On-Demand and Reliable vSD-EON Provisioning with Correlated Data and Control Plane Embedding

Jie Yin, Heqing Yin, Siqi Liu, Zhi Zhou, Xiaoliang Chen, Zuqing Zhu†
School of Information Science and Technology
University of Science and Technology of China, Hefei, China
†Email: {zqzhu}@ieee.org

***Abstract*—Software-defined elastic optical networks (SD-EONs) provide operators more flexibility to customize their optical infrastructure dynamically and adaptively, and network virtualization, *i.e.*, infrastructure-as-a-service (IaaS), enables multiple tenants to share the substrate infrastructure efficiently. In this paper, we study how to provision virtual SD-EONs (vSD-EONs) with the correlated data and control plane embedding (χ-VNE) that considers the quality-of-service (QoS) of virtual control plane (vCP), *i.e.*, availability and control channel latency. We propose a χ-VNE algorithm to solve the problem, design the network system to realize it, and accomplish proof-of-concept experimental demonstrations in an OpenFlow-based network testbed. Numerical and experimental results indicate that the proposed algorithm and system function well and can realize on-demand and reliable vSD-EON provisioning.**

***Index Terms*—Elastic optical networks (EONs), Virtual network embedding (VNE), Network function virtualization (NFV), Software-defined networking (SDN).**

I. Introduction

Recent advances on flexible-grid elastic optical networks (EONs) have demonstrated that the optical layer can be more adaptive and not be restricted by the fixed wavelength grids or modulation formats [1]. Hence, EON makes it easier for network operators to configure their networks according to customer demands. Meanwhile, to fully explore the benefits of EON, operators need an efficient and powerful network control and management (NC&M) mechanism, as the additional freedom and hence complexity on service provisioning have to be addressed properly [2]. Such an NC&M mechanism can be realized by implementing software-defined networking (SDN) in EONs (*i.e.*, building software-defined EONs (SD-EONs)) [3]. Previous work has indicated that SD-EONs can potentially achieve programmable and application-aware high-capacity networking with extended service reach [4, 5].

Note that, network virtualization, *i.e.*, infrastructure-as-a-service (IaaS), can further enhance the adaptivity of SD-EONs, since IaaS enables multiple tenants to share the substrate infrastructure efficiently [6]. Therefore, it is relevant to consider how to realize virtual SD-EONs (vSD-EONs) on a shared substrate EON. It is known that with data plane virtualization, one can provision virtual EONs [6]. However, the whole picture of vSD-EON provisioning should include control plane virtualization as well [7–9]. Basically, in addition to the virtualized data plane (vDP), vSD-EON operators need virtual SDN controllers to manage their tenants effectively. Previously, Vilalta *et al.* [7] leveraged the orchestration of SDN and network function virtualization (NFV) [10] to realize the virtual SDN controllers that can be dynamically deployed. They demonstrated that the NFV-based virtual controllers could effectively manage the virtual optical networks realized over a multi-domain multi-technology substrate network. Then, the work in [8] proposed to deploy NFV-based virtual controllers in clouds for virtual tenant networks.

The frameworks discussed in [7, 8] are promising and open up new opportunities for vSD-EONs. However, they did not addressed the algorithm to actually embed a vSD-EON in the substrate EON, *i.e.*, the correlated data and control plane virtual network embedding (χ-VNE). Specifically, we argue that to provision a practical vSD-EON, we need to consider the survivability of the virtualized control plane (vCP) and the quality-of-service (QoS) related to the control channel latency [11]. Hence, we may instantiate more than one NFV-based virtual controllers for the vCP and embed a NFV graph in the substrate network. Since the NFV graph has to be designed and embedded based on the status of substrate nodes/links (*i.e.*, their resources and availabilities) and the latency requirements on the control channels between vCP and vDP, vSD-EON provisioning desires an effective χ-VNE algorithm to optimize the arrangements of vCP and vDP jointly.

In this paper, we consider the orchestration of SDN and NFV for vSD-EON provisioning with correlated data and control plane embedding. Specifically, we study how to improve the survivability of vCP and limit the control channel latency between vCP and vDP during vSD-EON provisioning with χ-VNE. Here, we consider a substrate EON in which each optical switch is associated with an IT resource pool (*i.e.*, being attached with a high-performance server or a mini-datacenter) for control plane virtualization. Then, with the information on the substrate network and a vSD-EON, we design a NFV graph to represent the vCP, which models the required NFV-based virtual controller(s) and the relations among them as nodes and links, respectively. Hence, the problem of vSD-EON provisioning becomes how to embed the virtual switches of the vDP and the NFV graph based vCP jointly in the substrate EON. We propose a χ-VNE algorithm to solve the problem, design the network system to realize them, and accomplish proof-of-concept experimental demonstrations in an OpenFlow-based network testbed. Numerical and experimental results indicate that the proposed algorithm and system function well and can

978-1-5090-1328-9/16/$31.00 ©2016 IEEE

realize on-demand and reliable vSD-EON provisioning.

The rest of the paper is organized as follows. Section II describes the network model and formulates the problem of vSD-EON provisioning with χ-VNE. In Section III, the proposed algorithm is discussed in detail. We present the experimental demonstration and results in Section IV. Finally, Section V summaries the paper.

II. Problem Description

A. Network Model

1) Substrate Network: We model the substrate EON as an undirected graph $G(V^s, E^s)$, where V^s denotes the set of the substrate nodes (SNs), and E^s represents the set of substrate fiber links (SLs). Each SN $v^s \in V^s$ has an IT resource capacity as $c^s_{v^s}$, while the bandwidth resources on each SL $e^s \in E^s$ are B^s frequency slots (FS'). We assume that each FS has a bandwidth of 12.5 GHz. Note that, in χ-VNE, both vCP and vDP can consume IT and bandwidth resources in the substrate EON. We know that vDP requires IT and bandwidth resources as in the conventional VNE [6]. While vCP also needs IT and bandwidth resources to instantiate the NFV-based virtual controller(s) and establish the control channels. However, we hope to point out that compared with vDP, vCP consumes much less IT and bandwidth resources since the virtual controllers are very light-loaded and each control channel usually would not require more than 100 Mbps transmission capacity [12]. Therefore, in χ-VNE, we ignore the bandwidth requirement of vCP, and assume that a maximum number of $\hat{c}$ virtual controllers can be instantiated on each SN v^s no matter how many of its IT resources have already been allocated to vDP.

2) vSD-EON: We denote a vSD-EON request as $R = (V^r, E^r, a^r, d_m, d_s)$, where V^r is the set of virtual nodes (VNs), and E^r is the virtual links (VLs) set. In the vDP, each VN $v^r \in V^r$ demands for $c^r_{v^r}$ IT resources, while each VL $e^r \in E^r$ consumes a bandwidth of b^r. Here, for simplicity, we assume that the bandwidth requirement of each VL is the same. The rest three parameters in R are for the vCP.

We define the requested availability of vCP as [13]

$$a^r = \frac{T_{on}}{T}, \tag{1}$$

where T_{on} represents the working time of the vCP, and T is the total provisioning time of the vSD-EON. Note that, in practical situations, vCP is definitely not failure-proof without specific considerations, and since the vSD-EONs share substrate resources, the failure on an SN might bring down multiple virtual controllers. Meanwhile, as each vSD-EON counts on the centralized virtual controller to operate correctly, a controller failure can make the network unable to respond to any requests from the clients, and thus cause unimaginable losses to the vSD-EON operator. Therefore, we need to worry about the availability of vCP when provisioning a vSD-EON, and a straightforward way of enhancing the availability of vCP is to instantiate multiple virtual controllers instead of one [11]. The virtual controllers can leverage the mutual backup scheme to improve the availability of vCP, which means that they serve as the backups of one another, *i.e.*, for different vDP elements, a controller's role can be either working (*i.e.*, a master controller) or backup (*i.e.*, a slave one).

Apart from the requirement on availability, the QoS of vSD-EON also depends on the control channel latency, which is the messaging delay between a virtual controller and its vDP elements. Basically, a shorter control channel latency can make the controller respond faster to the clients' requests. Hence, we introduce parameters d_m and d_s to represent the upper-bounds of the control channel latencies between a master controller and its vDP elements and between a slave controller and its vDP elements, respectively[1]. Here, we have $d_m \leq d_s$ because the slave controllers are usually not in use and thus can have looser constraints on the control channel latency.

Fig. 1(a) shows an example on the vSD-EON request. It can be seen that the vDP consists of three VNs, each of which requires certain IT resources, and the three VLs among them have a bandwidth requirement of 3 FS'. According to the availability requirement of vCP, we figure out that two virtual controllers should be instantiated, *i.e.*, a master controller and a slave one. Meanwhile, we have $d_m = 1$ and $d_s = 2$ hops.

B. Problem Formulation

Basically, to provision a vSD-EON request with χ-VNE, we should accomplish two tasks: 1) embedding the vDP (*i.e.*, including both node mapping and link mapping) to satisfy its requirements on IT and bandwidth resources, and 2) designing the NFV graph to satisfy the vCP's requirements on availability and control channel latency and then embedding the NFV graph with node mapping. The first task is generally the same as the conventional VNE formulated in [6], with the only exception that the node mapping in it should consider the needs of the second task. For instance, the VNs should not be embedded too dispersedly in the substrate EON, otherwise, the constraints on the control channel latency would be difficult to satisfy with a relatively small number of virtual controllers.

As for the second task, we need to first determine the number of the virtual controllers that are needed. If we make sure that the virtual controllers are deployed on different SNs, the number of the controllers for each vDP element can be obtained by finding the smallest integer n that satisfies

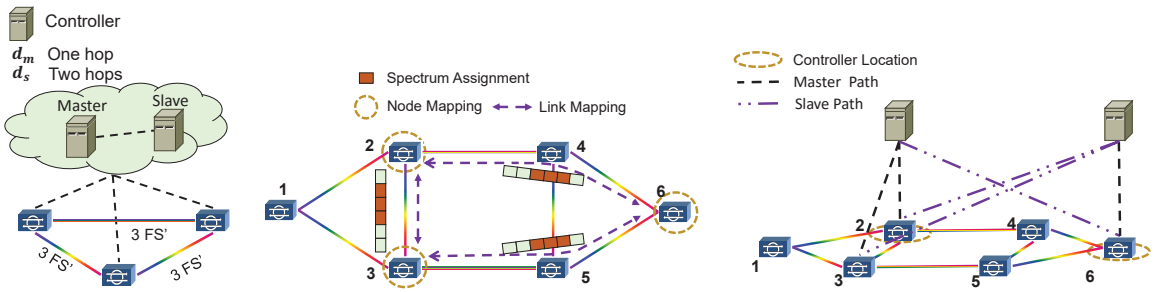
$$1 - (1 - \rho^s)^n \geq a^r, \tag{2}$$

where a^r is the availability requirement of the vSD-EON and ρ^s is the availability of an SN[2]. These virtual controllers constitute a NFV graph in which the master controllers can back up control information to the slave ones periodically. When a master controller is down, one of its slave controllers can take over its role to avoid service disruption.

We use the example in Fig. 1 to explain the procedure of provisioning a vSD-EON with χ-VNE intuitively. Basically, to provision the vSD-EON request in Fig. 1(a), we need to

[1]In this work, we use the hop-count of the shortest path between the SNs that carry the virtual controller and its vDP element to estimate the corresponding control channel latency.

[2]Here, for simplicity, we assume that the availabilities of all the SNs in the substrate EON are the same.



(a) vSD-EON request (b) χ-VNE result for vDP (c) χ-VNE result for vCP

Fig. 1. Example of provisioning a vSD-EON with χ-VNE.

accomplish the embedding tasks for both vDP and vCP. Fig. 1(b) shows the χ-VNE result for vDP, which includes mapping the three VNs to *SNs* 2, 3 and 6 and mapping the three VLs that each requires 3 FS' to substrate paths 2-3, 3-5-6 and 2-4-6. An FS-block that covers *FS'* $[3, 5]$ is allocated on each SL on the substrate paths to satisfy the bandwidth requirements of the vDP. The χ-VNE result for vCP is illustrated in Fig. 1(c). Since we only need to assign two virtual controllers to each vDP element and the constraints on control channel latency can also be satisfied with them, the NFV graph for the vCP is simply a link with two end-nodes. We embed the NFV graph by instantiating the virtual controllers on *SNs* 2 and 6, respectively. Then, the mutual backup scheme assigns the virtual controller on *SN* 2 as the master controller for the vDP elements on SNs 2 and 3 and as the slave controller for the one on *SN* 6. Similarly, the virtual controller on *SN* 6 serves as the master controller of the vDP element on *SN* 6 and as the slave controller for those on SNs 2 and 3.

III. χ-VNE Algorithms for vSD-EON Provisioning

Since the conventional VNE problem is already $\mathcal{NP}$-hard [14, 15], our χ-VNE problem is also $\mathcal{NP}$-hard. Hence, we propose a heuristic algorithm to solve it time-efficiently.

A. χ-VNE with Joint Mapping (χ-VNE-J)

We design the χ-VNE algorithm to handle the embedding of vDP and vCP jointly, *i.e.*, χ-VNE with joint mapping (χ-VNE-J). To limit the time complexity, χ-VNE-J embeds the vDP and vCP of a vSD-EON in two phases, but in the phase of vDP embedding, it considers the needs of vCP embedding. Specifically, for each vSD-EON request, χ-VNE-J first tries to embed the vDP by leveraging the LaLRC-LaSP algorithm we developed in [6]. Here, we reformulate the local resource capacity in LaLRC-LaSP as below to make it consider the needs of the subsequent vCP embedding.

$$w_{v^s} = \alpha \cdot c_{v^s} \cdot deg(v^s) - \beta \cdot \frac{\sum_{u^s \in \widetilde{V}^m} hops(v^s, u^s)}{|\widetilde{V}^m|}, \quad (3)$$

where function $deg(\cdot)$ returns the degree of a node, function $hops(\cdot)$ returns the hop-count of the shortest substrate path between two SNs, set $\widetilde{V}^m$ stores the SNs that have already been selected for vDP embedding for the vSD-EON, and α and β are the positive constants to balance the importance of the two terms in Eq. (3).

Basically, by adding the second term, we make LaLRC-LaSP tend to embed vDP elements close to each other in the substrate EON. Hence, the subsequent vCP embedding could be easier. Then, the modified LaLRC-LaSP has the procedure as follows. We first construct a layered auxiliary graph (LAG) based on the spectrum utilization in the substrate EON. Then, for each VN v^r in the vDP, we calculate its local resource capacity as the product of its IT resource capacity and node degree. While for each SN in the substrate EON, its local resource capacity is obtained with Eq. (3). We accomplish the node mapping by selecting the first feasible LAG, sorting both the SNs and the VNs in descending order of their local resource capacities separately, and then mapping the sorted VNs to the sorted SNs one by one. After embedding each VN, we update $\widetilde{V}^m$ and recalculate the local resource capacity of each SN. After the node mapping, link mapping is conducted with shortest path routing in the substrate EON.

When the vDP is embedded, we proceed to handle the vCP. With the availability requirement a^r, we can determine how many virtual controllers need to be assigned to each vDP element (*i.e.*, each VN in vDP) using Eq. (2). Then, considering the cost of deploying virtual controllers on SNs, we assume that the virtual controllers will only be deployed on the SNs that have been used in the embedding of vDP. We denote the set of these SNs as V^m, which includes the SNs that are chosen for node mapping and the intermediate SNs on the substrate paths used for link mapping. Then, the embedding of vCP becomes 1) to select SNs in V^m to deploy enough virtual controllers, and 2) to determine each virtual controller's role (*i.e.*, a master or slave controller) for the vDP elements to satisfy the constraints on control channel latency. The problems can be solved with the following integer linear programming (ILP) model, which tries to minimize the number of deployed virtual controllers in the vCP.

Notations:

- $G^m(V^m, E^m)$: substrate sub-topology that is used to embed vDP, where set E^m includes all the used SLs.
- V^n: set of SNs that the vDP elements are embedded on, *i.e.*, $|V^n| = |V^r|$ and $V^n \subset V^m$.
- $h_{(u,v)}$: hop-count of the shortest substrate path between an SN pair $u \in V^m$ and $v \in V^n$.

- $\hat{c}_u$: the remaining number of virtual controllers that can be deployed on SN $u \in V^m$.
- n: number of virtual controllers that need to be assigned to each vDP element, *i.e.*, being obtained with Eq. (2).

Variables:

- $x^m_{u,v}$: boolean variable that equals 1 if a virtual controller is deployed on SN $u \in V^m$ as the master controller of the vDP element on SN $v \in V^n$, and 0 otherwise.
- $x^s_{u,v}$: boolean variable that equals 1 if a virtual controller is deployed on SN $u \in V^m$ as the slave controller of the vDP element on SN $v \in V^n$, and 0 otherwise.
- x_u: boolean variable that equals 1 if a virtual controller is deployed on $u \in V^m$, and 0 otherwise.

Objective: The optimization objective is to minimize the total number of deployed virtual controllers.

$$Minimize \quad \sum_{u \in V^m} x_u. \tag{4}$$

Constraints:

$$\sum_{u \in V^m} x^m_{u,v} = 1, \quad \forall v \in V^n. \tag{5}$$

$$\sum_{u \in V^m} x^s_{u,v} = n - 1, \quad \forall v \in V^n. \tag{6}$$

Eqs. (5)-(6) ensure that each vDP element is assigned one master controller and enough slave controllers.

$$x^m_{u,v} \cdot h_{(u,v)} \leq m_d \quad \forall u \in V^m, v \in V^n. \tag{7}$$

$$x^s_{u,v} \cdot h_{(u,v)} \leq s_d \quad \forall u \in V^m, v \in V^n. \tag{8}$$

Eqs. (7)-(8) ensure that the constraints on control channel latency are satisfied.

$$x^m_{u,v} + x^s_{u,v} \leq x_u \quad \forall u \in V^m, v \in V^n. \tag{9}$$

Eq. (9) ensures that the master and slave controller of a vDP element are not deployed at the same SN.

$$x_u \leq \hat{c}_u \quad \forall u \in V^m. \tag{10}$$

Eq. (10) ensures that the number of the newly deployed virtual controllers on each SN does not exceed its capacity.

Algorithm 1 shows the detailed procedure of χ-VNE-J. *Lines* 1-5 are for embedding the vDP. The preprocessing for embedding the vCP are in *Lines* 6-7. Then, we solve the ILP to obtain the χ-VNE result for vCP as in *Lines* 8-12. Note that, the ILP only contains $|V^m| \cdot (2 \cdot |V^n| + 1)$ variables and $|V^m| \cdot (3 \cdot |V^n| + 1) + 2 \cdot |V^n|$ constraints. As $|V^m|$ and $|V^n|$ are usually relatively small numbers (*e.g.*, less than 10), the ILP can be solved within a fairly small amount of time (*i.e.*, less than 97 msec in the simulations that will be discussed below). Finally, *Line* 13 provisions the vSD-EON with χ-VNE.

Algorithm 1: χ-VNE with Joint Mapping (χ-VNE-J)

Inputs: Substrate EON $G^s(V^s, E^s)$ and vSD-EON request $R = (V^r, E^r, a^r, d_m, d_s)$.

1 apply the modified LaLRC-LaSP algorithm to obtain the χ-VNE result for vDP;
2 **if** *the vDP in vSD-EON cannot be embedded* **then**
3 mark R as blocked;
4 **return**;
5 **end**
6 store all the SNs used in vDP embedding in V^m;
7 determine n as the number of virtual controllers to be assigned to each vDP element with Eq. (2);
8 solve the ILP described by Eqs. (4)-(10) to get the χ-VNE result for vCP;
9 **if** *the vCP in vSD-EON cannot be embedded* **then**
10 mark R as blocked;
11 **return**;
12 **end**
13 provision the vSD-EON with χ-VNE;

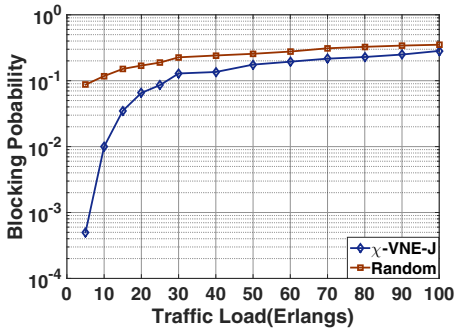


Fig. 2. Blocking probability of on-demand vSD-EON provisioning.

B. Performance Evaluation

We use numerical simulations to evaluate the performance of the proposed algorithms. The simulations use the Deutsche Telecom (DT) topology that consists of 14 nodes and 23 links [16]. Initially, each SN v^s has an IT capacity as $c^s_{v^s} = 200$ units, while the initial bandwidth capacity of each SL is 200 FS'. We assume that each SN's availability is $\rho^s = 0.9$. The dynamic vSD-EON requests are generated with the Poisson traffic model. They use the random topologies that are created by the GT-ITM tool [17]. The IT resource requirements of each vDP element is uniformly distributed within $[1, 10]$ units, and the bandwidth requirements of each VL follows the uniform

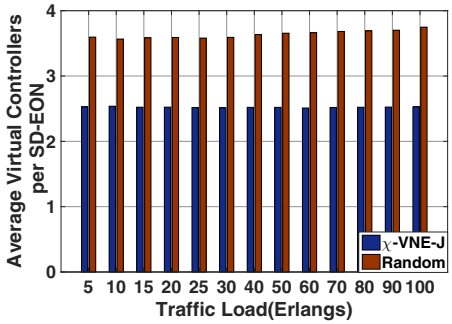


Fig. 3. Number of virtual controllers deployed for vSD-EON provisioning.

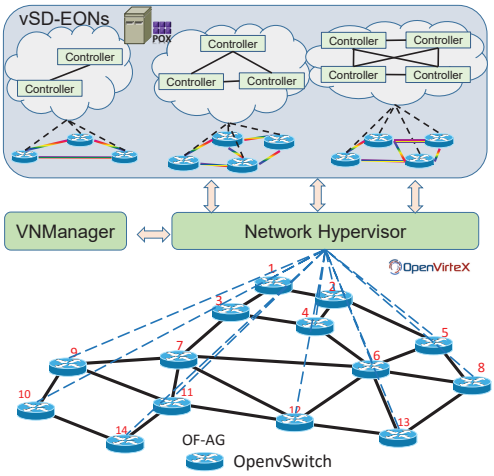


Fig. 4. System implementation of experimental testbed.

distribution within $[1, 10]$ FS'. The numbers of vDP elements in the vSD-EONs are uniform distributed within $[3, 4]$. The vCP's availability requirement is randomly selected within $[0.99, 0.999]$. We assume that all the vSD-EONs have the same constraints on control channel latency as $d_m = 1$ hop and $d_s = 2$ hops, respectively.

For the benchmark algorithm, we introduce one that deploys virtual controllers on SNs randomly (Random) for vCP embedding while its vDP embedding uses the same procedure as that in the original LaLRC-LaSP in [6]. Fig. 2 shows the results on blocking probability, which indicates that χ-VNE-J provides much lower blocking probability than the benchmark. This is because our algorithm achieves correlated data and control plane embedding and allocates the least number of virtual controllers for each vSD-EON. Fig. 3 shows the results on average number of virtual controllers deployed for each vSD-EON. It can be seen that χ-VNE-J deploys less virtual controllers than Random. Again, this is because when embedding vDP, χ-VNE-J considers the needs of subsequent vCP embedding. Hence, with χ-VNE-J, we can effectively reduce the deployment cost of vSD-EONs.

IV. Experimental Demonstration

We implement χ-VNE-J in an OpenFlow-based network system to demonstrate on-demand vSD-EON provisioning. The substrate optical switches are software-emulated, each of which is programmed with OpenvSwitch and runs on an independent Linux server. We connect 14 such servers together to emulate the substrate EON, which uses the DT topology as shown in Fig. 4. Fig. 4 also indicates that vSD-EONs are provisioned with the slicing achieved by the network hypervisor (NHV), which is implemented with OpenVirteX [18], while the actual χ-VNE schemes are calculated by the virtual network manager (VNManger). For each vSD-EON, its virtual controllers are implemented with POX.

We first conduct an experiment to show the procedure for vSD-EON creation. Here, the vDP of the vSD-EON takes a line topology that consists of 4 VNs and 3 VLs, and according to its availability requirement, the vCP assigns two controllers to each vDP element. Fig. 5(a) shows the communications

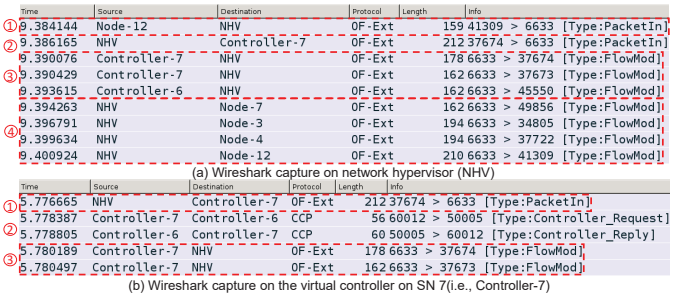
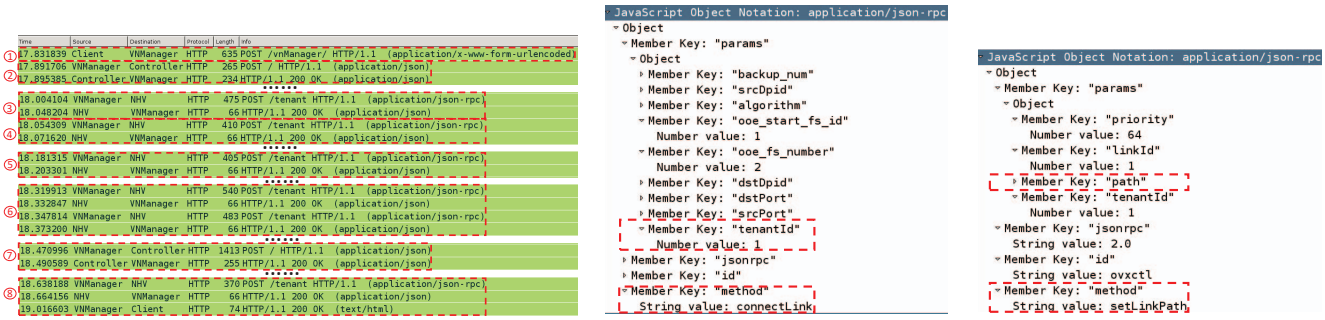


Fig. 6. Lightpath provisioning procedure in a vSD-EON.

we captured in the experiment, for provisioning the vSD-EON. In *Step* 1, the client submits the vSD-EON request to VNManager. In *Step* 2, VNManager applies χ-VNE-J to obtain the provisioning scheme. Specifically, for the vDP, the node mapping (*i.e.*, VN→SN) is $\{1{\to}12,\ 2{\to}3,\ 3{\to}4,\ 4{\to}6\}$ and link mapping is $\{(1,2){\to}(12,7,3),\ (2,3){\to}(3,4),\ (3,4){\to}(4,6)\}$. For the vCP, we allocate the virtual controllers on *SNs* 6 and 7, respectively. After instantiating the virtual controllers, VNManager communicates with them to assign their roles for the vDP elements. Here, the controller on *SN* 7 serves as the master controller for the vDP elements embedded on *SNs* 12 and 3, and serves as the slave one for those embedded on *SNs* 4 and 6, while the role assignment of the controller on *SN* 6 is the opposite. In *Step* 3, VNManager informs NHV about the provisioning scheme and create a tenant ID for the vSD-EON. Then, in *Steps* 4 and 5, by communicating with NHV, VNManager creates the virtual switches and the virtual ports on them, respectively. The link mapping to connect the virtual switches is accomplished in *Step* 6. Specifically, VNManager instructs NHV to create VLs by connecting the virtual switches through virtual ports and embed the VLs on substrate paths. Figs. 5(b) and (c) show the snapshots of the messages used for link mapping. The message in Fig. 5(b) indicates that the tenant ID of the vSD-EON is 1 and its bandwidth requirement for vDP is 25 GHz. Fig. 5(c) shows that VNManager informs NHV about the link mapping result. After the link mapping, VNManager makes the virtual controllers operational in *Step* 7. Finally, in *Step* 8, the vSD-EON is provisioned and handed over to the client.

Next, we demonstrate lightpath provisioning in the vSD-EON. Here, the vSD-EON operator needs to establish a lightpath from *VN* 1 (on *SN* 12) to *VN* 3 (on *SN* 4). Figs. 6(a) and 6(b) show the wireshark captures for the lightpath provisioning on NHV and Controller-7, respectively. Firstly, *VN* 1 (*i.e.*, Node-12 in Fig. 6(a)) encodes a *Packet-In* message and sends it to NHV. After checking the tenant ID of the vSD-EON, NHV forwards the message to the master virtual controller of *VN* 1, which has been deployed on *SN* 7 and is marked as Controller-7 in Fig. 6(b). Controller-7 determines that the lightpath should be routed as *VN* 1 to *VN* 2 to *VN* 3 in the vDP. Note that, since the master controller of *VN* 3 has been deployed on *SN* 6 (*i.e.*, Controller-6 in Fig. 6), Controller-7 needs to communicate with Controller-6 to set up the lightpath cooperatively [11]. This is why in Fig. 6(b), we can see that Controller-7 and Controller-6 talk with each



(a) Overall message exchanges for vSD-EON creation (b) Message used to connect a VL (c) Message used to embed a VL

Fig. 5. Example of vSD-EON creation on system architecture.

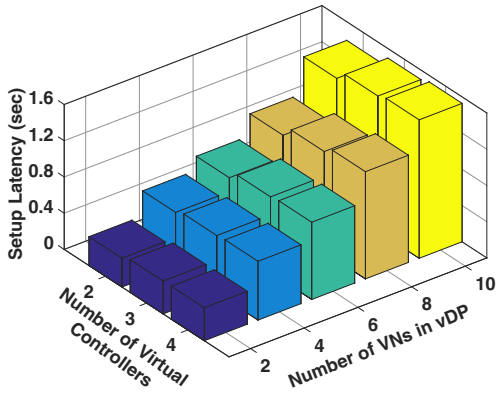


Fig. 7. Experimental results on setup latency for provisioning vSD-EONs.

other with the controller communication protocol (CCP) [11]. Next, the virtual controllers send *Flow-Mod* messages to *VNs* 1, 2 and 3 through NHV, to set up the lightpath, two of which are captured on Controller-7 as shown in Fig. 6(b). Note that, as VL $(1, 2)$ is mapped onto substrate path $(12, 7, 3)$, NHV translates the destinations of related *Flow-Mod* messages and sends them to *SNs* 12, 7, and 3, as shown in Fig. 6(a).

Finally, we try to dynamically provision vSD-EONs in the testbed, which have different vDP sizes and numbers of virtual controllers, and measure the setup latency. Fig. 7 shows the results, and we can see that even though the setup latency does increases with the scales of vDP and vCP, the longest setup latency is still shorter than 1.6 seconds when the vSD-EON has 10 VNs in vDP and 4 virtual controllers in vCP. These results verify that on-demand vSD-EON provisioning is feasible with the proposed system.

V. Conclusion

This paper studied the orchestration of SDN and NFV for vSD-EON provisioning with correlated data and control plane embedding. We proposed a χ-VNE algorithm for the problem, realized it in an OpenFlow-based network testbed, and accomplished proof-of-concept experimental demonstrations.

Acknowledgments

This work was supported in part by NSFC Project 61371117, Fundamental Research Funds for the Central Universities (WK2100060010), Natural Science Research Project for Universities in Anhui (KJ2014ZD38), and Strategic Priority Research Program of the CAS (XDA06010302).

References

[1] P. Lu *et al.*, "Highly-efficient data migration and backup for big data applications in elastic optical inter-datacenter networks," *IEEE Netw.*, vol. 29, pp. 36–42, Sept./Oct. 2015.

[2] Z. Zhu, W. Lu, L. Zhang, and N. Ansari, "Dynamic service provisioning in elastic optical networks with hybrid single-/multi-path routing," *J. Lightw. Technol.*, vol. 31, pp. 15–22, Jan. 2013.

[3] Z. Zhu *et al.*, "OpenFlow-assisted online defragmentation in single-/multi-domain software-defined elastic optical networks," *J. Opt. Commun. Netw.*, vol. 7, pp. A7–A15, Jan. 2015.

[4] R. Casellas *et al.*, "Control and management of flexi-grid optical networks with an integrated stateful path computation element and OpenFlow controller," *J. Opt. Commun. Netw.*, vol. 5, no. 10, pp. A57–A65, 2013.

[5] Z. Zhu *et al.*, "Demonstration of cooperative resource allocation in an OpenFlow-controlled multidomain and multinational SD-EON testbed," *J. Lightw. Technol.*, vol. 33, pp. 1508–1514, Apr. 2015.

[6] L. Gong and Z. Zhu, "Virtual optical network embedding (VONE) over elastic optical networks," *J. Lightw. Technol.*, vol. 32, pp. 450–460, Feb. 2014.

[7] R. Vilalta *et al.*, "Network virtualization controller for abstraction and control of OpenFlow-enabled multi-tenant multi-technology transport networks," in *Proc. of OFC 2015*, pp. 1–3, Mar. 2015.

[8] R. Munoz *et al.*, "Integrated SDN/NFV management and orchestration architecture for dynamic deployment of virtual SDN control instances for virtual tenant networks," *J. Opt. Commun. Netw*, vol. 7, pp. B62–B70, Nov. 2015.

[9] B. Zhao, X. Chen, J. Zhu, and Z. Zhu, "Survivable control plane establishment with live control service backup and migration in SD-EONs," *J. Opt. Commun. Netw.*, vol. 8, pp. 371–381, Jun. 2016.

[10] "Network functions virtualization (NFV)," Tech. Rep., Oct. 2014. [Online]. Available: https://portal.etsi.org/Portals/0/TBpages/NFV/Docs/NFV_White_Paper3.pdf

[11] X. Chen *et al.*, "Leveraging master-slave openflow controller arrangement to improve control plane resiliency in SD-EONs," *Opt. Express*, vol. 23, pp. 7550–7558, Mar. 2015.

[12] D. Kreutz *et al.*, "Software-defined networking: A comprehensive survey," *Proc. IEEE*, vol. 103, pp. 14–76, Jan. 2015.

[13] X. Chen *et al.*, "Flexible availability-aware differentiated protection in software-defined elastic optical networks," *J. Lightw. Technol.*, vol. 33, pp. 3872–3882, Sept. 2015.

[14] M. Chowdhury, M. Rahman, and R. Boutaba, "ViNEYard: Virtual network embedding algorithms with coordinated node and link mapping," *IEEE/ACM Trans. Netw.*, vol. 20, pp. 206–219, Jan. 2012.

[15] L. Gong, H. Jiang, Y. Wang, and Z. Zhu, "Novel location-constrained virtual network embedding (LC-VNE) algorithms towards integrated node and link mapping," *IEEE/ACM Trans. Netw., in Press*, pp. 1–14, 2016.

[16] F. Agraz *et al.*, "Experimental demonstration of centralized and distributed impairment-aware control plane schemes for dynamic transparent optical networks," in *Proc. of OFC 2010*, pp. 1–3, Mar 2010.

[17] E. Zegura, K. Calvert, and S. Bhattacharjee, "How to model an internetwork," in *Proc. INFOCOM 1996*, pp. 594–602, Mar. 1996.

[18] Openvirtex. [Online]. Available: http://http://ovx.onlab.us/