INTERNET-DRAFT
Inter-Domain Multicast Routing
Expires: 1st October 1998
File: draft-banerjea-qosmic-00.{txt,ps}

Anindo Banerjea (U. of Toronto)
Michalis Faloutsos (U. of Toronto)
Rajesh Pankaj (QUALCOMM)

Designing QoSMIC: A Quality of Service sensitive Multicast Internet protoCol

STATUS OF THIS MEMO

ABSTRACT

We present, QoSMIC, a multicast protocol for the Internet that supports QoS-sensitive routing, and minimizes the importance of *a priori configuration* decisions (such as *core* selection). The protocol is resource-efficient, robust, flexible, and scalable. In addition, our protocol is provably loop-free.

Our protocol starts with a resources-saving tree (Shared Tree) and individual receivers switch to a QoS-competitive tree (Source-Based Tree) when necessary. In both trees, the new destination is able to choose the most promising among several paths. An innovation is that we use dynamic routing information without relying on a link state exchange protocol to provide it. Our protocol limits the effect of pre-configuration decisions drastically, by separating the management from the data transfer functions; administrative routers are not necessarily part of the tree. This separation increases the robustness, and flexibility of the protocol. Furthermore, QoSMIC is able to adapt dynamically to the conditions of the network.

The QoSMIC protocol introduces several new ideas that make it more flexible than other protocols proposed to date. In fact, many of the other protocols, (such as YAM, PIM-SM, BGMP, CBT) can be seen as special cases of QoSMIC. The goal of this document is to present the motivation behind, and the design of QoSMIC, and to provide both analytical and experimental results to support our claims.

1 Introduction

Multicasting can be defined as the distribution of the same information stream from one to many nodes concurrently. In the last few years, multicast routing has attracted a lot of attention from the networks community, since many emerging applications are of multicast nature, such as tele-conferencing, tele-education, and computer supported collaborative work. A multicast connection can substitute for many unicast connections carrying the same information, while reducing the load on the network. Multicast algorithms try to minimize the routing cost of the tree, which forms the simple multicast routing problem, or Steiner tree problem [24]. In practice, the applications and practical constraints complicate the problem with additional requirements [7].

The Internet is a packet-switching network that principally provides best-effort service. That is, there are no guarantees for the services and applications that run over it; applications may "starve", end-to-end delays may be arbitrary, and packets may be lost. Traditional Internet routing protocols do not consider Quality of Service (QoS) metrics. For the support of multicast connections, a multicast backbone (MBone) was introduced as a virtual network "on top" of the Internet [9] in 1992. The MBone is expected to be phased out, as the routers in the Internet become multicast capable. In the rest of this document, we use the term router to imply a "multicast capable Internet router", unless otherwise stated.

Our work is motivated by the need to support QoS-sensitive multicast applications. The marketability of any kind of service depends heavily on its ability to to provide a level of quality. Currently, the services over the Internet are limited by the best-effort nature of the network. However, the scope of our work extends beyond the current use of the Internet to a fully-commercialized environment with competing service providers. We are convinced that such commercial services will need to guarantee their QoS, and that some users will be willing to pay to have such guarantees. The first step in this direction has already been taken; people pay for Internet access. It is only natural to expect that the requirements of these users will increase in terms of the expected quality.

A multicast protocol that considers QoS in its routing phase can create a tree better suited to the needs of QoS-sensitive applications. However, the straightforward approach of exchanging dynamic link-state metrics to compute QoS based routes does not scale to large networks. We can distinguish two categories among Internet multicast protocols based on QoS considerations: QoS-oblivious protocols [1] [10], [19] [17]; and QoS-sensitive protocols [3] [25]. An overview of the above protocols is presented in the next section.

Our protocol was designed with the following primary goals.

- *QoS Support.* We want to provide a framework to support arbitrary QoS requirements of users. To achieve this, we have to consider multiple paths, and handle the link asymmetry, e.g. for satellite connections. Note, that multiple paths can also be necessary for policy reasons.
- *Limited Impact of Pre-configuration Decisions.* We want to limit the impact of any *a priori* configuration decisions, such as the choice of special status routers (PIM-SM, CBT, and BGMP), or a special partitioning of the multicast address space (BGMP [19] [12]).

In our effort to propose an improved protocol, we identified a group of weaknesses of some of the current and proposed MBone protocols. We transformed this group into the following list of secondary design guidelines.

1. *Efficiency.* Our protocol should construct distribution trees that use the network resources efficiently.
2. *Application sensitive.* It should accommodate diverse applications types with minimal user input.
3. *Scalability.* It should scale well for large networks, many groups, large group-sizes etc

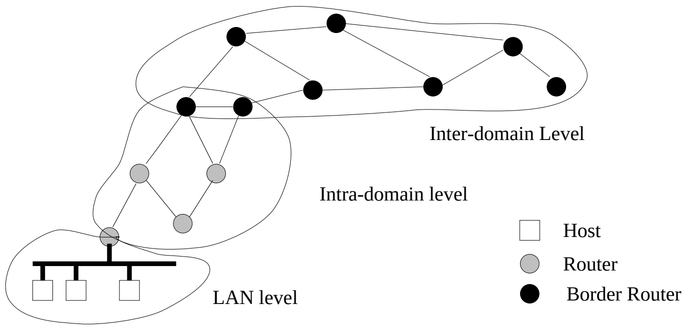


Figure 1: The architecture of the MBone/Internet network.

4. *Robustness.* It should be robust to failures, even of special status components, e.g. *core* routers.
5. *Loop-freedom.* It should not create cycles.

Our protocol constructs trees based on a greedy heuristic [18], that connects each user to the "closest" branch of the existing tree and leads to efficient resource use. Using the greedy approach, our protocol offers alternate paths to enable the support of QoS requirements. The protocol uses dynamic routing information, without assuming an underlying protocol to provide it. Another innovation is that our protocol does not require pre-configuration decisions; its efficiency does not depend on the selection of a special status router, or a special partitioning of the multicast address space. Finally, our protocol can be seen as a flexible framework that encompasses the behaviour of most of the previous protocols, when sensitivity to QoS is not required.

The rest of this document is structured as follows. In Section 2, we present our model and related work. Section 3, we present an overview of our protocol, while in Section 4, we offer a more detailed description. In Section 5, we compare the resource and control efficiency of our protocol with that of other protocols. Section 6 provides some simulation results concerning resource and message efficiency. In Section 7, we present some implementation details of our protocol. In Section 8, we summarize our work.

2 Model Definition and Background Work

Conceptually, the structure of the Internet can be decomposed into three levels (see Fig.1). The workstations (hosts) of the users are connected to a Local Area Network (LAN) such as Ethernet. Each LAN has a *Designated router* which communicates directly with each host using the Internet Group Membership Protocol (IGMP) [6]. This is the first or **LAN level**. We use the term **Destination** to refer to the designated router that has a group-member in its LAN. The designated router is connected with other routers forming domains, which is the second or **intra-domain level**. The domains are interconnected with routers that are called **Border** routers. The network of domains is the third or **inter-domain level**. This creates a three level hierarchy that allows the co-existence of different technologies and facilitates the scalability of the network. We can have different protocols at the intra-domain and inter-domain levels, as long as they can interoperate. Routing at the inter-domain level imposes more restrictions due to scalability considerations. Our protocol could be used in both levels. It is flexible enough to adapt to the needs of different environments, and scales well in the inter-domain case.

A multicast group is associated with a Class-D or multicast address. A host knows the address of the group it wants to join through an advertising or query mechanism such as the Session Description Protocol (SDP) [15]. A multicast group can have multiple sources and the distribution of the packets can be done in two ways. First, each source can create its own distribution tree, called a **Source-Based Tree**, with itself as the root. Second, all sources can distribute their packets using the same tree, called a **Shared Tree**. Source-Based Trees have better end-to-end performance (e.g. lower delay), and distribute the traffic of each group across the network, but lead to large routing tables[1] [2] [23]. On the other hand, Shared Trees concentrate the traffic of a group onto a few links in the network. This concentration is bad in the case where all sources are active simultaneously (e.g. Distributed Interactive Simulation), but it can be beneficial when sources take turns in transmitting (e.g. audio-conferencing). The two approaches have complementary behavior, and are both useful in different situations.

The efficiency of a multicast protocol can be defined by set of functional properties. *End-to-end delay* and *setup time* are properties of interest to the application, while *traffic concentration*, *packet replication*, *routing state* and *control overhead* are important to the service provider. *Scalability* of the above properties to large networks is an overriding concern for protocol and network architecture design. In Appendix A, we present an extended list of the efficiency aspects of protocols.

In our protocol, we compare paths in terms of their ability to support an application at a specific QoS level. Quality of Service (QoS) denotes the user-perceived quality. Recently, Zappala *et al.* [25] used the term Quality of Route (QoR) to refer to multiple static parameters of the route (e.g. link capacity, delay, or reliability), and this was adopted in the YAM protocol [3]. In our work, we suggest the use of dynamic metrics, (e.g. available bandwidth, current delay), because these provide paths that can meet QoS requirements at a given moment. Furthermore, routing with dynamic metrics can respond pro-actively to link congestion. However, exchange of dynamic link-state metrics has scaling problems. Thus, we do not require the use of dynamic metrics in the link state exchange protocol. Dynamic metrics are used instead to evaluate and select from the alternate paths proposed by our protocol. This distinction will become more clear after the protocol overview. To summarize, our protocol enjoys the benefits of QoS routing, without suffering from the scalability restrictions.

For the rest of this document, the term "QoS of a route" denotes the quality that the route is expected to deliver to a connection given its current state, and is calculated from dynamic metrics of properties such as bandwidth or delay. However, the terms "distance" and "proximity" for routers are defined using static metrics of the same properties. It is important to stress that **our protocol is compatible with any metric of the routing quality of a path**, and the specific metric to use is application dependent.

2.1 Previous Work

QoS-oblivious protocols. These protocols provide one route when a new member joins, and QoS is not considered in the selection of the route. Example of such protocols are Core Based Trees (CBT) [1], Protocol Independent Multicasting - Sparse Mode (PIM-SM) [10], Border Gateway Multicast Protocol (BGMP) [19], and Multicast Internet Protocol (MIP) [17]. All these protocols assume **rooted** trees, with a ***core*** router which is the center of the distribution tree. BGMP and MIP imply the use of rooted trees. In BGMP, for example, each domains "owns" an address space and is the root of the distribution tree with such an address. In all these protocols, the tree is the union

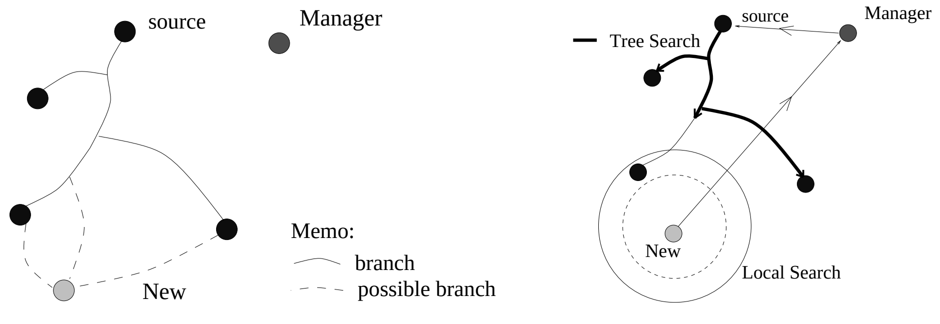
[1] Source-Based Trees require a routing entry per source per group, while Shared Trees require a single routing entry per group.

of the reverse shortest paths between the *core* and each destination. *Reverse* shortest paths are used because they can be computed from the unicast routing database without requiring additional information. CBT creates only Shared Trees while PIM-SM and BGMP, permit the creation of Source-Based Trees for very active sources (high data rate). The MIP protocol introduces novelties on administrative and correctness issues, and guarantees loop-free Shared Trees and Source-Based Trees. The assumption of rooted trees simplifies routing, but introduces the core selection [20] and address partitioning [12] problems, which can affect the protocol performance significantly.

QoS-sensitive protocols. These protocols offer multiple routes to a new member, who selects the best such path. Carlberg and Crowcroft [3] [4] suggested the **YAM** protocol, which creates Shared Tree considering multiple routes. The destination router searches its neighborhood and finds routers that are part of the tree of the desired group. This way, the new router selects the most promising route. Routing in YAM relies mainly on this neighborhood search, but this procedure can increase the control overhead significantly. A suggested method to address this problem requires a sense of source/root domain, which is not applicable in Shared Trees unless we assume an address partitioning mechanism. Zappala et al. [25] suggested a multicast protocol that provides alternate routes to avoid a bottleneck link. This protocol does not seem applicable in the case where QoS suffers from several mildly-congested links. Finally, both protocols use only static routing information, which can lead to poor QoS performance[2]. An analogy from real life is driving: when we drive we want to have the option of multiple routes, but also we want up to date information of the traffic on each of them. Following small streets is faster than driving on a congested highway.

The YAM protocol can be seen to implement the greedy heuristic. The greedy routing scheme has been shown to outperform the commonly used Shortest Paths routing in various analytical [18] [16] [13] and experimental [21] [8] [22] studies. On average, the studies suggest a 10-30% advantage of the greedy approach in the efficiency (cost) of the distribution tree.

3 Overview



(a) Identifying multiple routes.

(b) The Local Search and the Multicast Tree Searchprocedures.

Figure 2: An overview of QoSMIC.

[2] In private communication, the authors of YAM expressed their intention to adopt dynamic information in their protocol.

Our protocol creates Shared Trees by default and Source-Based Trees when needed. In both cases, the protocol offers alternate paths for each connection (see Fig. 2(a)).

We introduce the notion of a **Manager router** of a group. The Manager administers a specific multicast group, and facilitates the joining of the new group members. The fundamental difference between a *core* router and a Manager is that **the distribution tree is not rooted at the Manager.** This way, the selection of the Manager has marginal effect on the topology of the tree. Furthermore, we can have multiple Managers and change the Managers during the lifetime of the group without any data loss, as we will see later in this section.

Joining a group. The Designated router of the new member, which we call **New router**, will try to connect to the most promising path to the tree, according dynamic metrics defined by the application. The New router identifies several intree routers as **Candidates** or potential points to join to the tree. This search is conducted by two procedures: one from the side of the New router (Local Search), and one from the side of the tree (Multicast Tree Search) (see Fig.2.b), both operating in parallel. For the moment, we consider a purely intra-domain or purely inter-domain scenario.

1. *The Local Search Procedure.* The New router searches its neighborhood, using a reverse path multicast of probe messages, with scope limited by use of the time-to-live (TTL) field for its closest intree router. Every intree router that receives the probe message is considered a Candidate router, and responds with an "advertisement message" unicast to the New router. This procedure is also used in the YAM protocol.

2. *The Multicast Tree Search Procedure.* The New router contacts the Manager router of the group, and the Manager router "informs" the tree of the New router. Some intree routers are selected as Candidates. They advertise themselves to the New router with a unicast "advertisement" message. In the next section, we propose several mechanisms, centralized and distributed, to select Candidates.

Eventually, the New router compares all the possible connection paths, and selects the best path according to the needs of the application. The New router sends one more message (JOIN) towards the selected Candidate, to set up the routing state along the path and the chosen router starts forwarding the data. The routing state is soft-state, but pinned, so that as long as the New router keeps sending JOIN refreshes, the route does not change. It should be clear from the above description that our protocol implements a greedy routing heuristic.

The path chosen by the advertisement messages depends on the **static** QoR metrics contained in the routing information base. The specific metric used by the routing protocol depends on the needs of the application. For example, real-time applications will prefer paths of minimum end-to-end delay, while data transfers may prefer paths of maximum bandwidth or low packet loss ratio. Current routing protocols already have the ability to carry multiple static metrics.

However, the advertisement messages can collect up-to-date **dynamic** QoS metrics along the path that they travel (e.g, by interacting with RSVP [26], or using measurement based metrics). Thus, *although the paths that the Candidates choose is restricted by the static information in the routing information base, the New router selects among these paths using dynamic routing information.*

Source-Based Trees. Using Shared Trees minimizes the routing table information to an entry per group. However, Shared Trees may fail in the following two cases. First, when a group has many highly active sources simultaneously, the bandwidth of the shared links may not be able to accommodate all the traffic. Second, when the QoS requirements of a user are not met along the

Router Name	Role
Manager router	Supervises the group and facilitates the joining procedure; does not necessarily participate in the data distribution tree.
Candidate router	Considered as possible joining point for a new connection.
Designated router	Connected to a set of users (e.g. LAN), receives requests from users (e.g., using IGMP) and initiates searches.
New router	Designated router of the LAN containing the new member.
Destination router	Designated router of a LAN that has active group members.
Intree router	All routers that are part of the distribution tree.
Intermediate router	Intree router that is not a Destination or a Source.
Border router	Router that connects multiple domains.

Table 1: The different roles of the routers.

MESSAGE	Explanation
BID-REQ	New router searches locally for intree neighbors
BID	Candidate "proposes" to the New router; Message collects dynamic QoS metrics along the path
M-JOIN	New router contacts the Manager who initiates a Multicast Tree Search
BID-ORDER	The Manager "orders" the intree nodes to send bids
JOIN	The New router sends a message up its selected path to the tree to establish/refresh routing state
PRUNE	Departing Destination tears down unwanted part of tree; can be for a source or for a group

Table 2: An explanation of the messages of our protocol.

Shared Tree, we have to find a different source-to-destination path. In both these cases, we resort to Source-Based Trees. The switch from a Shared Tree to a Source-Based Tree of a specific source is initiated by the Designated router of a receiver. The procedure for establishing the Source-Based Tree is similar to the procedure of the Shared Tree and uses the Local Search and the Multicast Tree Search procedure to identify routers *in the Source-Based Tree*. To avoid packet duplication, the Shared Tree is pruned on a source-specific basis, for the sources for which Source-Based Trees have been established. More details for the co-existence the two types of trees is provided in Section 7.

4 Detailed Description

In this section, we describe the messages and mechanisms used by our protocol in detail. After that, we explain the process of Candidate selection, Manager selection, and interdomain operation. Table 1 lists the different roles of the routers. Table 2 contains a list of the protocol messages.

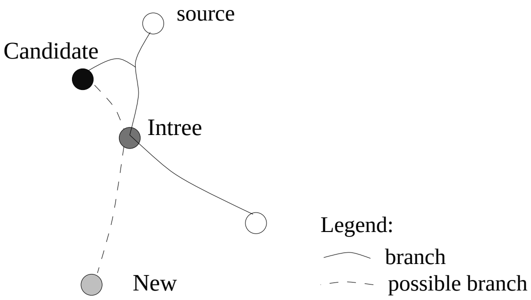


Figure 3: The Take-Over procedure: an intree router that lies in the path from the Candidate to the New router can replace the Candidate.

4.1 The Search for Candidates

We assume that the New router receives a request to join a multicast group. To simplify the presentation, we assume that the new member is a receiver[3]. The Join is received via IGMP on a LAN at the intra-domain level, or from an intra-domain protocol at a Border router at the inter-domain level. If the New router is part of the group already, the connection is established locally. If the New router is not part of the group, the Local Search and the Multicast Tree Search procedures are employed.

Local Search. The New router tries to identify neighboring intree routers.

1. The New router "floods" a BID-REQ message in its neighborhood. Reverse path multicasting with scope controlled by the Time To Live (TTL) field is used to control the message complexity of this phase. This procedure is the same as proposed in YAM, but because we also have the Multicast Tree Search, we can keep the final TTL much smaller. This advantage is quantified later analytically and experimentally.
2. Every intree router that receives a BID-REQ message, becomes a Candidate router, and replies with a BID message, which is unicast to the New router. The BID message on its way collects information on the expected performance of the path, based on dynamic QoS metrics. The Candidate router considers the New router as a tentative dependant, and cannot leave the tree unless the tentative status is timed out.
3. The New router collects the BID messages. The procedure terminates unsuccessfully, if the New router does not receive any replies before the expiration of a timer set for this purpose. Otherwise, we enter the phase of establishing the connection (see Section 4.2).

Multicast Tree Search. The New router contacts the Manager and the Manager causes some of the intree nodes to propose themselves as Candidate routers. The selection of Candidates is an important aspect of our protocol and it is discussed in later in this section. The sequence of actions is as follows:

1. New router sends an M-JOIN message to the Manager of the group.
2. The Manager "orders" a bidding session with a BID-ORDER message. Some subset of the routers that receive the BID-ORDER are selected as Candidates.

[3]A source joins a group in a way similar to a receiver; it connects to the closest router of the Shared Tree. A difference is that the dynamic metrics of the path are collected "towards the tree", i.e., in the direction the data will flow.

3. The Candidates unicast BIDs to the New router. The BIDs are identical to the BIDs in the Local Search.

For both the Local and the Multicast Tree Search, if an intree router receives a BID message (for the same tree), the router takes the place of the Candidate by "dumping" the original BID and initiating a new BID message in a procedure called **Take-Over** (see Fig. 3). This is important to prevent multiple copies of packets. If a router with tentative state (i.e., one that has received a BID but not yet established a connection), receives a BID for the same (tree, New router, Candidate), the BID is discarded to maintain loop freedom.

4.2 Connection Establishment

Having performed the BIDding phase, we will examine how the connection is established.

1. The New router selects the best Candidate according to the dynamic QoS metrics collected by the BIDs.
2. The New router sends a JOIN message to its best BID. This message traverses in the opposite direction the path used by the BID message, and establishes *soft* state along the path.
3. When the chosen Candidate receives the JOIN message, it changes state to the connected state, and starts transmitting data packets on the newly set up path.
4. If an intree router receives a JOIN message for a different Candidate, it performs a TAKE-OVER, terminates the message and starts forwarding data on the newly set up path.

4.3 Leaving a Group

A Designated router receives a leave request through the same protocol that communicated the join request. The request can be for a whole group or for a source of a group. Whenever a router senses[4] a change in the membership, it removes the link from the distribution tree, and checks whether it has become a leaf of the related tree. If so, it sends a PRUNE message up the tree and removes state for the tree from its database, thereby ceasing to be an intree router for that tree.

4.4 Candidate Selection.

During the Multicast Tree Search, the Manager must select an appropriate subset of the intree routers as Candidates. This can proceed in a centralized or a distributed way.

1. *Centralised Selection.* If the Manager has sufficient knowledge of the tree and the network topology, the manager can select the set of Candidates directly. This can happen if the domain is running a link state protocol, or if the manager has access to a routing information database. In this case, the Manager identifies promising Candidates, either on demand or statically. Then, the Manager unicasts the BID-ORDER to them. If the selection is based on accurate information, we can find the right Candidates with limited overhead cost. Note, that with this mechanism, our protocol can behave like CBT or BGMP; a Manager can trivially act as a *core* router by considering itself as the only Candidate.

[4] The membership change can be the result of an IGMP message, a PRUNE message from a neighboring router, or an expiration of the soft state if an update is not received for some time.

2. *Distributed Selection.* In the absence of centralized information, each intree router has to decide whether to become a Candidate in a distributed way. We assume that each router has a static estimate of its distance to the New router. We have the Manager join the tree as a source[5] and multicast the BID-ORDER along the tree. All intree routers receive the BID-ORDER and decide in a distributed way if they should become Candidates. Naturally, the more intree routers become Candidates, the more "accurately" we find the closest Candidate to the New router, but the more the control overhead. We propose the following orthogonal mechanisms that can be used in combination to strike the balance in this trade-off.

   (a) *Directivity.* We discourage distant routers from becoming Candidates. The BID-ORDER keeps track of the (so far) minimum distance of the tree and the New router. A router with a distance greater than this minimum does not become a Candidate, and, if the relative distance exceeds a threshold, the BID-ORDER is not forwarded further[6].

   (b) *Local Minima.* In absence of global knowledge, we select locally optimal routers as Candidates. As the BID-ORDER message travels along a branch, the distance to the New router of the previous two routers is included in the message. If router $i+1$ sees that router i was closer to the destination than both routers $i+1$ and $i-1$, it sends a message back to router i, which becomes a Candidate.

   (c) *Fractional Choice.* We can choose as Candidates a representative fraction $(1/n)$ of either all intree routers or the ones that meet the other two criteria. For the implementation, we only need a $\log n$-bit wrap-around counter in the BID-ORDER message.

Choice of Mechanisms. We identify combinations of the previous mechanisms that seem more promising. The final choice will have to consider the network topology and the traffic behavior. Some preliminary analysis is presented in Section 5. More detailed simulation studies are needed in this direction.

If topology information for the entire domain is available, **Manager Selection** offers the lowest control overhead for a reasonable selection of Candidates. In the absence of such information, **Fractional Choice and Directivity** together is simple to implement, and may lead to satisfactory solution with a careful choice of parameters. **Local Minima, Fractional Choice and Directivity** is more sophisticated and promises improved results, but we want to determine if the gain justifies the slightly increased complexity.

4.5 Manager and Inter-domain Issues

Manager Address Distribution. For the Multicast Tree Search, the New router needs to know the address of the Manager and of the group and both addresses can be made known through the same mechanism (e.g. Session Directory tool [15]). Alternatively, the address of a local Manager within a domain can be broadcasted via the Domain Wide Reporting [14] or can be provided on demand (by the Border routers or an administrative database).

Manager versus Core Router. The main purpose of the Managers is to invoke the Multicast Tree Search. Since Managers are not key routers for data distribution, we can replace a Manager during a session by simply advertising a new Manager. In contrast to a *core* change, a Manager change does not cause any data loss or any change in the distribution tree. For a smooth transition,

[5]Note, that this does not contradict the flexibility of the role of the Manager, since it only transmits control messages and receives nothing. For example, the Manager can change without disrupting the data distribution.

[6]Note that a similar mechanism appears in YAM [3], but it is used to reduce the overhead of its Local Search (YAM needs to assume rooted trees for this).

Symbol	Explanation
t	The maximum TTL value of the Local Search
c	The number of Candidates
w	The average degree of a router
$\|T\|$	The average size of a multicast tree
$\mathrm{Hop}(a, b)$	The average distance in hops from a to b
Hop_{avg}	The average hop distance between Manager and Candidates
$Join(c)$	The message complexity of the joining part with c Candidates

Table 3: Analysis parameters and their symbols.

the old Manager can "resign", after the new Manager has been around for a sufficiently long time (depending on the size of the network).

Multi-Domain Multi-Level Operation. Our protocol can be used at both the intra-domain and the inter-domain level. If the domain already has active group members, the search for Candidate routers is carried out within the domain. If not, the connection has an intra-domain and an inter-domain part. Inside the domain, the New router contacts one or more Border routers. The Border routers find their best path to the group at the inter-domain level. If the inter-domain protocol is QoSMIC, then the Border routers initiate searches using the procedures of QoSMIC. Each Border router that finds a path to the tree proposes itself to the New router (BID message). The New router chooses the "best" Border router. The selected Border router becomes the Designated Border router for this group (or Source-Based Tree). Finally, if one of the levels is running a different protocol, an interface similar to that in [19] will allow interoperation.

Selecting Manager Routers. In practice, we want to have multiple Managers that will co-operate for efficient and scalable solutions with reduced set-up time. For this we have concluded in the following scheme. We have **at least one Manager per domain**. This way the intra-domain and inter-domain protocol can be independent. We prefer **Border routers for Managers**. This way, the Manager can communicate with routers inside and outside the domain more easily. Thus, by default, **we select the Designated Border router of the group to be the Manager** in a domain. In the creation of a session, the Border router nearest to the session initiator becomes the Designated router. For a domain that joins the distribution tree, the selection of the Designated router was described in the previous paragraph. For a Source-Based Tree, we select the source as the Manager in its own domain to simplify the administration of the tree. It is important to note that these choices are the default ones, but as the execution progresses, we can change to more appropriate Managers without any service disruption.

5 The Control Overhead

We compare the control overhead of the YAM and QoSMIC with analytical methods. We do not consider other protocols, since they do not try to achieve QoS and resource optimized trees. Table 3 lists the parameters and their symbols. Most of these parameters depend on the topology and the membership behavior of the applications. In Table 3, the first two parameters can be altered by the protocol; t depends on protocol decisions solely in all cases, while c, the number of Candidates for the Multicast Tree Search is directly defined by the protocol for the centralized case, and indirectly controlled for the distributed cases. Note that the protocol can modify its parameters during the life-time of the group, and this accounts for the adaptability of our protocol.

The Complexity of Selecting Candidates.

The Local Search. Assume w to be the average degree of a router. For simplicity, we consider only one search with the a maximum TTL value of t. We consider the complexity to be a function of the t, since the average degree, w, is given. In a broadcast, a router that receives a message will forward it to $w-1$ links on average. Assuming a reverse path multicasting method, every router in the neighborhood receives such a message exactly once. This way, the complexity[7] of the Local Search with one expanding search is as follows

$$Local(t) = w \cdot (w-1)^{t-1}$$

While for multiple expanding rings the complexity becomes

$$Local(t) = \sum_{i}^{t} w \cdot (w-1)^{i-1}$$

Since YAM depends entirely on the Local Search to find the Candidates, we can assume that YAM will need to perform expanding rings search to keep the complexity under control, while QoSMIC can perform a single search with a small t, which explains the complexity difference in Table 5.

$$Local(t) = \sum_{i}^{t} w \cdot (w-1)^{i-1}$$

The Multicast Tree Search. This procedure exists only in QoSMIC, and can differ depending on which Candidate mechanism is used. The complexity is $c \cdot \mathrm{Hop}(M, CR)$ for the Centralised mechanism and loosely upper-bounded by the size of the tree, $|T|$, in the distributed mechanisms. In more detail, Table 4 demonstrates the complexity of the selection procedure and the relative advantages for each of the four Candidate mechanisms we saw earlier.

The Complexity of Joining. In any case, the joining overhead is low compared to the overhead of the search. Assume we have c Candidates. Each of them sends a BID message to the New router. The New router sends a JOIN message to one selected Candidate. The associated complexity is: $Join(c) = (c+1) \cdot \mathrm{Hop}(CR, NR)$ This complexity is approximately the same in both YAM and QoSMIC as long as the number of candidates selected is kept roughly equal.

We compare the performance of the selection of Candidates in QoSMIC and YAM protocols according to: the message complexity of searching, the type of the selection, the information used in the selection. The comparison is shown in Table 5. In terms of messages, QoSMIC can reduce drastically the complexity given that it can rely on the Multicast Tree Search procedure. In the centralised case it can be limited arbitrarily (down to one Candidate), and if necessary, we can also dispense completely with the Local Search. In the distributed case the Multicast Tree Search complexity is upper bounded by the size of the tree. YAM relies uniquely on the Local Search procedure whose extent should be large enough to reach the tree ($t' > \mathrm{Hop}(NR, CR)$), and thus, we claim that $t' >> t$.

6 Experimental Results

We study several aspects of our protocol through simulations. Here, we present only two of our studies. We compare the resource efficiency of the greedy routing scheme (YAM and QoSMIC)

[7] We calculate an upper bound of the complexity based on the average degree of a the graph. A more precise bound would depend on the topological properties of the graph. Such a study extends beyond the scope of this document.

Label	Method	Messages	Candidates	Pros	Cons
C	Centralised	$c \cdot \text{Hop}(M, CR)$	c	Low control overhead, Local Search can be omitted	Requires some aggregation of information
F	Fractional	$\|T\|$	$f \cdot \|T\|$	Simple	Random selection of Candidates
FD	Fract., Directivity.	$h \cdot \|T\|$	$h \cdot f \cdot \|T\|$	Simple, Limits the extend of the Multicast Tree Search	
LM	Local Minima, Fract., Direct.	$h \cdot (\|T\| + l)$	$f \cdot h \cdot l$	Intelligent selection of Candidates	Control of the number of Candidates

Table 4: Comparing the four Candidate selection methods in the Multicast Tree Search.

Protocol	Type	Routing Information	Number of Messages	Message Complexity
QoSMIC-Centr.	Central./Distr., Adaptable	Dynamic	$w \cdot (w-1)^{t-1} + c \cdot \text{Hop}_{avg}$	Low
QoSMIC-Distr.	Distributed, Adaptable	Dynamic	$w \cdot (w-1)^{t-1} + \|T\|$	Medium
YAM	Distributed	Static	$\sum_i^{t'} w \cdot (w-1)^{i-1}, \ \mathbf{t}' >> \mathbf{t}$	High

Table 5: A comparison of the search procedures of QoSMIC and YAM. Adaptable complexity type means that during the execution the protocol can limit the overhead control while still offering at least one path per join.

Symbol	Protocol	Comments
Offline	—	The off-line greedy algorithm used for reference
QoSMIC	QoSMIC, YAM	Greedy on-line using minimum cost join path
RSP	PIM-SM, CBT, BGMP	Reverse Shortest Paths using a hop metric.
SP	—	PIM-SM, CBT, BGMP **could** do Shortest Paths routing.

Table 6: The protocols in our simulations.

with the Shortest Paths routing scheme of other protocols. We also examine the overhead complexity of the Local Search, which dominates the overhead complexity of YAM and QoSMIC. Other parameters necessary to complete the efficiency profile of a protocol are end-to-end QoS, set-up delay, traffic concentration and robustness.

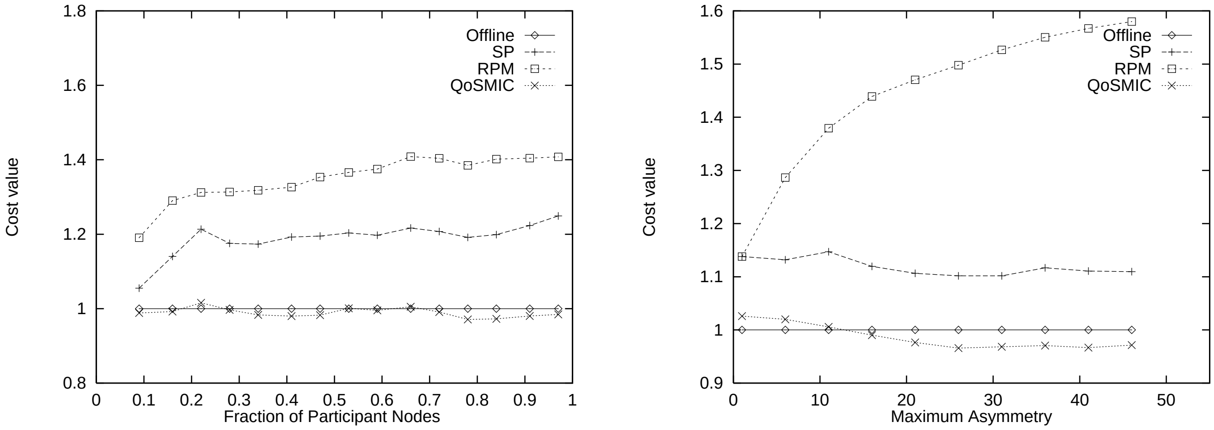
We vary two parameters in the experiments. The **group density** (Grp) is defined as the ratio of the group size over the network size (both measured in nodes). The **maximum asymmetry** A is defined as the maximum ratio of the opposite edges between a pair of routers, for all such pairs. Naturally, for $A = 1$, we have an undirected graph. For each figure, we supply the values of these two previous parameters.

For our network, we use the map of the major routers of the MBone in May 1994 produced by Casner [5]. This graph is appropriate for our experiments, since it represents an actual instance of the inter-domain multicast network. We eliminate routers with only one incident edge, since such routers do not affect routing. We create a directed version of the map replacing its unidirectional edge with a pair of directed edges which we call *opposite edges*. The final graph has 32 routers, 80 pairs of opposite edges, and average degree of 2.5 .

For QoS-sensitive routing, the model should consider more than just a hop metric. We need to include the notions of QoS performance of links and of asymmetry (e.g. satellite links). For this reason, each link is associated with a *cost*; a higher cost can be interpreted as congestion or low QoS ability of the link. In our simulator, we initialize the cost of each pair of directed links to a constant and then introduce uniformly distributed asymmetry between 1 and A. Naturally, the cost of a tree is the sum of the cost of its edges.

The protocols we simulate are presented in Table 6. We present measurements of the path lengths of a new branch and the efficiency of the distribution tree. More accurately, we focus on the Shared Tree that is created between one *core* (PIM, CBT) or one source (YAM, QoSMIC, BGMP) and a set of receivers. Recall that PIM, CBT, BGMP create their Shared Tree using the reverse shortest paths (receiver towards source) based on the *hop count metric* that we denote by **RSP**. We use hop counts rather than the cost metric for RSP, because that is what these protocols do, and also because it is better than using the actual cost in the wrong direction in an asymmetric link. However, we think that these three protocols can easily become cost and direction sensitive, which would correspond to the the Shortest Paths approach (**SP**).

In creating a session, we choose the participants and the *core*/source randomly. We start from the *core*/source and add one receiver at a time. We run every experiment 45 times to smooth the irregularities of special cases. The 95% confidence interval was roughly within 5% of the plotted value, and it is displayed when this does not clutter the figure.



(a) Tree Cost versus group density (b) Tree Cost versus maximum asymmetry

Figure 4: Routing Efficiency relative to the Offline multicast tree.

6.1 Routing Efficiency

In Figure 4, we study the cost of the distribution tree of the different protocols. Figure 4(a) shows the tree cost versus the group density for an asymmetry of ten. In Figure 4(b), we plot the tree cost versus the maximum asymmetry for a group density of 18%. Given the uniform distribution of the asymmetry, the average asymmetry is $(A + 1)/2 \sim A/2$. Based on these figures, we make the following observations.

QoSMIC and YAM create more efficient trees. The greedy routing of QoSMIC and YAM outperforms SP by up to 20% (Fig.4(a)) and RSP by up to 60% (Fig.4(b)), in agreement with previous literature (see Sec.3).

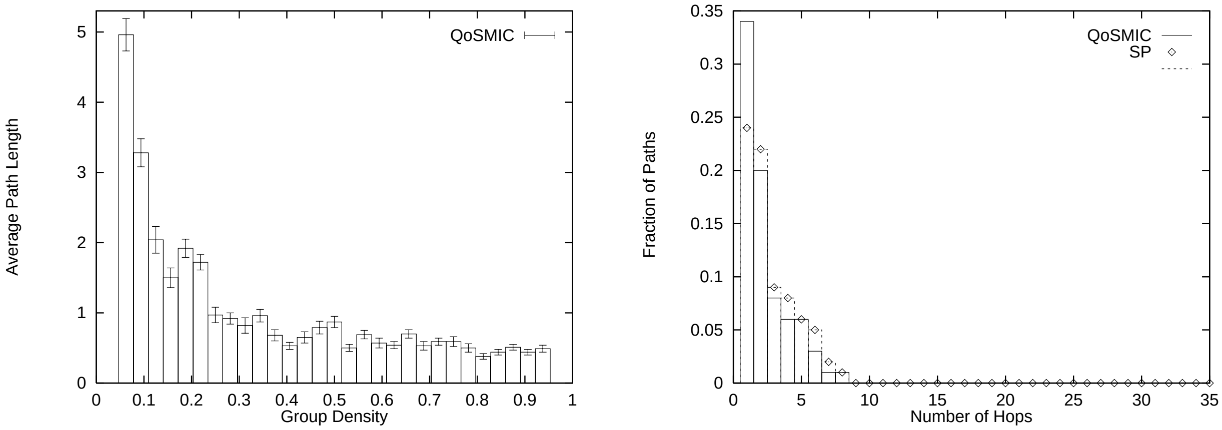
Direction-aware routing is crucial. In asymmetric environments the awareness of the data flow is significant. The direction-aware Shortest Paths approach, SP, outperforms RSP by as much 42%. This suggests that the PIM, CBT, BGMP can benefit greatly from a consideration of the directed cost.

6.2 Overhead Complexity

The complexity of Local Search depends on the TTL value, which should be at least as large as the expected hop-distance between the New router and the closest point to the tree. In Figure 5, we study the paths that QoSMIC and YAM use for their joins. In Figure 5(a), we plot the average path length versus the group density for a symmetric network. In Figure 5(b), we plot the distribution of paths according to their length in a symmetric network for groups up to half the size of the network. For comparison, we plot the path distribution of SP. We extract the following two conclusions.

Using Local Search pays off. In Figure 5(a), path length reduces quickly as the group density increases. This way, the Local Search can be useful even for relatively sparse groups. This is supported also by the evidence in Figure 5(b): more than 50% of the paths (first two columns) have length less than 2. Therefore, we conclude that a Local Search even with a small TTL value can be beneficial.

The importance of Multicast Tree Search. In Figure 5(b), we see that a small percentage of the paths have lengths as large as 8. If we rely uniquely on the Local Search (as YAM does) then



(a) Average path length versus group density. $(A = 1, Grp = 1 - 100\%\ )$

(b) The number of paths versus the path length $(A = 1, Grp = 3 - 50\%)$

Figure 5: The path length of a new join for QoSMIC.

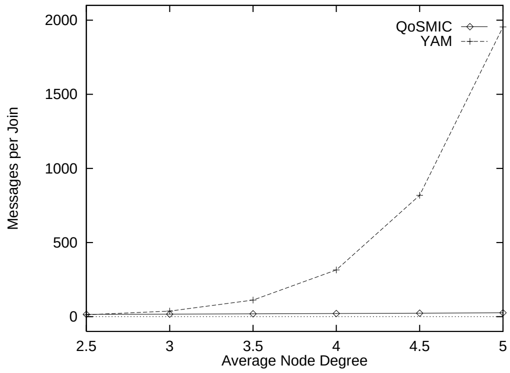


Figure 6: The control overhead of the Candidate search in YAM and QoSMIC.

the TTL of the search should be at least as large or else some routers would be excluded from some distant multicast sessions. This suggests that Local Search alone would need large TTL values, which increases the overhead complexity significantly.

As a conclusion, we want to have a limited Local Search to take advantage of the many "short" joins, but we need a second search mechanism to provide solutions when the Local Search falls short. For this environment, a single Local Search of 2 hops seems a reasonable choice for QoSMIC, while YAM would have to go up to 8.

Scalability of message complexity.

Using data from our simulations, we calculate of the overhead of the searches of YAM and QoSMIC using the analysis of Section 5. Namely, from the experiment of Figure 5(b), we get the number of joins, and the distance of the node from the tree.

YAM: The maximum TTL of a Local Search should be at least as large as the maximum path length or else some routers would be excluded from some distant multicasts. Thus, the maximum TTL is set equal to the path length of the join. We increase the diameter of each expanding ring

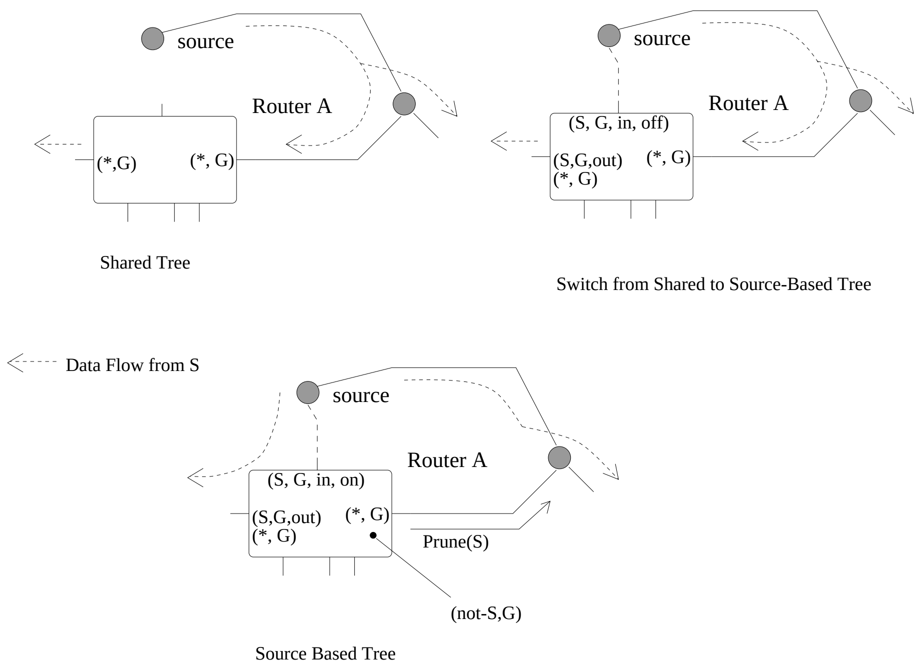


Figure 7: The entries at the routing table for Shared Trees, Source Based trees and their transition. The figure shows the flow of packets from the source.

by one. This way, although the set-up delay may suffer, we attempt to minimize the message complexity of YAM.

QoSMIC: The Local Search does not have to succeed for every join, since we have the Multicast Tree Search to fall back on. This way, we choose a maximum TTL value of 2, as suggested earlier. In addition, we over-estimate the messages of Multicast Tree Search by considering it equal to the number of intree nodes. However, using the mechanisms, the number of messages should be smaller in practice.

In Figure 6, we calculate the message complexity per join for various average degrees. The message complexity of YAM increases dramatically with the average degree, due to its large Local Search. In comparison, QoSMIC avoids this problem: having the Multicast Tree Search to fall back to we can afford to keep the Local Search small. Similarly, YAM would suffer from large networks, where the TTL value of the Local Search would have to be large. As a conclusion, YAM can be applied in small or sparse networks, but QoSMIC scales well to dense or large networks.

7 Implementation Details

The goal of this section is to highlight some implementation details of our protocol.

7.1 Intree Router State

For clarity, when we discuss the routing information of a router, we associate every link with a set of entries for each group. Note, that we try to use the terminology of the PIM-SM proposal [10] [11] to facilitate the readers familiar with these works. Furthermore, some of the actions of our protocol are done in a similar way to that of PIM-SM and we refer the interested reader there.

We can distinguish the following routing entries for each link.

- A (*, G) entry corresponds to the shared tree of a group. By default, it is bidirectional.
- An (S, G) entry corresponds to a source-based tree of source S of group G. It is by definition uni-directional, so we can distinguish IN-only (S, G, IN) and OUT-only (S, G, OUT) entries.
- We enable source-specific pruning that we denote by (not-S, G). A packet that arrives from a link with such state is dropped. The main reason is to avoid packet duplication for destinations that have joined an Source-Based Tree.

We use (*/S, G) as an abbreviation of the expression (*, G) or (S,G).

7.2 Router Actions

Here we study the actions of a router as triggered by control messages.

7.2.1 Joining a session

Join request.

The request (*/S, G) can be either to the Designated router (via the IGMP protocol) or the border router (via an intradomain protocol).

If the router is already part of the requested (*,G) or (S,G) tree, the join is accommodated there. If not, the router considers itself a destination router that we denote as New router, and performs the Local Search and Multicast Tree Search procedures. For the Multicast Tree Search, the New router sends an M-JOIN(*/S, G, New router) to the Manager of the group. For the Local Search, the New router broadcasts BID-REQ(*/S, G, New router, TTL) message in expanding rings. The messages follow a reverse path multicast. The TTL is initialised to a value as we saw before. A timer is set to monitor the responses. If the timer expires before any responses, then a new broadcast (ring) takes place with an increased TTL. The value of the TTL is bounded by a default value or is explicitly specified. When the timer expires for the maximum TTL value, without any responses, the procedure terminates unsuccessfully.

BID-REQ(*/S, G, New router, TTL) message.

On reception of a BID-REQ(*/S,G) message, a router that is part of the tree in question replies with a BID message. In a Source-Based Tree, the Candidate incorporates in the message estimates of the QoS it experiences (quality of the path from the source to the Candidate), when these are available. Note, that this way, we can maximize the end-to-end QoS that the New router will receive. In more detail, the BID message is of the form BID(*/S, G, New router, Candidate, Metrics): it includes the address of the Candidate, and estimates of the QoS of the path. The BID message on its way updates its metrics at each router to estimate the quality of the path.

MJOIN(*/S, G, New router) message.

This message is unicasted to the Manager of the group. For a (*,G) message, the manager sends BID-ORDER(*/S, G, bid-mechanism). The "bid-mechanism" represents the selection mechanisms that we saw earlier. The BID-ORDER can be either multicasted over the tree, or unicasted to pre-selected routers depending on the mechanism in use.

BID-ORDER(*/S, G, New router, bid-mechanism) message.

The actions on this message depend on the "bid-mechanism". In general, the message is either unicasted to the chosen Candidate or multicasted over the tree. In the first case, on receiving the message, the router unicasts a BID(*/S, G, Metrics) to the New router. In the second case, the router examines the criteria defined by the bid-mechanism, and if they are fulfilled, a BID message is sent to the New router, and the BID-ORDER is forwarded further along the tree.

BID(*/S, G, New router, Candidate, Metrics) message.

On receiving this message, if the router is the New router specified in the message, it updates a list of the best BIDs. An intree router invokes the TAKE-OVER procedure, and the router replace the Candidate. Any other router that receives a BID message, updates the Metrics field and forwards the packet.

JOIN(*/S, G, New router, Candidate) message.

This message travels towards the selected Candidate. When the Candidate router receives the message, it considers the connection as permanent (as opposed to tentative) and sets-up the path. Any intree router that receives the message invokes the TAKE-OVER procedure and starts forwarding data.

7.2.2 Leaving a Group

The leave request (*/S, G) can be either to the Designated router (via the IGMP protocol), the border router (via an intradomain protocol), or with a PRUNE message, or the result of state time out (assuming soft state refresh mechanism). If the request arrived from a LAN with active group members, the router does not do anything. In any other case, the link is removed from the distribution tree. Then, the router examines whether it has become a leaf of the tree.

1. If the router has become a leaf of the (*/S,G) tree[8], it sends a PRUNE(*/S,G) message to its only tree link.
2. If the router has two or more links in the tree (tentative or real), it does not take any further actions.

7.2.3 The TAKE-OVER procedure.

An intree router that receives a message between the Candidate and New router, may be able to take the position of the Candidate(see Fig. 3). This could lead to shorter delays and more efficient trees. This interception can happen in the messages: BID, JOIN.

- The message is (*, G).
  - If the router is in (*, G), then TAKE-OVER.
  - If the router is in (S,G), then just forward the message.
- The message is (S, G).
  - If the router is in (S,G), then TAKE-OVER. Optional refinement: if the current QoS from the source to here is worse than the expected QoS along the bidding path, then switch to the best path.
  - If the router is in (*,G). If the current QoS is not worse than the QoS of the bidding path, then TAKE OVER. If not, then just forward the message.

[8]Note, that leaf here implies also that the router does not have any hosts attached to its LAN.

7.3 Source Specific State

The source-specific pruning is similar to that of PIM (see Fig. 7). A router that is part of both the (S, G) and the (*, G) tree, ignores packets from source S that do not arrive from the (S,G, IN) link. Furthermore, when a router receives an (S,G) packet from another link (the link that the Shared Tree used for the S-packets), it sends an S-specific prune message on that link. We denote by (not-S, G) the state of such a Shared Tree link.

When we switch from the (*, G) to the (S, G) tree (see Fig.7), we want to minimize the data loss by stopping the reception of packets from the (*,G) tree only when data starts to arrive from the (S,G) tree. We use a flag SBT-FLAG, which is similar to the SPT-bit of the PIM-SM protocol. When the request for Source-Based Tree is sent, we install an (S,G, SBT-FLAG= OFF) entry, but we continue to forward S-packets that we receive from the (*,G) link. The first data packet from the (S,G) link sets the flag (SBT-FLAG= ON). When this happens, we drop any S-packets coming from the (*, S) link, and we send S-specific PRUNE(S) messages along these links to stop the transmission of the S-packets from the previous router. This continues recursively along the path that does not require S-packets anymore.

8 Conclusions

We propose QoSMIC, a protocol for supporting QoS-sensitive multicast applications over the MBone. Our protocol identifies multiple paths, and selects the most promising one using dynamic information that the control messages collect. Our protocol creates Shared Trees by default and destinations switch to Source-Based Trees in order to accommodate the needs of different applications. This switch can take place for highly active sources or when the QoS requirements of receivers are not met.

Our protocol includes the main concepts of the YAM protocol [3], and introduces several new ideas. Below, we list the characteristics of our protocol that differentiate it from YAM and/or the other protocols.

1. *Limited Impact of Pre-configuration Decisions.* We separate the management from the data transfer functions, by introducing the concept of the group Manager in the place of the *core* router (PIM, CBT, BGMP, MIP). The choice of the Manager does not affect the distribution tree. This way we eliminate the problems of *core* router selection and the address partitioning problems of other protocols.
2. *Dynamic Routing Information.* Our protocol uses dynamic routing information without relying on external mechanisms/protocols for information exchange.
3. *Scalability.* We introduce the Multicast Tree Search, which allows us to reduce the scope of the Local Search. The complexity of the Local Search appears as the bottleneck in YAM.
4. *Efficient Resource Use.* QoSMIC uses a greedy routing policy, which has been shown to outperform the Reverse Shortest Paths routing used by other protocols.
5. *Sensitivity to direction.* QoSMIC takes care to compute metrics in the direction that the data will flow. This ensures much better performance in asymmetric networks.
6. *Robustness.* The only special status router of QoSMIC are the Managers. If they fail, the recovery can be fast and with minimal data loss. Having multiple managers further enhances the robustness.
7. *Adaptivity during execution time.* In the MBone it is rather difficult to foresee the behavior of the application and the users. Our protocol can adapt to the changing network conditions and

requirements during the life of the group, for example, by changing the protocol parameters (see Section 5) or by switching Managers.

Our simulations and analyses support the above claims.

Future work. Several functions of our protocol, such as the candidate selection, can be implemented in different ways. A number of parameters of the protocol also need to be selected. While we have some analytic bounds on the expected behaviour, more detailed evaluation is needed to finalize the details. Currently, we are simulating our protocol and intend to implement a prototype. In parallel, we are interested in the QoS metric selection problem: we are examining the type of routing information appropriate for the different stages of our protocol, to describe the QoR and QoS of the routes. We believe that with the appropriate routing information our protocol can become an efficient solution for QoS-sensitive applications.

References

[1] A. Ballardie. Core Based Trees (CBT version 2) multicast routing. Internet-Draft: IETF RFC 2201, 1997.

[2] T. Billhartz, J. B. Cain, E. Farey-Goudreau, D. Fieg, and S. G. Batsell. Performance and resource cost comparisons for cbt and pim mulitcast routing protocols. *IEEE Journal of Selected Areas in Communications*, 15(3):304–315, April 1997.

[3] K. Carlberg and J. Crowcroft. Building shared trees using a one-to-many joining mechanism. *ACM Computer Communication Review*, pages 5–11, January 1997.

[4] K. Carlberg and J. Crowcroft. Yet another multicast (yam) routing protocol: Specification version 1. Unpublished manuscript, 1997.

[5] S. Casner. Major MBONE routers and links. Available from ftp.isi.edu:mbone/mbone-topology.ps, 1994.

[6] S. Deering. Host extensions for IP multicasting. Internet-Draft: IETF RFC 1112, 1989.

[7] C. Diot, W. Dabbous, and J. Crowcroft. Multipoint communications: A survey of protocols, functions, and mechanisms. *IEEE Journal of Selected Areas in Communications*, 15(13):277–290, April 1997.

[8] M. Doar and I. Leslie. How bad is naive multicast routing? *Proc. IEEE INFOCOM*, pages 82–89, 1993.

[9] H. Eriksson. Mbone: The multicast backbone. *ACM Communications*, 37(8):54–60, 1994.

[10] D. Estrin, D. Farinacci, A. Helmy, D. Thaler, S. Deering, M. Handley, V. Jacobson, C. Liu, F. Sharma, and L. Wei. Protocol independent multicast-sparse mode (PIM-SM): Protocol specification. Internet-Draft: IETF RFC 2117 available from ftp://ftp.ietf.org/internet-drafts/, 1997.

[11] D. Estrin, D. Farinacci, A. Helmy, D. Thaler, S. Deering, M. Handley, V. Jacobson, C. Liu, F. Sharma, and L. Wei. The PIM architecture for wide area multicast routing. *IEEE/ACM Transactions on Networking*, 4(2):153–161, April 1997.

[12] D. Estrin, M. Handley, S. Kumar, and D. Thaler. The Multicast Address Set Claim protocol. Internet-Draft:draft-ietf-idmr-masc-00.txt, 1997.

[13] M. Faloutsos, R. Pankaj, and K. C. Sevcik. Bounds for the on-line multicast problem in directed graphs. *Proceedings of 4th International Colloquium on Structural Information and Communication Complexity (SIROCCO '97), Monte Verita', Ascona, Switzerland July 24-26*, pages 81–98, 1997.

[14] W. Fenner. Domain wide multicast group membership reports. Distributed in the IDMR mailing list. To appear as a Working Draft, 1997.

[15] M. Handley and V. Jacobson. SDP: Session description protocol. Internet Draft. Work in porgress., 1995.

[16] M. Imase and B.M. Waxman. Dynamic Steiner tree problem. *SIAM Journal on Discrete Mathematics*, 4:369–384, 1991.

[17] M. Parsa and J. J. Garcia-Luna-Aceves. A protocol for scalable loop-free multicast routing. *IEEE Journal of Selected Areas in Communications*, 15(13):316–331, April 1997.

[18] H. Takahashi and A. Matsuyama. An approximate solution for the Steiner problem in graphs. *Math. Japonica*, 24(6):573–577, 1980.

[19] D. Thaler, D. Estrin, and D. Meyer. Grand Unified Multicast (GUM): Protocol specification. Internet-Draft: draft-ietf-idmr-gum-00.txt, 1997.

[20] D.G. Thaler and C.V. Ravishankar. Distributed center-location algorithms. *IEEE Journal of Selected Areas in Communications*, 15(13):291–303, April 1997.

[21] B. M. Waxman. Routing of multipoint connections. *IEEE Journal of Selected Areas in Communications*, pages 1617–1622, 1988.

[22] B. M. Waxman. Performance evaluation of multipoint routing algorithms. *Proc. IEEE INFOCOM*, pages 980–986, 1993.

[23] L. Wei and D. Estrin. The trade-offs of multicast trees and algorithms. *International Conference on Computer Communications and Networks*, 1994.

[24] P. Winter. Steiner problem in networks: a survey. *Networks*, 17:129–167, 1987.

[25] D. Zappala, D. Estrin, and S. Shenker. Alternate path routing and pinning for interdomain multicast routing. Technical Report USC CS TR 97-655, U. of South California, 1997.

[26] L. Zhang, S. Deering, D. Estrin, S. Shenker, and D. Zappala. RSVP: A new resource reservation protocol. *IEEE Network*, September 1993.

A Efficiency Parameters of Multicast Protocols

The efficiency of a multicast protocol can be defined by set of functional properties.

1. The efficiency of the multicast tree can be measured by the *packet replication* which reflects the number of packets that a single piece of information causes. In more detail, this can be defined as the number of packets exchanged between neighbor routers, P_x, over the number of packets with distinct content, P_d, over the number of receivers, M: $P_x/(P_d \cdot M)$. With this definition, we can characterize trees independently of the amount of data transmitted (P_d) or the group size (M).

2. The *end-to-end delay* is defined as the time it takes a packet to travel from the source to a destination router.

   The *maximum and average end-to-end delay* over all source and destinations pairs could be used to characterize the end-to-end performance of the tree.

3. The *traffic concentration* can be defined as the maximum link load over the average link load (for the traffic of a given multicast group or of all groups).

4. The *control overhead* represents the number of control packets that are required to support multicast connections.

5. The *routing table* is the table maintained by each router to support multicasting.

6. The *set-up time* is the time it takes a user to join a group: the time from the join request until the arrival of the first data packet.

CONTACT INFORMATION
Anindo Banerjea
Dept. of Electrical and Computer Engineering
University of Toronto
10 King's College Road
Toronto, Ontario
M5S 3G4 Canada
banerjea@comm.toronto.edu

Michalis Faloutsos
Dept. of Computer Science
U. of Toronto
Toronto ON
M5S 3G4 Canada
mfalou@cs.toronto.edu

Rajesh Pankaj
Qualcomm Inc
6455 Lusk Blvd
San Diego CA 92121
USA
pankaj@qualcomm.com