Distributed Bandwidth Broker for QoS Multicast Traffic*

Elena Pagani and Gian Paolo Rossi
Dip. di Scienze dell'Informazione, Università degli Studi di Milano
via Comelico 39, I-20135 Milano, Italy
email: {pagani,rossi}@dsi.unimi.it

Abstract

The Differentiated Services model is the emerging standard to provide Quality-of-Service (QoS) support for the multimedia applications in the future Internet. This model involves bandwidth broker agents to perform the admission control and network configuration functionalities. A great deal of efforts has been recently devoted to investigate viable approaches to the implementation of mechanisms that automatically perform the bandwidth broker functions. Yet, no standard policy has been so far proposed.

In this paper, we propose a distributed measurement-based protocol that performs admission control functionalities for multicast traffic in diff-serv networks. The protocol supports dynamic changes of the multicast group membership, it operates on-demand and it supports the Premium Service. We prove that the proposed protocol performs an effective and efficient admission control function.

1. Introduction

The *Differentiated Service* (diff-serv, DS) model [1] is the emerging standard to provide *Quality-of-Service* (QoS) support for the multimedia applications in the future Internet. This model guarantees scalability by offering a limited set of service classes. The same forwarding policy is applied to all the packets belonging to the same class, independently of their source and destinations. In the diff-serv model, the admission control and network configuration functionalities are in charge of the *bandwidth broker* (BB) agents [10]. Currently, no standard approach exists to realize the BB functionalities. Moreover, although many multimedia applications are characterized by a multicast communication pattern, the support of multicast communications in the diff-serv framework is difficult and rarely considered.

*This work was supported by the MURST under Contract no.MM09265173 "Techniques for end-to-end Quality-of-Service control in multi-domain IP networks".

In the literature, one of the approaches proposed to perform admission control consists of actively measuring the network resource availability to evaluate whether a new session can be supported or not [3]. So far, this approach has been studied for the unicast sessions. In this paper, we discuss the end-to-end *Call Admission Multicast Protocol* (CAMP). CAMP adopts an active-measurement approach to perform admission control for multicast sessions in IP networks. It can be used to provide the multimedia applications requiring bandwidth guarantees with the *Premium Service* [10]. CAMP operates on-demand; it is scalable and supports the dynamic changes of the multicast group membership, and it represents a possible implementation of a distributed BB. We prove that CAMP is able to avoid recipient misacceptances, thus guaranteeing high network utilization and fulfilment of the service level agreement signed with the users, under the hypothesis that the measurement phase provides the accurate evaluation of the available bandwidth. We have obtained a lower bound on the length of the measurement phase, that guarantees an accurate estimate of the resource availability; the result applies to the unicast case as well.

The paper is organized as follows: in Section 2, we introduce the system model considered throughout this work; we define the *admission control problem* and we discuss the related works. In Section 3, we describe the CAMP protocol. In Section 4.1, we discuss some issues related with the measurement-based approach, and we obtain a lower bound on the measurement phase length. In Section 4.2, we prove that the proposed protocol solves the admission control problem. Section 5 concludes the paper.

2. System model

A QoS-sensitive application specifies to the underlying service provider, the QoS communication requirements and the behaviors of the data flow it is going to generate (*traffic profile*). We assume that a session announcement protocol (e.g., `sdr` [6]) is used to announce session information, such as the transmission start time and the multicast address of the destination group. In this paper, we consider sources generating CBR traffic. All the recipients receive the same set of microflows; they are *homogeneous*, that is, they have the same QoS requirements.

QoS is provided according to the *differentiated services* (diff-serv, DS) model [1]. Diff-serv's offer QoS on a per-packet basis. According to this model, each DS domain offers a specified set of service classes. Each packet, through the use of the TOS/*DS* byte field of the IP header [9], is labeled with the identifier of the class (*DS codepoint*) to which it belongs. Resource reservation is performed on the base of the different service classes. All the packets which belong to a given service class C are subject to the same forwarding policy (PHB); they compose a *behaviour aggregate*. In this work, we adopt the same terminology proposed in [1]. Nichols et al. in [10] propose an architecture to provide QoS according with the diff-serv model, that exploits the so called *bandwidth broker* (BB) agents to perform the functionalities of call admission control, resource reservation and system configuration. A BB is assumed to operate in each domain. In this work, we propose an admission control mechanism that can be adopted to realize a distributed BB entity. The functional architecture that we consider in this work is in accordance with the proposal presented in [14]. The BB provides the applications with the interface to access the QoS services. It may exploit the services of both a membership protocol, and the multicast routing protocol to acquire knowledge about the network topology. When the QoS aggregate spans over multiple domains, an inter-domain protocol is executed amongst peer BBs, to guarantee the proper configuration of the transit and destination domains.

2.1. Problem definition and assumptions

A mechanism aiming at performing admission control must guarantee that recipient misacceptances do not occur, that is, a recipient accepts to receive a session if and only if enough network resources are available to deliver the session traffic with the adequate QoS level. In this work, we consider as the network resources the bandwidth. We define the available bandwidth b_a over a path as the bandwidth currently unused by the QoS traffic forwarded along that path. We formally define the admission control problem for multicast traffic as follows:

Definition 2.1 (Multicast Admission Control) *Given a group $\mathcal{G}$ of processes that want to receive the QoS traffic generated by a source s with rate r_s, an admission control procedure must satisfy the following properties:*

Effectiveness: *the service is accepted by a recipient $d \in \mathcal{G}$ only if $b_a \geq r_s$ on the path from s to d.*

Efficiency: *if $b_a \geq r_s$ on the path from s to a recipient $d \in \mathcal{G}$, and s is the only QoS traffic source executing the admission control procedure, then d accepts the service.*

Termination: *the admission control procedure terminates within a finite time and s either switches to the transmission of data traffic or turns off. The latter case occurs only when all the recipients refuse the session.*

Safety: *if the group membership changes when the QoS data transmission is ongoing, the admission control procedure does not incur in the Neglected Reservation Subtree Problem [2].*

The first two properties guarantee that the admission control procedure does not make mistakes. The Efficiency property guarantees high network utilization. The Effectiveness property guarantees that a new service acceptance does not affect other established sessions. The Safety property guarantees that the admission control procedure avoids the forwarding of QoS traffic along paths not configured to support it, in the events of recipients joining during the data transmission.

To solve the admission control problem, CAMP requires that the system satisfies the following assumption, whose role is made clear in the next sections:

Accuracy assumption: *A value k exists such that, if a node d receives k packets from a node s so that the measured reception rate is r, then along the path from s to d there is enough bandwidth to accommodate the traffic generated by s with rate less than or equal to r.*

2.2. Related works

So far, very few proposals have been presented about the BB implementation. The QBone testbed [14], under development by the Internet2 QoS Working Group, involves a BB that is initially "implemented" by a human operator who manually configures the system. Scenarios have been figured out to design an inter-BB communication protocol (SIBBS) [14]. No standard technique still exists to automatically perform intra-domain configuration.

In this paper, we discuss a mechanism to implement the resource reservation, call-admission control and system configuration functionalities both at intra-domain and inter-domain level, which adopt a distributed approach. We propose a measurement-based technique that tests for resource availability without requiring state information maintenance by the routers. Several solutions have been recently proposed to perform admission control, based on either *passive* or *active* measurements [12]. In the former case [7, 4], routers monitor the passing traffic. When they receive a service set-up request, they decide about the admission of that service basing on the gathered estimates about

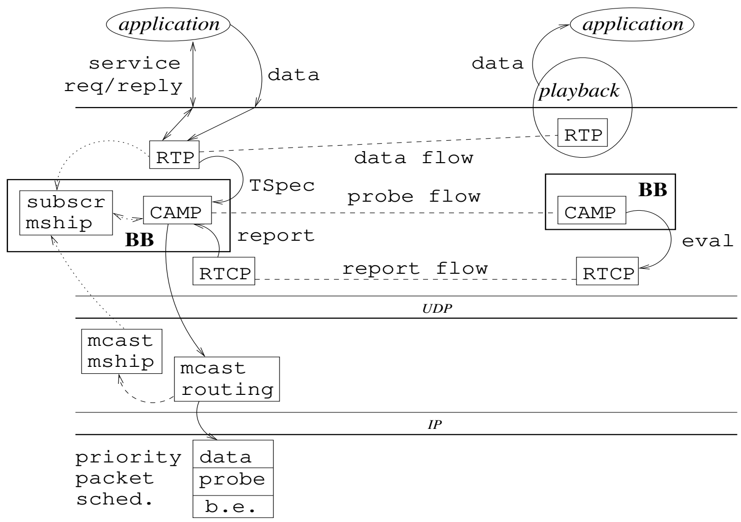


Figure 1. Architecture of the CAMP-based end stations.

the current resource usage. This technique is less precise than the active measurement approach in the estimation of the available resources, and it requires that each router is able to perform admission control. Yet, it is less intrusive. Through the active measurement approach, *probing* traffic must be generated with the same profile of the QoS data traffic. Admission is decided by evaluating the quality with which the probing traffic is received at the destination. Solutions that adopt the active measurement technique are, for instance, those presented in [8, 5]. In [8], that technique is used to estimate the capacity of the bottleneck link along a path. In [5], the authors evaluate the suitability of that approach to implement admission control for the *Controlled-Load* service [16] in *Integrated Service* systems. All those solutions do not involve mechanisms to deal with multicast communications.

3. The CAMP algorithm

In Figure 1, we show the system architecture in which CAMP works. CAMP operates within the RTP/RTCP [13, 11] protocol suite and performs the set-up of a RTP session. It receives from the application, via RTP, the profile of the data traffic that will be generated (specified for instance as a `TSpec` [16]). CAMP uses RTCP to monitor the QoS supplied to the recipients. According to the *active measurement* approach [12, 3], CAMP generates a *probing* traffic with the same profile of the data traffic generated by the application. Both the data and the `probe` packets are multicast. To this aim, we assume that both a membership protocol and a multicast routing protocol are available. This latter maintains a tree-based routing infrastructure connecting the multicast recipients. CAMP is independent of both those protocols. The *probing* phase has the aim of evaluating whether or not the available bandwidth is sufficient to support the new traffic. With respect to the classification given in [3], we adopt *out-of-band* probing with *dropping* of the `probe` packets as the congestion signal. All the routers use a *priority* packet scheduling discipline: the `probe` packets are marked with a higher priority than the best effort packets, but a lower priority than the QoS packets. This priority assignement ensures that the probing traffic does not affect the established QoS sessions. On the other hand, `probe` packets can drain the available bandwidth for the new QoS session at the expenses of the best effort traffic.

To support multicast, two issues must be considered: (i) the receiver group membership can dynamically change; and (ii) different destinations can experiment different QoS in receiving the same traffic. We cope with problem (ii) above as follows: the source CAMP diffuses the generated traffic profile via its own RTCP reports. Each receiver r, after having received k `probes`[1], matches the source traffic profile against the received probing traffic profile and decides whether to accept the *whole* session or not. In both cases, r informs the source about its decision with its own RTCP reports. If r refuses the service, it unsubscribes from the group and prunes from the tree *before* sending its refusing report. When all the reports have been received by the source, if the service is accepted by at least one recipient, the source switches from the transmission of `probe` packets to the transmission of the data packets generated by the application, *without discontinuity*. The data packets are forwarded along the pruned tree. If all the reports are refusals, then the source turns off.

[1]The k parameter is the same as in the Accuracy Assumption.

With respect to the problem (i) above: the data traffic must be forwarded along a subtree only after that all the recipients grafted to that subtree have completed the probing phase by either accepting the service or pruning from the tree. In the opposite case, in the event that there is not enough available bandwidth, other established sessions traversing the same links (whose packets belong to the same aggregate) would be negatively affected because their packets could be dropped. On the other hand, the admission control must be performed only on the new branch, to avoid the duplication of the resource reservation on the path from the source to the graft point.

We deal with those problems by means of a *proxy* mechanism. A CAMP proxy is instantiated in a router, when the router is an in-tree node and either: (1) it receives a `probe` packet for the group (source initialization phase); or (2) it notices the appearance of a new downstream interface for the group and it receives a data packet for that group (dynamic membership change). The proxy lifetime lasts until the set-up phase terminates for all the downstream destinations. The proxy uses the information in the local routing table to maintain a data structure (figure 2(a)) containing the list of downstream multicast interfaces (*oif*s)[2], the list of destinations for each *oif* and the *oif* state. Initially, all the *oif*s in the table are in the *probing* state. During its activity, the proxy remarks as `probe` packets all the incoming packets for the session, that must be forwarded to the probing *oif*s. The proxy labels the forwarded packets with its own address. The receivers send their RTCP reports to their proxies.

When a proxy receives an accepting report, and it is receiving `probe` packets from its parent, it knows that the source is in the initialization phase and it forwards that report to the source. If data packets are received from the parent, the reports are dropped. When a report has been received from each destination downstream a *probing oif*, and all the reports are refusals, the *oif* is removed from the table. If at least an accepting report exists, the proxy changes the state of the corresponding *oif* to *data* and stops the packet remarking for that *oif*. If new downstream interfaces appear while the proxy is active, they are added to the proxy table. When no *oif* in the table is anymore marked as *probing*, the proxy turns off. The source CAMP entity switches to the transmission of the data as soon as it receives an accepting report. Figure 2(b) summarizes the proxy operations.

The proxy mechanism achieves the following goals:

1. the "`probe`" packets allow to evaluate the resource availability only along the new branches;

2. the new destinations immediately start receiving the real data, although possibly with a lower quality than required;

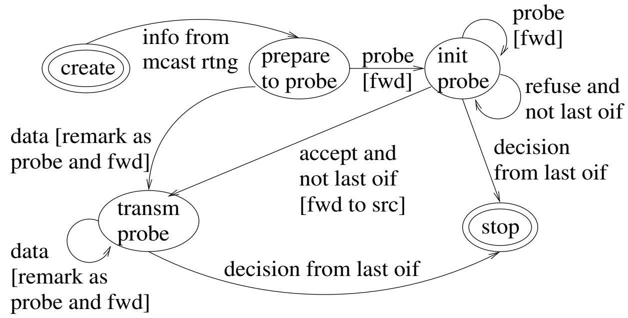
3. the data sent to the new destinations do not affect previously established sessions traversing the new branch, as the data packets are marked and treated as `probe` packets;

4. the membership is hidden to the source. Anyway, it must be known by the proxy.

[2]All the downstream interfaces in case (1); only the newly created downstream interface in case (2).

probing interfaces	*state*	*destinations*
oif 1	`data`	{rcvs}_1
...	...	...
...	...	...
...	...	...
oif n	`probing`	{rcvs}_n

(a)



(b)

Figure 2. (a) Data structure maintained by a proxy. (b) State transition diagram for the CAMP proxy.

The termination of the probing phase may be postponed if the network is congested and `probes` are dropped. In the worst case, a recipient might never collect enough `probes` to decide. A recipient that does not receive k `probes` within a time T_p from either the start time of the session or its graft time (whatever is the greatest), prunes from the tree and sends a refusing report. T_p must be long enough to allow the reception of at least k `probes`, that is, it must last at least k times the packet inter-arrival time. To guarantee the termination of the probing phase, the report loss must also be prevented. This can be achieved by adopting TCP as the transport protocol used by RTCP.

3.1. Cascading proxies

The concurrency amongst receivers in joining the group may generate *cascading* proxies. In the example shown in

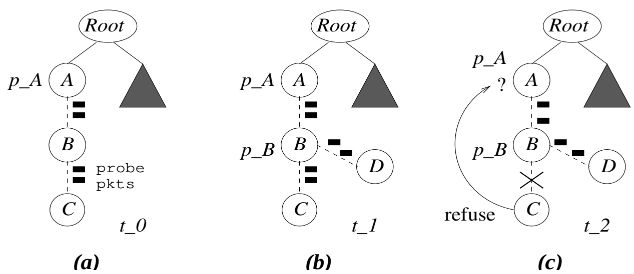


Figure 3. Example of cascading proxies.

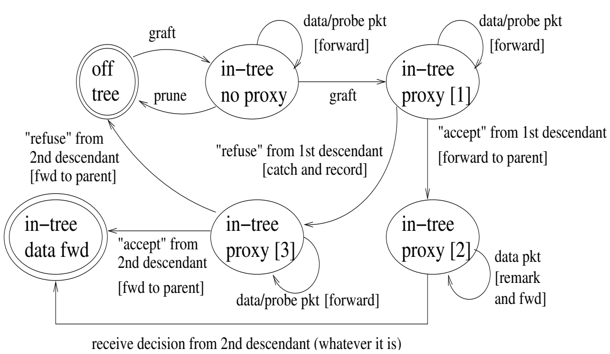


Figure 4. State transition diagram for the child proxy.

Figure 3, at time t_0 the data transmission is ongoing when the node C joins the group (Figure 3(a)) and it grafts the tree via the branch $A - B - C$. The node B becomes an in-tree node, and a proxy p_A is instantiated in the node A. Let us suppose that the node D joins the group at the time t_1 (Figure 3(b)), while the set-up phase for C is ongoing, and that D grafts the tree via the branch $B - D$. Since B is now an in-tree node, it instantiates a proxy, p_B. p_B receives data packets remarked as `probes` from p_A, and forwards them to both the downstream interfaces towards C and D. Let us suppose that at time t_2, C terminates its set-up phase (before D). If C refuses the service, it prunes from the tree (Figure 3(c)). Yet, p_A should not turn off after removing the unique downstream interface from its table (Figure 2); indeed, it must continue to forward the packets along the tree because D could still accept the service. To this purpose, B catches the C refusing report and waits for the D decision. On the other hand, if C accepts the service and B does not forward the C report to A, B damages C, which would receive the data with the QoS achievable by the probing traffic until D decides. Hence, B must forward the C accepting report to A, and remark the data as `probes` before forwarding them to D. The state transition diagram shown in Figure 4 summarizes the B behaviour. To support the cascading proxy mechanism, the reports are forwarded

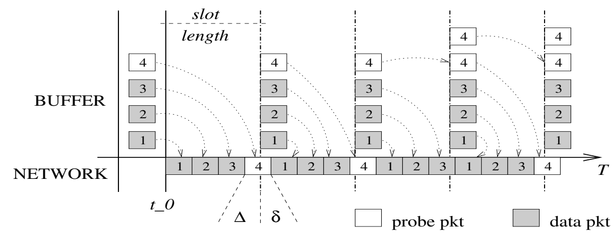


Figure 5. Example of bandwidth sharing among homogeneous sessions.

upstream the tree. The depicted interaction scheme between parent and child proxies is replicated for multiple tree levels if there are more than two cascading proxies.

4. Protocol analysis and correctness

4.1. Analysis of the measurement phase length

As we will see in section 4.2, the protocol correctness is founded on the fulfilment of the Accuracy assumption; hence, the characterization of the k value is fundamental to guarantee the proper algorithm behaviour. The k value determines the length of the probing phase. As noticed in [3], a long probing increases the probability of *starvation* and thus of low network utilization, because of the greater probability of overlapping amongst several probing traffics along the same path. Long probing phases negatively affect the best effort traffic as well. On the other hand, the probing phase cannot be too short, otherwise the bandwidth availability cannot be estimated with sufficient accuracy, thus possibly leading to misacceptances. Let us consider the example shown in Figure 5: at time t_0 three sessions have already been accepted, while a fourth session starts its set-up phase. All the sessions have the same rate and traverse the same link; the *slot length* equals the inter-packet delay for one session. Let us suppose that the receivers consider only two `probe` packets to estimate the available bandwidth. As shown in the figure, the first two `probes` are transmitted; if their inter-packet delay is such that the estimated rate is acceptable (with a given tolerance), the fourth session switches to the data transmission phase although there is not enough bandwidth for it. That example shows that a *lower bound* exists on the probing phase length[3]. A trivial lower bound is 2: this is the minimum because a recipient can estimate the incoming bandwidth. We have characterized the lower bound in the basic case. Let us assume that:

- the sources generate traffic which is *perfectly* CBR;

[3]These arguments apply to the unicast case as well.

- all the sessions have the same rate r;
- all the sessions use the same frame size f and the same packet size p (which is the frame size plus possible headers);
- the links are uniform, i.e., they all have the same capacity $\mathcal{C}$.[4]

The inter-frame delay i is given by: $i = f/r$; this is the period with which a source supplies new frames to be transmitted[5]. The available bandwidth in i is $i \cdot \mathcal{C}$; throughout i we can transmit $i \cdot \mathcal{C}/p$ packets. The maximum number of acceptable sessions is: $\lfloor i \cdot \mathcal{C}/p \rfloor$. The residual bandwidth is:

$$\Delta = (i \cdot \mathcal{C}) - \lfloor \frac{i \cdot \mathcal{C}}{p} \rfloor \cdot p < p$$

If one more session is transmitted, as session 4 in the example, the used bandwidth exceeds the bandwidth available in i of:

$$\delta = (\lfloor \frac{i \cdot \mathcal{C}}{p} \rfloor + 1) \cdot p - (i \cdot \mathcal{C}) = p - \Delta > 0$$

After $\lceil p/\delta \rceil$ slots, one `probe` packet cannot be transmitted because the channel is preempted by data packets (as it happens in the third slot in the example of Figure 5). Hence, the probing phase must last at least $\lceil p/\delta \rceil$ slots, and a recipient must wait for at least $\lceil p/\delta \rceil$ `probe` packets before deciding. After that interval, the session can be accepted only if the recipient tolerates a packet loss probability greater or equal to $(\lceil p/\delta \rceil)^{-1}$. It is worth to notice that the above lower bound can be computed using information (source traffic profile, bottleneck capacity) that the recipients can know. The evaluation does not take into account the possible interactions of a probing phase with the best effort traffic or other concurrent probing traffic. With those traffics the dropping of the `probes` would occur earlier, thus making congestion more evident.

Let us consider the case of VoIP traffic: according to G.711 [15], CBR traffic is generated with rate r of 64 Kbps, composed of frames of $f = 8$ bits with frequency $i = 125$ μsec. If $\mathcal{C} = 10$ Mbps, then the available bandwidth in a slot is $i \cdot \mathcal{C} = 1250$ bits. Assuming that each frame is sent within an IP packet, 40 bytes are needed for the TCP and IP headers; hence, $p = 328$ bits. The maximum number of acceptable sessions is 3, $\Delta = 266$ bits and $\delta = 62$ bits. Hence, the probing phase must last for at least 6 slots, that is, 750 μsec.

In the case of heterogeneous sessions, the probing phase must last enough to guarantee that a session with a high rate overlaps with the sessions with the lowest rate. Consider the

[4]This assumption can be easily relaxed by taking as $\mathcal{C}$ the capacity of the bottleneck link.

[5]In the example, $f = p$ and i is the slot length.

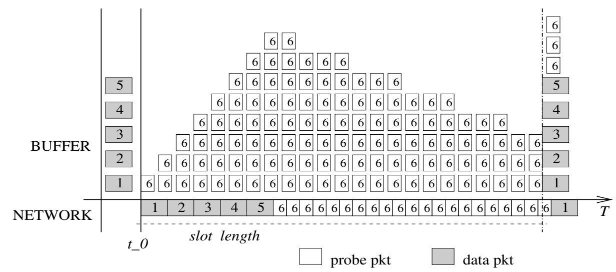


Figure 6. Example of bandwidth sharing among heterogeneous sessions.

example shown in Figure 6: over a channel with capacity 2 Mbps, 5 CBR sessions have been established having rate 128 Kbps and packet size 1 Kb each. Hence, the slot length is 7.8 msec., the bandwidth available in the slot is 15.6 Kb and the bandwidth occupancy in the slot is 5 Kb; hence, 10.6 Kb per slot are unused. At t_0, the probing is started for a session $\mathcal{S}$ with rate 1.5 Mbps and packet size 500 bits. Let us suppose that the bandwidth availability is estimated over 10 samples. The $\mathcal{S}$ source generates about 23 packets in a slot, and about 21 of them can be sent by exploiting the bandwidth available in a slot. The first 10 packets of $\mathcal{S}$ are sent at a rate higher than the source rate, as shown in the figure. The recipients measure that high rate, because they start the evaluation upon the reception of the first `probe`, and cannot be aware of the existing backlog. As the consequence, the recipients can accept the service, although the available bandwidth is not enough for $\mathcal{S}$. Similar problems must be dealt with when VBR sources are considered.

4.2. Correctness proof

In this section, we prove that CAMP solves the *Multicast Admission Control* problem as it is defined in section 2.1.

Lemma 4.1 CAMP *satisfies the Effectiveness property in the case of static group membership, with all the recipients joining before the source initialization phase starts.*

Proof. When the source initialization phase starts, `probes` are forwarded along the tree, testing the paths from s to every destination. According to the protocol, a recipient d decides whether to accept a session or not after having received k `probes`. In particular, if the received rate r is greater or equal to the source rate r_s, then d accepts the service. By the Accuracy assumption, along the path from s to d there must be enough bandwidth to accommodate a traffic with rate $r \geq r_s$, that is, to accommodate the traffic generated by s. ♣

Lemma 4.2 CAMP *satisfies the Effectiveness property in the case of dynamic group membership.*

Proof. In the case of recipients that join the group (and hence the tree) before the source initialization phase, the probing is performed along the whole tree branch from the source to each recipient; hence, the claim is satisfied by Lemma 4.1.
In the case of a recipient d that joins during the data transmission: by the algorithm, a proxy is instantiated in the graft node g, that forwards data packets remarked as "`probes`" to d. d decides only after having received k "`probes`". By applying the Accuracy assumption and the same arguments as in Lemma 4.1, the claim is proved for the subpath from g to d. The claim is proved for the subpath from s to g by recursively applying the Lemma 4.1 or this Lemma: since the branch $s \to g$ exists, the session has been accepted by the first destination joining the tree via that branch. As the consequence, the claim holds for the whole path from s to d.
In the case of a recipient d that joins during the source initialization phase, either:

1. d grafts to a node g that has already received all the reports from its other downstream destinations. As the consequence, g instantiates a proxy for the *oif* to d;

2. d grafts to a node g that has not yet received the report from any downstream destination. By the algorithm, a proxy is already active in g, and it adds to its table the downstream *oif* to d with state *probing*.

In both cases above, the same arguments apply as in the case of a recipient joining during the data transmission. ♣

Lemma 4.3 CAMP *satisfies the Efficiency property.*

Proof. Let us suppose that along the path from s to d the available bandwidth is b_{sd} such that $b_{sd} > r_s$ (with r_s the s rate), and that d refuses the s traffic. By the algorithm, d refuses if it receives $\geq k$ `probes` at a rate $r < r_s$. The `probes` compete for the available bandwidth only with best effort packets, by the hypothesis that s is the unique source currently executing the admission control procedure. By the priority assignment, `probes` are allowed to use all the available bandwidth, that is $\geq r_s$ by hypothesis. By the Accuracy assumption, r is an accurate measure of the available bandwidth; hence $r \cong b_{sd}$. Hence, the `probes` are received with rate $r \geq r_s$ and we obtain an absurd. ♣

The previous Lemma applies both to the tree branches tested for resource availability in the source initialization phase, and to the tree branches dynamically built for new recipients. Let us notice that, if several sources concurrently perform the set-up phase, and the sum of their rates is lower than the available bandwidth, the above result still applies to all the sessions.

Lemma 4.4 *If a recipient d joins the group $\mathcal{G}$ during the transmission of the data generated by the source s for $\mathcal{G}$, then the d's join does not originate the NRS problem.*

Proof. By the algorithm, when d joins, a proxy is instantiated in the graft node g. The proxy forwards to d the data packets remarked as "`probes`"; those packets have the `probe` forwarding priority and do not affect existing QoS traffic traversing the branch. The proxy turns off when it receives either an accepting or a refusing report from d. In the former case, the data traffic is forwarded along the branch from g to d; but, enough bandwidth is available to support it, by Lemma 4.2. In the latter case: by the algorithm, d prunes from the tree *before* sending the refusing report. Hence, when the proxy receives the report, the downstream *oif* to d has already been removed from the g multicast routing table. As the consequence, the QoS traffic is not forwarded to d when the proxy turns off. ♣

Lemma 4.5 CAMP *satisfies the Safety property.*

Proof. In the case of a single recipient joining the group during the data transmission, the claim is proved by Lemma 4.4. In the case of multiple recipients concurrently joining: let A be the node where the proxy p_A at the top of the proxy hierarchy is located. By the algorithm, p_A remarks the data packets as "`probes`" before forwarding them downstream, until it receives an accepting report. When the data traffic starts flowing through the p_A subtree, it may traverse either nodes where a proxy is running or nodes with no proxy. In the former case: a proxy forwards the QoS traffic only to the *oif*s in *data* state, leading to accepting recipients, and thus over paths able to support the requested QoS. The packets are remarked as "`probes`" before being forwarded to the *probing oif*s. No traffic is forwarded to the refusing *oif*s as they are pruned.

The in-tree nodes with no proxy, according to the algorithm, either (i) had a proxy that turned off, or (ii) entered the tree after the source initialization phase and had a proxy upstream. In the former case, when the proxy turned off, in the multicast routing table of the corresponding node only the output interfaces remained that lead to accepting destinations through paths able to support the QoS traffic. In the latter case, the same arguments can be applied to the ancestor node where the upstream proxy was located: since the QoS traffic is sent to the node, it is on a path not pruned by the tree before the proxy in the ancestor turned off, and thus able to support the QoS traffic. In both cases, the QoS traffic can be safely forwarded to the downstream output interfaces. ♣

Lemma 4.6 CAMP *satisfies the Termination property.*

Proof. A recipient d terminates the set-up phase when either d has received k `probes` or a time T_p has elapsed from the start of the set-up phase for d, whichever occurs first. In both cases, d sends a report upstream. By the algorithm, during the source initialization phase the reports are forwarded to the source, and they are reliably transmitted via TCP. Hence, within a finite time either:

- a recipient receives k `probes` and decides to accept the session. Its accepting report is reliably delivered to s, which switches to the transmission of the data traffic; or

- each recipient refuses the service, either because it has received k `probes` and it has measured an unacceptably low bandwidth availability, or because T_p has elapsed from the set-up start. As the consequence, every recipient reliably sends a refusal report to s. Since the number of recipients is finite, within a finite time s receives all the refusing reports and turns off.

♣

Theorem 4.1 CAMP *solves the admission control problem for multicast communications in the case of dynamic group membership.*

Proof. The claim follows from the Lemmata 4.1 through 4.6. ♣

5. Concluding remarks

In this paper we propose the CAMP protocol to provide bandwidth guarantees to multicast applications with dynamic changes of the destination group membership. We prove that the protocol terminates, and that it avoids receiver misacceptances. We implemented the CAMP protocol in the framework of the NS-2 simulation package [11]. The achieved results show that the devised mechanism effectively performs admission control.

So far, proxies can support dynamic multicast membership under the assumption that all the recipients grafted to the tree receive the same transmissions. We are extending their capabilities so that heterogeneous destinations, willing to accept different sets of data flows, can be supported. The characterization of a lower bound on the probing phase length must be extended to consider the case of heterogeneous sessions and heterogeneous packet sizes.

References

[1] Blake S., Black D., Carlson M., Davies E., Wang Z., Weiss W. "An Architecture for Differentiated Services". *RFC 2475*, Dec. 1998. Work in progress.

[2] Bless R., Wehrle K. "IP Multicast in Differentiated Services Networks". *Internet Draft, draft-bless-diffserv-multicast-01.txt*, Nov. 2000. Work in progress.

[3] Breslau L., Knightly E., Shenker S., Stoica I., Zhang H. "Endpoint Admission Control: Architectural Issues and Performance". *Proc. SIGCOMM'00*, pages 57–69, 2000.

[4] Cetinkaya C., Knightly E. "Egress Admission Control". *Proc. INFOCOM'00*, 2000.

[5] Elek V., Karlsson G., Rönngren R. "Admission Control Based on End-to-End Measurements". *Proc. INFOCOM'00*, 2000.

[6] Handley M. "The sdr Session Directory: An Mbone Conference Scheduling and Booking System". `http://mice.ed.ac.uk/mice/archive/-sdr.html`, Apr. 1996.

[7] Jamin S., Shenker S., Danzig P. "Comparison of Measurement-based Admission Control Algorithms for Controlled-Load Service". *Proc. INFOCOM'97*, 1997.

[8] Lai K., Baker M. "Measuring Link Bandwidths Using a Deterministic Model of Packet Delay". *Proc. SIGCOMM'00*, pages 283–294, 2000.

[9] Nichols K., Blake S., Baker F., Black D. "Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers". *Internet Draft, draft-ietf-diffserv-header-04.txt*, Oct. 1998. Work in progress.

[10] Nichols K., Jacobson V., Zhang L. "A Two-bit Differentiated Services Architecture for the Internet". *Internet Draft, draft-nichols-diff-svc-arch-00.txt*, Nov. 1997. Work in progress.

[11] Pagani E., Rossi G.P. "Measurement-Based Admission Control for Dynamic Multicast Groups in Diff-Serv Networks". *Proc. IFIP Networking 2002 Conference, LNCS*, May 2002. Springer, Berlin.

[12] Räisänen V. "Measurement-Based IP Transport Resource Manager Demonstrator". *Proc. International Conference on Networking, Lecture Notes in Computer Science 2094*, pages 127–136, Jul. 2001. Springer, Berlin.

[13] Schulzrinne H., Casner S., Frederick R., Jacobson V. "RTP: A Transport Protocol for Real-Time Applications". *RFC 1889*, Jan. 1996. Work in progress.

[14] Teitelbaum B., Chimento P. "QBone Bandwidth Broker Architecture". *Internet 2 QoS Working Group Draft*, Jun. 2000. Work in progress. `http://qbone.internet2.edu/bb/`.

[15] Tobagi F. et al. "Service Differentiation in the Internet to Support Multimedia Traffic". *Proc. IWDC 2001, Lecture Notes in Computer Science 2170*, pages 381–400, Sep. 2001. Springer, Berlin.

[16] Wroclawski J. "Specification of the Controlled-Load Network Element Service". *RFC 2211*, Sep. 1997. Work in progress.