International Journal of Information and Network Security (IJINS))
Vol. 2, No. 6, December 2013, pp. 471 – 481
ISSN: 2089-3299

Multicast in Multi Protocol Label Switching: A comparison study between LDP and RSVP-TE

**Mohamad Chaitou*[*] and Hussein Charara*[*]
[*]Faculty of Science, Lebanese University

Article Info

Article history:
Received Sep 12th, 2013
Revised Oct 18th, 2013
Accepted Nov 20th, 2013

Keyword:
Multicast
traffic engineering
network scalability
performance evaluation

ABSTRACT

Resource ReSerVation Protocol-Traffic Engineering (RSVP-TE), and Label Distribution Protocol (LDP) may be used to set up Point to Multi Point (P2MP) trees in MPLS. RSVP-TE performs better in optimizing network resources, while LDP is simpler but has no guarantee on resource optimization. This paper presents a comparative study between RSVP-TE and LDP in regards of resource optimization and the resulting impact on the amount of memory consumed. It shows that the amount of memory needed in the case of RSVP-TE grows linearly as the size of the tree increases. In contrast, an approximate constant behavior is observed in the case of LDP, yielding an important scalability property. The paper then proposes two extensions to LDP aiming to achieve better resource optimization. In both extensions, a new leaf is provided a partial tree knowledge, by involving either all the nodes of the tree or only its leaves. The leaf joins the tree by connecting to the closest node among the known ones. Valuable comparisons with RSVP-TE are performed, and they represent an important background to decide when and how to use each protocol.

Corresponding Author:
Mohamad Chaitou
Lebanese university, Faculty of Science
Address: beirut, Lebanon
Phone: 0096171574315
Email: mohamad.chaitou@gmail.com

1. INTRODUCTION

The need of establishing Multicast trees is more and more motivated by the permanent increase in multimedia based traffic, such as conferences, broadcasting, teaching, and distributed processing etc. In Point to Multi Point (P2MP) multicast trees, the root node sends the same information to many receivers, called leaves of the tree.

This paper focuses on the support of P2MP trees in Multi Protocol Label Switching (MPLS) networks [1]. We concern ourselves in this paper with Virtual Private Networks (VPNs) [2], where the network is abstracted as a set of edge nodes called Provider Edges (PEs), and a set of Provider routers (the P routers). Roots and leaves are always PE nodes. The control plane of MPLS offers the possibility to establish P2MP Label Switched Paths (LSPs)[1] in two different approaches. In the first approach, RSVP-TE is used as a signalling protocol in the goal of resource optimization and Traffic Engineering (TE) deployment [3]. Resource optimization refers here to the minimization of the number of links used by the P2MP tree. Intuitively, this increases the amount of bandwidth saving. This is achieved by calculating the path of the tree at the root node. The latter executes an algorithm that aims to minimize a cost function defined by the sum of links of the tree, which is known in the literature as the Steiner Problem in Networks (SPN) [4]. Since SPN is NP-Complete [4], several heuristics approximating it have been proposed [4]. One among them, proposed by Takahashi and Matsuyama [5], is known by its simplicity in grafting (i.e., adding) and pruning leaves, which promotes it for network applications. This heuristic, henceforth denoted as TakaMat heuristic, is hence used as a background for this study. The principle of this heuristic is simple: as soon as the root is aware about a new leaf arrival, it finds among the nodes of the tree the one which is closest to the new leaf and connects the latter to this node. Since the root is responsible of setting up the tree, this approach is called ”root initiated” approach. Besides resource optimization, RSVP-TE supports many TE features such as fast reroute including link, node and bandwidth

[1]The terms ”P2MP tree” and ”P2MP LSP” are used interchangeably throughout this paper.

protection, in addition to constraint-based routing, and Make-Before-Break rerouting. The second approach proposed to build P2MP LSPs, uses LDP as a signalling protocol [6]. In this approach, a leaf joins the P2MP tree by calculating the shortest path towards the root, yielding a Shortest Path Tree (SPT) and a "leaf initiated" approach. This is a very simple way to set up P2MP LSPs at the expense of no guarantee on resource optimization.

The comparison between RSVP-TE and LDP reveals that resource optimization property of the former necessitates a large amount of memory in the nodes of the tree. This is mainly caused by the size of Explicit Route Objects (EROs) that contain the identifiers (i.e., IP addresses or Autonomous System (AS) numbers) of all nodes that are on the path of downstream leaves. This is not needed when constructing an SPT using LDP, which ensures better scalability at the expense of obtaining higher tree cost than with RSVP-TE.

In this paper, we explore the possibility to extend LDP in order to minimize the cost of P2MP trees while conserving its scalability behavior. Several distributed heuristics are available in the literature [7], [8], [9]. However, such algorithms are not efficient in grafting since all nodes of the partial tree are involved in computing the best path towards a new leaf which yields a large number of message exchanges. In addition, the root takes the final decision on connecting the new leaf, and hence they are not "leaf initiated" algorithms. On the other hand, the TakaMat algorithm can be rendered distributed and "leaf initiated" by letting a new leaf have a global knowledge of the partial tree. However, this requires that each leaf has means to identify each node in the partial tree constructed so far which incurs a large number of node identifiers to be maintained in leaves. To avoid such limitation, we present two modified versions of a distributed and "leaf initiated" TakaMat heuristic by letting a new leaf have a partial knowledge of the partial tree. This partial knowledge may concern either the leaves or all nodes of the partial tree. These two alternatives are deeply explored and evaluated by using extensive simulations. The results are compared to those obtained in the case of RSVP-TE. The obtained results offer a valuable material to choose between the two protocols.

The remainder of this paper is organized as follows. In the next section we explain the actual extensions of RSVP-TE and LDP in order to support P2MP trees. Section 3. details our proposed extensions to LDP. In Section 4., the performance of our propositions are quantified by using extensive simulations. Finally, Section 5. concludes the paper.

2. RELATED WORKS

There are two Internet drafts which closely relate to this work. The first one [6] proposes an extension to LDP in order to support P2MP and Multi Point to Multi Point (MP2MP) LSPs. We are interested in P2MP LSPs. As mentioned in [6], leaf nodes initiate P2MP LSP setup and tear-down. Setup operation begins when a leaf node sends a Label Map message (LMAP) assigning a label to a Forward Equivalent Class (FEC) element. The FEC element contains the address of the root and an opaque value to uniquely identifying the P2MP LSP. The LMAP message is sent upstream by following a shortest path towards the root. A transit node checks if it has already a state for the FEC present in the LMAP and updates its forwarding state accordingly (more details about this process can be found in [6]). The example of Fig. 1 illustrates this approach. Let us consider that the P2MP tree has R_0 as root and $\{R_5,R_3,R_7\}$ as leaves. The FEC of this P2MP LSP is denoted as $<R_0, I>$, where I is an opaque value that is unique in the context of R_0. Dotted black arrows in Fig. 1 represent the shortest path from leaves to R_0. Suppose that leaves join the tree in the following order: R_5, R_3 and R_7 respectively. Upon the arrival of R_5, the latter sends an LMAP message towards its upstream R_4 by indicating the label it wishes to receive for the LSP identified by the FEC element. In this particular case we have: Label=L_5 and FEC=$<R_0, I>$. After sending the LMAP message, R_5 adds a state for the FEC $<R_0, I>$. When R_4 receives the LMAP message sent by R_5 it finds that no state exists for FEC $<R_0, I>$. R_4 must then send a new LMAP message towards R_1, with L_4 as a desired label, and must add a new state for FEC $<R_0, I>$ by swapping label L_4 to label L_5 as shown in Fig. 1. Transit node R_1 behaves like R_4. The LMAP message forwarding stops at root R_0 which adds a push state (label L_1 is pushed on an arrived packet). When R_3 joins the tree, the LMAP message forwarding stops at R_1 for the simple reason that the latter has already an existing state for FEC $<R_0, I>$. R_1 should only update its forwarding state as shown in Fig. 1. A new update in the state of R_1 is also required when R_7 joins the P2MP LSP.

It can be clearly observed from the above example that the amount of information, in terms of number of node identifiers to be maintained by a node, equals the number of downstream nodes plus one node identifier representing the upstream node. On one hand, an upstream node is the next hop in the shortest path towards the root and hence, without this information an LMAP message cannot be forwarded by a transit node. On the other hand, knowledge about downstream nodes is primordial in order to populate the forwarding states of the data plane. For instance, node R_1 in Fig. 1 must maintain four IP addresses (because it has three downstream nodes).

The second draft [3] defines extensions to RSVP-TE in order to support P2MP TE LSPs. Building such LSPs is made by two steps. First, the root sets up the P2MP tree by using an appropriate heuristic. Second, a P2MP LSP is regarded as a set of of multiple source-to-leaf (S2L) sub-LSPs each of which connects the root to a leaf by following

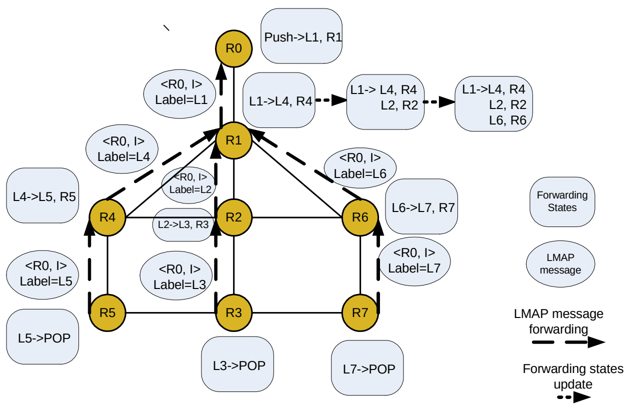

Figure 1. An example of P2MP set up under LDP.

the path calculated in the previous step. These S2L sub-LSPs, encoded in PATH messages, are appropriately combined by the branch nodes to result in a P2MP TE LSP. A PATH message contains several objects that can be classified into three main categories: 1) Identifying Objects (I-Os), 2) TE Objets (TE-Os) and 3) Routing Objects (R-Os). I-Os include, for instance, the session and sender template objects which are necessary to identify the P2MP LSP. TE-Os include bandwidth requirement objects such as Tspec and Adspec [10], the session attribute object, the fast reroute object [11], etc. R-Os may include timer and label objects in addition to Explicit Route Objects (EROs). The path computed by the root is encoded in EROs as a sequence of node identifiers, such as IP addresses. It can be observed that EROs are the only objects with a size proportional to the size of the multicast path. The other objets being of a constant size, one may consider that the size of a PATH message is in the order of the size of EROs.

The root may signal the S2L sub-LSPs belonging to a P2MP LSP in one PATH message or it may split them across multiple Path messages. In the first scenario, only one Path State Block (PSB), which contains a copy of the PATH message, must be maintained by a node. The PATH message carries a number of EROs that equals the number of downstream leaves. In the second scenario, a typical case is when each PATH message contains only one ERO that encodes the explicit route towards one leaf. In this case a node has to maintain a number of PSBs that is equal to the number of downstream leaves. The advantage of such scenario is that it minimizes extensions to RSVP-TE at the expense of increasing the number of PSBs per node. It should be noticed that in both scenarios a node will maintain the same number of node identifiers. In the following, only the first scenario is considered.

In [3] a compression mechanism that reduces the PATH message length is proposed. It relies on the elimination of redundant node identifiers in the EROs as much as possible (see Section 4.5 of [3] for an example). Figure 2 presents an illustrative example. As in Fig. 1, let us assume that the P2MP tree has R_0 as root and $\{R_5,R_3,R_7\}$ as leaves. However, suppose this time that the arrival order of leaves is: R_3, R_5 and R_7 respectively. Let the term "Partial Tree" (PT) accounts for the nodes of the P2MP tree when a new leaf decides to join it. At the beginning of the tree computation step, we have PT $= \{R_0\}$. When a leaf arrives, R_0 determines how to connect it to the PT by computing the shortest path from the leaf to PT. Call this path SP(leaf, PT). In the particular case of Fig. 2, the tree calculation begins when R_3 joins the tree where we have: SP(leaf, PT)=SP(R_3, R_0)=$R_0- >R_1- >R_2- >R_3$. When R_5 joins the tree, PT is equal to $\{R_0,R_1,R_2,R_3\}$. Hence, the result of calculating SP(R_5, PT) by R_0 is: $R_3- >R_5$, and the new PT equals $\{R_0,R_1,R_2,R_3,R_5\}$. The tree computation completes when R_7 joins the tree where we obtain: SP(R_7, PT)=$R_3- >R_7$, and PT=$\{R_0,R_1,R_2,R_3,R_5,R_7\}$. The tree computed this way follows TakaMat heuristic. The resulting tree path is shown in Fig. 2. Following this step, R_0 sends one Path message containing three EROs to R_1, its next downstream node on the tree. Each ERO depicts the explicit route towards one leaf. Each transit node sends a new Path message by including routing information to downstream nodes. Such information is refreshed and sent downstream periodically or when updates occur (i.e., it is a soft state). For instance, the number of node identifiers in the PSB of R_0 (Fig. 2) equals the number of nodes in the EROs of the PATH message, i.e., $3+4+4 = 11$ for the mode "without compression", and $3 + 2 + 2 = 7$ for the mode "with compression". When a Path message reaches a leaf, the latter replies by a Reservation (RESV) message containing a label and other TE-Os and I-Os and sends it to its upstream node. Thus, one more node identifier representing the upstream node must be added to the total number of node identifiers maintained by a node. Note that RESV messages are merged at branch nodes before being forwarded upstream (Fig. 3). To conclude this section, let us compare LDP and RSVP-TE in terms of cost and number of node identifiers kept by nodes, in the context of the examples of Figs 1 and 2. The cost of a tree T, $C(T)$, is defined as the sum of its links. This means that for LDP (Fig. 1) we have $C(T) = 7$, and for RSVP-TE (Fig. 2), we have $C(T) = 5$.

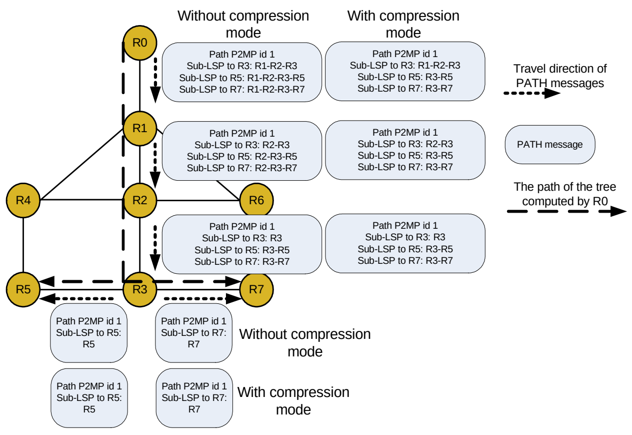

Figure 2. An example of P2MP set up under RSVP-TE: PATH messages.

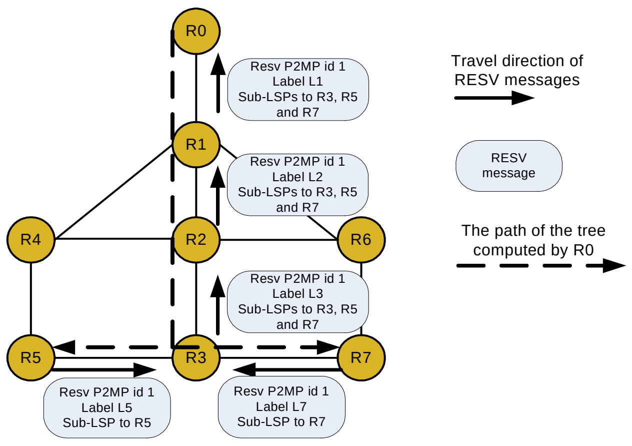
R0
Resv P2MP id 1 Label L1 Sub-LSPs to R3, R5 and R7
Travel direction of RESV messages
R1
Resv P2MP id 1 Label L2 Sub-LSPs to R3, R5 and R7
RESV message
R4
R2
R6
The path of the tree computed by R0
Resv P2MP id 1 Label L3 Sub-LSPs to R3, R5 and R7
R5
R3
R7
Resv P2MP id 1 Label L5 Sub-LSP to R5
Resv P2MP id 1 Label L7 Sub-LSP to R7

Figure 3. An example of P2MP set up under RSVP-TE: RESV messages.

Node	R_0	R_1	R_2	R_3
LDP	1	4	2	1
RSVP-TE (no compression)	12	9	6	3
RSVP-TE (compression)	8	7	6	3

Table 1. LDP vs RSVP-TE in terms of amount of information maintained by nodes

Table 1 depicts the number of node identifiers maintained in nodes R_0 through R_3, in both LDP and RSVP-TE cases. It can be observed that RSVP-TE enhances the cost of the tree at the expense of largely increasing the maximum number of node identifiers kept in nodes.

3. PROPOSED SOLUTIONS

The previous section has shown, on a simple example, that LDP scales better than RSVP-TE. A question arises about the scalability of LDP if it is extended to compute P2MP trees with a minimized cost. We propose the three following scenarios.

3.1. A Near Optimal Leaf Initiated Heuristic

To attain a cost minimization like RSVP-TE, a new leaf must be able to compute the shortest path towards the Partial Tree (PT). This requires that a leaf must identify all the nodes on the PT. Since each PE is capable of being a leaf, all PEs must have the same information at the same time. Refer again to the example of Fig. 2. Suppose that PE routers are R_0, R_3, R_5, R_6 and R_7. As in Fig. 2, suppose that the root is R_0, and the arrival order of leaves is R_3, R_5 and R_7. When R_3 joins the tree we obtain PT=$\{R_0,R_1,R_2,R_3\}$. The content of the PT must be delivered to R_5, R_6 and R_7 whether these PEs will join the tree in the future or not. This approach has two main drawbacks. First, the number of node identifiers in PE routers will be equal to the sum of the number of nodes on all P2MP LSPs in the network. Second, how to deliver such information to PEs is a very complex task. In the following, we let a PE have a partial knowledge of the PT and we propose two simple methods to deliver such information.

3.2. Extended LDP-Neighbors Knowledge Approach (eLDP-NKA)

In this method, as soon as a node (P or PE) becomes a member of a P2MP LSP, all nodes situated at $d \geq 1$ hop distance from it must be aware about this information. LDP should be extended to support a new type of message, called INFO message, which carries the information to deliver. INFO messages contain a Time To Live (TTL) value that equals d. In addition, they involve the FEC object to which the information applies. The following method is proposed to manipulate INFO messages.

When a node R joins a P2MP LSP, it sends an INFO message, containing its address, to all neighbors except those which are members of the same LSP. LDP provides means to know these neighbors as mentioned before. A node that receives an INFO message, updates its information database as appropriate, decrements the TTL by one and forwards the INFO message to all of its neighbors except that from which it received the message and those which belong to the P2MP LSP identified by the FEC element. The node identifies these LSP members thanks to previous INFO messages. The forwarding of the INFO message stops when the TTL becomes null. Figures 4 and 5 show an example for $d = 1$ and 2 respectively. In these two figures, the partial tree contains the path $PE_0- >PE_1- >P_3- >PE_3$ where PE_0 is the root and PE_3 is the only leaf (dotted arrows represent the forwarding direction of LMAP messages as in Fig. 1). Assume that $d = 2$ (Fig. 5) and suppose that at a later time, PE_5 decides to join the tree. By consulting its information database, PE_5 realizes that PT=$\{PE_0,PE_3\}$: PE_0 is the root of the tree and must be a priori known, and PE_3 is known by an INFO message as shown in Fig. 5. Hence SP(PE_5,PT) will be $\{PE_3- >PE_4- >PE_5\}$ rather than $\{PE_0- >PE_2- >P_4- >PE_5\}$ if the normal LDP procedure [6] is followed. It can be remarked that for $d = 2$ some nodes may receive the same information from multiple INFO messages. For instance, in Fig. 5, node P_1 receives the information P_3 twice, one from P_3 and one from P_2. One may solve such a problem by allowing a node that receives similar INFO messages for the same FEC to drop all but the one with the highest TTL value.

3.3. Extended LDP-Leaves Knowledge Approach (eLDP-LKA)

In P2MP LDP and RSVP-TE approaches ([6] and [3] respectively) node identifiers are maintained only on the nodes of P2MP LSPs (see Fig. 1 and Fig. 2). However, in eLDP-NKA, nodes that do not belong to P2MP LSPs may be involved in maintaining information (as for instance node P_1 in Fig. 5). To avoid this behavior, we propose the following scheme. When a leaf joins a P2MP LSP it should notice all PEs falling into the range of d hops about this information. This can be done by using the Border Gateway Protocol (BGP). Indeed, BGP enables PEs to inform Route Reflectors (RRs) nodes about their membership to P2MP LSPs [12]. RRs then use the deployed Interior Gatway Protocol (IGP) to determine all PEs that fall inside the d hop distance scope, and feed them with the membership information. We propose that RRs send this information periodically. This is shown in Fig. 6. As soon as PE_3 tells the RR about its membership, the latter sends this information via BGP to PE_4 and PE_5 which are into the scope of $d = 2$.

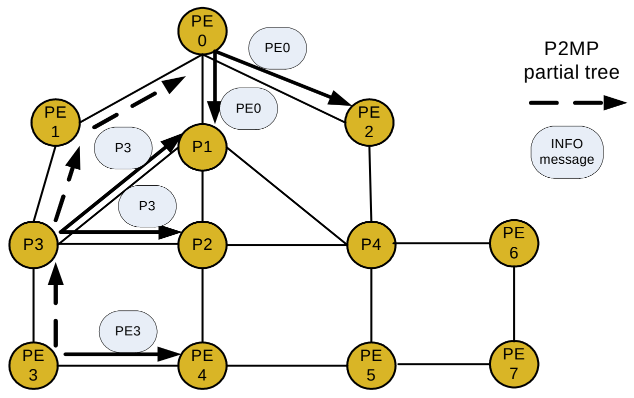

Figure 4. INFO messages forwarding for $d = 1$.

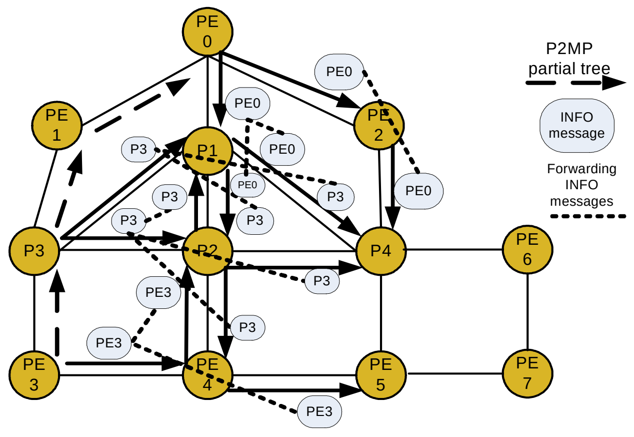

Figure 5. INFO messages forwarding for $d = 2$.

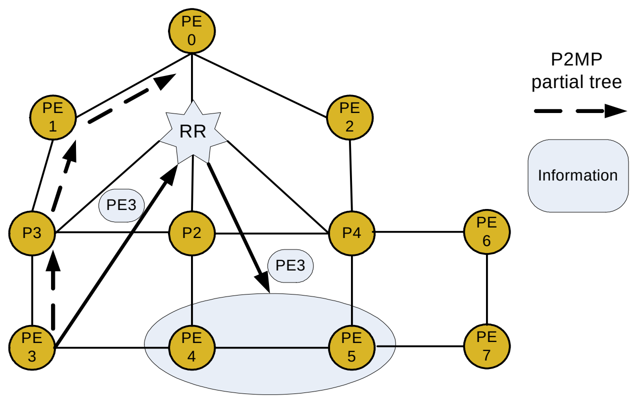

Figure 6. eLDP-LKA, $d = 2$, periodic model.

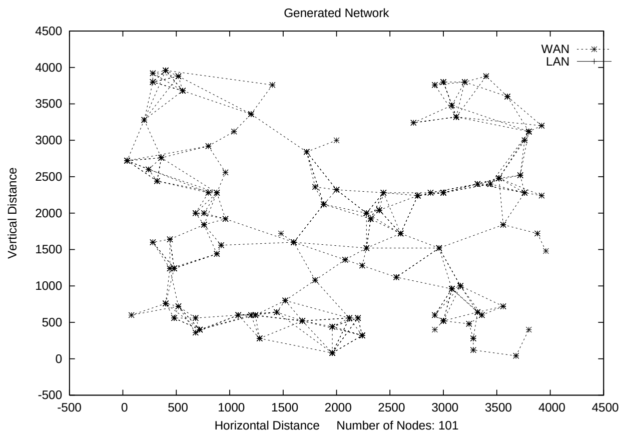

Figure 7. A mesh network instance.

In order to support this approach, the LMAP message must be appended to contain a new field representing the destination. Indeed, in the current P2MP LDP procedure, the destination of an LMAP message is always the root which is known by checking the FEC element. In eLDP-LKA, however, any PE can be the destination of an LMAP message, so this information must be provided. For instance, in Fig. 6, if PE$_5$ wishes to join the tree, the LMAP message destination should be PE$_3$ not PE$_0$.

4. PERFORMANCE EVALUATIONS

In this section we evaluate our propositions in regards of the optimization of the cost and the amount of information needed to maintain P2MP LSPs. Simulation results are obtained by using a simulator written in C. We used 10 network instances with a meshed topology generated by the Tiers topology generator [13]. Each instance has 101 nodes. The number of links for an instance is between 428 and 484. In each instance we assume that there are 80 PEs and 21 Ps distributed randomly around the network. Figure 7 provides the topology of one of the ten instances.

The simulation scenarios are organized as follows. For each topology, a P2MP tree with a variable number of leaves, starts from each PE in the network. This is a realistic scenario where there are as many P2MP LSPs as PEs. When comparing the different approaches, the cost represents the cumulated cost of all generated P2MP trees. Recall that the cost of a tree is the number of links it uses. On the other hand, to maintain each P2MP LSP, each node must keep a certain number of node identifiers. The amount of information stands for the maximal sum of the number of node identifiers needed to maintain all the generated P2MP LSPs in a PE. Figures representing the amount of information per P are not shown because they are very similar to those representing the amount of information per PE. Unless mentioned differently, the results represent the average over the ten instances.

4.1. LDP vs RSVP-TE

Figure 8 shows a cost comparison between LDP and RSVP-TE. As mentioned before, P2MP trees are built by LDP using a shortest path algorithm, and by RSVP-TE using TakaMat algorithm. The cost gain plotted in Fig. 8 is given by the following equation:

$$G^{RSVP-TE} = 100 \times \frac{C^{LDP} - C^{RSVP-TE}}{C^{LDP}} \quad (1)$$

where C^{LDP} and $C^{RSVP-TE}$ account for the total cost obtained in the case of LDP and RSVP-TE respectively. The gain for three different instances, in addition to the average gain over the 10 instances are shown. It can be observed that the cost gain offered by RSVP-TE (Takahashi) varies from one instance to another. In particular, for some instances ("instance 3" in Fig. 8), the gain may attain 19%. For the two other instances (instances 1 and 2), the maximum gain obtained is about 16%. The average gain presents similar behavior as the two latter instances.

The curves in Fig. 8 show that the gain of RSVP-TE attains a maximum value for a relatively low number of leaves (between 11 and 15 leaves), and then decreases as the number of leaves increases. In other words, RSVP-TE performs better than LDP as the number of leaves is low, and it tends to have cost values close to the latter for a large number of leaves. This is a very interesting result that can be explained as follows. A tree with a small size uses a

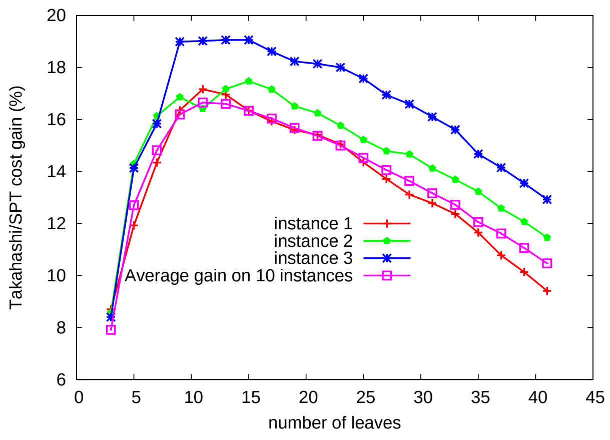

Figure 8. LDP vs RSVP-TE: cost comparison.

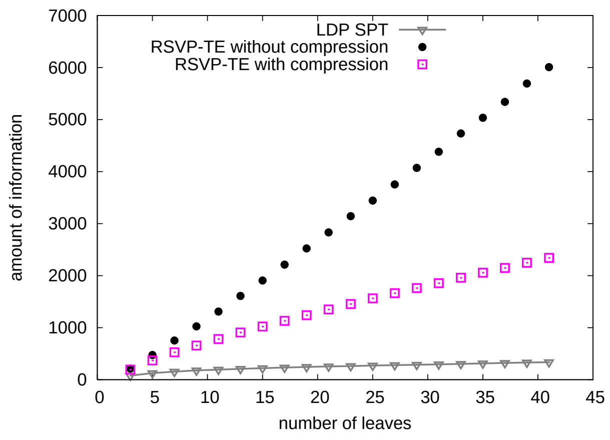

Figure 9. RSVP-TE vs LDP: max. amount of information per PE.

small number of links. In such case, it will be very probable that all links of the shortest path that connects a new leaf to the root do not belong to the tree constructed so far. This may largely increase the cost of an SPT with respect to the TakaMat algorithm. However, as the size of the tree increases, the cost of an SPT gets closer to that of TakaMat since the probability to have some links of the shortest path already belonging to the partial tree increases.

Figure 9 depicts the maximal number of node identifiers (amount of information) that should be maintained by a PE router in both LDP and RSVP-TE cases. In the latter case, the impact of the compression mode, discussed in Section 2. is shown. Clearly, a huge difference exists between LDP and RSVP-TE even in the case of the compression mode of the latter. For instance, when the number of leaves equals 21, a PE should maintain the identifiers of about 250 nodes in the case of LDP and 1500 nodes in the case of RSVP-TE with compression mode. As a conclusion, Figs. 8 and 9 indicate that as the number of leaves increases, the cost gain of RSVP-TE decreases while the amount of information continues increasing. Furthermore, it is clear that LDP scales better than RSVP-TE, and that it gets closer to the latter in terms of cost for a large number of leaves. The only drawback of LDP is observed at relatively low number of leaves where the gain of RSVP-TE may attain 19%. In the following, we evaluate our propositions aiming to ameliorate LDP in what concerns this drawback.

4.2. eLDP-NKA

Figure 10 illustrates the cost gain observed in the case of eLDP-NKA for $d = 1, 2$ and 3 respectively. The comparison with the gain of RSVP-TE is also shown. The amount of information (maximum number of node identifiers) in a PE is given in Fig. 11. It can be seen that for $d > 1$, the behavior of the cost gain is the same as that of RSVP-TE remarked in Fig. 8. Only for $d = 1$, the gain increases and then approximately stabilizes at about 6%. However, to achieve an amount of gain close to RSVP-TE, d must be set at least to 3 (see Fig. 10). As shown

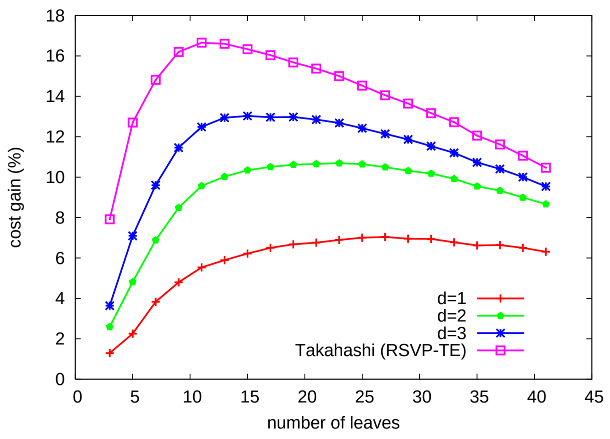

Figure 10. eLDP-NKA vs RSVP-TE and LDP: cost comparison.

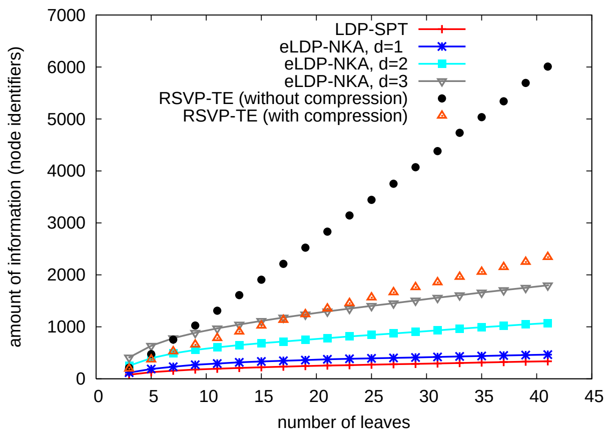

Figure 11. eLDP-NKA vs RSVP-TE and LDP: max. amount of information per PE.

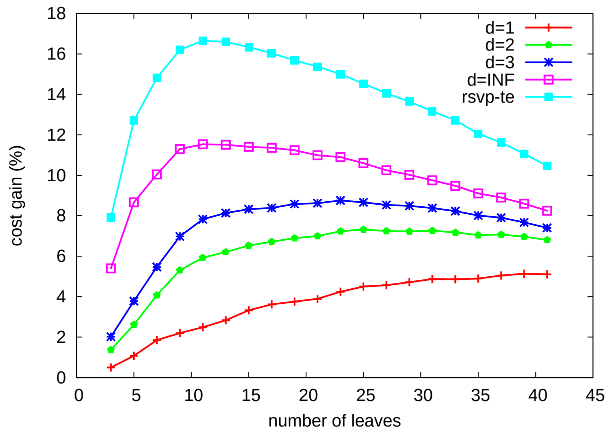

Figure 12. eLDP-LKA vs RSVP-TE and LDP: cost comparison.

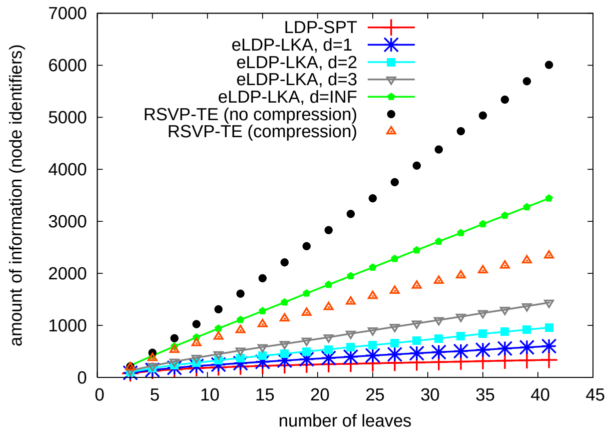

Figure 13. eLDP-LKA vs RSVP-TE and LDP: max. amount of information per PE.

in Fig. 11, this leads to the same amount of information observed in the case of RSVP-TE with compression or even worse for a relatively low number of leaves (i.e., less than 15). Figure 11 also shows that only for $d = 1$ the amount of information is very close to that obtained by LDP and preserves the same approximate constant behavior as the number of leaves increases.

As conclusion, eLDP-NKA preserves the scalability property of LDP for $d = 1$ only. At this value, the cost gain with respect to LDP achieves a maximum of about 6%. Although for $d > 1$ eLDP-NKA achieves a cost gain close to that of RSVP-TE, the amount of information (Fig. 11) becomes linear which leads to the same scalability problem of RSVP-TE.

4.3. eLDP-LKA

As in the previous section, in the case of eLDP-LKA the cost gain and the amount of information are evaluated. This is given by Fig. 12 and Fig. 13 respectively.

Again, Fig. 12 demonstrates that the gain of eLDP-LKA behaves like that of RSVP-TE unless for $d = 1$. For a number of leaves between 10 and 15 the gain obtained for $d = 1$ is about 3% only. This represents the half of the value obtained in the case of eLDP-NKA. This is intuitively explained by observing that in eLDP-NKA a new leaf has more knowledge about the partial tree than in eLDP-LKA. This also explains why the gain of eLDP-LKA remains less than that of RSVP-TE even for $d = INF$. $d = INF$ accounts for the case where all PEs in the network are aware about membership information of each others. Figure 13 shows that in this case, the amount of information becomes greater than that of RSVP-TE (with compression). Furthermore, for $d = 3$ the maximum gain is about half that of RSVP-TE with compression (Fig. 12). Figure 13 indicates that for $d = 3$ the amount of information is even more the half that of RSVP-TE with compression. Clearly, eLDP-LKA does not behave better than RSVP-TE.

5. CONCLUSION

We have presented a comparative study between LDP and RSVP-TE in terms of P2MP tree cost and the amount of information needed to maintain such tree in an MPLS environment. It has been shown that RSVP-TE achieves better cost in the case of P2MP trees with relatively low number of leaves. Indeed, as the number of leaves increases the cost gain of RSVP-TE with respect to LDP decreases. On the other hand, the amount of information observed in the case of RSVP-TE grows linearly as function of the number of leaves, while it remains approximately constant in the case of LDP. In other words, LDP scales better than RSVP-TE. In order to ameliorate the cost of LDP, we have proposed two approaches, called eLDP-NKA and eLDP-LKA respectively. Both approaches are based on the TakaMat heuristic. Results show that while eLDP-NKA may achieve a cost gain close to that of RSVP-TE, its scalability becomes comparable to the latter. Hence, it seems that it is irrelevant to investigate an extension to LDP in order to minimize the cost of P2MP trees.

The above observations have many consequences on the choice between LDP and RSVP-TE. Indeed, for a network operator that aims to deploy multicast applications with low number of leaves and traffic engineering support, the use of RSVP-TE is recommended. This is motivated by the non negligible resource optimization observed in this case in addition to the efficient traffic engineering features available in RSVP-TE. Namely, explicit routing, fast reroute, constraint-based routing and make-before-break rerouting. However, for multicast applications with a large number of leaves there are two possibilities. First, if traffic engineering support is not required, it should be better to use LDP since the optimization of RSVP-TE is not significant in this case in addition to the scalability problem of the latter. Second, if traffic engineering support is required, a network operator may use RSVP-TE for its important traffic engineering characteristics though it has a serious scalability problem in this case. Nevertheless, since LDP scales well for large number of leaves, it may also be used in this case. This is motivated by several works aiming to extend LDP in order to support some basic traffic engineering features such as make-before-break rerouting and fast reroute.

REFERENCES

[1] E. Rosen, A. Viswanathan, and R. Callon, "Multiprotocol Label Switching Architecture," RFC 3031, Jan. 2001. [Online]. Available: http://www.ietf.org/rfc/rfc3031.txt?number=3031

[2] L. Andersson and T. Madsen, "Provider Provisioned Virtual Private Network (VPN) Terminology," RFC 4026, Mar. 2005. [Online]. Available: http://www.ietf.org/rfc/rfc4026.txt?number=4026

[3] R. Aggarwal, D. Papadimitriou, and S. Yasukawa, "Extensions to RSVP-TE for Point-to-Multipoint TE LSPs," Internet Draft, Jan. 2007. [Online]. Available: http://www.ietf.org/internet-drafts/draft-ietf-mpls-rsvp-te-p2mp-07.txt

[4] P. Winter, "Steiner Problem in Networks: A Survey," *Networks*, vol. 17, no. 2, pp. 129–167, 1987.

[5] H. Takahashi and A. Matsuyama, "An Aproximate Solution for the Steiner Problem in Graphs," *Math. Japonica*, vol. 24, no. 6, pp. 573–577, 1980.

[6] I. Minei, K. Kompella, I. Wijnands, and B. Thomas, "Label Distribution Protocol Extensions for Point-to-Multipoint and Multipoint-to-Multipoint Label Switched Paths ," Internet Draft, June 2006. [Online]. Available: http://tools.ietf.org/html/draft-ietf-mpls-ldp-p2mp-02

[7] F. Bauer and A. Varma, "Distributed Algorithms for Multicast Path Setup in Data Networks," *IEEE/ACM Trans. Networking*, vol. 4, no. 2, pp. 181–191, Apr. 1996.

[8] X. Jia, "A Distributed Algorithm of Delay-Bounded Multicast Routing for Multimedia Applications in Wide Area Networks," *IEEE/ACM Trans. Networking*, vol. 6, no. 6, pp. 828–837, Dec. 1998.

[9] J. Rugelj, "Distributed Multicast Routing for Global Point-to-Point Networks," in *Proceedings of the 20th EUROMICRO Conference*, Sept. 1994, pp. 389–395.

[10] R. Braden, *et al.*, "Resource ReSerVation Protocol (RSVP)–Version 1 Functional Specification," RFC 2205, 1997. [Online]. Available: http://www.ietf.org/rfc/rfc2205.txt?number=2205

[11] P. Pan, G. Swallow, and A. Atlas, "Fast Reroute Extensions to RSVP-TE for LSP Tunnels," RFC 4090, May 2005. [Online]. Available: http://www.ietf.org/rfc/rfc4090.txt?number=4090

[12] E. Rosen and Y. Rekhter, "BGP/MPLS VPNs," RFC 2547, Mar. 1999. [Online]. Available: http://www.ietf.org/rfc/rfc2547.txt?number=2547

[13] [Online]. Available: http://www.isi.edu/nsnam/ns/ns-topogen.html