Abstract—Multicast routing is again gaining popularity with the development of new group services, like Mobile TV and pay-per-view. The aim is to reach big groups of users with the quality level they expect, while using network resources in an efficient way. This is a challenge to be fulfilled as current multicast protocols still have difficulties in dealing with basic issues like asymmetric routing. In this work we evaluate an overlay that allows the use of the source-specific multicast standard in environments with asymmetric routing. A set of tests is made both in the data and in the control plane, to expose the pros and cons of this new proposal.

Index Terms—Multicast routing, routing asymmetry, network simulation

I. INTRODUCTION

The deployment of real-time group communications with efficient use of network resources implies that multicast data should always travel in the best possible path. To accomplish this, routing asymmetries, as well as Quality of Service (QoS) requirements, should be considered when building multicast trees. However, these requirements are not accomplished by most IP multicast routing protocols. One of the reasons is that those protocols build multicast trees from the receivers to the sender, while data travels in the reverse direction. This fact may lead to an inefficient use of the network and to the failure to deliver the quality levels expected by the users, not only because each direction may present different quality characteristics, but also because the paths can even be completely different. The result is that data is forced to travel in a path that is not optimized and sometimes not even suitable, leading to inevitable loss of performance. Studies made by Vern Paxson [1] indicate that routing asymmetry has an usual presence in the Internet, with paths visiting 50% of the time at least one different city in each direction (in 1995), and at least one different autonomous system, 30% of the time. Routing asymmetries can happen due to distinct factors, such as different paths for each direction, same path but different bandwidth for each direction in the same link, as well as quality of service or network access restrictions.

In this technical report, a new protocol called Overlay for Source-Specific Multicast in Asymmetric Routing environments (OSMAR) [2] is described, and evaluated using the Network Simulator version 2 (NS2) [3] [10] [11]. The aim of the OSMAR protocol is to overcome the lack of adaptation of current multicast protocols to asymmetric routing. Specifically, OSMAR protocol allows source specific protocols such as the Protocol Independent Multicast -
Source Specific Multicast (PIM-SSM) [4], to operate in asymmetric routing environments, without requiring any change to their state machine [2]. The adaptation that OSMAR offer for multicast protocols to operate in asymmetric routing environment is used to support the transport of multi-user multimedia traffic in the Q3M architecture [14]; In Q3M, the multicast operation supported by OSMAR allows the implementation and control of group communications [12] [13].

The tests in this paper were made using PIM-SSM as the underlying multicast routing protocol, and concern both the data plane and the control plane of the OSMAR protocol.

II. OSMAR AND PIM-SSM

In this section an overview of OSMAR is presented, together with some details of its implementations in NS2. Implementation of the PIM-SSM protocol and of the Multicast Routing Information Base (MRIB) in NS2 are also addressed.

A. Overview of OSMAR

OSMAR can be used with current source-specific multicast protocols, like PIM-SSM, enabling them to deal with network asymmetries and to provide QoS-aware content distribution, while maintaining their specifications. This is accomplished by changing the values of the MRIB, used by the multicast routing protocol to build multicast trees, in a way that permits the construction of optimal trees from the source to receiver. The new OSMAR enabled trees are built based on the paths used by data while travelling from source to destination, in opposition to usual protocols, that create trees based on the path from receiver to source. Therefore, multicast branches are created taking into consideration the QoS characteristics of the real path to be used by data.

1) Detailed functionality of OSMAR

OSMAR aim is to enable source-specific multicast protocols to operate in environments with asymmetric routes, both inside a domain and between domains. The design of OSMAR allows the adaptation of source-specific multicast protocols to asymmetric routes in a distributed manner, in which most of the functionality is done by OSMAR agents residing in edge routers. Interior agents only perform the update of their MRIB, keeping no state. Moreover, OSMAR can be progressively deployed, because it does not require the installation of agents in all network domains.

In the inter-domain, OSMAR helps building QoS multicast Autonomous System (AS)-trees from the AS where the source is, to the AS where each receiver resides. In the intra-domain, OSMAR helps building QoS multicast trees from the ingress-router of each AS included in the AS tree, to each egress router where receivers have subscribed the session. The way OSMAR helps to build QoS multicast trees relies only in the change of MRIBs, in inter-domain and intra-domain source-specific multicast branches, independently from multicast protocols. Every MRIB in those branches is updated in a receiver-initiated, source-originated operation.

When a receiver subscribes a source-specific multicast channel, the update of MRIBs in the inter-domain multicast branch is initiated if the AS where the receiver resides is not yet part of the requested multicast tree. In this case, the egress agent that initiated this process has the opportunity to select from starting updating the multicast branch from the AS where the source is, or from the first AS that is found in the requested inter-domain tree. This means that the initiating agent can decide for a longer construction time, but a probably more suitable QoS multicast tree, if it chooses to start updating from the AS where the source is, or can decide for a shorter construction time, but a probably less suitable QoS multicast tree, otherwise. In any case, it is the responsibility of the Exterior Gateway Protocol to provide the most suitable route from the agent in the selected AS until the agent in the ingress agent of the AS where the receiver that triggered update of the inter-domain resides.

Intra-domain multicast branches are updated when an ingress agent for the desirable multicast tree is already known. In this case, it is the ingress agent that starts updating MRIBs in the path until the egress agent that wants to subscribe to that multicast channel. It is the responsibility of the Interior Gateway Protocol to provide the most
suitable route to the egress router. It is the responsibility of ingress agents not only to trigger the update of MRIBs in existing intra-domain branches, but also to notify other agents in the same domain about the multicast channels that enter the domain at its location.

Fig. II-1 shows the request to update the MRIB for a given \(<S,x>\) channel. In this scenario, \(<\text{Receiver 1}>\) already subscribed to a multicast channel from source S, and so the MRIB in the path from the source was already updated by OSMAR for \(<S,x>\) (multicast channel of the source S towards any destination x). There are also two egress agents requesting the update of MRIBs, since Receivers 2 and 3 wish to join channels from source S. The egress agent near \(<\text{Receiver 2}>\) sends a local request, since there is one ingress agent in the domain that has already state for \(<S,x>\). Finally, the egress agent near \(<\text{Receiver 3}>\) requests the update of the inter-domain branch since it has no egress agent in the domain, having selected to update the multicast branch from the first AS that belongs to the inter-domain tree.

![Fig. II-1 – Request to update MRIB](image)

Having requested the update of MRIB, OSMAR is now ready to perform the actual update. Fig. II-2 depicts how the new branches towards the leaf routers of Receiver 2 and 3 are updated, and how the ingress agents announce to other local agents the channels \(<S,x>\) that enter the domain at their location. Interior agents only perform the update of the MRIB and do not keep any state.
If the egress agent near <Receiver 3> had requested the update of the inter-domain branch from the AS where the source is, trying to benefit from a longer but probably more suitable QoS multicast tree, the request would go up to the agent near the <Multicast Source>. Therefore, the following update of MRIBs would be done from the source of the multicast group to the receiver.

The path taken by the request to update the MRIB and the following update path may not follow the same links due to network asymmetries, or by the existence of a QoS-aware unicast routing protocol. In this manner, the updated MRIB, with which PIM will build the multicast tree, has the best path from the source to receiver.

2) OSMAR operation (simplified)

The use of OSMAR [2] begins when a receiver intends to join a multicast channel <S,G>, where S is the IP address of the source and G is the multicast group selected by the source for this channel. At the time OSMAR detects that the PIM JoinDesired(S,G) variable was set to True, OSMAR process is initiated. The OSMAR agent in the receiver’s access-router starts by sending an MribReq message to request for the updating of the MRIB (Fig. III-1). This message is sent towards the source of the multicast group with the IP alert [5] option active. By default the message is captured by the access-router near the source. However, if the Fast_Branch field of the message is set, the message is halted by the first visited agent belonging to the specified multicast channel. The path taken by the message is the path indicated by the available unicast routing protocol. On reaching its destination the MribReq message is caught and a new MribUp message with IP alert option set is sent towards the sender of the MribReq message. The MribUp message updates the MRIB table in every node along the path with the address of its upstream neighbour (Fig. II-2). As previously said, the path taken by the MribUp message can be different from the one used by the previous MribReq message. After receiving the MribUp message, the OSMAR agent in the access-router near the receiver allows the continuation of the join process by the multicast protocol. OSMAR does not intend to alter in any way the PIM protocol, it simply forces the graphing of a new path by PIM. Periodically, the access-router near the source of the multicast group sends MribUp messages toward any access-router that has receivers that have joined the multicast channel, in order to refresh the paths.
B. Implementing OSMAR in NS2.

In this simulation, a simplified version of OSMAR mechanisms was implemented (Fig. II-3).

While OSMAR defines three types of messages to exchange control between agents, in this simulation only MribReq and MribUp messages were simulated. The reason for not implementing the SessAn message, is that its usage, to keep signalling local to access networks, is a clear advantage of OSMAR, and we do not aim to analyse OSMAR in best-case scenarios. The time spent internally to update the MRIB is also not simulated as it is not representative. The size of each OSMAR packet is of 42 bytes: 20 bytes for the Internet Protocol (IP) header, 8 bytes for the User Datagram Protocol (UDP) header and 14 bytes for OSMAR. These 14 bytes are composed by the OSMAR message type, the address of the previous node in the path, the source address of the multicast channel, the group address to join, and by the Fast_Branch field. Although not every field is used in any OSMAR message, the same packet size was used in simulating all messages.

Periodically, a timer sends an MribUp message toward each stored egress agent. This message refreshes the MRIB information in all interior nodes of the path toward each egress agent.

C. Implementing PIM-SSM in NS2

As PIM-SSM is not currently distributed with NS2, a basic implementation [6], based on the bundled NS2 Centralized Multicast routing protocol [3], was used. This implementation was then revised and extended to support the use of the MRIB table. Nevertheless, this new version of PIM-SSM still does not include the exchange of packets necessary to the creation of the multicast tree, nor the exchange of packets used for the subscription to a multicast group by Internet Group Management Protocol (IGMP) [7]. This two features are computed at once and, at the same time, membership is set in the node interfaces for the specific <S,G> channel.

In order to reduce simulation time, the normal 30 s taken by PIM-SSM “Hello” messages, that provide the refresh of multicast trees, was reduced to 0.3 s. PIM-SSM “Hello” message is simulated by a timer that periodically forces the re-computation of multicast trees.
D. Implementing MRIB in NS2

PIM relies on an underlying topology-gathering protocol to populate the MRIB [8]. Regardless of how it is created, the primary role of the MRIB in the PIM protocol is to provide the next-hop router along a multicast-capable path to each destination subnet. The routes in this table may be taken directly from the unicast routing table, or they may be provided by a separate routing protocol. In this simulation, when OSMAR is active, it is the only one that changes the MRIB. When OSMAR is not active, MRIB values are taken from the unicast routing tables.

In order to implement OSMAR in NS2 a structure was created to simulate the MRIB. In its initial status, this structure is initialized with values taken from the unicast routing table. As more nodes join the multicast tree, and before the multicast routing protocol in use is triggered, the MRIB is modified by OSMAR.

III. OSMAR EVALUATION

In order to test OSMAR three different scenarios were created. For each scenario, three asymmetry levels were considered - 0%, 30% (average Internet asymmetry level [1]), and 60%. Then, six topologies were randomly created with Brite [9] for each scenario (two for each asymmetry level), in a total of 18 topologies. Each of these topologies has 37 nodes (including 1 node that acts like a multicast source and six receivers), and links with a bandwidth of 1 Mb/s. The multicast source sent traffic at a constant bit rate of 500 Kb/s, with a 128 bits packet size. Asymmetry levels were created by assigning different node delays (and corresponding weights) for traffic travelling in the two different directions of some links. There is no additional traffic in the network.

The first scenario has symmetrical links with a 2 ms delay in both directions. Asymmetric links have 1ms and 3 ms randomly put in each different direction of the link. The second scenario has random delays (between 1 and 4 ms) in each symmetric link and random delays in each direction of each asymmetric link. This scenario is the most random one. Finally, the third scenario has 2 ms link delay for every link, either symmetric or asymmetric, and random link weights in each direction of asymmetric links. This last scenario corresponds to a network where the routing is only affected by decisions of the network administrators.

In every test there is a multicast source node and six receiving nodes, connected to different points of the network. Each of the six nodes will join the multicast channel during the simulation.

The defined scenarios aim to evaluate the potential of OSMAR while working together with PIM-SSM, in comparison to a PIM-SSM alone approach. The main objective is to determine until what extend OSMAR allows PIM-SSM to build more efficient multicast trees, providing a better traffic flow independently of the network conditions. In this analysis, we use as criteria the end-to-end delay, the session setup time, the signalling overhead of OSMAR, its robustness, and the overall network load.

In this paper only averages including all scenarios are presented. Detailed results for a specific scenario or particular topology are referenced in the text only when relevant.

A. Impact on data end-to-end delay

This test focuses on the data plane, by measuring how much time the data packets take from source to receiver (the multicast tree is already built). Both, PIM-SSM alone and OSMAR + PIM-SSM approaches are used. In the later case OSMAR is used with Fast_Branch option flag set and unset. The reason for using Fast_Branch set and unset is that, in spite of apparently only affecting the time to build the multicast tree, that flag can also lead to the creation of different multicast trees, and therefore producing different delays.

While measuring end-to-end delay, it is clear that OSMAR brings benefits in asymmetric topologies, either with Fast_Branch set and unset. These benefits grow along with the asymmetry level as can be seen in Fig. III-1, which contains the average of the delay measured in all topologies.

By analysing results in more detail it is found that, in most cases, OSMAR with the Fast_Branch flag unset leads to lower end-to-end delays. This may be explained by the fact that without this option the tree is built from the
multicast source, and not from an intermediary branch. This difference leads to the construction of multicast trees with lower end-to-end delay since all the possible links from source to destination are considered.

For a better analysis of how much faster is OSMAR when compared with PIM-SSM, we calculated the speedup of OSMAR (with Fast_Branch flag unset) vs PIM-SSM. The speedup, illustrated in Fig. III-2, is the ratio between the average delay induced by PIM-SSM alone and the average delay induced when using OSMAR.

The speedup averages 1.0508 in 30% asymmetry level topologies, and 1.0907 in 60% asymmetry level topologies. In 0% asymmetry level topologies, a small decrease in the performance of OSMAR, relatively to a PIM-SSM approach can be detected. This is due to the small overhead of the OSMAR control messages - the transmission of MribUp messages in response to MribReq messages, and the MribUp messages that OSMAR refresh implies, results in a slight delay of some CBR data packets. However, the difference is minimal, averaging more 0.035 ms than when using PIM-SSM alone.

By analysing results in more detail it was found that, in most cases, OSMAR protocol with Fast_Branch option flag unset achieves a lower delay. This may be explained by the fact that with this option not set the tree is built from the multicast source to destination, not from a previous branch that is found when the node starts the join. This difference leads to the construction of better multicast trees.

B. Impact on session setup time

This test measured the time taken by OSMAR messages to make the necessary changes to the MRIB. Time is measured from the initial MribReq message sent when a node wants to join the multicast channel <S,G>, until the time when the corresponding MribUp message is received by the initial joining node. Results using OSMAR with
and without the *Fast_Branch* option are displayed in Fig. III-3.

![Average MRIB changing time by asymmetry](image)

**Fig. III-3 – Average MRIB changing time by asymmetry**

As shown in Fig. III-3, which includes all tests done in every topology, average times to change MRIB are slightly bigger with higher asymmetry levels, both with and without *Fast_Branch* option flag set. The measured average setup time, due to the time that OSMAR need to change the MRIB in all routers, range from 13.686 ms (0% symmetry scenario) to 14.534 ms (60% symmetry scenario) with *Fast_Branch* option set, and from 22.273 ms (0% symmetry scenario) to 24.574 ms (60% symmetry scenario) with *Fast_Branch* option unset.

As could be forecasted, the *Fast_Branch* field set option shows clear benefits in what respects the setup time, since with the *Fast_Branch* field OSMAR starts configuring the MRIB closer to the receivers.

In Fig. III-4, the speedup is the ratio between the average time to change the MRIB without the *Fast_Branch* option and with it. By analysing it, we can see that the benefits of using the *Fast_Branch* option in the different asymmetry levels, present small differences. Moreover, it can be noticed that in scenarios with 30% of asymmetries, the speedup ratio is lower than the ratio with 0% asymmetries, what can be explained by the dependency of this results on the specific topologies used.

Based on this test, we can also conclude that the asymmetry levels of the different topologies scenarios do not influence the results significantly.

![Speedup of Fast_Branch set vs Fast_Branch unset](image)

**Fig. III-4 – Speedup of *Fast_Branch* set vs *Fast_Branch* unset**

C. **Impact on session setup time (optimized)**

One of the drawbacks of using OSMAR is the initial time spent by OSMAR to update the MRIBs, which delays the start of the reception of data by the receiver. However, another approach is possible. This alternative approach permits the occurrence of a normal multicast join and, at the same time, the start of OSMAR operation. This is,
OSMAR does not block the multicast join request, and so, the join is made before OSMAR finishes changing all MRIBs. Hence, the multicast trees build by PIM-SSM use two types of MRIB values: first, during tree setup, the MRIB based on the unicast routing table, and after that, during tree refreshment, the MRIB updated by OSMAR.

The objective of this test (without Fast_Branch) is to measure how many data packets are lost in the transition from the original PIM-SSM built tree to the new OSMAR changed tree. The number of out-of-order packets that arrive to the receiver is also measured.

Simulation results show that, with a 500 Kb/s CBR source and 128 bytes data packets, the average number of lost packets, to each of the receiver access routers, is minimal (Fig III-5). Out-of-order packets do not surpass one packet (it is the first packet received after the lost ones, therefore being out-of-order). It can also be seen that the average number of lost and out of order packets slightly increases with higher asymmetry levels. This can be explained by the fact that higher asymmetries imply that more branches of the multicast tree will be changed in the transition from the first built multicast tree to the OSMAR enabled multicast tree, as OSMAR will try to build the optimal tree.

In the interpretation of these results, it should be noted that due to the fact that the PIM-SSM implementation does not have packet exchange, therefore being immediate either in the first time it is used as in consecutive refreshes, there are packets that are caught in the middle of their path when the multicast tree is changed. This results in situations where the packets suddenly do not have a path to follow, and therefore are lost. With a packet exchange implementation, the overall number of lost packets should be less that the ones resulting from this simulation.

D. Signalling overhead

The main objective of this test was to measure the overhead produced by OSMAR in the network, and its scalability.

As would be expected the number of OSMAR packets accounted in each link is higher when Fast_Branch option flag is not set, as in that situation the packets have to travel the complete path between each receiver access router to the multicast source. The packets that are accounted for, are the OSMAR packets sent by each node (that simulates a router) in the network, either being the initial sending node or any of the routers that forward the packet through the network. In Fig. III-6 the average number of OSMAR packets in the network is displayed by asymmetry (includes all scenarios). The average number of OSMAR messages in the network varies slightly with asymmetry, reflecting the different number of links in the many multicast trees tested - as the tree has one more link there is one more OSMAR control message (travelling downstream) set to the network through that specific link.
The distribution of the average number of OSMAR and CBR packets along all the simulation time is displayed in Fig. III-7 and Fig. III-8. This average is the average of packets in all topologies tested. In this simulation, receiver access routers signal their intention to join the multicast channel at instants 0.1 s, 0.3 s, 0.5 s, 0.7 s, 0.9 s, and 1.1 s.

When the first receiver joins the multicast channel, OSMAR is triggered leading to a small bump in the number of OSMAR packets in the network. The reception of MRIBup messages signals the access-node that the multicast join can now be completed, as the MRIB is already changed (the optimized version analysed in section III-C is not used). This leads to the growing of CBR packets in the network. The successive joins lead to successive increases in the number of CBR packets in the network, that only decrease in the end of the simulation as nodes leave the multicast channel. Each 0.3 seconds there is an increase of OSMAR messages in the network, which corresponds to OSMAR refresh messages (the ones that permit the maintenance of the MRIB table). It can also be seen that, when Fast_Branch option is set, the number of OSMAR messages is lower than with Fast_Branch unset.

Moreover, the simulation set the OSMAR refresh timer to 0.3 s, while it is expected that it should be close to the
value of PIM-SSM refresh messages that is 30s. This bandwidth can also be reduced as OSMAR specification separates inter from intra-domains.

As for scalability, we can observe that the number of packets grows linearly with the number of receivers, and can even be reduced by using the Fast_Branch field set, and by separating the topology in different domains, in which the OSMAR announcement mechanism is used.

It should also be noticed that the bandwidth occupied by OSMAR does not relate to the CBR data packets bandwidth, as it only depends on the number of receivers that join the multicast channel.

E. Robustness

In this test it is measured the time that the network takes to recover from a broken link. In every topology a link was chosen from those that belonged to both the PIM-SSM alone, and PIM-SSM with OSMAR (with Fast_Branch flag unset), multicast trees. Then, from time 1,50 s to 2,10 s this link was disconnected, simulating a link failure. As the link is interrupted and then re-activated two different situations are produced. In the first, the multicast tree has to be rebuilt because at least part of it was affected by the cut. The second situation happens when the link is restored. In this case the tree does not need to be rebuilt in every case because the routing may not be changed.

Fig. III-9 to Fig. III-11 depict this situation, giving an average of lost and out-of-order packets in all topologies, and by showing a timeline of the average number of received packets.

Simulation results show that OSMAR averages a lower percentage of packet loss than PIM-SSM alone approach (Fig. III-9). The packet loss does not have a direct correlation to the asymmetry level, depending mostly in the chosen topologies. Out-of-order packets are minimal in both OSMAR and PIM-SSM alone scenarios (Fig. III-10).

![Average number of CBR lost packets in the network during the simulation](image)

**Fig. III-9** – Average number of CBR lost packets in the network during the simulation

![Average number of CBR out-of-order packets received by access-routers during the simulation](image)

**Fig. III-10** – Average number of CBR out-of-order packets received by access-routers during the simulation

In Fig. III-11 it is shown the average number of CBR data packets that are received by access-routers along the simulation time. After the break of the link at 1,50 s it is clear that the number of CBR packets that are received in all
destinations falls sharply. It does not reach zero, because not all receivers were affected by the break. Moreover, as the graphic displays an average of all receivers in all topologies, there are still packets being received by some receiver access-router after the link break, as those packets were already in the network, in a point not affected by the break. After the creation of a new tree (that happens at time 1,8 s, when PIM-SSM refresh rebuilds the tree – PIM-SSM refreshes every 0,3 s) some receivers start to receive CBR packets.

When the link connection is re-established, at time 2,10 s, nothing happens as the multicast tree remains the same, only changing when the next PIM-SSM refresh takes place, at time 2,4 s. At this point in time, there is only a change of path, as some routes are changed, which creates a small turbulence but not a heavy packet loss.

It must be noted that these results depend highly upon the OSMAR refresh time, and in its relation to the PIM-SSM refresh time. If after the link break the first OSMAR refresh occurs after the first PIM-SSM refresh, the latter (that relies on an updated MRIB table to cope with the link failure) will not have updated routes. The update of the routes will only happen after the OSMAR refresh (and after subsequent updating process), and till that time PIM-SSM continues to use the old paths which do not cope with the link failure. The optimal solution is to ensure that OSMAR is closely followed by PIM-SSM refresh. In this manner when PIM-SSM builds the multicast tree it uses the most recent updated routes. In this simulation the refresh time equals that of PIM-SSM, i.e., the refresh happens every 0,3 seconds.

F. Overall network load

In this test the number of CBR data packets generated are counted in either OSMAR (with different Fast_Branch settings) and PIM-SSM alone scenarios. A distribution of packets along the simulation is also shown.

In the depicted graphics, OSMAR shows different behaviours, when compared to the PIM-SSM alone approach, depending on the status of the Fast_Branch option flag (Fig. III-12).
With a 0% asymmetry scenario OSMAR with \textit{Fast\_Branch} unset achieves the worst result, averaging a difference of more than 300 CBR packets to the PIM-SSM alone approach. With \textit{Fast\_Branch} set the difference is only of 10 packets (again favouring PIM-SSM). With a 30\% asymmetry scenario OSMAR with \textit{Fast\_Branch} unset presents slightly more packets than PIM-SSM alone, while OSMAR with \textit{Fast\_Branch} set has a lower number of packets. Only in a 60\% asymmetry scenario the number of CBR packets is lower than PIM-SSM only in both OSMAR options.

Fig. III-13 depicts the average number of CBR packets in the network, in 0.01s intervals, along simulation time, considering all scenarios and topologies. In the graphic presented, the six steps represent the joining of each of the receiver access-router. Oscillations in each step occur because despite having a CBR source, 0.01s intervals are used in the graphics, therefore producing some differences. Fig. III-14 shows the detail from 1s to 1.4s of simulation time.

From the results achieved, it can be concluded that OSMAR brings no clear benefit over PIM-SSM only approach, even when asymmetries exist. The number of packets depends heavily upon the specific topologies. When OSMAR is used, setting the \textit{Fast\_Branch} option flag tends to achieve better results.

The fact that OSMAR with \textit{Fast\_Branch} option flag unset achieves the worst result should indicate that it has a more spread multicast tree. However this is not exactly what happens. As an example, the three multicast trees built for topology 13 (this topology has an asymmetry of 0\% and belongs to the scenario where only link weight change) are shown in Fig. III-15. This topology was chosen as it is one of the cases where OSMAR with \textit{Fast\_Branch} unset achieves worst results. All trees are depicted with the differences between them shadowed. The multicast tree that exists after the join of node 32 (the node that is responsible for the main differences in the trees) is in full line, while the tree branches built after are hatched (as they have not yet been built at that time – remember that nodes 31 to 36 join sequentially).
By counting the number of links in all trees, the same number is achieved - 17. So, the number of CBR packets should be the same, but it is not. To understand this fact, the sequence of joins must be followed. When both nodes 31 and 32 have joined the number of links in the OSMAR tree with Fast_Branch unset is 9, while in other both cases the number equals 8. The same happens after as showed in Fig. III-16. This difference causes a different number of CBR packets in the topology as can be stated in Fig. III-17. In this example the number of links is the same after the join of node 36 but, until the last node joins, OSMAR with Fast_Branch unset has always an higher overall number of links in the multicast tree. In this manner, only in the last stage all trees have the same number of CBR packets in their links (last step in the graphic). However, the overall number of packets during all the simulation penalizes the OSMAR with Fast_Branch unset. Despite the fact that OSMAR with Fast_Branch unset builds a tree with more links (at least until the join of last node in the example presented), it must be noticed that the number of links to reach each of the nodes from the source is the same in all scenarios, only the number of common links varies.

\[
\begin{array}{|c|c|c|c|}
\hline
\text{Joining nodes} & \text{OSMAR with FB=0 tree links} & \text{OSMAR with FB=1 tree links} & \text{PIM-SSM tree links} \\
\hline
31 & 4 & 4 & 4 \\
32 & 9 & 8 & 8 \\
33 & 12 & 10 & 10 \\
34 & 13 & 11 & 11 \\
35 & 15 & 14 & 14 \\
36 & 17 & 17 & 17 \\
\hline
\end{array}
\]

\(\text{Fig. III-16 – Number of multicast tree links in topology 13}\)
From the result of this example it can be said that OSMAR, with Fast\_Branch set, averages a lower number of tree links, therefore having a lower overall number of packets in the network. This can be explained by the fact that with Fast\_Branch set a joining node only tries to reach the first node with the wanted multicast channel in its interfaces, therefore optimizing the use of existing multicast branches. With Fast\_Branch unset the OSMAR enabled tree is built from the multicast source what may lead to the creation of a multicast tree that does not favour the use of existing multicast branches.

IV. CONCLUSION

The need of QoS support in multicast delivery of multimedia contents is growing, not only to provide for the best use of the network resources, but also to allow the fulfil of needs of real-time group communications. However, most of today’s multicast routing protocols do not provide for any QoS restraints in the building of multicast trees, while taking into account routing asymmetries. In this paper we present OSMAR, an overlay that allows enhancing PIM-SSM trees with QoS awareness in network with routing asymmetries, without changing the standard.

In the simulations OSMAR was tested over PIM-SSM multicast routing protocol, either in data and control plane, to find out if it could provide a better overall performance. As result of all the tests made some conclusions can be drawn. The first is that in the presence of asymmetries in the network, OSMAR brings clear advantages, especially by averaging a lower end-to-end packet delay with a negligible overhead. At the same time, OSMAR with Fast\_Branch option flag set, averages a better overall network load of data packets (which implies that it does not build longer multicast trees than a PIM-SSM alone solution). OSMAR also showed to be at least as robust as PIM-SSM, by averaging equal interruption times after a link break, and a smaller number of lost packets. However, care must be taken to ensure that OSMAR refresh is closely followed by PIM-SSM refresh in order to minimize the effects on the recovering time from a broken link.

The main drawback initially expected from OSMAR, the overhead that it would create, turned out to be almost irrelevant. This forecasted overhead has two causes: the impact on the session setup time and the packet overhead in the network. The former results from the need to initially change the MRIB tables that would slow the multicast join. Hence, an alternative approach that enables simultaneous PIM-SSM joins and OSMAR MRIB updates is tested. This optimized version proved to be more efficient than the initially proposed, achieving a performance equal to PIM-SSM despite a small cost in the amount of lost packets. In what respects the packet overhead, it was found that it grows linearly with the number of receivers, while consuming a negligible portion of the bandwidth.

After analyzing all the results, it can be said that OSMAR allows PIM-SSM to build a better multicast tree than when PIM-SSM is used alone in scenarios where routing asymmetries exist. By averaging a lower end-to-end delay, while not increasing the overall network load, it may enable traffic to travel through the more advisable links, as the network managers characterized them (as it follows lower cost links), providing a QoS-aware protocol that PIM-SSM by itself cannot guarantee.
REFERENCES


[9] "brite: Boston University Representative Internet Topology Generator"


