein, Jim Kurose, and Don Towsley. The impact of multicast layering on network fairness. In *Proc. of SIGCOMM Symposium on Communications Architectures and Protocols*, Cambridge, Massachussetes, USA, September 1999. 102
16. Saswati Sarkar and Leandros Tassiulas. Fair allocation of discrete bandwidth layers in multicast networks. In *Proc. of the Conference on Computer Communications (IEEE Infocom)*, Tel Aviv, Israel, March 2000. 102
17. David Taubman and Avideh Zakhor. Multirate 3-D subband coding of video. *Journal of IEEE Transactions on Image Processing*, 3(5):572–588, September 1994. 101
18. Telstra. "AS 1221 BGP statistics". http://bgp.potaroo.net/as1221/bgp-active.html. 108, 109