Analysis and Evaluation of QoS-Sensitive Multicast Routing Policies*

Elena Pagani and Gian Paolo Rossi

Dip. di Scienze dell'Informazione, Università degli Studi di Milano
via Comelico 39, I-20135 Milano, Italy
{pagani, rossi}@dsi.unimi.it

Abstract The provision of *Quality-of-Service* to multimedia applications in Internet is currently a hot research topic. Many approaches, architectures, models and protocols have been proposed in the literature. The proposals impact on the whole system architecture. So far, only a few of them have been implemented and tested. The standardization process is far to be completed. In this paper, we discuss and compare three multicast routing algorithms that have been proposed in the literature, aiming at characterizing tree structures according to the QoS constraints imposed by either the traffic sources or the recipients. Two of those algorithms have been evaluated by means of simulations.

Keywords: Quality-of-Service, admission control, multicast routing, performance evaluation.

1 Introduction

The deployment of protocols and architectures supporting *Quality-of-Service* (QoS) in IP networks is strongly demanded by the growing diffusion of multimedia and real-time applications for the Internet. Those protocols must provide multicast support to fulfil the requirements of a large part of the applications, characterized by a 1-to-many or many-to-many interaction schema. Many proposals have been presented in the literature, aiming at either defining QoS-sensitive functional architectures [1,2,3], or providing possible implementations for the architectural modules [4,5]. So far, however, few proposals have been realized and tested.

In this work, we analyze three approaches proposed in the literature to perform the admission control for QoS multicast traffic. Those approaches allow to characterize tree structures for the forwarding of QoS traffic, in either int-serv domains or access networks to diff-serv backbones. We compare the performance of these approaches by means of simulation techniques. The paper is structured as follows: in Section 2, we introduce the system model considered throughout the work. In Section 3, we describe the different approaches proposed in the

* 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".

literature to the QoS-sensitive multicast routing problem, focusing in particular on the QoSMIC [6], QoSCBT [7] and QoSPIM [8] algorithms. In Section 4, we analyze the pros and cons of those three protocols, while in Section 5, we discuss the simulation results obtained by implementing QoSMIC and QoSCBT in the framework of the NS-2 simulation package [9]. Some concluding remarks are presented in Section 6.

2 The System Model

A network $\mathcal{N}$ can be represented as a weighted graph $G = \langle V, E\rangle$ where the vertices in V correspond to the network routers and the edges in E correspond to the links connecting the routers. Let $N = |V|$ be the number of routers in $\mathcal{N}$. The weight of an edge e is a vector of n elements. Each element represents the quality of the link in terms of a different metric. Examples of metrics are the bandwidth, the transmission delay and the packet loss probability. The metrics may be either *static* or *dynamic*. The static metrics (e.g., link capacity, reliability) only depend on the network technology. By contrast, the dynamic metrics (e.g., bandwidth availability, current queueing delay) depend on the current network status. The static metrics do not reflect the current traffic situation; considering them for the QoS routing may lead to inaccurate decisions, thus failing in providing the requested service level. On the other hand, the maintenance of updated estimates of the dynamic metrics can be costly.

The weight of a path p between two nodes $u, v \in V$ is computed from the weights of the links forming p. Two weight elements compose differently, depending on whether the corresponding metric is *additive* (such as the delay), *multiplicative* (such as the loss probability) or *concave* (such as the bandwidth) [7]. In this work, we indicate with $m(u, v)$ the value of the metric m along the path from u to v provided by the unicast routing protocol.

An application s generating QoS traffic uses a session announcement protocol (e.g., `sdr` [10]) to announce the needed session information such as the transmission start time and the multicast address of the destination group. According to the int-serv model [2], a host h that wants to receive the s' flow joins the corresponding destination group $\mathcal{G}_s$: h requests to graft the distribution tree $\mathcal{T}_s$ for $\mathcal{G}_s$ and it specifies the QoS it wants to receive, represented as a vector req_h of n elements. In this paper, we only focus on the int-serv model, because in the diff-serv model [1] many problems arise concerning the support to dynamic changes of the multicast group membership [11], that deserve further study.

3 Proposed QoS Multicast Routing Policies

Several algorithms have been recently proposed in the literature, to compute tree structures to forward QoS multicast traffic. They differentiate in the number and nature of the considered metrics, and in the methods used to characterize the tree. The (sub)optimal tree may be searched using *heuristics*, according to the result of NP-completeness for the problem of computing an optimal path considering more than one additive metric constraint [8].

Some proposals [12,13,14] present algorithms that exploit the knowledge of the network topology and resource availability, the sources, destinations and resource requirements of the QoS traffic, to search for a tree satisfying the QoS constraints. Part of the needed information may be obtained for instance from an underlying link-state protocol. However, the specifications of those algorithms usually disregard the implementation issues. Neither the system architecture nor the communication pattern among the involved parties are detailed.

In this paper, we focus on algorithms which have a distributed control, possibily derived by adding QoS-awareness to standard multicast routing protocols. In particular, we have studied the behaviours of the QoSMIC [6], QoSCBT [7] and QoSPIM [8] multicast routing protocols. All these protocols aim at characterizing a tree structure $\mathcal{T}$ that connects the hosts in a group $\mathcal{G}$, and is *shared* among all the sources sending traffic to $\mathcal{G}$. Let $\mathcal{R}$ be the root of $\mathcal{T}$. $\mathcal{T}$ is such that $\forall g \in \mathcal{G}$ a path p exists in $\mathcal{T}$ from $\mathcal{R}$ to g so that, for each metric m_i, $1 \leq i \leq n$, p satisfies $m_i(\mathcal{R}, g) \leq req_g[m_i]$ if m_i is either an additive or a multiplicative metric, or $m_i(\mathcal{R}, g) \geq req_g[m_i]$ if m_i is a concave metric. The algorithms can support *heterogeneous* recipients, that is, recipients having different QoS requirements. We describe them in detail in the next sections.

3.1 QoSMIC

According to the QoSMIC protocol [6], a new router nr joining $\mathcal{G}$ sends its request to $\mathcal{R}$, which forwards it along $\mathcal{T}$ to identify a subset of the nodes in $\mathcal{T}$ as the nr *candidates* to graft to the tree. Several heuristics can be used to characterize the set of candidates; we adopted the *multicast tree search* policy with the *directivity* mechanism for the selection of the candidates. A node $n_c \in \mathcal{T}$ is a candidate if the quality of the unicast path connecting n_c to nr, according to the static metrics, is such that the QoS requested by nr can be provided and n_c realizes a local optimum in terms of *quality-of-path*, that is, it offers a path whose quality is equal to or better than that offered by its ancestors in the tree. Each node in $\mathcal{T}$ carries its estimated QoS towards nr in the request forwarded along the tree, if it is better than that currently reported, to inform its descendants about it. The candidates send a `bid` message to nr; the `bid` is used to evaluate the quality-of-path according to the dynamic metrics, and establishes a tentative reservation in the traversed nodes. nr waits for an interval Δt to receive the `bid`s from all the candidates. Basing on the quality estimates carried by the received `bid`s, nr chooses the router (and the branch) that offers the best QoS. nr sends to the chosen candidate a `graft` message, that establishes the resource reservation in the traversed nodes. The tentative reservations for the other candidates are removed upon a timer expiration. `graft` messages are periodically re-sent to refresh the reservations. QoSMIC can consider only one *quality-of-path* metric at a time (i.e., $n = 1$); it requires an underlying QoS-sensitive unicast routing protocol that finds "good" paths according to that metric. As a router only uses the static metrics to decide whether it is a candidate, no status information exchange amongst nodes is needed upon join/leave events.

3.2 QoSCBT

QoSCBT [7] can contemporarily consider multiple metrics. A host h wishing to join the group $\mathcal{G}$ sends its request on the unicast path toward $\mathcal{R}$, with its QoS requirements and the indication about whether it is only a recipient in $\mathcal{G}$ or it may be also a source for $\mathcal{G}$. Depending on this information, the join request is processed differently. Each intree node maintains status information concerning the dynamic metrics measuring the local resource availability. The status is updated according to a *soft-state* approach via the periodic exchange amongst neighbours of control messages, that also refresh the resource reservations. An intree router r records:

- for each additive metric m_a: both the maximum value of m_a from r to any downstream destination, $M_D^r[m_a] = max\{m_a(r, dest)\ \forall r's\ downstream\ destinations\}$, and the maximum value of m_a from any downstream source to r, $M_S^r[m_a] = max\{m_a(src, r)\ \forall r's\ downstream\ sources\}$;
- similar information is maintained for multiplicative metrics;
- for a concave metric m_c: the local value of m_c for each downstream interface, and the number of downstream sources.

A join request is catched by the first intree node r it encounters while traveling towards $\mathcal{R}$, and it is processed as follows:

- for an additive metric m_a: the request is accepted if $M_S^r[m_a] + m_a(r, h) \leq req_h[m_a]$ and, if h is also a source for $\mathcal{G}$, $M_D^r[m_a] + m_a(h, r) \leq req_h[m_a]$. Similarly for multiplicative metrics;
- for a concave metric m_c: the request is accepted if $m_c \geq req_h[m_c]$ for the r's downstream interface to h.

If the request is rejected, a notification is sent to h. Otherwise, for additive and multiplicative metrics the request is forwarded up to $\mathcal{R}$, with each intermediate router repeating the test. In case of success, $\mathcal{R}$ sends an acknowledgement to h; each router that receives the ack performs the appropriate resource reservation, and updates its state if needed. For concave metrics, if h is not a source the request is immediately acknowledged by the first router r performing the test, which allocates the requested resource. Otherwise, if r has not any other downstream source, it installs a tentative state and forwards the request upstream to perform the resource reservation. The procedure is repeated until an ancestor is encountered, which already has downstream sources, or the request is rejected. The described mechanism works for many-to-many applications with one active source at a time; this way, the reservation can be shared among all the sources. To avoid conflicts, QoSCBT imposes that a node processes only one join request at a time; other requests received in the meantime stay waiting until the (n)ack for the previous request has been sent.

3.3 QoSPIM

In [8] two techniques are proposed to extend the PIM-SM protocol [15] adding QoS-awareness. In this work we focus in particular on the Tree Information-Based QoS Multicast (TIQM) protocol. TIQM may contemporarily satisfy bandwidth, delay and loss constraints. It exploits the service of an underlying *link state* protocol, that maintains updated information about the network topology and the resource availability. A host h that wants to join a group $\mathcal{G}$, uses the link-state information to compute a subgraph $G' \subseteq G$ obtained from G by pruning the links that do not have enough available bandwidth. Then, h runs the Dijkstra's algorithm to characterize all the paths in G' connecting it to the nodes in $\mathcal{T}$, and having a delay lower than $req_h[delay]$. h chooses the minimum delay path. As the link state protocol, although maintaining the dynamic metrics, could be temporarily inaccurate, h tests the current delay and loss metrics for that path, by sending a join request along the path using source routing. When the request arrives at an intree node, it is forwarded upstream along the tree up to $\mathcal{R}$, and a local reservation is recorded if needed. Every router traversed by the request checks for the actual availability of the requested resources and performs the reservation. If the check fails, a negative ack is sent to h along the reverse path, to release the reserved resources. Otherwise, when the request is received by $\mathcal{R}$, $\mathcal{R}$ sends an ack to h.

4 Analysis of the Described Algorithms

In Table 1 we summarize the main characteristics of the studied protocols. The main drawback of QoSMIC is its dependence on a QoS-sensitive unicast routing. Both the QoSMIC and the QoSCBT protocols operate by checking whether enough resources are available along the path provided by the unicast routing between an intree router and the joining node. Since the unicast routing considers only static metrics[1], the consequence of that policy is that flows tend to converge on the paths with greater resource availability. When those paths are congested, no new service can be established until ongoing transmissions terminate and release the used resources. To some extent, QoSMIC overcomes this problem, thanks to its capability of testing alternative branches starting from different candidates.

A critical aspect of QoSCBT is how the router status is initialized. For some metric (e.g., the bandwidth availability), the initial status equals the static metric. The status is updated by each accepted join request, that results in a local resource reservation. Hence, dynamic metrics can be locally maintained. Other metrics, such as the delay, dynamically vary according to the traffic conditions. In [7], the authors propose to initialize the router status using the information gathered by the join request along its path. If that information is the quality-of-path experimented by the request itself, it concerns the direction *opposite* to that followed by the QoS data, and it does not provide any information about

[1] Dynamic metrics are evaluated during the tree construction procedure.

the traffic congestion on the path towards the joining node. If, by contrast, every router has to measure and piggyback over the request the estimates of its own quality-of-path towards the joining node, this charges the routers with an unacceptable overhead. Moreover, not all the routers could be capable of performing those measurements. On the other hand, the inaccuracy in estimating the queueing delay causes the failure in providing the requested QoS.

In QoSCBT, the lack of concurrency in managing join requests may increase the set-up latency observed by the recipients, particularly at the start of a session when the most part of the receivers join at almost the same time. Another QoSCBT drawback concerns its suitability limited to applications having one active source at a time.

Table 1. Comparison among QoSMIC, QoSCBT and QoSPIM

	QoSMIC	QoSCBT	QoSPIM
#metrics	1	≥ 1	≥ 1
unicast	same metric	independent	link-state
type of metrics	static/dynamic	dynamic	dynamic
multipath	yes	no	yes
tree	shared unidirectional source-based	shared bidirectional	shared unidirectional source-based
tentative state	soft	hard	hard/soft
sources	concurrent	one at a time	concurrent
concurrent JoinRq	yes	no	yes
reservation state	soft	soft	hard/soft
comm.complexity	$O(N)$	$O(N)$	$O(N^2)$

Both the QoSMIC and the QoSCBT communication overheads are $O(N)$. But, QoSCBT has a greater memory overhead because of the recording of the needed status information. The TIQM communication cost is $O(N^2)$ due to the use of a link state protocol. Moreover, it has the greatest computation and memory overhead. The update of the status information must be performed upon either each accepted join request or each leave event. On the other hand, the link state information provides the routers with many alternative paths that can be tested until one with the appropriate characteristics is found. In fact, we expect that TIQM outperforms QOSMIC in the probability of successfully grafting requesting recipients.

Both QoSMIC and TIQM may build either shared unidirectional trees or source-based trees. In the former case, additional mechanisms are needed to reserve resources along the unicast path from each source to the tree root.

5 Performance Evaluation

The above considerations are confirmed by the experimental results. We implemented the algorithms in the frame of the NS-2 simulation package [9]. We simulated a meshed network of 64 nodes, connected by optical links of 2 Mbps bandwidth and variable length in the range 50 to 100 Km. QoS CBR traffic originates in the tree root; we performed our measures with different transmission rates. The background, best effort traffic is uniformly distributed all over the network and uses on average the 33% of the bandwidth resources. The packet size is 512B. Bandwidth reservation is ensured by using the WFQ [16] packet scheduling policy; the queue length is 20 packets. The DV-based unicast routing uses the link capacity as the quality-of-path metric. In the following, we report the results concerning QoSMIC and QoSCBT; the TIQM measurements are currently ongoing. We simulated an interval of 60 sec.

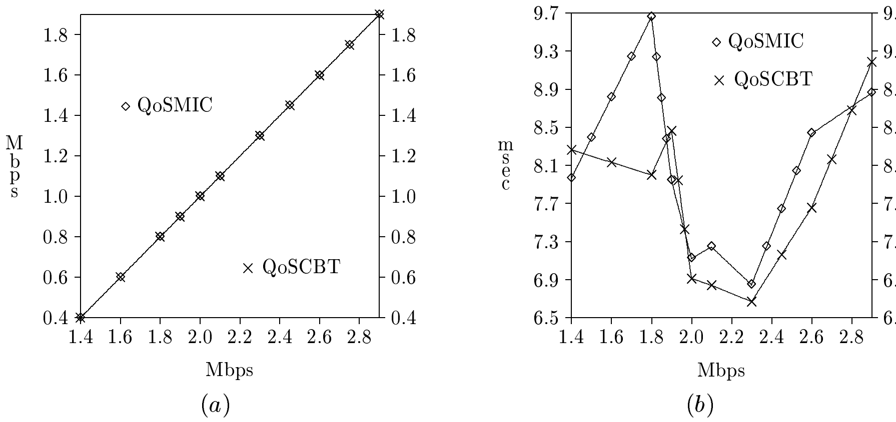

Figure 1. (*a*) Throughput and (*b*) end-to-end delay of the second group vs. offered load, for $|\mathcal{G}| = 10$

We performed the first set of experiments with two QoS sources: the rate of the first source is 1 Mbps, while the rate of the second source varies between 0.4 and 1.9 Mbps. Both algorithms consider the bandwidth availability as the unique QoS constraint. Each source forms its own group; groups partially overlap. The second group is created after the first one has been established. We performed experiments with different membership cardinalities; however, the number of recipients has no impact on the delivered QoS.

Both algorithms guarantee the required bandwidth to the grafted recipients (figure 1(*a*)). For both algorithms, the end-to-end delay (figure 1(*b*)) increases when approaching the network congestion; while it drops when there is no space left for the best effort traffic (both source rates of 1 Mbps), thanks to the pipeline effect. When the second source rate increases, both algorithms successfully perform access control: only the destinations that can be grafted via non overlapped branches join the groups. Hence, on those branches, QoS traffic competes with the best effort traffic only. The observed fair delay and jitter are as well comparable for the two algorithms.

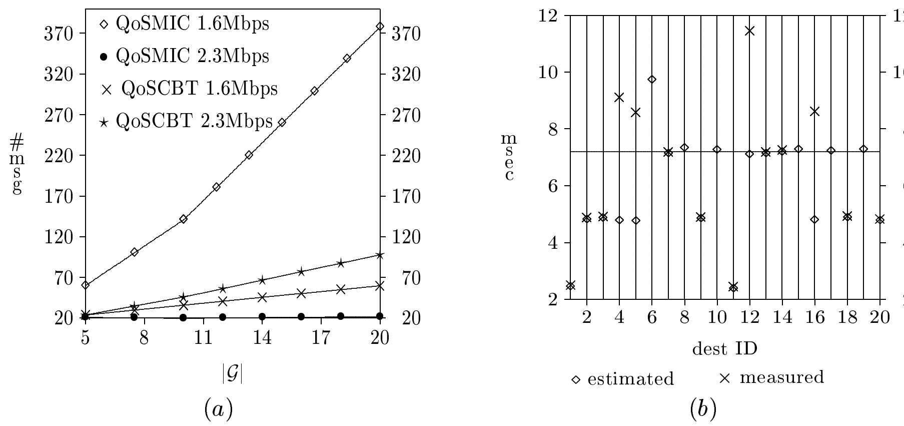

Figure 2. (a) Number of control messages vs. number of destinations, for different QoS traffic loads. (b) Estimated and measured end-to-end delay for the different destinations, with delay constraint of 7.2 msec

By contrast, the algorithms differentiate in terms of costs. In QoSMIC, the need to wait for the `bids` sent by the candidates affects the latency observed by the joining nodes, if Δt is far greater than the round-trip delay between the joining node and the farest intree node. In the opposite case, the joining node could not receive all the `bid` messages, thus possibly reducing its probability of successfully grafting the tree. QoSMIC also has a greater communication overhead than QoSCBT. In figure 2(a), we report the control messages generated to form the second source tree for both algorithms, with respect to the number of recipients. For low rate of the second source (0.6 Mbps), all the destinations are grafted. As QoSMIC characterizes several candidates, each of which sends its own `bid`, its overhead grows more than linearly with the number of recipients. On the other hand, for increasing rate, the testing of alternative branches allows QoSMIC to graft around 10% more recipients than QoSCBT. In the case of network congestion (offered load: 2.3 Mbps), very few intree nodes are candidates, and the QoSMIC overhead drops, differently from QoSCBT that has to send explicit rejection messages to remove the tentative reservations.

We performed experiments with QoSCBT considering both bandwidth and delay constraints. A request is accepted if the estimated transmission and propagation delay over the path from $\mathcal{R}$ to the joining node is lower than the required threshold. Queueing delays are not considered. In figure 2(b), we show the values of the estimated and measured end-to-end delay for each recipient, with only one source generating traffic at 1.6 Mbps, and uniform background traffic as before. With the considered threshold of 7.2 msec., 6 out of 20 receivers have not been grafted. Because of the contention with the best effort traffic and the consequent queueing delay, some destinations (4, 5, 12, 16) are accepted although their actual delay is greater than the threshold. The use of priority packet scheduling does not help in smoothing those peak queueing delays. If the delay bound imposed by the application is strict, the implemented architecture must be enhanced by adding mechanisms to monitor the queueing delay. Anyway, the achieved delay profile is more homogeneous than without any delay constraint (figure 3(a)). This reflects as well on the fair delay: the lower the delay bound

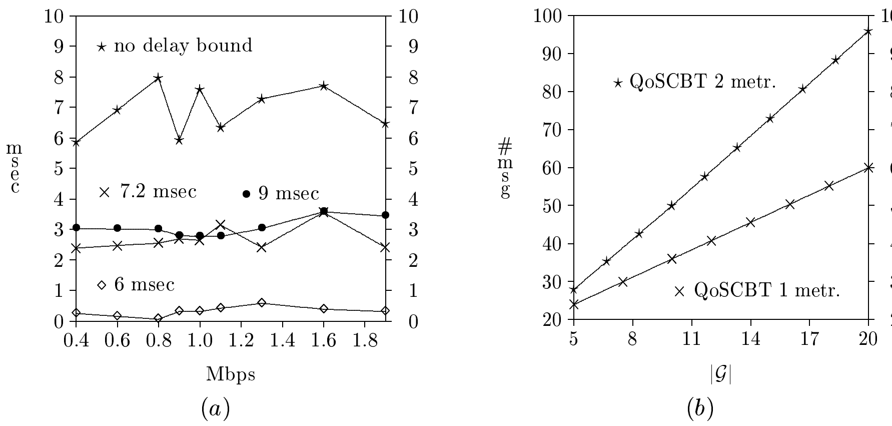

Figure 3. (a) Average delay vs. source rate for $|\mathcal{G}| = 20$ and different delay constraints. (b) QoSCBT communication overhead vs. number of considered metrics, with source rate 0.6 Mbps and different group cardinalities

is, the lower the number of grafted recipients is, and the lower the difference among the delays they perceive is. On the other hand, as delay is an additive metric, the join requests are checked by all the nodes on the path from $\mathcal{R}$ to the joining node[2], thus increasing the communication overhead (figure 3(b)) and the latency (of roughly 1 msec.).

6 Concluding Remarks

In this work, we analyze the behaviours of three QoS-sensitive multicast routing protocols. All the algorithms build a shared tree, and are receiver-driven, thus supporting heterogeneous recipients. The study of their characteristics and the simulation results show that a trade-off exists between the algorithm costs and the probability of characterizing a suitable tree. In particular, this probability grows with the capability of exploring multiple paths towards a given receiver. Possibly, source-based trees could further increase that probability, by allowing a better traffic distribution all over the network, at the expenses of a larger status maintained by the nodes. We will investigate this issue by performing experiments with TIQM, that supports source-based trees as well.

All the algorithms require the modification of the current Internet routers. The choice of which algorithm to adopt depends on several issues, such as (i)

[2] Rather than only by the first encountered intree node, as in the case of bandwidth.

the application characteristics (e.g., one or more sources contemporarily active) and requirements (e.g., one or more QoS constraints), and (*ii*) the dependences on the lower layer services.

Our future work involves the investigation of techniques to perform queueing delay estimation, so that QoSCBT is more accurate in fulfilling the application requirements, without greatly increasing the overhead. Moreover, we are trying to reduce the dependency of QoSMIC and QoSCBT from the unicast routing.

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. Braden R., Clark D., Shenker S.: Integrated Services in the Internet Architecture: an Overview. RFC 1633, Jun. 1994. Work in progress.
3. Pagani E., Rossi G.P.: A Functional Architecture for End-to-end Quality-of-Service in a Multi-domain Network. Proc. 1st IEEE European Conference on Universal Multiservice Networks (ECUMN 2000), pp. 20-34, Oct. 2000.
4. Zhang L., Braden R., Berson S., Herzog S., Jamin S.: Resource reSerVation Protocol (RSVP) - Version 1 Functional Specification. RFC 2205, Sep. 1997. Work in progress.
5. Schulzrinne H., Casner S., Frederick R., Jacobson V.: RTP: A Transport Protocol for Real-Time Applications. RFC 1889, Jan. 1996. Work in progress.
6. Faloutsos M., Banerjea A., Pankaj R.: QoSMIC: Quality of Service sensitive Multicast Internet protoCol. Proc. SIGCOMM 1998, Sep. 1998.
7. Hou J., Tyan H.-Y., Wang B., Chen Y.-M.: QoS Extension to CBT. 9th International Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV 99), Jun. 1999. (Also appeared as an Internet Draft, draft-hou-cbt-qos-00, Feb. 1999. Work in progress.)
8. Biswas S., Izmailov R., Rajagopalan B.: A QoS-Aware Routing Framework for PIM-SM Based IP-Multicast. Proc. GLOBECOM 2000, Nov. 2000. (Also appeared as an Internet Draft, draft-biswas-pim-sm-qos-00, Jun. 1999. Work in progress.)
9. Fall K., Varadhan K.: ns Notes and Documentation. The VINT Project, Jul. 1999. `http://www-mash.CS.Berkeley.EDU/ns/`
10. Handley M.: The sdr Session Directory: An Mbone Conference Scheduling and Booking System. `http://mice.ed.ac.uk/mice/archive/sdr.html`. Apr. 1996.
11. Bless R., Wehrle K.: IP Multicast in Differentiated Services Networks. Internet Draft, draft-bless-diffserv-multicast-01.txt, Nov. 2000. Work in progress.
12. Matta I., Guo L.: QDMR: An Efficient QoS Dependent Multicast Routing Algorithm. Journal of Communications and Networks, Vol. 2, No. 2, Jun. 2000, pp. 168-176.
13. Rouskas G.N., Baldine I.: Multicast Routing with End-to-end Delay and Delay Variation Constraints. IEEE Journal on Selected Areas in Communications, Vol. 12, No. 3, Apr. 1997, pp. 346-356.
14. Borella A., Cancellieri G.: Improving the Cost/Performance Ratio in Multicast Communications. Proc. NOC 2000, Jun. 2000, pp. 187-190.
15. Estrin D. et al.: Protocol Independent Multicast - Sparse Mode (PIM-SM). RFC 2362, Jun. 1998. Work in progress.
16. Shreedhar M., Varghese G.: Efficient Fair Queuing Using Deficit Round Robin. IEEE/ACM Transactions on Networking, Vol. 4, No. 3, pp. 375-385. Jun. 1996.