This document is published in:

IEEE 13th International Conference on High Performance Switching and Routing (2012), pp. 133-140

DOI: 10.1109/ICC.2013.6655142

On Forwarding State Control in VPN Multicast Based on MPLS Multipoint LSPs

Gonzalo M. Fernández[1], David Larrabeiti[2], Juan A. de la Fuente,
Department of Telematics Engineering
Universidad Carlos III de Madrid
Madrid, Spain
gfernandezdc@ucsp.edu.pe[1], dlarra@it.uc3m.es[2]

***Abstract*— The demand for multicast-capable VPN services, like Virtual Private LAN Service (VPLS), has grown quickly in the last years. In order to save bandwidth, MPLS point-to-multipoint LSPs could be used, but the VPN-specific state information to be handled inside the network may exceed the capacity of core nodes. A well-known solution for this is to aggregate the multicast/broadcast traffic of multiple VPNs into shared p2mp LSP trees. In shared trees, although some bandwidth is wasted because a fraction of the packets are delivered to non-member leaves (either not in the VPN broadcast or multicast group), there is wide working range where a good state vs. bandwidth trade-off is achieved.**

In this paper we enhance and improve previous works that analyze this trade-off. We propose new techniques for multicast traffic aggregation of VPNs in MPLS-based networks, with the objective of observing the behavior of the aggregation philosophy for different aggregation degrees, which should be very useful for network design and deployment purposes. We assess the aggregation heuristics over different reference networks and VPN geographic distributions. Simulations give a quantitative indication of the relevance of intelligent aggregation, of geographical distribution and group sizes.

***Index Terms*— VPN, MPLS, aggregation, multicast, VPLS**

I. Introduction

Among the technologies to deliver the VPN service, Multi-Protocol Label Switching (MPLS) is one of the most popular, due to the capability to realize traffic engineering and the compatibility with connection-oriented frame relay and ATM networks, optical networks (with generalized MPLS), and any layer-2 mechanism. One weakness of MPLS VPNs with respect to carrier-grade Ethernet is the multicast feature. Until recently, the standards for MPLS-based Virtual Private Network (VPN) service implementation provided only point-to-point delivery, and multicast could only be deployed by means of customer tunnels. However, the need for multipoint support inside the Service Provider networks soon became clear with the advent of Virtual Private LAN Services (VPLS), where the service provider (SP) must emulate a shared layer-2 broadcast/multicast network [1].

The Internet Engineering Task Force (IETF) has made notable efforts to provide solutions for multicast MPLS VPN communications [1-4]. The support of multicast for L3VPN [3] is at present in the Standards Track at IETF and soon will enhance RFC4364 (BGP/MPLS IP VPNs). And the same evolution has followed the work in multicast for VPLS: Internet Draft [1] has just entered the standards track and will soon become RFC. As noted in that work, one of the limitations of the existing VPLS implementation of multicast in RFC4761 and RFC4762 is that they rely on ingress replication. This means that the ingress Provider Edge (PE) replicates the multicast packet for each egress PE and sends them to the egress PEs using several unicast tunnels. Ingress replication may be an acceptable model when multicast traffic is low or/and the number of replications is small. Otherwise [1] recommends the use of multicast trees to distribute VPLS multicast packets [5] for its inherent bandwidth saving. This multipoint service would be alternative or complementary to unknown MAC address unicast packet flooding to egress PEs (before the ingress PE learns the destination MAC address of those unicast packets) that may still use ingress replication.

It is important to note that ingress replication is not intrinsically a disadvantageous feature for VPN Service Providers (SP): VPLS based on ingress replication makes the customer subscribe higher committed information rates because ingress replication requires more inter-site bandwidth. That means higher revenue for the SP. In other words, from the SP's viewpoint, if customers want the plug-and-play features of Ethernet, they will pay more, because it requires much more bandwidth than the unicast-only service. But then, what is the interest in featuring multipoint inside the network? The answer is *competition* and *scalability*. On the one hand, SP featuring tree-based multicast can provide cheaper VPLS and better QoS for the same subscribed rate than their ingress-replication-based competitors. On the other hand, ingress replicating VPLS would not scale to thousands of VPN sites. Furthermore, there are emerging large-scale VPN-based applications for which unicast MPLS LSPs is not an option due to their extreme scalability requirements. Most service providers (SP) have deployed private multicast configurations for high-speed multipoint traffic across their networks. It is the case of triple-play providers [6], that deliver TV over IP multicast to their ADSL residential clients, where usually the last hop is delivered over IP unicast from a multimedia relay at the SP point of presence (PoP), or over native IP multicast to the set-

top-box. This can easily scale to thousands of channels and users, if implemented with IP/MPLS over point-to-multipoint (P2MP) label switched paths (LSP) sent from a content delivery root to the relays [7]. Therefore, a careful utilization of multicast LSPs is paramount to build a more efficient and scalable multi-service network. However, as pointed out in [1, 3, 5, 7], the trouble with the tree approach is the trade-off between *used bandwidth* and *forwarding state*. The problem is that the core network nodes must keep *per-multicast-group per-vpn per-tree* information. A first step to mitigate this effect is the introduction of a single tree per VPN, in a similar fashion to the BUS server of LAN Emulation in ATM. In this setting, all sites in a VPN forward their multicast packets to the root of the tree. Note that this is a more radical approach than the Multicast Server (MCS) of RFC2022 for IP over ATM, or the PIM-SM rendezvous point in the case of IP multicast routing, in the sense that the tree provides a broadcast channel per VPN, not per multicast group. However, even with a single shared tree per VPN, the network still keeps *per-vpn* forwarding state, which is against the basic design rule of VPN implementations: having VPN-specific information in intermediate nodes does not scale well. All VPN specific information should remain at the PE (Provider Edge) routers instead. Therefore, the standard approach [1, 2] recommends to use a number of trees to be shared by many VPNs, that conversely, introduces new bandwidth consumption penalty [7] and a complex traffic grooming optimization problem: how to aggregate multicast groups into shared trees. In [5] authors identify the trade-off issue and claim that further work is required to study it. In this paper we progress on the study of this problem and analyze the margin for improvement derived from intelligent aggregation heuristics.

II. Related Work

The present work deals with aggregation of *multicast groups*. We shall use the generic term *multicast group* to refer to a set of Customer Edge (CE) recipients of a VPN multipoint packet, be it members of an IP multicast group, the destination nodes of a multicast MAC address or the whole set of VPN member sites in the case of VPLS broadcast emulation. In this procedure, multiple multicast groups are forced to share a single *multicast distribution tree* (MDT) a.k.a. *aggregation tree* (AT) [8-10].

The idea of aggregation was firstly studied in [8], further exploited by A. Fei et al for IP multicast routing with/without MPLS in [9-11], and it has been revisited in the specific context of MPLS VPNs in several works [7, 12] where different aspects are addressed together with aggregation, including P2MP tree signaling and frame encapsulation mechanisms on the shared tree. In [10] the authors address many implementation aspects of group management and introduce a centralized management entity called *tree manager*, which is in charge of assigning groups to existing trees or create new ones. They propose that the set of aggregated trees to be established can be determined based on traffic pattern from long-term measurements. They introduce the notion of perfect match (identical trees) and leaky match to denote trees delivering to non-member leaves. When no perfect match is found, a leaky match may be used if it satisfies certain constraint (e.g., bandwidth overhead (sum of leaked links) is within a certain limit). Otherwise the incoming tree is added to the network. In this paper, we shall use a more restrictive approach given the fact that in practice the maximum number of forwarding entries is pre-determined. Thus we enforce the usage of the best-match existing tree unless the group or VPN is very small; in this latter case, we shall enable ingress-replication in order to reuse the default unicast forwarding paths and then it causes no impact in the reducible state metric (core node state) at the price of extra bandwidth.

In [10] the authors also introduce a number of metrics to measure the saving of IP forwarding state by aggregation that we shall reutilize in our work for the case of MPLS. In [12] authors use the same aggregation algorithm and applies the same to the particular case of VPN MPLS implementation based on label stacking. In [11], the authors extend their work from [9, 10] to support QoS by including measurement-based admission control to aggregated MPLS trees featuring Diffserv. Yet the forwarding state analysis and aggregation method is similar.

An alternative methodology to analyze the behavior and benefits of using shared aggregation trees in MPLS-based VPN networks was proposed in [7]. In this work, a simple state vs. bandwidth trade-off analysis was presented, showing the benefits of aggregation even if the best tree-matching algorithm is replaced by uniform random tree allocation. Like in [7] we present our results with respect to the aggregation degree, as this parameter is more general than the absolute group numbers of [10] or [12] and can be more useful in network planning. Another difference with previous analysis is the fact that in all works mentioned above the distribution of VPN sizes was considered uniform over a narrow range of sizes (e.g. 2-10 in the case of [10]). In this paper we show that this factor is relevant to the state gain obtained from intelligent aggregation.

Finally, related work in traffic grooming for multicast in optical WDM networks should be mentioned [13]. Even though the forwarding state problem of MPLS does not map exactly to this RWA problem, the ILP (Integer Linear Programming) formulation developed could be adapted to the VPLS context in order to compute the optimal tree aggregation scheme for a given data set. However, the theoretical optimum has little interest in practice, because of its computational complexity and it limits the real applicability only to very small networks.

III. Multicast in IP/BGP/MPLS VPNs

In this section we present the multicast aggregation method addressed by the respective IETF group in [3], and in a conceptually similar way in [1]. We will take the former reference as the primary. Clear descriptions of the multipoint

signaling in MPLS LSPs, BGP MPLS-based VPNs, and multicast service for VPNs can be found in [7].

A. Optimal VPN multicast routing

In a BGP/MPLS VPN network, packets are unicast-routed without any state information about the VPNs kept in core routers. That information is only known by the provider edge (PE) routers, which connect sites directly to the VPN. Client data travels from a PE to another PE through core nodes within tunnels, usually Label Switched Paths (LSPs). The associated state information in core routers depends only on the number of PEs, instead of the number of active VPNs. It may be the case that the SP is not interested in building ATs over the backbone, or is not able to do it, as it happens in GMPLS optical core networks. In this event, the ingress PE router could make multiple copies of the packet and, by unicast, send each of them through a tunnel to the correspondent egress PE router.

In the case of multicast, routing for a specific group is optimal if and only if: when a PE router receives a packet from a Costumer Edge (CE) router for the multicast group, it forwards it to all the PE routers connected to CE routers of the group; the packet is not received by any other PE router, only one copy of the packet traverses each link, and the packet goes through the minimum cost multicast tree. The problem is that this optimality requires at least one AT per source and per multicast demand. Therefore, we notice that the state information at every provider router reaches a triple dimension: multicast source, multicast group and VPN. In this way networks scale poorly. Potentially, this would require unlimited amount of state information at the provider routers, because the SP has no control over multicast groups within the VPNs, or over the number of transmitters at each group, or over distribution of receivers.

B. Multicast Traffic Aggregation

Let us name MVPN a given VPN transporting multicast traffic, which is made up of multicast packets that a CE sends to the other CEs members (also attached to their respective PEs) of the multicast group. There are two aggregation models, defined by [1] and [2], as depicted in fig. 1.

1) Inclusive aggregation tree

In this model (fig. 1.a), a distribution tree could attend traffic of one or more MVPNs. The singularity here is that every PE router supporting a site associated to any of those MVPNs becomes a part of the tree. Since trees are unidirectional, the number of routing forwarding entries (state information) is proportional to n x m; where n is the number of ATs and m is the average of PEs per MVPN. Even if each tree attends a single MVPN, the upper bound of the state information is proportional to the number of VPNs, not to the number of multicast groups inside those VPNs. This model has the inconvenience that, the more MVPNs are aggregated to a tree, the higher probability that some PEs receive useless leaked packets (packets not from their VPNs), incurring in bandwidth waste. This is because the aggregation is made without considering memberships to multicast groups.

2) Selective aggregation tree

Here (fig. 1.b), a tree is used to transport traffic of a set of multicast groups from one or more MVPNs. In this case only PE routers from MVPNs are included. In other words, there is no aggregation unless another group had the same members than other, in which case a single tree would aggregate more than one multicast group. For very high bandwidth-consuming groups, the selective aggregation model should be preferred.

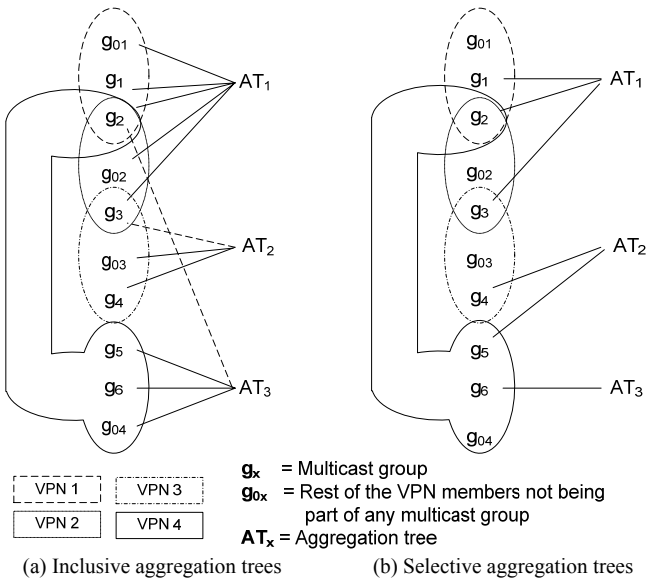

Fig. 1. Inclusive and selective trees aggregation models

In fig. 1(a) we can see that the aggregation tree AT_1 would deliver pointless traffic into PEs that do not belong to the same group (g_1, g_2, and g_3), and also to sites not associated to any group (g_{01}, and g_{02}). On the other hand, in the selective case of fig. 1(b), unwanted traffic would be received only by PEs associated to different groups.

IV. Heuristics for MVPN Aggregation

At this point, a logical question is raised: what are the quantitative effects of the aggregation? What is more, how can aggregation be measured and modeled to achieve a fair performance trade-off? In this section, we present the scenario and some heuristics in order to approach to these issues.

A. Scenario of the Problem

Since the objective of this work is to obtain some patterns in order to distinguish some thresholds and bounds in which it is convenient to apply multicast aggregation, we consider the inclusive aggregation tree model, with the aim to reduce state data and make a clear bandwidth vs. state comparison. In this sense, we also consider a centralized data control plane, which has the knowledge of the network topology, routing and VPNs.

Regarding construction of ATs, PEs will send the multicast traffic to a determined RP (root) core node, because source-

rooted trees provide more reliability and lower delay, but they increase the state number of state forwarding entries. Therefore, every AT has its own RP for its own set of MVPNs. The RP selected is the one that, in average, causes de lowest delay to MVPN PEs. This is done by calculating distances (in number of hops) with the Dijkstra algorithm. We discarded optimal solutions to get the optimal RP, because of their NP-completeness condition.

We consider that MVPNs arrive to the backbone network dynamically and sequentially, and a number of ATs are available for including them on the fly. Therefore, an incoming MVPN is assigned to one of the ATs, and it is possible that the selected AT will change after the inclusion operation. Note that this selection mechanism is very important. If an MVPN is assigned to an AT that will cause much bandwidth waste (i.e. packet drops), the whole network performance will be affected.

B. Aggregation Techniques

For a clear explanation of aggregation mechanisms proposed here, let us consider the following variables:

$V = \{v_1, v_2, \ldots, v_M\}$: Set of MVPNs

$T = \{t_1, t_2, \ldots, t_N\}$: Set of aggregation trees

$M = |V|$: Number of MVPNs

$N = |T|$: Number of aggregation trees

$V_j = \{v_1^j, v_2^j, \ldots, v_P^j\}$: Subset of MVPNs aggregated to t_j

$E^{t_j} = \{e_1^{t_j}, e_2^{t_j}, \ldots, e_z^{t_j}\}$: Set of edges of t_j

$P^{v_i} = \{p_1^{v_i}, p_2^{v_i}, \ldots, p_w^{v_i}\}$: Set of core nodes of v_i with at least one PE attached to a site belonging to v_i

$P^{t_j} = \{p_1^{t_j}, p_2^{t_j}, \ldots, p_x^{t_j}\}$: Set of core nodes of t_j with at least one PE attached to a site belonging to any of the $v_i^j \in V_j$

$PE^{v_i} = \{pe_1^{v_i}, pe_2^{v_i}, \ldots, pe_y^{v_i}\}$: Set of PEs of v_i

$PE^{t_j} = \{pe_1^{t_j}, pe_2^{t_j}, \ldots, pe_z^{t_j}\}$: Set of PEs of t_j

$w_{i,j}$: Bandwidth wasted for aggregating v_i traffic to t_j

Besides, ATs are built for aggregating a given group of MVPNs, depending on the aggregation degree (*AD*) defines as:

$$AD = 100\left(1 - \frac{N}{M}\right) \quad (1)$$

From eq. 1, $AD \approx 100\%$ ($AD = 100\%$ only for $M = \infty$) means that only one AT is used for all MVPNs, and $AD = 0\%$ means that there is a unique AT assigned to every MVPN.

In the first three following aggregation mechanisms proposed, ad-hoc ATs are built for the first *N* MVPNs, using a heuristics based on the Dijkstra algorithm. So, the first *N* MVPNs start having their own single AT after the first round. AT's RPs are selected following the procedure described previously. Aggregation mechanisms differ in how they assign MVPNs to the existing set of ATs.

1) Non-Intelligent Aggregation (NIA)

In this mechanism, after the creation of $t_1, t_2, \ldots, t_N$, the remaining $v_{N+1}, v_{N+2}, \ldots, v_M$ are aggregated to the trees by following a round robin mechanism:

if ($j = i \bmod N$) then

v_i is aggregated to t_j

As it can be warned, in this way MVPNs are aggregated without considering the bandwidth waste generated because of aggregations.

In case $P^{v_i} \subseteq P^{t_j}$ (i.e. the set of core nodes of v_j are included in the set of core nodes of t_j), the incoming v_i is aggregated straightly to t_j. Otherwise ($P^{v_i} \supset P^{t_j}$), t_j has to grow up in order to be able to include all the set of core nodes of v_i into the set of core nodes of t_j.

2) Intelligent Aggregation (IA)

Here, $v_{N+1}, v_{N+2}, \ldots, v_M$ are aggregated to the ATs looking forward to reduce the waste of bandwidth ($w_{i,k}$) and trying to fit v_i into the most similar t_k available, in the following manner:

$a = P^{t_j} - P^{v_i}$: core nodes present in t_j but not in v_i

$b = P^{v_i} - P^{t_j}$: core nodes present in v_i but not in t_j

$c = PE^{t_j} - PE^{v_i}$: PEs present in t_j but not in v_i

$d = PE^{v_i} - PE^{t_j}$: PEs present in v_i but not in t_j

$e = E^{t_j} - E^{v_i}$: Edges present in t_j but not in v_i

$f = E^{t_j} - E^{v_i}$: Edges present in v_i but not in t_j

$w_{i,j} = |V_j|(a + b + e + f) + c + d$

$w_{i,k} = \min(w_{i,1}, w_{i,2}, \ldots, w_{i,k}, \ldots, w_{i,N})$

Finally, v_i is aggregated to t_k. The idea here is to get the best AT into which to aggregate the incoming MVPN, in terms of bandwidth performance.

As in the former technique, in case $P^{v_i} \subseteq P^{t_k}$, the incoming v_i is aggregated straightly to t_k. Otherwise ($P^{v_i} \supset P^{t_k}$), t_k has to grow up to include P^{v_i} into P^{t_k}.

3) Intelligent Aggregation with Reconfiguration (IA+R)

This technique makes exactly the same aggregation procedure explained previously (IA) with a unique difference. Considering that $t_1, t_2, \ldots, t_N$ would keep growing up after some aggregation operations, they will probably increase the average RP-core distance, the total number of hops, etc. This could happen because they were built taken into account only the topologies of the *N* first incoming MVPNs ($v_1, v_2, \ldots, v_N$). After some other MVPNs arrivals, there should have a better idea of how the topology of the ATs must be. Therefore, in

order to rebuild them, they are reconfigured every u_t times (i.e. recalculation of the RP, and of the list of predecessor nodes), in the following manner:

$$\text{if } (P_j \bmod u_t) = 0 \text{ then}$$

$$\text{Recalculate } t_j$$

$$\text{where, } u_t = \left\lceil \log_2\left(\frac{M}{N}\right)\right\rceil$$

It has to be pointed out that u_t has been determined experimentally, after running extensive simulations. The number of MVPNs that t_j should have in average is:

$$\left(\frac{M}{N}\right) \approx \frac{\sum_j |V_j|}{N}$$

The logarithmic form of u_t makes this value significative even for small amounts of the M/N factor.

4) Hybrid Tree Aggregation (HTA)

Although the last aggregation technique seems to be an adequate effort for saving bandwidth, it is possible that a special group of MVPNs would cause a tremendous waste of bandwidth. It is the case in which MVPNs have only a few PEs with Customer Edges (CE) with an MVPN site attached. This phenomenon was observed during the simulations, especially with random samples with a Zipf distribution, where most VPN samples have a few members, and only very few VPN samples have a large amount of members (as we will explain in the next section). Why Zipf distribution for group or VPN sizes? Because in reality the number of VPN sites of -e.g. a bank in a city- is expected to be proportional to the number of inhabitants in that city for a fixed per-capita income, and the distribution of population per city in the world is known to tend to a Zipf distribution. It can be easily noted that the main drawback about this is that, the smaller the aggregated MVPNs are, the more useless (leaky) traffic will be delivered to PEs that are not part of those MVPNs.

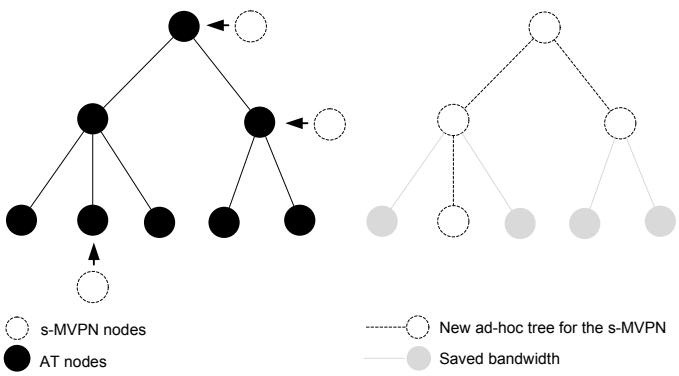

Fig. 2. Creation of ad-hoc trees for and s-MVPN

In order to avoid this, the fourth hybrid aggregation technique is proposed, in which small ATs are provided for a certain group of small MVPNs (s-MVPNs). As shown in fig. 2, if an s-MVPN with only 3 members was aggregated to a regular AT, all the packets of this s-MVPN will be dropped to PE nodes uselessly. Therefore, an ad-hoc tree is built for it. We work under the premise that only a few forwarding state entries would be needed to serve small MVPNs with small ATs. Avoiding to aggregate s-MVPNs to large aggregation trees, will let us save bandwidth with a small increase of the state information.

In algorithm 1, the line 2 states that if the number of nodes that would be involved in the s-MVPN v_i is less than a certain threshold, then v_i has to be aggregated to an ad-hoc tree. In case there were previous ad-hoc trees built (line 3), v_i is aggregated to the ad-hoc tree that generates the lower bandwidth waste (lines 4 - 6). If there were not any ad-hoc tree built previously, an ad-hoc tree is built for v_i, and v_i is aggregated to it (lines 8 - 9). Finally, lines 11 - 12 refer to the case in which v_i does not need an ad-hoc tree, therefore, the IA+R technique is applied.

1. **for each** $v_i \in V$ **do**
2. **if** $|C^{v_i}| \le s_h$ **then** (where s_h = upper bound of $|P^{v_i}|$)
3. **if** $T^s \notin \varnothing$ **then**
4. find $w_{i,k} = \min(w_{i,1},..,w_{i,k},...)$, waste of $t_k^s \in T^s$
5. **if** $w_{i,k} \le |v_i|$ (i.e. one ad hoc tree saves BW)
6. aggregate v_i to t_k^s
7. **end**
8. **else**
9. build $t^s \in T^s$ and aggregate v_i to the new t^s
10. **end**
11. **else**
12. aggregate v_i to $t_j \in T$ applying the IA+R procedure
13. **end**
14. **end for each**

Algorithm 1: Pseudocode of the Hybrid Tree Aggregation

V. Simulations and Results

Extending the referred methodology presented in [7] -used for the analysis of aggregation in a generic topology-, major extensions have been developed in order to achieve more realistic simulations, in order to observe the aggregation behavior. Firstly, we constructed a simulation model to work with real network topologies -listed in table 1-, with the aim to test the tree aggregation techniques in diverse (i.e. different number of core nodes and links, different topologies, etc.) and real scenarios. Secondly, we generated random sets of VPN samples with different distributions in terms of scope (i.e. number of core routers involved in the VPN) and density (i.e. number of PEs involved in the VPN) -as described in table 2-, in order to observe the implications of these VPNs in the behavior and efficiency of the aggregation techniques proposed. According to table 2, it can be deduced that 12 possible combinations of VPN samples distributions were applied. From these distributions, we believe that Zipf (for core and PE routers distribution) is probably the closest to the reality.

TABLE I. REFERENCE & COMERCIAL NETWORKS FOR SIMULATIONS

Name	Core Nodes	Core Links	Average Grade (core)
Abilene[a]	10	13	1.30
NSFNET[b]	14	20	1.43
KPN (Europe)[c]	39	52	1.33
Tiscali (World)[d]	45	73	1.62

a. http://abilene.internet2.edu/, b. http://www.nsf.gov/,
c. http://www.kpn.com/kpn/show/id=1561743, d. http//www.tiscali.net/

Regarding the network topologies specifications, in a lower level, two PA routers (provider aggregation routers) are attached to each core router, and the set of PE routers attached to those both PA routers. PE routers provide Internet and Layer 3 MPLS VPN services from these major locations, as proposed in [14] (fig. 3). PA routers reduce the number of IGP adjacencies that have to be maintained by the backbone routers to two, because each core router has to peer only with two aggregation routers (in addition to the other core routers in the backbone) instead of with all the PE routers attached to it, whose number can be fairly high. Each PE router is connected to both PA routers via PoS (Packet over SONET) links.

TABLE II. DISTRIBUTION OF VPNS AMONG NODES FOR SIMULATIONS

Type of Scope	Distribution of Core Routers	Distribution of PE Routers
Fixed[a]	25%, 50%, 75%, 100%	Uniform or Zipf
Variable[b]	Uniform or Zipf	Uniform or Zipf

a. It means that all the VPNs involve x % of the core routers
b. It means that all the VPNs involve a random variable (not fixed) percentage of routers

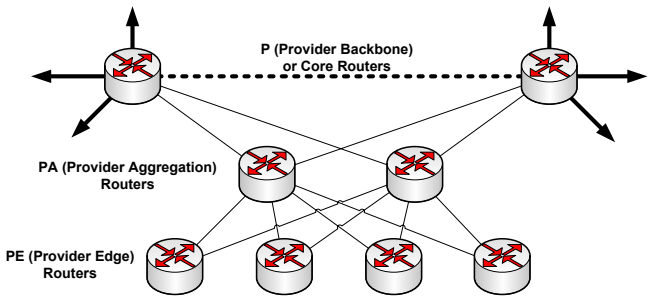

Fig. 3. Provider aggregation model for each core node, which is actually made up of three levels of routers.

Considering these scenarios, extensive simulations were run under Matlab 7.1, with 1,000 VPNs samples generated randomly according to the distributions explained before. In the results, snapshots of the different variables and metrics are taken, considering that every VPN has a source CE node already sending packets to the other members of the multicast VPN session. Results measure the impact of one packet per VPN sent to all its members. As pointed out before, four different core network topologies have been simulated, in order to observe the behavior of the techniques; however, for space limitations of this document, we only present the graphical results for the NSFNET reference network; and regarding the other networks, the most relevant observations will be addressed.

A. State Forwarding Entries Performance

One of the most representative results is that of figs. 4 and 5, which depict the multicast forwarding entries for all the aggregation types. It is important to note that, as long as the AD increases (towards to having a unique AT for all VPNs), the number of forwarding entries are reduced, as expected.

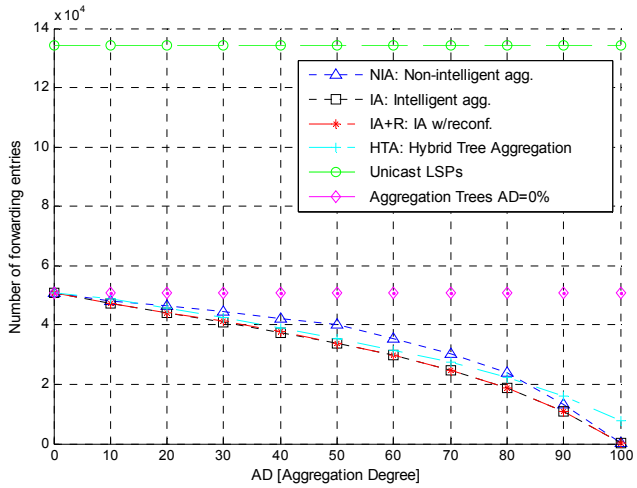

Fig. 4. State entries of NSFNET network, uniform VPN size distribution.

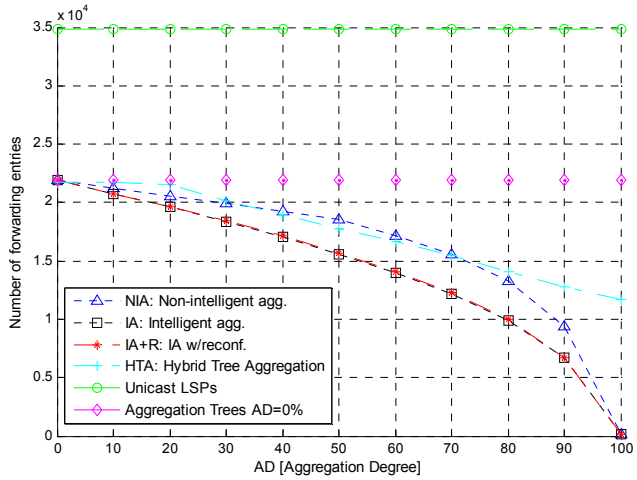

Fig. 5. State entries of NSFNET network, Zipf VPN size distribution.

When using aggregation trees, there are two types of state forwarding entries: Those related to the set of ATs established in the network, and those of each MVPN attended at PE nodes. The number of state forwarding entries related to the first case is equal to the number of core routers and PA routers. These entries are shared for all MVPNs that are part of that aggregation tree. Instead, entries stored at PE routers are specific for VPNs. In [9] authors introduce the concepts of *reducible* and *non-reducible* state information. By using ATs, or even by using traditional IP multicast, terminal nodes necessarily need state information related to multicast traffic of VPNs, and that is why this state information is called *non-*

reducible state information. On the contrary, the number of forwarding entries of distribution trees is variable and will depend on the a*ggregation degree* (*AD*) used in the network. The following metric is called *reducible state reduction ratio*:

$$RSRR = 1 - \frac{\sum_{t_j \in T} S_{reducible}}{\sum_{v_i \in V} S_{reducible}^{AD=0\%}}$$

This represents the relation between the reducible state for the aggregation trees technique, and the reducible state when building one tree per MVPN. This is an effective way to monitor the amount of savings on state information obtained by the different aggregation techniques (figs. 6, 7).

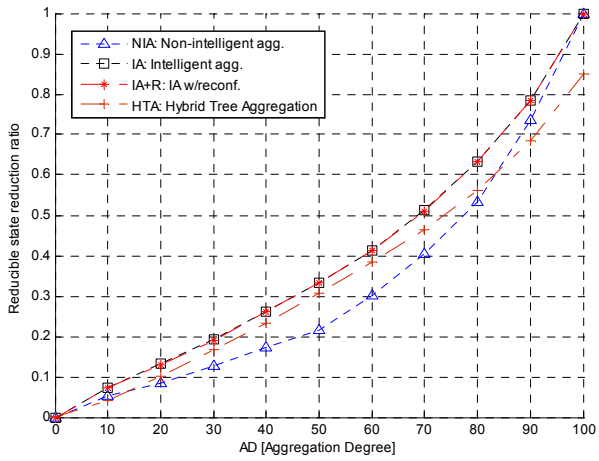

Fig. 6. RSRR of NSFNET network, uniform VPN size distribution

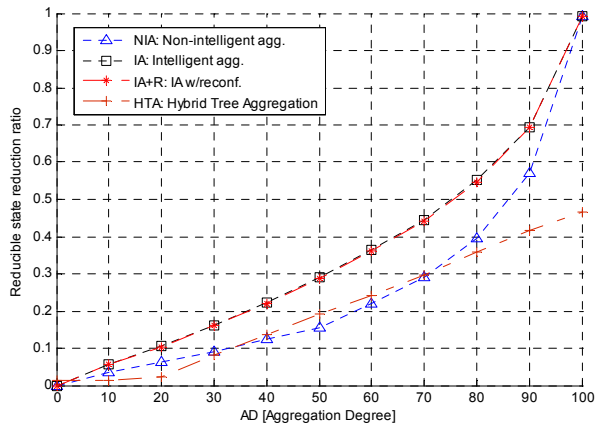

Fig. 7. RSRR of NSFNET network, Zipf VPN size distribution

B. *Bandwidth Performance*

In figs. 8 and 9 it is shown that, naturally, NIA is the worst of the aggregation techniques (because it does not consider the bandwidth impact on ATs selection), and IA and IA+R techniques perform as expected, with just an insignificant difference between them. HTA has the better performance here, mainly because it saves bandwidth by building ad-hoc aggregation trees to small VPNs. However, as shown before, HTA sacrifices bandwidth savings instead of state information. The graphic also presents the bandwidth consumption when AD = 0%.

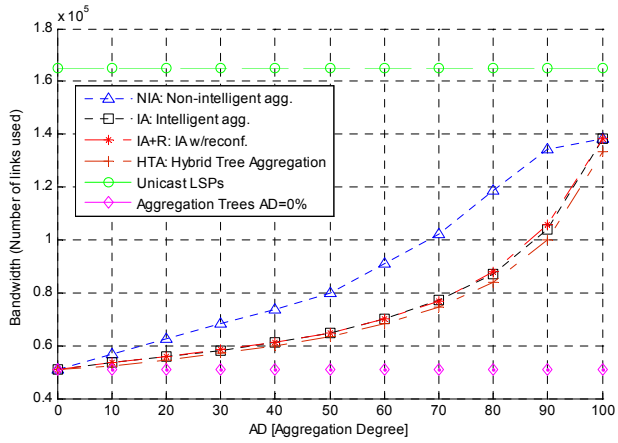

Fig. 8. BW consumption, NSFNet network, uniform VPNs size distribution.

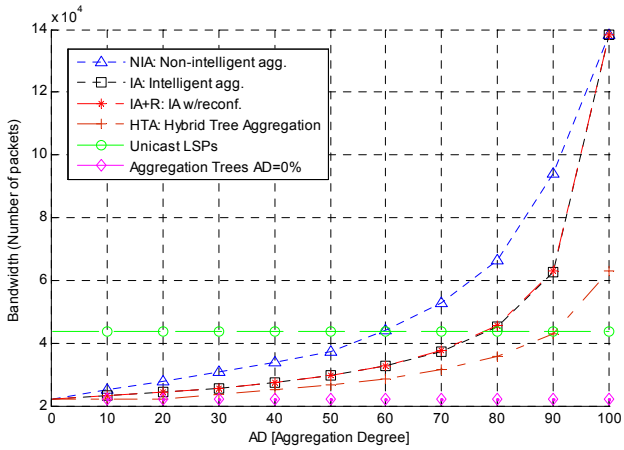

Fig. 9. BW consumption, NSFNet network, uniform Zipf size distribution.

As long as AD increases, the number of ATs decreases, therefore, the global waste of bandwidth raises up. In order to see this in a more adequate manner, we define a metric named *multicast efficiency* (δ) which represents the improvement (in the case where the value is positive) or degradation (in the case that the value is negative) achieved when using ATs against the unicast LSPs solution (fig. 10); and the *average ratio of useless bandwidth* (β), which rates the bandwidth wasted by using ATs against the use of tree per VPN (AD = 0%) (fig. 11):

$$\delta = 1 - \frac{bw_{AT}}{bw_{LSP}}; \quad \beta = \frac{bw_{AT}}{bw_{AD=0\%}} - 1$$

where, bw_{AT}: BW consumed with ATs
bw_{LSP}: BW consumed with unicast LSPs
$bw_{AD=0\%}$: BW consumed with AD = 0%

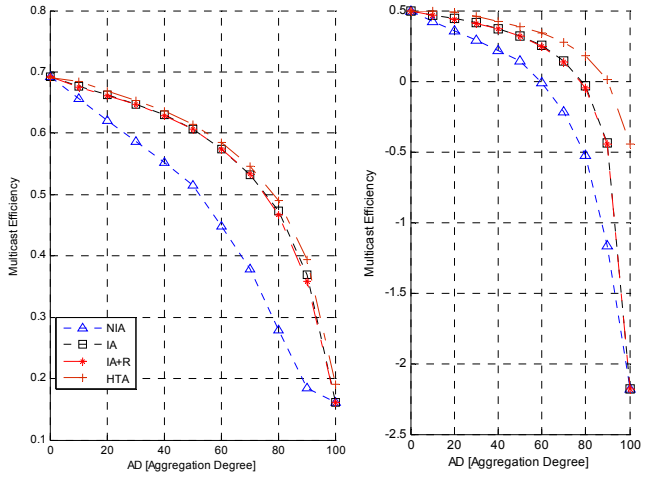

Fig. 10. BW multicast efficiency, NSFNet network, uniform (left) and Zipf (right) VPN size distribution.

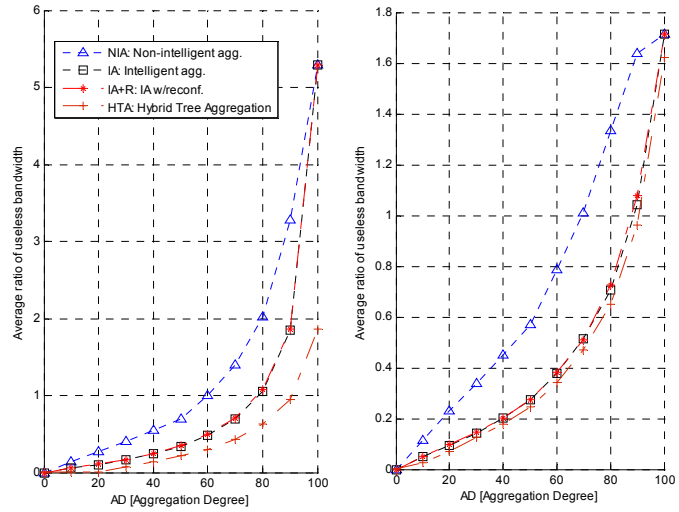

Fig. 11. Average ratio of useless BW, NSFNet network, uniform (left) and Zipf (right) VPN size distribution.

C. *Network Design and Deployment Issues*

For network design and deployment purposes, the previous simulation results can be used to determine the range over which aggregation trees could be useful. It is clear, for example, that roughly under AD = 80%, the creation of aggregation trees is useful, with a uniform distribution of VPNs to core nodes and PEs.

If we assume that the most common VPN size distribution is Zipf, we can see that LSP unicast improves its performance. As shown before, it is specially in this case where the HTA technique could be very useful. By building ad-hoc small trees for small VPNs (situation very common with Zipf), it reduces the bandwidth waste by paying only a few more state forwarding entries.

It should also be noticed that no consideration has been made about QoS. Aggregation strategies should take into account impact on current link loads before a new tree is included in an aggregation tree, like in [11].

VI. Conclusions and Future Work

In this paper, we have analysed different new aspects of multicast traffic aggregation in an MPLS-based VPN network that can be a useful input to create automatic tools for multicast VPN traffic engineering, and for design and deployment purposes. It was clearly shown the advantages of the aggregation concept, and results show the behavior of the trade-off along the range of aggregation ratios. We have quantified the gain obtained by means of different heuristics and metrics and we have shed some light on the impact of a Zipf distribution of group sizes on its effectiveness in terms of number of forwarding entries and saved bandwidth.

VII. Acknowledgements

The work described in this paper was carried out with the support of MEDIANET PRICIT 2009/TIC-1468, from the Community of Madrid; and Fundación Carolina, Spain.

References

[1] R. Aggarwal, Y. Kamite, and L. Fang, "Multicast in VPLS," IETF, Internet draft draft-ietf-l2vpn-vpls-mcast-10.txt, standards track, Feb. 2012.

[2] E. Rosen and Y. Rekhter, "BGP/MPLS IP Virtual Private Networks (VPNs)," IETF, RFC http://www.ietf.org/rfc/rfc4364.txt, Feb. 2006.

[3] E. Rosen and R. Aggarwal, "Multicast in MPLS/BGP IP VPNs," IETF, Internet draft draft-ietf-l3vpn-2547bis-mcast-10.txt, submitted to IESG for publication Jan. 2010.

[4] Ed. IJ. Wijnands, Ed. I. Minei, K. Kompella, and B. Thomas, "Label Distribution Protocol Extensions for Point-to-Multipoint and Multipoint-to-Multipoint Label Switched Paths," IETF, RFC 6388 http://www.ietf.org/rfc6388 Nov. 2011.

[5] E. Y. Kamite, Y. Wada, Y. Serbest, T. Morin, and L. Fang, "RFC 5501: Requirements for Multicast Support in Virtual Private LAN Services," *(Informational),* March 2009.

[6] J. Sanchez, P. Manzanares, and J. Malgosa, "Spanish Telco Strategies Facing New Integrated Digital Transmission Advances," *Global Communications Newsletter. IEEE Communications Magazine,* vol. 44, 10-02-2006 2006.

[7] I. Martínez-Yelmo, D. Larrabeiti, and I. Soto, "Multicast Traffic Aggregation in MPLS-Based VPN Networks," *IEEE Communications Magazine,* vol. 45, pp. 78-85, 2007-10-08 2007.

[8] P. I. Radoslavov, R. Govindan, and D. Estrin, "Exploiting the Bandwidth-Memory Tradeoff in Multicast State Aggregation, Technical report," USC Dept. of CS Technical Report 99-697 (Second Revision) July 1999.

[9] A. Fei, J. Cui, M. Gerla, and M. Faloutsos, "Aggregated Multicast with Inter-Group Tree Sharing," in *Third International COST264 Workshop on Networked Group Communication* London: Springer-Verlag, 2001.

[10] A. Fei, J. H. Cui, M. Gerla, and M. Faloutsos, "Aggregated Multicast: an Approach to Reduce Multicast State," in *Proceedings of Sixth Global Internet Symposium (GI2001) in conjunction with Globecom 2001*, November 2001.

[11] J. H. Cui, A. Fei, M. Gerla, and M. Faloutsos, "An Architecture for Scalable QoS Multicast Provisioning," UCLA CSD Technical Report #010030 August 2001.

[12] G. Apostolopoulos and I. Ciurea, "Reducing the Forwarding State Requirements of Point-to-Multipoint Trees Using MPLS Multicast"," *ISCC Proc.,* pp. 713-718, June 2005.

[13] R. Ul-Mustafa and A. E. Kamal, "Design and Provisioning of WDM Networks with Multicast Traffic Grooming," *IEEE Journal on Selected Areas in Communications, Part II: Optical Communications and Networking,* vol. 24, Apr. 2006.

[14] Jim Guichard, François Le Faucheur, and J.-P. Vasseur, "Definitive MPLS Network Designs," Cisco Press, March 14, 2005, p. 552.