

Segment Routed Traffic Engineering with Bounded Stretch in Software-Defined Networks

Tossaphol Settawatcharawanit*, Vorapong Suppakitpaisarn†, Shigeki Yamada‡, and Yusheng Ji§

Department of Informatics, The Graduate University for Advanced Studies, Tokyo, Japan*§

Department of Computer Science, The University of Tokyo, Tokyo, Japan†

Information Systems Architecture Science Research Division, National Institute of Informatics, Tokyo, Japan*‡§

Email: {tossaphol*, shigeki‡, kei§}@nii.ac.jp, vorapong@is.s.u-tokyo.ac.jp†

Abstract—Segment Routed Traffic Engineering is emerging as an important application for network operators to manage resource utilization by using segment routing paths as candidates for route selection. In order to facilitate the network operator demands, a traffic engineering program should be fast and efficient. These two characteristics are very essential since the traffic engineering program must be invoked periodically in short intervals. The segment routing paths can be constructed by concatenating the shortest paths between two nodes such that there is a path from source to destination. We are interested in the problem to find intermediate nodes to construct segment routing paths minimizing the maximum link utilization. However, the existing approaches have the shortcomings that either they require a substantial amount of time to find a solution or they must sacrifice a considerable amount of link utilization. To address these issues, we propose to limit the number of intermediate node candidates by using a bounded stretch constraint relative to the shortest path of the source-destination pair. Then, we evaluate the computation time and link utilization against the existing work. We show that the bounded stretch constraint helps reduce the computation time while a near optimal link utilization can be achieved.

Index Terms—network optimization, segment routing, software defined networking, traffic engineering

I. INTRODUCTION

Segment Routed Traffic Engineering (SRTE) is emerging as a key mechanism for network operators to achieve efficient resource utilization by overcoming the scalability issues of Multiprotocol Label Switching Traffic Engineering (MPLS-TE) since MPLS-TE relies on a resource reservation protocol (RSVP-TE) to request a resource reservation along a routing path [1]. Consequently, the resource states must be maintained and distributed in a network. With the advancement of Software-Defined Networks (SDN), Traffic Engineering (TE) with SDN [2] has overcome the drawback of MPLS-TE by leverage the centralized controller to maintain network states [3], [4]. Similar to Label Switching Path (LSP) of MPLS and Source Routing paradigm, Segment Routing path can be constructed by combining two or more segments into an end-to-end path, and each segment is the shortest path between two nodes. The end-to-end path can be seen as a logical tunnel from ingress to egress with the segment label specified in the packet header. However, not every segment has to be specified in the packet header with a segment label since we can leverage the shortest path to specify each part of the end-to-end path

that traverses to some intermediate nodes between source and destination. Thus, we can use these intermediate nodes or a destination node as segment labels. This mechanism relaxes the restrict condition of the shortest path routing because a considerable amount of paths can be constructed with the shortest path information given by Interior Gateway Protocol (IGP) [5]. Moreover, This mechanism helps limit the overhead processing of packet header greatly since only few segment labels are still needed for constructing an end-to-end tunnel.

To leverage TE, network operators usually deploy a traffic engineering program to optimize networking resources and repeatedly execute the program in short intervals (e.g., 5 minutes) [6]–[10]. Recently, several approaches have been proposed to optimize the link utilization with SRTE [6], [11]–[13]. In this paper, we focus on minimizing the maximum link utilization, and we compare our results against the results in [12] since this work has the same assumptions as our work while other works have different assumptions. The idea is to find intermediate nodes to construct segment routing paths aiming at minimizing the maximum utilization. Unfortunately, existing approaches have many serious drawbacks either the maximum link utilization is too high, or the computation complexity is too large. We argue that an alternative approach is necessary especially in large networks. We suggest that only some important nodes should be considered as the candidate set for intermediate nodes in order to cope with the frequent change in traffic demands.

In this paper, we propose the Segment Routed Traffic Engineering with Bounded Stretch (SRTE-BS) to cope with the computation complexity of SRTE problem. In contrast to the works in [12] and [11], our approach aims to reduce the computation time of the problem by limiting the number of candidates for intermediate nodes with a bounded stretch constraint relative to the shortest path of each source-destination pair. Notice that these intermediate node candidates can be obtained in advance with only the shortest paths information.

The simulation results show that SRTE-BS can reduce the computation time by at most 99.6% while the maximum link utilization is increased by only 5% at most from the optimal link utilization in the tested scenarios.

II. SEGMENT ROUTED TRAFFIC ENGINEERING OVERVIEW

In this section, we introduce the theoretical model of SRTE. We first describe the model assumptions, and we then describe

the SRTE problem in details.

A. System Models

1) *Network Models*: Consider a network topology represented by a graph $G(\mathcal{V}, \mathcal{E})$ consisting of a set of nodes $\mathcal{V}$ and a set of links $\mathcal{E}$. Each node is controlled by a logically centralized controller which is responsible for traffic flow management and monitoring. Each edge $e \in \mathcal{E}$ is associated with link capacity $c(e) \in \mathbb{R} > 0$ and link weight $w(e) \in \mathbb{R} > 0$.

2) *Traffic Demand*: A demand of traffic from a source node $i \in \mathcal{V}$ to a destination node $j \in \mathcal{V}$, $t_{ij} \in \mathbb{R} > 0$ has to be accommodated in the network.

B. Problem Description

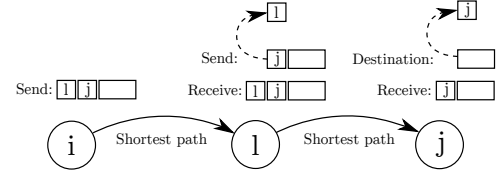
We formulate SRTE as an optimization problem with an objective to minimize the maximum utilization. This variant of TE is used in carrier networks where traffic matrix can be approximated in advanced, and the network operator aims to minimize the traffic congestion in the network.

Similar to previous works on SRTE [12], [6], [14], our proposed scheme only considers 2-Segments Routing. This assumption is not overly simplified as the performance of n -Segments Routing can only be improved marginally comparing with 2-Segment Routing [12], [15]. Moreover, there are many other reasons for choosing this design choice. For example, the level of simplicity in order to bound the amount of time for packet header processing and the hardness of SRTE with an unlimited number of segments also becomes intractable [11], [16].

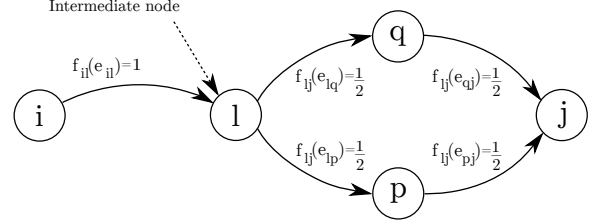
Let $\mathcal{L}$ be the set of all intermediate nodes and $\mathcal{L} \subseteq \mathcal{V}$. For each traffic demand, the traffic from a source node i to a destination node j is routed through an intermediate node or multiple intermediate nodes $l \in \mathcal{L}$. Segment Routing path from i to j can be constructed by concatenating two shortest paths: the shortest path from the source node i to an intermediate node l and the shortest path from an intermediate node l to the destination node j . Each shortest path on the segment routing path is called as a segment of the routing path. For each segment of a segment routing path, the amount of traffic demand is divided into equal proportions if multiple equal cost shortest paths are used.

Fig. 1(a) illustrates the segment routing operation. Consider a traffic demand from a source node i to a destination node j , let l be an intermediate node selected by the centralized controller for this traffic flow. Then, a source node i can send a traffic flow to an intermediate node l with segment labels of l and j . When the intermediate node l receive the first packet of the flow, the intermediate node l can be popped out its segment label. Then, the intermediate node l can reroute the traffic flow to the destination node j with segment label of the destination node j . The segment labels that need to be specified in the packet header are the only segment labels of rerouting nodes and a destination.

We denote a unit of traffic flow from i to j that resulted on a link e as $f_{ij}(e)$. If there is only one shortest path from i to j , then $f_{ij}(e) = 1$ for all links e on the path from i to j .



(a) The operation of Segment Routing



(b) A fractional flow that resulted on links when ECMP is used

Fig. 1: **Segment Routing** - (a) shows the operation of segment routing. The routing path is divided into multiple segments. Each node pops out the outer most segment label after receiving a message. (b) shows the fractions of flow on each link when the traffic flow is sent from a source node i to a destination node j through an intermediate node l . The amount of traffic flow is equally divided into the fractions when equal-cost paths exist.

However, If a link e is not on the shortest path from i to j , then $f_{ij}(e) = 0$. If there are multiple equal cost shortest paths from i to j , the amount of flow is divided equally. Thus, $f_{ij}(e)$ can be fractional for each link e on the shortest path from i to j . We denote by sp_{ij} a shortest path from i to j and $sp_{ij} \in \mathcal{SP}_{ij}$, where $\mathcal{SP}_{ij}$ is the set of all shortest paths from i to j .

Fig. 1(b) shows an example of how traffic splitting affects the traffic split ratio on links. In this example, we consider 2-segment routing for simplicity. We want to send a traffic demand from i to j , and the controller has decided to use an intermediate node l for this flow.

In order to obtain $f_{ij}(e)$ for each link $e \in \mathcal{V}$, let $R_{ij}(e)$ denote the fractions of flow on link e along the shortest paths from i to j . We denote by $|\delta_{ij}^+(u)|$ the cardinality of the set of outgoing edges from node u where u is a node on the $\mathcal{SP}_{ij}$. In other word, the number of outgoing edges from u where all outgoing edges are on the $\mathcal{SP}_{ij}$, and by $|\delta_{ij}^-(u)|$ the cardinality of a set of incoming edges to u where all incoming edges are on the $\mathcal{SP}_{ij}$. Thus, we can formalize $f_{ij}(e)$ as:

$$f_{ij}(e) = \begin{cases} R_{ij}(e) & \text{if } e \in \mathcal{SP}_{ij} \\ 0 & \text{if } e \notin \mathcal{SP}_{ij} \end{cases} \quad (1)$$

where

$$R_{ij}(e) = \begin{cases} \frac{1}{|\delta_{ij}^+(u)|} & \text{if } u = i \\ \frac{|\delta_{ij}^-(u)|}{|\delta_{ij}^+(u)|} & \text{if } u \neq i \end{cases} \quad (2)$$

Therefore, the flow $f_{ij}^l(e)$ that passes through an intermediate node l from i to j is defined as:

$$f_{ij}^l(e) = f_{il}(e) + f_{lj}(e) \quad (3)$$

where f_{il} and f_{lj} represent the fraction of flow in the first and second segment of a segment routing paths from i to j respectively.

We can now introduce the Segment Routed Traffic Engineering (SRTE) problem. We denote a decision variable x_{ij}^l as the amount of flow which is routed from i to j through an intermediate node l . Given traffic flow demand t_{ij} , the goal is to find x_{ij}^l , the amount of traffic flow from i to j passing through an intermediate node l with the objective to minimize the maximum link utilization θ . The linear programming is as follow:

$$\min \theta \quad (4)$$

$$\sum_l x_{ij}^l \geq t_{ij} \quad \forall ij \in \mathcal{V} \quad (5)$$

$$\sum_{i,j} \sum_l f_{ij}^l(e) x_{ij}^l \leq \theta \cdot c(e) \quad \forall e \in \mathcal{E} \quad (6)$$

$$x_{ij}^l \geq 0 \quad \forall ij, l \in \mathcal{V} \quad (7)$$

The objective is to minimize the maximum utilization θ of all links (4). Constraint (5) ensures that each traffic is routed through some intermediate node. Constraint (6) is a capacity constraint which ensures that sum of all the traffic routed through a link does not exceed the link capacity. Constraint (7) assures that the amount of traffic is non-negative.

III. SEGMENT ROUTED TRAFFIC ENGINEERING WITH BOUNDED STRETCH

Two of the most important characteristics for TE program are the efficiency of a solution and the computation time to obtain a solution. The reason is that an optimal solution with the best quality may not be a good solution if the computation time to obtain the solution is too long. We argue that a good solution should be a solution which can be obtained in a short time while retaining a near-optimal performance. In this section, we propose a fast and efficient approach called SRTE-BS. The SRTE-BS introduce a bounded stretch constraint to the linear programming formulation. The role of the bounded stretch constraint is to limit the number of candidates for intermediate nodes. If an intermediate node is too far away from a source node i to a destination node j by some constant value relative to the length of the shortest path, then this intermediate should not be considered as a candidate. We called the aforementioned constant value as a stretch. In other words, the length of a segment routing path from i to j through l must be less than or equal to the length of the path stretch scaled by α times the

length of the shortest path from i to j where $\alpha \geq 1$. If $\alpha = 1$, then only the shortest paths are used.

We denote $|sp_{il}|$ as the length of the shortest path from a source node i to an intermediate node l , and denote $|sp_{lj}|$ as the length of the shortest path from the intermediate node l to a destination node j . The bounded stretch constraint can be formulated as follows:

$$|sp_{il}| + |sp_{lj}| \leq \alpha \cdot |sp_{ij}| \quad \forall ij, l \in \mathcal{V} \quad (8)$$

A. Offline Computation

In this part, we observed that the bounded stretch constraint (8) can be computed offline since the bounded stretch constraint requires only the shortest paths information from the IGP protocol. Consequently, this constraint can be removed from the linear program. We denote S_{ij} as a set of candidates for intermediate nodes that satisfy a bounded stretch constraint relative to the shortest path from i to j , and $S_{ij} \subset \mathcal{L}$. The size of S_{ij} is $|\mathcal{V}|^2$ since we could compute the S_{ij} for all source-destination pairs in advanced.

Algorithm 1 presents the procedures to obtain S_{ij} . For each source-destination pair (i, j) , we select $l \in \mathcal{L}$ into S_{ij} satisfying the bounded stretch constraint (8).

Algorithm 1 Offline computation of S_{ij}

```

1:  $S_{ij} \leftarrow \emptyset$ 
2: for  $(i, j) \in \mathcal{V}$  do
3:   for  $l \in \mathcal{V}$  do
4:     if  $|sp_{il}| + |sp_{lj}| \leq \alpha \cdot |sp_{ij}|$  then
5:        $S_{ij} \leftarrow S_{ij} \cup l$ 

```

Now, we can replace an intermediate node l with a candidate for intermediate node $s \in S_{ij}$ which satisfied the bounded stretch constraint (8) in the constraints (5)-(7). Then, the constraint (8) can be eliminated from our formulation.

IV. PERFORMANCE EVALUATION

In this section, we evaluate the performance of SRTE-BS in comparison to SRTE with two metrics: (1) the computation time, and (2) the maximum link utilization.

We conduct our experiments to evaluate the validity of our proposed approaches on two network topologies from [6]: 100 nodes 572 links topology with 9,817 traffic demands, and 79 nodes 294 links topology with 6,208 traffic demands. The 100 nodes topology is a synthetic network topology, and the 79 nodes topology is a real topology from the Rocketfuel project [17]. On each topology, we use link weights, link capacities and traffic demands provided with the topologies. The traffic demands matrices are generated with the gravity model fed by exponentially distributed random variables. However, we scale the traffic demand volume by 10 in order to saturate the network links. We conduct our simulations on 2.60 GHz 16-core Xeon processor and 64 GB of memory running Ubuntu Linux Server version 16.04 with Linux kernel 4.4.0. As for linear programming solver, we use IBM ILOG CPLEX version 12.8 for all of our simulations. We ran the experiment 10 times

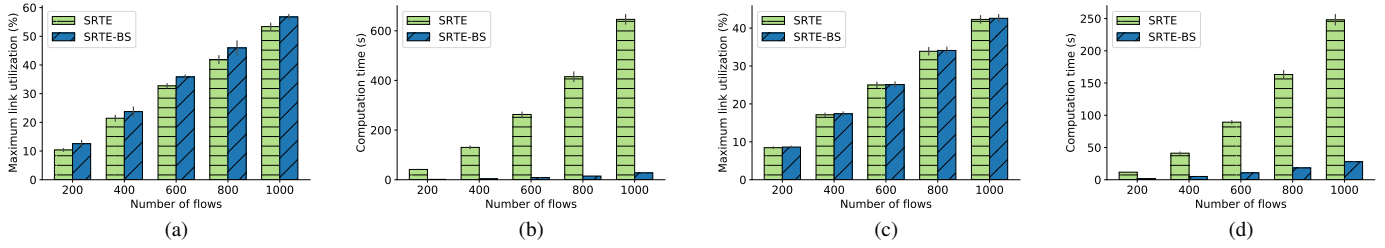


Fig. 2: Comparing with the optimal: (a) The maximum link utilization ratio in 100 nodes topology. (b) The computation time in 100 nodes topology. (c) The maximum link utilization ratio in 79 nodes topology. (d) The computation time in 79 nodes topology.

for each data points. In addition to that, we have error bars with 95 percents confident intervals on all data points. The results show that the fluctuation from the bars is minimal.

A common question that may arise is "how large the stretch size should be?". If we choose the stretch too large, then the computation time to solve the linear programming will be high, or the maximum utilization may increase if the stretch is too short. Hence, we can set the stretch size to be a small value such as 1.3, and we use this value throughout the paper. The complete evaluation of the stretch size is not presented here due to limited space.

To show the performance of the proposed SRTE-BS, we compare the SRTE-BS to the optimal solution SRTE [12]. Fig. 2(a) and Fig. 2(b) depict the maximum link utilization and the computation time of 100 nodes topology. The results show that SRTE-BS reduces the computation time by at most 99.6% and the maximum link utilization is increased by only 5% at most from the optimal link utilization in SRTE.

Fig. 2(c) and Fig. 2(d) show the maximum link utilization and the computation time of 79 nodes topology. The results show that our approach achieves almost the same value of the optimal maximum link utilization in SRTE while the computation time is reduced by at most 82% in this topology.

V. CONCLUSION

In this paper, we proposed Segment Routed Traffic Engineering with bounded stretch that relative to the shortest path of the source-destination pair to limit the number of intermediate node candidates. The simulation results show that our approach achieves a near optimal solution while the computation time is reduced by 99.6% at most. We believe that our approach is a promising solution for Segment Routed Traffic Engineering.

ACKNOWLEDGMENT

The authors would like to thank Kalika Suksomboon, Sirod Sirisup, Kensuke Fukuda, Michihiro Koibuchi, Shunji Abe, and Ying Cui for their valuable comments and suggestions.

REFERENCES

- [1] D. Awduche, J. Malcolm, J. Agogbua, M. O'Dell, and J. McManus, "Requirements for traffic engineering over mpls," United States, Tech. Rep., 1999.
- [2] S. Agarwal, M. Kodialam, and T. V. Lakshman, "Traffic engineering in software defined networks," in *INFOCOM, 2013 Proceedings IEEE*, April 2013, pp. 2211–2219.
- [3] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, "OpenFlow: Enabling innovation in campus networks," *SIGCOMM Comput. Commun. Rev.*, vol. 38, no. 2, pp. 69–74, Mar. 2008.
- [4] M. Casado, M. J. Freedman, J. Pettit, J. Luo, N. McKeown, and S. Shenker, "Ethane: Taking control of the enterprise," *SIGCOMM Comput. Commun. Rev.*, vol. 37, pp. 1–12, 2007.
- [5] C. Filsfil, S. Previdi, B. Decraene, S. Litkowski, and R. Shakir, "Segment Routing Architecture," Internet Engineering Task Force, Tech. Rep., 15 Dec. 2015.
- [6] R. Hartert, H. Renaud, V. Stefano, S. Pierre, B. Olivier, F. Clarence, T. Thomas, and F. Pierre, "A declarative and expressive approach to control forwarding paths in Carrier-Grade networks," *ACM SIGCOMM Computer Communication Review*, vol. 45, no. 5, pp. 15–28, 2015.
- [7] S. Jain, A. Kumar, S. Mandal, J. Ong, and others, "B4: Experience with a globally-deployed software defined WAN," *ACM SIGCOMM*, 2013.
- [8] C. Y. Hong, S. Kandula, R. Mahajan, M. Zhang, and others, "Achieving high utilization with software-driven WAN," *ACM SIGCOMM*, 2013.
- [9] B. Fortz and M. Thorup, "Internet traffic engineering by optimizing OSPF weights," in *Proceedings IEEE INFOCOM 2000. Conference on Computer Communications. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies (Cat. No.00CH37064)*, vol. 2, 2000, pp. 519–528 vol.2.
- [10] H. H. Liu, S. Kandula, R. Mahajan, M. Zhang, and D. Gelernter, "Traffic engineering with forward fault correction," in *Proceedings of the 2014 ACM conference on SIGCOMM*, vol. 44. ACM, Aug. 2014, pp. 527–538.
- [11] G. Trimponias, Y. Xiao, H. Xu, X. Wu, and Y. Geng, "On traffic engineering with segment routing in SDN based WANs," Tech. Rep., Mar 2017. [Online]. Available: <https://arxiv.org/abs/1703.05907>
- [12] R. Bhatia, F. Hao, M. Kodialam, and T. V. Lakshman, "Optimized network traffic engineering using segment routing," in *2015 IEEE Conference on Computer Communications (INFOCOM)*, Apr. 2015, pp. 657–665.
- [13] A. Cianfrani, M. Listanti, and M. Polverini, "Incremental deployment of segment routing into an ISP network: a traffic engineering perspective," *IEEE/ACM Trans. Netw.*, vol. 25, no. 5, pp. 3146–3160, Oct. 2017.
- [14] T. Schüller, N. Aschenbruck, M. Chimani, M. Horneffer, and S. Schnitter, "Predictive traffic engineering with 2-segment routing considering requirements of a carrier IP network," in *2017 IEEE 42nd Conference on Local Computer Networks (LCN)*, Oct. 2017, pp. 667–675.
- [15] —, "Traffic engineering using segment routing and considering requirements of a carrier IP network," in *IFIP Networking Conference (IFIP Networking) and Workshops, 2017*, 2017, pp. 1–9.
- [16] R. Hartert, P. Schaus, S. Vissicchio, and O. Bonaventure, "Solving segment routing problems with hybrid constraint programming techniques," in *Principles and Practice of Constraint Programming*. Springer International Publishing, 2015, pp. 592–608.
- [17] N. Spring, R. Mahajan, and D. Wetherall, "Measuring isp topologies with rocketfuel," in *Proceedings of the 2002 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, ser. SIGCOMM '02. New York, NY, USA: ACM, 2002, pp. 133–145.