Computer Networks 114 (2017) 23–31

Contents lists available at ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet

Traffic engineering in segment routing networks

Eduardo Moreno[a,*], Alejandra Beghelli[b], Filippo Cugini[c]

[a] Faculty of Engineering and Sciences, Universidad Adolfo Ibáñez, Santiago, Chile
[b] Faculty of Engineering and Sciences, Universidad Adolfo Ibáñez, Viña del Mar, Chile
[c] Consorzio Nazionale Interuniversitario per le Telecomunicazioni (CNIT), Pisa, Italy

ARTICLE INFO

Article history:
Received 16 March 2016
Revised 9 December 2016
Accepted 11 January 2017
Available online 12 January 2017

Keywords:
Segment routing
Integer Linear Programming
Heuristic

ABSTRACT

Segment routing (SR) has been recently proposed as an alternative traffic engineering (TE) technology enabling relevant simplifications in control plane operations. In the literature, preliminary investigations on SR have focused on label encoding algorithms and experimental assessments, without carefully addressing some key aspects of SR in terms of the overall network TE performance.

In this study, ILP models and heuristics are proposed and successfully utilized to assess the TE performance of SR-based packet networks. Results show that the default SR behavior of exploiting equal cost multiple paths (ECMP) may lead to several drawbacks, including higher network resource utilization with respect to cases where ECMP is avoided. Moreover, results show that, by properly performing segment list computations, it is possible to achieve very effective TE solutions by just using a very limited number of stacked labels, thus successfully exploiting the benefits of the SR technology.

1. Introduction

The Segment Routing (SR) technology has been recently introduced to enable effective traffic engineering (TE) while simplifying control plane operations [1,2]. SR can be operated in packet networks supporting Multiprotocol Label Switching (MPLS). In particular, according to SR, packet flows are enforced through a specific path by applying, at the ingress node, a specifically computed stack of segment identifiers (SIDs). The stack of SIDs, called *segment list*, corresponds to the *stack of labels* in the MPLS architecture. In principle, only the top SID in the list is considered by transit nodes to perform packet forwarding. In particular, each packet is forwarded along the shortest path toward the network element represented by the top SID. For instance, a SID can represent an Interior Gateway Protocol (IGP) prefix which identifies a specific router, such as the IGP router ID (called IGP-Node Segment in the context of SR [1]).

Differently with respect to traditional MPLS networks, SR maintains per-flow state only at the ingress node, where the segment list is applied. Therefore, no signaling protocol (e.g., Reservation Protocol with traffic engineering extensions – RSVP-TE) is required to populate the forwarding table of transit nodes. This way, a simplified control plane is employed, just relying on the IGP that is properly extended to advertise SIDs [3]. Scalability is significantly improved, also because transit nodes do not have to maintain MPLS Label Switch Paths (LSPs) state information.

To fully exploit the SR benefits, it is necessary to efficiently compute the segment list to be applied on the ingress node. Such computation, provided by a Path Computation Element (PCE) possibly located within a Software Defined Network (SDN) Controller, has to be carefully performed to achieve effective TE solutions in the whole network.

Thus, in addition to traditional objective functions and constraints that characterize current MPLS TE solutions (e.g., minimization of the maximum link utilization subject to the link capacity within the whole network), the segment list computation has to take into account specific constraints and additional objective functions.

First, each path has to be encoded as a combination of one or more shortest segments.

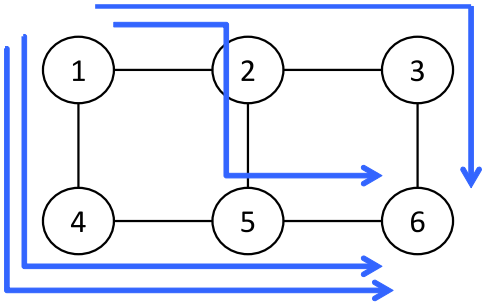
Second, since currently deployed MPLS equipments do not support a large stack of labels, path encoding has to consider the constraint on the maximum number of stacked SIDs, called *segment list depth* (SLD). Today's MPLS routers typically support SLD values in the range between 5 and 8 labels, determined by the internal forwarding engine (i.e., ASIC).

Third, since the segment list introduces packet overhead, path encoding has to minimize the introduced packet overhead.

Finally, as it will be detailed later, equal-cost multiple paths (ECMP) require specific treatment since, by default in the context of SR, they are exploited whenever available. However, to avoid packet misordering at the destination, packet inspection operations

* Corresponding author .
E-mail address: eduardo.moreno@uai.cl (E. Moreno).

http://dx.doi.org/10.1016/j.comnet.2017.01.006
1389-1286/



Segment list	Paths
6	1-2-3-6, 1-2-5-6, 1-4-5-6
2,6	1-2-3-6, 1-2-5-6,
3,6	1-2-3-6
4,6	1-4-5-6
2,5,6	1-2-5-6

Fig. 1. Example of segment list (i.e., label stack) for path 1–6 over the 2 × 3 network topology.

may be required, introducing constraints on minimum hardware requirements on SR equipments.

So far, these aspects have not been adequately investigated.

For example, the work in [4] considers the SR application in Carrier Ethernet networks. In particular, the authors propose to combine the benefits of SR with those of a software defined networking (SDN) architecture [5].

In [6], the authors proposed a SR implementation for Carrier Ethernet networks aiming at reducing the required segment list depth thorugh integrations of the segment lists at some intermediate nodes (named swap nodes).

In [7] and [8], algorithms to encode the segment lists are proposed. However, in these works, path encoding is applied only on previously identified paths and no TE solutions in the whole network are addressed.

The works in [9,10] propose two experimental implementations of SR based on an OpenFlow-based controller and on a PCE-based controller.

Finally, the works in [11] and [12] focus on SR experimental demonstrations in the context of multi-domain and reliable scenarios, respectively.

All these studies do not address the definition and evaluation of suitable algorithms for effective TE solutions in SR networks. The only work closely related to this paper is [13]. In this valuable study, a so called traffic matrix oblivious algorithm including a game theoretic like analysis for offline and online segment routing scenario is proposed. However, the reported analysis is not suitable to drive considerations on how to efficiently exploit the SR technology.

This study proposes effective ILP models and heuristics for packet networks exploiting the segment routing (SR) technology. The TE performance of SR is then assessed over a number of different network scenarios.

Obtained results allow on the one hand to assess the possible drawbacks due to the use of SR-based ECMP and on the other hand to show that efficient segment list computation can successfully provide effective TE solutions without experiencing scalability issues.

2. Segment routing

To clarify the SR behavior, a 2x3 reference network composed of six nodes and seven links is considered (see Fig. 1). The control plane consists of an IGP routing protocol extended to advertise IGP-Node Segments (i.e., router IDs [1]). Hop count is assumed as metric. No signaling protocol is configured. The data plane consists of packet nodes supporting MPLS forwarding.

A request from node 1 to node 3 is first considered. A PCE/SDN Controller computes the segment list as a combination of shortest segments. In this case, just one (unique) shortest route exists from 1 to 3, passing through node 2. The SDN Controller then computes and configures on node 1 a segment list including a single SID (i.e., a single label) representing destination node 3. Node 1 then pushes label SID 3 and sends packets towards the shortest route, i.e. along link 1–2. Node 2, by just elaborating label SID 3, is able to forward the packet along the shortest route towards node 3, i.e. on link 2–3, successfully reaching the destination where the label is popped.

To detail the case where equal cost multiple paths (ECMP) are present, or specific strict routes need to be selected, a second request from node 1 to node 6 is here considered. In this case, there are three equal cost routes (see Fig. 1): 1–2–3–6, 1–2–5–6 and 1–4–5–6. In this case, following the default SR behavior, if a single label SID 6 is pushed at node 1, all three routes are exploited. In particular, node 1 splits the traffic between link 1–2 and 1–4. Packets reaching node 4 are then forwarded towards node 6 along the route 1–4–5–6. Instead, packets reaching node 2 are further split between link 2–3 and link 2–5, before arriving at the destination 6. In case four units of traffic are generated from node 1, given the split operated at node 1, effective load balancing is actually performed on links 1–4 and 1–2, each carrying two units of traffic. Then, load balancing is further performed at node 2, obtaining just one unit of traffic on links 2–3 and 2–5. However, given the considered topology, the traffic in the network then recombines in an unbalanced way: the traffic entering node 6 is composed by three units forwarded by node 5 and just one by node 3. That is, exploiting ECMP, traffic load in the network is then distributed in a way that strongly depends on the actual traffic matrix and topology, potentially driving to ineffective TE solutions.

In general, the default SR behavior exploiting ECMP may need to be avoided if:

1. traffic distribution in the network leads to unbalanced situations and/or traffic congestions;
2. some routes present inadequate quality of service (e.g., excessive latency);
3. the forwarding device is not able to guarantee per-flow forwarding. Indeed, traffic split among ECMP needs packet inspection operations (e.g., at the TCP/UDP level) to perform per-flow forwarding and avoid packet mis-ordering at the destination. That is, when ECMP are exploited, the top label is not sufficient to determine the forwarding action and adequate hardware capabilities are needed to operate packet inspection at the wire speed. In case such performance is not available on some routers, it is recommended to configure just strict SR routes and avoid the use of ECMP.

In the following, the cases where ECMP is avoided are discussed with reference to the example above of Fig. 1.

If path 1–4–5–6 needs to be specifically selected, a stack of labels is required having SID 4 as top label (to be popped by node 4) and SID 6 as bottom of the stack (to become top one after node 4).

In case path 1–4–5–6 needs to be specifically avoided, and ECMP can be exploited on paths 1–2–3–6 and 1–2–5–6 the stack of labels requires SID 2 as top label (to be popped by node 2) and SID