Traffic Engineering Approach to Virtual-link Provisioning in Software-defined ISP Networks

Slavica Tomovic, *Member, IEEE,* and Igor Radusinovic, *Member, IEEE*

***Abstract* — In this paper, we propose a new approach to virtual-link provisioning in ISP (Internet Service Provider) networks. The approach relies on Software Defined Networking (SDN) architecture based on OpenFlow that allows highly granular traffic splitting across multiple paths. The traffic control logic of SDN controller is divided into offline and online component. The online component handles dynamic arrivals of virtual-link requests in accordance with Service Level Agreements (SLAs) requirements. The offline component is responsible for periodic optimization of traffic distribution in the network. In this way, we tend to increase acceptance ratio for virtual-link requests and minimize degradation of best-effort traffic in a scalable manner. Our simulation results show that the proposed method for virtual-link provisioning outperforms solutions that rely on Constrained Shortest Path First (CSPF) and Equal Cost Multi-Path (ECMP) routing.**

***Keywords* — OpenFlow, SDN, traffic engineering, QoS.**

I. Introduction

With the evolution of Internet of Things (IoT) paradigm and increasing popularity of multimedia services, Quality of Service (QoS) provisioning becomes one of the most challenging aspects for next-generation networks. Virtual-link services, also known as Virtual Leased Line (VLL), are a fundamental part of the Internet Service Provider (ISP) offering, used to provide QoS guarantees for some applications or to support Virtual Private Network (VPN) functionalities (e.g. interconnect different sites of a company or IoT devices with Cloud services). In order to provide the necessary QoS, ISP providers usually adopt a strategy of link overprovisioning [1-2]. However, due to high costs associated with Wide Area Network (WAN) links, such a strategy cannot be economically justified in the long run. Thus, there is an ever-rising expectation from Traffic Engineering (TE) to improve network utilization.

Although TE techniques have been widely exploited in the past, it has been shown that complex TE and QoS can hardly be achieved with distributed routing protocols [3]. Namely, in most of the existing production Multi Protocol Label Switching (MPLS) networks, ingress routers greedily select routes for traffic flows. As a result, the network usually stucks in globally suboptimal routing patterns [1]. To overcome limitations of classic MPLS TE, the Internet Engineering Task Force (IETF) proposed Path Computation Element (PCE) architecture, where a dedicated element is in charge of centralized path computation [4]. However, PCE based architecture also suffers from slow path establishment issues, since it relies on distributed Resource Reservation Protocol (RSVP-TE) to set up, maintain and tear down traffic tunnels. Moreover, from the TE perspective, there is an issue with per-flow traffic splitting granularity in multipath routing, which is determined by the forwarding element capabilities and might deviate significantly from that assumed by TE solution [5]. This motivated emergence of Software Defined Networking (SDN) architecture with out-of-band programming capabilities and higher granularity of traffic control [5].

SDN TE mechanisms for Data Center (DC) networks, inter-DC networks and small-scale WANs have already shown a good performance [1-3], [6-8]. However, these solutions cannot be directly applied to ISP networks, which are characterized with unique requirements and properties. For example, DC and private WAN networks have a privilege to throttle bulky data transfers when necessary [1, 3]. In this way, sudden leaps of traffic matrix could be efficiently controlled. On the other side, ISP networks must fulfill Service Level Agreements (SLAs) to their customers and thus cannot "shape" traffic matrix. Recent works [9,10] take into account this limitation and propose TE methods for Software Defined ISP (SD-ISP) networks. In [9], flow migration approach has been proposed for dynamically manage network resources. In [10], a trade-off between scalability of SDN control plane and routing optimality is investigated. Authors of [10] argue that performance of ISP networks that use load balancing over multiple paths is not significantly affected by the timeliness of traffic matrix information. However, unlike our work, they have not considered any traffic differentiation.

This paper presents a new design of SDN controller with QoS on-demand and TE capabilities. The control logic is divided into two main components: online and offline. The online component quickly handles arrivals of virtual-link requests with strict QoS requirements. The offline component aims to optimize load balancing in the network periodically, such that Maximum Link Load (MLL) is minimized and SLA policies are met. In our analysis MLL stands for total offered load that is routed over the link. When MLL is larger than 100% of usable link capacity best-effort flows are degraded due to insufficient bandwidth.

Paper received April 21, 2018; revised June 18, 2018; accepted July 15, 2018. Date of publication July 31, 2018. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Grozdan Petrović.

This paper is a revised and expanded version of the paper presented at the 25th Telecommunications Forum TELFOR 2017 [12].

S. Tomovic is with the Faculty of Electrical Engineering, University of Montenegro, Dzordza Vasingtona bb, 81000 Podgorica, Montenegro (e-mail: slavicat@ac.me).

I. Radusinovic is with the Faculty of Electrical Engineering, University of Montenegro, Dzordza Vasingtona bb, 81000 Podgorica, Montenegro (e-mail: igorr@ac.me).

Under usable link capacity we assume 70% of link capacity because ISP has to limit usage of links due to delay guarantees. Thus, in the rest of the paper, by "link capacity" we assume this "usable" link capacity. By organizing the control logic into online and offline component, we trade some routing optimality for the sake of the control plane scalability. Simulation study on two ISP topologies has shown that the proposed approach outperforms TE solution based on Constrained Shortest Path First (CSPF) and Equal Cost MultiPath (ECMP) routing, that is commonly used in today's production networks [1]. The performance improvement is evident even when load balancing is optimized in relatively long intervals between successive offline phases.

The rest of the paper is organized as follows. Section II presents the system model as well as assumptions made in our analysis. Section III explains the simulation framework, while Section IV presents the obtained results. Finally, we conclude the paper in Section V.

II. The Network Model

We consider the SD-ISP network architecture from Fig. 1. The data plane consists of Core OpenFlow switches and Provider Edge (PE) OpenFlow switches.

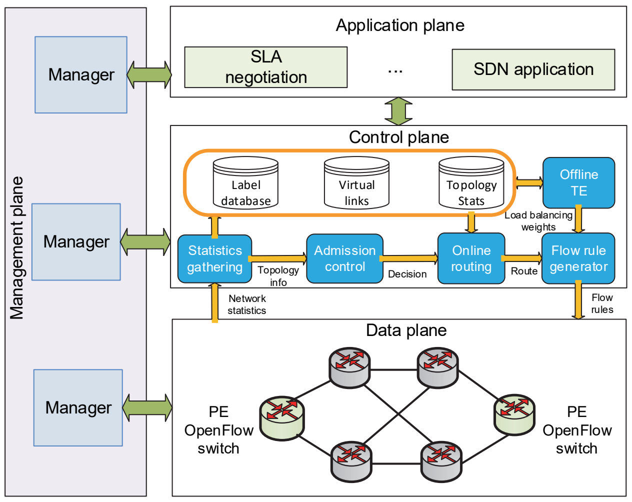

Fig. 1. The SD-ISP network model.

The application plane offers three types of SLAs for virtual-links:

1. *Best-effort* – offers connectivity service without guaranteeing performance for the virtual-link. Thus, this type of virtual-link requests is never rejected, and its bandwidth demand is not always met.
2. *QoS1* – guarantees minimum bandwidth and bounds maximum delay.
3. *QoS2* – guarantees lower delay bound than QoS1 and the required bandwidth for time-critical data transfers.

Decisions on SLAs are made in coordination with the control plane, which comprises the following functional modules:

- *Statistics gathering* – collects network statistics in regular intervals.
- *Admission control* – checks availability of network resources in order to determine whether QoS requirements of a virtual-link request can be met. If not, the request is rejected.
- *Online routing* – computes the most suitable route for QoS1 and QoS2 requests.
- *Offline TE* – runs TE algorithm periodically.
- *Flow rule generator* – installs new flow-table rules.

The management plane performs various operations to support the network infrastructure and services. For example, it configures contracts and SLAs in the applications plane, defines policies for switch-controller association in the control plane and manages configurations of OpenFlow switches in the data plane.

Up to K tunnels are installed a priori between each Ingress-Egress (IE) pair of PE OpenFlow switches. New tunnels are installed only when topology is updated (e.g. due to link or switch failure). Ingress OpenFlow switches traffic class in one of total three classes that correspond to the three SLA types. Each traffic class is marked with a specific label. The controller does not react to individual arrivals of best-effort traffic flows. Incoming best-effort traffic with the common destination node is splitted across (up to) K tunnels, according to pre-configured load balancing weights that are periodically optimized by the controller's offline TE component. This is acceptable because QoS is not guaranteed for best-effort class. On the other side, QoS1 and QoS2 virtual-links must follow bandwidth-delay constrained shortest paths, thus the controller's intervention is necessary for admission control. The initial setup of QoS virtual-links is done by the computationally simple online routing component. However, the offline TE helps in achieving an optimal routing pattern by periodically re-optimizing load balancing over multiple tunnels. Traffic splitting is performed on per-flow basis in order to avoid packet reordering issues. This is done by applying a hash function over the set of packets' header fields. Since OpenFlow switches can use a wide range of classifiers to forward packets [11], fine-granularity of traffic splitting is expected [5]. Unequal traffic splitting ratios could be implemented with Group tables in the processing pipeline of OpenFlow switches [11]. The basic Flow table in the pipeline maps a packet based on its destination and the label attached to one of the three Group tables. Each Group table corresponds to a different traffic class, and it is configured with a set of up to K tunnels and K weights, which determine the ratio of the traffic to be sent to each tunnel [12].

A. *Online routing*

The online TE component reacts to QoS1 and QoS2 virtual-link requests initiated by users. Due to scalability issues of SDN controller(s), we strive to reduce the load and complexity of the online component. Admission control for QoS1 and QoS2 virtual-links is performed based on the result of the simple CSPF algorithm. The algorithm seeks for the tunnel that can fulfil guarantees specified in the corresponding SLA. If no such tunnel exists in the network, virtual-link request is rejected. At this stage, the goal is not to optimize the routing decision, but rather to find a quick solution to route demands. This is so because the combination of TE and bandwidth-delay constrained routing would result in a problem unsolvable in real-time. Namely, TE techniques usually optimize network utilization by choosing the appropriate link weights. Since link weight and delay are two additive constraints, the routing problem becomes the NP-complete constrained path optimization [9]. Thus, we decided for a simple CSPF algorithm, which runs in two steps. In the first step, the algorithm prunes all the tunnels with insufficient bandwidth. During this step the existence of best-effort virtual links is ignored. In the second step, a tunnel with the least delay is found. If a tunnel delay is higher than required, virtual-link request is rejected. This procedure tends to always satisfy bandwidth and delay requirements of QoS virtual-links. However, best-effort traffic is treated as low-priority traffic, and its performance is degraded when total offered load cannot be met with the bandwidth reserved for virtual-link services by the ISP.

B. *Offline TE*

The offline TE module periodically collects aggregate traffic statistics. Then, it runs optimization algorithm, which tends to minimize MLL subject to the QoS constraints. The inputs of the algorithm are: the network graph, traffic matrix and list of tunnels for each IE pair.

The algorithm models SD-ISP network as a directed graph G=(V, L), where V is the set of nodes and L is the set of edges. Each link $l \in L$ is described by two parameters: usable link capacity c_l and propagation delay d_l. We assume that propagation delay predominantly determines the total path delay. Note that this would require ISP to limit usable bandwidth on links 20-30% below their capacity. On the other hand, because of our focus on the highly-multiplexed ISP core where resources for the user are reserved by considering only maximum aggregate flow size, the impact of queuing delay is expected to diminish [13]. Traffic matrix TM is $|N|x|N|$ matrix, where N is the set of edge OpenFlow switches in the network. Matrix element t_{ij} refers to aggregated traffic demand between ingress switch i and egress switch j. For the purpose of network optimization, matrix TM is decomposed into three matrices TM_{be}, TM_{qos1} and TM_{qos2}, one for each traffic class. A traffic-class demand $t_{ij}^{(C)}$, $C \in \{be, qos1, qos2\}$ is splitted across the set of tunnels P_{ij}, which is computed a priori. For each traffic demand $t_{ij}^{(C)}$, the offline TE algorithm computes optimal splitting ratios $x_{t_{ij}^{(C)},p}$ for tunnels $p \in P_{ij}$, i.e. portions of the traffic demand that would be routed over each tunnel from the set P_{ij}. The optimization algorithm is formulated as follows:

$$\min z \tag{1}$$

Subject to:

$$\forall C, \forall (i,j) \in N: \quad \sum_{p \in P_{ij}} x_{t_{ij}^{(C)},p} == 1 \tag{2}$$

$$\forall l \in L: \quad \sum_{\forall C} \sum_{(i,j) \in N} \sum_{p \in P_{ij}} R_{p,l} \cdot x_{t_{ij}^{(C)},p} \cdot t_{ij}^{(C)} \le z \cdot c_l \tag{3}$$

$$\forall l \in L: \quad \sum_{C \in \{qos1, qos2\}} \sum_{(i,j) \in N} \sum_{p \in P_{ij}} R_{p,l} \cdot x_{t_{ij}^{(C)},p} \cdot t_{ij}^{(C)} \le c_l \tag{4}$$

$$\forall (i,j) \in N, \forall C, \forall p \in P_{ij}: \quad y_{t_{ij}^{(C)},p} \cdot \left(\sum_{l \in p} d_l - D_{\max}^{(C)}\right) \le 0 \tag{5}$$

$$\forall (i,j) \in N, \forall C, \forall p \in P_{i,j}: \quad y_{t_{ij}^{(C)},p} \ge x_{t_{ij}^{(C)},p} \tag{6}$$

$$\forall (i,j) \in N, \forall C, \forall p \in P_{ij}: \quad x_{t_{ij}^{(C)},p} \ge 0 \tag{7}$$

$$\forall (i,j) \in N, \forall C, \forall p \in P_{ij}: \quad y_{t_{ij}^{(C)},p} \in \{0,1\} \tag{8}$$

$$z \ge 0 \tag{9}$$

The optimization problem from above minimizes MLL z subject to the set of constraints (2) - (9). The decision variable $x_{t_{ij}^{(C)},p}$ defines a fraction of traffic demand $t_{ij}^{(C)}$ that should be routed over a path p. The constraint (2) ensures that all traffic demands are routed. The usable link capacity constraints are given with (3) and (4). $R_{p,l}$ is a binary value which indicates whether a path p includes a link l. If yes, $R_{p,l}$ is 1, otherwise it is 0. Since QoS classes require bandwidth guarantees, a total amount of reserved bandwidth must not exceed a usable link capacity. On the other hand, we assume that best-effort traffic flows enter the network unshaped. Thus, z could be greater than 1. In (5), we have introduced binary decision variables $y_{t_{ij}^{(C)},p}$, which serve for providing delay guarantees for QoS demands. This is necessary because total path delay for traffic classes $C \in$ {*qos1, qos2*} should not exceed $D_{\max}^{(C)}$, which stands for a maximum tolerable delay for a class. If path $p \in P_{ij}$ cannot meet this requirement for traffic class, (5) ensures that no traffic of that class is routed over p. That is, whenever statement $\sum_{l \in p} d_l - D_{\max}^{(C)}$ does not hold, $y_{t_{ij}^{(C)},p}$ (and thus also $x_{t_{ij}^{(C)},p}$) are forced to be 0.

III. Evaluation Methodology

To evaluate the performance of proposed virtual-link provisioning approach, we have developed an event-driven simulator in Python. Simulator solves optimization problems by using CPLEX [14] software. We considered two real ISP topologies in simulations: Sprint [15] and Abilene [16]. In both cases, only Points of Presence (PoP) nodes are considered. The usable link capacity is set to 1Gb/s. Delay on each link was derived from a great circle distance between the connected PoPs and speed of light in the fibre [2]. Since for most ISP networks real traffic traces are not available, we used a well-known gravity model [17] to generate Traffic Matrix (TM). From the gravity model matrix, we derived admissible TM, which certainly can be accommodated in the network if all incoming flows are best-effort. Finally, based on admissible TM, the dynamic evolution of the virtual-link requests was simulated. We used exponential distribution with the mean t=120s to model virtual-link durations, and the Poisson distribution to model arrivals of virtual-link requests for each IE pairs. Although the Poisson model is often considered overly simplistic for Internet traffic modelling, it can be useful to represent the limiting behaviour of an aggregate traffic flow created by multiplexing large numbers of sessions [13]. The mean rate of the request arrivals is computed such that the resulting traffic model on average converges to the admissible TM. The bandwidth requirement of each virtual-link was chosen randomly from the set [10Mbps, 15Mbps, 20Mbps]. QoS2 delay bound was set to 62ms for Sprint and 44ms for Abilene, which corresponds to delay on the shortest path between the mutually most distant ingress and egress node on these topologies. A maximum tolerable delay for QoS1 traffic class was set to 100ms. We defined a default simulation scenario such that 30% of overall virtual-link requests accounts for best-effort class, 30% to QoS1 class and 40% to QoS2 class. The number of paths used for multi-path routing between an IE pair was limited to K=15. The proposed TE approach has been compared with TE solution that implies uniform load balancing of best-effort virtual links over a set of tunnels installed a priori, and CSPF routing of QoS1 and QoS2 virtual-links. This approach is commonly used in today's production MPLS networks [1], and we denote it with CSPF-ECMP TE in the rest of the paper.

IV. SIMULATION RESULTS

The first set of results corresponds to the Sprint network scenario. Fig. 2 shows the maximum MLL, the time averaged MLL and the average link utilization (ALU) achieved with CSPF-ECMP approach and SDN-TE approach for different recurrence period of offline TE phases (T). It should be noted that in our analysis link load is expressed as percentages, relative to the usable link capacity. Thus a value of MLL higher than 100% corresponds to the scenario when the total bandwidth demand of the incoming best-effort flows (whose route goes over this link) cannot be met with the capacity reserved for virtual-links. This is done in order to better illustrate how the analysed TE schemes impact performance of the best-effort traffic class, for which bandwidth is not guaranteed. The results from Fig. 2 clearly show that SDN-TE approach is able to balance the network load more efficiently. Also, we evaluated reactive SDN-TE approach, which recalculates load balancing weights only when the offered network load gets too large, causing degradation of best-effort traffic. During these "critical" intervals, load balancing optimisations are done in response to each virtual-link arrival and departure. Obviously, the reactive SDN-TE manages to achieve the lowest MLL. However, when the network is heavily loaded the number of reconfigurations could reach an unacceptable level and lead to instability [18]. On the other side, if network is overprovisioned, as is the case for today's ISP networks, some kind of reactive TE solution might be appropriate. We leave a more detailed analysis of reactive optimization approaches for our future work. For a regular (non-reactive) SDN-TE approach, as expected, the best results were obtained for T=2 minutes. Increase in the interval between consecutive offline phases resulted in an increase of MLL. This performance drop could be attributed to the fact that load-balancing weights that are considered optimal at the beginning of the interval, deviate significantly from the real optimum after a while.

Fig. 3 shows results in terms of total amount of rejected QoS demands, expressed in Mbps. It can be observed that no QoS requests were rejected with the reactive SDN-TE model. When it comes regular SDN-TE model, in general, the number of rejected QoS requests increases with the period T between consecutive offline TE phases.

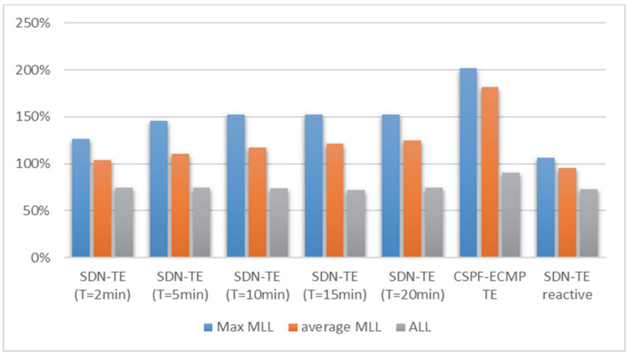

Fig. 2. MLL, averaged MLL and ALL results.

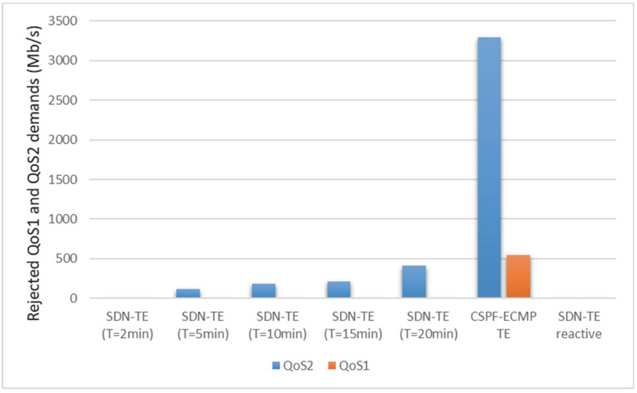

Fig. 3. Rejected QoS requests (Sprint topology).

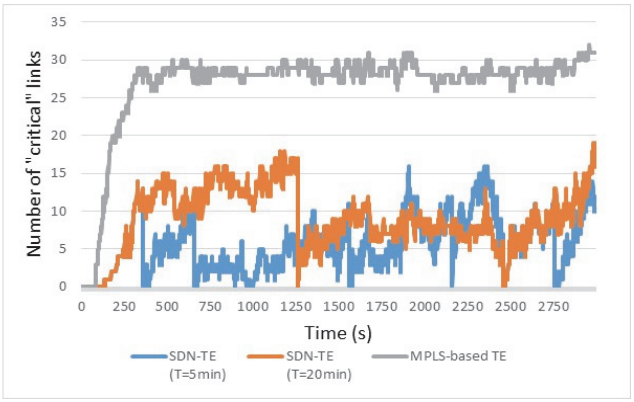

Fig. 4. The number of "critical" links (Sprint topology).

Fig. 4 shows the number of "critical" links on which best-effort traffic is degraded, for SDN-TE case with T=5 and T=20 minutes, and CSPF-ECMP TE. In all simulations of SDN-TE approach we performed an early offline optimization in the 60th second. After that, offline phases were repeated with the period T. At the beginning of SDN-TE simulations, when all elements of traffic matrix are zero, uniform load balancing over a set of available tunnels is assumed. Thus, by performing early optimization, we wanted to reduce the impact of the traffic-oblivious load balancing in period before the first regular offline phase. From Fig. 4 it could be observed that performance improvement over CSPF-ECMP is significant even for T=20min. This suggests that SDN-TE could be scalable for future ISP networks.

We also investigated the impact of path diversity on TE performance. MLL results as a function of the maximum number of tunnels (K) used between IE pairs are given in Table 1. The results in terms of the total amount of rejected

QoS traffic are shown in Table 2. In general, splitting traffic across more tunnels leads to lower MLL when SDN-TE is used. However, *K* is usually constrained by hardware capabilities of OpenFlow switches. An experimental study on SWAN inter-DC network [1] has shown that using more than 15 tunnels between IE switch pairs results in up to 20K rules at switches. On the other hand, flow tables of SDN switches usually have much smaller capacity, since TCAM (Ternary Content Addressable Memory) is a default choice of a memory subsystem to implement flow tables [19]. The largest memory space on a TCAM chip is far less than that of Binary Content Addressable Memory (CAM). For example, HP ProCurve 5406zl TCAM switch hardware can support 1500 OpenFlow rules [20]. The main reason behind the design of TCAM with such limited space is the inherent trade-off between the table size and other factors such as power and cost. When compared to SRAM, TCAM consumes almost 100 times more power and has almost 100 times more cost [21]. In SDN networks, flows could be identified by a 15 field tuple (unlike in IP where it is defined by 2-5 field tuples depending on the layer they belong to) [11]. For this reason, each flow entry in TCAM mandates 7 times bits than that of conventional IP switch.

TABLE 1: MLL FOR DIFFERENT VALUES OF K PARAMETER.

Algorithm	MLL (K=5)	MLL (K=8)	MLL (K=10)	MLL (K=15)	MLL (K=20)
SDN-TE (T=5min)	165 %	161 %	147 %	146 %	141 %
SDN-TE (T=20min)	162 %	164 %	161 %	146 %	158 %
CSPF-ECMP TE	186 %	193 %	196 %	202 %	201 %

TABLE 2: AMOUNT OF REJECTED QOS (ARQ) TRAFFIC (IN Mbit/s).

Algorithm	ARQ (K=5)	ARQ (K=8)	ARQ (K=10)	ARQ (K=15)	ARQ (K=20)
SDN-TE (T=5min)	6365	2520	50	120	90
SDN-TE (T=20min)	9115	4270	310	415	355
CSPF-ECMP TE	23605	16990	5915	3835	3605

The complexity of offline optimization increases with *K*. The running time of offline phase as a function of *K* is shown in Fig. 5. Simulations were run on PC with Windows 10 OS, Intel Core i7 processor and 16GB of RAM. Therefore, we can conclude that the results from Tables 1 and 2 are optimistic in the sense that SDN-TE improvement flattens at around 10 paths. Moreover, shifting from 10 to 15 tunnels per IE pair resulted in a slight increase in the amount of rejected QoS demands. This is so because for *K*=10 scenario QoS traffic is routed over shorter routes on average, and thus consumes a smaller amount of the network capacity.

Figs. 6 and 7 compare the results in terms of MLL for 4 simulations scenarios with different ratios of best-effort, QoS1 and QoS2 traffic classes, as described in Table 3. In Scenario 1, where 80% of all requests were best effort, the acceptance ratio was 100%. In Scenario 4, where QoS2 requests are dominant, difference in performance between the analysed solutions increases. SDN-TE (T=20 minutes), which we refer to as a scalable SDN-based approach, rejected 6 times less user demands than CSPF-ECMP TE.

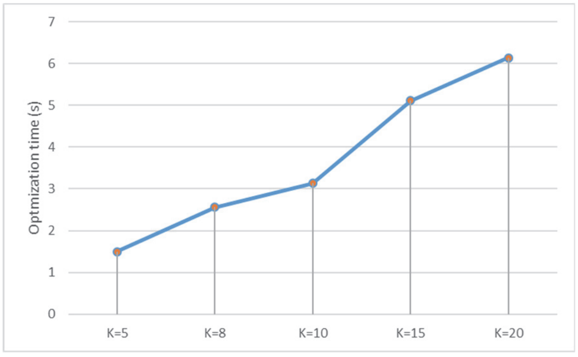

Fig. 5. Execution time of the offline optimization algorithm as a function of K.

TABLE 3: TRAFFIC PROPERTIES IN DIFFERENT SIMULATION SCENARIOS.

Class	Scenario 1	Scenario 2	Scenario 3	Scenario 4
Best-effort	80%	50%	30%	20%
QoS1	10%	30%	30%	30%
QoS2	10%	20%	40%	50%

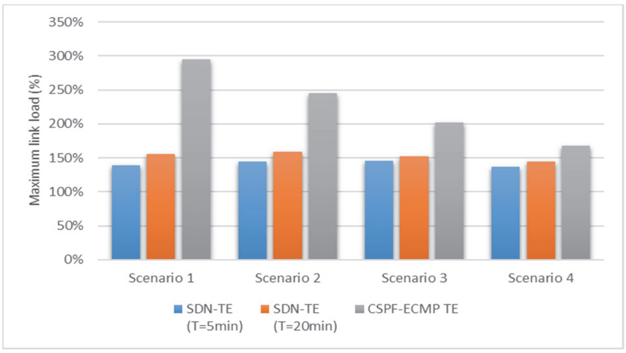

Fig. 6. MLL in 4 scenarios with different ratios of traffic classes (Sprint topology).

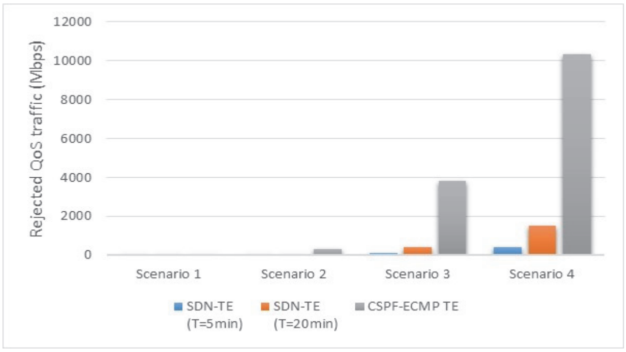

Fig. 7. Rejected QoS traffic demand in 4 scenarios with different ratios of traffic classes (Sprint topology).

Some of the obtained results for Abilene network topology are given in Figs. 8 and 9. We can see that relative ranking of the analysed TE solutions remained unaffected. MLL in this scenario is slightly lower, but a larger amount of QoS requests is rejected. Similarly to Sprint topology, best-effort traffic experienced the worst service during the initial period of simulation. This is so because a traffic matrix used as an input argument of the early offline phase is still significantly below the average value of traffic matrix projected during the simulation setup. As a consequence, optimality of load balancing parameters computed in the early offline phase rapidly decreases over time. The parameter K influenced TE performance in a similar manner, as was the case on Sprint topology. Thus, we can conclude that the proposed TE approach on both analysed topologies performs almost equally well.

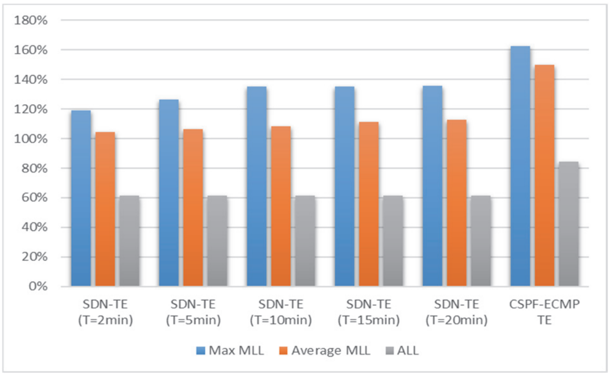

Fig. 8. MLL, averaged MLL and ALL on Abilene network topology.

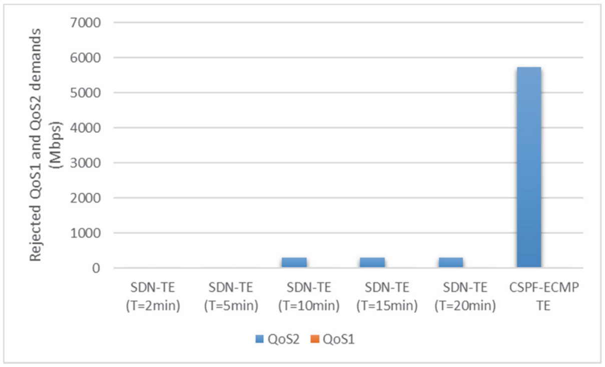

Fig. 9. Rejected QoS traffic (Abilene network topology).

V. Conclusion

This paper extends the work from [12], where we proposed a new TE approach to virtual-link provisioning in SD-ISP networks. The controller's logic for traffic control consists of the online routing component and the offline TE component. The online routing component is designed to quickly find a solution to route virtual-link requests with strict QoS requirements. Best-effort traffic is routed over multiple tunnels installed a priori, according to load balancing weights that are periodically reconfigured by the controller's offline TE component. Offline TE is defined as a linear programming problem that tends to minimize MLL subject to the QoS constraints. The extensive simulations on Sprint and Abilene ISP network topologies demonstrate the attractive properties of the proposed approach. The obtained results are promising in the sense that an improvement over the static load balancing is evident even in scenarios with infrequent offline TE phases. Therefore, the proposed controller design alleviates concerns for scalability of the centralized TE. The proposed control logic is implemented OpenFlow controller, but the results are not shown due to lack of the space in this paper.

In our future work, we include reliability requirements in our TE model, and investigate a trade-off between routing optimality and reconfiguration rate more closely.

References

[1] C.-Y. Hong et al., “Achieving high utilization with software-driven WAN,” presented at *ACM SIGCOMM 2013*, pp. 15–26, 2013.
[2] X. Jin, Y. Li, D. Wei, S. Li, J. Gao, L. Xu, G. Li, W. Xu, and J. Rexford, “Optimizing bulk transfers with software-defined optical WAN,” in *Proceedings of the ACM SIGCOMM '16*, New York, pp. 87-100, 2016.
[3] S. Jain et al., “B4: Experience with a globally-deployed software defined WAN”, in *Proc.* of *ACM SIGCOMM,* vol. 43, no. 4, pp. 3–14, 2013.
[4] A. Farrel, J.-P. Vasseur, and J. Ash, “A Path Computation Element (PCE)-Based Architecture,” Internet Requests for Comments, RFC Editor, RFC 4655, August 2006.
[5] A. Mendiola et al, "A Survey on the Contributions of Software-Defined Networking to Traffic Engineering," in *IEEE Comm. Surveys & Tutorials*, vol. 19, no. 2, pp. 918-953, 2017.
[6] M. Al-Fares, S. Radhakrishnan, B. Raghavan, N. Huang, and A. Vahdat, “Hedera: dynamic flow scheduling for data center networks”, *NSDI'10*, San Jose, CA, USA, 2010, vol. 10, pp. 19–29.
[7] A. R. Curtis, J. C. Mogul, J. Tourrilhes, P. Yalagandula, P. Sharma, and S. Banerjee, “DevoFlow: scaling flow management for high-performance networks”, in *Proc. ACM SIGCOMM*, vol. 41, no. 4, pp. 254–265, 2011.
[8] J. M. Wang, Y. Wang, X. Dai and B. Bensaou, "SDN-based multi-class QoS-guaranteed inter-data center traffic management," in *Proc. IEEE Intern. Conference on Cloud Networking,* pp. 401-406, 2014.
[9] S. Tomovic and I. Radusinovic, "Fast and efficient bandwidth-delay constrained routing algorithm for SDN networks," in *Proc. IEEE NetSoft Conference and Workshops*, Seoul, pp. 303-311, 2016.
[10] J. L. Martins and N. Campos, "Short-sighted routing, or when less is more," *IEEE Communications Magazine*, vol. 54, no. 10, pp. 82-88, October 2016.
[11] Open Networking Foundation - OpenFlow v1.5.1 specification. Available:http://www.opennetworking.org/images//openflow-switch-v1.5.1.pdf , March 2015.
[12] S. Tomovic and I. Radusinovic, “Traffic engineering approach to virtual-link provisioning in software-defined isp networks,” in 2017 25thTelecommunication Forum (TELFOR), Nov 2017, pp. 1–4.
[13] T. Karagiannis, M. Molle, M. Faloutsos and A. Broido, "A nonstationary Poisson view of Internet traffic," IEEE INFOCOM 2004, Hong Kong, 2004, pp. 1558-1569 vol.3.
[14] CPLEX Optimization Studio. [Online]. Available: http://www-03.ibm.com/software/products/en/ibmilogcpleoptistud, 2017.
[15] A. Porxas, S. C. Lin and M. Luo, "QoS-aware virtualization-enabled routing in software-defined networks," in *Proc. IEEE International Conference on Communications*, London, pp. 5771-5776, 2015.
[16] O. Bonaventure, “Computer networking: principles, protocols and practice,” Rel. 0.25, Ch. 5, p. 194.
[17] A. Gunnar, M. Johansson, and T. Telkamp, “Traffic matrix estimation on a large IP backbone: a comparison on real data”, In Proc. of the 4th ACM SIGCOMM, pp.149-160, Italy, 2004.
[18] U. Ashraf, "Rule Minimization for Traffic Evolution in Software-Defined Networks," in *IEEE Communications Letters*, vol. 21, no. 4, pp. 793-796, April 2017.
[19] J. P. Sheu, W. T. Lin and G. Y. Chang, "Efficient TCAM Rules Distribution Algorithms in Software-Defined Networking," in IEEE Transactions on Network and Service Management, vol. 15, no. 2, pp. 854-865, June 2018.
[20] S. Q. Zhang et al., "Sector: TCAM Space Aware Routing on SDN," 2016 28th International Teletraffic Congress (ITC 28), Würzburg, Germany, 2016, pp. 216-224.
[21] S. Bhowmik et al. “Addressing TCAM Limitations of Software-Defined Networks for Content-Based Routing,” In Proc. of ACM DEBS '17, pp. 100-111, NY, USA.