# Virtual Network Embedding with Path-based Latency Guarantees in Elastic Optical Networks

Sepehr Taeb*, Nashid Shahriar*, Shihabur Rahman Chowdhury*, Massimo Tornatore†, Raouf Boutaba*, Jeebak Mitra‡, and Mahdi Hemmati‡
*David R. Cheriton School of Computer Science, University of Waterloo,
{staeb | nshahria | sr2chowdhury | rboutaba}@uwaterloo.ca
†Politecnico di Milano, massimo.tornatore@polimi.it
‡Huawei Technologies Canada Research Center, {jeebak.mitra | mahdi.hemmati}@huawei.com

***Abstract*—Elastic Optical Network (EON) virtualization has recently emerged as an enabling technology for 5G network slicing. A fundamental problem in EON slicing (known as Virtual Network Embedding (VNE)) is how to efficiently map a virtual network (VN) on a substrate EON characterized by elastic transponders and flexible grid. Since a number of 5G services will have strict latency requirements, the VNE problem in EONs must be solved while guaranteeing latency targets. In existing literature, latency has always been modeled as a constraint applied on the virtual links of the VN. In contrast, we argue in favor of an alternate modeling that constrains the *latency of virtual paths*. Constraining latency over virtual paths (vs. over virtual links) poses additional modeling and algorithmic challenges to the VNE problem, but allows us to capture end-to-end service requirements. In this paper, we first model latency in an EON by identifying the different factors that contribute to it. We formulate the VNE problem with latency guarantees as an Integer Linear Program (ILP) and propose a heuristic solution that can scale to large problem instances. We evaluated our proposed solutions using real network topologies and realistic transmission configurations under different scenarios and observed that, for a given VN request, latency constraints can be guaranteed by accepting a modest increase in network resource utilization. Latency constraints instead showed a higher impact on VN blocking ratio in dynamic scenarios.**

## I. Introduction

Ultra-low latency communication is an important quality-of-service (QoS) requirement for many emerging applications such as intelligent transportation, industry automation, immersive media experience through virtual and augmented reality, online multi-player gaming, and tactile Internet [1]–[4]. Additionally, safety, real-time control, and healthcare related applications can have life-threatening consequences if their stringent latency requirements are not met [4]. Latency also significantly impacts the revenue generated by services such as electronic commerce and high-frequency trading [5]–[7]. Consequently, latency-sensitive applications have become one of the major business drivers for the development of the fifth-generation (5G) mobile networks [3], [8]–[10]. A key enabling technology for deploying latency-sensitive applications in 5G networks is network virtualization (*aka* network slicing) [11], which allows the instantiation of one or more virtual networks (VNs) with dedicated substrate resources to guarantee latency between application end nodes.

A key challenge in instantiating VNs for latency-sensitive applications is to devise a mechanism to efficiently map VN nodes and links on the substrate network (SN). This problem, known as *Virtual Network Embedding (VNE)* [12], has been extensively studied in the past, with a particular focus on ensuring bandwidth and reliability requirements [13]. Less attention has been devoted to the VNE problem with latency constraints. In existing literature, latency requirements have been modeled as constraints applied to the virtual links of the VN (*i.e.*, each virtual link is mapped to satisfy a given latency target) [14]–[18]. However, many 5G verticals require that their end-users perceive ultra-low latency. This requires a given latency constraint to be enforced along an entire path between application end nodes [19]. For instance, two virtual nodes in a VN can represent two trading sites running an online trading service. In this case, the network operator must compute one or more virtual paths between these two sites while guaranteeing stringent latency requirements of the service [1].

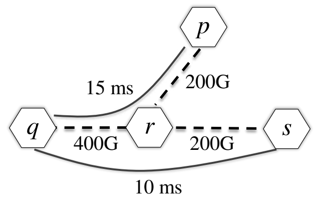


Fig. 1. VN with latency requirement on virtual paths (marked by solid lines)

From a resource allocation perspective, per-virtual-path latency constraints have the advantage of granting more flexibility in the selection of substrate paths for embedding compared to per-virtual-link latency constraints. For instance, in Fig. 1, the latency bound on virtual link *qr* can be varied as long as the latency constraints on virtual paths *p-r-q* and *q-r-s* are not violated. Hence, more options to embed virtual link *qr* become available if we assume path-based constraints, compared to having a fixed latency budget on *qr*. This flexibility comes with the challenge of cleverly distributing the latency budgets of virtual paths to constituent virtual links in a way that results in the optimal selection of substrate paths for embedding. To the best of our knowledge, no previous work has addressed the VNE problem with latency constraints on virtual paths.

The networking infrastructure supporting 5G slicing can span different technology domains such as wireless radio, access/core/metro transport networks, and multi-tier data centers [11]. In this paper, we focus on transport networks and assume that a portion of the total latency budget is assigned to the transport segment to satisfy the end-user latency requirements. Traditional "fixed grid" optical networks have been for a long time a dominant technology in the transport segment thanks to their high-bandwidth and low-latency. Recently, network operators are adopting Elastic Optical Networks (EONs) to overcome the limitations of fixed grid networks, such as inflexible and coarse-grained resource allocation [20], [21]. EONs have the capability to allocate an arbitrary number of spectrum slices for right-size spectrum allocation to customer needs (i.e., EONs employ a flexible grid, as opposed to the traditional fixed grid, to allocate optical spectrum slices). They also leverage the advances in coherent-transmission technology allowing the tuning of transmission parameters such as baud rate, modulation format and forward error correction (FEC) overhead. In what follows, we will consider EON as the underlying substrate network (SN) for our problem. We assume the existence of a logically centralized network controller with a global view of the EON resources [22]–[25], which will execute our solutions to provision VN requests.

EON virtualization has recently garnered interest from the research community, especially due to its importance in 5G network slicing [21]. VNE over EON introduces unique challenges arising from the large number of configurable transmission parameters available in EONs [26]. Several proposals for optimal resource allocation in EONs have considered tuning all or a subset of these transmission parameters for satisfying bandwidth requirements [26]–[28]. However, no work in the EON virtualization literature has considered satisfying path-based latency requirements.

In this paper, we address the problem of VNE over EON with latency guarantees on a set of given virtual paths, while considering all the configurable transmission parameters. We first present a mathematical representation of the inputs and characterize the latency contributors in an EON (Section III). Then, we present an Integer Linear Program (ILP) formulation and a heuristic solution for solving the VNE problem with path-based latency requirements (Section IV and Section V). We perform extensive evaluations of our proposed solution using real network topologies from [29] under different evaluation scenarios (Section VI). Our simulation results show that the heuristic incurs only 2.5% and 0.8% additional cost on average compared to the ILP for fixed- and flex-grid EONs, respectively, while executing three to four orders of magnitude faster. Moreover, we observed that guaranteeing latency constraints of a VN causes a modest increase in resource utilization, while it results in a more significant impact on VN blocking ratio in dynamic scenarios.

## II. Related Works

**VNE over EON** Variants of the VNE problem have been extensively studied over the last decade considering layer 2/3 SN [13]. VNE has also been investigated in the context of fixed-grid optical networks [30] and, more recently, in the context of EONs [26]–[28], [31]. Most of the research on VNE over EON considers tuning a subset of the flexible transmission parameters for optimizing resource allocation. More recently, Shahriar *et al.*, proposed an optimization model for allocating resources to VNs while considering all the flexible transmission parameters made available by EONs, namely baud rate, modulation format, and FEC overhead. Their study shows that considering all degrees of freedom made available by EONs results in 60% less spectrum resource utilization compared to the case with no flexibility.

**Latency-guaranteed VNE on Layer-2/3 Networks** A number of research efforts have been dedicated to investigate the VNE problem on layer-2/3 SN while considering various forms of latency constraints such as per-VLink latency [14]–[18] and differential delay in a multi-cast VN [16], [32]. While most of these works consider a single delay source, Chochlidakis *et al.*, models latency in SN as a combination of propagation delay and queuing delay [33]. Queuing delay, modeled as a hyperbolic function of residual link capacity in the authors' earlier work [34], is then approximated using a piecewise linear function for simplifying the optimization model. Finally, an elaborate discussion on the contributing factors to layer-2/3 substrate network delay and literature survey on how they have been modeled in the past are presented in [35]. Another related problem that has received recent attention is low-latency service function chain provisioning and scheduling in 5G networks [19], [36]–[42]. However, service function chain provisioning and scheduling is a fundamentally different problem than the one addressed in this paper.

**Latency Considerations in Optical Networks** Several research papers and industry white papers have identified different sources of latency in an optical network [2], [43]–[46]. Accordingly, latency in optical networks stems from various technology domains such as physical layer operations [43], components inside an optical switching element [44], and transport layer processing [2], [45], [46]. These studies were helpful to us in determining which latency components should be considered while modeling the latency-constrained VNE problem for EON, and which could be omited to keep model complexity at a reasonable level.

To the best of our knowledge, VNE with latency requirements on virtual paths over a substrate EON with flexible transmission parameters has not been addressed. Furthermore, the literature on latency-guaranteed VNE over layer-2/3 SNs has considered per-VLink latency constraints. In contrast, we propose a novel latency modeling for the VN, *i.e.*, latency requirements on virtual paths, which is more suited to capture the requirements of emerging ultra-low latency applications.

## III. Mathematical Model and Problem Statement

### A. Substrate EON

The substrate EON (SN) is an undirected graph $G = (V, E)$, where $V$ and $E$ are the set of substrate optical nodes (SNodes) and substrate optical links (SLinks), respectively. Without loss of generality, we assume the optical nodes to be colorless, directionless, and contentionless [47]. We also assume the SLinks to be bi-directional, *i.e.*, adjacent optical nodes are connected by one optical fiber in each direction. The optical frequency spectrum on each SLink $e = (u, v) \in E$ is divided into equal-width slices represented by the set $S$ and enumerated as $1, 2 \ldots |S|$. In what follows, we use the term frequency/spectrum slot and slice interchangeably. $\mathcal{P}$ and $\mathcal{P}^k_{uv} \subset \mathcal{P}$ represents the set of all paths in $G$ and the set of $k$-shortest paths between nodes $u, v \in V$, respectively. The two endpoints of a path $p \in \mathcal{P}$ are represented by $src(p)$ and $dst(p)$. The number of SLinks and the physical length of a path $p$ in kilometers are represented by $|p|$ and $len(p)$, respectively. We use the binary variable $\delta_{pe}$ to denote the association between a path $p \in \mathcal{P}$ and any link $e \in E$.

The following transmission parameters can be configured on a path $p$ with physical length $len(p)$ to enable data transmission with different data-rates $d \in \mathcal{D}$: *baud-rate* or *symbol-rate*, $b$, *modulation format*, $m$, and *forward error correction code* (FEC) *overhead*, $f$, selected from the set of possible values $\mathcal{B}$, $\mathcal{M}$, and $\mathcal{F}$, respectively. We use a tuple $t = (d, b, m, f) \in \mathcal{T} = (\mathcal{D} \times \mathcal{B} \times \mathcal{M} \times \mathcal{F})$ to represent a transmission configuration that dictates the combination of $b \in \mathcal{B}$, $m \in \mathcal{M}$, and $f \in \mathcal{F}$ that can be used to yield a data-rate $d \in \mathcal{D}$. For the sake of representation we use $t^{(d)}, t^{(b)}, t^{(m)}$, and $t^{(f)}$ to denote the data-rate, baud-rate, modulation format, and FEC overhead of a configuration $t \in \mathcal{T}$. A *reach table* $\mathcal{R}$, computed based on physical layer characteristics, specifies the maximum length of a path (*i.e.*, the *reach* $r_t$) capable of retaining a satisfactory optical signal to noise ratio when configured according to a transmission configuration $t \in \mathcal{T}$. Finally, $n_t$ denotes the number of slices required to accommodate a transmission configuration $t \in \mathcal{T}$, which is dependent on the parameters of $t$.

## B. Virtual Network Request

The VN requests are made in the form of an undirected graph $\bar{G} = (\bar{V}, \bar{E})$, where $\bar{V}$ and $\bar{E}$ are the set of virtual nodes (VNodes) and virtual links (VLinks), respectively. The function $\tau : \bar{V} \rightarrow V$ represents VNode to SNode mapping and is an input to our problem (a common assumption for optical network virtualization [48]). Each virtual link $\bar{e} \in \bar{E}$ has a bandwidth requirement $\bar{\beta}_{\bar{e}}$. We allow the VLinks to be mapped to multiple substrate paths (SPaths) (similar to [26], [49]), each with a lower data-rate than $\bar{\beta}_{\bar{e}}$, since the reach becomes smaller for higher data-rates (*e.g.*, more than 400Gbps) limiting the number of usable paths. However, we limit the number of VLink splits to maximum $q$ $(\geq 1)$. Such multi-path provisioning is supported by technologies such as Virtual Concatenation (VCAT) in Optical Transport Network (OTN) [50] or bonding capabilities of FlexE [51]. Some VPaths in a VN request have latency constraints depending on the QoS of the services provided by the VN. We represent such constraints as follows:

*1) Latency Constraints:* We assume there exists a function $latency(.)$, which, when applied on any path, either virtual or substrate, returns the latency of that path. Let, $\bar{a}$ be a loop-free path in the VN (*i.e.*, a VPath). A VPath can have one (*i.e.*, representing a single VLink) or more VLink(s), therefore, $|\bar{a}| >= 1$. We represent the latency budget of a VPath using a tuple $\ell = (\bar{a}, L) \in (P_{\bar{G}} \times \mathbb{R}^+)$, where $P_{\bar{G}}$ is the set of all paths in $\bar{G}$ that require a bounded latency as mandated by the service provider and $\mathbb{R}$ represents the set of all positive real numbers. The tuple $\ell$ implies that, after embedding, the latency of the VPath $\ell^{(\bar{a})}$ should not exceed $\ell^{(L)}$, *i.e.*, $latency(\ell^{(\bar{a})}) \leq \ell^{(L)}$, where $latency(\ell^{(\bar{a})}) = \sum_{\bar{e} \in \bar{a}} latency(\bar{e})$ and $latency(\bar{e})$ is the latency of the VLink $\bar{e}$. All the latency constraints of a VN request are represented by the set of such tuples $\mathcal{L}(\bar{G})$.

## C. Latency Model

We present a latency model that captures different key latency contributors in an EON. The latency contributors can be broadly classified into those at EON nodes and those on a lightpath (Table I). This latency model will be used to compute the latency perceived along a lightpath, which in turn will help determine the usable lightpaths for the VN embedding.

TABLE I
LATENCY CONTRIBUTORS IN EON

| **Node latency** = $2(L_{txp} + L_{fec})$ | |
|---|---|
| OTN/FlexE elements ($L_{otn}$) | Negligible |
| Transponders ($L_{txp}$) | 30 ns |
| FEC processing ($L_{fec}$) | 10 $\mu$s (std.), 150 $\mu$s (super) |
| **Path latency** = $len(p)L_{prop} + n_{amp}L_{amp} + (\lvert p\rvert + 1)L_{roadm}$ | |
| Fiber propagation ($L_{prop}$) | 4.9 $\mu$s/km |
| Regenerators ($L_{rgn}$) | Not considered |
| Amplifiers ($L_{amp}$) | 150 ns/unit |
| CD compensation ($L_{dcf}$) | Not considered |
| ROADMs & BV-OXCs ($L_{roadm}$) | O(nanoseconds) |

*1) Latency contributors at an EON node:* We assume a multi-layer flexible node architecture as presented in [52], that consists of a set of OTN/FlexE line-cards (with or without a FEC module), a set of bandwidth-variable transponders (BVTs), and a reconfigurable optical add-drop multiplexer (ROADM). A transparent lightpath goes through all these components only at the source and destination EON nodes of the lightpath. At intermediate nodes, also known as the bypass nodes, the lightpath only passes through ROADMs bypassing OTN/FlexE elements and BVTs. Therefore, we will consider the latency contributed by ROADMs as a latency component at lightpath level to be discussed in Section III-C2.e.

*a) OTN/FlexE elements:* One potential source of latency is due to processing and/or congestion at OTN/FlexE nodes for virtual link demand splitting/merging. In case of FlexE, a pair of FlexE shims at the source and destination nodes map/recreate FlexE client(s) to/from a group of bonded Ethernet signals. In OTN, a mapper/demapper converts higher layer signal(s) to OTN frame(s) and vice versa. Research literature on the subject reports that the processing delay at FlexE shim or OTN mapper/demapper is negligible compared to other delay along the fibers [45], [46]. Furthermore, OTN is a deterministic transport technology, eliminating queuing delay due to congestion at the nodes. Similarly, FlexE nodes can be scheduled with static resource allocation, thereby avoiding any congestion altogether [46]. Therefore, we ignore the latency incurred by OTN/FlexE nodes in our model.

*b) Transponders (BVTs in case of EONs):* Transponders convert a client demand to an optical signal with just enough spectrum to carry the demand [52]. The latency incurred on transponders varies depending on their design and supported functionality. More complex transponders include functionality such as in-band management and can have latencies in the range 5 – 10 $\mu$s [44]. However, many equipment vendors offer simpler and lower-cost transponders without features such as in-band management. These devices result in transponder processing time $L_{txp}$ as low as 30 ns [2].

*c) FEC processing:* FEC is used for detecting and correcting transmission errors due to noise and other impairments present in transmissions. A stronger FEC increases system margin for a given Bit Error Rate (BER) and optical signal power, thereby increasing the signal-to-noise ratio, enabling longer optical reaches. FEC encoder/decoder modules introduce a processing delay $L_{fec}$ of $\approx$10 $\mu$s for standard FECs [46]. However, processing delay for super FECs with increased correction ability can go up to 150 $\mu$s [44]. Since a pair of FEC modules and transponders are involved in a lightpath, the delay introduced at the terminal nodes is:

$$L_n = 2(L_{txp} + L_{fec}) \tag{1}$$

*2) Latency contributors on a lightpath:* A lightpath can span several optical fibers through several bypass EON nodes. Depending on the path's length there can be re-generators, amplifiers, dispersion compensating fibers (DCF), and ROADMs placed along the path, each contributing towards latency.

*a) Fiber propagation:* The major latency contributor on a lightpath is the propagation delay, $L_{prop}$, which amounts to $\approx$4.9 $\mu$s per kilometer of fiber.

*b) Regenerator:* 3R (re-amplification, re-shaping, retiming) regenerators are used to increase the lightpath reach. They significantly contribute to latency, as 3R regeneration involves optical-to-electrical-to-optical conversion that can take $\approx$100 $\mu$s [44]. In this study, we assume that the substrate EON does not contain any 3R regenerators (rather the modulation format and FEC levels can be tuned to attain different reaches for a transparent lightpath). Therefore, we do not consider any latency incurred by 3R re-generators.

*c) Amplifier:* Amplifiers are needed to boost the signal strength on long transmission lines [43]. Unlike regenerators, amplifiers operate completely in the optical domain, eliminating the need to separately amplify each individual channel. Erbium doped fiber amplifiers (EDFAs), widely used in long-haul networks, introduce a delay, $L_{amp}$, of $\approx$150 ns [2]. One way to eliminate this delay is to use more expensive Raman amplifiers [44]. The number of amplifiers on a lightpath $p$, $n_{amp}(p)$ is $\approx \lceil len(p)/f_{span} \rceil$, where $len(p)$ is the physical length of $p$ and $f_{span}$ is the typical distance between two amplifiers, called fiber span (in the order of 80km).

*d) Dispersion compensating fiber:* Another potential latency contributor in a fiber transmission line is the dispersion compensating fiber (DCF) [2]. DCF is used to compensate Chromatic Dispersion (CD) of the optical signal that stems from the differential speeds of lightpaths occupying different spectrum ranges in fibers. The length of DCF is typically between 15% to 25% of the overall fiber length, which increases fiber propagation delay [43]. Nonetheless, in modern coherent transmission systems, CD is mainly compensated through digital signal processing (DSP) at the receiver, eliminating the need for DCF [53]. Therefore, we do not consider any latency pertaining to dispersion compensation in our model.

*e) ROADM and BV-OXC:* Another component present along a transparent lightpath is the ROADM, installed on the bypass EON nodes. ROADMs may also include BV-OXC that can switch spectrum at intermediate EON nodes. Since optical flows are optically and independently switched by these intermediate devices for different lightpaths, the delays introduced by ROADMs and BV-OXCs, $L_{roadm}$, are in the order of tens of $ns$ [1], [45]. The total number of ROADMs (or, BV-OXCs) on a lightpath $p$ is $(|p| + 1)$, where $|p|$ is the number of SLinks on the lightpath.

Based on the above discussion, the latency incurred on a lightpath can be expressed as follows:

$$L_p = L_n + len(p)L_{prop} + n_{amp}(p)L_{amp} + (|p| + 1)L_{roadm} \tag{2}$$

In case of VLink embedding with splits (as discussed in Section III-B), the set of lightpaths supporting a VLink $\bar{e} \in \bar{E}$ can differ with each other in terms of physical length and number of intermediate hops, resulting in different latency for these lightpaths. Hence, the latency of $\bar{e}$ is determined by the lightpath $p$ with the maximum $L_p$ since the destination OTN/FlexE node has to wait for the slowest split before merging all the splits of a VLink. If $\mathcal{P}_{\bar{e}}$ is the set of paths used for embedding the splits of a VLink $\bar{e} \in \bar{E}$, the latency perceived for the VLink $\bar{e} \in \bar{E}$ is as follows:

$$latency(\bar{e}) = \max_{p \in \mathcal{P}_{\bar{e}}} L_p \tag{3}$$

*3) Differential delay requirement:* VLink embedding by splitting its demand over multiple SPaths imposes additional buffering overhead at the destination OTN/FlexE node to offset the different delays experienced by different splits of the VLink transmitted on different lightpaths. This is also known as the differential delay [54], [55]. To account for differential delay, the destination node needs to store all but the most delayed flow in a buffer until the last flow arrives. The amount of buffer needed to compensate the differential delay depends on both the data-rates of the individual paths and the maximum allowed differential delay ($DD_{max}$). Both VCAT and FlexE impose very strict bound on $DD_{max}$, specifically, 250 $\mu$s and 10 $\mu$s, respectively [51], [55]. We express differential delay requirement for a VLink in terms of the maximum and minimum delay of its embedding SPath set as follows:

$$\forall \bar{e} \in \bar{E} : (\max_{p \in \mathcal{P}_{\bar{e}}} L_p - \min_{p \in \mathcal{P}_{\bar{e}}} L_p) \leq DD_{max} \tag{4}$$

## D. Problem Statement

Given an SN $G$, a reach table $\mathcal{R}$, and a VN request $\bar{G}$ with VNode mapping function $\tau$ and latency constraint set $\mathcal{L}(\bar{G})$:

- Compute the link embedding function $\gamma : \bar{E} \rightarrow \chi : \chi \subset \mathcal{P} \times \mathcal{T} \times S^2$ and $1 \leq |\chi| \leq q$, *i.e.*, compute up to a maximum of $q$ splits for the bandwidth demand $\beta_{\bar{e}}$ of each VLink $\bar{e} \in \bar{E}$. For each split, $\gamma$ should select an SPath and an appropriate transmission configuration $t \in \mathcal{T}$ from the reach table $\mathcal{R}$, and allocate a contiguous segment of slices represented by the starting and ending slice index on each

SLink along the SPath. Note that the same SPath can be used multiple times as the splits of a VLink following the reasoning in [26]. $\chi_{\bar{e}i} = (p, t, s_b, s_t)|1 \le i \le q$ represents the $i$-th split, where $\chi_{\bar{e}i}^{(p)}$ and $\chi_{\bar{e}i}^{(t)}$ denote the selected SPath and transmission configuration for the $i$-th split, respectively. In addition, allocation of spectrum slices for the $i$-th split begins at index $\chi_{\bar{e}i}^{(s_b)}$ and ends at index $\chi_{\bar{e}i}^{(s_t)}$ along each SLink in the SPath $\chi_{\bar{e}i}^{(p)}$. The VLink embedding should be computed in a way such that the latency constraints on VPaths are satisfied, *i.e.*, $\forall \ell \in \mathcal{L}(\bar{G}) : latency(\ell^{(\bar{a})}) \le \ell^{(L)}$.

- The total number of slices required to provision the VN is minimum according to the following cost function:

$$\sum_{\forall \bar{e} \in \bar{E}} \sum_{i=1}^{q} (\chi_{\bar{e}i}^{(s_t)} - \chi_{\bar{e}i}^{(s_b)} + 1) \times |\chi_{\bar{e}i}^{(p)}| \tag{5}$$

  Here, $|\chi_{\bar{e}i}^{(p)}|$ is the number of SLinks on the SPath $\chi_{\bar{e}i}^{(p)}$.

The above is subject to substrate resource constraint, and spectral contiguity (*i.e.*, the allocated slices of each split are always adjacent to each other) and continuity (*i.e.*, the same sequence of slices are allocated on each SLink along an SPath) constraint, and differential delay requirement on the lightpaths.

### E. Pre-computations

For each VLink $\bar{e} \in \bar{E}$, we pre-compute $\mathcal{P}_{\bar{e}}^{k}$, a set of $k$ shortest paths between the pair of SNodes where the VLink's endpoints' are mapped. For each SPath $p \in \mathcal{P}_{\bar{e}}^{k}$, we pre-compute the set of admissible transmission configurations, $\mathcal{T}_{\bar{e}p} \subset \mathcal{T}$, such that each configuration $t \in \mathcal{T}_{\bar{e}p}$ results in a reach $r_t \ge len(p)$ and has a data-rate $t^{(d)}$. $\mathcal{T}_{\bar{e}}$ contains all the distinct tuples suitable for $\bar{e}$ and is defined as $\bigcup_{\forall p \in \mathcal{P}_{\bar{e}}^{k}} \mathcal{T}_{\bar{e}p}$.

## IV. Problem Formulation

We present a path-based ILP formulation for optimally solving the problem given a set of candidate SPaths for each VLink. Note that some of the constraints except the latency and differential delay constraints have been presented in different forms in different research works [26], [56], [57]. In the interest of completeness, we present both the common ones and the ones specifically required for the problem at hand.

### A. Decision Variables

We allow a VLink's bandwidth demand to be satisfied by provisioning slices over one or more SPaths where an SPath can be used more than once (up to a maximum of $q$) as discussed in III-D. To model the same SPath appearing more than once in a VLink's embedding, we assume each transmission configuration on an SPath can be instantiated multiple times (up to a maximum of $q$ times). The following variable represents VLink mapping:

$$w_{\bar{e}pti} = \begin{cases} 1 & \text{if } \bar{e} \in \bar{E} \text{ uses } i\text{-th instance of } t \in \mathcal{T}_{\bar{e}p} \\ & \quad \text{on path } p \in \mathcal{P}_{\bar{e}}^{k} \\ 0 & \text{otherwise} \end{cases}$$

Finally, the following decision variable creates the relationship between a mapped SPath and the slices in its SLinks:

$$y_{\bar{e}ptis} = \begin{cases} 1 & \text{if } \bar{e} \in \bar{E} \text{ uses slice } s \in S \text{ on path } p \in \mathcal{P}_{\bar{e}}^{k} \\ & \text{with the } i\text{-th instance of } t \in \mathcal{T}_{\bar{e}p} \\ 0 & \text{otherwise} \end{cases}$$

### B. Constraints

*1) VLink demand constraints:* We provision a VLink by splitting it across up to $q$ SPaths. Constraint (6) ensures that for each VLink $\bar{e} \in \bar{E}$, the sum of data-rates resulting from applying the selected transmission configuration on the selected paths is equal to the VLink's demand. Then, (7) enforces an upper limit on the number of splits.

$$\forall \bar{e} \in \bar{E} : \sum_{\forall p \in \mathcal{P}_{\bar{e}}^{k}} \sum_{\forall t \in \mathcal{T}_{\bar{e}p}} \sum_{i=1}^{q} (w_{\bar{e}pti} \times t^{(d)}) = \bar{\beta}_{\bar{e}} \tag{6}$$

$$\forall \bar{e} \in \bar{E} : \sum_{\forall p \in \mathcal{P}_{\bar{e}}^{k}} \sum_{\forall t \in \mathcal{T}_{\bar{e}p}} \sum_{i=1}^{q} w_{\bar{e}pti} \le q \tag{7}$$

*2) Slice assignment and Spectral Contiguity constraints:* We ensure by (8) that if a path $p$ is selected with a specific transmission configuration $t$, then the required number of slices $n_t$ to support the data-rate $t^{(d)}$ is allocated on the path. (9) ensures that each slice on an SLink is allocated to at most one path. Finally, (10) ensures the slices allocated on each link of a path form a contiguous frequency spectrum.

$$\forall \bar{e} \in \bar{E}, \forall p \in \mathcal{P}_{\bar{e}}^{k}, \forall t \in \mathcal{T}_{\bar{e}p}, 1 \le i \le q : \sum_{\forall s \in S} y_{\bar{e}ptis} = n_t w_{\bar{e}pti} \tag{8}$$

$$\forall e \in E, \forall s \in S : \sum_{\forall \bar{e} \in \bar{E}} \sum_{\forall p \in \mathcal{P}_{\bar{e}}^{k}} \sum_{\forall t \in \mathcal{T}_{\bar{e}p}} \sum_{i=1}^{q} w_{\bar{e}pti} y_{\bar{e}ptis} \delta_{pe} \le 1 \tag{9}$$

$$\forall \bar{e} \in \bar{E}, \forall p \in \mathcal{P}_{\bar{e}}^{k}, \forall t \in \mathcal{T}_{\bar{e}p}, 1 \le i \le q, 1 \le s \le |S| - 1 :$$
$$\sum_{s'=s+2}^{|S|} y_{\bar{e}ptis'} \le |S| \times (1 - y_{\bar{e}ptis} + y_{\bar{e}pti(s+1)}) \tag{10}$$

*3) Latency Constraints:* Recall from Section III-B1 that the VN embedding has to satisfy a set of path-based latency constraints $\mathcal{L}(\bar{G})$. Since each such constraint $\ell \in \mathcal{L}(\bar{G})$ involves one or more VLinks, we first introduce a new decision variable $vlink\_lat_{\bar{e}}$ that finds the latency of each VLink $\bar{e}$ using the following linear constraint derived from (3):

$$\forall \bar{e} \in \bar{E}, \forall p \in \mathcal{P}_{\bar{e}}^{k}, \forall t \in \mathcal{T}_{\bar{e}p}, 1 \le i \le q : L_p w_{\bar{e}pti} \le vlink\_lat_{\bar{e}} \tag{11}$$

The following constraint uses (11) to ensure that each latency constraint $\ell \in \mathcal{L}(\bar{G})$ is satisfied:

$$\forall \ell \in \mathcal{L}(\bar{G}) : \sum_{\forall \bar{e} \in \ell^{(\bar{a})}} vlink\_lat_{\bar{e}} \le \ell^{(L)} \tag{12}$$

*4) Differential Delay Constraints:* Recall from Section III-C3, VLink embedding must satisfy the differential delay requirement (4). We use (11) and (13) to find the maximum and minimum delay of each VLink, respectively. In (13), we need to avoid getting zero as the minimum latency of a VLink induced by a non-selected path, tuple, and instance combination. The second term on the *r.h.s* of (13) achieves this goal by assuming that $\lambda$ is larger than the maximum value of $L_p$ in the EON. For a non-selected path combination, second term on the *r.h.s* of (13) becomes active, generating constraint $min\_delay(\bar{e}) \leq \lambda$. For a selected path, tuple, and instance combination, the second term becomes zero and the first term generates the constraint $min\_delay(\bar{e}) \leq L_p$. As $\lambda >> L_p$, $min\_delay(\bar{e}) \leq L_p$ dominates over $min\_delay(\bar{e}) \leq \lambda$ to generate the minimum delay of all the selected path, tuple, and instance combinations. Finally, (14) uses (11) and (13) to enforce the differential delay requirements:

$$\forall \bar{e} \in \bar{E}, \forall p \in \mathcal{P}^k_{\bar{e}}, \forall t \in \mathcal{T}_{\bar{e}p}, 1 \leq i \leq q :$$
$$min\_delay(\bar{e}) \leq w_{\bar{e}pti} \times L_p + (1 - w_{\bar{e}pti}) \times \lambda \tag{13}$$
$$\forall \bar{e} \in \bar{E} : vlink\_lat_{\bar{e}} - min\_delay(\bar{e}) \leq DD_{max} \tag{14}$$

### C. Objective Function

Our cost function minimizes the total number of spectrum slices required to embed all the VLinks of a VN as shown in the first part of (15). However, to break ties among multiple solutions with the same total number of slices, we use the second term with a fractional weight $\epsilon$ in (15) that minimizes the number of splits over all the VLinks.

$$\text{minimize}\Big(\sum_{\forall \bar{e} \in \bar{E}} \sum_{\forall p \in \mathcal{P}^k_{\bar{e}}} \sum_{\forall t \in \mathcal{T}_{\bar{e}p}} \sum_{i=1}^{q} \sum_{\forall s \in S} y_{\bar{e}ptis} \times |p| + \epsilon \times \sum_{\forall \bar{e} \in \bar{E}} \sum_{\forall p \in \mathcal{P}^k_{\bar{e}}} \sum_{\forall t \in \mathcal{T}_{\bar{e}p}} \sum_{i=1}^{q} w_{\bar{e}pti}\Big) \tag{15}$$

### D. Hardness of the Problem

A restricted sub-problem of VNE with latency constraints on VPaths is VNE with latency constraints on individual VLinks, *i.e.*, when the lengths of the VPaths are only one. Rost *et al.*, have shown that the latter is NP-complete and cannot be approximated under any objective unless $\mathcal{P} = \mathcal{NP}$ [58]. Therefore, by restriction, the VNE problem with path-based latency constraints is also $\mathcal{NP}$-complete. More recent results have shown that parameterized approximation algorithm in terms of the treewidth of the VN request can be devised for the VNE problem with latency constraints on individual VLinks [59]. However, computing the treewidth of a graph by itself is an $\mathcal{NP}$-hard problem [59].

## V. Heuristic Algorithm

### A. Heuristic solution for VN Embedding

We propose a heuristic solution to tackle the computational complexity of the optimal solution. We first give an overview of the heuristic's embedding process outlined in Alg. 1. Alg. 1 takes as input a VN $\bar{G}$, an EON $G$, set of latency constraints $\mathcal{L}(\bar{G})$, and a node mapping function $\tau : \bar{V} \rightarrow V$. One way of computing a cost efficient VLink mapping for a given VNode mapping is to consider all $|\bar{E}|!$ possible orders of sequentially embedding VLinks. The lowest cost mapping among all possible VLink orders then can be chosen as a cost effective solution for the given VNode mapping. However, this brute-force approach is not scalable. Instead, Alg. 1 considers only one sequential VLink order that is dynamically computed to converge to a solution within a reasonable time.

To embed a VLink $\bar{e}$, one needs to compute the maximum latency allowable for $\bar{e}$ that will not violate any of the latency constraints. This is not trivial because $\bar{e}$ can belong to multiple VPaths with different latency constraints, *e.g.*, VLink *qr* on VPaths *p-r-q* and *q-r-s* in Fig. 1. What makes it even more difficult is that the mappings (and corresponding latencies) of other VLinks on these VPaths may not be known at this time due to the sequential nature of Alg. 1. For instance, at the time of estimating the latency budget of the VLink *qr* in Fig. 1, the mappings and latency budgets of VLinks *rp* and *rs* might not be known. Therefore, Alg. 1 estimates an upper bound of the latency budget of a VLink $\bar{e}$ based on the latencies of already mapped VLinks, and on the estimated latencies and slice availability in the candidate SPaths of the other unmapped VLinks. Assuming the SPaths in $\mathcal{P}^k_{\bar{e}}$ are sorted in increasing order of their lengths, this upper bound is set to the latency of the maximal $i$-th SPath $p_i \in \mathcal{P}^k_{\bar{e}} | i \leq k$, such that this chosen value does not violate any of the latency constraints. Therefore, the estimated latency budget generates an allowed set of candidate SPaths $\mathcal{P}^i_{\bar{e}} = \{p_1, p_2, .., p_i\}$ for $\bar{e}$'s embedding.

**Algorithm 1:** Algorithm for VN Embedding

```
1  function VNEmbedding($G, \bar{G}, \mathcal{L}(\bar{G}), \tau$)
2      $\bar{E}_{remaining} \leftarrow \bar{E}$
3      while $\bar{E}_{remaining} \neq \phi$ do
4          $< \bar{e}, i > \leftarrow$
               GetNextVLinkToEmbed($G, \bar{G}, \bar{E}_{remaining}, \mathcal{P}^k_{\bar{e}}$)
5          $Z_{\bar{e}} \leftarrow$FindOptimal($G, \bar{e}, \mathcal{P}^i_{\bar{e}}, \mathcal{T}_{\bar{e}}$)
6          foreach $e \in p | p \in \mathcal{I}_{\bar{e}}.\mathbb{P}^i_{\bar{e}}$ do
7              Perform slice assignment using $\mathcal{I}_{\bar{e}}.\mathcal{S}$
8          $\chi_{\bar{e}}.\mathcal{P} \leftarrow \mathcal{I}_{\bar{e}}.\mathbb{P}^i_{\bar{e}}$, $\chi_{\bar{e}}.\mathcal{T} \leftarrow \mathcal{I}_{\bar{e}}.\mathbb{T}_{\mathbb{P}^i_{\bar{e}}}$
9          if $\chi_{\bar{e}} = \phi$ then return $< \phi, \phi >$
10         $\bar{E}_{remaining} \leftarrow \bar{E}_{remaining} - \{\bar{e}\}$
11     return $\gamma : \bar{E} \rightarrow \chi >$
```

To increase the chances of finding a feasible solution, Alg. 1 embeds the most constrained VLink $\bar{e}$ in terms of slice availability in its allowed set of candidate SPaths $\mathcal{P}^i_{\bar{e}}$ during each iteration. Alg. 1 uses Alg. 2 to find the most constrained VLink $\bar{e}$ from the set of unmapped VLinks and the allowed set of candidate SPaths $\mathcal{P}^i_{\bar{e}}$ for $\bar{e}$ (details of Alg. 2 will be given in the next subsection). For the chosen VLink $\bar{e}$, Alg. 1 invokes $FindOptimal$ procedure to compute the optimal solution for $\bar{e}$ based on $\mathcal{P}^i_{\bar{e}}$ (line 5). $FindOptimal$ procedure is adapted from [26] to take latency and differential delay constraints into account. Alg. 1 then allocates spectrum slices on all SLinks present in the solution returned by $FindOptimal$ and updates the VLink mapping function $\chi$. If no solution can be found for any $\bar{e}$, VN embedding fails. Note that after a VLink is mapped, its actual latency can be computed from the set of SPaths used for mapping, and slice availability in the candidate SPaths of the unmapped VLinks will need to be updated. Therefore, Alg. 2 is invoked in each iteration of Alg. 1 to dynamically find the most constrained VLink leveraging the up to date status of embedding and EON.

### B. VLink Ordering and Latency Budget Allocation

Alg. 2 estimates the latency budgets of the unmapped VLinks in a way such that embedding of VLinks appearing later in the order become more unlikely to fail due to very tight budgets or due to insufficient number of contiguous slices in their candidate SPaths. For instance, the estimated latency budget of the VLink $qr$ belonging to VPaths $p$-$r$-$q$ and $q$-$r$-$s$ in Fig. 1 should not be too large to leave close to zero latency budgets for other VLinks on $p$-$r$-$q$ or $q$-$r$-$s$. One way of ensuring this is to maximize the number of allowed candidate SPaths for the VLinks while satisfying the latency constrains. While this approach works well in a green-field EON with ample spectrum slices, it may lead to infeasible embedding in a dynamic environment where some of the SPaths in $\mathcal{P}_{\bar{e}}^{i}$ are fragmented due to accommodating existing VNs. Therefore, Alg. 2 takes slice availability in the candidate SPaths into account while estimating latency budgets.

Alg. 2 computes an index $i_{\bar{e}}$ for each VLink $\bar{e}$, denoting the maximum index of the SPath in the set of allowed SPaths for $\bar{e}$ corresponding to the latency budget. While computing this index, Alg. 2 employs spectrum awareness by maximizing the minimum number of usable slices in the allowed set of SPaths for each of the remaining VLinks while satisfying all latency constraints. To do so, Alg. 2 performs binary search over the interval between 1 and the minimum value of the number of free slices in the candidate SPaths of the remaining VLinks. In each iteration of binary search, Alg. 2 chooses a median value $med$ in the range and increases $i_{\bar{e}}$ until the total number of free slices in the SPaths in $\mathcal{P}_{\bar{e}}^{i}$ becomes equal or more than $med$. It then uses $i_{\bar{e}}$ to compute the estimated latency of $\bar{e}$. After estimating latencies for all the unmapped VLinks, Alg. 2 checks if any latency constraint in $\mathcal{L}(\bar{G})$ is violated. Based on this test, the binary search continues in the appropriate range and terminates when the range size becomes 1.

The binary search in Alg. 2 provides a lower bound $a$ on the minimum number of usable slices across all the VLinks. However, it does not identify the VLink(s) with the minimum number of usable slices. According to the property of binary search, there exists at least one VLink for which $i_{\bar{e}}$ cannot be increased beyond $a$ without violating at least one latency constraint. Alg. 2 identifies such VLink by attempting to increase the $i_{\bar{e}}$ for each virtual link $\bar{e}$ and checking if all the latency constraints can still be satisfied using the expanded set of allowed SPaths. The VLink that fails this test is then returned as the most constrained one along with its corresponding index $i_{\bar{e}}$ (line 28). If multiple such VLinks exist, Alg. 2 returns the one with higher demand, which is ensured by traversing the VLinks in decreasing order of their demand (line 23).

*a) Running Time Analysis:* Alg. 2 performs a binary search on a range of $k \times |S|$, resulting in $O(\log k|S|)$ iterations. In each iteration of the search, latency budget estimation can take $O(k|\bar{E}|)$ time since we need to check all the VLinks and all their $k$ candidate shortest paths at the initial stage. Each iteration of the binary search also involves checking $|\mathcal{L}|$ latency constraints, each of which can have $|\bar{V}|$ VLinks, requiring $O(|\mathcal{L}||\bar{V}|)$ time. Finally, finding the most constrained VLink takes $|\bar{E}| \times |\mathcal{L}||\bar{V}|$ time as $|\mathcal{L}|$ latency constraints need to be checked for each VLink. Therefore, the total running time of Alg. 2 is $O(\log(k|S|) \times (k|\bar{E}| + |\mathcal{L}||\bar{V}|) + |\bar{E}| \times |\mathcal{L}||\bar{V}|)$.

**Algorithm 2:** Algorithm for finding a VLink based on latency constraints and spectrum availability

```
1  function GetNextVLinkToEmbed($G, \bar{G}, \bar{E}_{rest}, \mathcal{P}_{\bar{e}}^{k}, \mathcal{L}(\bar{G})$)
2      foreach $\bar{e} \in \bar{E} \setminus \bar{E}_{rest}$ do
3          $p \leftarrow$ SPath with the longest length in $\chi_{\bar{e}}.\mathcal{P}$
4          $est\_latency(\bar{e}) \leftarrow L_p$ using (2)
5      $\mathcal{L} \leftarrow$ all latency constraints in $\mathcal{L}(\bar{G})$
6      $a \leftarrow 1, b \leftarrow \min_{\bar{e} \in \bar{E}_{rest}} \sum_{p_i \in \mathcal{P}_{\bar{e}}^{k}} free\_slices(p_i)$
7      while $b - a > 1$ do
8          $med \leftarrow \frac{a+b}{2}$
9          foreach $\bar{e} \in \bar{E}_{rest}$ do
10             $i_{\bar{e}} \leftarrow free(\bar{e}) \leftarrow 0$
11             while $free(\bar{e}) < med$ do
12                 $i_{\bar{e}} \leftarrow i_{\bar{e}} + 1$
13                 $free(\bar{e}) \leftarrow free(\bar{e}) + free\_slices(p_{i_{\bar{e}}})$
14             $est\_latency(\bar{e}) \leftarrow L_{p_{i_{\bar{e}}}}$ using (2)
15         all_constraints_satisfied$\leftarrow true$
16         foreach $l \in \mathcal{L}$ do
17             $est\_latency(\ell^{(\bar{a})}) = \sum_{\bar{e} \in \bar{a}} est\_latency(\bar{e})$
18             if $est\_latency(\ell^{(\bar{a})}) > \ell^{(L)}$ then
19                 all_constraints_satisfied$\leftarrow false$
20                 break
21         if all_constraints_satisfied $= true$ then $a \leftarrow med$
22         else $b \leftarrow med$
23     foreach $\bar{e} \in \bar{E}_{rest}$ in decreasing order of $\bar{\beta}_{\bar{e}}$ do
24         $i_{\bar{e}} \leftarrow i_{\bar{e}} + 1$
25         $est\_latency(\bar{e}) \leftarrow L_{p_{i_{\bar{e}}}}$ using (2)
26         foreach $l \in \mathcal{L}$ do
27             $est\_latency(\ell^{(\bar{a})}) = \sum_{\bar{e} \in \bar{a}} est\_latency(\bar{e})$
28             if $est\_latency(\ell^{(\bar{a})}) > \ell^{(L)}$ and $free(\bar{e}) = a$
                 then return $< \bar{e}, i_{\bar{e}} - 1 >$
```

### C. Compute optimal solution for a single VLink

Alg. 1 invokes $FindOptimal$ procedure that computes the optimal embedding of a VLink given its mapped endpoints and a set of allowed SPaths in $\mathcal{P}_{\bar{e}}^{i}$. The optimal solution for a VLink consists of a multi-set of SPaths where each SPath in the multi-set has an associated transmission configuration and spectrum slice allocation. $FindOptimal$ is adapted from [26] that exhaustively explores all possible multi-sets of SPaths in $\mathcal{P}_{\bar{e}}^{i}$. We modified $FindOptimal$ to consider only those multi-sets of SPaths that satisfy the differential delay constraints enforced by (14). For each of the candidate multi-set of SPaths, $FindOptimal$ checks the feasibility of all possible transmission configurations. Finally, for each pair of a multi-set of SPaths and a corresponding transmission configuration, $FindOptimal$ performs slice allocation using First-Fit approach [30]. Among all the feasible solutions consisting of multi-set of SPaths, an associated transmission configuration, and a spectrum slice allocation, $FindOptimal$ returns the one that requires the minimum number of spectrum slices.

## VI. Evaluation

### A. Simulation Setup

*1) Testbed and Topology:* We have implemented both the ILP formulation and the heuristic algorithm in C++. We use Nobel Germany (17 nodes and 26 links) and Germany50 (50 nodes and 88 links) networks from SNDlib repository [29] as the SNs for small and large scale simulations, respectively. Ten shortest paths between all pairs of SNodes in each SN are pre-computed as input. Spectrum bandwidth on each SLink is set to 600GHz and 4THz for small scale and large scale simulations, respectively. For the steady state analysis, we use the Nobel Germany SN and set 4THz spectrum bandwidth on the SLinks. Depending on the evaluation scenario, we consider fixed- and flex-grid variations of the SN. The fixed-grid variation allocates spectrum slices in 50Ghz granularity and considers only a few data rates supported by the transponders (*i.e.*, 100Gbps, 200Gbps, and 400Gbps). Whereas, the flex-grid case allocates spectrum in 12.5Ghz granularity and allows transponders to support a higher number of data rates up to 800Gbps. To achieve a given data rate, a transponder is allowed to choose from a number of transmission configurations provided in reach tables (as discussed in Section III-A) for both fixed- and flex-grid variations. Latency characteristics of the SNs are set according to Table I. We synthetically generate the VNs with different properties based on the simulation scenario. We map each VNode to a random SNode while ensuring that no two VNodes from the same VN are mapped to the same SNode. VLink demands are varied between 100Gbps to 1Tbps with possible values as multiples of 100Gbps. Simulations are performed on a machine with $8\times10$-core 2.40GHz Intel Xeon E7-8870 processors and 1TB memory.

*2) VN generation for microbenchmarking:* In this scenario, we consider each VN request in isolation for each of the compared approaches discussed in Section VI-A5. We vary the VNs by changing their link to node ratio (LNR) and the total bandwidth demand, while keeping the number of VNodes fixed at 8 and 50 for small- and large-scale simulations, respectively. For each simulation run, we generate 5 and 50 VNs with the same LNR and similar total bandwidth demands for small- and large-scale simulations, respectively, and report the mean of the performance metrics over those VNs.

*3) VN generation for steady state analysis:* This analysis considers VN arrivals and departures over a period of time. This provides insight into the number of accepted VNs for different compared variants. We have developed an in-house discrete event simulator that generates simulation scenarios representing the arrival and departure of VNs. The VN arrival rate of each scenario follows a Poisson distribution with a given mean. We vary the mean of Poisson distribution between 4 – 12 VNs per 100 time units. VN life time in all the scenarios is exponentially distributed with a mean of 100 time units. The number of VNodes of each arrived VN is kept at 8, whereas, the LNR of the VN is chosen randomly between 1 and 3.5. The simulation time of each scenario is 10000 time units, and we exclude measurements from the first 1000 time instances to capture the steady state performance. For each problem instance, we generate 5 random simulation scenarios and report mean performance metrics to gain statistical confidence. These parameters have been chosen in accordance with those used in the network virtualization literature (*e.g.*, [60]–[65]).

*4) Latency constraint generation:* For each VN, we generate $|\bar{E}|$ VPath-based latency constraints. To do so, we sort all the shortest VPaths between all pairs of VNodes in non-increasing order (in terms of the number of VLinks) and choose the first $|\bar{E}|$ paths. For each selected VPath $\bar{a}$, we generate a latency budget $\ell^{(\bar{a})}$ by leveraging the set of input candidate SPaths for each VLink $\bar{e} \in \bar{a}$, which ensures the existence of a feasible latency budget for embedding. Otherwise, an arbitrarily generated latency budget can be too strict (or too relaxed) causing none (or all) of the candidate SPaths to satisfy that budget. If $p^*_{\bar{e}}$ is the SPath with shortest distance among the set of candidate SPaths for a VLink $\bar{e} \in \bar{a}$, then the latency budget for the VPath $\bar{a}$ is generated as follows:

$$\ell = (\bar{a}, L): \quad \ell^{(L)} = \alpha \sum_{\forall \bar{e} \in \bar{a}} L_{p^*_{\bar{e}}} \tag{16}$$

$1.0 \leq \alpha \leq 2.0$ is a tuning parameter allowing us to vary the strictness of the latency constraints. For instance, $\alpha = 1.0$ implies only the shortest candidate SPath $p^*_{\bar{e}}$ for each VLink in $\bar{a}$ can satisfy the latency constraints. When $\alpha > 1.0$, (16) increases the latency bound for a VPath $\bar{a}$ allowing more candidate SPaths for each VLink to satisfy the constraints.

TABLE II
COMPARED VARIANTS

| **Variant** | **Latency Constraints** | **Differential Delay Constraints** |
|---|---|---|
| L($\infty$)-DD($\infty$) | $\alpha = \infty$ | $DD_{max}$=$\infty$ |
| L($\alpha$)-DD($\infty$) | Variable $\alpha$ | $DD_{max}$=$\infty$ |
| L($\infty$)-DD(250) | $\alpha = \infty$ | $DD_{max}$=250 $\mu$s |
| L($\infty$)-DD(10) | $\alpha = \infty$ | $DD_{max}$=10 $\mu$s |
| L($\alpha$)-DD(250) | Variable $\alpha$ | $DD_{max}$=250 $\mu$s |
| L($\alpha$)-DD(10) | Variable $\alpha$ | $DD_{max}$=10 $\mu$s |

*5) Compared Variants:* We consider six problem variants representing different combinations of latency and differential delay constraints (Table II). L($\infty$)-DD($\infty$) is the baseline that considers neither latency nor differential delay constraints, whereas, L($\alpha$)-DD($\infty$) imposes only latency constraints with varying $\alpha$. In contrast, L($\infty$)-DD(250) and L($\infty$)-DD(10) consider only differential delay constraints. Depending on the different values of $DD_{max}$ imposed by the enabling technologies discussed in Section III-C3, there are two more variants L($\infty$)-DD(250) and L($\infty$)-DD(10). Finally, L($\alpha$)-DD(250) and L($\alpha$)-DD(10) apply both constraints for varying $\alpha$ with the corresponding values of $DD_{max}$, respectively.

## B. Evaluation Metrics

*a) Spectrum slice usage (SSU):* The percentage of spectrum slices allocated to a VN embedding with respect to the total number of slices in the SN.

*b) Avg. number of splits used for a VLink (NSU):* The ratio of the total number of splits used to embed the VLinks of a VN to the total number of VLinks in a VN.

*c) Avg. number of distinct SPaths used for a VLink (NDP):* The ratio of the total number of distinct SPaths used to map the VLinks of a VN to the number of VLinks in a VN.

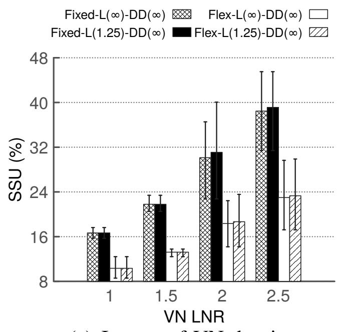
*d) Execution Time:* The time required for an algorithm to find a VN embedding.

*e) Blocking ratio:* The fraction of VN requests over all the VN requests that could not be embedded on the EON.

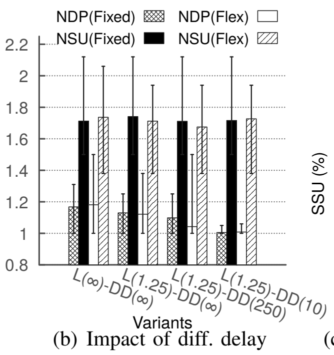
*f) Cost ratio:* The ratio of costs obtained by two different approaches for solving the same problem instance, where cost is computed using (15).

## C. Microbenchmarks

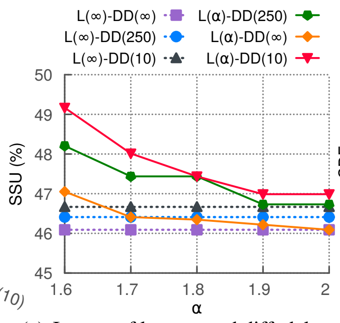
*1) Impact of latency and differential delay constraints:* Fig. 2(a) shows the impact of latency constraint ($\alpha$=1.25) on



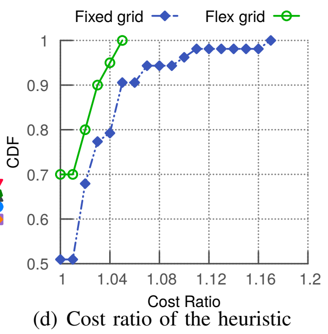
(a) Impact of VN density



(b) Impact of diff. delay



(c) Impact of latency and diff. delay



(d) Cost ratio of the heuristic

Fig. 2. Small scale analysis

SSU, compared to the case where VNs do not apply any latency guarantee (L($\infty$)-DD($\infty$)), for both fixed- and flex-grid EON. To show the actual impact, ILP formulation is executed on VNs with LNR ranging from 1 to 2.5. We observe that relatively-sparse VNs (*i.e.*, LNR $< 2$) do not exhibit much increase in SSU for both fixed- and flex-grid, while denser VNs (*i.e.*, LNR $\geq 2$) result in higher SSU when subject to latency guarantees. This is because VLinks in dense VNs cannot always use the most spectrally-efficient SPaths due to spectrum resource exhaustion/fragmentation on those SPaths. The increase in SSU is more marked for fixed-grid due to its lack of flexibility in transmission configuration and spectrum allocation. Finally, note that, in the solutions obtained with L($\infty$)-DD($\infty$), up to 32% of latency constraints would be violated, which demonstrates the importance of having an explicit enforcement of latency bounds.

Fig. 2(b) shows how NDP and NSU are affected by the differential delay constraints. For instance, $DD_{max} = 10$ forces the splits of a VLink to use SPaths that differ by at most 2km in length. To ensure that in Nobel Germany SN, the same SPath is used by all the splits of a VLink in most cases of our simulation, as reflected by the values of NDP ($\approx$1) and NSU ($>$1) incurred by L($\infty$)-DD(10). Such strict requirement also rendered some of the VN embeddings infeasible. The NDP for L($\infty$)-DD(250) does not significantly differ from that of L($\infty$)-DD($\infty$). In fact, since the objective function minimizes spectrum usage, shorter SPaths are preferred whenever possible and splits with high length difference among candidate SPaths are selected rarely.

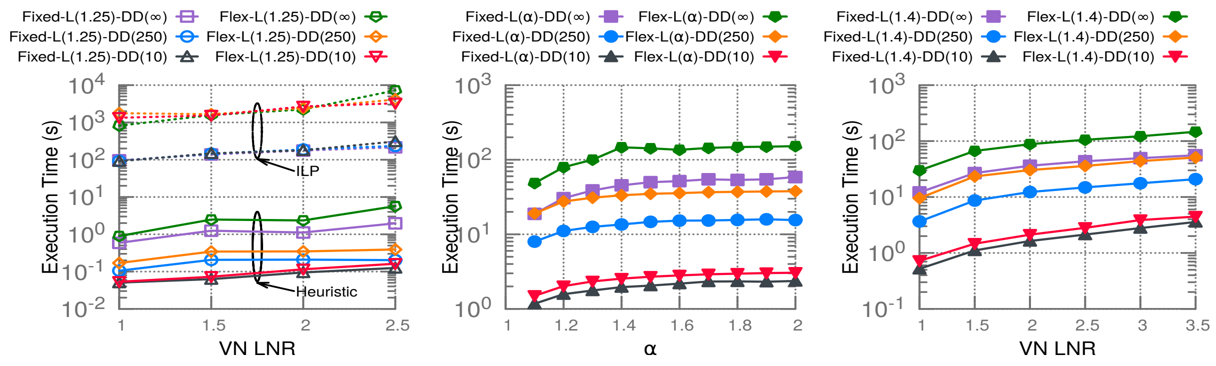
We observed that considering differential delay constraints does not significantly impact resource usage, *i.e.*, SSU. This is because the first few candidate SPaths (sorted in increasing order of length) of a VLink had sufficient capacity in most cases, allowing VLinks to be mapped on them. In order to force the VLinks to use a wider range of SPaths, we deliberately generated a set of VNs that maximize the VLinks with overlapping candidate SPaths, to increase the chances of having bottleneck SLinks along the first few candidate SPaths. For this scenario, Fig. 2(c) presents the SSU (obtained using the ILP) on fixed-grid while varying the latency budget. Even in such a scenario, we do not observe significant impact of the differential delay constraint on SSU. Rather, latency constraints have larger impact on SSU as the $\alpha$ governs which SPaths can be used in the embedding of a VLink.

*2) Comparison between Heuristic and Optimal solution:* Our cost function (15) is dominated by resource usage in the EON. Therefore, cost ratio between heuristic and ILP gives an empirical measure of how much additional resources are allocated by the heuristic. We present the cumulative distribution function (CDF) of cost ratios in Fig. 2(d). Over all instances, the heuristic resulted in only 2.5% and 0.8% additional cost on average compared to the optimal solution for fixed- and flex-grid EONs, respectively.

*3) Scalability Analysis:* Fig. 3(a) presents a comparison between the heuristic's and the ILP's execution time for solving the same problem instances across different variants. In general, flex-grid increases complexity due to higher number of data rates and finer spectrum granularity, resulting in increased running time for both ILP and heuristic. Higher LNR implies a larger number of VLinks, hence, the larger execution time. ILP's execution time is not influenced by the adoption of differential delay constraint. However, the heuristic becomes faster for the same scenarios as it prunes the search space by considering only the combination of SPaths that satisfy differential delay constraint. In all cases, heuristic was observed to run 3 to 4 orders of magnitude faster than ILP. We also run the heuristic on much larger problem instances (described in Section VI-A1) and report the execution time by varying $\alpha$ in Fig. 3(b) and by varying VN LNR in Fig. 3(c). Fig. 3(b) shows that increasing $\alpha$ expands heuristic's search space by allowing more SPaths as the potential candidates, resulting in increased running time. Fig. 3(c) shows that even for a VN with as many as 175 VLinks (*i.e.*, VN LNR 3.5), the heuristic finds a solution in $\approx$2 minutes.

## D. Steady State Analysis

We simulate the arrival of VNs at different rates on both fixed- and flex-grid EON and embed a VN as it arrives using the heuristic. Fig. 4 presents the blocking ratio for a number of variants by varying arrival rate of VNs. L($\infty$)-DD($\infty$), our baseline variant, yields the lowest blocking ratio, but does not guarantee that all the latency constraints will be satisfied. Among the compared variants, the one with the strictest latency constraint (*i.e.*, $\alpha = 1.1$) results in the highest blocking ratio, *i.e.*, 20% and 12% more on average than the baseline for fixed- and flex-grid EONs. This is because only a small fraction of the candidate SPaths of a VLink can satisfy

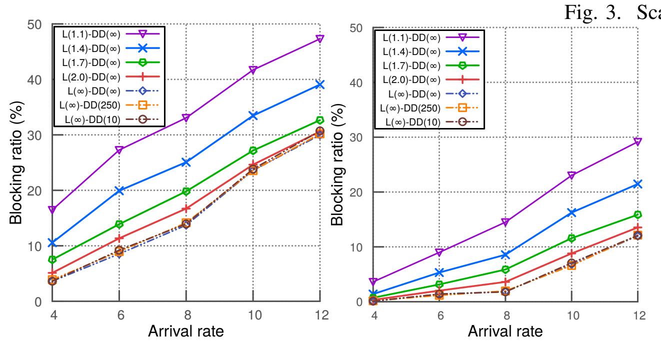


(a) Impact of VN density in small scale

(b) Impact of $\alpha$ in large scale

(c) Impact of VN density in large scale

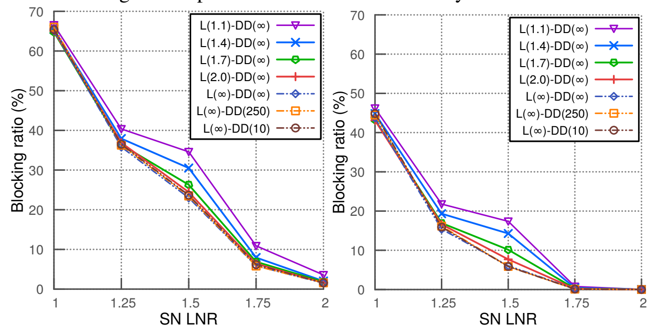
Fig. 3. Scalability analysis



(a) Fixed-grid

(b) Flex-grid

Fig. 4. Impact of arrival rate and latency constraints



(a) Fixed-grid

(b) Flex-grid

Fig. 5. Impact of SN density and latency constraints

such latency constraints, which also skews the resource usage on the SLinks of those SPaths. Relaxing the latency constraint (*i.e.*, increasing $\alpha$) allows the VNs to be embedded using other (potentially longer) SPaths avoiding the bottleneck SLinks, hence, blocking ratio decreases for higher $\alpha$. Applying only differential delay constraints does not significantly impact the blocking ratio, which conforms with our previous finding on SSU. Unlike latency constraints, differential delay constraints do not reduce the size of the candidate SPath pool for a VLink, thus yields a blocking ratio similar to the baseline variant.

We also vary SN density (*i.e.*, SN LNR) for a fixed arrival rate of 10 and present the resulting blocking ratio for different variants in Fig. 5. For a sparse SN, *i.e.*, a ring SN, only 2 SPaths are available for mapping each VLink and candidate SPaths of different VLinks overlap with each other. Consequently, spectrum resources in SLinks exhaust faster, forcing all the variants to block similar percentages of VNs. With the increase in SN LNR and consequently spectrum resources up to a certain point, spectrum capacity no longer remains a bottleneck and blocking ratio becomes dominated by latency constraints, yielding a similar behavior as the one in Fig. 4. Finally, as the SN become denser, higher SPath diversity eliminates the impact of latency constraints, resulting in almost 100% VN requests to be accepted.

## VII. Conclusion

Motivated by the increasing demand for low-latency services, in this paper, we address the problem of VN embedding over EON with latency guarantees. We consider both fixed- and flex-grid EONs with flexible transmission parameters, namely baud rate, modulation format, and FEC overhead to efficiently allocate resources to VLinks. We present a novel approach that constrains latency over virtual paths, instead of bounding the latency on each virtual links, to better capture application latency requirements. We also identify latency contributions of various active/passive elements in an EON. To analyze the impact of latency constraints on resource allocation, we present an ILP formulation for embedding VNs that respect latency and differential delay constraints when using multiple SPaths to satisfy VLink demands. We also devise a heuristic algorithm to address the computational complexity of the ILP formulation. Our proposed heuristic performs close to the ILP formulation while executing several orders of magnitude faster. In static scenarios, our extensive simulations, based on realistic reaches, transmission configurations, and topologies, show that guaranteeing latency does not require significant additional resources. However, in a dynamic scenario, an additional 20% and 12% VN requests were blocked for fixed- and flex-grid EONs (Nobel Germany topology), respectively, when subject to latency constraints. We also show that by adding 15% SLinks to the unmodified Nobel Germany topology, we can bring the blocking ratio down to 10% and 1% for fixed- and flex-grid EONs, respectively.

## Acknowledgement

We thank our shepherd Stefan Schmid and the anonymous reviewers of IEEE ICNP 2019 for their valuable feedback. This work was funded in part by Huawei Canada and in part by an NSERC CRD Grant. This work benefited from the use of CrySP RIPPLE Facility at University of Waterloo.

## REFERENCES

[1] S. Zhang, R. Wang, U. Mandal, M. F. Habib, and B. Mukherjee, "Connecting the clouds with low-latency, low-cost virtual private lines enabled by sliceable optical networks," in *Proceedings of IEEE Global Communications Conference (GLOBECOM)*, 2013, pp. 2370–2375.

[2] Infinera, "Low latency – how low can you go?" White paper, 2016.

[3] M. A. Lema, A. Laya, T. Mahmoodi, M. Cuevas, J. Sachs, J. Markendahl, and M. Dohler, "Business case and technology analysis for 5G low latency applications," *IEEE Access*, vol. 5, pp. 5917–5935, 2017.

[4] I. Parvez, A. Rahmati, I. Guvenc, A. I. Sarwat, and H. Dai, "A survey on low latency towards 5G: Ran, core network and caching solutions," *IEEE Communications Surveys & Tutorials*, vol. 20, no. 4, pp. 3098–3130, 2018.

[5] "Information week – wall street's quest to process data at the speed of light," 2007. [Online]. Available: https://www.informationweek.com/wall-streets-quest-to-process-data-at-the-speed-of-light/d/d-id/1054287

[6] V. Reporter, "The value of a millisecond: Finding the optimal speed of a trading infrastructure," Vision note, April 2008.

[7] A. Technologies, "State of online retail performance – 2017 holiday retrospective," White paper, 2017.

[8] G. G. M. S. Association, "The road to 5G: Drivers, applications, requirements and technical development," White paper, 2015.

[9] Nokia, "5G use cases and requirements," White paper.

[10] Huawei, "5G: A technology vision," White paper.

[11] X. Foukas, G. Patounas, A. Elmokashfi, and M. K. Marina, "Network slicing in 5G: Survey and challenges," *IEEE Communications Magazine*, vol. 55, no. 5, pp. 94–100, 2017.

[12] N. M. K. Chowdhury and R. Boutaba, "A survey of network virtualization," *Computer Networks*, vol. 54, no. 5, pp. 862–876, 2010.

[13] A. Fischer, J. F. Botero, M. T. Beck, H. De Meer, and X. Hesselbach, "Virtual network embedding: A survey," *IEEE Communications Surveys & Tutorials*, vol. 15, no. 4, pp. 1888–1906, 2013.

[14] J. Inführ and G. R. Raidl, "Introducing the virtual network mapping problem with delay, routing and location constraints," in *Network optimization*. Springer, 2011, pp. 105–117.

[15] L. Shengquan, W. Chunming, Z. Min, and J. Ming, "An efficient virtual network embedding algorithm with delay constraints," in *Proceedings of IEEE International Symposium on Wireless Personal Multimedia Communications (WPMC)*, 2013, pp. 1–6.

[16] S. Ayoubi, C. Assi, K. Shaban, and L. Narayanan, "Minted: Multicast vIrtual NeTwork Embedding in cloud data centers with Delay constraints," *IEEE Transactions on Communications*, vol. 63, no. 4, pp. 1291–1305, 2015.

[17] F. Bianchi and F. Lo Presti, "A markov reward based resource-latency aware heuristic for the virtual network embedding problem," *ACM SIGMETRICS Performance Evaluation Review*, vol. 44, no. 4, pp. 57–68, 2017.

[18] K. Hejja and X. Hesselbach, "Online power aware coordinated virtual network embedding with 5G delay constraint," *Elsevier Journal of Network and Computer Applications*, vol. 124, pp. 121–136, 2018.

[19] B. Martini, F. Paganelli, P. Cappanera, S. Turchi, and P. Castoldi, "Latency-aware composition of virtual functions in 5G," in *Proceedings of IEEE Conference on Network Softwarization (NetSoft)*, April 2015, pp. 1–6.

[20] O. Gerstel, M. Jinno, A. Lord, and S. B. Yoo, "Elastic optical networking: A new dawn for the optical layer?" *IEEE Communications Magazine*, vol. 50, no. 2, 2012.

[21] R. Boutaba, N. Shahriar, and S. Fathi, "Elastic optical networking for 5G transport," *Springer Journal of Network and Systems Management*, vol. 25, no. 4, pp. 819–847, 2017.

[22] L. Liu, Y. Yin, M. Xia, M. Shirazipour, Z. Zhu, R. Proietti, Q. Xu, S. Dahlfort, and S. J. Ben Yoo, "Software-defined fragmentation-aware elastic optical networks enabled by openflow," in *Proceedings of European Conference and Exhibition on Optical Communication (ECOC 2013)*, 2013, pp. 1–3.

[23] N. Cvijetic, A. Tanaka, P. N. Ji, K. Sethuraman, S. Murakami, and T. Wang, "SDN and Openflow for dynamic flex-grid optical access and aggregation networks," *IEEE/OSA Journal of Lightwave Technology*, vol. 32, no. 4, pp. 864–870, 2013.

[24] L. Liu, R. Muñoz, R. Casellas, T. Tsuritani, R. Martínez, and I. Morita, "Openslice: An openflow-based control plane for spectrum sliced elastic optical path networks," *OSA Optics express*, vol. 21, no. 4, pp. 4194–4204, 2013.

[25] R. Casellas, R. Martínez, R. Muñoz, R. Vilalta, L. Liu, T. Tsuritani, and I. Morita, "Control and management of flexi-grid optical networks with an integrated stateful path computation element and openflow controller," *IEEE/OSA Journal of Optical Communications and Networking*, vol. 5, no. 10, pp. A57–A65, 2013.

[26] N. Shahriar, S. Taeb, S. R. Chowdhury, M. Tornatore, R. Boutaba, J. Mitra, and M. Hemmati, "Achieving a Fully-Flexible Virtual Network Embedding in Elastic Optical Networks," in *Proceedings of IEEE International Conference on Computer Communications (INFOCOM)*, 2019, pp. 1756–1764.

[27] L. Gong and Z. Zhu, "Virtual optical network embedding (VONE) over elastic optical networks," *IEEE/OSA Journal of Lightwave Technology*, vol. 32, no. 3, pp. 450–460, 2014.

[28] R. Lin, S. Luo, J. Zhou, S. Wang, A. Cai, W.-D. Zhong, and M. Zukerman, "Virtual network embedding with adaptive modulation in flexi-grid networks," *IEEE/OSA Journal of Lightwave Technology*, vol. 36, no. 17, pp. 3551–3563, 2017.

[29] S. Orlowski, R. Wessäly, M. Pióro, and A. Tomaszewski, "SNDlib 1.0—survivable network design library," *Networks: An International Journal*, vol. 55, no. 3, pp. 276–286, 2010. [Online]. Available: http://sndlib.zib.de/home.action

[30] B. C. Chatterjee, N. Sarma, and E. Oki, "Routing and spectrum allocation in elastic optical networks: A tutorial," *IEEE Communications Surveys & Tutorials*, vol. 17, no. 3, pp. 1776–1800, 2015.

[31] J. Zhao, S. Subramaniam, and M. Brandt-Pearce, "Virtual topology mapping in elastic optical networks," in *Proceedings of IEEE International Conference on Communications (ICC)*, 2013, pp. 3904–3908.

[32] M. Zhang, C. Wu, M. Jiang, and Q. Yang, "Mapping multicast service-oriented virtual networks with delay and delay variation constraints," in *Proceedings of IEEE Global Telecommunications Conference (GLOBECOM)*, 2010, pp. 1–5.

[33] G. Chochlidakis and V. Friderikos, "Low latency virtual network embedding for mobile networks," in *Proceedings of IEEE International Conference on Communications (ICC)*, 2016, pp. 1–6.

[34] G. Chochlidakis and V. Friderikos, "Mobility aware virtual network embedding," *IEEE Transactions on Mobile Computing*, vol. 16, no. 5, pp. 1343–1356, 2016.

[35] W. Mandarawi, A. Fischer, A. M. Houyou, H.-P. Huth, and H. De Meer, "Constraint-based virtualization of industrial networks," in *Principles of Performance and Reliability Modeling and Evaluation*. Springer, 2016, pp. 567–586.

[36] L. Qu, C. Assi, and K. Shaban, "Delay-aware scheduling and resource optimization with network function virtualization," *IEEE Transactions on Communications*, vol. 64, no. 9, pp. 3746–3758, 2016.

[37] H. A. Alameddine, L. Qu, and C. Assi, "Scheduling service function chains for ultra-low latency network services," in *Proceedings of IEEE/ACM/IFIP International Conference on Network and Service Management (CNSM)*, 2017, pp. 1–9.

[38] R. Cziva, C. Anagnostopoulos, and D. P. Pezaros, "Dynamic, latency-optimal VNF placement at the network edge," in *Proceedings of IEEE Conference on Computer Communications (INFOCOM)*, 2018, pp. 693–701.

[39] S. Fichera, R. Martinez, B. Martini, M. Gharbaoui, R. Casellas, R. Vilalta, R. Muñoz, and P. Castoldi, "Latency-aware network service orchestration over an SDN-controlled multi-layer transport infrastructure," in *Proceedings of International Conference on Transparent Optical Networks (ICTON)*, 2018, pp. 1–4.

[40] D. Harutyunyan, N. Shahriar, R. Boutaba, and R. Riggio, "Latency–aware service function chain placement in 5G mobile networks," in *Proceedings of IEEE Conference on Network Softwarization (NetSoft)*, 2019, pp. 133–141.

[41] M. D. M. Nguyen and M. Ghaderi, "Proactive service orchestration with deadline," in *Proceedings of IEEE Conference on Network Softwarization (NetSoft)*, 2019, pp. 369–377.

[42] H. Alameddine, M. H. K. Tushar, and C. Assi, "Scheduling of low latency services in softwarized networks," *IEEE Transactions on Cloud Computing*, March 2019.

[43] J. Jay, "Low signal latency in optical fiber networks," in *Corning Optical Fiber, Proc. of the 60th IWCS, Conference, Charlotte, NC, USA,(6-9 Nov. 2011)*. Citeseer, 2011.

[44] V. Bobrovs, S. Spolitis, and G. Ivanovs, "Latency causes and reduction in optical metro networks," in *Optical Metro Networks and Short-Haul*

*Systems VI*, vol. 9008. International Society for Optics and Photonics, 2014, p. 90080C.

[45] Optelian, "A sensible low-latency strategy for optical transport networks," White paper, 2014.

[46] Y. Pointurier, N. Benzaoui, W. Lautenschlaeger, and L. Dembeck, "End-to-end time-sensitive optical networking: Challenges and solutions," *IEEE/OSA Journal of Lightwave Technology*, vol. 37, no. 7, pp. 1732–1741, April 2019.

[47] N. Amaya, G. Zervas, and D. Simeonidou, "Introducing node architecture flexibility for elastic optical networks," *IEEE/OSA Journal of Optical Communications and Networking*, vol. 5, no. 6, pp. 593–608, 2013.

[48] G. Zhang, M. De Leenheer, A. Morea, and B. Mukherjee, "A survey on OFDM-based elastic core optical networking," *IEEE Communications Surveys & Tutorials*, vol. 15, no. 1, pp. 65–87, 2012.

[49] A. Pagès, J. Perelló, S. Spadaro, and J. Comellas, "Optimal route, spectrum, and modulation level assignment in split-spectrum-enabled dynamic elastic optical networks," *IEEE/OSA Journal of Optical Communications and Networking*, vol. 6, no. 2, pp. 114–126, 2014.

[50] G. Bernstein, D. Caviglia, R. Rabbat, and H. Van Helvoort, "Vcat-lcas in a clamshell," *IEEE Communications Magazine*, vol. 44, no. 5, pp. 34–36, 2006.

[51] Flex ethernet 2.0 implementation agreement [online] https://www.oiforum.com/wp-content/uploads/2019/01/oif-flexe-02.0-1.pdf.

[52] V. López and L. Velasco, *Elastic Optical Networks*. Architectures, Technologies, and Control, Switzerland: Springer Int. Publishing, 2016.

[53] T. Xu, G. Jacobsen, S. Popov, J. Li, E. Vanin, K. Wang, A. T. Friberg, and Y. Zhang, "Chromatic dispersion compensation in coherent transmission system using digital filters," *OSA Optics express*, vol. 18, no. 15, pp. 16 243–16 257, 2010.

[54] S. Huang, C. U. Martel, and B. Mukherjee, "Survivable multipath provisioning with differential delay constraint in telecom mesh networks," *IEEE/ACM Transactions On Networking*, vol. 19, no. 3, pp. 657–669, 2011.

[64] A. Blenk, P. Kalmbach, J. Zerwas, M. Jarschel, S. Schmid, and W. Kellerer, "Neurovine: A neural preprocessor for your virtual network embedding algorithm," in *Proceedings of IEEE International Conference on Computer Communications (INFOCOM)*, 2018, pp. 405–413.

[55] X. Chen, A. Jukan, and A. Gumaste, "Multipath de-fragmentation: Achieving better spectral efficiency in elastic optical path networks," in *Proceedings of IEEE International Conference on Computer Communications (INFOCOM)*, 2013, pp. 390–394.

[56] N. M. K. Chowdhury, M. R. Rahman, and R. Boutaba, "Virtual network embedding with coordinated node and link mapping," in *Proceedings of IEEE International Conference on Computer Communications (INFOCOM)*, 2009, pp. 783–791.

[57] Y. Wang, X. Cao, and Y. Pan, "A study of the routing and spectrum allocation in spectrum-sliced elastic optical path networks," in *INFOCOM, 2011 Proceedings IEEE*. IEEE, 2011, pp. 1503–1511.

[58] M. Rost and S. Schmid, "Charting the complexity landscape of virtual network embeddings," in *Proceedings of IFIP Networking Conference and Workshops*, 2018, pp. 1–9.

[59] M. Rost, E. Döhne, and S. Schmid, "Parametrized complexity of virtual network embeddings: Dynamic & linear programming approximations," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 49, no. 1, pp. 3–10, Feb. 2019.

[60] M. Yu, Y. Yi, J. Rexford, and M. Chiang, "Rethinking virtual network embedding: substrate support for path splitting and migration," *ACM SIGCOMM Computer Communication Review*, vol. 38, no. 2, pp. 17–29, 2008.

[61] M. Chowdhury, M. R. Rahman, and R. Boutaba, "Vineyard: Virtual network embedding algorithms with coordinated node and link mapping," *IEEE/ACM Transactions on Networking (TON)*, vol. 20, no. 1, pp. 206–219, 2012.

[62] C. Papagianni, A. Leivadeas, S. Papavassiliou, V. Maglaris, C. Cervello-Pastor, and A. Monje, "On the optimal allocation of virtual resources in cloud computing networks," *IEEE Transactions on Computers*, vol. 62, no. 6, pp. 1060–1071, 2013.

[63] S. R. Chowdhury, R. Ahmed, M. M. A. Khan, N. Shahriar, R. Boutaba, J. Mitra, and F. Zeng, "Protecting virtual networks with drone," *IEEE Transactions on Network and Service Management*, vol. 13, pp. 913–926, 2016.

[65] S. R. Chowdhury, S. Ayoubi, R. Ahmed, N. Shahriar, R. Boutaba, J. Mitra, and L. Liu, "Multi-layer virtual network embedding," *IEEE Transactions on Network and Service Management*, vol. 15, no. 3, pp. 1132–1145, Sept 2018.