# Asymmetric Tunnels in P2MP LSPs as a Label Space Reduction Method

F. Solano
Institut d'Informàtica i Aplicacions
Universitat de Girona, Spain.
fsolanod@eia.udg.es

R. Fabregat
Institut d'Informàtica i Aplicacions
Universitat de Girona, Spain.
ramon@eia.udg.es

Y. Donoso
Departamento de Ingeniería de Sistemas y Computación,
Universidad del Norte
Barranquilla, Colombia.
ydonoso@uninorte.edu.co

J.L. Marzo
Institut d'Informàtica i Aplicacions
Universitat de Girona, Spain.
marzo@eia.udg.es

***Abstract* - Traffic Engineering objective is to optimize network resource utilization. Although several works have been published about minimizing network resource utilization, few works have been focused in LSR label space. This paper proposes an algorithm that gets advantage of the MPLS label stack features in order to reduce the number of labels used in LSPs. Some tunnelling methods and their MPLS implementation drawbacks are also discussed. The described algorithm sets up the NHLFE tables in each LSR creating asymmetric tunnels when possible. Experimental results show that the described algorithm achieves a great reduction factor in the label space. The presented works applies for both types of connections: P2MP and P2P.**

***Keywords* – Asymmetric tunnels, label space, label stack, label space reduction, longest segment first, NHLFE, traffic engineering, MPLS.**

## I. INTRODUCTION

Traffic engineering (TE) is concerned about improving performance of operational networks usually considering quality of service (QoS) requirements. The main objective is to reduce congestion hot spots, improve resource utilization and provide adequate QoS for final users. These can be achieved by setting up explicit routes over the physical network in such a way that the traffic distribution is balanced across several traffic trunks, giving the best possible service [1].

Multi Protocol Label Switching (MPLS) aims to work with these TE schemes by setting up label switched paths (LSPs) as needed to transmit efficiently Internet Service Provider's (ISP) customer's flows with their requirements. Customer requirements are flow dependent, i.e. delay, packet loss, jitter, etc. Although this can be achieved in many ways using different algorithms [2], ISPs must be aware of label switched router (LSR) internal resources utilization such as the label space.

Each time a LSP is established, all the LSR that belongs to it must use a label in order to identify the LSP transiting by it, and consequently every packet of this LSP must carry this label encoded inside it when arriving at that LSR. When a packet is received by a LSR, the LSR must look for the packet label and then search for a Next Hop Label Forwarding Entry (NHLFE) that refer to this label in order to decide which interface will be used to reach the next hop in the network [3]. Clearly, the more LSPs a LSR support, the more NHLFEs will exist.

Reference [3] establishes that each label must be encoded in a 20-bits field, allowing only $2^{20}$ (1.048.576) possible different labels in a LSR. Despite this is a sufficient number for label encoding in a single LSR, large enough NHLFE could cause long delays while a LSR looks up in its *forwarding table* for the next hop LSR each time a packet is received. Therefore, a smaller forwarding table will reduce LSR memory requirements and aids LSR to forward packets faster [4], [5] [6].

Now, considering Multi Protocol Lambda Switching (MPλS), this problem achieves a greater magnitude. The MPLS label space is comparatively large (one million per port), whereas there is a relatively limited number of lambdas and Time-division Multiplexing (TDM) channels.

To support LSP tunnelling, MPLS defined a label stack for packets [7] and some stack operations set inside NHLFEs [3]. These operations are: a) replace the label at top for a new one (label swapping), b) pop the stack, c) replace the label at top for a new one and then push one or more onto stack.

Although IETF have not decide yet how to set up Point-to-MultiPoint (P2MP) LSPs in MPLS, this paper proposes an algorithm that uses the MPLS label stack in a different way to reduce the label space and hence improve the way MPLS uses NHLFE in P2MP LSPs. The terminology used in this document is the one established in [8].

This work has been organized as follows. Some studies about label space reduction and label stack size are discussed in section II. Several label space reduction techniques are explained in section III together with the asymmetric tunnel concept. An asymmetric tunnelling algorithm for P2MP LSPs using MPLS label stack is described in section IV. Section V shows some simulation results with different topologies and randomly generated P2MP requests. Finally, conclusions and further studies are presented in section VI.

This work was partially supported by the MCyT under the project TIC2003-05567.

The work of Yezid Donoso was supported by the Universidad del Norte (Colombia) under contract G01 02-MP03-CCBFPD-0001-2001.

The work of Fernando Solano was supported by the Ministry of Universities, Research and Information Society (DURSI) of the Government of Catalonia under contract 2004FI-00693 and 2005FIR-00379.

This work has been focused in P2MP connections since they can be seen as a general version of P2P connections; thus, the presented works applies for both types of connections.

## II. Label Space Reduction Methods

The MPLS architecture allows aggregation in Point to Point (P2P) LSPs. Aggregation reduces the number of labels that are needed to handle a particular set of flows, and may also reduce the amount of label distribution control traffic needed [3].

With aggregation, [3] refers to a MultiPoint-to-Point tree (MP2P) created by merging many P2P LSPs, i.e. a tree rooted at an egress LSR and has ingress LSRs as leaves. In other words, if two P2P LSPs follow the same path from an intermediate LSR to the egress LSR, this method allocates the same label to both P2P LSPs and thus reduce the number of used labels. In this case, labels assigned to different incoming links are merged into one label assigned to an outgoing link. Fig. 1 shows different P2P connections in which a single MP2P is established between three ingress LSRs {N1, N5, N8} and the egress LSR N11.

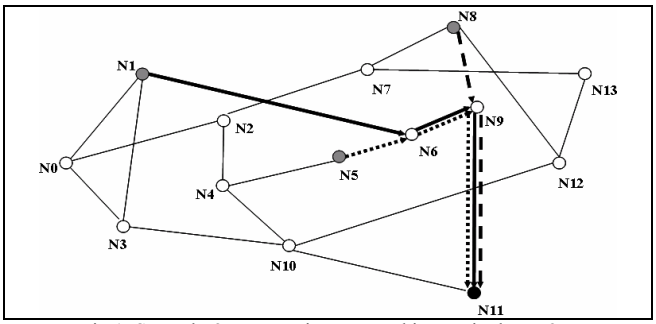

Fig 1. Several P2P connections merged into a single MP2P.

In the downstream to upstream label assignation process [3], i.e. downstream LSR assign to previous LSR (upstream LSR) its outgoing label, N9 confers the same label L1 to N8 and N6 (see section 3.14 of [3] for more information).

In [5], [9] and [10], algorithms that find P2P LSPs which can be merged into a minimal number of MP2P LSPs are considered. Reference [10] proves an upper bound of N (number of nodes) + M (number of links) for the label space. In these works ([5], [9] and [10]), note that minimising the label space is a base criterion to find out LSP' routes. Because ISP's customer's requirements are often measured as QoS requirements (flow dependent), we think that the label space should not be considered as an objective function (at most a model restriction) in any optimisation model that deals with finding LSP's path.

To reduce the number of used labels for multicast traffic, another label aggregation algorithm is presented in [6]. In this case, if two P2MP LSPs follow *entirely* the same tree from ingress LSR to the egress LSR set, the aggregation algorithm allocates the same labels to both P2MP LSPs. The algorithm can not reduce the number of labels when equals sub-P2MP trees[1] are considered.

In [4], a comprehensive study of label size versus stack depth trade-off for MPLS routing protocols on P2P connections is undertaken. They show that, in addition to LSP tunnelling, label stacks can also be used to reduce the number of labels required for setting up LSPs in a network using a special coding technique; and proved some label space upper bounds under certain type of conditions. Gupta, Kumar and Rastogi [4] inferred a lower and upper bound for two basic problems: (1) FIXED STACK ROUTING: Given a bound on the stack depth, minimize the number of labels used, and (2) FIXED LABEL ROUTING: Given a fixed number of labels, minimize the stack depth in P2P connection, but never proposed and algorithm which set ups the LSR forwarding table using the label stack when a set of LSPs are given.

It should be pointed out that so far we have not found an algorithm that *only* set ups LSR forwarding tables (i.e. no path finding algorithm) in order to minimise the number of labels by using the label stack in P2MP connections. Even more, to date, there is no literature about the novel asymmetric tunnel concept.

## III. Asymmetric Tunnels as a Label Space Reduction Method

To illustrate these label space reduction methods (aggregation and LSP tunnelling using label stack), suppose that $P2MP_1$ and $P2MP_2$ are two trees (see Fig. 2) that can be established on the NSF network.

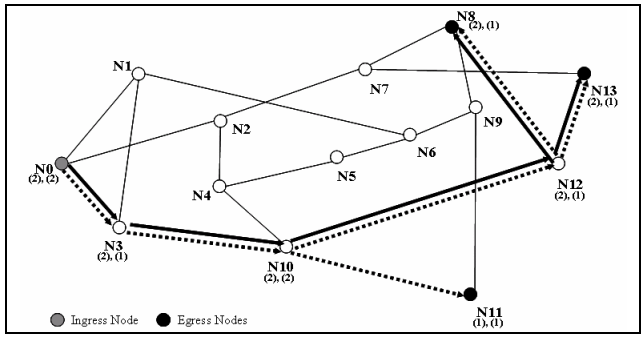

Fig 2. Two P2MP aggregated at N10-N12-{N8, N13} and "stacked" at N0-N3-N10.

As $P2MP_1$ and $P2MP_2$ have equal sub-P2MP tree starting at N10 and ending at {N8, N13} through N12, the aggregation scheme can be used and therefore a single label is needed in this sub-P2MP tree. Although {N0→N3→N10} is a path used by both P2MPs trees, previous aggregation scheme can not be used here because it will cause either N10 forwards $P2MP_2$ packets to N11 (i.e. packets duplication), or N10 stops forwarding $P2MP_1$ packets to N11 (i.e. multicast incomplete replication). To reduce the label space the label stack scheme can also be used. In this case, N0 can push a label into $P2MP_1$ and $P2MP_2$ packets stack and this label can be popped when packets reach N10. Using these two reduction methods in the example, the total amount of labels in the network is dropped off from 13 to 9.

The LSR tables of the example above can be summarised as follows (Table I) when both reduction methods are applied.

Because the labels are upstream assigned, what should be regarded to minimise is the number of incoming labels per LSR, e.g. the number of incoming labels for N3, N11, N12 is 1 and for N10 is 2. Also, note that as the number of incoming labels is reduced, the number of NHLFEs is reduced too.

TABLE I
ACTIVE LSR'S NHLFES FOR FIGURE 2

| LSR | Incoming Label | Outgoing LSR | Operation |
|---|---|---|---|
| N0 | $P2MP_1$ | N3 | Push L1, L0 |
| | $P2MP_2$ | N3 | Push L2, L0 |
| N3 | L0 | N10 | Pop |
| N10 | L1 | N11 | Swap L4 |
| | | N12 | Swap L3 |
| | L2 | N12 | Swap L3 |
| N11 | L4 | N11 | $Pop^T$ |
| N12 | L3 | N8 | Swap L5 |
| | | N13 | Swap L6 |

We will refer as a P2MP configuration as a set of P2MP LSPs that should be configured on a given topology. The problem of finding a near-optimal label space reduced solution is not trivial since it can be achieved in many ways.

First at all, branch nodes will be discussed since they are discarded as a tunnel member in our solution. Each time a branch LSR needs to forward a packet, in order to assure P2MP LSPs consistency, the LSR should swap more than once the label of the incoming packet for those incoming labels of the downstream LSRs. Because a LSR can not swap a label that is not at the top of the stack, branch LSRs could not be a member of any tunnel since it can not assign to the stacked LSPs of a tunnel the correct label in the replication process.

As an example, consider P2MP configuration in figure 3 with 2 P2MP LSPs and the weak P2MP tunnel in figure 4 which stacks both P2MP LSPs. Without a tunnel, figure 3, LSR N10 will replace an incoming label by two different outgoing labels in order to assure correct packet forwarding to N12 and N11.

In figure 4, LSR N10 forwards both tunnelled P2MP LSPs by swapping the top label but not the stacked label, hence LSR N13 and LSR N9 should receive packets with the same labels, e.g. Lx and Ly in figure 4.

Unless the architecture is changed in order to get an agreement between many downstream LSRs about their incoming label, this weakness leads us to consider only P2P branches in tunnels. To make an easier explanation of other tunnelling techniques, only 5 LSRs of NSF network are considered (see figure 5) and 3 P2MP LSPs are contemplated. All P2MP LSPs forward packets from N0. In a common MPLS NHLFE set up procedure, each LSR allocates space for 3 incoming labels in memory, since no label space reduction scheme is considered.

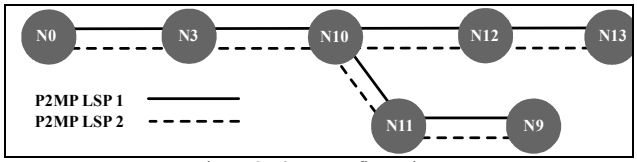

Figure 3 P2MP configuration.

TABLE II
ACTIVE LSR'S NHLFES FOR FIGURE 3

| LSR | Incoming Label | Outgoing LSR | Operation |
|---|---|---|---|
| N0 | $P2MP_1$ | N3 | Swap L3A |
| | $P2MP_2$ | N3 | Swap L3B |
| N3 | L3A | N10 | Swap L10A |
| | L3B | N10 | Swap L10B |
| N10 | L10A | N11 | Swap L11A |
| | | N12 | Swap L12A |
| | L10B | N11 | Swap L11B |
| | | N12 | Swap L12B |
| N11 | L11A | N9 | Swap L9A |
| | L11B | N9 | Swap L9B |
| N12 | L12A | N13 | Swap L13A |
| | L12B | N13 | Swap L13B |
| N9/13 | … | … | $Pop^T$ |

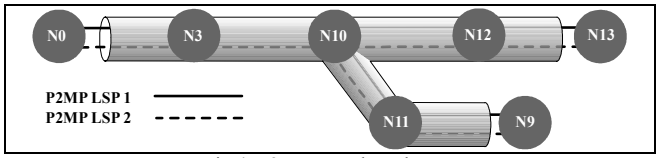

Fig 4. P2MP tunnel weakness.

TABLE III
ACTIVE LSR'S NHLFES FOR FIGURE 4

| LSR | Incoming Label | Outgoing LSR | Operation |
|---|---|---|---|
| N0 | $P2MP_1$ | N3 | Push L3,Lx |
| | $P2MP_2$ | N3 | Push L3,Ly |
| N3 | L3 | N10 | Swap L10 |
| N10 | L10 | N11 | Swap L11 |
| | | N12 | Swap L12 |
| N11 | L11 | N9 | Pop |
| N12 | L12 | N13 | Pop |
| N9/13 | Lx/Ly | … | $Pop^T$ |

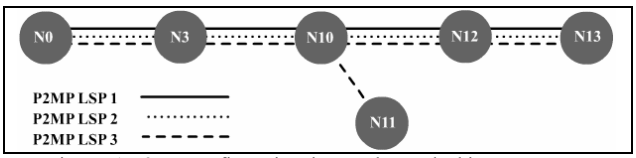

Fig. 5. A P2MP configuration that can be stacked in many ways.

It is clear that the sub-P2MP tree {N10→N12→N13} is the same from here on and therefore can use a single label since the P2MP configuration. Preceding sub-P2MP tree {N0→N3→N10} cannot use the same label because N10 should multicast packets for LSP3 through the sub-P2MP tree {N10-N11}. This lead us to an initial solution where the sub-P2MP tree {N0→N3→N10} can be stacked and the sub-P2MP tree {N10→N12→N13} can be aggregated (figure 6). Despite the fact that N10 and N12 use a single label to forward the

P2MP configuration, N10 should be still using 3 incoming labels for the configuration.

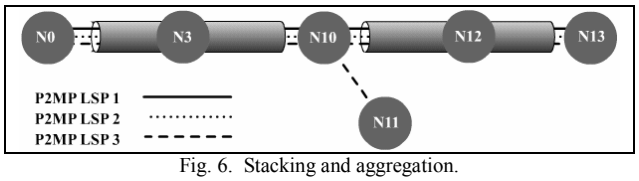


Fig. 6. Stacking and aggregation.

A more complex and efficient solution can be contemplated if nested tunnels are considered, i.e. tunnels that are within another. Figure 7 shows a solution where LSP1 and LSP2 are stacked across all the topology and LSP3 is stacked in the sub-P2MP tree {N0→N3→N10} and the sub-P2MP tree {N10→N12→N13}. Unfortunately this solution can not be set in MPLS because it does not allow *multiple popping* of stacked labels in a NHLFE [3], i.e. popping of more that one label by a LSR, as N3 and N12 should do.

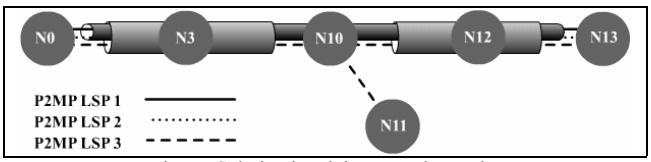


Fig. 7. Solution involving nested tunnels.

A similar MPLS solution that solves this drawback can be regarded as an asymmetric tunnel (in figure 8). The asymmetric tunnel concept comes from the idea that *not all* the stacked LSPs are tunnelled along all LSRs. .

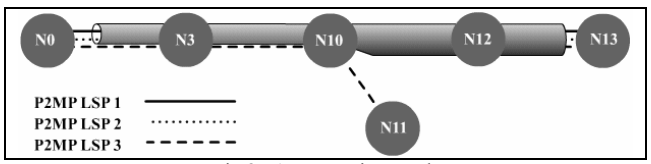


Fig 8. Asymmetric tunnel.

In the example, N10 stacks LSP3 by pushing the same label pushed before by N0 to LSP1 and LSP2. Here, N3 and N10 use 2 incoming labels, but N12 only uses 1. Since there is no way a LSR can look labels behind the top, all LSP must be *unstacked* at the same time and therefore, asymmetric tunnels will be usually 'bigger' at the end than at the beginning.

Next section will present an algorithm that makes tunnels in a P2MP configuration by selecting the longest P2P branch.

## IV. LONGEST SEGMENT FIRST

Consider a P2MP configuration as a set *P2MP* of P2MP LSPs. For a m ∈ *P2MP* consider a P2P decomposition d(m): (m ∈ *P2MP* → {u(i,j) ∈ P2P}) in which each element $u_{(i,j)}$ is a P2P LSP that connects a subset of LSRs of the P2MP LSP starting at LSR i (an ingress LSR, bud LSR or branch LSR) and ending at LSR j (an egress LSR, bud LSR or branch LSR). In Fig 2, the P2MP LSP that connects the ingress node N0 with egress nodes N11, N8 and N13 can be decomposed in 5 P2P LSP: $u_{(0,10)}$, $u_{(10,11)}$, $u_{(10,12)}$, $u_{(12,8)}$ and $u_{(12,13)}$. It is clear that this decomposition is unique and easy to find.

Let |u(i,j)| be the number of LSRs that u(i,j) uses to forward the information. The intersection of two P2P LSPs, $u_{(i1,j1)}$ and $u_{(i2,j2)}$, is the longest $u_{(i,j)}$ that is contained in both. For example, in figure 1, $u_{(1,11)} \cap u_{(5,11)} = u_{(6,11)}$.

The difference between two P2P LSP, $u_{(i1,j1)}$ and $u_{(i2,j2)}$ where $u_{(i1,j1)} \subseteq u_{(i2,j2)}$, are two sub-P2P LSP: one starting at $i_2$ and ending at $i_1$ and the second starting at $j_1$ and ending at $j_2$.

Table V describes, using the notation explained before, a procedure to find a set of sub-P2P LSPs, $T \subseteq P2P$, that can be tunnelled with a single label.

TABLE IV
ALGORITHM TO FIND P2MP LSP TUNNELS.

| | |
|---|---|
| 1 | $U = \{u_{(a,b)} \mid \forall m \in P2MP, u_{(i,j)} \in \cup d(m), \lvert u_{(a,b)} \rvert \geq 3\}$ , and $W = \phi$ |
| 2 | Find a P2P tunnel $t_{(i,j)}$ such that $\max\lvert t_{(i,j)} \rvert, t_{(i,j)} = u_{(a_1,b_1)} \cap u_{(a_2,b_2)}, u_{(a_1,b_1)} \in U, u_{(a_2,b_2)} \in U$ |
| 3 | If such tunnel was not found, stop. |
| 4 | Let $U' = \phi$ |
| 5 | For each $u_{(a,b)} \in U$ do |
| 6 | Find $u'_{(k,j)} = u_{(a,b)} \cap t_{(i,j)}$ |
| 7 | If $\lvert u'_{(k,j)} \rvert < 3$ then |
| 8 | $U' = U' \cup \{u_{(a,b)}\}$ |
| 9 | Else |
| 10 | $W = W \cup \{(t_{(i,j)}, u_{(a,b)})\}$ |
| 11 | $U' = U' \cup \{u_{(a,b)} - u'_{(k,j)}\}$ |
| 12 | End if |
| 13 | Repeat |
| 14 | Let $U = U'$ |
| 15 | Repeat from 2 |

In line 1, a set named *U* is created which contains all the P2P LSPs that are part of any P2MP LSP decomposition. Because a tunnel can not be done with less than 3 LSRs, each P2P LSP in *U* should satisfy this constraint. *U* will be our working set. *W* will be a mapping set of tunnelled P2P LSPs, initialised as empty.

Line 2 finds out a maximum length P2P LSPs, $t_{(i,j)}$, that intersects at least two P2P LSPs in *U*. The algorithm iterates only if this tunnel is found. In all iterations, a new working set *U'* is computed because each tunnel found *stacks* several P2P LSPs in *U*. Line 4 initialize this set as empty.

To compute the new working set, each P2P LSP in U is regarded in order to see whether it can be aggregated using this tunnel or not. A P2P LSP $u_{(a,b)}$ can be stacked using this P2P LSP tunnel $t_{(i,j)}$ if a both LSP intersects in more than 3 LSRs, $u'_{(k,j)}$. In order to assure asymmetric tunnels, the intersected LSP should include the last LSR of the tunnel $t_{(i,j)}$. Line 8 safe

$u_{(a,b)}$ in the new working set if no intersection could be found. Otherwise, in line 11 non intersected sub-P2P LSP are included in the new working set and $u_{(a,b)}$ is included in *W* as part of the algorithm response.

Note that to build a tunnel, the penultimate LSR, $j - 1$, LSR must do a POP in the stack, the first LSR, *i*, must do a PUSH of two labels and all intermediates LSRs, from $i + 1$ to $j - 2$, must do a SWAP.

## V. EXPERIMENTAL RESULTS

The algorithm in the preceding section was tested in two topologies with several P2MP configurations. The network topologies used are square-like, in which each node is placed in the cross-points of a rectangular grid of X rows and Y columns and each node at position (x,y) in the grid has connecting links to the next column (x,y+1), the upper row (x+1,y), and the lower row (x-1,y) when possible. In one of the tested network topology X=5 and Y=10 (50 nodes x 125 links). In another tested network topology X=10 and Y=10 (100 x 270).

For each experiment on both topologies, a set of randomly generated P2MP LSPs is created. Each P2MP LSP connects an ingress LSR with a set of 5 egresses LSRs. The ingress LSR is chose randomly from the first 5 LSRs in the grid. The 5 egresses LSRs are selected randomly from the last 10 LSRs in the grid. Once the ingress and egresses LSRs are picked, the tree is built by selecting a random path for every egress LSRs. Finally, redundant segments are deleted.

To evaluate the performance of the solution presented here, it should be stated that the number of labels will increase as the network load increases, measured as the number of P2MP LSPs in the network. Each time a simulation was ran, the reduction factor was computed as the relation between the amount of labels been dropped-off using tunnels and the number of labels used without tunnelling.

Figure 9 shows that the reduction factor experienced in both topologies.

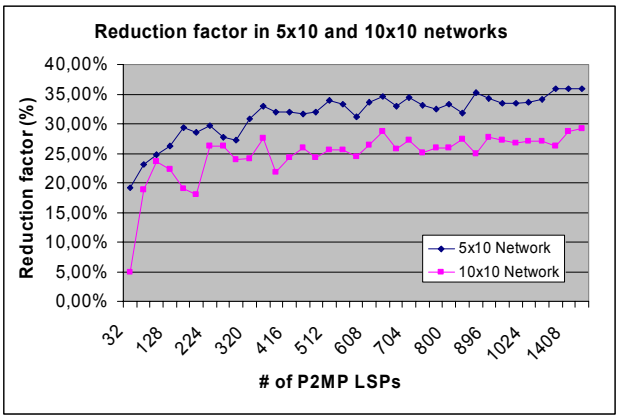


Fig. 9. Reduction factor in two tested networks.

Both topologies reach a stable point when 400 P2MP LSPs approximately are considered. Since the 10-10 network is larger that 5-10 network and there are more ways to reach a destination, the number of intersected LSPs is less and hence each tunnel will deal with less P2MP LSPs. In the 5-10 network the reduction factor reaches a 32.5% when it gets stable. In the 10-10 network this factor is about 27.5%. Therefore, the reduction factor depends on the network topology, the P2MP configuration, and the network load.

## VI. CONCLUSIONS AND FURTHER STUDIES

The presented work stated that the number of used labels can be dropped dramatically using tunnels in a P2MP configuration. Among all the discussion presented here, it should be noticed that there are many ways to do tunnels in P2MP connections but the better ones are far away from a feasible implementation with the current IETF standard. Despite this fact, the novel asymmetric tunnel concept was discussed and tested in some network topologies with several P2MP configurations using the *longest segment first* algorithm satisfactorily. The simulation of this algorithm showed that the reduction factor is dependent on network topology, the P2MP configuration, and especially on the network load.

Asymmetric tunnels solutions, described here, did not contemplate the aggregation feature in P2P connections. Moreover, P2MP aggregation, which is beyond the scope of this paper, presents a good reduction factor. It is possible to merge these ideas with the one presented here as further work in order to achieve better results. Since there are many ways to create asymmetric tunnels, it is also desired to find an optimization model which finds the best way to create asymmetric tunnels; our next goal. Also, an algorithm to create tunnels for on-line requests could be considered for future study as an extension for RSVP-TE P2MP [11].

## REFERENCES


[1] Z. Wang. Internet QoS: Architectures and Mechanisms for Quality of Service. Morgan Kaufmann Publishers. ISBN 1-55860-608-4.

[2] Y. Donoso, R. Fabregat, F. Solano and J.L. Marzo. "Generalized Multiobjective Multitree model for Dynamic Multicast Groups". IEEE ICC 2005.

[3] E. Rosen, A. Viswanathan, R. Callon. "Multiprotocol Label Switching Architecture". RFC 3031. January 2001.

[4] A. Gupta, A. Kumar, R. Rastogi. "Exploring the trade-off between label size and stack depth in MPLS Routing". INFOCOM'03.

[5] S. Bhatnagar, S. Ganguly, B. Nath. "Creating multipoint-to-point LSPs for traffic engineering: NP-completeness and heuristics". Workshop oh High Performance Switching and Routing. June 2003.

[6] Y.K. Oh, D.K. Kim, H.J. Yoen, M.S. Do, J. Lee. "Scalable MPLS multicast using label aggregation in Internet broadcasting systems". ICT'03.

[7] E. Rosen, et al. "MPLS Label Stack Encoding". RFC 3032. January 2001.

[8] S. Yasukawa, et al. "Signaling Requirements for Point to Multipoint Traffic Engineered MPLS LSPs". IETF Draft. December 2004.

[9] H. Saito, Y. Miyao, M. Yoshida. "Traffic engineering using multiple multipoint-to-point LSPs". INFOCOM 2000

[10] D. Applegate, M. Thorup. "Load optimal MPLS routing with N+M labels". INFOCOM'03.

[11] R. Aggarwal, D. Papadimitriou, S. Yasukawa. "Extensions to RSVP-TE for Point to Multipoint TE LSPs". November 2004.