Ann. Telecommun. (2008) 63:579–596
DOI 10.1007/s12243-008-0058-x

Class-based multicast routing in interdomain scenarios

Maria João Nicolau · António Costa · Joaquim Macedo · Alexandre Santos

Received: 17 February 2008 / Accepted: 1 July 2008 / Published online: 31 October 2008

Abstract DiffServ-like domains bring new challenges to quality of service (QoS) multicast routing simply by shifting the focus from individual flows into classes of flows. Packets are marked at edge routers and receive differentiated treatment according to the class and not the flow that they belong to. DiffServ therefore became adverse to multicast, as packet replication inside the domain may require classification and remarking functions not present in core nodes. At the interdomain level, no doubt multicast QoS complexity is increased by the interleaving of DiffServ and non-Diffserv domains, making it more difficult to address QoS multicast in an end-to-end perspective. In today's real interconnection world, classes of service have no meaning in certain links of a full interdomain path. While the problem is not new, as already pointed out, there are no real efforts to bring multicast back to a class-of-service domain without compromising its model of operation. In this article, we present an innovative multicast QoS routing strategy, clearly designed for the new class-of-service paradigm. The solution is based upon the construction of multiple trees, one per class of service available, while still allowing receivers to shift for source-specific trees in its own class of service. The strategy is presented in a full end-to-end perspective. Intradomain trees use differentiated routing paths thus helping traffic differentiation. Intradomain receivers are allowed to shift from shared trees into an adequate class-of-service source tree. At interdomain level, each class-of-service interdomain tree branch is accomplished by means of an improved path probing strategy enabling for QoS path establishment. This paper presents this new strategy, and associated protocols, for constructing several multicast and directed distribution trees, one per class of service, within each multicast group. This new strategy and associated protocols are then simulated using NS-2 platform. Simulation results are analyzed and compared with other multicast routing solutions, both at intra- and interdomain levels.

Keywords QoS routing · Multicast · Interdomain

M. J. Nicolau (✉)
Departamento de Sistemas de Informação,
Universidade do Minho, Campus de Azurém,
4800 Guimarães, Portugal
e-mail: joao@dsi.uminho.pt

A. Costa · J. Macedo · A. Santos
Departamento de Informática, Universidade do Minho,
Campus de Gualtar, 4710 Braga, Portugal

A. Costa
e-mail: costa@di.uminho.pt

J. Macedo
e-mail: macedo@di.uminho.pt

A. Santos
e-mail: alex@di.uminho.pt

1 Introduction

Many of the applications in the Internet, such as video conference, distance learning, IP-TV, and video/audio broadcast, would benefit from multicast support from the underlying network. These applications involve multiple users (several receivers and, some of them, several sources too); thus, they need to use network resources efficiently. Multicast communication is the ability to send in an efficient way information to one or more receivers at the same time without incurring into network overloads. Hence, at each router, only a single copy of any incoming multicast packet is sent per active link, rather than sending one copy of the packet per number of receivers accessed via that link.

Routing multicast traffic requires the construction of a distribution tree (or a set of trees). Data packets are delivered through that tree; thus, the major objective of the routing protocol is to build a tree with minimum cost in order to optimize the network resources involved.

In addition, most of the multicast applications are quality-of-service (QoS) sensitive in nature; thus, they will need QoS support too, if available. The new challenge lies on how to build a multicast tree (or a set of multicast trees) to deliver the data from sources to multiple receivers so that QoS requirements are satisfied and the cost of the multicast tree is still minimized.

There are two different models to provide QoS at the network level: IntServ and DiffServ model. The main strength of the IntServ model is to provide an absolute service guarantee. However, it has several weaknesses too. Each router is required to maintain state information for each flow; thus, scalability problems may arise. In addition, each router requires a significant amount of processing overhead, and the connection setup time is sometimes greater than the time required for the transmission of all packets belonging to a specific flow. The goal of DiffServ [4] is to provide the benefits of different levels of QoS while avoiding the limitations of the IntServ model. This is accomplished by aggregating traffic into classes. DiffServ does not maintain any per-flow information and also eliminates the connection setup costs.

Most proposals for differentiated services involve control algorithms for aggregating service levels, packet marking and policing, and preferential treatment of marked packets in the network. The issue of routing for enhancing aggregate QoS has not yet received the necessary attention.

In this paper, an innovative multicast QoS-aware routing strategy is proposed addressing several of the problems faced by interdomain and intradomain multicast routing. This strategy is based upon the class-of-service (CoS) paradigm and is well suitable for DiffServ architectures.

1.1 Multicast routing

The multicast routing problem is to find and establish the most efficient path between a single or multiple sources and multiple receivers. As this problem is similar to the unicast routing problem, the solution is also expected to be closely related.

To build a theoretical definition of the routing problem, a computer network is represented as a graph $G = (V, A)$, where each $v \in V$ is a router and each $a \in A$ is a communication link between two routers. Each graph edge has an associated cost $c \rightarrow R^+$.

For one-to-one (unicast) routing, the problem is the establishment of the lowest-cost path between two arbitrary vertexes $v, u \in V$. For one-to-many or many-to-many (multicast) routing, a communication group $M \subseteq V$ is also defined, and the problem is to find a lowest-cost subgraph $R = (V', A')$ connecting all group members, that is $M \subseteq V' \subseteq V$. The R subgraph must be connected and acyclic, which are the properties of a tree. To use R as a multicast route, m ($m \in M$) originated packets are replicated along all the edges $a \in A'$. Some R vertexes may not be members of M and are only traversed by packets to reach group members.

The optimal solution results from the computation of the lowest-cost $c = \sum_{a \in A'} c_a$ tree $R' = (M', A')$. When all the vertexes of graph G are members of the group, that is $M = M' = V$ (broadcast), the lowest-cost tree is named the *minimal spanning tree (MST)*. There are several efficient algorithms to compute the MST. For the remaining trees, where there are vertexes not included in the multicast group, $M \subset V$, the optimal solution is the *Steiner tree*. The Steiner tree computation is a classical problem of nondeterministic combinatory calculus, focused by research in the last 50 years. As a NP complete problem, suboptimal solutions are welcome, and there are several good Steiner tree approximations. Some of the existing heuristic algorithms [14] use, as a first step, the reduction of a given Steiner problem into a MST one.

1.2 Reverse path multicast

Steiner tree approximation algorithms are not used in real multicast protocols because they are not scalable and most of them require a prior and complete knowledge of the multicast group membership. The relative dimension of network topology $|V|$ and multicast group $|M|$ are key issues on the choice of the multicast routing best approach. If $|M| \sim |V|$, the scenario is the *dense mode*, and if $|M| \ll |V|$, the scenario is the *sparse mode*. In the former mode, there are several algorithms based on *flooding* techniques. In the later one, a strategy based on explicit join requests is usually used, where the multicast tree is built finding the shortest path from the new participant to the nearest node of the multicast tree already built. Flooding-based strategies have scalability problems and they are often improved using reverse path forwarding (RPF) checking. RPF checking consists in the following: when a router receives a multicast packet, it checks if the packet's incoming interface is the one used by the router to reach the packet's source. If it is, the packet is then forwarded through all the interfaces of the router except the packet's incoming interface. Otherwise, the router simply discards the packet. This type of strategy leads to a multicast tree that is actually built in the opposite direction of the one used by traffic. A variant of this algorithm is used by the dense mode variant of protocol-independent multicast (PIM-DM [1]). In the sparse mode version of PIM (PIM-SM [12]), multicast tree branches are established as PIM-join messages propagate towards the rendezvous points (RPs) or the source. The multicast tree is built by explicit join request messages, but it is also a reverse path tree. Assuming that link costs may be asymmetric, the path taken by the join message may not be the shortest path that actual traffic toward the receiver should follow. Thus, the resulting shared or source-based trees may not be optimal.

1.3 Asymmetric networks

The already defined multicast problem assumes a symmetric network, represented by a undirected graph, where all group members, transmitters, and receivers, have an equivalent role. However, real communication networks may be asymmetric due to the communication links used and also, and in most cases due to asymmetric network load conditions that are very relevant when there are QoS requirements, needed by the traffic source nodes and even by the receivers. So, a more realistic model for a communication network is a directed graph. The above solutions based on Steiner trees or MSTs are not usable in such cases.

With asymmetric networks, the used approach is to build a tree assuming a group member (or even an arbitrary node at the center of the network topology) as the root, e.g., the source of the traffic. When the multicast tree is source-based, the complexity of the problem grows (S times, where S is the number of sources) as the multicast routing protocol needs to build a different tree for each potential source of the group. This characteristic is expensive when there are a large number of groups and sources. This is even worse for sparse groups on large network topologies. In such cases, it would be better to build a single multicast tree for each group, shared by all sources, with a root on a central and predefined node on the topology. Nevertheless, to find this central node in the presence of dynamic groups is also a NP-complete problem. The choice of this central node is critical and influences the quality of the multicast tree built.

As the delay penalty for a shared tree in comparison with a source-based tree may reach an average of 1.4, some proposals give the receivers the possibility to switch from an initial shared tree to a source-based tree. In this scenario, a new source starts its transmission using a shared tree, but it may also transmit using a source-based tree after some receiver issues a join to a source based-tree to request it. In the worst case, there are $(S+1)$ trees for each group, one shared tree and S source-based trees.

One possible approach to build directed source-based trees is to use the shortest paths from the root to each group member. The tree that results from this strategy is not the Steiner tree but it is a good approximation. The shortest-paths tree (SPT) is built using a centralized Dijkstra shortest-path algorithm. For this reason, SPTs are frequently used in conjunction with intradomain link state algorithms, as open shortest path first (OSPF) and IS–IS. The computation of multicast routes is a simple extension of the unicast shortest path algorithm. However, this type of strategy does not scale well because it maintains too much state information in each node. For example, multicast OSPF (MOSPF) uses link state advertisements (LSAs) to ensure that all nodes know about the current network topology and link state information. Also, a group membership LSA is used to ensure that all nodes know which are the members of all groups. This amount of information brings complexity to nodes but also brings the possibility to, at every instant, calculate the real SPTs (not reverse-path trees!) that connect any source to all members of a multicast group.

1.4 QoS routing

Routing in the Internet has so far been based on a best-effort service model, primarily concerned with connectivity. Packets are delivered using a route based on destination addresses. Typically a single metric, a cost assigned to each link, is taken into account to make route decisions. Thus, routing protocols build routing tables having the goal of minimizing the cost of each path. However, this model is not adequate to satisfy the growing demands of the applications, most of which demand QoS assurances. In order to support a wide range of QoS requirements, routing protocols need to have a more complex model where the network is characterized with multiple metrics, such as bandwidth, delay, and loss probability. The basic problem of QoS routing is then to find a path able to satisfy multiple constraints.

Most of these new metrics are network-load-dependent values, so even symmetric communication links become asymmetric ones due to the presence of an asymmetric network load. It is assumed that instantaneous values of each metric result from direct monitoring of output links and can be kept updated along the time, as soft state information. Link metrics can be combined to find the corresponding value for the whole path. This is done using composition rules that depend on the correspondent classification: additive (e.g., delay, jitter), multiplicative (e.g., losses), and concave (e.g., minimum bandwidth). Multiplicative metrics can be converted into additive ones using logarithmic transformation. Most of time, some of these path metric compositions are applicable to multicast trees also.

QoS requirements are usually presented as a restriction set. Restrictions can be classified as link, path, or tree restrictions. When a group member is requesting a B bps connection to a group, this means that only links with bandwidth equal to or greater than B may be considered. The same is true for eligible paths and the resultant tree. Path and tree are *feasible* if they can satisfy requested QoS requirements.

Once requests have QoS requirements specified as a set of restrictions, a reformulated version of the multicast routing problem may be introduced. Given a network graph $G = (V, A)$, a multicast group M with a set of tree restrictions C_M and sequence of join/leave requests $E = (e_0, e_1, .., e_n)$, where each event is described by (v, op, C_v), C_v is a set of restrictions for the operation op of the new member v. The objective is the computation of a sequence of multicast trees satisfying C_M and C_v restrictions.

A considerable number of QoS-aware algorithms result from extensions on existing multicast algorithms. Source-based trees usually need more network resources but have potential superior quality in both best effort and QoS scenarios. On the other side, shared bidirectional trees introduce some traffic concentration and some degradation of end-to-end multicast sessions. Even with resource reservation mechanisms, the reserved resources must be shared by all group sources. Core router selection, not considering receiver QoS requirements, also introduces difficulties on the setup of multicast routes. For flood-based algorithms, the multicast flow reaches the potential receivers before any explicit receiver request. In this way, receivers have no chance to explicitly give their QoS requirements for path selection.

The approximate Steiner tree with restriction-proposed solutions can be grouped into two main classes: distributed and centralized algorithms. The last ones assume that a single network node has a complete knowledge about group members. This assumption is not suitable due to inherent dynamism of network load dependent QoS parameters and group membership. In this way, partial information based strategies like path probing are superior to others, at least in large topologies, in which each node must have up-to-date information of the global network state.

2 Interdomain multicast QoS routing

The global scenario for both intra- and interdomain multicast routing is generally a two-tier problem, as it may be depicted in Fig. 1, where different domains may identify different autonomous systems, eventually with very different routing policies.

Most of the various proposals actually presented for multicast routing are, to a smaller or greater extent, somehow related with PIM, no doubt the most commonly used multicast routing protocol. There are two main PIM protocols, PIM-DM [1] and PIM-SM [12]. PIM uses unicast routing information originating from any routing protocol to perform multicast forwarding, instead of exchanging independent multicast routing information between routers. PIM-SM is used in most cases, under the assumption that only a subset of networks within any routing domain will be interested in receiving any multicast group information.

PIM-SM [12] is then a widely deployed multicast routing protocol, especially useful for groups where members are sparsely distributed over the routing domain. It is based upon the concept of RPs, predefined points within the network known by all routers. A router with attached hosts interested in joining a multicast group will start a multicast tree by sending a *join*

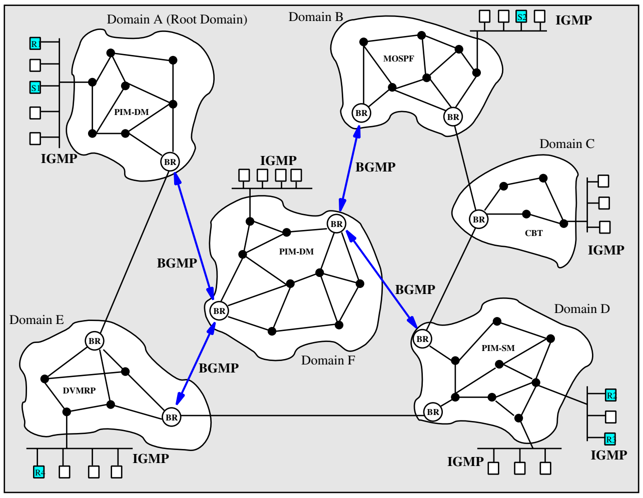


Fig. 1 Hierarchical routing: intra- and interdomain multicast routing protocols

request message on the shortest path to the RP. This join request message is processed by all the routers in between the receiver and the RP so that a new branch for the new member is setup within the multicast tree.

PIM-SM has important advantages when compared to other multicast routing protocols: it does not depend on any specific unicast routing protocol and source rooted trees may be used, instead of the shared tree, if the data rate of a source exceeds a certain threshold. However, PIM-SM assumes symmetric routing paths as it uses reverse-path routing, and thus, it is not suitable for use in conjunction with any kind of constraint imposed by QoS routing.

Indeed, most of the deployed multicast routing protocols, like core-based trees (CBT) [3], and PIM-SM are based upon reverse-path routing. Only multicast extensions to OSPF, MOSFP [18, 19], handling the topological database as a directed graph, deal with asymmetric networks topologies. MOSPF has seen implementations and has been deployed in intradomain networks but cannot cope, because it does not scale, with interdomain.

There are few proposals for interdomain constrained multicast routing. Among those are border gateway multicast protocol (BGMP) [15, 22], yet another multicast routing protocol (YAM) [6], QoS-sensitive multicast Internet protocol (QoSMIC) [11], and QoS-aware multicast routing protocol (QRMP) [8].

BGMP is an interdomain multicast routing protocol able to cope with several of the scaling problems that other multicast protocols exhibit. BGMP has several features that, like its unicast counterpart, the border gateway protocol (BGP) [21], make it suitable for ISPs usage. BGMP has been designed in order to support both unidirectional source and shared trees, as well as bidirectional shared trees for multicast data distribution. Each type of tree is specially useful for certain types of applications (e.g., unidirectional trees for single-source and also for backward compatibility; shared trees for many-to-many distribution). BGMP is able to build shared trees that are rooted at the autonomous system where the multicast group address originates, using several different mechanisms in order to *discover* which autonomous system "owns" the multicast group address being distributed.

So, BGMP nicely supports "source-specific multicast" and, by means of shared trees, is also able to support "any-source multicast." Nevertheless, as a major drawback, BGMP has not taken any QoS metric into account, although some extensions have later been proposed. YAM, proposed by Carlberg and Crowcrof, builds shared trees with the capability to provide multiple routes to connect a new node onto an existing tree.

YAM establishes on-demand shared trees, handling dynamic membership, making use of the discovery feature associated with the one-to-many join mechanisms, as depicted in Fig. 2. Nevertheless, YAM does not require any global network state at routers, although it implies excessive communication overhead because it relies on flooding to find a feasible tree branch to connect a new member.

YAM itself does not take into account any specific QoS metric but it is able to operate over alternate paths. Several extensions to YAM have been proposed [7, 11], able to cope with multicast QoS requirements.

QoSMIC, proposed by Michalis Faloutsos et al., alleviates the flooding behavior but introduces a new complex element: the manager router. QoSMIC uses two different procedures to find a feasible tree: a local search and a multicast tree search. Local search, as shown in Fig. 2, is initiated by the new member router by flooding BID-REQ messages to its neighborhood, with scope controlled by time to live. Any in-tree router that receives a BID-REQ message becomes a

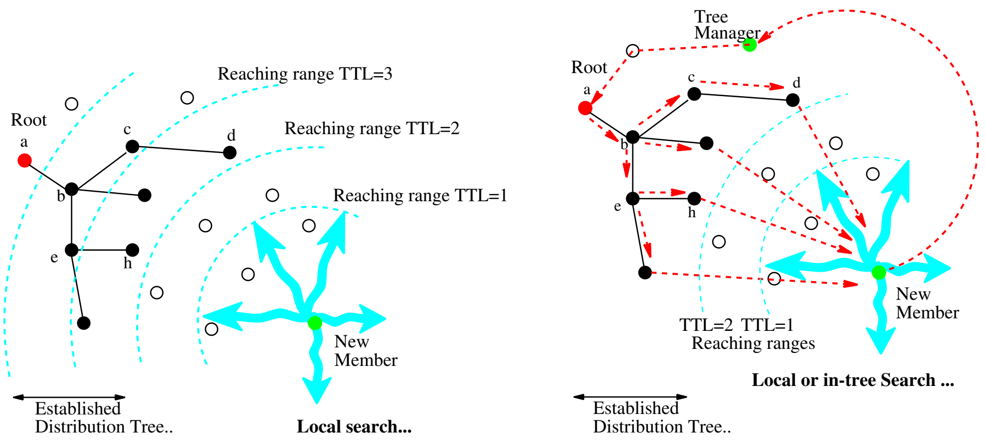


Fig. 2 Search policies for new member: YAM (*left*) vs QoSMIC (*right*)

candidate router and replies with a BID message, which is unicast to the new router. The BID message collects information about the path on its way that can be used for selection purposes. The multicast tree search occurs at the same time, initiated by a manager after receiving a M-JOIN request from the new receiver. The manager sends a BID-ORDER message to a set of in-tree routers, that become candidate routers and reply with BID messages exactly as described for the local search procedure.

In QRMP [8], proposed by Shigang Chen et al. in 2000, two search modes are defined: single-path mode and multiple-path mode. The routing process starts with the single-path mode, attempting to search only the unicast routing path travelled by the *join request message* through the multicast tree. The *join request message* carries the QoS requirements. As it travels, it checks the resource availability of every intermediate node and proceeds only when the node has the required resources. If an intermediate node does not have the required resources, it triggers the multiple path mode by sending a *not acknowledge message* to the previous node. Upon receipt of the *not acknowledge message*, the previous node sends the *join request message* to all neighbor nodes except those from which the *join request messages* and *not acknowledge messages* were previously received. Once a feasible branch is detected, an *acknowledge message* is sent back along the branch that triggers the multiple path mode. If more then one acknowledge message arrives at this node, the node will select the best branch and reject all the others. In QRMP, tree construction occurs from the new receiver in the direction toward the tree instead of from the first in-tree node found backwards toward the receiver; therefore, it does not seam adequate for asymmetric topologies. In multiple-path mode, the *join request messages* are flooded. Besides, it does not support the establishment of multicast routing policies.

3 CoS multicast routing: a new approach

There are two different approaches in order to provide QoS to routing processes: per flow and per class routing. Per flow routing strategies are based on the principle that QoS routes must be computed for each request, being that requests explicitly express their resource requirements, resorting to resource reservation in order to maintain those requirements after a feasible path has been found. This type of strategy can be easily adapted to the multicast scenario. In a multicast scenario, the path searching process is usually initiated by the new receiver, which explores different alternative paths and evaluates them in terms of how well they fulfil requirements. When a feasible path is found, a new multicast tree branch is built joining the receiver to the multicast tree. This type of strategy is usually adopted by interdomain multicast QoS routing strategies, like YAM and QoSMIC, because it does not require the nodes to keep global state information. There is no need to keep any type of link state information since QoS path metrics are evaluated by each node during the setup phase. Nevertheless, after a feasible path is found, per-flow information is kept in those in-path nodes. As the number of simultaneous per node flows may grow indefinitely, resource consumption becomes a critical issue.

The alternative approach is per-class routing. Instead of trying to deal with each specific flow individually, the idea is to group them into a small amount of predefined classes. Packets on each flow are first marked into one of the available classes and receive, thence, a class-specific treatment in all forwarding tasks. By changing the focus from flows to classes of flows, this approach really introduces a big paradigm shift. Flows are aggregated by affinity in terms of QoS requirements, thus making per-flow strong guaranties difficult to achieve. However, since the number of classes is both small and well known, routes can be precomputed per class instead of computed on demand. Since routes can differ from class to class, there is an enormous potential for traffic engineering and class differentiation by means of routing differentiation. This approach has been proposed for unicast but not yet for multicast, or at least not with the same conviction. However, this strategy can easily be used in multicast, with the same benefits and the same withdraws as in unicast.

In a DiffServ multicast scenario with heterogeneous group members, each one demanding a different CoS treatment, multiple multicast trees should be built (instead of a route per CoS like in unicast routing), at least one tree per CoS, in order to comply with different per-class QoS requirements. Thus, each router will have to deal with more state information than in traditional multicast routing; however, the total number of different classes will be much smaller than the total number of members in a multicast group. So, a real improvement is expected.

Another problem that must be solved in order to implement QoS multicast routing is the way multicast distribution trees are built. Most of the deployed multicast routing protocols, such as DVRMP [23], CBT [3], and PIM-SM, are based on reverse path routing. Only MOSFP [18] handles asymmetric network topologies since the topological database in MOSFP is stored as a directed graph. In PIM-SM, the packet deliver path is set-up as PIM-join messages propagate towards the RP or the source. Therefore, the multicast distribution tree actually built by PIM-SM protocol is in fact a multicast reverse path tree, as it is built in the opposite direction to the one used by multicast traffic. This is an important problem to address when dealing with QoS routing because these routing constraints expose link asymmetry in terms of the QoS they offer. Due to these asymmetries, the path taken by the *join request message* may not be the shortest path that data traffic should follow. Thus, the resulting trees, shared or source-based, may not be optimal. In order to address the multicast routing problem, aware of DiffServ context, a new strategy is therefore proposed: a strategy based on establishing directed multiple CoS multicast trees.

3.1 Directed shared tree construction

In order to give receivers the ability to join a group without knowing a priori who and where the sources are, a shared tree is established to begin with.

To build a directed shared tree, explicit *join request messages* issued by new receivers must be sent towards the RPs router. When the RP router receives a *join request message*, it must send back an acknowledgement packet. This acknowledgement packet is sent back to the receiver along the shortest path between the RP router and the new receiver; this path may differ from the one followed by the *join request message*.

Routers along this path, when receiving such an acknowledgement packet, may then update their routing tables in order to build new multicast tree branches. Updating is basically accomplished by registering with the multicast routing entry for that tree, both the incoming and outgoing router interfaces traversed by acknowledgement packets.

When CoS multicast routing is considered, a shared tree per CoS is needed (instead of a single shared multicast tree) in order to enable sources to start sending data within any class. It is assumed that the total number of "available" classes of service has a preestablished upper limit and is small, when compared to the number of group members. Figure 3a, b illustrates a receiver joining directed shared trees.

Data packets issued by sources, previously marked according to source-defined QoS parameters, are sent towards the RP router. This RP router forwards data packets, based on the CoS they are marked with, using one of the shared trees.

When a new receiver decides to join, the designated router sends an explicit join request towards the RP router. Routers along the way between the new receiver and the RP just forward the *join request message* and no state information is kept. When the RP receives a *join request message* from a new receiver, it must send a *join acknowledge message* per CoS. These messages must travel towards the new receiver through the best available unicast path per CoS, building new branches in each CoS multicast tree (see Fig. 3b). When joining a group, receivers may initially connect to any of the RP shared trees.

3.2 Heterogeneous QoS receivers: switching from shared to source-based trees

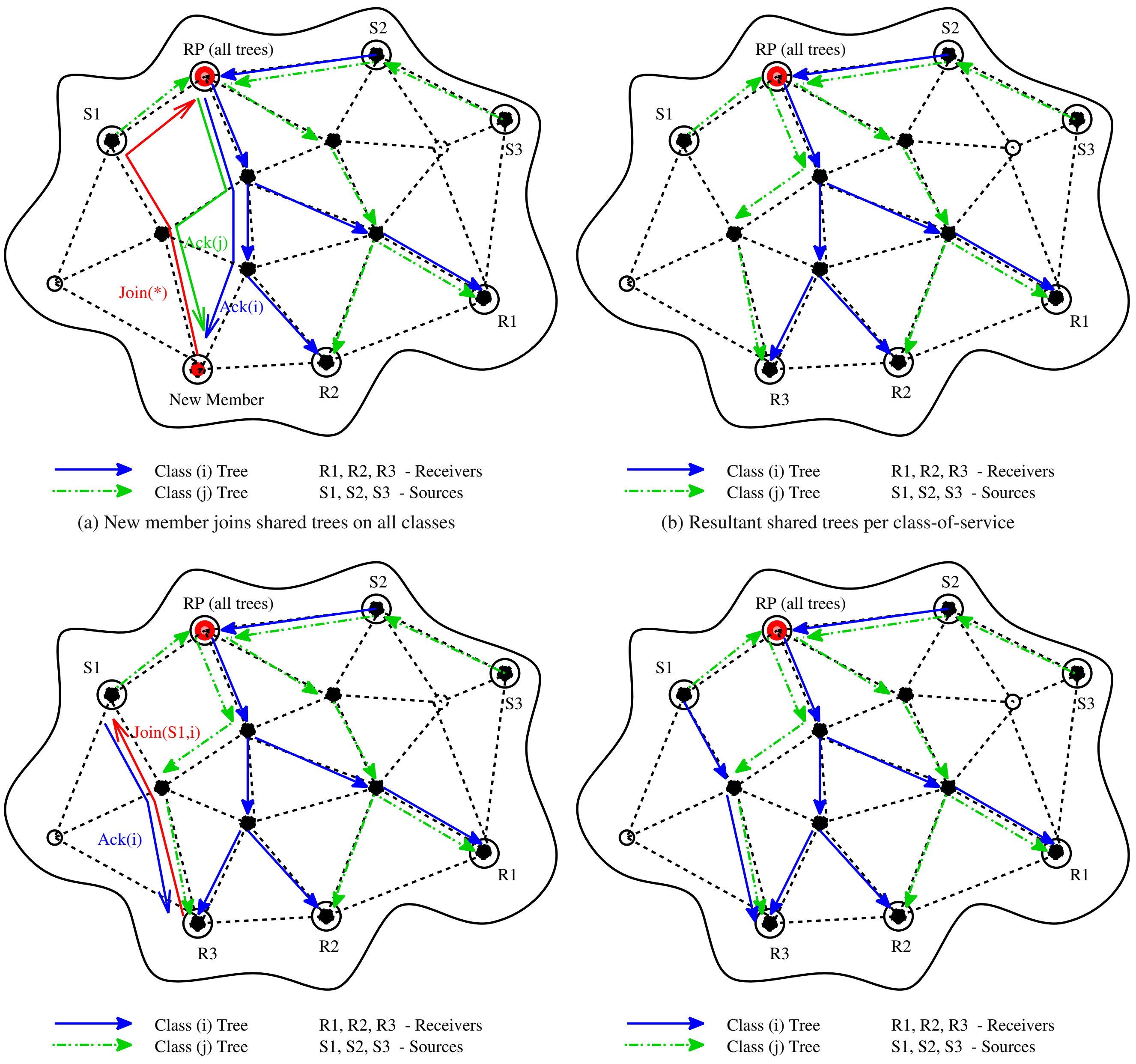
The multiple RP shared tree mechanism, presented so far, does not really allow receivers to specify their own QoS requirements. Traffic flows from sources to receivers using one of the shared trees, according to the QoS parameters defined by sources. After a starting period, a receiver may demand for a reclassification of 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 specify the desired CoS and include it in the *join request message*. 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 CoS must be generated.

When accepting a join for a new CoS, 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, being that now a single *join acknowledge message* per join request is generated. Two different situations may still occur: the receiver decides to switch to a source lying in its own domain or it may want to switch to a source in a different administrative domain.

3.2.1 Join a source-based tree in the same administrative domain

Even in this specific situation, two different types of source-based tree join may occur: the receiver decides to switch to a source-based tree in the same CoS or it may decide both to switch to a source-based tree and request a different CoS.

The first case is similar to the switch, from shared tree to source-based tree, issued by a receiver in PIM-SM, with some changes due to the directed nature of multicast trees. In PIM-SM, when a receiver decides to switch from a shared to a source-based tree, it sends a *join request message* to the source. All routers in the path between the receiver and the source are responsible by building the new tree branch in the source-based tree. Besides, when a router lying between the source



(a) New member joins shared trees on all classes

(b) Resultant shared trees per class-of-service

(c) New member requests a specific class to a source

(d) Resultant set of shared and source-based trees

Fig. 3 Intradomain multiple tree construction (**a**–**d**)

and the receiver starts receiving data from that source, it must issue a prune of that source on the shared tree because now packets must be received via the source-based tree and not from the shared one. When building directed trees, the same idea is to be applied, with some minor changes. Here, the new source based tree branch is built by a *join acknowledge message* instead of the *join request message*.

In [20], the source-based tree *join request message* sent by the receiver is just forwarded towards the source by all the routers along the path. When the source receives the *join request message*, it sends back a *join acknowledge message* towards the receiver and this message will induce the construction of the new tree branch. Each router in the path between the source and the receiver will process and forward the *join acknowledge message*, updating its routing tables. The interface to be added in the corresponding outgoing interface list is the one that has been used to forward the *join acknowledge message*. When one of the routers belonging to the new multicast tree branch begins receiving traffic from the source, it must issue a corresponding prune on the shared tree of that class. This prune indicates that packets from this source, on CoS `i`, must not be forwarded using 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 class `i` shared tree. When a router at the shared tree of the class `i` receives this type of prune, it creates a special type of entry [an `(i,S,G)RPT-bit` entry], where the outgoing interface list of the new `(i,S,G)RPT-bit` entry results from a copy of the `(i,*,G)` entry; the interface to be deleted is the one being used to reach the node that issued the prune (may not be the arriving interface of the prune packet). This is because we are dealing with directed trees, not reverse-path ones. These `(i,S,G)RPT-bit` entries must be updated whenever a *join acknowledge message* arrives in order to enable any new receiver to join a shared tree (even if it had source-specific prune state established).

When a receiver decides to join a source, requesting a different CoS, the process develops as depicted in Fig. 3c, d. 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 still need to receive data packets in the source's default CoS. 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 tree" mechanism must be triggered by the receiver 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 recently commuted. This is because the receiver will start to receive the source's packets via the source tree in the desired class, so it should not keep on receiving them via the shared tree established for the source's default CoS. Figure 3 summarizes the process of multiple tree construction in the intradomain.

3.3 Connecting administrative domains

Despite the huge number of QoS multicast routing proposals that have emerged in the past few years, very few were actually designed for the dual-level hierarchical model of the current Internet. This is also true for non-QoS-aware protocols.

One of the most common difficulties, absent in intradomain routing, is the need to address policy issues. Domains are delimited by administrative boundaries and domain border routers must be configured to enforce domain policies. This is usually carried out by filtering routing information that enters or leaves a domain. BGP long life lasting derives mostly from its policy friendliness. Because BGP routes are qualified with a full AS-Path attribute, it is not hard to write complex filtering patterns without the risk of inducing loops. This issue really changes the focus of interdomain routing: from efficiency to political connectivity. With such a focus, QoS usually turns out to be a secondary issue.

Furthermore, in order to preserve domain independence, it is also desirable that independent domains do not have to rely on external resources managed by others. This simple nontechnical issue frustrates the construction of global shared trees rooted at single external RP nodes.

Another specific interdomain problem is the accuracy of the QoS information collected. Currently, a route update can take several minutes to spread across multiple domains. Absolute measures of QoS metrics like bandwidth, losses, or delays are useless in such a time interval. Efforts are therefore directed to provide statistical values, such as means and deviations, which are more valuable in near future predictions [16]. The engineer task turns out to be selecting the right amount of information to include in summary metrics without severely degrading its accuracy.

All other issues are common to both intra- and interdomain levels but are eventually more stressed by scale and policies. One example is the asymmetric nature of routing. As pointed out in the intradomain section, routes are asymmetric when dynamic QoS metrics are considered. Resource consumption depends upon flows admitted; for most multicast applications, flows show intense packet rates from sources to destinations and almost no traffic in the reverse direction. A good path in one direction may not be a feasible one for the reverse direction. Interdomain specific issues, like policies, enforce this asymmetric nature.

3.4 CoS and non-CoS domains coexistance

The analysis of interdomain specific requirements leads to a fundamental question: can we simply expand the intradomain strategy presented so far or is it inadequate according to those requirements?

Preserving domain independence implies allowing each domain to decide whether to use CoS or not. It implies also to allow domains to not depend on external RPs, as already pointed out. This compromises the goal of multiple shared trees at the interdomain level, but not its usage inside the domain. RPs within each domain can establish peer relations between them, as in MSDP [13] or [9], in order to exchange information about their active sources. Some announcements will carry CoS, while others will not.

Figure 4 illustrates this relationship with non-CoS domains. A CoS domain internally builds multiple shared trees, a tree for each class, and announces all its sources to external peers with the associated CoS. Classes are supposed to be standard and well known by all domains, without invalidating the freedom to use them or not. It is also assumed that it is always possible to map a set of QoS metrics obtained for a flow, a link, or a path into a specific CoS. A mapping function, as well as its reverse function, are both needed in order to avoid class mismatches.

Domains having no members on a specific group will just resend the announcement to all peers except the originator. Non-CoS domains having active receivers inside will probably proceed with a join to that source, as usual, ignoring any class mark Fig. 5a. CoS domains should consider all external unmarked announcements as belonging to a default class.

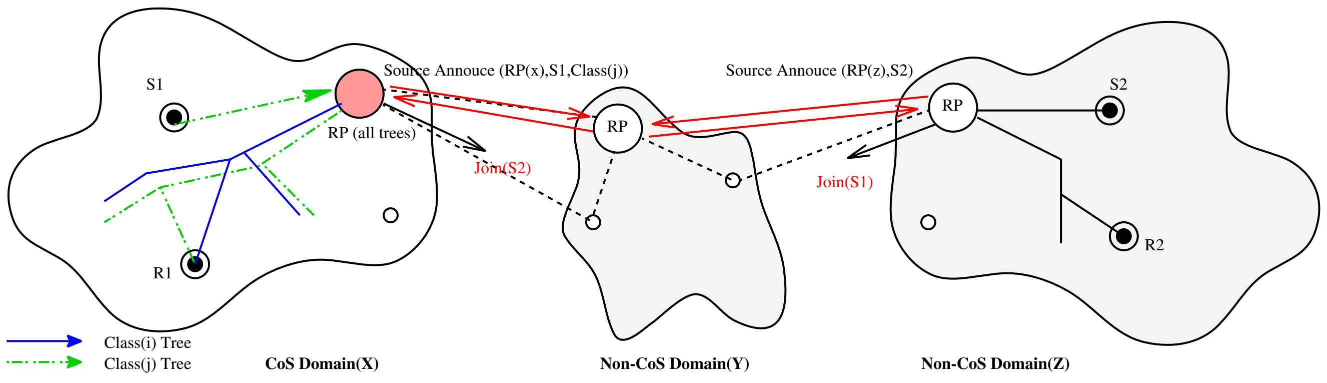
Figure 4b illustrates the way two CoS domains could interact. Each domain announces its sources' availability with the correspondent CoS. In order to provide traffic to its domain receivers, RPs join the interdomain tree for the CoS the source announces. An internal receiver, within any domain, may shift to a source-specific tree branch, on the same or on any other CoS, by sending a *join request message* on the desired CoS. This is also illustrated in Fig. 4b.

3.5 Interdomain via path probbing: effort vs. efficacy trade-off

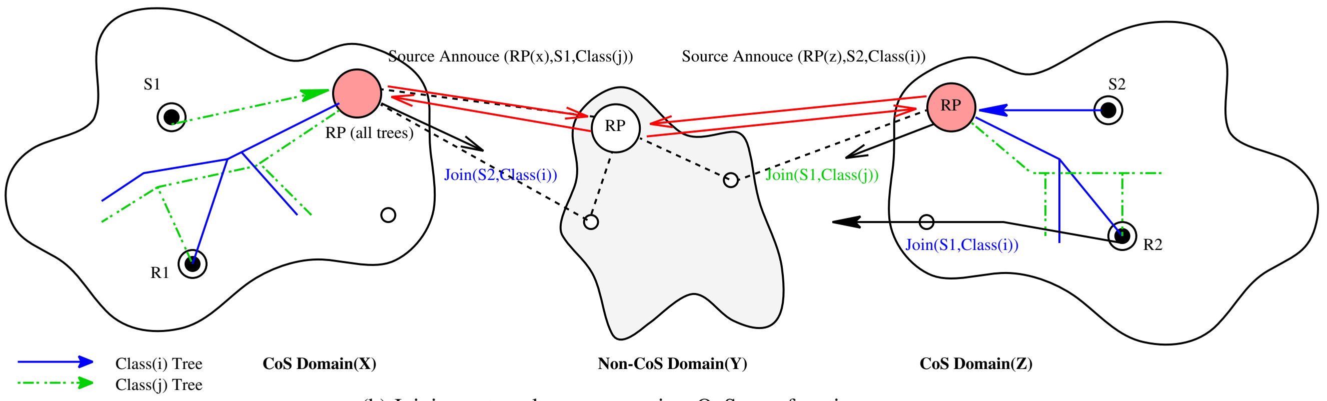
The next question is how to build the interdomain QoS-aware multicast tree. In intradomain, precomputed unicast paths can be used if they exist on unicast routing tables. A CoS unicast routing protocol may be used in order to find the best unicast path per CoS. However, applying this type of strategy to the interdomain context introduces some scalability problems related to state information accuracy and memory consumption on routers.

The alternative is to build tree branches on demand, probing the QoS of the available paths. Special care must be taken in order to build directed trees instead of reversed path ones because asymmetries are highly stressed at the interdomain level.

Probing techniques pointed out so far differ basically in the spread of network regions covered by the probes



(a) Source announcements exchanged with non-CoS domains



(b) Joining external sources requires QoS map functions

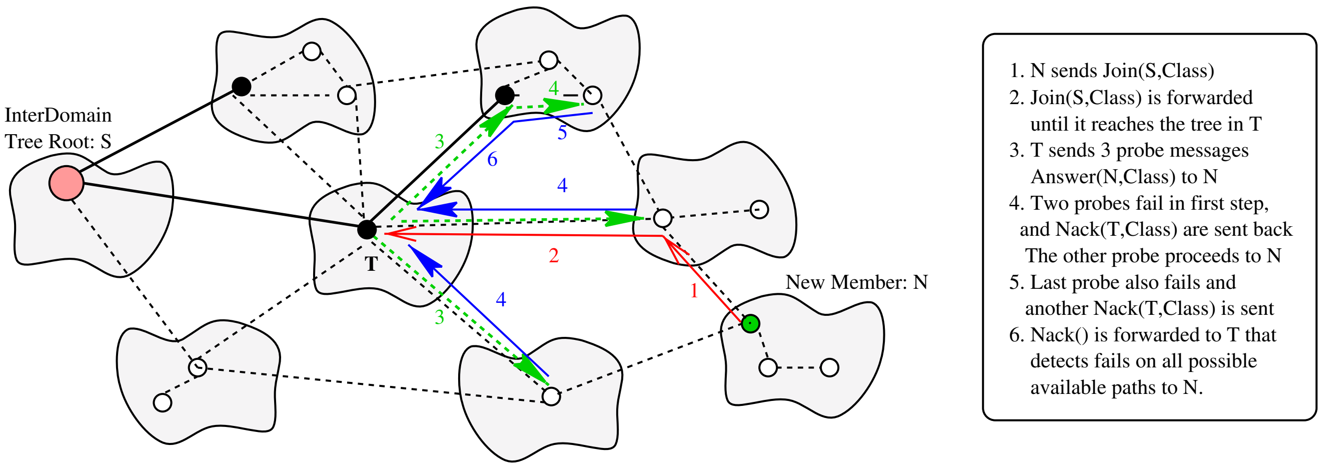
Fig. 4 Interconnecting administrative domains (**a**, **b**)

and in the direction they follow. Larger areas probed means greater efficacy but also a greater control overhead. There is a clear trade-off between the number of probes launched and the number of feasible paths that can be found. Techniques like YAM perform well when the new member is in the neighborhood of an existing tree. This is, of course, unlikely to occur in large networks. This technique can be turned up by enlarging the size of the expansion rings used, but the overhead grows exponentially. Other approaches, such as QoSMIC, try first to locate the multicast tree and then select a subset of in-tree nodes to launch probes. This technique can be turned up on the in-tree node selection procedure. The best results are, however, achieved if all in-tree nodes launch probe messages, but that introduces huge overheads. Perhaps the best approach, in this effort vs efficacy trade-off, is the one followed by QRMP: first follow the usual path towards the tree and measure it; launch probes in multiple directions only if the obvious path is not feasible. This technique can also be tuned up by controlling the number of times a probe can be forked. It has, however, a major drawback: the direction of the probing process. Paths are probed from new members to tree roots, and they should be issued in the reverse direction, due to asymmetries.

The path probing strategy proposed here is based on these key ideas: find the tree as fast as possible, launch probes in the downstream direction, and increase probing efforts only when paths cannot be found. The strategy description that follows will show that these options lead to the best fit to the interdomain level.

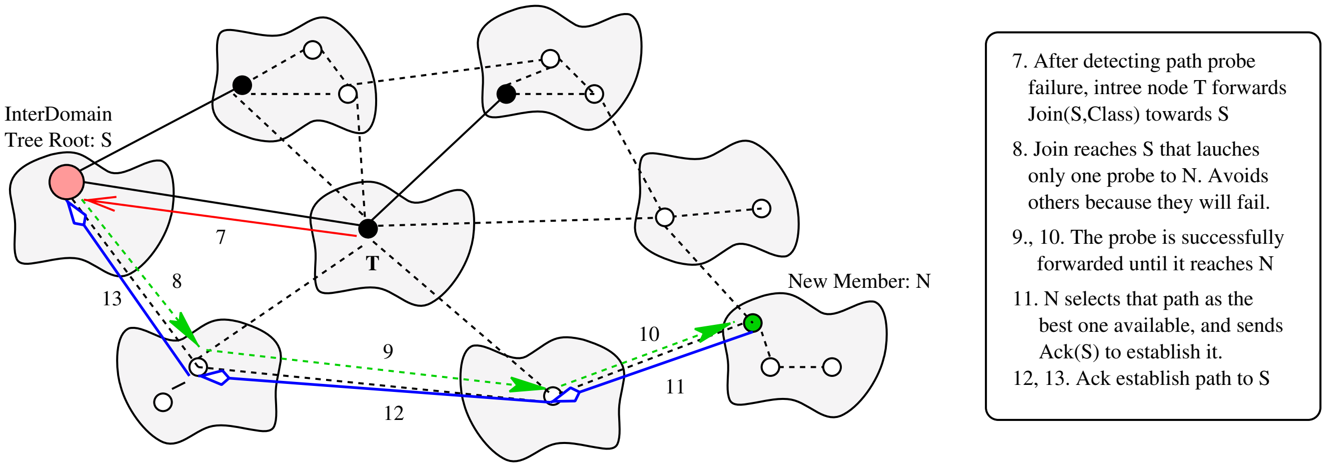
3.6 Interdomain tree branches construction

Figure 5 shows the way interdomain trees are constructed. Multiple domains are connected by border routers that exchange MBGP routes. Policies are applied first by filtering MBGP announcements, sent or



1. N sends Join(S,Class)
2. Join(S,Class) is forwarded until it reaches the tree in T
3. T sends 3 probe messages Answer(N,Class) to N
4. Two probes fail in first step, and Nack(T,Class) are sent back The other probe proceeds to N
5. Last probe also fails and another Nack(T,Class) is sent
6. Nack() is forwarded to T that detects fails on all possible available paths to N.

(a) First in-tree node found conducts probing process



7. After detecting path probe failure, intree node T forwards Join(S,Class) towards S
8. Join reaches S that lauches only one probe to N. Avoids others because they will fail.
9., 10. The probe is successfully forwarded until it reaches N
11. N selects that path as the best one available, and sends Ack(S) to establish it.
12, 13. Ack establish path to S

(b) Limited retries are issued further inside the tree

Fig. 5 Interdomain tree built by path probing strategy (**a**, **b**)

received. So, only policy compliant routes remain to be used on multicast tree construction.

An interdomain tree is needed only when a receiver (or a RP-node on the receiver's behalf) decides to join an external source. The request to join is generated either within the original class, extracted from the source announcement received, or in a different one, if explicitly expressed by receivers.

The *join request message* is forwarded between border routers, without creating state information, until it reaches the first node already in the source tree for that CoS. Note that the request is not forwarded up to the tree root, as it was inside the domain, in order to avoid overloading it. Tree branches are constructed based on the QoS values maintained on each tree node and loops are avoided by analyzing the AS-path value of the branch. After having received the *join request message*, the newly found in-tree node sends a probe message on every possible path, directed to the requesting domain. That is the right way to cope with asymmetries.

Each node that receives a probing message must update its QoS path value by using the measured QoS on the next link in the path leading to the joining node. If the computed cumulate QoS value can no longer be mapped into the requested CoS, or if the node is already in the tree, the probe procedure fails. In this case, a negative ack is sent back to the in-tree node that is controlling the probe process.

Eventually, one or more probes do reach the new member and a path is selected. However, if all attempts fail, as illustrated in Fig. 5a, a new retry must be initiated, further inside the tree, by resending the original join request towards the tree root. The total number of retries is limited to keep the strategy scalable. The next node that receives the request will then conduct its own probing, using exactly the same procedure. As soon as the new member receives one valid probe message, it establishes the new tree branch by sending an ack on this specific path.

4 Experiments in multiclass multicast routing

4.1 Experimental setup

NS [10] has been used in order to simulate and evaluate our multicast routing proposal. First, directed trees construction inside a single domain has been implemented and results have been compared with a PIM-SM implementation. In these simulations, the CoS link state protocol (CoSLSP) has been used. CoSLSP is a CoS unicast routing protocol based on LS (a link-state unicast routing protocol implementation included in NS-2 distribution).

CoSLSP aims to provide a class-based unicast routing mechanism. The basic idea is to find one route per CoS, able to satisfy the QoS requirements of that class. It is a unicast link-state protocol that uses a modified Dijkstra algorithm capable of finding the shortest path routes, if they exist at all, that can meet the QoS requirements of different classes of service. In a 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 CoS and it is installed in the unicast routing table. The packet forwarding process has been modified too in order to lookup for the appropriate route depending on the CoS of each packet. These CoS paths in unicast routing tables are used by *join acknowledge messages*, which travelled from RP and sources to receivers in order to build CoS-directed multicast trees. This was just a simple way to achieve per-CoS routes in unicast routing tables.

4.2 Used metrics

To evaluate the proposal, different metrics have been used. First of all, it is needed to measure the quality of the multicast trees built. The best way to achieve this, since there are multiple trees, is to count the number of data packet replicas that are originated by nodes while forwarding those packets across the distribution tree. In order to realize how well the tree construction mechanism deals with link asymmetries, instead of using only the number of data packet replicas, a metric combining this number with the cost associated to each link traversed by each packet replica is used.

The second thing we intend to evaluate is the gain of using CoS multicast routing. In our simulation experiments, CoS requirements were defined in terms of packet losses. Therefore, to measure the gain of our multiclass multicast routing strategy, we used the average packet drops that occurred in the flows of each CoS (Fig. 6).

4.3 Results obtained

The topology used in the first simulation scenario (a single DiffServ domain) is a typical large ISP network [2]. This topology (Fig. 6) includes 36 nodes; 18 of them are core nodes, and the other 18 are edge nodes. Each of the core nodes is connected with one edge node

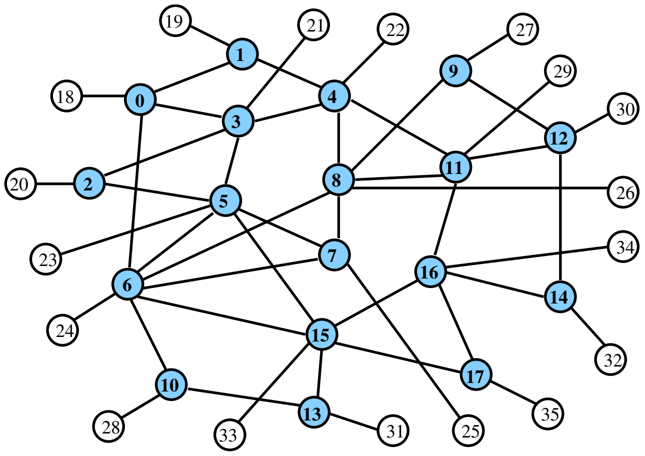


Fig. 6 Typical large ISP topology

by a symmetric link with the cost 1. The core nodes are interconnected with each other by 30 asymmetric links. There are different alternative paths with different costs between any pair of core nodes. Link costs are integers randomly chosen from the interval [1, 12].

Three different classes of service with different QoS requirements in terms of losses were considered. Class 1 does not support any losses; class 2 supports well losses; and, finally, class 3 can deal with greater losses. Each link has three physical queues (one per class) and two virtual queues corresponding to two different drop precedence. All queues are configured exactly in the same way in order to prevent inside node differentiation. Therefore, the only class differentiation that can occur is caused by the action of the routing protocol.

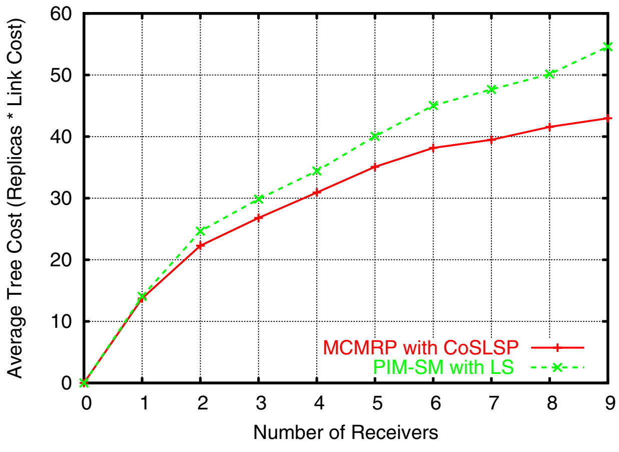
For each simulation run, only one group is considered, and the RP is randomly chosen within the set of all nodes. There are four fixed sources, and each source generates traffic in a CoS randomly chosen. It is assumed that a single receiver is connected to each edge node in the topology and that all edge nodes have one potential receiver attached.

At the beginning of the simulation, there are no receivers joining the group. After an initial period, nine receivers start to join the group building three shared trees rooted at RP. After all the receivers have joined, eight randomly chosen receivers join the four different sources requesting a CoS that is also randomly chosen. This scenario is then kept until the end of the simulation. Before the simulation ends, all the receivers leave the group.

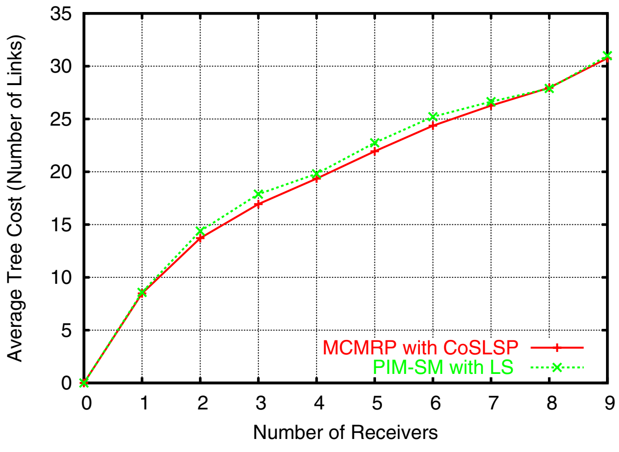
Simulation results are presented in Figs. 7 and 8, where shown values reflect the computed average after 100 independent simulations.

Figure 7 shows the characteristics of the trees built with the two strategies (MCMRP and PIM-SM). The curves presented in Fig. 7a show the average tree cost in function of number of receivers. The tree cost is measured in terms of `number of replicas` *times* `link cost`. The curves presented in Fig. 7b show the total number of links in the topology that are involved in the multicast trees as a function of the number of join or leave operations.

The results shown in Fig. 7a bring to evidence that CoS trees have costs smaller than those created by PIM-SM. This is because they are directed trees instead of reverse-path trees. Note that CoSLSP, the underlying unicast routing protocol, does not choose the best unicast routing path; it chooses the best unicast path that, furthermore, can also meet the QoS requirements of each CoS. Even with this characteristic, it is able to



(a) *Tree Cost(number of replicas* `times` *the link cost)*



(b) *Number of Links in the multicast trees*

Fig. 7 Multicast trees quality (**a**, **b**)

build better trees than PIM-SM. In addition, observing Fig. 7b, we conclude that MCMRP is able to build *better* trees than PIM-SM without enlarging their size.

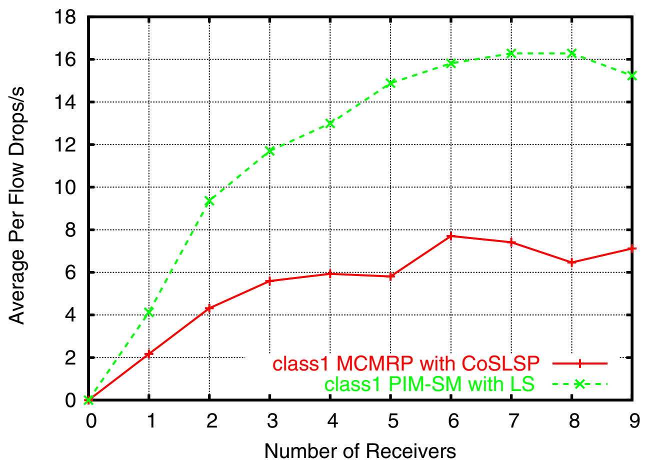
Figure 8 shows the average packet drops suffered in function of number of receivers. Figure 8a, b, and c show the average packet drops that occurred in the flows of each CoS when using the two protocols (MCMRP and PIM-SM). Figure 8d shows the results obtained for all the three classes, in terms of packet drops per flow, when using MCMRP.

These results demonstrate that when MCMRP is used a considerably lower amount of drops is verified. This is because MCMRP try to find routes less congested when links became bottlenecks. In addition, results show that MCMRP routing strategy promotes the expected differentiation between classes. Observing Fig. 8d, we conclude that the average number of packet drops suffered by class 3 is greater than the average number of packet drops suffered by class 2 and the average number of packet drops suffered by class 2 is greater than the average number of packet drops suffered by class 1. This is because class 1 has the highest QoS requirements, followed by class 2, and finally, class 3 is the least demanding one.

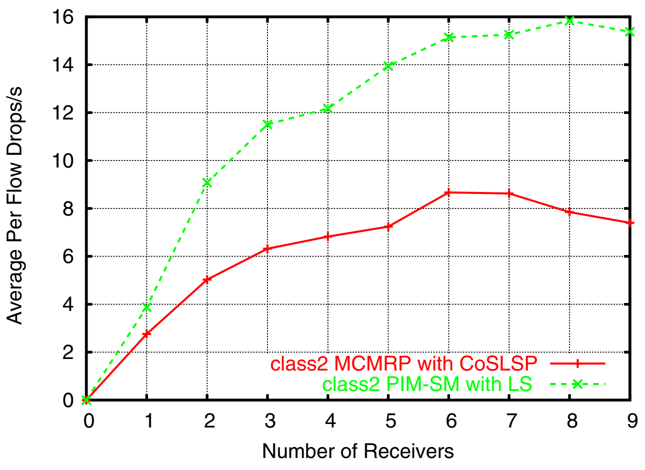
5 Experiments in interdomain via path probing

5.1 Experimental setup

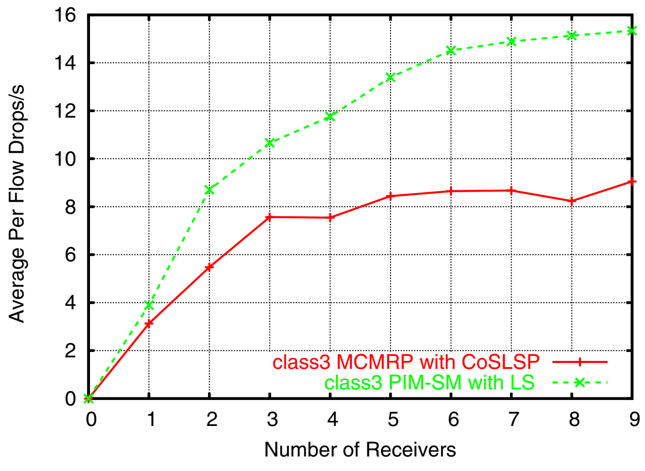
NS [10] was also used to simulate and evaluate our proposal at the interdomain level. Interdomain tree branches are created only when a new member joins a source-based tree rooted at another administrative domain. When that happens, a path probing procedure is initiated. In this set of simulations, two other protocols



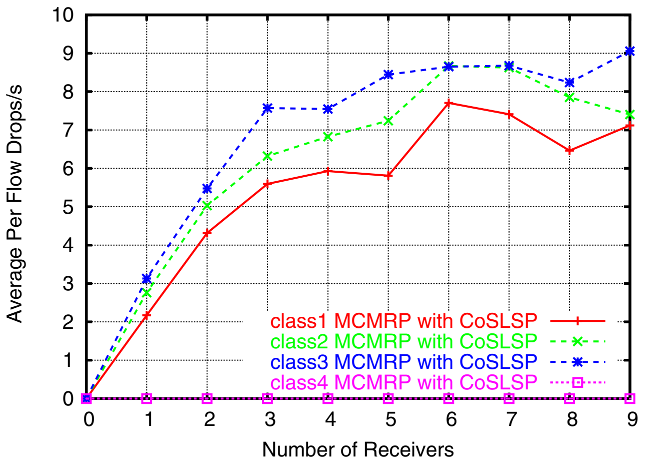
(a) *Flow Drops-Class 1*



(b) *Flow Drops - Class 2*



(c) *Flow Drops - Class 3*



(d) *Flow Drops per Class-of-Service*

Fig. 8 Average packet drops (**a**–**d**)

have been used: QoSMIC and PIM-SM. They are seen as top and bottom lines in results analysis. PIM-SM is the obvious bottom line because it connects each member on the shortest path towards the tree without doing any probings at all. On the other hand, QoSMIC is the reference for upper values because it can be configured in very aggressive local and tree-based search procedures.

GT-ITM [5] and BRITE [17] were used to generate interdomain topologies. Those topology generators use different methods to synthesize topologies that can be representative or have characteristics similar to the autonomous system structure of the Internet. Examples of the topologies generated are presented in Fig. 9.

Simulations were run only at the interdomain level, ignoring inside domain routing approaches. Relations between intra- and interdomain were not considered at this phase. A domain is therefore abstracted as a single node that initiates a join procedure. Instead of building complex traffic generation scenarios, the available bandwidth at each link was also generated with uniform distribution in the interval 1.5–10 Mb. Heavy tailed and exponentional distributions were also considered. Traffic generation has been avoided because it is very expensive in computational terms and usually impracticable. From extensive conducted experiments with different scenarios, only the node degree function seems to affect the results. Other parameters do not impact protocol comparison.

Since all path probing actions occur only when nodes join the group and no tree reconstruction is done at leave events, only join operations were considered in each simulation scenario. For each simulation, a node was randomly chosen as source and tree root. Then, one by one, 60% of all nodes join the group. Each node is randomly chosen among all unconnected ones. Multicast receivers attached try to connect on paths with higher available bandwidth. The multicast sender generates CBR traffic and 100 simulations were run for each topology.

5.2 Used metrics

The goal of the path probing strategy is to find the best available path that can be used to connect a new member to the multicast tree, according to its QoS requirements. So the first step to evaluate this procedure is to define the path quality, in terms of QoS, and measure it for all joining members. This *QoS obtained* metric can therefore be used to compare the efficacy of the probing strategy.

In this set of experiments, we have considered path quality as a function of the available bandwidth. Available bandwidth on a path is the minimum available bandwidth of all its links. Values are collected for the entire tree branches, from the tree root towards the new member. Each routing strategy, after path probing, provides a set of available connecting branches to the new member. Within the set of results, the best possible path is the one with the higher available bandwidth. From the new member perspective, the most efficacious strategy provide paths with higher available bandwidth.

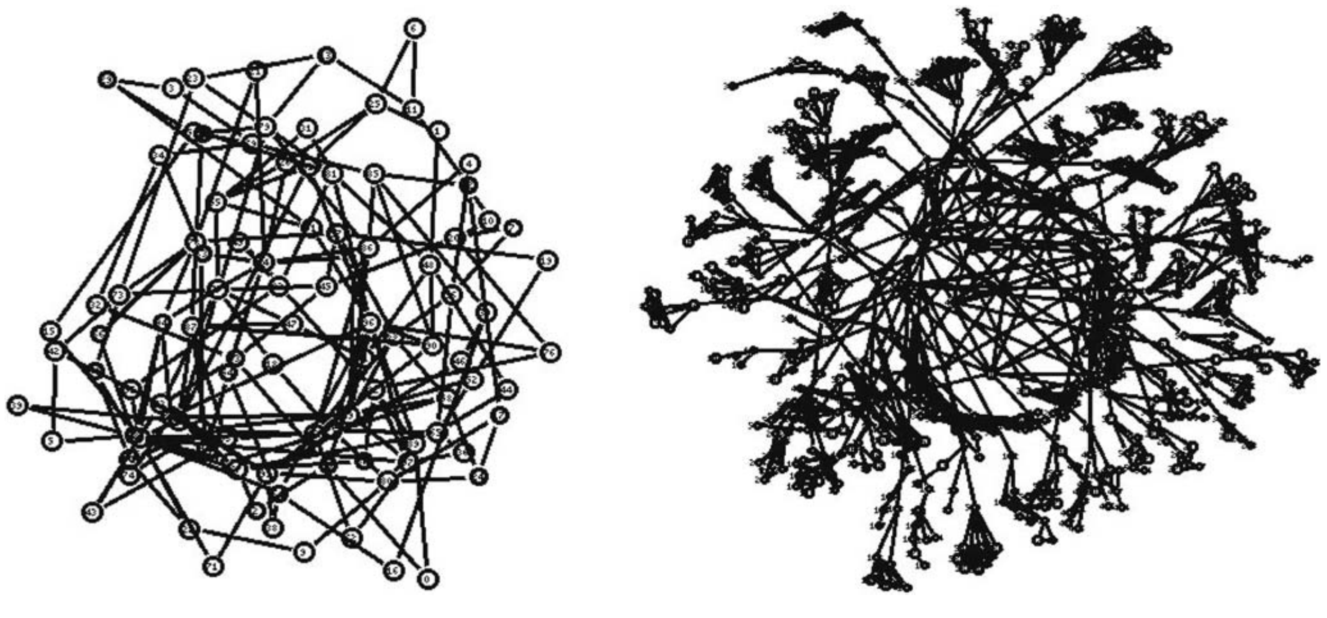


Fig. 9 Examples of BRITE AS (100) and GT-ITS (600) interdomain topologies

Success measures can be stressed graphically with the percentage of members vs the QoS obtained.

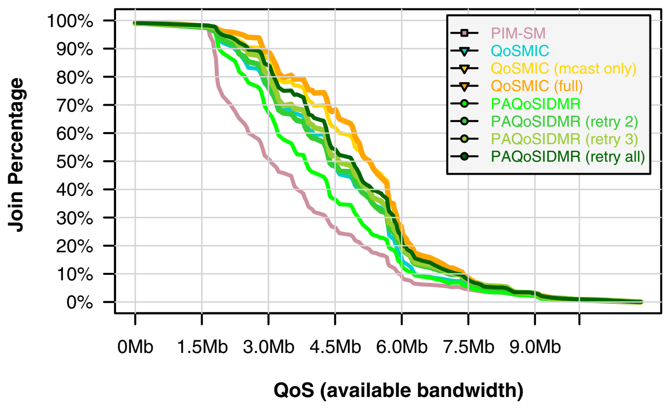
To measure the efficiency of the path probing strategy, the construction effort is estimated in terms of the number of control messages generated by each operation and the time spent in the entire probing phase. Some multicast routing approaches achieve better efficacy but at the expense of more construction effort, so to be more fair on result analysis, all metrics are needed.

5.3 Results obtained

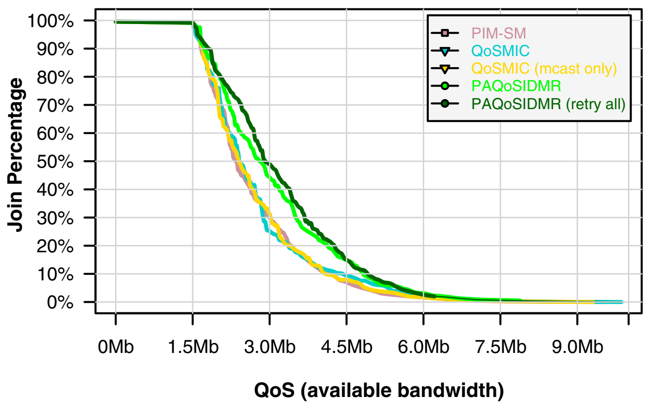
Figure 10 presents a set of results obtained when evaluating the path probing strategy. The top two graphics (Fig. 10a and b) show efficacy measures for the synthesized topologies illustrated in Fig. 9. Figure 10a shows QoS obtained for the BRITE AS (100) topology for all protocols. As expected, PIM-SM appears as the bottom line. This graph shows that the proposed strategy performs good but not as well as QoSMIC when only the first intree node launches probe messages. The graph shows that, with PIM-SM, only 50% of the members that joined the group achieve more than 3 Mbps available bandwidth. That percentage grows to almost 70% when using the proposed strategy and to near 80% with QoSMIC. However, the performance improves when the number of retries is increased. For two retries (two intree nodes initiate the probe procedure), the results are already similar to the normal QoSMIC version. This brings to evidence that inside tree search is a good approach, and it can avoid the complexity of the tree manager element in QoSMIC architecture. This improvement remains true for the GT-ITM AS (600) topology.

The relative positions of the other protocols remain the same in both graphics, showing no strong evidence that the type of topology affects results. This reading can be later transformed into a conclusion if confirmed by further experiments.

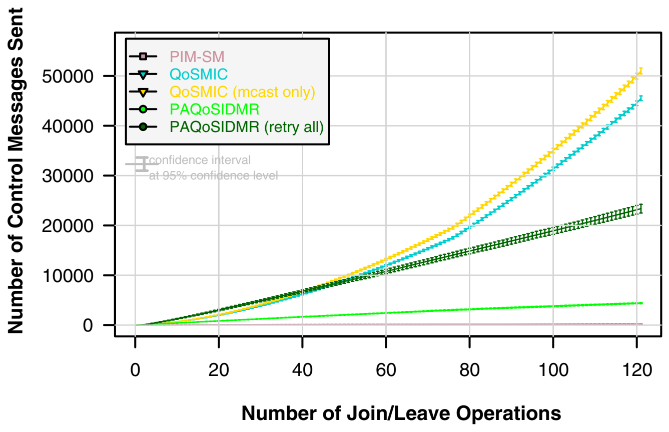
The two bottom graphics (Fig. 10c and d) show the efficiency metrics for the larger topology. The number



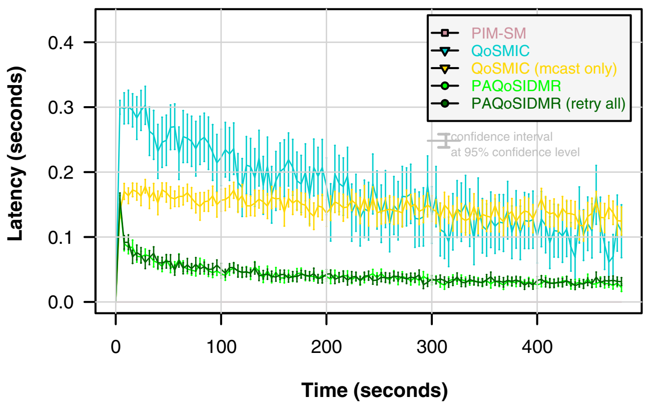
(a) QoS obtained by Receivers on BRITE AS (100) topologies



(b) QoS obtained by Receivers on GT-ITM AS (600) topologies



(c) Control messages over head on BRITE AS (100) topologies



(d) Join Latency on GT-ITMAS (600) topologies

Fig. 10 Simulation results: interdomain tree branch construction (**a–d**)

of control messages needed to do the probing and connect a new member is counted for every join event operation, and an average of all simulations is computed. Results are presented graphically using the accumulated values in order to stress the differences. The number of control messages is therefore always increasing as the number of join events increases. Results show that PIM-SM has little overhead and that the proposed strategy uses fewer control messages than QoSMIC. The join latency (time spent in the join procedure) is also presented in Fig. 10d. The normal QoSMIC starts with greater join latency because its local search procedure is unable to find any intree node in the new member neighborhood when the group is small and sparse. It becomes more effective for larger groups (join events increase with time). When we use only the tree-based search procedure of QoSMIC (QoSMIC-mcast), the values become more stable in time because they only depend on the distance between the new member and the tree manager that conducts the probing procedure. The join latency value is smaller for the proposed strategy because it only depends on the time to reach the first intree node. The values do not increase with the number of retries because, in these simulations, they are launched all at once.

6 Discussion and future work

Multicast has somehow been put aside from the differentiated service QoS model simply because it may involve tasks that are too complicated for domain core nodes. While packet replication is not complex at all, and may be allowed inside the domain, that is no longer true for packet classification and marking, usually done at edge routers. One of the problems to face is the difficulty to reflect receivers' QoS requirements, due to the unidirectional nature of DiffServ: packets are to be marked in the downstream path to the receivers.

At the interdomain level, difficulties arise from the need to interconnect different administrative domains, either DiffServ or non-DiffServ. Notice that traditional any-source multicast model allows multiple sources and receivers to join and leave any group at any time, whatever their administrative domain would be.

In this article, an innovative intra- and interdomain QoS multicast routing strategy has been proposed and evaluated by means of NS-2 simulations. The strategy was specially designed for class-based domains and allows every member, source, or receiver to join a QoS-aware multicast group according to its own QoS requirements. There is no need to remark packets inside a domain because multiple trees can be built, one tree per CoS. With this approach, multicast fits in DiffServ architectures, as well as unicast.

Simulation results show important decreases in packet drops when the new strategy is used and also evidence the ability to dynamically find, and commute traffic to, multicast distribution paths that avoid bottlenecks and congested links. Furthermore, this new routing strategy promotes CoS traffic differentiation, being able to keep traffic from priority classes within more stringent limits.

At the interdomain level, the same multiclass/multitree approach has been analyzed. Interdomain tree branches are now constructed using an efficient path probing mechanism that border domain routers issue in order to build directed QoS-aware trees, instead of establishing simple reverse (non-QoS-aware) path trees. The path probing strategy used in interdomain scenarios has been shown to be efficient both on flat and hierarchical topologies, without incurring significant overheads even when compared against non-QoS-aware interdomain solutions. Interdomain scenarios with 600 border domain routers have been used in NS-2 simulations. In such a large simulation scenario, building trees from roots towards leaf members induces greater join latencies. However, results have shown that this problem may be addressed by building multiple directed unidirectional multicast distribution trees. Furthermore, to avoid tree root routers overload, join requests are handled in a distributed manner by the first in-tree border domain router that receives them, thus relieving the tree root domain from this task.

Simulation results have also shown that QoS obtained with this new strategy, as perceived by receivers, is adequate, and join latency is kept small because new members try to find any in-tree BDR node, as fast as possible, by sending control packets towards the root domain.

Finally, results presented also show that, using this strategy, there are significant gains in the costs of distribution trees, even when compared with non-QoS-aware interdomain multicast routing solutions. Future work includes a careful evaluation of the usage of per-class precomputed routes, even without dynamic probing, also at the interdomain (BGP) level.

References

1. Adams A, Nicholas J, Siadak W (2005) Protocol independent multicast-dense mode (PIM-DM): protocol specification (revised). RFC 3973 (Experimental). http://www.ietf.org/rfc/rfc3973.txt

2. Apostolopoulos G, Guerin R, Kamat S, Tripathi SK (1998) Quality of service based routing: a performance perspective. In: SIGCOMM, pp 17–28. citeseer.nj.nec.com/apostolopoulos98quality.html
3. Ballardie A (1997) Core based trees (CBT version 2) multicast routing—protocol specification. RFC 2189 (Experimental). http://www.ietf.org/rfc/rfc2189.txt
4. Blake S, Black D, Carlson M, Davies E, Wang Z, Weiss W (1998) An architecture for differentiated service. RFC 2475 (Informational). Updated by RFC 3260. http://www.ietf.org/rfc/rfc2475.txt
5. Calvert K, Zegura E (1996) Gt-itm: Georgia tech internetwork topology models (software). http://www.cc.gatech.edu/fac/Ellen.Zegura/gt-itm/gt-itm.tar.gz
6. Carlberg K, Crowcroft J (1997) Building shared trees using a one-to-many joining mechanism. Comput Commun Rev 27:5–11
7. Carlberg K, Crowcroft J (1998) Quality of multicast service (qoms) by yet another multicast (yam) routing protocol. In: Proceedings of HIPARCH'98, June 1998
8. Chen S, Nahrstedt K, Shavitt Y (2000) A qos-aware multicast routing protocol. In: INFOCOM (3), pp 1594–1603. citeseer.nj.nec.com/article/chen00qosaware.html
9. Costa A, Nicolau MJ, Santos A, Freitas V (2005) A new path probing strategy for inter-domain multicast routing. In: First conference on next generation internet networks traffic engineering (NGI'2005). IEEE Comm Society, Catalog Number 05EX998C, ISBN 0-7803-8901-8, Libr of Congr 2004116428
10. Fall K, Varadhan K (2001) The NS manual. http://www.isi.edu/nsnam/ns/ns-documentation.html
11. Faloutsos M, Banerjea A, Pankaj R (1998) Qosmic: quality of service sensitive multicast internet protocol. In: Proceedings of the ACM SIGCOMM '98 conference on applications, technologies, architectures, and protocols for computer communication, pp 144–153. ACM, New York. doi:10.1145/285237.285276
12. Fenner B, Handley M, Holbrook H, Kouvelas I (2006) Protocol independent multicast-sparse mode (PIM-SM): protocol specification (Revised). RFC 4601 (Proposed Standard). http://www.ietf.org/rfc/rfc4601.txt
13. Fenner B, Meyer D (2003) Multicast source discovery protocol (MSDP). RFC 3618 (Experimental). http://www.ietf.org/rfc/rfc3618.txt
14. Kou L, Markowsky G, Berman L (1981) A fast algorithm for Steiner trees. Acta Inform 15:141–145
15. Kumar S, Radoslavov P, Thaler D, Alaettinoglu C, Estrin D, Handley M (1998) The masc/bgmp architecture for inter-domain multicast routing. In: Proceedings of the ACM SIGCOMM '98 conference on applications, technologies, architectures, and protocols for computer communication, pp 93–104, Vancouver, 31 August–4 September 1998
16. Lui KS, Nahrstedt K, Chen S (2004) Routing with topology aggregation in delay-bandwidth sensitive networks. IEEE/ACM Trans Netw 12(1):17–29
17. Medina A, Lakhinam A, Matta I, Byers J (2001) Brite: an approach to universal topology generation. In: Proceedings of the international workshop on modeling, analysis and simulation of computer and telecommunications systems-MASCOTS '01, Cincinnati, August 2001
18. Moy J (1994) MOSPF: analysis and experience. RFC 1585 (Informational). http://www.ietf.org/rfc/rfc1585.txt
19. Moy J (1994) Multicast extensions to OSPF. RFC 1584 (Proposed Standard). http://www.ietf.org/rfc/rfc1584.txt
20. Nicolau MJ, Costa A, Santos A (2007) Design and evaluation of a multi-class based multicast routing protocol. In: Proc of the 21st edition of the international conference on information networking (ICOIN 2007), Estoril
21. Rekhter Y, Li T, Hares S (2006) A border gateway protocol 4 (BGP-4). RFC 4271 (Draft Standard). http://www.ietf.org/rfc/rfc4271.txt
22. Thaler D (2004) Border gateway multicast protocol (BGMP): protocol specification. RFC 3913 (Informational). http://www.ietf.org/rfc/rfc3913.txt
23. Waitzman D, Partridge C, Deering S (1988) Distance vector multicast routing protocol. RFC 1075 (Experimental). http://www.ietf.org/rfc/rfc1075.txt