Noname manuscript No.
(will be inserted by the editor)

A Bi-objective Optimization Model for Segment Routing Traffic Engineering

Antonio José Silvério · Rodrigo S. Couto · Miguel Elias M. Campista · Luís Henrique M. K. Costa

The final publication is available at Springer via
https://doi.org/10.1007/s12243-022-00907-w

Abstract The number of tunnels configured and state kept in IP/MPLS backbones depends on the number of flows and traffic engineering requirements. Segment routing automates tunnel configuration and reduces state in the network, based on the concept of segments: subpaths of the graph. A flow can be defined using only one segment if the route matches the shortest path computed by the IGP, while this number grows with the need for different subpaths. As a consequence, there is a trade-off between traffic engineering and the number of segments used, which translates to header overhead and state in routers. The challenge then is to have flows sharing as many segments as possible. We advance the state-of-the-art with a two-step bi-objective optimization model to reduce the number of configured segments, considering two traffic engineering requirements, load balancing and latency. Our results show that, as we increase the number of flows in the network, the number of configured segments also increases, then stabilizes regardless of the number of additional flows. Hence, using a real telecommunication network, we show that we can meet traffic engineering requirements with less than 22% of the total number of states as compared to the usual case of IP/MPLS backbones.

Keywords Traffic Engineering · Multiprotocol Label Switching · Segment Routing · Network Optimization

1 Introduction

Since the late 90s, the Multiprotocol Label Switching (MPLS) has been intensively deployed by telecommunication operators to provide connectivity to

Antonio Silvério · Rodrigo Couto · Miguel Campista · Luís Costa.
Programa de Engenharia Elétrica (PEE), COPPE, Universidade Federal do Rio de Janeiro (UFRJ), RJ, 21941-972, Brazil
E-mail: {silverio, rodrigo, miguel, luish}@gta.ufrj.br.

their customers, fulfilling Service Level Agreements (SLAs) [8]. The SLAs define requirements on the quality of the provided networking services, often measured with metrics such as the fraction of packets lost, latency, jitter, and throughput. Packet forwarding inside the operator network has an evident impact on the Quality of Service (QoS) as it can affect all those metrics. MPLS came into play by introducing an efficient mechanism for packet forwarding based on virtual circuit switching, using labels to identify virtual circuits. Bypassing hop-by-hop routing decisions of IP, MPLS eases QoS provision. Moreover, compared with IP backbones, where packet forwarding is based on table lookup for best matching prefix, MPLS packets are forwarded faster based on label switching. MPLS tunnels are also used for Traffic Engineering (TE) purposes. The configuration of multiple tunnels between source-destination pairs allows network optimization, fault tolerance, and service differentiation, taking the network status and flow requirements into account [4]. These benefits are achieved by using, respectively, load balancing, backup tunnels, and traffic steering. From the network operator viewpoint, however, the challenge is to configure and maintain a huge number of tunnels to answer to their clients' needs.

The configuration effort of MPLS tunnels is proportional to the traffic demand and topology changes, as well as the number of metrics and traffic policies. As the number of tunnels is large, tunnel configuration is typically conducted using offline planning tools [22]. Even though these tools help administrative tasks, they are still rudimentary in terms of scale, also given that changes in the traffic matrix and failures in the network are frequent. One alternative solution, often used, is the automatic configuration of tunnels in a full mesh. The inconvenience of this solution is the maintenance of larger Forwarding Information Bases (FIB) in MPLS routers, as well as higher control overhead to set up tunnels, i.e., more LDP (Label Distribution Protocol) and RSVP-TE (Resource Reservation Protocol with Traffic Engineering) traffic [1]. Today, traffic engineering for telecommunication operators is still a configuration challenge.

Segment Routing (SR) [10] is a technology that reduces the number of states stored at the forwarding nodes. To this end, SR relies on source routing [10] and on Software Defined Networks (SDN) [9] to concentrate path computation in a controller located at the network edge [6, 24]. The controller thus configures segments to steer traffic inside the network. This contrasts with the traditional IP/MPLS technology in which core nodes have to store states on FIBs and compute paths on a per-flow basis. The SR architecture supports both MPLS and IPv6 data planes [10]. In this work, we consider an MPLS data plane, where each path is encoded as a stack of segments and each segment is identified by a label. It is worth mentioning that the packet overhead for a Segment Routing ID (SID) with MPLS dataplane is equivalent to an MPLS label, i.e., 4 bytes per label. The segment routing technology uses the label stack from MPLS to address more segments in order to steer traffic to a path considering traffic engineering (SR-TE path). A segment can then be understood as a subpath, i.e., a path containing a subset of nodes and links

computed at the SDN controller. Each segment end-point must process the packet to pop up its identifier from the stack before forwarding it [11].

One interesting feature of SR is the possibility to improve the network performance by reducing the number of segments configured on each path considering traffic engineering parameters and network resource bottlenecks. This, however, raises a trade-off between the number of segments and effective traffic engineering. The fewer segments, the fewer number of alternative paths and fewer choices for steering traffic flows according to TE parameters. Hence, an optimization problem should address this trade-off, by sharing segments among different data flows while meeting TE requirements.

In this paper, we formulate an optimization model to choose paths and assign segments to improve load balancing and network latency, two key aspects of traffic engineering. We employ a bi-objective optimization differently from the literature, which considers only a single TE objective function [2, 23, 25, 3, 13]. Our model also minimizes the number of states in the network, by sharing segments among flows. We thus formulate a two-step optimization problem to distribute data flows. In the first step, a bi-objective linear programming model finds the best path between source-destination pairs taking network load and latency into account. In the second step, a single-objective linear programming model is used to minimize the number of segments of each path found in the first step. We evaluate our model using three realistic network topologies. Our results show that, as we increase the number of flows in the network, the number of configured segments also increases, then stabilizes regardless of the number of additional flows. In contrast, with MPLS, the number of tunnels increases linearly with the number of flows, adding more states in the network than our proposal. Hence, we show that the configuration proposed by our model scales with the number of flows in the network. Moreover, we study the dependency between traffic engineering requirements and the number of segments per flow. Our results show that a large number of traffic engineering configurations can be infeasible if network paths employ a small number of segments (i.e., less than the optimal number proposed by our problem). Finally, we apply our optimization model to an operator network, with real traffic demands, showing its actual network impact.

This paper is structured as follows. Section 2 presents the state of the art on segment routing optimization. Section 3 describes the basic concepts of segment routing with traffic engineering. Section 4 presents our proposed optimization model, while the results are shown in Section 5. Finally, Section 6 concludes the paper and identifies future work.

2 Related Work

Many papers related to segment routing optimization try to set an arbitrary limitation on the number of segments used for each flow, producing k-segment paths [2, 23, 25]. Thus, segment routing optimization is usually modeled as a

multi-commodity flow problem, where a fixed value for the maximum number of segments is set.

Bhatia *et al.* [2] address segment routing optimization, considering offline and online cases. For the offline case, they propose a routing algorithm which uses a heuristic based on game theory to derive the segment routing parameters from the traffic flow matrix. For the online case, they propose an algorithm where demand requests arrive at one time and the segments previously calculated are chosen to keep network load balance. Bhatia *et al.* try to use routes with at most two segments (i.e., 2-segment paths). They also consider the split of traffic flows using equal cost multipath (ECMP) routing, a native functionality of segment routing. Their main conclusion is that, considering a single objective for balancing network load, using 2-segment paths is almost as efficient as using paths with an unlimited number of segments.

Schuller *et al.* [23] propose a two-step model for an IP backbone network with real-world topologies and traffic demands. The goal of the first step is to distribute the load with a 2-segment path using the same model as [2], for a multi-commodity flow problem. In the second step, these results are used in a model to minimize the number of segment routing tunnels. Two variants of the second step are defined: one tunnel per demand and multiple tunnels per demand, which allow traffic splitting. The proposed method focuses only on optimizing the maximum link utilization and SR-tunnels using the 2-segment restriction. Schuller *et al.* conclude that a 2-segment approach can be used from a network operator perspective, although network utilization increases specially when demands are split.

Cianfrani *et al.* [3] consider a network transition scenario. Given a pure IP network, they choose which nodes have to be upgraded to segment routing. Hence, they consider a hybrid topology, where part of the network has segment routing enabled and the other one consists of a pure IP network. The results show that the proposed optimization can design a hybrid topology with a maximum link utilization close to that obtained in a full SR-enabled network.

Hartert *et al.* [13] propose a hybrid constraint programming framework to be used on segment routing traffic engineering. This work focuses on finding an SR-path with TE constraints for each demand, where link utilization is minimized. Segment routing paths are computed and assigned for each demand. Results with large real Internet Service Provider (ISP) topologies show that this approach can find a near-optimal solution. In some of these topologies, however, the solution is not feasible because some links are overloaded when using ECMP.

The aforementioned models are restricted to one TE metric. In this work, we consider TE by optimizing both latency and network load. To this end, we propose a two-step optimization model. In the first step, objective functions related to traffic engineering metrics are taken into account in a bi-objective linear programming model, giving the best paths for TE. In the second step, a single objective linear programming model is used to minimize the number of segments matching the best paths found in the first step. Moreover, differently

from the literature, our approach does not limit the number of segments per path, allowing telecommunication operators to be closer to their SLAs.

3 Segment Routing with Traffic Engineering

The key idea of segment routing is to rely on source routing to use paths different from those computed by IGP protocols [20]. Within SR, the path is defined as a sequence of segments encoded on the packet header at the ingress edge router. The path is computed by the SDN controller taking traffic engineering requirements into account. In SR, each segment identifies a subpath towards an intermediate node in the network. Hence, the sequence of segments builds a complete path. The idea is similar to the loose source routing option of IP: in SR, segment identifiers replace IP addresses.

Segment routing defines different types of segments. Segments may define instructions for a packet to follow a path; they may also define types of service [10]. In our work, we only consider segment types used for constructing routes: node segments and adjacency segments. A node segment corresponds to the shortest path to a given destination, as computed by the IGP. Hence, the node segment to destination node v_i as seen by node v_j, denoted by $ns_{(v_j,v_i)}$, is the shortest path from v_j to v_i. Figure 1 illustrates the node segment to edge node v_2, as seen by v_1. In this example, the node segment $ns_{(v_1,v_2)}$ is the shortest path computed with the IGP using the number of hops as the routing metric. The adjacency segment, on the other hand, is equivalent to a link to a neighbor in the network topology. Hence, node v_i has an adjacency segment to a node v_j, denoted by $as_{(v_i,v_j)}$, if there is an edge between v_i and v_j. The combination of adjacency and node segments allows arbitrary paths construction, which do not necessarily match the shortest ones. This property is useful for traffic engineering. For example, in Figure 1, the edge node v_3 establishes an alternative path to the edge node v_4. Nevertheless, different from the shortest path computed by the IGP, which minimizes the number of hops, node v_3 deviates from the shortest path using an adjacency segment at the core node v_6 to privilege link capacity, represented in this figure as an integer label on top of each link. Hence, the edge node v_3 uses one node segment to the core node v_6, an adjacency segment to v_8, and another to the destination node v_4, instead of using a single node segment to v_4. The downside of using multiple segments is packet header overhead, which can be attenuated by minimizing the maximum number of segments per path. This reduces the number of segments installed in the network and, consequently, the states stored at the network edges.

Segments are pushed into the packet header (stacked) at the entry of the network, by an edge node. They are popped up as the packet reaches segment intermediate end-points towards the destination. Each segment has an ID (Segment IDentifier - SID) represented as a label in MPLS networks [19] or an address in IPv6 networks [21]. In IPv6, source routing is possible using the corresponding extension header [21]. Figure 1 shows a path from the edge node

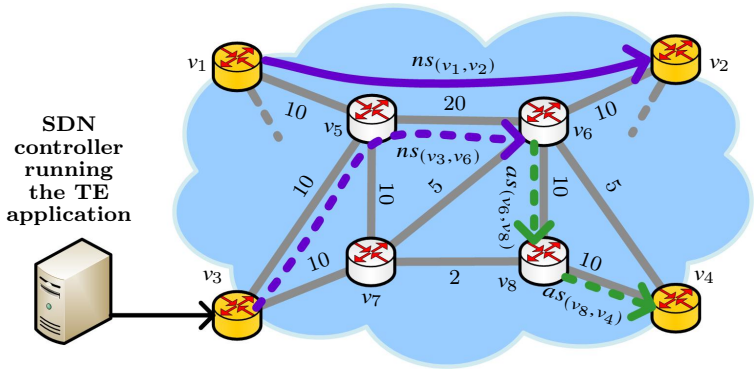


Fig. 1 Path connecting the edge nodes v_1 and v_2 using a single node segment $ns_{(v_1,v_2)}$, which matches the shortest path computed by the IGP; and the path connecting the edge nodes v_3 and v_4 using a combination of one node segment $ns_{(v_3,v_6)}$ and two adjacency segments $as_{(v_6,v_8)}$ and $as_{(v_8,v_4)}$. This combination allows the configuration of an alternative path using link capacity as the routing metric instead of the number of hops used by the IGP, for example. Segments, as well as paths, are chosen by the traffic engineering application running on the SDN controller.

v_3 to the edge node v_4, which would require three labels in an MPLS network or three addresses with IPv6. One label would be required to represent the node segment $ns_{(v_3,v_6)}$, another to the adjacency segment $as_{(v_6,v_8)}$, and a last one to the adjacency segment $as_{(v_8,v_4)}$.

Combining node and adjacency segments allows traffic engineering. The choice and installation of these segments are generally performed by SDN controllers at the network edge [10, 26]. This alleviates intermediate routers because they neither maintain states in FIBs nor overload the network with control traffic (e.g., using RSVP-TE protocol). At the network edges, ingress routers classify the incoming packets and install on their headers the corresponding path computed by the SDN controller. Traffic engineering is then implemented with segment routing and SDN. Moreover, the central configuration and maintenance can improve predictability and speed up network convergence in case of failures, topology or traffic matrix changes [6]. For example, Figure 1 shows that the SDN controller combines node and adjacency segments to establish alternative paths from v_3 to v_4, which can be different from the path computed by the IGP. As a consequence, we can increase the residual capacity on all network links and reduce the end-to-end latency.

In summary, the IP/MPLS technology uses a distributed control plane with protocols (LDP and RSVP-TE) to automatically establish full-mesh TE tunnels in all network routers. This strategy results in considerable high complexity for the configuration and the operation of the service provider's network. An alternative in MPLS to mitigate the amount of full-mesh tunnels is to configure MPLS PP tunnels manually or aided by SDN controllers. This solution, however, maintains the complexity of label distribution protocols. Using segment routing, SDN controllers also centralize the control plane, even though the data plane is still MPLS. This permits more control and allows more traffic

steering actions using segments, which simplifies the network operation and configuration without the need for MPLS control plane protocols.

4 Proposed Model

The number of segments configured for a flow can vary from one to the number of links in the chosen path. The first case occurs when the path is exactly the same as chosen by the IGP. The latter case occurs when all subpaths are composed of a single link, i.e., the path is exclusively composed of adjacency segments. Since flows can share segments, the worst case occurs when all the possible adjacency and node segments are employed, which corresponds to the number of edges in the network. Note that if the number of flows increases, the number of segments can increase until all possible adjacency and node segments are used. After that, more flows are configured without the need to configure new segments. As a consequence, we conclude that segment routing scales with the number of flows. On the other hand, even though SR reduces the scalability problem of MPLS, it is of interest to reduce the number of segments configured in the network. More segments used means larger forwarding information base (i.e., SR-FIB - Segment Routing FIB) in edge routers, with a negative impact on performance [3]. Considering that, our model proposed in Section 4 chooses traffic engineering paths in the network and their associated segments, minimizing the number of segments. Additionally, the proposed model does not consider ECMP. We assume an SR-TE use case, where an SDN controller steers the traffic to an explicit path. This approach is suitable for frequent changes and different impairments on the network topology of service providers [3].

The entry of the proposed model is a set of flows and corresponding traffic demands. The output is the sequence of segments to be used for each flow. In other words, the problem is to choose the path for each flow out of those possible and to configure each path as a sequence of segments. The proposed optimization problem is solved in two steps. In the first step, paths for each flow are computed. The computation of these paths minimizes the load on the network links and optimizes traffic engineering (TE) cost. Load minimization is a common approach adopted by the segment routing literature [2, 23, 25]. Regarding TE cost, we employ in this work latency minimization, which is an important approach to providing critical telecommunication services [16]. In the second step, we choose the segments to implement the paths computed in the first step. The second step aims to reduce the number of segments configured in the network.

4.1 First optimization step

The optimization model of the first step is bi-objective [17, 15] and computes a set of paths for each flow in the network. The Mixed Integer Linear Programming (MILP) problem is formulated as follows. The notations used in

the formulation of the first step are presented in Table 1. To simplify the notation, the nodes v_i are represented using only the index i. Hence, we represent as (i, j) the flow from node v_i to node v_j. In addition, we represent as (m, n) the link between node v_m and node v_n.

$$f_1(x) \Rightarrow \text{minimize } \alpha_{max}. \tag{1}$$

$$f_2(x) \Rightarrow \text{minimize } \sum_{(i,j)\in F} \sum_{(m,n)\in E} C_{TE}^{mn} x_{ij}^{mn}. \tag{2}$$

$$\sum_{n\in V \mid (m,n)\in E} x_{ij}^{mn} - \sum_{n\in V \mid (m,n)\in E} x_{ij}^{nm} = \begin{cases} 0 & \text{if } i, j \neq m \\ 1 & \text{if } i = m \\ -1 & \text{if } j = m \end{cases} \quad \forall (i,j) \in F, m \in V. \tag{3}$$

$$\alpha_{max} - \sum_{(i,j)\in F} \frac{f_{ij} x_{ij}^{mn}}{CAP^{mn}} \geq 0 \quad \forall (n,m) \in E. \tag{4}$$

$$0 \leq \alpha_{max} \leq 1. \tag{5}$$

$$x_{ij}^{mn} \in \{0,1\} \quad \forall (i,j) \in F, (n,m) \in E, \alpha_{max} \in \mathbb{R}. \tag{6}$$

The first objective, represented by Equation 1, minimizes the maximum link load. The load of a given link, a real number between 0 and 1, is defined as the amount of traffic transported by this link divided by its capacity. The variable α_{max} is defined as the maximum link load in the network, considering all links. As a consequence, Equation 1 tries to minimize the worst-case link load. The second objective, represented in Equation 2, minimizes the TE cost in the network. We define the TE cost as the sum of the latency of the used network links. The decision variables are given by x_{ij}^{mn}, which are Boolean variables that indicate the presence/absence of traffic flow (i, j) over the edge (m, n). The constraint described in Equation 3 guarantees network flow conservation for the choices of x_{ij}^{mn}. Equation 4 evaluates α_{max}, whereas Equation 5

Table 1 Notations used in the first step.

Notation	Description	Type
F	Flow Matrix	Set
V	Nodes	Set
E	Links	Set
f_{ij}	Traffic flow $(i, j) \in F$	Parameter
CAP^{mn}	Capacity of link $(m, n) \in E$	Parameter
C_{TE}^{mn}	TE cost of link $(m, n) \in E$	Parameter
α_{max}	Maximum link load	Variable
x_{ij}^{mn}	Presence of traffic in the link $(m, n) \in E$ corresponding to the traffic flow $(i, j) \in F$	Variable

guarantees that link capacities are respected. Finally, Equation 6 defines the domain of each variable.

To solve the bi-objective function, we adopt a scalarization technique [7], which transforms multi-objective problems into single-objective ones. This is done by combining Equations 1 and 2 to form a global criteria harmonization solution [7, 14], given by Equation 7. In this strategy, the problem is executed considering each of the single-objective functions separately. Then, using the obtained values, the same problem is executed considering a global criteria function. This way, firstly only Equation 1 is considered and the optimal f_1^* is obtained. Then, only Equation 2 is considered and the optimal value of f_2^* is found. Finally, the two obtained values are used to form Equation 7, where $f_1(x)$ and $f_2(x)$ correspond, respectively, to Equations 1 and 2. Thus, Equation 7 becomes the objective function of the new problem to be solved, which has the same constraints as the previously formulated problem. The bi-objective programming method obtains a minimal set of Pareto-optimal solutions which meet the single objective functions. A Pareto-optimal solution is the best one to comply with all criteria in objective functions and is strictly the best in at least one criterion. The Pareto-optimal solution defines an efficient frontier, close to the optimal solution of each function independently [18].

$$f(x) = \frac{f_1(x) - f_1^*}{f_1^*} + \frac{f_2(x) - f_2^*}{f_2^*}. \tag{7}$$

4.2 Second optimization step

The second step consists of choosing which segments are needed to configure each path computed in the first step. The choice is performed to minimize the amount of configured segments. The problem receives the paths of the first step and the set of all possible segment sequences which can correspond to each path. The output of the problem is the indication, for each path, of the sequence of segments to be used. The notations used in this formulation are presented in Table 2. The problem is then modeled as a matching problem [12] between the paths in $\boldsymbol{C}$ and the possible sequences of segments in $\boldsymbol{B}$. Our model assumes that the ECMP functionality is turned off on the underlying routing protocol. The Integer Linear Programming (ILP) problem is presented next.

$$f_3(x) \Rightarrow \text{minimize} \sum_{k \in S} y_k. \tag{8}$$

$$\sum_{j \in B \,|\, B_{ij}=1} z_{ij} = 1 \quad \forall i \in C. \tag{9}$$

$$\sum_{i \in C} \sum_{j \in B \,|\, B_{ij}=1, H_{kj}=1} z_{ij} - y_k |C| \leq 0 \quad \forall k \in S. \tag{10}$$

$$z_{ij} \in \{0,1\} \quad \forall i \in C, j \in B, y_k \in \{0,1\} \forall k \in S. \tag{11}$$

The objective function, given by Equation 8, minimizes the number of segments. Equation 9 guarantees that each path from C set, evaluated in the first step, uses one and only one sequence of segments from B. The set B contains all possible sequences of segments. Equation 10 is used to compute y_k, indicating if a segment k from S set is used or not in any path. This is the main variable for minimizing the number of segments. Equation 11 shows the domain of y_k and z_{ij}, which denote if the path i from C set uses or not a sequence j of segments from B set. The following section evaluates our problem using different network topologies.

5 Results

We use CPLEX to implement the model described in Section 4, with three example network topologies. The first one, called US Domestic, is available in the Cisco planning tool WAE Design. It represents a topology of a domestic telecommunication operator in USA. This network is composed of 11 routers and 34 links. The WAE Design tool also specifies 92 traffic flows for this network as well as their demands. The other two topologies, Renater and Géant, are research and education networks in France and Europe, respectively. These topologies are described in [5]. Renater has 41 nodes and 58 links while Géant has 48 nodes and 67 links. The links in Renater have a capacity from 1 Gb/s to 30 Gb/s. The links in Géant have a capacity from 1 Gb/s to 120 Gb/s. As the description of these topologies does not specify their flows, we randomly generate these flows as detailed next. As we used CPLEX to find the optimal solution, we did not tackle complexity issues as this is intrinsic to the algorithm deployed by the solver to handle MILP problems.

Table 2 Notations used in the second step.

Notation	Description	Type
S	Possible segments	Set
C	Paths computed in the first step	Set
B	Segment sequences than can implement to the paths computed in the first step	Set
B_{ij}	Indicates if a sequence of segments $j \in B$ corresponds to the path $i \in C$	Parameter
H_{kj}	Indicates if a sequence of segments $j \in B$ contains segment $k \in S$	Parameter
z_{ij}	Indicates if the path $i \in C$ uses the sequence of segments $j \in B$	Variable
y_k	Indicates if the segment $k \in S$ is used by some path in the network	Variable

5.1 Objective Function Evaluation

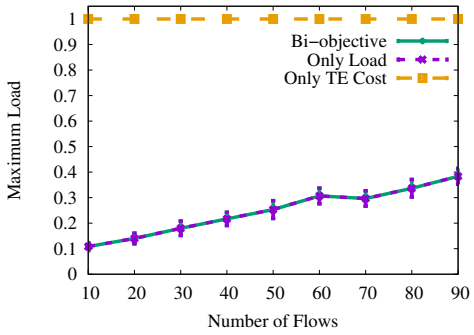
This section shows the values of the objective functions (i.e., maximum load and TE Cost) obtained in the two optimization steps. To perform this evaluation, the problem is executed for different numbers of flows inserted in the network. For each sample of US domestic, a given number of flows is randomly chosen from the 92 total flows. In the case of Géant and Renater, for each flow in a sample, we randomly choose a source-destination pair and set the flow demand as 100 Mb/s. On the one hand, for the US Domestic topology, we used the WAE Design tool to specify 92 traffic flows with different demands. On the other hand, for Géant and Renater topologies, we had to randomly generate traffic flows as their description does not specify traffic flows. This demand represents 10% of the lowest capacity link in both topologies. All results are presented with sample averages and a confidence interval of 95%.

5.1.1 First Optimization Step

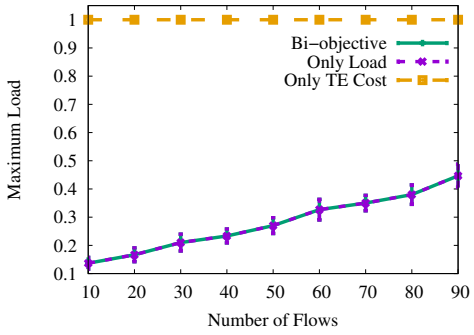
Figures 2 and 3 show the values obtained for each objective of the first optimization step, as a function of the number of flows in the network. In each figure, there is a sub-figure showing the results for a given topology. The result of the first optimization step (i.e., when Equation 7 is minimized) is presented in the curve labeled as "Bi-objective". The curve labeled as "Only Load" represents the result of a mono-objective problem where only Equation 1 (i.e., network load) is minimized. In contrast, the curve labeled as "Only TE Cost" represents the result of a mono-objective problem where only Equation 2 (i.e., TE Cost) is minimized. The last two curves serve to evaluate the impact of a bi-objective optimization. They help understanding how much we deviate from the optimal value of maximum load or TE cost if we consider both metrics in the optimization.

The results of Figure 2 for the bi-objective problem show that the maximum load α_{max} grows linearly with the number of flows, indicating that the network is well designed to support increasing demand. Furthermore, we can note in Figure 2 that, for all considered topologies, the bi-objective problem has the same maximum load result as if we only minimize the load. In other words, these results show that considering also the TE cost in the bi-objective problem does not increase the maximum load. In contrast, Figure 2 shows that we achieve a high maximum load if we exclusively minimize the TE cost.

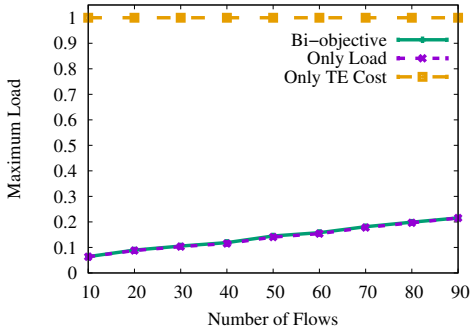
The results for the TE Cost are shown in Figure 3. This figure shows that the bi-objective problem has a TE cost close to the case where only the TE cost is minimized. More specifically, for Renater and Géant 4, 5 and 6, the TE cost of the bi-objective problem is the same as the one achieved by the problem that considers only TE cost. For US Domestic (Figure 3(c)), the bi-objective problem has a small deviation from the best possible TE cost. This means that, in US Domestic, we need to choose paths with slightly higher costs to maintain the maximum load low. Finally, the obtained results show that if we consider only the load, we can have higher TE costs in all topologies.



(a) Renater.



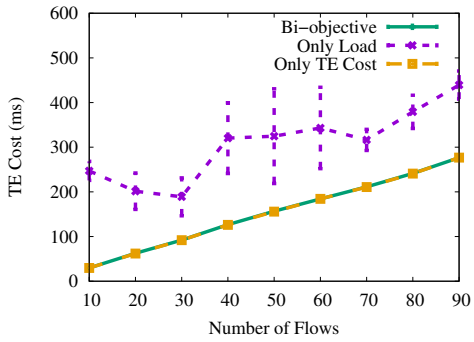
(b) Géant.



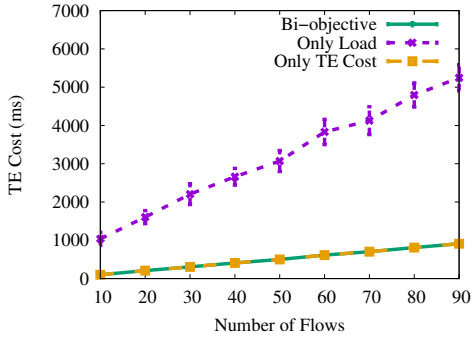
(c) US Domestic.

Fig. 2 Network Load evaluated on the first optimization step.

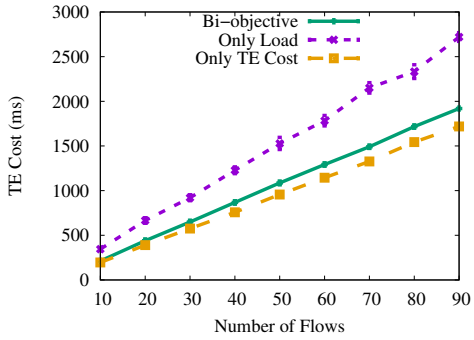
As a conclusion, the results of Figures 2 and 3 show the importance of the bi-objective problem since it achieves the best possible maximum load combined with low TE cost.



(a) Renater.



(b) Géant.



(c) US Domestic.

Fig. 3 TE cost - latency evaluated on the first optimization step.

5.1.2 Second Optimization Step

Figures 4, 5 and 6 show the results of the second optimization step for US Domestic, Renater, and Géant topologies, respectively. In each figure, the SR curve shows the number of segments chosen by our problem according to the number of flows. For comparison, we also present the number of explicit PP

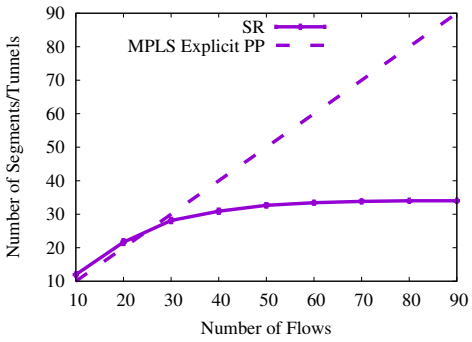


Fig. 4 Number of segments/MPLS Tunnels for US Domestic topology.

tunnels needed if MPLS is employed: in this case, one tunnel is configured per flow. This is the best case for MPLS, since the other option is to use full-mesh MPLS tunnels where US Domestic, Renater, and Géant need, respectively, 110, 2,256, and 1,640 tunnels, regardless of the number of flows. Note that the explicit PP tunnels are the output of a pure SDN control plane without segment routing. Then, the results compare the performance of an SDN control plane with and without segment routing.

Figure 4 shows that, for the US Domestic topology, the number of segments grows up to 34 and then remains constant as the number of flows increases. This is because, from a given number of flows configured, all new flows share segments with previous ones. Hence, there is no need to configure more segments. This shows that our problem can reduce the amount of state in the network for a growing number of flows. Figures 5 and 6 show for Géant and Renater that the reduction of states in the network does not occur for the considered number of flows (i.e., up to 90 flows). This means that the number of configured segments is still insufficient to enable efficient sharing between network flows. To check the number of flows at which the number of segments stabilizes, for Géant and Renater, we plot Figures 7 and 8. Those consist of the same experiment as before but using a larger number of flows. The results show that the benefit of our approach depends on the number of flows, as well as on the network topology.

5.2 Number of Segments per Flow

Using the same experiments as in Section 5.1.2, we evaluate, for each sample, the fraction of flows that employ a given number of segments. For conciseness, we present only the results when 90 flows are configured. Figure 9 shows the fraction of flows that employ a given number of segments with average values for all samples and corresponding confidence intervals. This figure shows that, for all topologies, more than 50% of the paths use up to two segments. Even though Bhatia *et al.* [2] indicate that two segments per flow are enough to

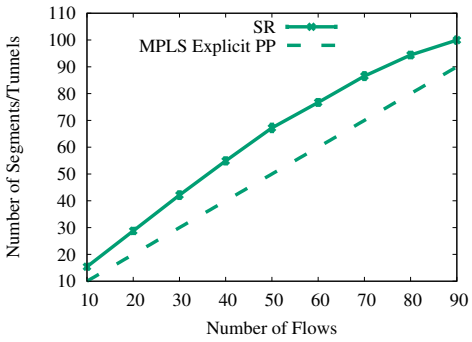


Fig. 5 Number of segments/MPLS Tunnels for Renater topology.

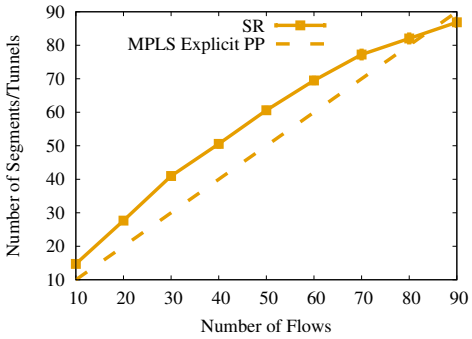


Fig. 6 Number of segments/MPLS Tunnels for Géant topology.

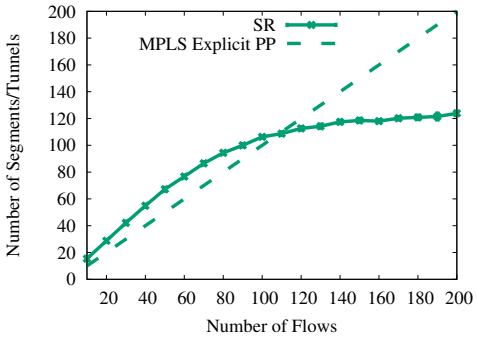


Fig. 7 Number of segments/MPLS Tunnels for Renater topology considering more flows.

meet maximum link load close to optimal, Figure 9 shows that there is a significant number of paths that need more than two segments to achieve the optimal objectives of our problem. However, operators may choose to have sub-optimal TE metrics to reduce the number of segments per flow [2]. This can be interesting to reduce the packet header and the complexity of the optimization

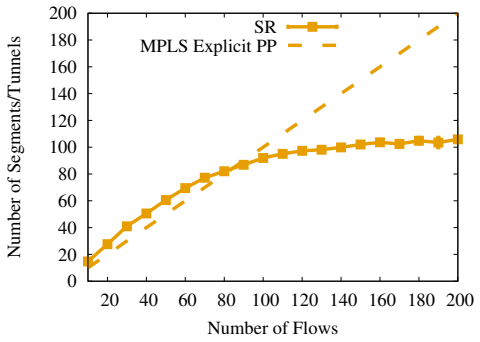


Fig. 8 Number of segments/MPLS Tunnels for Géant topology considering more flows.

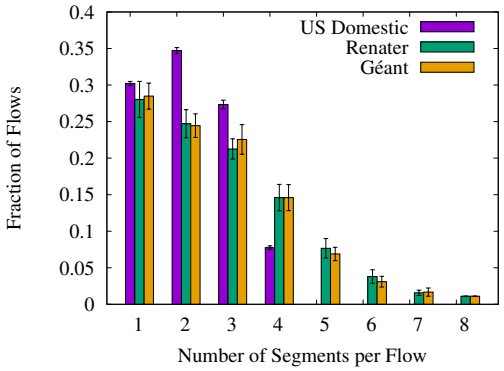


Fig. 9 Number of Segments per Flow for a scenario with 90 flows.

problem. It is important to note that, different from the literature, we consider both metrics of maximum link utilization and TE cost. A further investigation is needed to evaluate at which extent optimizing these both objectives can change the conclusions of the work conducted by Bhatia *et al.*, which considers only maximum link utilization.

5.3 Impact of the Number of Segments

In this section, we repeat the experiment of Sections 5.1 and 5.1.2, but we modify the second step of our optimization problem to limit the maximum number of segments per flow. Our goal is to complement the results of Section 5.2, measuring the impact of the number of segments per flow on TE decisions of the first optimization step. Focusing on the US Domestic topology, which is provided with real flows instead of synthetic as Renater and Géant, we limit the number of segments per flow, from one to four, and perform 30 experimental runs for each number of flows. The limit on the number of segments can make the second optimization step unfeasible, since some paths may require

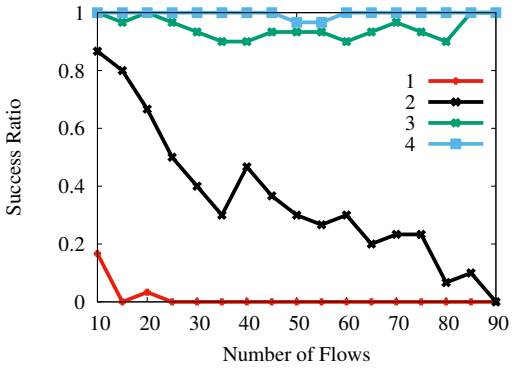


Fig. 10 Number of segments distribution per path calculated on $f_3(x)$ problem.

more segments. Hence, considering all samples for a given number of flows and the maximum number of segments, we evaluate the success ratio metric. It indicates the number of samples that have a feasible solution, divided by 30, the total number of samples. Figure 10 shows the obtained results. Each curve represents the maximum number of segments per flow imposed by the problem. We observe that few samples return a feasible solution when each path is limited to one segment. This is expected since one segment per flow corresponds to the paths configured by the IGP. Since IGP paths may be different from those required by TE, a very large number of samples will fail. The figure also shows that choosing only two segments per path has not a severe impact for a small number of flows, but it tends to increase with the number of flows. Finally, when we limit the number of segments to three, the impact is low. This can be explained since, as shown in Section 5.2, a large number of flows employs up to 3 segments.

5.4 Case Study: optimization in a real telecommunication operator

To validate the optimization problem in a real scenario, we employ the data obtained from a large-scale telecommunications operator in Brazil, Embratel[1]. The data, which is not publicly available due to privacy concerns, consists of a simplified version of their network, describing the topology of IP/MPLS core routers and flow demands. We employ two snapshots of the Embratel network, one obtained in the year 2017 and another obtained in 2019. The snapshot of 2017 has 66 nodes, 394 links and 1,130 flows. The snapshot of 2019 has 80 nodes, 508 links and 2,036 flows. We run the optimization problem of Section 4 using these two snapshots with all flow demands. It is worth mentioning that the Embratel dataset is composed of different flows with different traffic demands.

[1] `https://www.embratel.com.br/`

Table 3 shows the maximum load and TE cost obtained in the first step of the optimization problem. We also present the average TE cost, which is the TE cost divided by the total number of flows. Similarly to Section 5.1.1, we present results for the bi-objective problem as well as for the mono-objective ones, where only maximum load or TE cost are minimized. The results for both snapshots confirm the conclusions of Section 5.1.1, where we show that the bi-objective problem leads to the best possible maximum load with a small increase in TE cost.

Table 3 Results for Embratel scenario in the first optimization step.

Objective function	Snapshot	Max. Load	TE Cost	Avg. TE Cost
Bi-objective	2017	0.67	62741 ms	55.52 ms
	2019	0.46	210746 ms	103.51 ms
Only Load	2017	0.67	1732643 ms	1533.31 ms
	2019	0.46	10773114 ms	5291.31 ms
Only TE Cost	2017	1.00	62720 ms	55.50 ms
	2019	1.00	190494 ms	93.56 ms

Table 4 shows the number of segments obtained in the second optimization step. For comparison, this table also shows the number of tunnels that are needed if MPLS Explicit PP is employed instead of segment routing. The results of Table 4 confirm those of Section 5.1.2, showing that our problem provides a high reduction in the number of states in the network. For example, when employing our optimization in Embratel 2019, Segment Routing uses less than 22% of the total number of states as compared to MPLS.

Table 4 Number of segments/tunnels for the Embratel scenario, in the second optimization step.

Scenario	Segment Routing	MPLS Explicit PP
Embratel 2017	285	1130
Embratel 2019	444	2036

Finally, Figure 11 shows, for the Embratel scenario, the fraction of flows that have a given number of segments. Similarly to Section 5.2, these results also show that most flows (i.e., 66.46% for 2017 and 59.37% for 2019) use up to two segments. Nonetheless, the fraction of flows that employ more than two segments are non-negligible. Hence, as already noted in Section 5.2, providers must use a high number of segments per flow if they need to improve both load and TE cost for all data flows. Note that our model converges even considering realistic scenarios with a large number of flow demands. This is a characteristic found ata national-wide operator, which is not expected to change for different countries.

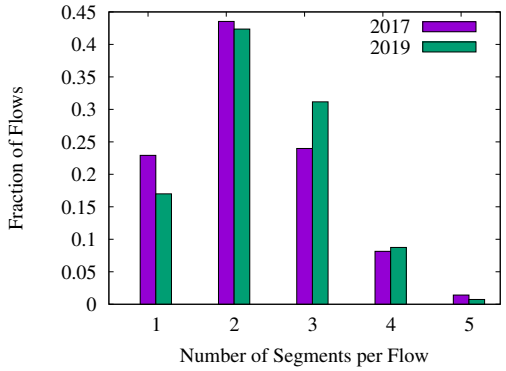


Fig. 11 Number of segments per flow for the Embratel scenario.

6 Conclusion

In this work, we have proposed a two-step optimization model that uses the concept of segment routing to find alternative paths to the shortest ones typically found by the IGP. The idea was to propose a bi-objective problem and thus ensure traffic engineering requirements while minimizing the number of segments. The results have shown that the number of segments introduces fewer states in routers. In addition, we have shown that to meet the optimal maximum link load and TE Cost, the number of segments per flow can be high. This last finding contrasts with the work conducted by Bathia *et al.* [2], which indicates that using two segments per flow is enough to meet close to the optimal maximum link load. Hence, as future work, we plan to study in more detail the impact of the number of segments per flow. For example, we can evaluate how the conclusions of Bathia *et al.*, obtained using a problem that only minimizes the maximum link load, will be affected if we consider our bi-objective problem. Also, future work can include other objectives in our optimization, such as fault tolerance.

When using ILP solvers, such as CPLEX, our problem may not scale for a network with a large number of flows or nodes. Future work thus includes proposing a heuristic to solve our problem to provide scalability.

Acknowledgements This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001. It was also supported by CNPq, FAPERJ (Prof. Rodrigo Couto would like to particularly thank for Grant E-26/203.211/2017 and E-26/201.300/2021, Prof. Miguel Campista would like to thank for Grants E-26/211.144/2019 and E-26/202.689/2018, and Prof. Luís Costa, Grants E-26/202.932/2017 and E-26/200.892/2021), and FAPESP Grant 15/24494-8.

References

1. Applegate D, Thorup M (2003) Load optimal MPLS routing with N + M labels. In: IEEE Conference on Computer Communications (INFOCOM), vol 1, pp 555–565
2. Bhatia R, Hao F, Kodialam M, Lakshman T (2015) Optimized network traffic engineering using segment routing. In: IEEE Conference on Computer Communications (INFOCOM), pp 657–665
3. Cianfrani A, Listanti M, Polverini M (2017) Incremental deployment of segment routing into an ISP network: A traffic engineering perspective. IEEE/ACM Transactions on Networking 25(5):3146–3160
4. Cisco Systems (2001) Implementing MPLS traffic engineering. Tech. rep., Cisco Systems, URL `http://www.cisco.com/c/en/us/td/docs/routers/crs/software/crs\_r3-/mpls/configuration/guide/gc39crs1book\_chapter4.html`
5. Couto RS, Secci S, Campista MEM, Costa LHMK (2015) Server placement with shared backups for disaster-resilient clouds. Computer Networks 93:423–434
6. Davoli L, Veltri L, Ventre PL, Siracusano G, Salsano S (2015) Traffic engineering with segment routing: SDN-based architectural design and open source implementation. In: European Workshop on Software Defined Networks (EWSDN), pp 111–112
7. Ehrgott M, Gandibleux X (2002) Multi Objective Combinatorial Optimization, 1st edn. Kluwer's International Series in Operations Research and Management Science
8. El-Sayed M, Jaffe J (2002) A view of telecommunications network evolution. IEEE Communications Magazine 40:74–81
9. Farhady H, Lee H, Nakao A (2015) Software-defined networking: A survey. Computer Networks 81:79–95
10. Filsfils C, Nainar NK, Pignataro C, Cardonay JC, François P (2015) The segment routing architecture. In: IEEE Global Communications Conference (GLOBECOM), pp 1–6
11. Filsfils C, Michielsen K, Talaulikar K (2017) Segment Routing Part I, 1st edn. Cisco Press, San Francisco
12. Grotschel M, Holland O (1985) Solving matching problems with linear programming. Mathematical Programming 33:243–259
13. Hartert R, Schaus P, Vissicchio S, Bonaventure O (2015) Solving segment routing problems with hybrid constraint programming techniques. In: International Conference on Principles and Practice of Constraint Programming, Springer, pp 592–608
14. Marler RT, Arora JS (2004) Survey of multi-objective optimization methods for engineering. Structural and multidisciplinary optimization 26:369–395
15. de Mello MOMC, Borges VCM, Pinto LL, Cardoso KV (2016) Improving load balancing, path length, and stability in low-cost wireless backhauls. Ad Hoc Networks 48:16–28

16. Pathak A, Zhang M, Hu YC, Mahajan R, Maltz D (2011) Latency inflation with MPLS-based traffic engineering. In: ACM SIGCOMM Internet Measurement Conference (IMC), pp 463–472
17. Pinto LL, Pascoal MMB (2010) On algorithms for the tricriteria shortest path problem with two bottleneck objective functions. Computers & Operations Research 37:1774–1779
18. Pinto LL, Fernandes KCC, Cardoso KV, Maculan N (2019) An exact and polynomial approach for a bi-objective integer programming problem regarding network flow routing. Computers & Operations Research 106:28–35
19. Previdi S, Filsfils C, Bashandy A, Gredler H, Litkowski S, Decraene B, Tantsura J (2018) Segment routing interoperability with LDP. Tech. rep., Internet Engineering Task Force (IETF), URL {https://www.ietf.org/archive/id/draft-filsfils-spring-segment-routing-ldp-interop-15.txt}
20. Previdi S, Filsfils C, Bashandy A, Gredler H, Litkowski S, Decraene B, Tantsura J (2019) IS-IS Extensions for Segment Routing. Tech. rep., Internet Engineering Task Force (IETF), URL https://tools.ietf.org/html/draft-ietf-isis-segment-routing-extensions-25
21. Previdi S, Filsfils C, Field B, Leung I, Linkova J, Aries E, Kosugi T, Vyncke E, Lebrun D (2019) IPv6 Segment Routing Header (SRH). Tech. rep., Internet Engineering Task Force (IETF), URL {https://www.ietf.org/archive/id/draft-ietf-6man-segment-routing-header-22.txt}
22. Ricciato F, Salsano S, Belmonte A, Listanti M (2002) Off-line configuration of a MPLS over WDM network under time-varying offered traffic. In: IEEE Conference on Computer Communications (INFOCOM), vol 1, pp 57–65
23. Schuller T, Aschenbruck N, Chimani M, Horneffer M, Schnitter S (2017) Traffic engineering using segment routing and considering requirements of a carrier IP network. In: IFIP Networking Conference, pp 1–9
24. Sgambelluri A, Paolucci F, Giorgetti A, Cugini F, Castoldi P (2015) SDN and PCE implementations for segment routing. In: European Conference on Networks and Optical Communications (NOC), pp 1–4
25. Trimponias G, Xiao Y, Xu H, Wu X, Geng Y (2017) On traffic engineering with segment routing in SDN based WANs. arXiv preprint arXiv:170305907
26. Ventre PL, Tajiki MM, Salsano S, Filsfils C (2018) SDN architecture and southbound APIs for IPv6 segment routing enabled wide area networks. IEEE Transactions on Network and Service Management 15(4):1378–1392