Towards Multi-class Based Multicast Routing

Maria João Nicolau[1], António Costa[2], Alexandre Santos[2], and Vasco Freitas[2]

[1] Departamento de Sistemas de Informação,
Universidade do Minho, Campus de Azurém,
4800 Guimarães, Portugal
joao@uminho.pt
[2] Departamento de Informática,
Universidade do Minho, Campus de Gualtar,
4710 Braga, Portugal
{costa,alex,vf}@uminho.pt

Abstract. While Differentiated Services reach maturity, and a few per hop aggregate behaviors are being standardized, little efforts are being carried out to enforce class differentiation by selecting alternative routes. Within a differentiated services multicast scenario, multiple multicast forwarding trees must be found, one per Class of Service (CoS), in order to comply with different per-class Quality of Service (QoS) requirements. This paper presents a new multicast routing protocol enabling per class multicast tree computation. The proposed heuristics enable directed trees establishment, instead of reverse path ones, due to the importance of link asymmetry within an environment which is, essentially, unidirectional. The main assumption supporting this work is that a per class path computation may well complement, at routing level, node level differentiation techniques, thus providing per class differentiated handling. The strategy presented is also useful for network traffic engineering as it potentially enables traffic distribution along different network links.

1 Introduction

Routing multicast traffic requires the construction of a distribution tree (or set of trees). Data packets are delivered using that tree, thus the major goal of the routing protocol is to build a tree with minimum cost. The problem of finding such a tree is NP-complete and is known as *Steiner Tree Problem*[1]. Plenty of heuristics have been proposed to efficiently find multicast trees [2]. The one mostly used by multicast routing protocols consists of building a spanning tree by adding each participant at a time, by means of finding the shortest path from the new participant into the nearest node of the spanning tree. Such a tree is called *Reverse Path Tree*. This heuristic assumes that links connecting any two nodes are symmetric, in other words, assuming that link costs in each direction are equal. However, links can be asymmetric due to different reasons, thus links costs are likely to be different in each direction. Therefore reverse-path routing in asymmetric networks may lead to poor routes. Finding a minimal multicast tree in asymmetric networks, called the *Direct Steiner Tree Problem*, is also NP-complete. There are some theoretical studies [3], focusing on directed graphs, aiming to present approaches to this problem. However, most of the deployed multicast routing protocols, like CBT[4] and PIM-SM[5] are based upon reverse path routing.

In addition, most of the multicast applications are QoS sensitive in nature, thus will benefit from the QoS, or even CoS, support from the underlying network, if available. As well as for unicast routing, there are two different approaches in order to provide QoS to multicast routing: per flow and per class routing. The first one performs routing at flow level. Several strategies have been proposed [6] [7], most of them relying on flooding in order to find a feasible tree branch to connect a new member. The underlying idea is to obtain multiple paths where a new member may connect to the tree. Among candidate paths the new member selects the one that is able to satisfy its QoS requirements. This strategy is suited within the Integrated Services model (IntServ)[8] that aims to provide QoS service guarantees for each individual flow crossing the network, by means of resource allocation.

The main strength of the IntServ model is its ability to provide service guarantees by means of (state-full) resource reservation. However it has several weaknesses too. Each router is required to maintain state information for each flow, thus, scalability problems do arise in operational environments. In addition a significant amount of processing overhead is required within each router, and the connection setup time may even sometimes be greater than the time required for the transmission of all the packets belonging to a specific flow.

The goal of Differentiated Service Architecture (DiffServ) [9] is to provide the benefits of different CoS levels while avoiding the limitations of the IntServ model. This is accomplished by aggregating traffic into specific classes, thus changing the scope from QoS (and per flow) to CoS (per class) guarantees. As DiffServ does not maintain any per flow information, connection setup costs are also eliminated. Most differentiated services implementation proposals make use of control algorithms for aggregating service levels, packet marking and policing, and preferential treatment of marked packets within the network. The issue of routing as a means to enhance aggregate QoS has not yet received the necessary attention. In presence of DiffServ networks, per flow path computation is not adequate. Instead, per class path calculation must be made, and so multiple multicast trees must be computed in order to satisfy different QoS requirements of different traffic classes.

In this paper a new multicast routing protocol is proposed enabling per class multicast routing implementation. The proposed protocol takes link asymmetry into account as it defines a *shortest-path-tree* based routing strategy as opposite to a *reverse-path-tree* based one. This is an important feature because when routing constraints are introduced links become asymmetric in terms of the quality of service they may offer, thus link costs are likely to be different in each direction. Therefore reverse path routing is not adequate to address Quality of Service Routing.

2 A model for Multi-class Based Multicast Routing

In this section a new model for implementing a QoS aware Multi-class Multicast Routing is presented. This model supports the Multicast Routing Protocol proposed in this article. Implementation details are given in the next section.

First, a multiple shared tree mechanism is proposed in order to give receivers the ability to joining the group without knowing where are the sources located. Nevertheless, sources are the elements in better conditions to define QoS requirements since they are the ones generating traffic. Having multiple sources per group, with perhaps different QoS requirements, one may have different data flows of different classes of service. In this situation, receivers must join a group with no restrictions in terms of traffic classes they are able to receive. Furthermore, they must be able to receive all classes of all sources, at least in the starting period of group membership.

The multiple shared tree mechanism proposed is inspired in Protocol Independent Multicast-Sparse Mode (PIM-SM)[5] with trees rooted at a Rendez-Vous Point (RP) router. A shared tree per class of service available is needed, in order to give sources the ability to start sending data in any class. It is assumed that the total number of classes of service "available" has a pre-established upper limit and is small when compared to the number of participants.

Data packets originated by sources are sent towards the RP router, previously marked according to source defined QoS parameters. The RP router forwards data packets from sources through one of the shared trees, based on their class of service. Receivers must connect to all of the RP shared trees when joining the group.

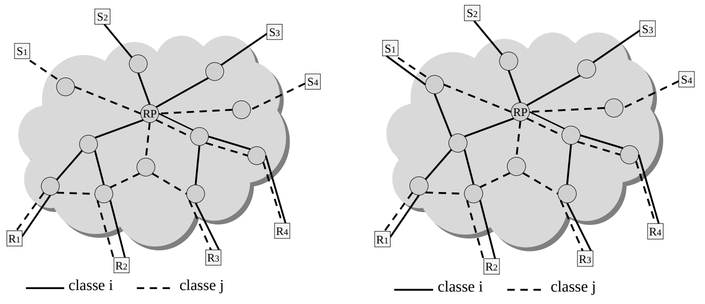
At this point, the question lies on how to build several distinct shared trees, one per class of service. Explicit join requests must be sent by the receivers towards the RP router. When RP router receives a join request it must send back to the new receiver an acknowledge message per class through the best unicast path for that class. Routers, along those paths, receiving such acknowledge message may then update their routing tables in order to build new multicast trees branches. Updating is done basically by registering with the multicast routing entry for that tree, the acknowledge message's incoming and outgoing router interfaces.

The multiple RP shared tree mechanism, presented so far, does not really allow receivers to specify their own QoS requirements. Traffic flows from sources to receivers through one of the shared trees, according only to the QoS parameters defined by sources. How can a receiver, after a starting period, specify a given requirement? Can a receiver demand for a reclassification of a source multicast traffic?

This issue cannot be accomplished by a shared tree, but it may be met if the receiver joins a source-based tree. When initiating the join to source procedure, the receiver should include in the join request the desired Class of Service. It is up to the source to decide whether or not to accept the join, knowing that when accepting a join, traffic in the requested class of service must be generated.

In this situation, each source may face several distinct requests of several distinct receivers for different classes of service within the same group. At the limit, for larger groups, there may be requests for all classes. Even with this worst case situation scalability problems do not arise because the total number of different classes will be much smaller than the total number of receivers. In practice this implies one source-based tree per class of service, unless some order relationship between the classes can be established.

When accepting a join for a new Class of Service, a source must generate an acknowledge message, addressed to the corresponding receiver. This procedure is similar to the one described for the construction of the shared trees. But in this situation only one join acknowledge message is generated per join request.



(a) Shared Trees

(b) Shared and Source-based Trees

Fig. 1. QoS Parameters defined by sources and receivers

Figure 1(a) illustrates a scenario with four receivers initially connected to a RP router by two different shared trees, constructed for *class i* and *class j* respectively. There are also four sources, two of them generating traffic marked for *class i* and the other two generating traffic for *class j*. In Figure 1(b) after a short period of time, the receiver `R2` decides to join source `S1`, requesting the class of service `i`. That single request originates a new source tree, rooted at source `S1` for Class of Service `i`. Note that all the routers along this source tree should stop receiving data from that source through the shared tree in the *class i* in order to prevent duplicate data packets. To accomplish this, a mechanism similar to "prune of source S in shared tree", proposed in PIM-SM specification, must be implemented. In addition, the designated router should stop receiving packets from that source in any other class unless there are other receivers in the same group attached. Therefore the "prune of source S in shared tree" mechanism should be used too when designated router receives the join acknowledge message.

When participants leave a group, additional mechanisms must be implemented to tear down state, and eventually cut out tree branches. As the multicast trees are built from RP or sources towards the receivers, the usual prune mechanism should be modified. To prevent an overload of RP and source nodes and an amount of unnecessary control messages in the de-construction tree process, an additional field should be included in the multicast routing tables, with the identification of the upstream neighbor in the tree. The prunes must be sent directly towards that router instead of being sent to the RP or sources nodes. We believe that this field may be used to implement the periodic tree refresh, too.

3 Multi-Class based Multicast Routing Protocol Implementation

Multi-class based Multicast Routing Protocol (MCMRP) is an implementation of the strategy described in the last section. MCMRP is based on Directed Trees Multicast Protocol (DTMP)[10] and uses Class-of-Service Link State Protocol (CoSLSP).

DTMP is a multicast routing protocol that builds directed trees instead of reverse-path-ones. A complete description of these protocol, with implementation details and comparative results with PIM-SM may be found in [10]. DTMP uses both shared trees and source based trees, like PIM-SM, in order to get the advantages of the both strategies. It is suited for use in asymmetric networks where link costs between any two nodes are different in each direction.

Multi-Class Based Multicast Routing Protocol (MCMRP) extends DTMP in order to implement class based multicast routing. Another major element of MCMRP is the unicast routing protocol in use. Although MCMRP is independent of the underlying unicast routing protocol, it must be a multi-class enabled unicast routing protocol. In other words, the unicast routing protocol must be able to find the unicast routes that can meet the QoS requirements of each Class of Service. In order to build a new tree branch for each Class of Service the multicast routing protocol will search the unicast routing table for the unicast path that is more adequate to satisfy the QoS requirements of each class. To accomplish this new feature, a new unicast routing protocol must be used so an NS-2[11] implementation of CoSLSP has been produced.

3.1 CoSLSP - Class of Service Link State Protocol

CoSLSP aims to provide a class based unicast routing mechanism. The basic idea is to find one route per class-of-service, able to satisfy the QoS requirements of that class. Apart from the goal of satisfying the QoS requirements of each class, this protocol also addresses the problem of optimizing network utilization. Therefore, instead of computing just the routes that might meet the QoS requirements of each class, CoSLSP tries to find the shortest path that might satisfy those requirements.

It is a unicast link-state protocol that uses a modified Dijkstra algorithm capable of finding the shortest path routes, if they exists at all, that can meet the QoS requirements of different classes of service. In few words: the path calculation algorithm starts by finding the shortest path, whose feasibility is then verified against the QoS requirements. If unfeasible, the next shortest path is then iteratively verified, until a feasible path is found or a configured threshold is reached. In this way, a different route is found for each class of service and it is installed in the routing table. The packet forwarding process has been modified too in order to lookup for the appropriate route depending on the class of service of each packet.

CoSLSP has been implemented and evaluated with Network Simulator. The simulations results show that CoSLSP in case of network congestion is able to find "better" routes in respect to the QoS metrics of each class of service.

3.2 DTMP - Directed Trees Multicast Protocol

DTMP is a multicast routing protocol that implements directed trees construction in opposite to usual reverse path ones. The original idea is based on PIM-SM protocol, a widely deployed multicast routing protocol in the Internet. The PIM-SM, as the majority of multicast routing protocols, builds reverse path trees. This fact may lead to poor routes in the presence of asymmetric networks and problems may arise when trying to implement QoS Routing, as links usually have different characteristics in each direction.

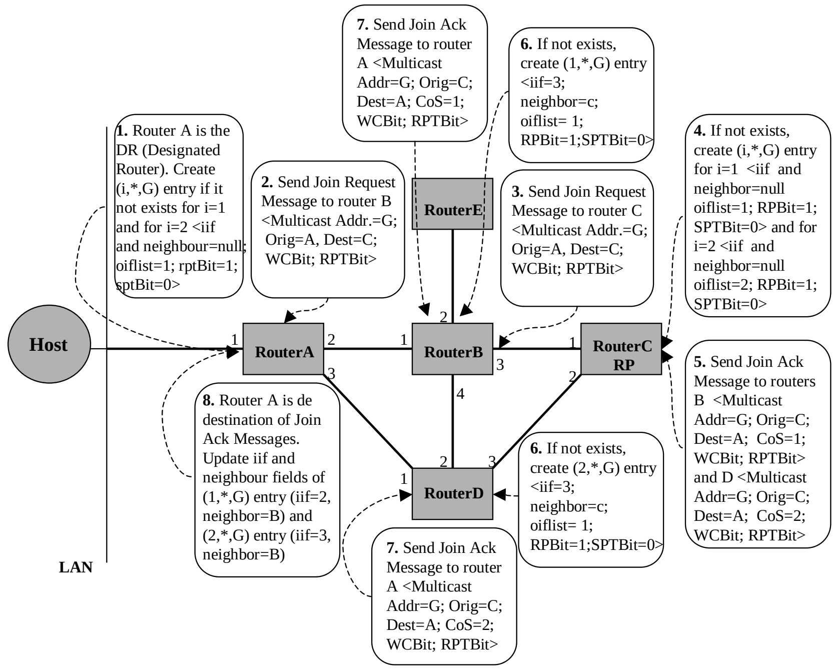
Like PIM-SM, DTMP uses both shared trees and source based trees. Receivers begin joining a shared tree, rooted in a pre-defined point called Rendez-Vous Point. After having received a certain amount of data packets from a source, a receiver may switch to a source based tree. The protocol allows for an easy way of constructing a source based tree, pruning unnecessary tree branches within the shared tree.

3.3 MCMRP - Multi-Class Based Multicast Routing Protocol

We used CoSLSP and extended DTMP in order to implement the multi-class based multicast strategy proposed in section 2.

When a new receiver decides to join, the designated router uses the shared tree join mechanism proposed by DTMP. A join request message is sent towards the RP. The routers along the way between the new receiver and the RP just forward the join request message and no sate information is introduced in these routers. When the RP receives a join request message from a new receiver it must send one join acknowledge message per class of service. These messages must travel towards the new receiver through the best unicast path per each class of service. Those paths are calculated and installed in the unicast routing table by CoSLSP. All join acknowledge messages should be marked in the corresponding class of service in order to follow the best unicast path per each class of service. When a router between the RP and the new receiver receives one acknowledge message it must create or update the corresponding routing table entry in order to create the new tree branch.

The process of joining the shared tree in MCMRP is detailed in Figure 2, where variables and flags have the same meaning as defined in PIM-SM[5]. In the illustrated scenario there are two different classes of service ($CoS = 1$ and $CoS = 2$) and router `A` (the designated router of the new receiver) issues a join request message.



1. Router A is the DR (Designated Router). Create (i,*,G) entry if it not exists for i=1 and for i=2 <iif and neighbour=null; oiflist=1; rptBit=1; sptBit=0>
2. Send Join Request Message to router B <Multicast Addr.=G; Orig=A, Dest=C; WCBit; RPTBit>
3. Send Join Request Message to router C <Multicast Addr.=G; Orig=A, Dest=C; WCBit; RPTBit>
4. If not exists, create (i,*,G) entry for i=1 <iif and neighbor=null oiflist=1; RPBit=1; SPTBit=0> and for i=2 <iif and neighbor=null oiflist=2; RPBit=1; SPTBit=0>
5. Send Join Ack Message to routers B <Multicast Addr=G; Orig=C; Dest=A; CoS=1; WCBit; RPTBit> and D <Multicast Addr=G; Orig=C; Dest=A; CoS=2; WCBit; RPTBit>
6. If not exists, create (1,*,G) entry <iif=3; neighbor=c; oiflist= 1; RPBit=1;SPTBit=0>
6. If not exists, create (2,*,G) entry <iif=3; neighbor=c; oiflist= 1; RPBit=1;SPTBit=0>
7. Send Join Ack Message to router A <Multicast Addr=G; Orig=C; Dest=A; CoS=1; WCBit; RPTBit>
7. Send Join Ack Message to router A <Multicast Addr=G; Orig=C; Dest=A; CoS=2; WCBit; RPTBit>
8. Router A is de destination of Join Ack Messages. Update iif and neighbour fields of (1,*,G) entry (iif=2, neighbor=B) and (2,*,G) entry (iif=3, neighbor=B)
Host
LAN
RouterA
RouterB
RouterC RP
RouterD
RouterE
1
2
3
1
2
3
4
1
2
1
2
3

Fig. 2. *Set Up Shared Trees Implementation. Actions are numbered in the order they occur*

The routing table entries have the same fields as the PIM-SM ones, and an extra one: the upstream neighbor in the tree. This field has been introduced in order to be able to implement the prune mechanism.

The process of commuting to a source based tree is similar to the above described one, with one major difference. When a source receives a join request message, only one join acknowledge message is generated and sent. The join request message is marked in the class of service requested by the receiver. If the source decides to accept this join, a join acknowledge message marked in the same class is generated and sent towards the new receiver following the best unicast routing path for that class. Two different situations may occur. The receiver may decide to switch to a source based tree in the same class used by the source, or it may want to switch to a source based tree requesting a different class of service.

In the first case, when a router in the path between the source and the receiver receives the join acknowledge message, if it is not already in the source based tree it must create a `(i,S,G)` entry and copy the outgoing interfaces list from the `(i,*,G)` entry to the outgoing interfaces list of the `(i,S,G)` new entry. This is because, in the future, packets from source `S` will be forward based on this new entry. Besides, when a router lying between the source and the receiver starts to receive data from that source, it must issue a prune of that source on the shared tree of that class. This prune indicates that packets of the class of service `i` from this source must not be forwarded down this branch of the shared tree, because they are being received by means of the source based tree. This mechanism is implemented by sending a special prune to the upstream neighbor in the shared tree of the class `i`. When a router at the shared tree of the class `i` receives this type of prunes, it creates a special type of entry (an `(i,S,G)RPT-bit` entry) closely like a PIM-SM router. In MCMRP the outgoing interface list of the new `(i,S,G)RPT-bit` entry is copied from the `(i,*,G)` entry and the interface deleted is the one being used to reach the node that had originated the prune, which may not be the arriving interface of the prune packet. This is because in MCMRP there are directed trees not reverse path ones. These `(i,S,G)RPT-bit` entries must be updated too when a join acknowledge message arrives in order to allow the join of a new receiver on a shared tree with source-specific prune state established.

When a receiver decides to join a source requesting a different class of service, the process is a little different. When a new `(i,S,G)` entry is created, the outgoing interface list should not be copied from the `(i,*,G)` entry, because in this case the other receivers connected through the corresponding shared tree want to receive data packets in the source's default class of service. For the same reason these entries should not be updated when a posterior join to shared tree acknowledge message is received. In addition, the "prune of source in the shared three" mechanism must be triggered by the Designated Router when it receives the join acknowledge message. The prune messages must be sent to the shared trees of all classes except to the shared tree of the class for which the receiver commuted. This is because the receiver will start to receive the source's packets through the source tree in the desired class, so it can not continue to receive it by the shared tree of the source's default class of service.

The process of switching from the shared tree to a source based tree in MCMRP is detailed in Figure 3. In the illustrated situation the receiver decides to switch to a source based tree in class of service `1` ($CoS = 1$), supposing the source's default class of service is `2`.

As was referred in the last section the leave group functionality is implemented by means of sending explicit prune requests towards the upstream neighbor in the tree. When the upstream neighbor receives this type of prunes (which are different from the "prune of a source on the shared tree") it must delete the interface used to reach the node that had originated the prune from the out-

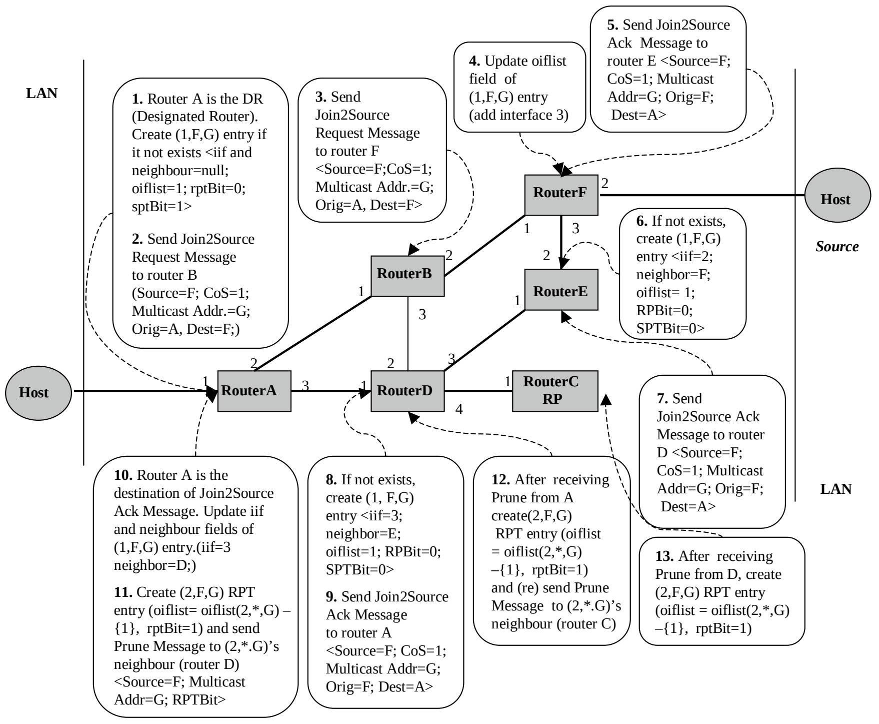

LAN
1. Router A is the DR (Designated Router). Create (1,F,G) entry if it not exists <iif and neighbour=null; oiflist=1; rptBit=0; sptBit=1>
2. Send Join2Source Request Message to router B (Source=F; CoS=1; Multicast Addr.=G; Orig=A, Dest=F;)
3. Send Join2Source Request Message to router F <Source=F;CoS=1; Multicast Addr.=G; Orig=A, Dest=F>
4. Update oiflist field of (1,F,G) entry (add interface 3)
5. Send Join2Source Ack Message to router E <Source=F; CoS=1; Multicast Addr=G; Orig=F; Dest=A>
RouterF
Host
Source
6. If not exists, create (1,F,G) entry <iif=2; neighbor=F; oiflist= 1; RPBit=0; SPTBit=0>
RouterB
RouterE
Host
RouterA
RouterD
RouterC RP
7. Send Join2Source Ack Message to router D <Source=F; CoS=1; Multicast Addr=G; Orig=F; Dest=A>
LAN
10. Router A is the destination of Join2Source Ack Message. Update iif and neighbour fields of (1,F,G) entry.(iif=3 neighbor=D;)
11. Create (2,F,G) RPT entry (oiflist= oiflist(2,*,G) – {1}, rptBit=1) and send Prune Message to (2,*.G)'s neighbour (router D) <Source=F; Multicast Addr=G; RPTBit>
8. If not exists, create (1, F,G) entry <iif=3; neighbor=E; oiflist=1; RPBit=0; SPTBit=0>
9. Send Join2Source Ack Message to router A <Source=F; CoS=1; Multicast Addr=G; Orig=F; Dest=A>
12. After receiving Prune from A create(2,F,G) RPT entry (oiflist = oiflist(2,*,G) –{1}, rptBit=1) and (re) send Prune Message to (2,*.G)'s neighbour (router C)
13. After receiving Prune from D, create (2,F,G) RPT entry (oiflist = oiflist(2,*,G) –{1}, rptBit=1)

Fig. 3. *Switching from Shared Tree to Source Based Tree Implementation*

going interface list of the corresponding (i,*,G) or (i,S,G) entry. This interface may not be the arriving interface of the prune packet. If the outgoing interfaces list become empty the entry may be deleted and the prune should be forward to the upstream neighbor in the tree. Thus, although the construction process of the multicast tree was inverted, from RP or source toward the new receiver, the de-construction process is identical to traditional multicast routing protocols, like PIM-SM. This fact saves a lot of unnecessary control messages.

4 Discussion

A new protocol is presented in this paper, MCMRP - a multicast routing protocol that implements multi-class based multicast routing, to be used in a DiffServ environment.

Because class differentiation is inherently unidirectional, we propose the usage of source and shared directed trees instead of typical reverse path forwarding ones. The heuristic is based upon explicit join acknowledges sent by either source or RP routers in response to explicit join requests sent by receivers. Furthermore the multicast framework presented tends to distribute multicast traffic in a evenly fashion as multiple shared-trees will distribute traffic along different links, instead of concentrating traffic on a smaller number of links (those in the single tree). So, although indirectly, this framework is also useful for network traffic engineering purposes as multicast load balancing becomes manageable.

Class differentiation is mainly achieved by means of usual DiffServ mechanisms *plus* routing differentiation, making it possible to use different routes for different Classes of Service. Generic class characteristics and identifications are directly derived from its DiffServ counterparts. Far from conflicting with traditional inside node differentiation strategies, this proposal is their routing-level complement.

The model introduces acceptable changes to node behavior in a DiffServ environment, but it does not introduce additional complexity to node Per-Hop-Behavior, besides normal complexity for simple multicast support. For instance, routing tables changes made to accommodate class information are acceptable within a class-based environment. End-nodes may still use normal join mechanisms, and in addition they may negotiate, after join request, the desired/possible QoS while the multicast application is running.

MCMRP has been implemented in Network Simulator (NS-2)[11] and is perfectly functional. MCMRP performance is currently being evaluated.

References

1. P. Winter. Steiner problem in networks: A survey. *Networks*, 17:129–167, 1987.
2. P. Berman and V. Ramaiyer. Improved approximations for the Steiner tree problem. In *Proceedings of Third Symposium on Discrete Algorithms, pp 325-334*, 1992.
3. Moses Charikar, Chandra Chekuri, To yat Cheung, Zuo Dai, Ashish Goel, Sudipto, and Ming Li. Approximation Algorithms for Directed Steiner Problems. *Journal of Algorithms*, 33(1):73–91, October 1999.
4. A. Ballardie. Core based trees (CBT version 2) multicast routing. RFC 2189, IETF, September 1997.
5. D. Estrin, D. Farinacci, A. Helmy, D. Thaler, S. Deering, M. Handley, V. Jacobson, C. Liu, P. Sharma, and L. Wei. Protocol independent multicast-sparse mode (PIM-SM): protocol specification. RFC 2362, IETF, June 1998.
6. Shigang Chen, Klara Nahrstedt, and Yuval Shavitt. A qos-aware multicast routing protocol. In *INFOCOM (3)*, pages 1594–1603, 2000.
7. Michalis Faloutsos, Anindo Banerjea, and Rajesh Pankaj. Qosmic: Quality of service sensitive multicast internet protocol. In *SIGCOMM*, pages 144–153, 1998.
8. A. Mankin, Ed., F. Baker, B. Braden, S. Bradner, M. O'Dell, A. Romanow, A. Weinrib, and L. Zhang. Resource ReSerVation protocol (RSVP) – version 1 applicability statement some guidelines on deployment. RFC 2208, IETF, September 1997.
9. S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss. An architecture for differentiated service. RFC 2475, IETF, December 1998.
10. Maria João Nicolau, A. Costa, A. Santos and V. Freitas. Directed Trees in Multicast Routing. In *Quality of Service in Multiservice IP Networks QoSIP2003*, February 2003. LNCS 2601, pp 321-333, Ed. Marco A. Marsan et al, Springer-Verlag, 2003.
11. K. Fall and K. Varadhan. *The NS Manual*, Jan 2001. URL=http://www.isi.edu/nsnam/ns/ns-documentation.html.