MULE: Multi-Layer Virtual Network Embedding

Shihabur Rahman Chowdhury*, Sara Ayoubi*, Reaz Ahmed*, Nashid Shahriar*, Raouf Boutaba*,
Jeebak Mitra†, and Liu Liu‡
*David R. Cheriton School of Computer Science, University of Waterloo
{sr2chowdhury | sayoubi | r5ahmed | nshahria | rboutaba}@uwaterloo.ca
†Huawei Technologies Canada Research Center
jeebak.mitra@huawei.com
‡Huawei Technologies
liuliu1@huawei.com

***Abstract*—Network Virtualization (NV), considered as a key enabler for overcoming the ossification of the Internet allows multiple heterogeneous virtual networks to co-exist over the same substrate network. Resource allocation problems in NV have been extensively studied for single layer substrates such as IP or Optical networks. However, little effort has been put to address the same problem for multi-layer IP-over-Optical networks. The increasing popularity of multi-layer networks for deploying backbones combined with their unique characteristics (*e.g.*, topological flexibility of the IP layer) calls for the need to carefully investigate the resource provisioning problems arising from their virtualization. In this paper, we address the problem of MUlti-Layer virtual network Embedding (MULE) on IP-over-Optical networks. We propose two solutions to MULE: an Integer Linear Program (ILP) formulation for the optimal solution and a heuristic to address the computational complexity of the optimal solution. We demonstrate through extensive simulations that on average our heuristic performs within ≈1.47× of optimal solution and incurs ≈66% less cost than the state-of-the-art heuristic.**

I. Introduction

Multi-layer IP-over-Optical networks are becoming a popular choice among Infrastructure Providers (InPs) for deploying wide area networks [1]. Such multi-layer network typically consists of an optical substrate for the physical communication with an IP overlay on top [2]. This network model is being increasingly adopted for backbone networks as it offers the best of both worlds, *i.e.*, the flexibility in addressing, resource allocation, and traffic engineering of IP networks along with the high capacity provided by optical networks. Despite their increasing popularity, research on addressing resource provisioning challenges for virtualizing such networks is still in its infancy. A classical resource provisioning problem in network virtualization is Virtual Network Embedding (VNE), which consists in establishing a Virtual Network (VN) on a Substrate Network (SN) with objectives such as minimizing resource provisioning cost [3], [4], maximizing the number of admitted VNs [5], *etc*. VNE has been extensively studied for single-layer SNs [6] with significantly lesser attention paid to the multi-layer network substrates [7]. The topological flexibility provided by multi-layer networks [8] poses some unique challenges for VNE and calls for new investigations.

Similar to multi-layer networks, Mule, a hybrid species brings the best of two species together.

Several deployment models exist for multi-layer IP-over-Optical networks [9] including but not limited to: (i) IP over Dense Wavelength Division Multiplexed (DWDM); (ii) IP over Optical Transport Network (OTN) over DWDM. DWDM networks have specific constraints such as wavelength continuity for optical circuits and typically do not have transparent traffic grooming capabilities. A more favorable choice (also our choice of technology) is to deploy an OTN [10] over a DWDM network with advanced transport capabilities (*e.g.*, traffic grooming without optical-electrical-optical conversion). The OTN in turn can be *static*, *i.e.*, necessary interfaces on OTN nodes have been configured and the corresponding light paths in the DWDM layer have been lit to provision fixed bandwidth between OTN nodes. Or, the OTN can be *dynamic*, *i.e.*, more bandwidth between OTN nodes can be provisioned by lighting new light paths in the DWDM. Clearly, the VNE problem for each of these scenarios requires dedicated explorations due to their unique constraints. As a first step towards addressing VNE for multi-layer networks, we limit the scope of this paper to the case of a *static OTN* and leave the other possible deployment scenarios for future investigation.

Solving the VNE problem for multi-layer networks exhibits many unique challenges due to the topological flexibility offered by such networks. Concretely, although the OTN is fixed, the IP network is dynamic, *i.e.*, new IP links can be established when needed by provisioning necessary capacity from the OTN. Such flexibility can be exploited if residual resources in the IP layer are insufficient to admit a new VN, or to reduce the cost of VN embedding by creating new IP links that reduce network diameter. Provisioning new IP links in optical networks has been a tedious and manual task with a long turnaround time. However, with the advances in optical networking technologies [11] and centralized optical control plane [12]–[15], such provisioning tasks are more and more automated. Even then, one should not abuse such capability to sporadically establish new IP links since it remains more expensive than embedding virtual links on existing IP links. In this regard, we are faced with the following challenges: (i) strike a balance between obtaining a low cost VN embedding while minimizing the establishment of new IP links; (ii) simultaneously decide on whether to create an IP link or not and its embedding in the OTN.

978-3-901882-98-2

In this paper, we study the problem of **MU**lti-**L**ayer Virtual Network **E**mbedding (*MULE*) focusing on IP-over-OTN substrate networks with the objective of minimizing total resource provisioning cost for embedding the VN while considering the possibility of establishing new IP links when necessary. Specifically, the contributions of this paper are as follows:

- *OPT-MULE*: An Integer Linear Program (ILP) formulation to find the optimal solution to *MULE*. The state-of-the-art in multi-layer VNE [7] does not optimally solve the problem. To the best of our knowledge, this is the first optimal solution to *MULE*.
- *FAST-MULE*: A heuristic to tackle the computational complexity of *OPT-MULE*. We also prove that our heuristic solves the problem optimally for a specific class of VNs, *i.e.*, star-shaped VNs. Further, we evaluate our heuristic and compare it against the optimal solution and with the state-of-the-art solution in the literature [7].

The rest of the paper is organized as follows. We begin with a discussion of related works in Section II. Then we introduce our model and formally define the problem in Section III. In Section IV, we present *OPT-MULE*, an ILP formulation to optimally solve *MULE*, followed by our proposed heuristic, *FAST-MULE* in Section V. Our evaluation of the proposed solutions are presented in Section VI. Finally, we conclude with some future research directions in Section VII.

II. Related Works

VNE is a well studied problem in network virtualization and a significant body of research has solved a number of its variants [4], [16]–[24]. However, it has been mostly studied for single layer SNs, *i.e.*, for IP, Optical or Wireless networks. Despite the existence of a significant number of proposals [25]–[27], VNE solutions for IP networks commonly involve allocating compute and bandwidth resources for the virtual nodes and links, respectively. In the case of optical networks, solving VNE involves allocating compute resources and wavelength for virtual nodes and links, respectively [28]. Optical networks have technological constraints such as discrete wavelength allocation, wavelength continuity *etc*. that add additional challenges to the VNE problem [29]. The state-of-the-art in optical network virtualization has mostly focused on single layer optical networks.

More recently, Zhang *et al.*, proposed a heuristic for solving the multi-layer VNE problem for IP-over-DWDM networks [7]. They also consider the possibility of modifying IP layer topology by allocating wavelengths from the underlying DWDM network. Zhang *et al.*, proposed a two step embedding process that first embeds the virtual nodes then the virtual links, which limits the solution space and hence the optimality of the embedding. In contrast, we propose an ILP formulation for optimally solving the multi-layer VNE problem. Also, our heuristic does not embed the virtual nodes and links independently from each other, rather jointly embeds them.

An orthogonal but somehow related area of research in multi-layer network optimization focused on the issue of capacity planning in multi-layer networks [29], [30]. During the initial capacity planning a traffic matrix for the IP layer is given and sufficient capacity needs to be allocated in both IP and Optical layers to support that traffic matrix. Different variants of the problem exist that take different technological constraints, deployment models, and failure scenarios into account [31]–[38]. In contrast, the endpoint of the demands, *i.e.*, virtual node placement, is not known in advance in multi-layer VNE, making this one a fundamentally different problem from multi-layer capacity planning. Having said that the substantial body of research in multi-layer capacity planning has demonstrated clear advantages of resource allocation when the layers are jointly optimized as opposed to considering them in isolation [31], [39]. Our solution approach also takes a joint optimization approach to the multi-layer VNE problem.

III. MULE: Multi-Layer Virtual Network Embedding Problem

We first present a mathematical representation of the inputs, *i.e.*, the IP topology, the OTN topology, and the VN request. Then we formally define *MULE*.

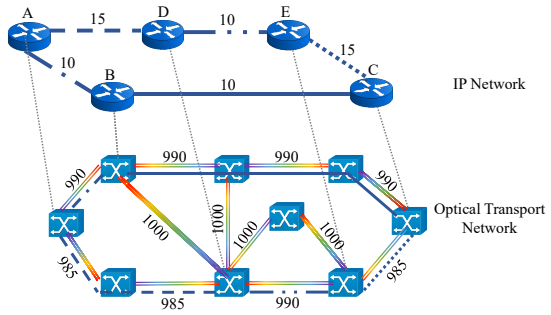


Fig. 1. Multi-Layer Infrastructure

A. Substrate Optical Transport Network (OTN)

We represent the substrate OTN as an undirected graph $\hat{G} = (\hat{V}, \hat{E})$, where $\hat{V}$ and $\hat{E}$ are the set of Optical Cross Connects (OXCs) (referred as OTN nodes in the remaining) and OTN links, respectively (similar to [33]. Neighbors of an OTN node $\hat{u} \in \hat{V}$ are represented with $\mathcal{N}(\hat{u})$. We assume the OTN to be fixed, *i.e.*, light paths atop a DWDM layer have been already lit to provision OTN links $(\hat{u}, \hat{v}) \in \hat{E}$ with bandwidth capacity $b_{\hat{u}\hat{v}}$. This pre-provisioned bandwidth can be used to establish IP links between IP routers. The cost of allocating one unit of bandwidth from an OTN link $(\hat{u}, \hat{v}) \in \hat{E}$ is $C_{\hat{u}\hat{v}}$. Fig. 1 illustrates an example of an OTN network, where the numbers on each link represent its residual capacity.

B. Substrate IP Network

The substrate IP network is an undirected graph $G' = (V', E')$. Each IP node $u' \in V'$ has $p_{u'}$ number of ports with homogeneous capacity $cap_{u'}$. An IP node is connected to an OTN node through a short-reach wavelength interface. Attachment between an IP and an OTN node is represented using a binary input variable $\tau_{u'\hat{u}}$, which is set to 1 only when IP node u' is attached to OTN node $\hat{u}$. An IP link is provisioned by establishing an OTN path that connects

its end points. Note that, it is common in operator networks to establish multiple IP links between the same pair of IP nodes and bundle their capacities using some form of link aggregation protocol [40]. We also follow the same practice and use (u', v', i) to represent the i-th IP link between u' and v', where $1 \leq i \leq p_{u'}$. We set the binary input variable $\Gamma_{u'v'i}$ to 1 when IP link (u', v', i) is present in G', 0 otherwise. Bandwidth of an IP link is represented by $b_{u'v'i}$. Capacity of a new IP link (u', v', i) is set to $\min(cap_{u'}, cap_{v'})$. Fig. 1 illustrates an example IP network, where each IP link is mapped on an OTN path and the residual bandwidth capacity of an IP link is represented by the number on that link. The cost of allocating one unit of bandwidth from an IP link $(u', v', i) \in E'$ is $C_{u',v',i}$.

C. Virtual Network (VN)

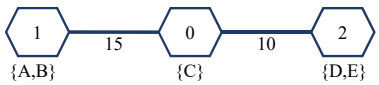


Fig. 2. Virtual Network

A VN request is 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. We assume online VN arrival, *i.e.*, VNs arriving one at a time (similar to [5]). Each VLink $(\bar{u}, \bar{v}) \in \bar{E}$ has a bandwidth requirement $b_{\bar{u}\bar{v}}$. Each VNode $\bar{u} \in \bar{V}$ has a location constraint set $\mathcal{L}(\bar{u}) \subset V'$ that represents the set of IP nodes where $\bar{u}$ can be embedded. We represent the location constraints using a binary input variable $\ell_{\bar{u}u'}$, which is set to 1 if IP node $u' \in \mathcal{L}(\bar{u})$. Fig. 2 illustrates a VN, where the number on each link represents VLink demand, and the set next to each node denotes that VNode's location constraints.

D. Problem Definition

Given a multi-layer SN composed of an IP network G' on top of an OTN network $\hat{G}$, and a VN request $\bar{G}$ with location constraint set $\mathcal{L}$:

- Map each VNode $\bar{u} \in \bar{V}$ to an IP node $u' \in V'$ according to the VNode's location constraint.
- Map each VLink $(\bar{u}, \bar{v}) \in \bar{E}$ to a path in the IP network. This path can contain a combination of existing IP links and newly created IP links.
- Map all newly created IP links to a path in the OTN.
- The total cost of provisioning resources for new IP links and cost of provisioning resources for VLinks should be minimized subject to the following constraints:
  - IP links cannot be over-committed to accommodate the VLinks, and
  - the demand of a single VLink should be satisfied by a single IP path.

IV. OPT-*MULE*: An ILP Formulation

A. Decision Variables

A VLink must be mapped to a path in the IP network. The following decision variable indicates the mapping between a VLink $(\bar{u}, \bar{v}) \in \bar{E}$ and an IP link, $(u', v', i) \in E'$.

$$x^{\bar{u}\bar{v}}_{u'v'i} = \begin{cases} 1 & \text{if } (\bar{u}, \bar{v}) \in \bar{E} \text{ is mapped to } (u', v', i) \in E', \\ 0 & \text{otherwise.} \end{cases}$$

The following decision variable denotes VNode mapping:

$$y_{\bar{u}u'} = \begin{cases} 1 & \text{if } \bar{u} \in \bar{V} \text{ is mapped to } u' \in V', \\ 0 & \text{otherwise.} \end{cases}$$

Creation of new IP links is decided by:

$$\gamma_{u'v'i} = \begin{cases} 1 & \text{when } i\text{-th IP link is created between } u' \text{ and } v', \\ 0 & \text{otherwise.} \end{cases}$$

Finally, a newly created IP link must be mapped to an OTN path. This mapping is indicated by the following variable:

$$z^{u'v'i}_{\hat{u}\hat{v}} = \begin{cases} 1 & \text{if } (u', v', i) \in E' \text{ is mapped to } (\hat{u}, \hat{v}) \in \hat{E}, \\ 0 & \text{otherwise.} \end{cases}$$

In what follows, V'^2 denotes the set of all pairs of IP nodes (u', v') such that $u' \neq v'$.

B. Constraints

1) VNode Mapping Constraint: (1) and (2) ensure that each VNode is mapped to exactly one IP node according to the location constraints. (3) restricts multiple VNodes to be mapped on the same IP Node.

$$\forall \bar{u} \in \bar{V}, \forall u' \in V' : y_{\bar{u}u'} \leq \ell_{\bar{u}u'} \tag{1}$$

$$\forall \bar{u} \in \bar{V} : \sum_{u' \in V'} y_{\bar{u}u'} = 1 \tag{2}$$

$$\forall u' \in V' : \sum_{\bar{u} \in \bar{V}} y_{\bar{u}u'} \leq 1 \tag{3}$$

2) VLink Mapping Constraints: (4) ensures that VLinks are mapped only to existing or newly created IP links. (5) ensures that each VLink is mapped to a non-empty subset of IP links. We prevent the formation of loops between parallel IP links by (6). (7) prevents overcommitment of IP link bandwidth. Finally, (8), our flow-conservation constraint, ensures that VLinks are mapped on a continuous IP path.

$$\forall (\bar{u}, \bar{v}) \in \bar{E}, \forall (u', v') \in V'^2, 1 \leq i \leq min(p_{u'}, p_{v'}) : x^{\bar{u}\bar{v}}_{u'v'i} \leq \gamma_{u'v'i} + \gamma_{v'u'i} + \Gamma_{u'v'i} \tag{4}$$

$$\forall (\bar{u}, \bar{v}) \in \bar{E} : \sum_{\forall (u', v') \in V'^2} \sum_{i=1}^{p_{u'}} x^{\bar{u}\bar{v}}_{u'v'i} \geq 1 \tag{5}$$

$$\forall (\bar{u}, \bar{v}) \in \bar{E}, \forall (u', v') \in V'^2 : \sum_{i=1}^{p_{u'}} x^{\bar{u}\bar{v}}_{u'v'i} \leq 1 \tag{6}$$

$$\forall (u', v') \in V'^2, 1 \leq i \leq p_{u'} : \sum_{\forall (\bar{u}, \bar{v}) \in \bar{E}} x^{\bar{u}\bar{v}}_{u'v'i} \times b_{\bar{u}\bar{v}} \leq b_{u'v'i} \tag{7}$$

$$\forall (\bar{u}, \bar{v}) \in \bar{E}, \forall u' \in V' : \sum_{\forall v' \in V'^2} \sum_{i=1}^{\min(p_{u'}, p_{v'})} (x^{\bar{u}\bar{v}}_{u'v'i} - x^{\bar{u}\bar{v}}_{v'u'i}) = y_{\bar{u}u'} - y_{\bar{v}u'} \tag{8}$$

3) IP Link Creation Constraints: (9) limits the number of incident IP links on an IP node to be within its available number of ports. Then, (10) ensures that a specific instance of IP link between a pair of IP nodes is either decided by the ILP or was part of the input, but not both at the same time.

$$\forall u' \in V' : \sum_{\forall v' \in V' | v' \neq u'} \sum_{i=1}^{\min(p_{u'}, p_{v'})} \gamma_{u'v'i} + \gamma_{v'u'i} + \Gamma_{u'v'i} \leq p_{u'} \tag{9}$$

$$\forall (u', v') \in V'^2, 1 \leq i \leq p_{u'} : \gamma_{u'v'i} + \Gamma_{u'v'i} \leq 1 \tag{10}$$

4) IP-to-OTN Link Mapping Constraints: First, we ensure, using (11), that only the newly created IP links are mapped on the OTN layer. Then, (12) is the flow conservation constraint that ensures continuity of the mapped OTN paths. Finally, (13) is our capacity constraint for OTN links.

$$\forall (u', v') \in V'^2, 1 \leq i \leq p_{u'}, (\hat{u}, \hat{v}) \in \hat{E} : z_{\hat{u}\hat{v}}^{u'v'i} \leq \gamma_{u'v'i} \tag{11}$$

$$\forall (u', v') \in V'^2, 1 \leq i \leq p_{u'}, \forall \hat{u} \in \hat{V} : \sum_{\forall \hat{v} \in \mathcal{N}(\hat{u})} (z_{\hat{u}\hat{v}}^{u'v'i} - z_{\hat{v}\hat{u}}^{u'v'i}) = \begin{cases} \gamma_{u'v'i} & \text{if } \tau_{u'\hat{u}} = 1, \\ -\gamma_{u'v'i} & \text{if } \tau_{v'\hat{u}} = 1, \\ 0 & \text{otherwise.} \end{cases} \tag{12}$$

$$\forall (\hat{u}, \hat{v}) \in \hat{E} : \sum_{\forall (u', v') \in V'^2} \sum_{i=1}^{p_{u'}} z_{\hat{u}\hat{v}}^{u'v'i} \times b_{u'v'i} \leq b_{\hat{u}\hat{v}} \tag{13}$$

C. Objective Function

Our objective is to minimize the cost incurred by creating new IP links and also the cost of provisioning bandwidth for the VLinks. Cost for provisioning new IP links is computed as the cost of allocating bandwidth in the OTN paths for every new IP link. The cost of embedding a VN is computed as the total cost of provisioning bandwidth on the IP links for the VLinks. Our objective function is formulated as follows:

$$\begin{aligned} \text{minimize} \sum_{\forall (u', v') \in V'^2} \sum_{i=1}^{p_{u'}} \sum_{\forall (\hat{u}, \hat{v}) \in \hat{E}} z_{\hat{u}\hat{v}}^{u'v'i} \times b_{u'v'i} \times C_{\hat{u}\hat{v}} \\ + \sum_{\forall (\bar{u}, \bar{v}) \in \bar{E}} \sum_{\forall (u', v') \in V'^2} \sum_{i=1}^{p_{u'}} x_{u'v'i'}^{\bar{u}\bar{v}} \times b_{\bar{u}\bar{v}} \times C_{u'v'i} \end{aligned} \tag{14}$$

D. Hardness of OPT-MULE

Consider the case where the IP layer has sufficient capacity to accommodate a given VN request. In this case, MULE becomes a single-layer VNE, which has been proven to be NP-Hard via a reduction from the multi-way separator problem [5]. Given that single-layer VNE is an instance of MULE, by restriction we conclude that MULE is also NP-Hard.

V. *FAST-MULE*: A Heuristic Approach

Given the NP-Hard nature of the multi-layer VNE problem and its intractability for large network instances, we propose FAST-*MULE*, a heuristic to solve the Multi-Layer VNE problem. We begin by explaining the challenges behind the design of FAST-*MULE* in Section V-A, followed by a description of its procedural details and an illustrative example in Section V-B and Section V-C, respectively. Finally, we prove in Section V-D that *FAST-MULE* yields the optimal solution for star VN topologies with uniform bandwidth requirement.

A. Challenges

1) Joint Mapping in IP and OTN Layers: One challenge of *MULE* is the fact that the embedding can take place in both layers. This occurs when a VN could not be accommodated by the existing IP links, and requires the creation of new ones. A plausible approach is to handle the embedding at each layer separately, *i.e.*, start by mapping the VN on the IP layer followed by mapping the new IP links on the OTN layer. Clearly, such disjoint embedding is far from the optimal as there may not be sufficient bandwidth at the OTN level to accommodate the new IP links. To overcome this limitation, we equip *FAST-MULE* with the ability to consider both layers simultaneously when embedding a VN. This is achieved by collapsing the IP and OTN into a single layer graph, similar to [7]. Our collapsed graph contains all the IP and OTN nodes and links, as well as the links connecting IP nodes to OTN nodes. In contrast, [7] keeps the IP links and replaces the shortest paths in OTN with potential IP links that could be created with those paths. In our case, a VLink embedding containing OTN links indicates creation of new IP links.

2) Joint VNode and VLink Embedding: Another challenge is to perform simultaneous embedding of a VNode and its incident VLinks. Embedding VNodes independently of their incident VLinks increases the chances of VN embedding failure. However, such joint embedding is hard to solve since it is equivalent to solving the NP-hard *Multi-commodity Unsplittable Flow with Unknown Sources and Destinations* [41]. Our goal is to equip FAST-*MULE* with the ability to perform joint embedding of VNodes along with their incident VLinks. To achieve this, we augment the collapsed graph with meta-nodes and modify its link capacities to convert the VNode and VLink embedding problem into a min-cost max-flow problem that we solve using Edmonds-Karp (EK) algorithm [42]. The flows returned by EK indicate both the VNodes and VLinks mapping. In what follows, we elucidate the details of this transformation along with how the embedding solution is extracted from the flows obtained from EK.

B. Heuristic Algorithm

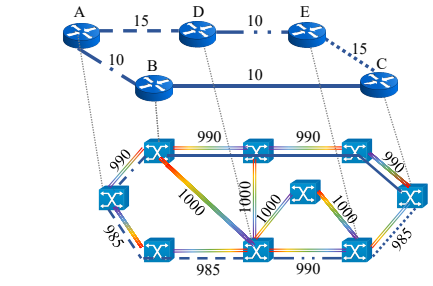
Alg. 1 presents a high level view of FAST-*MULE*. The details of every stage are as follows:

Stage 1: Creation of a Collapsed Graph: We begin by collapsing the OTN and IP networks to a single-layer to achieve joint mapping at the IP and OTN layers. We keep the residual capacities of the IP and OTN links as they are. We

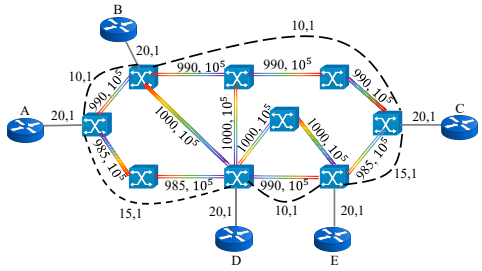
Algorithm 1: Multi-Layer VNE Algorithm

Input: $\hat{G} = (\hat{V},\hat{E})$, $G' = (V',E')$, $\bar{G} = (\bar{V},\bar{E})$
Output: Overlay Mapping Solution $\mathcal{M}$

function FAST-*MULE*()
  */*Initialize List of Settled Nodes*/*
  $\mathcal{S} = \{\}$
  Step 1: Create Collapsed Graph
  G = *CreateCollapsedGraph*(G',$\hat{G}$)
  forall $\bar{v} \in \bar{V}$ **do**
    if $\bar{v} \in \mathcal{S}$ **then**
      continue
    $\mathcal{S} = \mathcal{S} \cup \bar{v}$
    Step 2: Create Meta-Nodes
    $\mathcal{M}$.nmap = $\mathcal{M}$.nmap $\cup$ *MapNode*($\bar{v}$,$\mathcal{L}(\bar{v})$)
    for ***each*** $(\bar{u} \in \mathcal{N}(\bar{v}))$ **do**
      if $(\bar{u}\ in\ \mathcal{S})$ **then**
        continue
      if ($\mathcal{M}$.*nmap*($\bar{u}$) == *NULL*) **then**
        $V = V \cup$ *CreateMetaNodes*($\mathcal{L}(\bar{u})$)
      else
        $V = V \cup$ *CreateMetaNodes*($\mathcal{M}$.*nmap*($\bar{u}$))
    Step 3: Create Ref-Nodes
    $V = V \cup$ *CreateRefNodes*(V)
    Step 4: Run Link Embedding Algorithm
    $\mathcal{M}$.emap = $\mathcal{M}$.emap $\cup$ *EdmondsKarp*(G)
    $E = E \cup$ *GetNewIPLinks*($\mathcal{M}$.emap)
    $S = S \cup$ *isSettled*($\mathcal{N}(\bar{v})$)
  Return $\mathcal{M}$;



(a) IP-over-OTN Graph



(b) Collapsed IP-OTN Graph

Fig. 3. Transformation from multi-layer to single-layer substrate network

assume the OTN links have significantly higher cost than the IP links. Therefore, new IP links are created only when they are really needed and can significantly reduce embedding cost. Between every IP node u' and its corresponding OTN node $\hat{u}$, we create $p_{u'}$ links with capacity $cap_{u'}$. This guarantees that at most $p_{u'}$ new IP links can be created from u', and that the capacity of these IP links cannot exceed the ports capacity.

Stage 2: Extraction of Star-shaped Sub-graphs from VN: Next, we randomly pick a VNode $\bar{v} \in \bar{V}$ and embed $\bar{v}$ with its incident VLinks. Embedding $\bar{v}$'s incident links entails embedding its neighbors as well. This means that we are embedding a star-shaped subgraph of the VN at each iteration. To achieve this, we begin by mapping our current VNode $\bar{v}$, *i.e.*, the center of the star to a random IP node in its location constraint set (denoted as *source* in the following). Then we construct a flow network in such a way that the paths contributing to a min-cost max-flow in the flow network correspond to the embedding of the VLinks incident to $\bar{v}$.

Stage 3: Addition of Meta-Nodes: We create a flow network by replacing every link in the collapsed graph with directional links in both directions. Then, $\forall \bar{u} \in \mathcal{N}(\bar{v})$, we add a meta-node in the flow network that we connect to every node in $\mathcal{L}(\bar{u})$. These meta-nodes are in-turn connected to a single meta-node, that we denote as the *sink*. After adding the meta-nodes we set the link capacities as follows:

- We set the flow capacity of a link (u, v) from the collapsed graph that is not connected with any meta-node to $\frac{b_{uv}}{\max_{\forall \bar{u} \in \mathcal{N}(\bar{v})}(b_{\bar{u}\bar{v}})}$. Setting such capacity puts an upper limit on the maximum number of VLinks that can be routed through these links. Although this can lead to resource fragmentation and in the worst case rejection of a VN, it ensures that no capacity constraints are violated.
- We set the capacity of the links incident to a meta-node to 1. This guarantees that at most $|\mathcal{N}(\bar{v})|$ flows can be pushed from *source* to *sink*.

Stage 4: Addition of Referee Nodes: Location constraint sets of different VNodes in a single VN may overlap. We denote such VNodes as "conflicting nodes" and the intersection of their location constraint sets as the "conflict set". Every node in the conflict set is denoted as a conflict node. When conflicting VNodes are incident to the same *start* node, we end up with an augmented graph where all the nodes in the conflict set are connected to more than one meta-node. This is problematic because EK may end up routing multiple VLinks via the same conflict node, thereby violating the one-to-one node placement constraint. To resolve this issue, we introduce "Referee Nodes" (Ref-Nodes). Ref-Nodes are meta-nodes that are added to resolve the case of conflicting VNodes. In presence of a conflict, conflict nodes will be connected to more than one meta-node at the same time. Ref-Nodes are thus introduced to break this concurrency by removing the conflicting connections, and replacing them with a single connection to a Ref-node. The Ref-node is subsequently connected to all the meta-nodes of the conflicting nodes. This ensures that at most a single VLink will be routed through

any conflict node. Further, when a conflict node is selected to host a given VNode, no other IP nodes for the same VNode will be selected, thereby ensuring an one-to-one assignment.

Stage 5: Execution of the Edmonds-Karp Algorithm: Now we have an instance of the max-flow problem that we will solve using EK. We have set the capacity of the links in the flow network in such a way that EK can push at most $|\mathcal{N}(\bar{v})|$ flows, indicating the VLink embedding of $\bar{v}$'s incident links. Note that the only way to push $|\mathcal{N}(\bar{v})|$ flows is by having each flow traverse a unique meta-node to reach the *sink*. The VNode embedding of $\bar{v}$'s neighbors can be extracted by examining each flow to find the incident IP node of each meta-node. If any of the obtained flows is routed via an OTN path, then a new IP link is established and added to the collapsed graph. This allows subsequent iterations to use the newly created IP link. If at any iteration EK returns less than $|\mathcal{N}(\bar{v})|$ flows, this indicates an embedding failure, and the algorithm terminates. Otherwise, the algorithm returns to Stage 2 and repeats until all the VNodes are settled.

Let I be the number of iterations of Fast-*MULE*. During each iteration we run the EK algorithm to find min-cost max-flow. We replaced the augmenting path finding procedure of EK with Dijkstra's shortest path algorithm. Therefore, the running time of EK becomes $O(|V||E|^2 \log |V|)$. This renders the time complexity of our proposed approach to $O(I|V||E|^2 \log |V|)$. If we consider the worst-case scenario where the VN is in the form of a chain, and the nodes are traversed sequentially, then $I = |\bar{V}| - 1$, which results in a worst-case complexity of $O(|\bar{V}||V||E|^2 \log |V|)$. Note that, $|V|$ and $|E|$ represent the number of nodes and links in the collapsed graph, respectively, where $|V| = O(|\hat{V}| + |V'|)$, $|E| = O(|\hat{E}| + |E'|)$.

C. Illustrative Example

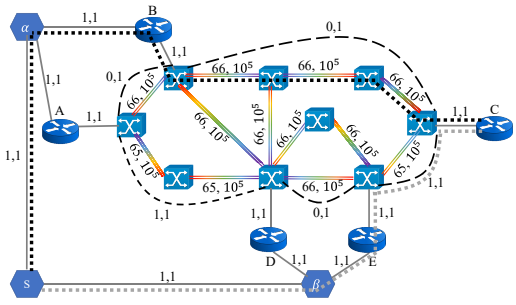


Fig. 4. Illustrative Example

Fig. 3(b) illustrates how the IP-over-OTN graph in Fig. 3(a) has been converted into a collapsed IP-OTN graph. The collapsed graph is composed of the OTN nodes, OTN links, IP Nodes, IP-OTN links (represented by the single straight grey lines), and IP links (represented by the dashed black lines). The latter are represented by direct links between the OTN nodes incident to the endpoints of each IP link. Here, we assume that each IP node has a single residual port of capacity 20. The numbers on each link represent the capacity of the link followed by the cost of using this link. Observe that we set the cost of the IP links to 1, whereas the cost of the OTN links is set to a really high number to discourage the routing from passing through OTN links.

Next, we showcase how *FAST-MULE* embeds the VN in Fig. 2 atop the collapsed graph, as illustrated in Fig. 4. We consider that VNode 0 is the start node. Hence, the source node at this iteration of EK is IP node C. The sink node is meta-node s attached to the meta-nodes α and β of VNodes 1 and 2, respectively. Given that the maximum demand of VNode 0's incident links is 15, the capacity of each link in the collapsed graph (except links incident to meta-nodes whose capacity is fixed to 1) is replaced by the number of VLinks of capacity 15 it can accommodate. Running EK on the augmented graph (Stage 5) returns two flows between the source node C and the sink node s, indicated by the black and grey dotted lines in Fig. 4. Here, we observe that EK can only route VLink (0,2) via existing IP links (grey flow); whereas VLink (0,1) is routed through OTN links (black flow), thereby creating a new IP link (B,C) with capacity 20. Further, by examining the terminating IP nodes in every flow, we identify the VNode embedding of nodes 1 and 2 as IP nodes B and E, respectively.

D. Optimality of FAST-MULE for Star VN Topology

Recall that in Alg. 1, the joint node and link embeddings are executed iteratively on a subgraph of the VN until all the VNodes are settled. This iterative scheme renders a sub-optimal solution. However, if we could perform a joint node and link embedding on the entirety of the VN in a single iteration, that would guarantee that the obtained solution is indeed optimal. Such embedding is possible when all the nodes in the VN are only connected to a single node, and if the latter is selected as the start node, *i.e.*, the VN topology is a star. A star VN topology $S(N)$ contains a center node $\bar{u}$ and N links connecting $\bar{u}$ to N leaf nodes $\{\bar{v}_1, \bar{v}_2, \dots \bar{v}_N\}$. In the sequel, we prove that Alg. 1 can find the optimal solution in polynomial time when the VN request is a star topology (typically used to support multi-cast services [4]) with identical bandwidth demand on all VLinks.

Theorem 1. *Given a star VN topology $\bar{G} = S(N)$ with uniform bandwidth demand β for all VLinks, Alg. 1 obtains the optimal solution in polynomial time.*

Proof. The optimal embedding of $\bar{G}$, $\mathcal{M}^*$, is the one where the VNodes are placed on the IP nodes that provide the lowest cost link embedding. The cost includes both the cost of provisioning new IP links and the cost of allocating bandwidth for VLinks. We denote the cost of $\mathcal{M}^*$ as $\theta^* = \beta \sum_{i=1}^{N} \sum_{u'v' \in P_{\bar{u}\bar{v}_i}} C_{u'v'}$, where $P_{\bar{u}\bar{v}_i}$ is the embedding path for VLink $(\bar{u}, \bar{v}_i)$. Without loss of generality, we abstract a newly created IP link (u', v')'s cost as $C_{u'v'}$. Let $\mathcal{M}$ be the solution obtained by Alg. 1. For simplicity, we assume the central node $\bar{u}$ has exactly one IP node in its location constraint set. $\mathcal{M}$ consists of placing $\bar{u}$ on the IP node in its location constraint set, v', followed by running EK from

v' to the sink node s. EK will return the min-cost max-flow from v' to the sink node s. Given that the capacity of all the incident links to s are 1, the number of flow augmenting paths will be at most the number of leaf nodes in $\bar{G}$ and exactly 1 unit of flow will be pushed through each of these augmenting paths. Therefore, upon successful embedding, EK will return N flow augmenting paths with minimum cost θ. Now recall that the only way to push N flows towards the sink is to traverse every meta-node once; which entails the traversal of one node from each location constraint set. The traversed nodes represent the VNode embedding of all the leaf nodes in $S(N)$. Therefore, the flow augmenting paths represent a valid embedding of $S(N)$. We can characterize θ as, $\theta = \sum_{i=1}^{N} \sum_{(u,v)\in F_i} C_{uv} \times f_{uv}$, where F_i is the i-th flow augmenting path and f_{uv} is the flow pushed along link (u, v) in the flow network constructed from the collapsed graph. Note that, $f_{uv} = 1$, therefore, the cost becomes, $\theta = \sum_{i=1}^{N} \sum_{(u,v)\in F_i} C_{uv}$. If we can prove that $\sum_{i=1}^{N} \sum_{(u,v)\in F_i} C_{uv} = \sum_{i=1}^{N} \sum_{u'v'\in P_{\bar{u}\bar{v}_i}} C_{u'v'}$ then our proof is complete. Since θ^* is the optimal objective value, let, $\sum_{i=1}^{N} \sum_{(u,v)\in F_i} C_{uv} > \sum_{i=1}^{N} \sum_{u'v'\in P_{\bar{u}\bar{v}_i}} C_{u'v'}$. Then it implies that if we pushed the flows along the paths $\bigcup_{i=1}^{N} P_{\bar{u}\bar{v}_i}$ (the newly created IP links can be expanded to a set of OTN links to match the paths in the collapsed graph), we would have obtained a lower cost solution to min-cost max-flow problem, which contradicts that θ is the minimum cost of our min-cost max-flow problem for the converted flow network. Therefore, $\sum_{i=1}^{N} \sum_{(u,v)\in F_i} C_{uv} = \sum_{i=1}^{N} \sum_{u'v'\in P_{\bar{u}\bar{v}_i}} C_{u'v'}$, completing our proof. □

If the central node, $\bar{u}$, has more than one candidate node in its location constraint set, then running Alg. 1 $|\mathcal{L}(\bar{u})|$ times is sufficient to obtain the lowest cost mapping solution, and the running time of Alg. 1 still remains polynomial.

VI. Evaluation Results

We evaluate our proposed solutions for *MULE* through simulations. Due to the lack of publicly available multi-layer network topologies, we generate synthetic topologies with varying sizes for our performance evaluation. We first describe our simulation setup in Section VI-A and the evaluation metrics in Section VI-B. Our evaluation is performed based on the following scenarios: (i) cost comparison between *FAST-MULE* and *OPT-MULE* to evaluate how well *FAST-MULE* compares to the optimal, and (ii) comparison of *FAST-MULE* with the state-of-the-art heuristic [7] for solving multi-layer VNE problem.

A. Simulation Setup

1) Testbed: We have implemented *OPT-MULE* and *FAST-MULE* using IBM ILOG CPLEX 12.5 C++ libraries and Java, respectively. *OPT-MULE* was run on a machine with 4×8 core 2.4Ghz Intel Xeon E5-4640 CPU and 512GB of memory, whereas, we used a machine with 2×8 core 2Ghz Intel Xeon E5-2650 CPU and 256GB memory to evaluate *FAST-MULE*.

2) Network Topologies: As previously mentioned, we synthetically generated random graphs for both SN and VN. We generated OTNs by varying the size between 15–100 nodes. For each OTN, we generated an IP topology with a node count of 60% of that of the OTN and a link generation probability chosen to match the average nodal degree of known ISP topologies [43]. OTN links were assigned a capacity of 100Gbps, while IP links were assigned a random capacity between 10–20Gbps. For each combination of IP and OTN, we generated 20 VNs with 4–8 VNodes and a 0.5 probability of having a link between every pair of VNodes.

B. Evaluation Metrics

1) Cost Ratio: This is the ratio of costs obtained by two different approaches for solving the same problem instance. Cost is computed using (14) and measures the relative performance of two approaches.

2) Execution Time: The time required for an algorithm to solve one instance of *MULE*.

C. Comparison of FAST-MULE with OPT-MULE

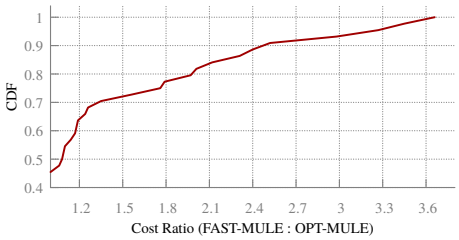


Fig. 5. FAST-*MULE* to OPT-*MULE* Cost Ratio

1) Cost Ratio Evaluation: First, we empirically measure the extent of additional resources allocated by *FAST-MULE* compared to *OPT-MULE*. Our cost function is proportional to the total bandwidth allocated for a VN and the new IP links. Therefore, cost ratio of *FAST-MULE* to *OPT-MULE* gives the extent of additional resources allocated by *FAST-MULE*. Fig. 5 shows the Cumulative Distribution Function (CDF) of cost ratio between *FAST-MULE* and *OPT-MULE*. Note that, *OPT-MULE* scaled up to only 35-node OTN. To mitigate the impact of VNode ordering during embedding, we run *FAST-MULE* 75 times, each time with a different VNode embedding order and take the best solution at the end. We observe from the results that 50% of the VNs admitted by FAST-*MULE* have an embedding cost within 10% of the optimal solution. On average, the admitted VNs have a cost within 1.47× of that of the optimal solution. These results are indeed promising given that *FAST-MULE* achieves this while executing 440× faster than *OPT-MULE* on average (10s for *FAST-MULE* vs. more than an hour per VN for *OPT-MULE*).

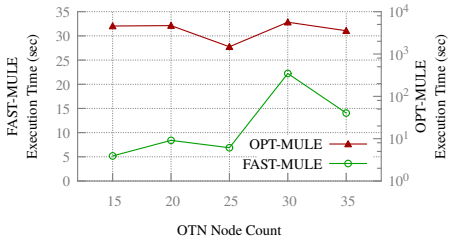


Fig. 6. Comparison of Execution Time

To further showcase the advantage of FAST-*MULE* compared to *OPT-MULE* we plot their execution times against varying SN size in Fig. 6. For similar problem instances in our evaluation, *FAST-MULE* executed 200× to 900× faster than *OPT-MULE*. Even after increasing the SN size, the execution time of *FAST-MULE* remained in the order of tens of seconds.

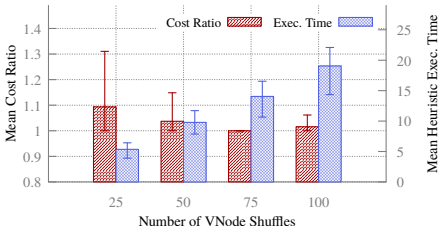


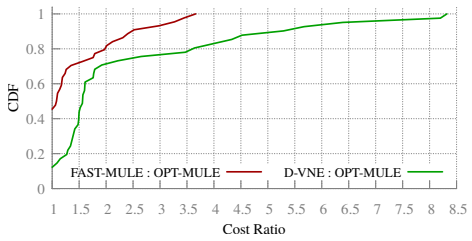
Fig. 7. Impact of VNode Shuffle on *FAST-MULE*'s Performance

2) Trade-off between Cost Ratio and Execution Time: We also evaluated the impact of the number of VNode orderings considered for the embedding. We present the results in Fig. 7, which shows how increasing the number of considered VNode orderings impacts the mean cost ratio (error bars represent the 5th and 95th percentile values) and the execution time of *FAST-MULE*. Clearly, as we increase the number of considered VNode orderings, *FAST-MULE* to *OPT-MULE* cost ratio decreases. This comes at the expense of increased execution time, which still remains in the order of tens of seconds. However, the gain becomes marginal as we go beyond 75 iterations. Hence, in our evaluation we opt for feeding *FAST-MULE* with 75 VNode orderings and select the best solution.

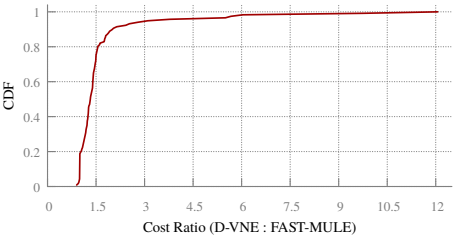
D. Comparative Analysis

Now, we evaluate how well FAST-*MULE* performs compared to the state-of-the art heuristic for multi-layer VNE [7]. We refer to [7] by D-VNE in the remaining. D-VNE constructs an auxiliary graph from the IP and Optical layers. The auxiliary graph contains precomputed optical paths that can be potentially chosen for creating new IP links. In contrast, we do not precompute paths in the OTN layer and let the embedding decide the best set of paths for jointly embedding VLinks and possible new IP links. D-VNE first embeds the VNodes using a greedy matching approach and then uses shortest path algorithm to route the VLinks between embedded VNodes. We modified D-VNE to fit to our context where we do not perform wavelength allocation and omit node resource requirements.

We begin by evaluating the cost ratio of D-VNE to *OPT-MULE* (Fig. 8(a)). The performance gap between D-VNE and *FAST-MULE* is evident from Fig. 8(a). D-VNE could embed VNs within 1.5× the cost of the optimal for ≈40% of the



(a) Comparison with OPT-*MULE*



(b) D-VNE to FAST-*MULE* Cost Ratio

Fig. 8. Comparison between FAST-*MULE* and D-VNE [7]

cases, whereas, *FAST-MULE* remains within the same bound for more than 70% of the cases. A head-to-head comparison between D-VNE and FAST-*MULE* (Fig. 8(b)) shows that on average D-VNE allocates ≈66% more resources compared to *FAST-MULE*. These results reflect the advantage of a joint embedding scheme compared to a disjoint approach adopted by D-VNE.

VII. Conclusion

This paper studied MULE, *i.e.*, multi-layer virtual network embedding on an IP-over-OTN substrate network. We proposed an ILP formulation, *OPT-MULE*, for optimally solving *MULE* and a heuristic, *FAST-MULE*, to address the computational complexity of the ILP. To the best of our knowledge, this is the first optimal solution to multi-layer VNE. Our evaluation of *FAST-MULE* shows that it performs within 1.47× of the optimal solution on average. *FAST-MULE* also outperformed the state-of-the-art heuristic for multi-layer VNE and allocated ≈66% less resources on average. Finally, we also proved that our proposed heuristic obtains optimal solution for star shaped VNs with uniform bandwidth demand in polynomial time.

We hope that this first endeavor will stimulate further research in multi-layer network virtualization. One possible future direction is to consider a dynamic OTN where more capacity can be provisioned by establishing new light paths in the underlying DWDM. Technological constraints posed by different optical network technologies such as wavelength continuity of DWDM networks or sub-wavelength resource allocation capabilities of elastic optical networks [28] are other interesting directions worth exploring in the future.

Acknowledgement

This work was supported in part by Huawei Technologies and in part by an NSERC Collaborative Research and Development Grant. Additionally, this work benefited from the use of the CrySP RIPPLE Facility at the University of Waterloo.

REFERENCES

[1] X. Zhao, V. Vusirikala, B. Koley, V. Kamalov, and T. Hofmeister, "The prospect of inter-data-center optical networks," *IEEE Communications Magazine*, vol. 51, no. 9, pp. 32–38, 2013.

[2] N. Ghani, S. Dixit, and T.-S. Wang, "On ip-over-wdm integration," *IEEE Communications Magazine*, vol. 38, no. 3, pp. 72–84, 2000.

[3] J. F. Botero, X. Hesselbach, M. Duelli, D. Schlosser, A. Fischer, and H. De Meer, "Energy efficient virtual network embedding," *Comm. Letters, IEEE*, vol. 16, no. 5, pp. 756–759, 2012.

[4] S. Ayoubi, C. Assi, K. Shaban, and L. Narayanan, "Minted: Multicast virtual network embedding in cloud data centers with delay constraints," *IEEE Trans. on Comm.*, vol. 63, no. 4, pp. 1291–1305, 2015.

[5] N. M. K. Chowdhury, M. R. Rahman, and R. Boutaba, "Virtual network embedding with coordinated node and link mapping," in *Proc. of IEEE INFOCOM*, 2009, pp. 783–791.

[6] 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.

[7] J. Zhang, Y. Ji, M. Song, H. Li, R. Gu, Y. Zhao, and J. Zhang, "Dynamic virtual network embedding over multilayer optical networks," *Journal of Optical Communications and Networking*, vol. 7, no. 9, pp. 918–927, 2015.

[8] 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 2016 conference on ACM SIGCOMM 2016 Conference*. ACM, 2016, pp. 87–100.

[9] F. Rambach, B. Konrad, L. Dembeck, U. Gebhard, M. Gunkel, M. Quagliotti, L. Serra, and V. López, "A multilayer cost model for metro/core networks," *Journal of Optical Communications and Networking*, vol. 5, no. 3, pp. 210–225, 2013.

[10] "ITU-t recommendation g.709/y.1331: Interfaces for the optical transport network," International Telecommunication Union, Tech. Rep., 2016. [Online]. Available: http://www.itu.int/rec/T-REC-G.709/

[11] A. L. Chiu, G. Choudhury, G. Clapp, R. Doverspike, M. Feuer, J. W. Gannett, J. Jackel, G. T. Kim, J. G. Klincewicz, T. J. Kwon *et al.*, "Architectures and protocols for capacity efficient, highly dynamic and highly resilient core networks [invited]," *Journal of Optical Communications and Networking*, vol. 4, no. 1, pp. 1–14, 2012.

[12] "OpenFlow-enabled Transport SDN," https://www.opennetworking.org/images/stories/downloads/sdn-resources/solution-briefs/sb-of-enabled-transport-sdn.pdf.

[13] C. Janz, L. Ong, K. Sethuraman, and V. Shukla, "Emerging transport sdn architecture and use cases," *IEEE Communications Magazine*, vol. 54, no. 10, pp. 116–121, October 2016.

[14] "Cisco nlight technology: A multi-layer control plane architecture for ip and optical convergence." [Online]. Available: https://www.cisco.com/c/en/us/products/collateral/switches/catalyst-3750-series-switches/whitepaper_c11-718852.html

[15] S. Dahlfort and D. CAVIGLIA, "IP-optical convergence: a complete solution," *Ericsson Review*, 2014.

[16] M. Chowdhury, F. Samuel, and R. Boutaba, "Polyvine: policy-based virtual network embedding across multiple domains," in *Proceedings of the second ACM SIGCOMM workshop on Virtualized infrastructure systems and architectures*. ACM, 2010, pp. 49–56.

[17] I. Houidi, W. Louati, W. B. Ameur, and D. Zeghlache, "Virtual network provisioning across multiple substrate networks," *Computer Networks*, vol. 55, no. 4, pp. 1011–1023, 2011.

[18] 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.

[19] M. R. Rahman and R. Boutaba, "Svne: Survivable virtual network embedding algorithms for network virtualization," *IEEE Transactions on Network and Service Management*, vol. 10, no. 2, pp. 105–118, 2013.

[20] M. F. Zhani, Q. Zhang, G. Simona, and R. Boutaba, "Vdc planner: Dynamic migration-aware virtual data center embedding for clouds," in *Integrated Network Management (IM 2013), 2013 IFIP/IEEE International Symposium on*. IEEE, 2013, pp. 18–25.

[21] Z. Zhang, A. X. Liu, X. Cheng, Y. Wang, X. Zhao, and S. Su, "Energy-aware virtual network embedding," *IEEE/ACM Transactions on Networking (TON)*, vol. 22, no. 5, pp. 1607–1620, 2014.

[22] S. Zhang, Z. Qian, J. Wu, S. Lu, and L. Epstein, "Virtual network embedding with opportunistic resource sharing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 3, pp. 816–827, 2014.

[23] N. Shahriar, R. Ahmed, S. R. Chowdhury, M. M. A. Khan, R. Boutaba, J. Mitra, and F. Zeng, "Connectivity-aware virtual network embedding," in *2016 IFIP Networking Conference (IFIP Networking) and Workshops*, May 2016, pp. 46–54.

[24] S. R. Chowdhury, R. Ahmed, M. M. Alam Khan, N. Shahriar, R. Boutaba, J. Mitra, and F. Zeng, "Dedicated protection for survivable virtual network embedding," *IEEE Transactions on Network and Service Management*, vol. 13, no. 4, pp. 913–926, 2016.

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

[26] M. F. Bari, R. Boutaba, R. Esteves, L. Z. Granville, M. Podlesny, M. G. Rabbani, Q. Zhang, and M. F. Zhani, "Data center network virtualization: A survey," *IEEE Communications Surveys & Tutorials*, vol. 15, no. 2, pp. 909–928, 2013.

[27] R. Jain and S. Paul, "Network virtualization and software defined networking for cloud computing: a survey," *IEEE Communications Magazine*, vol. 51, no. 11, pp. 24–31, 2013.

[28] 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, 2013.

[29] R. Nejabati, E. Escalona, S. Peng, and D. Simeonidou, "Optical network virtualization," in *15th International Conference on Optical Network Design and Modeling - ONDM 2011*, Feb 2011, pp. 1–5.

[30] Ć. Rožić, D. Klonidis, and I. Tomkos, "A survey of multi-layer network optimization," in *Optical Network Design and Modeling (ONDM), 2016 International Conference on*. IEEE, 2016, pp. 1–6.

[31] W. Bigos, B. Cousin, S. Gosselin, M. Le Foll, and H. Nakajima, "Survivable mpls over optical transport networks: Cost and resource usage analysis," *IEEE Journal on Selected Areas in Communications*, vol. 25, no. 5, 2007.

[32] M. Duelli, E. Weber, and M. Menth, "A generic algorithm for capex-aware multi-layer network design," in *Photonic Networks, 2009 ITG Symposium on*. VDE, 2009, pp. 1–8.

[33] I. Katib and D. Medhi, "Ip/mpls-over-otn-over-dwdm multilayer networks: an integrated three-layer capacity optimization model, a heuristic, and a study," *IEEE Transactions on Network and Service Management*, vol. 9, no. 3, pp. 240–253, 2012.

[34] O. Gerstel, C. Filsfils, T. Telkamp, M. Gunkel, M. Horneffer, V. Lopez, and A. Mayoral, "Multi-layer capacity planning for ip-optical networks," *IEEE Communications Magazine*, vol. 52, no. 1, pp. 44–51, 2014.

[35] Y. Ye, C. Assi, S. Dixit, and M. A. Ali, "A simple dynamic integrated provisioning/protection scheme in ip over wdm networks," *IEEE Communications Magazine*, vol. 39, no. 11, pp. 174–182, 2001.

[36] H. Zhang and A. Durresi, "Differentiated multi-layer survivability in ip/wdm networks," in *IEEE/IFIP Network Operations and Management Symposium*. IEEE, 2002, pp. 681–694.

[37] M. Tornatore, D. Lucerna, B. Mukherjee, and A. Pattavina, "Multilayer protection with availability guarantees in optical wdm networks," *Journal of Network and Systems Management*, vol. 20, no. 1, pp. 34–55, 2012.

[38] A. Alashaikh, D. Tipper, and T. Gomes, "Supporting differentiated resilience classes in multilayer networks," in *IEEE DRCN*. IEEE, 2016, pp. 31–38.

[39] P. Demeester, M. Gryseels, A. Autenrieth, C. Brianza, L. Castagna, G. Signorelli, R. Clemenfe, M. Ravera, A. Jajszczyk, D. Janukowicz *et al.*, "Resilience in multilayer networks," *IEEE Communications Magazine*, vol. 37, no. 8, pp. 70–76, 1999.

[40] "Link aggregation control protocol," http://www.ieee802.org/3/ad/public/mar99/seaman_1_0399.pdf.

[41] Y. Dinitz, N. Garg, and M. X. Goemans, "On the single-source unsplittable flow problem," in *Foundations of Computer Science, 1998. Proceedings. 39th Annual Symposium on*. IEEE, 1998, pp. 290–299.

[42] J. Edmonds and R. M. Karp, "Theoretical improvements in algorithmic efficiency for network flow problems," *Journal of the ACM (JACM)*, vol. 19, no. 2, pp. 248–264, 1972.

[43] N. Spring, R. Mahajan, and D. Wetherall, "Measuring isp topologies with rocketfuel," *ACM SIGCOMM Computer Communication Review*, vol. 32, no. 4, pp. 133–145, 2002.