Distributed Quality of Service Multicast Routing with Multiple Metrics for Receiver Initiated Joins

Miguel Rio Peter F. Linington

Computing Laboratory
University of Kent at Canterbury
Canterbury, Kent, CT2 7NF
United Kingdom

Abstract

This paper describes a novel method of building multicast trees for Real Time Traffic with Quality of Service constraints.

There is a wide range of heuristics to calculate the optimal multicast distribution trees with bounds on the maximum delay from the source to all members. However these heuristics require all the members to be known in advance and assume the existence of a centralized service.

We present a heuristic - Best Cost Individual Join (BCIJ) - that joins members one by one, randomly to the existing tree. The method doesn't need previous knowledge of the group members. Trees are dynamically built when each member arrives in the group.

A distributed method - Multiple Metric Broadcast (MMB) - for nodes to obtain the best valid path to the existing tree is also presented. MMB is inspired by Reverse Path Forwarding and broadcasts queries to the network that reach existing on-tree members. Theses reply with the best valid paths to the joining member. The member then selects the best path. This avoids the use of any centralized service and the need for link-state information to be available in any node.

Evaluation presented shows that the BCIJ produces trees with better cost than existing centralized heuristics and that MMB doesn't have a major effect on the network if the group participation is sufficiently large.

1 Introduction

The future Internet will have to accommodate lots of applications that require multicast capabilities. Examples are audio and video conferences, distributed white boards, distributed simulation, on-line gaming or virtual reality environments.

In order for multicast to be efficient a distribution tree has to be calculated and established in the network. The final goal is to minimize the total number of packets to be transmitted. This problem becomes harder when applications have Quality of Service [1] requirements like bandwidth or delay. The calculation and construction of the distribution tree has to take these constraints into account.

The current Internet infra-structure for Multicast is the MBONE [2]. Hosts join multicast groups through IGMP [3] to the local router and routers join the distribution tree using some standardized multicast routing protocol like DVMRP [4] and MOSPF [5] for source based trees or CBT [6] and PIM [7] if center based trees are being used.

All These protocols use a best-effort policy with no place for QoS constraints. It is the responsibility of a signalling protocol like RSVP [8] to allocate resources and check if the QoS constraints can be met. But because the routes are already established the signalling protocol may reject a call that could have been established through other routes.

To overcome this problem work on the IETF is being done to implement QoS routing [9], where the routes are established taking into account the QoS wished for by the application. In this paper we propose a method for building these trees that doesn't require knowledge of the members before the session begins.

In the next section we describe some theoretical work done to address this problem and the query/answer distributed methods that provide a framework for our model.

In section 3 we propose a new heuristic followed by a distributed method of obtaining the necessary information in section 4.

We then address some practical considerations in section 5.

Evaluation of tree efficiency, traffic overhead and required state memory is made in section 6. We terminate with the description of some related work and with conclusions and future work.

2 Background

2.1 Centralized heuristics

From the theoretical point of view networks and Multicast groups have always been modelled as graphs. A network is modelled by a graph $G =< V, E >$ where V is a set of vertices and E is a set of edges with a set of metrics.

The problem of finding the set of edges with minimal total cost is known as the steiner tree problem [10] and is known to be NP-complete. Therefore some heuristics [11],[12] try to minimize the total cost without obtaining the optimal solution.

The problem gets even harder if the group has a bound on delay (e.g.audio) and the final solution must include, for all members, a path to the source that obeys to the delay bound. Some heuristics that solve this problem are Kompella's [13], Waters' [14] and Sun's [15].

A serious drawback of this approach is that it requires some form of centralized service which collects link-state information, calculates the trees and uses a signalling protocol to establish them. An alternative would be to replicate all link state information and group membership in all nodes and apply the same heuristic on-the-fly in each one. A similar technique is used by MOSPF. This would, however, require complete synchronisation and a large amount of information on every node.

2.2 Query/Answer Distributed Methods

A different approach is used by YAM [16] and QoSMIC [17]. A node wishing to join sends a query to the network. When the message reaches on-tree members, these calculate the "cost" of joining the tree through them and send answers back to the joining node. The query messages may update one or more metric while they traverse the network. A route may also be traced in each message, and then included in the answer messages.

Although these methods do not specify which metrics to use as the cost or the way messages are propagated in the network they provide a good starting point because they don't require any kind of centralized service.

3 Best Cost Individual Join

The heuristics presented in the previous section require that the set of members of the multicast group is known before the session starts. This may not be a likely scenario in many situations. They also require a centralized service that collects link-state information, calculates the tree and establishes it.

Our heuristic - Best Cost Individual Join (BCIJ) - joins members one by one to the existing tree. Each node joins the existing tree through the closest node already in the tree. By "closest" we mean the node in the tree at a smaller cost from the joining node, providing that the session delay bound is met. The meaning of cost is not specified here. It can be a monetary fixed amount or, for example, a function of available bandwidth.

The first non-source member to arrive at the group joins directly to the source. this may involve intermediate nodes. Other members follow joining to the closest on-tree node (which is not necessarily a group member).

Each on-tree node maintains the value of the delay from the source to itself, so that this can be added to the delay from it to the joining node to give a proposed total delay.

To search for all possible and useful paths an exhaustive search is made in the network.

The steps of the heuristic are (for each member randomly chosen):

1 - Search for all possible valid paths to any of the on-tree members

2 - Choose the path with smallest cost, for each the delay bound is met.

3 - Join member to the tree by this path

4 - Calculate the delay from the source to the new on-tree members

This heuristic can be used by a centralized service that obtains requests for joining and then uses a signalling protocol to add the new nodes through the best resulting on-tree member. Nevertheless it can be implemented in a complete distributed way as described in the next session.

4 Protocol Implementation: Multiple Metric Broadcast

In order for the new member to have access to the best costs and best delays to all nodes already in the tree a method to propagate the queries through the network has to be defined. This assumes that the node doesn't have access to the link state updates of the entire network.

Protocols like YAM rely on a "Pull" model in which information is requested on demand. Queries are propagated trough the network, on-tree members reply with metric(s) and the originator of the request chooses. The messages record the route as they travel across the network.

Query/Answer methods use RPF (Reverse Path Forwarding) [18] to broadcast a message. RPF was first proposed as a way of broadcasting packets to the entire network without flooding it.

A joining node using RPF sends a message to all its links. Every other node replicates the message on every link except the incoming and if and only if the message arrived on

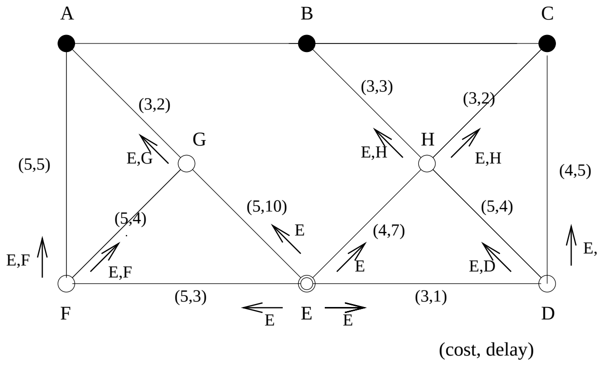

Figure 2. Reverse Path Forwarding

the link used by the unicast routing protocol to reach the original sender from this node.

As applied here messages stop at on-tree nodes reducing the total number of messages that need to be sent An example can be seen in figure 2 where nodes A, B and C are already in the tree and node E is joining the group. Arrows are labelled with the route of each message and in parenthesis the cost and delay of each link is indicated. In this figure as in all other presented on this paper arrows show the direction of the joining messages and not the direction of the data flow, which is the opposite.

For example node G propagates message $< E, G >$ and not the message $< E, F, G >$ because this one didn't arrive on the interface that G would use to reach E (assuming that a best cost policy is being used by the unicast routing protocols).

Each link in the network will, therefore, pass one message in each way providing that the correspondent nodes are not already in the tree.

At each node metrics in the messages can be updated and nodes added to the route description.

```
IF DestinationNode = ALLNodes THEN
BEGIN /* Broadcast case */
    IF IncomingLink = Routing[SourceNode]
    THEN OutgoingLinkSet ← AllLinks -
                                        IncomingLink
    ELSE OutgoingLinkSet ← { }
END
    ELSE /* Unicast case */
OutgoingLink = Routing[DestinationNode]
```

Figure 1. RPF Algorithm

Figure 1 shows in pseudo-code the algorithm each node using RPF runs.

This method has the disadvantage of only using one metric (in this case cost or delay) because it uses the unicast routing tables.

To overcome this problem, we propose MMB (Multiple Metric Broadcasting) instead of RPF. The algorithm is presented in figure 3 Messages are replicated on every link except the incoming one but because we are dealing with more than one metric we have to accept messages arriving on every interface. Counters for the best cost and best delays are kept at each node for every member. These counters are only kept during the joining process and are not needed after the node chooses a route. Each message is only propagated if it contains better metrics than the best ones received so far. The metrics in each message are updated with the delay and cost of the link in which they are propagated.

In figure 4 we can see the messages produced. Node G propagates both messages that arrive from F and E because one has better cost and the other has better delay. The same for node H.

```
BestDelay=$\infty$
BestCost=$\infty$
IF DestinationNode = ALLNodes THEN
BEGIN /* Broadcast Case */
    IF MessageDelay $<$ BestDelay OR
    MessageCost $<$ BestCost THEN
    BEGIN
    OutgoingLinkSet ← AllLinks - IncomingLink
    IF BestDelay $>$ MessageDelay THEN
                    BestDelay = MessageDelay
    IF BestCost $>$ MessageCost THEN
                    BestCost = MessageCost
     TransmitUpdatedMessage()
    END
END
ELSE /* Unicast Case */
OutgoingLink = Routing[DestinationNode]
```

Figure 3. MMB algorithm

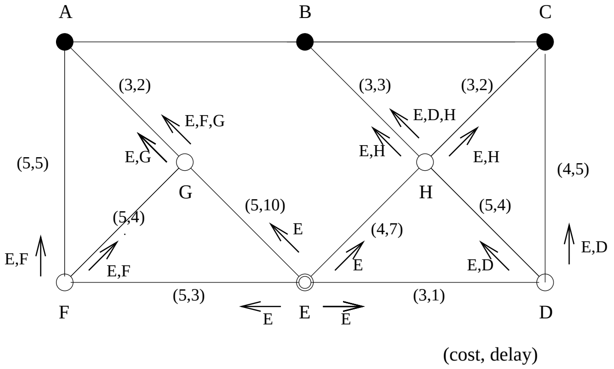

Figure 4. Multiple Metric Broadcast

This method will produce a larger number of messages but will find every useful path to the tree. The best path will then be the one that would be found by the heuristic described in section 3. Like previous work the meaning of cost is not defined. Nevertheless the method requires a coherent configuration policy for all nodes for calculating the cost of each link.

If the MMB message contains a traffic description enabling to calculate the necessary bandwidth for the group, nodes can stop the propagation of MMB messages if they estimate that there is no available bandwidth. This is equivalent of removing edges corresponding to saturated links in the heuristic of the las section.

5 Optimizations and Problems

The last section described the essential of MMB. Nevertheless there are some practical considerations to deal with. We discuss these in this section.

5.1 MMB with delay bound testing

In big networks, when the tree is still small the number of MMB messages can grow considerably. Some of these messages may be useless because they contain in their updated metrics a delay that is already larger than the delay bound.

If the session delay bound in known *a priori*, (which is not difficult since it can be transmitted by the application) it can be included in the query message An extra test can then be made at each router to prevent useless messages (corresponding to paths that don't obey to the delay bound) being propagated.

Each router checks, then, if the delay of the path correspondent to that message is smaller than the delay bound. If it is not, the message is discarded. This reduces the total number of messages and limits the search space to the nodes in the network within the delay bound.

5.2 The "impossible path" problem

As said earlier, in order to prevent constant flooding of the network messages stop when they reach an on-tree member. In a normal case it would be useless for a on-tree member to propagate the request to other nodes because the cost would always be bigger then joining through itself.

But in some cases an "Impossible Path" problem can arise, since the only possibility solution may have to cross the tree. This is illustrated in figure 5 where node A is joining to an already established tree. Suppose the delay of path <A,C> is larger than the delay of <B,C> and <B,C,D,S> already exists as the lowest cost path, but the delay of <A,C,E,S> is bigger then the delay bound. The only possibility of node A joining the tree without breaking the delay bound is through path <A,C,E,S>.

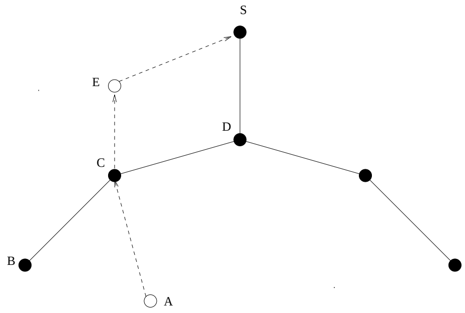

Figure 5. The impossible path problem

This scenario doesn't happen very frequently as will be shown in the evaluation scenario. We had three choices to solve this problem:

- Let messages propagate to all the network
- Each on-tree node keeps track of all possible paths to the source and answer with the best one
- Each on-tree node keeps track of the minimum delay path to the source and sends it in the reply

Because of the small frequency with which this happens, we opted for the third choice. The first would imply messages flooding the network and the second would imply more state in routers.

The final algorithm for a on-tree member receiving a query is presented in pseudo-code in figure 6. Each on-tree member receiving a query checks if it contains the delay bound. If it doesn't it sends back to the source of the request the best delay path to the source and the normal path to the source.

If the information about the delay bound is available it sends the path corresponding to the normal path to the source. If a valid path didn't arrive it sends the path correspondent to the best delay path to the source.

5.3 Degenerate trees

The use of Multiple metrics introduces the possibility of degenerate trees (a normal tree may only have one possible path between any two nodes). In the usual scenarios with only one metric this may only arise with temporarily inconsistencies in the unicast routing tables. But because two nodes may join using a different metric (cost or delay) trees may degenerate. This may be seen in figure 7. Node

```
IF (delay bound information is NOT
        present in query)
    BEGIN
    Send best delay path to source
    Send normal path to source
    END
ELSE
    BEGIN
    Send Normal path to source
    IF (no valid path yet identified)
        Send best delay path to the source
    END
```

Figure 6. Algorithm for on-tree nodes

D joined through the best cost path. But because of the bigger delay on link $< E, F >$ E had to join through node A, creating two valid paths to reach node F.

The consequence of this is that node F will receive the same data traffic from A and B creating packet duplication and useless consumption of bandwidth.

The tree has, then, to be reconfigured by pruning redundant links. The node receiving data from more than one neighbour (in the example, node F) must send prune messages on the link with bigger delay (link <A,F> in the example). Nodes that don't have any downward member repeat the process upwards pruning unnecessary links (a process similar to DVMRP). This process will increase the total cost of the tree but the tree becomes valid and the delay bound is always met.

The timing of the pruning message may be crucial because data may be lost if the pruning message is sent too early or data packets can be duplicated if the pruning message is delayed. It is out of the scope of this paper any study on the criteria of pruning timing. Two alternatives are sending a pruning message when duplicate data is received or configuring appropriate timers to trigger such messages.

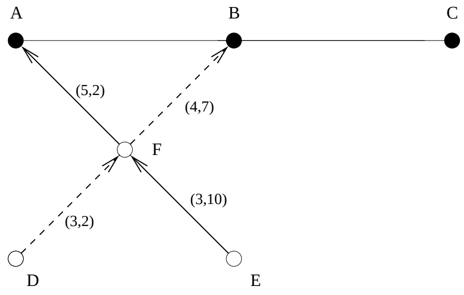

Figure 7. Degenerate tree

5.4 Asymmetric links

Sometimes, due to policies (or certain types of subnet technologies like satellite links) routes can be asymmetric. Messages between two nodes may not be able to follow the same path in both directions. Nevertheless the propagation of MMB messages is not affected since messages are sent hop by hop. If final data will travel from node A to node B but node B cannot send messages directly to node A the message may follow another path. The metrics are updated with the link state information of link A $\rightarrow$ B. This assumes that both nodes in all links have knowledge of the link state information in both directions of the link. MMB messages are in this case sent from A to B with the possibility of going through intermediate nodes. These nodes, however, do not check or change the content of the messages.

6 Evaluation

We evaluated our method with respect to two parameters: Tree efficiency and traffic overhead.

6.1 Tree Efficiency

To compare efficiency we generated random networks using the Waxman model [19] with 100 nodes. We then generated groups between 5 and 95 members (with a step of 5 members) and applied several heuristics, calculating the costs of the resulting tree.

For each scenario we also calculated the resulting tree of our BCIJ algorithm. In figure 8 we plot the cost of all the methods compared with the Minimal Spanning Tree (MST) cost. MST uses Kou's [11] heuristic to calculate the minimal spanning tree between all members. Note that this heuristic doesn't look at delay. It produces trees which may not obey the delay requirement. We also include the results obtained using Dijkstra's algorithm to optimize delay.

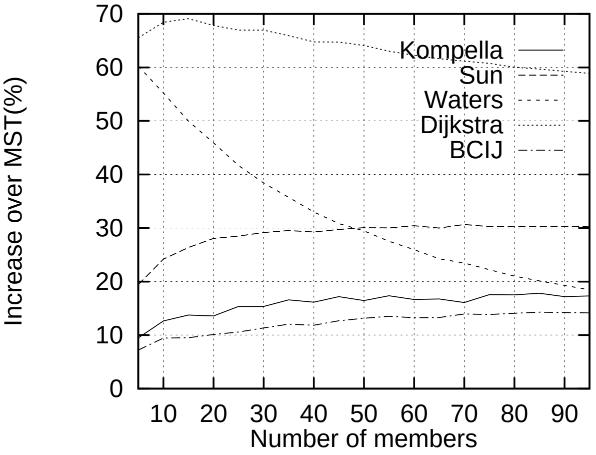

Figure 8. Percentage excess cost over MST

Our heuristic always performs better than every other heuristic in spite of joining the nodes one by one in random order. Its efficiency improves slightly Kompella's results, a well known heuristic for its efficiency.

6.2 Traffic Overhead

As could be seen from figure 4, MMB may produce more messages than normal RPF. In order to quantify this overhead we simulated random networks once more using the Waxman model and performed discrete event simulation using the package provided by Watkins [20]. Nodes are selected randomly to join the network and events corresponding to MMB messages are triggered. We used 200 random networks with 10 runs for each.

The results can be seen in figure 9 where we plotted the average number of messages per link in each join. The first members produce more messages but as soon as the tree is being built (not only more members cease to propagate messages but at some point the branches of the tree "divide" the network isolating the propagation area) the number of messages drops.

The curve labelled MMB2 shows the results when the delay bound is checked. Here no significant difference between MMB and MMB2 were found because the networks had a small delay diameter (the biggest delay between any two members). If the networks were bigger the difference would certainly be bigger.

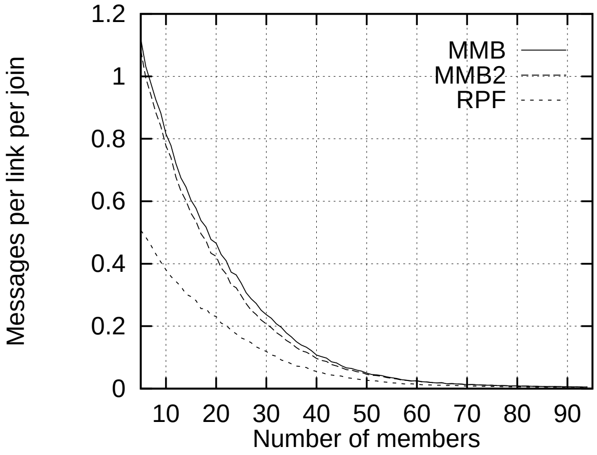

Figure 9. Messages per link per join

For small groups the overhead of MMB is quite high, although in the same order of magnitude (around 100% more for the first 5 members). But as soon as the group grows, the difference is reduced (50% more with 20 members and almost the same after 60 members. This indicates that for big groups MMB doesn't have a major effect on the network.

To better understand this number we plot in the graph of figure 10 the number of links that receives 0,1,2,3,4 and 5 messages. The number of links that receives 5 messages is each join is never noticeable. The number of links with zero messages, and therefore not affected by MMB overhead, grows considerably as the group grows.

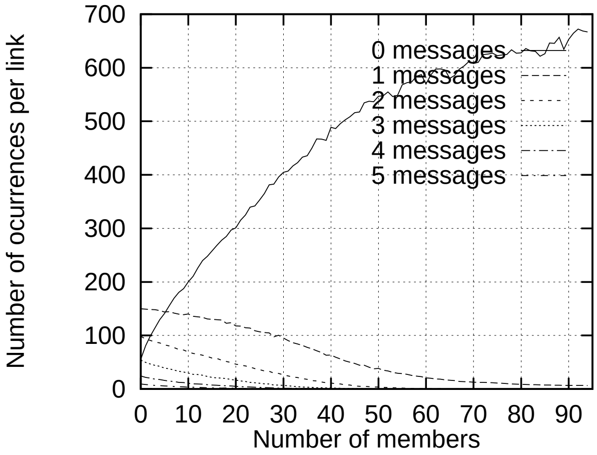

Figure 10. Number of Occurrences per link

As said earlier one of the reasons why the number of messages is reduced as soon as the group grows is because they "hit" the tree. In figure 11 we plot the probability of two non-members of the tree being disconnected, that is, every path between them has at least one on-tree node. As the group grows this probability increases, which explains the decrease of messages shown in figures 9 and 10.

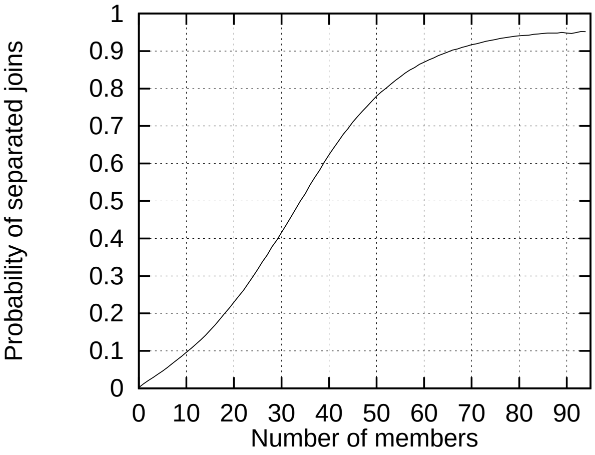

Figure 11. Probability of Disconnection

Finally we shown the percentage of impossible path cases found as the group grows. This corresponds to the scenario where no valid path can be found without crossing the boundaries of the already established tree. The percentage of such occurrences varies between 0% and 0.032 %. The small amount found even with big groups justifies the choice made in section 5.2.

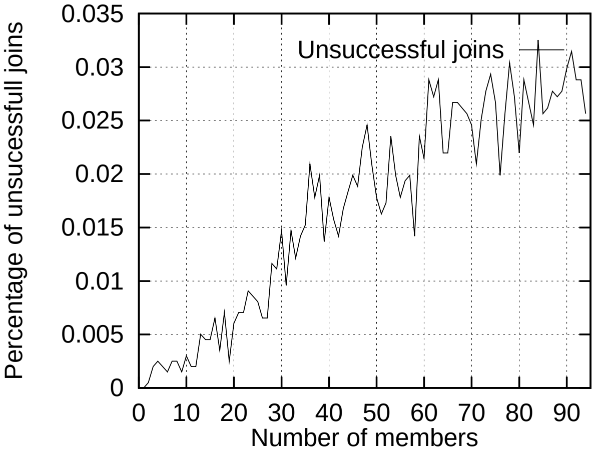

Figure 12. Percentage of cases with "impossible path" occurrences

7 Related Work

In the literature the QoS Routing has been addressed with several and different approaches. Wang [21] proposed a method that considers propagation delay and available bandwidth to optimize, in a centralized way, the best possible routes. Apostolopoulos et al. study the efficiency of QoS Routing in [22]. Hodel [23] describes an extension to PIM called policy tree multicast routing (PTMR) with some extensions to RPF to deal with asymmetric links and with QoS constraints.

Vogel [24] proposes QoSFinder. QoSFinder is based on a path vector protocol that takes into account throughput, delay and loss rate of individual segments.

The most similar technique to our work is RIMQoS recently proposed by Fei [25]. RIMQoS also joins members one by one with multiple metrics without previous knowledge of group members but it requires total propagation of link state information to each router. This is a significant difference from BCIJ/MMB and makes it less attractive than this proposal.

8 Conclusions and Further Work

This paper proposes a new heuristic for receiver initiated joins to multicast distribution trees. The heuristic uses cost and delay as metrics and members can join randomly. Multiple Metric Broadcast is proposed to distribute the calculation of the best joining route/node.

Our evaluation showed that the trees are more efficient than the ones produced by existing heuristics and that MMB doesn't produce much traffic overhead for big groups. We also addressed some practical considerations that improve the original MMB and solve some of its problems.

The evaluation of MMB with delay bound testing has not yet be done to an acceptable extent due to the limitations of the simulation scenario. Nevertheless, intuitively, this extra test reduces the search space considerably in very large internets for applications/flows with strict delay bounds.

As further work we include the study of inter-domain MMB. For worldwide sessions where participating hosts are in different areas/autonomous systems the straightforward propagation of MMB messages to all the Internet is not efficient because it is very unlikely that the majority of the areas has potential members for the session. An Hierarchical method is necessary to scale MMB to this scenario. Work done under BGMP [26] may be used as a starting point. If some form of hierarchical aggregation of link state information, like the one proposed by Lee [27], it can also provide feedback for a inter-domain use of MMB.

References

[1] Paul Ferguson and Geoff Huston. *Quality of Service, delivering QoS in the Internet and in coorporate networks*. Wiley, 1998.

[2] Hans Eriksson. MBONE: The Multicast Backbone. *Communications of the ACM*, 37(8):54–60, August 1994.

[3] S. Deering. RFC 1112 - Host Extensions for IP Multicasting. 1989.

[4] D. Waitzman, S. Deering, and C. Partridge. RFC 1075 - Distance Vector Multicast Routing Protocol. 1995.

[5] John Moy. Multicast Routing Extensions for OSPF. *Communications of the ACM*, 37(8):61–66, August 1994.

[6] Tony Ballardie, Paul Francis, and Jon Crowcroft. Core Based Trees (CBT). In *Proceedings of SIGCOMM'93*, pages 85–95, 1993.

[7] Stephen Deering, Deborah Estrin, Dino Farinacci, Mark Handley, Ahmed Helmy, Van Jacobson, Ching gung Liu, Puneet Sharma, David Thaler, and Liming Wei. Protocol Independent Multicast-Sparse Mode(PIM-SM), Motivation and Architecture. 1996.

[8] L. Zhang, B. Brandon, D. Estrin, S. Herzog, and S. Jamin. ReSource ReserVation Protocol (RSVP) - Functional Specefication, Internet-Draft. March 1997.

[9] Eric Crawley, Raj Nair, Bala Rajagopalan, and Hal Sandick. RFC 2386 - A Framework for QoS-based Routing in the Internet. August 1998.

[10] Ian Stewart. Trees, Telephones and tiles. *New Scientist*, November 1991.

[11] L. Kou, G. Markowsky, and L. Berman. A Fast Algorithm for Steiner Trees. *Acta Informatica*, 15:141–145, 1981.

[12] E.N.Gilbert and H.O. Pollock. Steiner Minimal Trees. *SIAM Journal on Applied Mathematics*, 16(1), 1968.

[13] Vachaspathi P. Kompella, Joseph C. Pasquale, and George C. Polyzos. Multicast Routing for Multimedia Communication. *IEEE/ACM Transactions on Networking*, 1(3):286–292, June 1993.

[14] A. Gill Waters. Multicast provision for high speed networks. In *Proceedings of 4th IFIP Conference on High Speed networking (liege, Belgium)*. Elsevier Science, December 1992.

[15] Q. Sun and H. Langendoerfer. Efficient Multicast Routing for Delay-Sensitive Applications. In *Proceedings of Second International Workshop os Protocols for Multimedia Systems (PROMS'95)*, pages 452–458, 1995.

[16] Ken Carlberg and Jon Crowcroft. Building Shared Trees Using a One-to-Many Joining Mechanism. *Computer Comunication Review*, 27(1):5–11, January 1997.

[17] Michalis Faloutsos, Anindo Banerjea, and Rajesh Pankaj. QoSMIC: Quality od Service sensitive Multicast Internet Protocol . In *Proceedings of SIGCOMM'98*, volume 28. ACM, October 1998.

[18] Yogen K. Dalal and Robert M. Metcalfe. Reverse Path Forwarding of Broadcast Packets. *Communications of the ACM*, 21(12):1040–1048, December 1978.

[19] Bernard M. Waxman. Routing of Multipoint Connections. *IEEE Journal on Selected Areas in Communications*, 6(9):1617–1622, December 1988.

[20] Kevin Watkins. *Discrete Event Simulation in C.* McGraw Hill International, 1993.

[21] Zheng Wang and Jon Crowcroft. Qality-of-Service Routing for Supporting Multimedia Applications . *IEEE Journal on Selected Areas in Communications*, 14(2):1228–1234, September 1996.

[22] George Apostolopoulos, Roch Guerin, Sanjay Kamat, and Satish K. Tripathi. Quality of Service Routing: A Performance Perspective. In *Proceedings of SIGCOMM'98*, volume 28, pages 17–28, October 1998.

[23] Horst Hodel. Policy Tree Multicast Routing: An extension to Sparse Mode source Tree Delivery. *Computer Communication Review*, 28(2):78–97, April 1998.

[24] Ronny Vogel, Ralf Guido Herrtwich, Winfried Kalfa, Hartmut Wittig, and Lars C. Wolf. QoS R-Based Routing of Multimedia Streams in Computer Networks. *IEEE Journal on Selected Areas in Communications*, 14(7):1235–1244, September 1996.

[25] Aiguo Fei and Mario Gerla. Receiver-Initiated Multicasting with Multiple QoS Constraints. In *Proceedings of INFOCOM'2000*, March 2000.

[26] Satish Kumar, Pavlin Radoslavov, Dave Thaler, Cenzig Alaettinoglu, Deborah Estrin, and Mark Handley. The MASC/BGMP Architecture for Inter-Domain Multicast Routing. In *Proceedings of SIGCOMM'98*, volume 28, pages 93–104. ACM, August 1998.

[27] Whay C. Lee. Topology Aggregation for Hierarchical Routing in ATM Networks. *Computer Communication Review*, 29(2):82–92, April 1995.