AS Tree Selection for Inter-Domain Multipoint MPLS Tunnels

Stefano Secci[1,2], Jean-Louis Rougier[1], Achille Pattavina[2]
[1]TELECOM ParisTech (ENST), France. E-mail: {secci, rougier}@telecom-paristech.fr
[2]Politecnico di Milano, Italy. E-mail: {secci, pattavina}@elet.polimi.it

***Abstract*—In this paper, we study the problem of inter-domain AS tree selection for multipoint tunnel set-up within an alliance of ASs. We first describe the framework of our work, based on the introduction of a service plane for automatic multi-domain service provisioning. We introduce an abstract representation of domain relationship by means of directional metrics which are applied to a triplet (ingress point, transit AS, egress point) where the ingress and egress points can be ASs or routers. Then, we focus on the multipoint AS Selection problem that arises in such an architecture. The corresponding constrained Steiner problem is known to be a hard problem, and the introduction of directional metrics increases its complexity. We propose an original approach that allows one to reach almost optimal solutions with tractable computation times. Besides its performance, one contribution of this paper is that some steps of the proposed heuristic can be precomputed, independently of the tunnel demands. By extensive tests on random topologies derived from the Internet, we show that our heuristic is often equal or a few percent close to the optimal, and that, in the case of precomputation, its time consumption can be much lower than other well-known algorithms.** [1]

I. Introduction

The success of IPTV and, more generally, of multimedia transmissions over the Internet have recently increased the interest in point-to-multipoint transmission services. Multipoint transmissions allow optimizing network resources, which is of strategic importance for bandwidth consuming flows such as HD-TV. Most video transmissions take place within a single domain, but a demand for inter-domain multipoint services is increasing. Inter-domain multipoint capabilities are also interesting for VPNs. In this paper, we shall concentrate on the connection-oriented multipoint approach (simply referred as multipoint in the following). Recent works carried within IETF have extended the MPLS-TE architecture in order to offer point-to-multipoint tunnels (i.e. P2MP TE-LSPs). Extensions have also been defined for inter-domain tunnel set-up [1]. However, research efforts are still needed in order to provide a complete architecture for inter-domain P2MP tunnel set-up. In this paper, we initiate some works in this direction by first concentrating on the problem of inter-domain AS tree selection. Other related aspects, such as signaling, policy and management issues, are only briefly discussed.

Given the complexity of multipoint route computation with QoS constraints, the use of Path Computation Elements (PCE) seems particularly appropriate [2]. As TE information is not shared between ASs for scalability and confidentiality reasons [1], a single PCE is unlikely to be able to compute a full inter-AS path, and has to collaborate with the PCEs of other ASs. Some distributed path computation methods based on interactions between PCEs have thus been defined (e.g. [3]). However, these works rely on an a-priori known set of AS-paths, from which the end-to-end path is then computed. How this AS-path is selected is not specified. In [4], we tackled this issue for point-to-point tunnels, with both QoS and economical constraints. In this paper, we identify the requirements in order to generalize this approach for P2MP tunnels and we propose algorithms for inter-AS tree (or multipoint routes) selection.

The rest of this manuscript is organized as follows. The architectural framework is explained in Sect.II. The AS tree selection problem is then analyzed in Sect.III. Sect.IV describes a novel algorithm for the AS tree selection for our architecture, and in Sect.V we evaluate it on realistic topologies.

II. Architectural Framework

Inter-AS path computation should be based on economical constraints and consider individual AS routing policies (whereas most works on inter-domain routing tend to consider technical constraints only). The introduction of a "service plane", working on abstract representations of inter-domain relationships, seems attractive in order to capture these features. In this context, the current developments at the IP Sphere Forum (IPSF) [5] are of interest. They are modeling the functional features of a multi-domain service plane supporting, among other things, the advertisement of providers' network service capabilities. This service plane does not carry explicit routing data, but multi-domain *service* data (customer order, service owner, etc.) that the current protocols do not handle. This data may include guarantees on the offered transit QoS performances, and policies for this service. We believe that such a service plane is not necessarily meant to be extended to the whole Internet, but could be used by a limited group of neighboring providers wishing to jointly offer inter-domain TE services (in the context of an alliance for instance).

Consistently to IPSF requirements, service elements could be used by local "AS Selection Agents (ASAs)" for service selection and instantiation. Each ASA can retrieve a service element repository indicating routing policies between domains offering inter-domain transit services, their costs, and potentially some TE information and statistics on past

[1]The work described in this paper was carried out with the support of the French ANR ACTRICE project, and of the BONE-project ("Building the Future Optical Network in Europe"), a Network of Excellence funded by the European Commission through the 7th ICT-Framework Programme.

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the ICC 2008 proceedings.

transactions. The ASAs use this service-layer information to compute constrained inter-AS routes, i.e., point-to-point (AS paths) or multipoint (AS trees) routes based on both cost and QoS constraints. The service plane is also responsible for managing the transactions needed for service acceptance by all the domains of an inter-AS route, and is then interfaced to the underlying PCE-based control plane for router-level path computation and for tunnel signaling. [6] gives more details on functional elements and protocol extensions. In the following, we assume the availability of such an architecture, involving multiple domains interested in tunnel brokering, and analyze the issue of selecting multi-domain AS trees for multipoint tunnels that meet both TE and economical requirements.

A. Directional Transit Metrics

In order to model the economical relationship of autonomous systems, we believe that each domain would be interested in advertising *directional metrics* for tunnel transit. In other words, an AS willing to offer transit services would be likely to announce different transit costs and capabilities as function of both the entry and the exit ASs. More formally, in the following, a directional metric (representing either a cost or a capability) is a metric associated with a triplet (a, b, c) where a, b, c are vertices of the service graph, with the following convention: 'a' represents the ingress AS, 'b' the transit AS and 'c' the egress AS. 'a' and 'c' may also represent ingress and egress border routers or group of routers [6].

It is worth noticing that directional metrics could also represent different operational costs for wide area carriers, allowing one operator to differentiate long distance intra-domain, intra-regional or intra-metropolitan transits.

In this framework, every AS advertises its transit policies via directional service elements at the service plane, potentially with a specific scope. Upon arrival of a request, an ASA employs this information to compose service elements; locally, it could modify the service elements' directional metrics to apply local policies or bilateral agreements. The selected service elements consist in a chain of domains that could potentially configure the required connectivity.

B. Inter-Domain Multipoint MPLS-TE Network Service

Once the the source ASA composes an appropriate AS chain, the selected domains can either reject the service - because of local policies for certain connection requests or of transient lack of resources - or instantiate it. These transactions are managed at the service plane. If a service can be instantiated, it is then activated at the management plane by configuring local policy managers to filter further inter-AS (PCEP and RSVP-TE) messages [6]. Then, at the network control planes, the different branches of the tree need to be set-up within and among the selected ASs, which require intra-domain tree computations, under QoS constraints and with respect to individual routing policies. Given the complexity of the distributed multipoint path computation, the use of the PCE architecture seems natural [2]. Once the service is activated, the head-end router queries the local PCE, which collaborates with the other PCEs over the AS tree (using methods such as [3]) for the inter-AS multipoint LSPs computation and configuration. Finally, we rely on the inter-AS (G)MPLS-TE technologies, currently under development [1][7], for the set-up of LSPs across provider boundaries.

As already suggested before, it should be noted that some of the parts of this service architecture and tunnel instantiation procedures rely on current works that are still in progress. These aspects of the architecture are, however, not further considered in this paper which focuses on AS tree computation.

III. Possible AS Tree Selection Schemes

As explained before, we consider that the ASA of the source domain has access to a repository where all the service elements are stored. This repository allows building an AS graph, which, as previously mentioned, does not represent the whole Internet but only a subset of collaborating ASs sharing TE issues. Information mainly consists in *directional* transit costs associated with a 3-uple: transit AS-node, incoming AS and outgoing AS. Transit AS may declare in the repository some TE information associated with these directional links (3-uples), for instance some bounds on transit latency, jitter and their bandwidth availability (potentially for each service type). Over this weighted and constrained graph, the P2MP AS tree is to be selected by the source ASA.

A. AS Tree elements

In the following, we discuss possible selection algorithms. An agreed taxonomy is however needed to identify the elements of a P2MP AS tree (Fig.1):

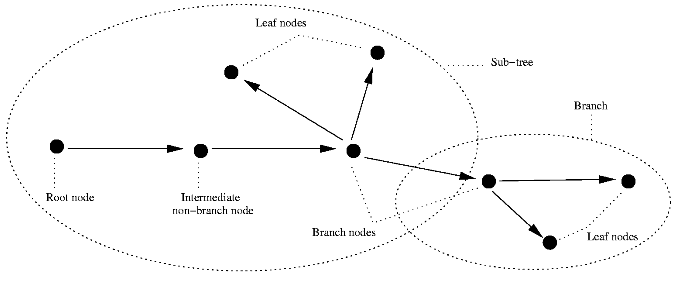


Fig. 1. Point-to-multipoint tree

- Root node: source node of a P2MP data transmission.
- Leaf node: node destination of the data transmission.
- Branch node: node that performs data replication.
- Intermediate node: non-branch and non-root node.
- Bud node: a leaf-and-branch node.

Furthermore, a set of nodes can be classified as:

- P2MP sub-tree: part of a tree such that the root or an intermediate node is connected to a subset of leaves;
- P2MP Branch: part of a sub-tree such that a single branch is connected to a subset of leaves.

B. Irrespective Routes Computation with Post Merging

A simple method relies on the following steps: compute the shortest inter-AS route subject to all constraints for each leaf AS; join the sub-routes of the routes sharing directional arcs.

We refer to this algorithm with the acronym IRC-PM. The resulting AS tree has sparse branches in not optimal positions. It is important to remark that resources (e.g. bandwidth) are shared on common links. Hence it is better to adopt algorithms allowing to reduce the tree cost by encouraging arcs sharing.

C. Iterative Point to Point Selection

Breaking the P2MP problem into multiple P2P route selections, inter-AS routes tend to share (directional) arcs: compute the shortest inter-AS route subject to all constraints from the root AS to a first leaf AS; assign null cost to all directional arcs taken by the first route and compute the inter-AS route to the second leaf; repeat the process for every leaf AS.

We refer to this algorithm with the acronym I-P2P. An advantage of this approach is that it still does not require the knowledge of all leaf ASs during the tree computation, while being more sensitive to link sharing than IRC-PM. However, the solution (and its optimality) strongly depends on the order in which routes to leaf nodes have been computed.

D. Steiner Tree

To avoid the dependency on leaf ordering, it is needed to compute the optimal tree that spans all the destinations at once, i.e., the so called Steiner tree [8]. This optimization problem is known to be NP-hard, and is more complex when taking into account additive constraints. The problem not being tractable for large instances, heuristics are needed.

Note that classical heuristics can not be used directly within our framework because of directional metrics. In order to use them, we first need to extend the graph to obtain a classical weighted graph (with weights on each vertex). This is possible using the following method: each node is to be exploded in a number of nodes equal to the number of neighbors to which it is connected. Then, directional metrics are applied to simple arcs connecting these new nodes, while null metrics are to be applied to arcs connecting nodes of different ASs. Being aware that the AS-graph has a scale-free nature (i.e. a few nodes attracting the most of the arcs), we analyzed the three differently connected AS-subgraphs at the internet backbone built as described in Sect.V. The backbone is a selection of the ASs that may correspond to network providers potentially interested in inter-domain tunnel provisioning. For these AS-graphs, we found that the average outdegree of an AS-node can be approximated by $\sqrt[3]{n}$. This suggests that the aforementioned graph extension requires approximately $n\sqrt[3]{n}$ new nodes and arcs, for a initial graph of n nodes with directional metrics.

Heuristics to the Steiner problem have been studied extensively. A comparison of some of the main heuristics can be found in [9] for instance. In the following we shall consider two algorithms for the sake of performance comparison described in this survey. The first one consists in a variant of I-P2P (described above), employing a constrained version of the Bellman Ford algorithm [10]. The second one is the Kompella's centralized algorithm [11], which can be summarized as shown: it computes the all pair constrained shortest paths and builds the closure graph of shortest paths from the root to the leaves; then it finds the constrained spanning tree of the closure graph; finally, it expands the spanning tree avoiding possible loops. The overall time complexity of the Kompella's algorithm is $O(n^3 D)$, where n represents the number of vertices, and where D is the integer value of the delay bound. For graphs with directional metrics, the time complexity of this heuristic after the graph explosion thus becomes $O((n\sqrt[3]{n})^3 D) = O(n^4 D)$.

IV. The RCOM AS Tree Selection Scheme

We have seen that the constrained Steiner problem with directional metrics can be relaxed to its classical form only by creating a larger and denser graph with "classical" metrics. Alternatively, to solve this specific problem we devise an ad-hoc heuristic called Routes Collection and Optimal Matching (RCOM), composed of two steps:

1) Routes Collection: some feasible point-to-point routes towards each leaf AS node are collected
2) Optimal Matching: the optimal matching of collected routes is reached minimizing the tree cost.

Contrarily to IRC-PM, RCOM retains a subset of feasible routes instead of only one route per destination. With respect to I-P2P, RCOM should be more flexible in branch and bud nodes placement, since it can reach a wider set of solutions. Last but not least, in Sect.IV-C we show that the main time consuming tasks can be pre-computed before the request arrivals and independently of these requests.

A. Routes Collection

To collect the per-destination routes set, we devise an ad-hoc breadth-first-search algorithm with limited depth. It starts at the root, moves to unvisited neighbors, collects the routes if a destination is attained, and so on, until no longer routes can be collected. It stops at a given number of hops or during the search by pruning branches depending on metric bounds.

This approach was inspired by the A*prune algorithm [12], proposed to solve the constrained k-shortest paths problem. Our approach differs from it in that: since the final objective is the selection of the optimal tree, further pruning (besides that on the additive metrics) depending on the route cost is performed, giving priority to the least hop routes; given that there is no need to sort the candidate routes (as A*prune does), the number k of shortest routes is not fixed and all the experienced (feasible) routes are collected (i.e., we do not need a best-first-search approach).

Collection algorithm: Let $\overline{v}$ be the threshold cost vector with one entry per destination. Each entry is a threshold re-calculated at each new route collection. The starting values are infinite. Then, an entry is initialized when at least F routes have been collected for that destination; F has to be chosen conveniently (we use $F = \sqrt[3]{n}$ in our simulations). Each threshold is calculated as the average cost of those routes, with a variance on the average cost minor than the average of this variance: simply, within the first F routes, those with a very high cost with respect to the others are not kept into account. In this way, the threshold has a decreasing trend, with a starting value not excessively high. The least hop routes are, thus, privileged because the cost bound is higher in the first hops. Favoring routes of a few hops is a suitable approach for our specific problem, since long routes crossing several ASs risk to have a small number of arcs joint with the previously selected ones, and tend to have very high costs. In this way we try to cut a lot of branches that would have been considered by general purpose solvers for an exhaustive optimization.

We also define the *projected cost* of a sub-route as the sum of the current sub-route cost (from source to intermediate node 'i') and the cost of the shortest path from the tail of this route (i.e. node 'i') towards the destination. This requires having pre-calculated the shortest paths costs from all potential intermediate node towards the destinations.

Algorithm IV.1: ROUTES COLLECTION(G)

```
procedure POP($c, d, h, \pi$)
  – $\overline{f}$ : per-destination vector with counters of found routes so far
  – $a$, $d_a$, $c_a$ : next directional arc, delay and cost of $a$
  – $M$ : multicast group (set of leaf nodes)
  if $h = H$
    then
      if $\exists!$ leaf $d \mid c + SPC(\pi[h], d) < \upsilon(d)$
        then add $\pi$ to $\zeta_{cand}$
      if $\pi[h] \in M$
        then
          if $c < \upsilon(\pi[h])$
            then
              add $\pi$ to $\zeta_{sel}$
              $f(\pi[h]) \leftarrow f(\pi[h]) + 1$
              if $f(\pi[h]) \geq F$
                then update $\upsilon(\pi[h])$
    else
      for $i \leftarrow 1$ to $N$
        do
          if $i$ adjacent to $\pi[h]$, and $i \notin \pi$
            then
              $\pi[h+1] \Leftarrow i$
              $a \leftarrow (\pi[h-1], \pi[h], \pi[h+1])$
              if $h = 0$
                then POP($c, d, h+1, \pi$)
                else if $d + d_a < D$
                then POP($c + c_a, d + d_a, h+1, \pi$)

main
  $H \leftarrow 1$, $\overline{\upsilon} \leftarrow \overline{\infty}$, $\zeta_{cand} \leftarrow \{\pi_0 = (root)\}$
  while $\zeta_{cand} \neq \emptyset$ or $H < H_m$
    do
      extract a subroute $\pi$ from $\zeta_{cand}$
      POP($cost(\pi), delay(\pi), H-1, \pi$)
      $H \leftarrow H + 1$
```

The pseudo-code is in Alg.IV.1. The search starts looking for feasible routes at 2 hops, then 3, and so on. At every iteration, the search looks only at those routes with equal hop number H, up to a given bound H_m. At every iteration, the sub-routes in the set ζ_{cand} are the starting point of the search. At every call of POP(), c and d are the cumulative cost and delay of the route handled by the current route vector π with h hops number. When visiting the root neighbors ($h = 0$), π has only the root, and the delay is not verified. Then, the function recursively visits every neighbor of the sub-route tail node, updating π, and evaluating the route feasibility on the cumulative delay. At the H^{th} hop, the route is collected in the set ζ_{sel} if a leaf is visited, if its cost is minor than the threshold, and if the delay bound is respected; it is also added to ζ_{cand} for further expanding and possible selection in the next hop only if, for at least one destination, its projected cost is equal to or minor than the threshold.

B. Routes Matching

The routes in ζ_{sel} define a subgraph built as superimposition of their directional arcs. The optimal tree is thus the minimal composition of directional arcs linking the root to the leaves within this subgraph. This is to be solved through Integer Linear Programming, which is not complex given the limited size of ζ_{sel} and given that there is no need to verify the additive metrics any longer. Indeed, forcing each destination to be crossed by at least one route, we assure that the leaves are reached and the delay constraint is satisfied. The RCOM complexity is thus dominated by the collection algorithm.

C. Complexity and Pre-computation

The majority of the time is spent in computing the (unconstrained) shortest path costs, which are needed to determine the projected costs, in the collection algorithm. We propose to pre-compute them, prior to any request, and after any topological and costs update. This can stand when costs and topology are expected to change much less frequently than the requests arrival, and this hypothesis would apply to the presented multi-domain architecture. Hence prior to any request (characterized by root, leaves, and end-to-end constraints) a simplified version of the Floyd's algorithm [13] can be used in order to pre-computed the cost of the shortest paths (SPC matrix in Alg.IV.1) from any node to any node (A2ASP). Floyd's algorithm takes $O(n^3)$ time to compute, which becomes $O(n^4)$ for graphs with directional metrics assuming an average degree of $\sqrt[3]{n}$ (see Paragraph III-D). The subsequent breadth-first search would have, without pruning, a time complexity of $O(n^{\frac{1}{3}H_m})$ for the worst case, approximating the base (branching factor) to $\sqrt[3]{n}$. Because of pruning, it is more efficient than that.

To improve the execution time, A2ASPs computation should be pre-computed, prior to any request, and triggered by topology and costs update. In this way the post-request worst case complexity of the collection becomes O($n^{\frac{1}{3}H_m}$).

For the sake of comparison, the centralized heuristics proposed so far for constrained multicast routing, as those in [9], do not have a sub-algorithm independent of the constraint values. For example, the Kompella's algorithm computes *constrained* A2ASPs to build the closure graph with a complexity proportional to the delay bound (see Paragraph III-D). Or, the Zhu's algorithm [10] uses as starting point a least-delay spanning tree. Both Kompella's and Zhu's algorithms have an overall complexity equal to the post-request complexity, which is, for a graph with directional metrics, bigger than O(n^4) [9].

It is worth mentioning that given the breath-first-search nature of the collection algorithm and the additive constraint transparency of the routes matching, an extension of the RCOM approach to multiple additive constraints would scale with the number of constraints (besides the delay).

V. Performance analysis

We compare the described algorithms in terms of optimality and execution time, and analyze the characteristics of the selected AS trees. We chose to use realistic topologies: we dumped the AS whois database containing interconnection data available at [14]. As stated before, our architecture is not meant to be used at Internet-wide scale (even the PCE-based one is not meant to be) but on a set of ASs collaborating to a common service plane. We use Internet topology estimations in order to be as realistic as possible. Three topologies are considered. The first is selected so: among all the ASs, only those with at least 7 adjacencies are kept, focusing so on network providers potentially interested in inter-domain tunnel provisioning; then, only those ASs with more than 2 adjacencies within the selection are kept in. The final topology, called ATL7, has 643 AS-nodes. The second topology, TOP300, is built with the 300 most connected nodes of ATL7.

Then we generated the directional metrics to apply. We ranked the nodes depending on their degree. We then assigned transit delays and inter-AS capacities normally around different values depending on AS ranking. The transit costs are calculated with a $log(x)/x$ law where x is the minimal directional capacity between two neighbors ASs. More details can be found in [4] (not included because of page limit).

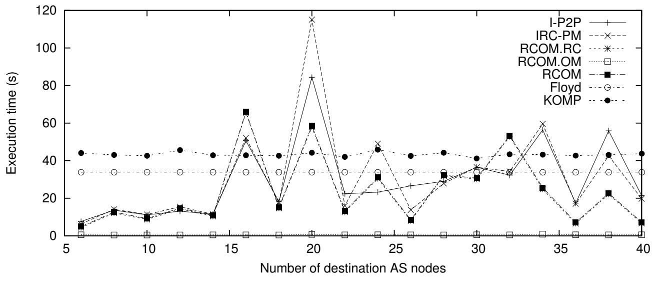
A. Algorithms performances

We run the algorithms for different sizes of the destinations group. Root and leaves are generated randomly. The delay bound is set to 1.5 s and the bandwidth to 6 Mb/s.

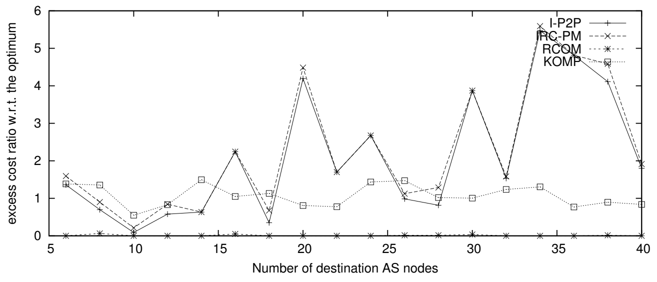
1) Execution times: Figs.2,3a display the execution times obtained for the TOP300 and ATL7 topologies as function of the dimension of the destination group. For ATL7 H_m is set to 8 (that is a sufficient value for this topology [4]), while for TOP300 it is set to 5 (also sufficient because of the smaller diameter). The case of the optimal approach is not plotted: it grows more exponentially with $|M|$. For RCOM we display: the total time (RCOM), the times of the collection (.RC) and matching (.OM) procedures. The cases of IRC-PM, I-P2P and Kompella's (KOMP) algorithms are also plotted. The time of the A2ASP computation (Floyd) is separated since we assume that it can be pre-computed. As it can be noticed, it is constant since it is independent of the request parameters.

We can assess that: (i) The complexity part of RCOM due to matching becomes as more negligible as the topology grows. (ii) As expected, KOMP is lower bounded by Floyd since it implements a constrained version of Floyd. (iii) Including the A2ASP computation, RCOM has an execution time comparable to that of KOMP; without, it has almost always the lowest time. (iv) I-P2P and IRC-PM have a close behavior, and both seem to scale worst than the other algorithms with $|M|$ and the topology size. (v) An increase of M does not worsen the complexity of RCOM and of KOMP.

2) Optimality: Fig.2b displays the excess cost ratio (i.e. 1 $\rightarrow$ 100%) w.r.t. the optimal solution for TOP300. For ATL7 this could not be computed, but Fig.3b displays the excess cost w.r.t. RCOM for ATL7.

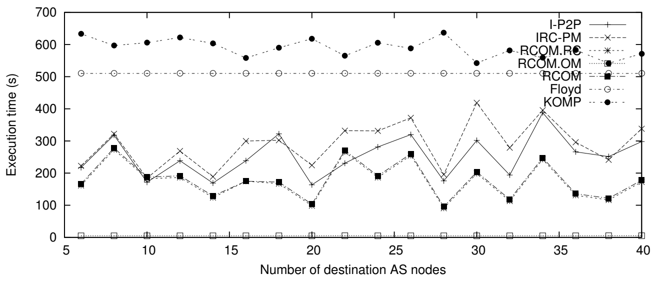


(a) TOP300: Execution times

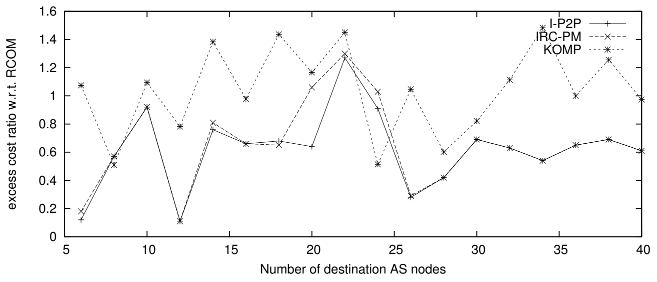


(b) TOP300: Tree cost gaps w.r.t. the optimum

Fig. 2. Results for TOP300 topology



(a) ATL7: Execution times



(b) ATL7: Tree cost gaps w.r.t. RCOM

Fig. 3. Results for ATL7 topology

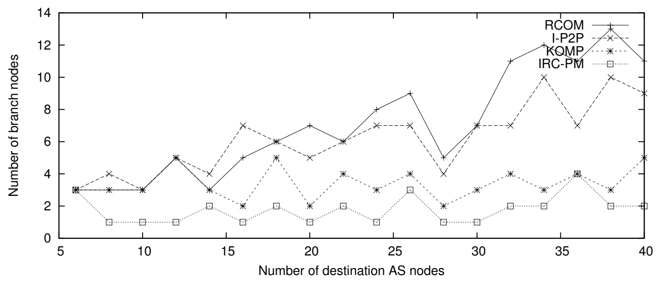
We can assess that: (i) RCOM is the best option since it yielded with TOP300 solutions with an optimality gap largely under the 10%; (ii) KOMP has always at least 50% excess cost w.r.t. RCOM; (iii) I-P2P and IRC-PM give similar solutions.

B. AS Tree Characterization

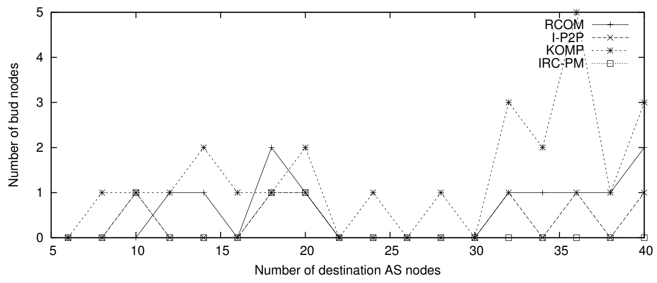
1) Node type: Fig.4 displays the number of branch and bud nodes (see Par.III-A). The ATL7 results are considered.

We can assess that: (i) for RCOM and I-P2P the number of branch nodes increases with $|M|$; (ii) the number of branch nodes is lower bounded by KOMP and IRC-PM; (iii) interestingly KOMP often gives more bud nodes than the other algorithms; (iv) on the contrary, RCOM often has more branch nodes and less bud nodes than KOMP.

(ii) and (iii) may be explained as follows. While RCOM has an unconstrained A2ASP pre-computation for projecting costs during the constrained exploration and pragmatically discarding routes, KOMP has a constrained A2ASP computation for producing a closure graph where the minimum spanning tree is computed. KOMP seems falling easier in local minima represented by longer routes and the possibility of branching at leaves is thus higher, the closure graph not being sensitive to real hop number.



(a) Branch nodes number



(b) Bud nodes number

Fig. 4. Node characterization of the solution tree

2) Tree slimness: Let the utility of a directional arc be the number of destinations it allows to serve minus one. Let the tree slimness be defined as the ratio between the sum of all these utilities and the number of directional arcs the tree is composed of. The slimness expresses how much the selected tree is exploited, or how much the selected tree has directional arcs that are very used to reach several destinations. This is not intended as an overall evaluation parameter of a tree. However, it can be seen that the less optimal a tree is, the smaller its slimness is expected to be. We are motivated in analyzing this parameter because in multi-layer network, a major application of these algorithms, a computation in one layer can be followed by computations in other lower layers along the routes chosen in the upper layer. Hence slimmer the tree is, simpler the under-layer path computation (and maybe signaling) may be in the case of multi-layer networks.

Fig.5 displays the slimness of solution trees obtained for the ATL7 graph. We can assess that: (i) RCOM offers the best slimness, i.e. the better utility of the tree; (ii) KOMP offers the worst slimness; (iii) I-P2P and IRC-PM behave better than KOMP but worst than RCOM.

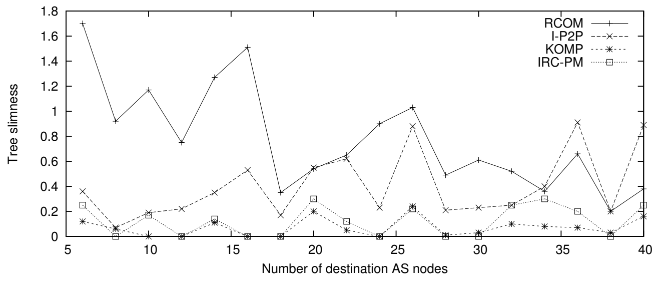


Fig. 5. Solution tree slimness as function of M

VI. Conclusions

In this paper, we studied the problem of inter-domain multipoint tunnel set-up, which is becoming interesting with the success of multimedia services. We presented an architecture based on service plane and showed how this architecture seems adapted for this problem. We then proposed heuristics for the AS tree selection under both QoS and economical constraints. We have showed that with our heuristic, pre-computation of some tasks can be performed which drastically speeds up subsequent routing computations at tunnel request arrivals.

By means of extensive simulations, we demonstrated that: (i) exploiting pre-computation, our algorithm (RCOM) is much faster that the well-known algorithms; (ii) multiple additive constraints do not affect RCOM asymptotic time complexity; (iii) it reaches often the optimality and has an optimality always largely under 10% on realistic AS graphs; (iv) it produces efficient trees w.r.t. under-layer computation issues.

References

[1] R. Zhang, J. Vasseur, "MPLS Inter-Autonomous System (AS) Traffic Engineering (TE) Requirements", RFC 4216, Nov. 2005.
[2] A. Farrel, J.-P. Vasseur, J. Ash, "A Path Computation Element (PCE)-based architecture", RFC 4655, Aug. 2006.
[3] J.-P. Vasseur et al., "A BRPC procedure to compute optimal inter-domain TE Label Switched Paths", draft-vasseur-pce-brpc-04, Mars 2007.
[4] S. Secci, J.-L. Rougier, A. Pattavina, "On the Selection of Optimal Diverse AS-Paths for Inter-Domain IP/MPLS Tunnel Provisioning", in *Proc. of 4th IT-NEWS (QoS-IP) 2008*, 13-15 Feb. 2008, Venezia, Italy.
[5] IP Sphere Framework Tech. Specification, www.ipsphereforum.org
[6] R. Douville, J.-L. Le Roux, J.-L. Rougier, S. Secci, "A Service Plane over the PCE Architecture for Automatic Multi-Domain Connection-Oriented Services", submitted to IEEE Communications Magazine.
[7] A. Farrel, I. Bryskin, *GMPLS Architecture and Applications*, Morgan Kaufmann, 2006.
[8] S. Chopra1, M.R. Rao, "The Steiner tree problem I: Formulations, compositions and extension of facets", *J. Math. Progr.*, Vol. 64 (1994).
[9] H.F. Salama, D.S. Reeves, Y. Viniotis, "Evaluation of multicast routing algorithms for real-time communication on high-speed networks", *IEEE J. on Sel. Areas in Communications*, Vol. 15, No. 3 (1997).
[10] Q. Zhu et al., "A source-based algorithm for delay-constrained minimum-cost multicasting", in *Proc. of INFOCOM 1995.*
[11] V.P. Kompella, J.C. Pasquale, G.C. Polyzos "Multicast routing for multimedia communication", *IEEE/ACM Transactions on Networking*, Vol. 1, No. 3 (1993).
[12] G Liu, KG Ramakrishnan, "A*Prune: an algorithm for finding K shortest paths subject to multiple constraints", in *Proc. of INFOCOM 2001*
[13] R.W. Floyd, "Algorithm 97: Shortest Path", *Communications of the ACM*, Vol. 5 (6) (1962)
[14] The CIDR report, http://www.cidr-report.org.