Availability-Aware Survivable Virtual Network Embedding in Optical Datacenter Networks

Huihui Jiang, Yixiang Wang, Long Gong, and Zuqing Zhu

***Abstract*—In this work, we study the availability-aware survivable virtual network embedding (A-SVNE) problem in optical interdatacenter networks that use wavelength-division multiplexing. With A-SVNE, we try to satisfy the availability requirement of each virtual component (i.e., a virtual link or a virtual node) in a virtual network. We first analyze the availability of a virtual component based on the availabilities of the substrate link(s) and node(s). Then, we formulate an integer linear programming model for the A-SVNE problem and propose several time-efficient heuristics. Specifically, we design two node mapping strategies: one is sequential selection using efficient weights defined by the availability information, while the other uses auxiliary graphs to transform the problem into a classical problem in graph theory, i.e., the maximum-weight maximum clique. Finally, we use extensive simulations to compare the proposed A-SVNE algorithms with existing ones in terms of the blocking probability, availability gap, and penalty due to service-level agreement violations, and the results indicate that our algorithms perform better.**

***Index Terms*—Network virtualization; Service availability; Survivable virtual network embedding (SVNE); Wavelength-division multiplexing (WDM).**

I. Introduction

Recently, the boosting of cloud computing has stimulated large enterprises, e.g., Google, Amazon, and Facebook, to deploy datacenters (DCs) in a geographically distributed manner. Hence, the network that interconnects DCs, i.e., the inter-DC network, started to attract intensive attention from both academia and industry. Due to the fact that DCs usually carry massive data and applications with ever-increasing demands, the inter-DC networks need to ensure high throughput and low latency [1]. As optical networks with wavelength-division multiplexing (WDM) provide huge bandwidth capacity and employ wavelength switching to offer low-latency data transmission, they become the only feasible candidate for the physical layer of inter-DC networks [2–4].

As one of the key enabling technologies for cloud computing, network virtualization allows multiple virtual networks (VNTs) to coexist on the same substrate network (SNT) [5,6]. It is known that constructing VNTs over the SNT requires virtual network embedding (VNE) [7–11], which involves node mapping and link mapping to satisfy the VNTs' resource requirements. Note that the basic VNE itself is a relatively complex problem and has been proved to be NP-hard [5]. However, we may need to consider more sophisticated VNE for practical network virtualization in optical inter-DC networks [12,13]. For instance, because multiple VNTs can share the same substrate components (i.e., DCs and fiber links), the network failure that occurs on a single substrate component might bring down the services of multiple VNTs simultaneously. This would be catastrophic in optical inter-DC networks, since the DCs (i.e., facility nodes) and fiber links usually carry massive data and live communications. Moreover, a major natural disaster like an earthquake can destroy multiple DCs and/or fiber links and induce even longer service interruption and more revenue loss [14,15]. Hence, survivable VNE (SVNE) that can ensure the intactness of VNTs during network failures should be considered.

Nevertheless, realizing cost-effective SVNE in optical inter-DC networks is challenging. First of all, due to the complexity of the problem, time-efficient algorithms are highly desired to adapt SVNE to real-time network operations. Second, in SVNE, there is a trade-off between a VNT's availability and the substrate resources that it consumes, and VNTs from different customers may have differentiated availability requirements. Hence, we should study an availability-aware SVNE (A-SVNE) that can satisfy the various availability requirements of VNTs. Lastly but most importantly, in A-SVNE, a customer may apply various availability requirements on different virtual components (i.e., virtual nodes and virtual links), as they may play different roles in the VNT.

In this paper, we study the A-SVNE scheme for an optical inter-DC network that uses WDM. Basically, with the A-SVNE, we try to satisfy the availability requirement of each individual virtual component in a VNT. We first analyze the availability of a virtual component based on the availabilities of the substrate link(s) and node(s). Then, we formulate an integer linear programming (ILP) model for the A-SVNE problem and propose several time-efficient heuristics. Specifically, we design two node mapping strategies based on the sequential selection and the maximum clique in an auxiliary graph (AG), respectively. In

Manuscript received June 5, 2015; revised October 11, 2015; accepted October 12, 2015; published November 24, 2015 (Doc. ID 242446).

The authors are with the School of Information Science and Technology, University of Science and Technology of China, Hefei, Anhui 230027, China (e-mail: zqzhu@ieee.org).

http://dx.doi.org/10.1364/JOCN.7.001160

sequential node mapping, the weights of virtual nodes (VNs) and substrate nodes (SNs) are defined based on their availability information (AI) to quantify the corresponding embedding potential. For maximum-clique-based node mapping, we transform it into a classical problem in graph theory, i.e., the maximum-weight maximum clique (MWMC) problem, by leveraging an AG. We then prove that any maximal clique in the AG is also the maximum clique, and hence ensure that the MWMC problem can be solved in linear time. Regarding link mapping, we leverage a complete bipartite graph between the mapped SNs to present the connections for each virtual link (VL) and find substrate paths for the connections. Finally, we use extensive simulations to compare the proposed A-SVNE algorithms with the existing ones in [16,17] in terms of the blocking probability, availability gap, and penalty due to service level agreement (SLA) violations, and the results indicate that our algorithms perform better.

The contributions of this work are listed as follows.

- To the best of our knowledge, this is the first work on A-SVNE with WDM substrate networks that considers the availabilities of both substrate links (SLs) and SNs and tries to satisfy the explicit availability requirements of each VN and VL.
- We formulate an ILP model to solve the A-SVNE problem based on the analytical expressions to calculate the availability of a virtual component based on the availabilities of the substrate components.
- We design efficient weights for VNs and SNs based on the AI of the incident VLs and substrate paths in the sequential node selection, which can quantify the corresponding embedding potential.
- To coordinate with link mapping and map all the VNs simultaneously, we transform the node mapping into the MWMC problem with an AG. We prove that any maximal clique in the AG is also the maximum clique, and hence the MWMC problem can be solved in linear time.

The rest of the paper is organized as follows. Section II summarizes the related work. We formulate the problem of A-SVNE in Section III. We present the ILP model for A-SVNE in Section IV. Section V discusses the time-efficient heuristics. The performance evaluation is shown in Section VI, and, finally, Section VII summarizes the paper.

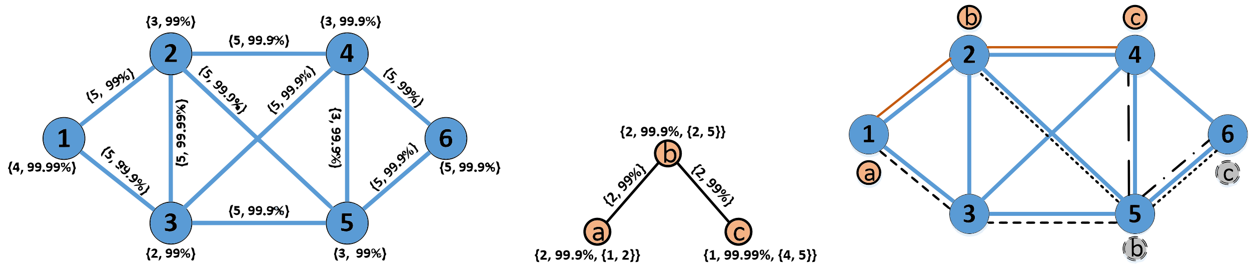
II. Related Work

Previously, people have studied SVNE with link protection to address SL failures [15], and the work in [18] considered switch-node failures and proposed an SVNE scheme that adopted preconfigured cycles. Since in an inter-DC network, the DCs (i.e., facility nodes) are very valuable and vulnerable assets, the node-failure-proof SVNE problem has recently attracted notable interest [19,20]. By solving the multicommodity flow problem for splittable and nonsplittable flows, the authors of [19] formulated two ILP models to design node-failure-proof SVNE schemes. The work in [20] proposed failure-dependent protection (FDP) for SVNE, which may rearrange all the node mappings of a VNT when a node failure happens. Even though the node remapping in FDP leads to efficient resource utilization during restoration, the additional operational complexity from the service migrations could become an issue. The investigation in [21] designed SVNE schemes that can address region failures (i.e., multiple node and link failures that are geographically correlated in the SNT). More recently, the authors of [17] studied the SVNE scheme to protect VNTs against single substrate node/link failures with dedicated protection. Note that these studies only focused on designing the protection schemes to address certain substrate failure(s) but did not consider A-SVNE that can use different protection schemes to satisfy the VNTs' availability requirements.

Generally, the service availability of a VNT can be defined as the ratio of its on-service duration to the total provisioning period, i.e., availability is the probability that the VNT's service is available at any given time. As most of the substrate failures are unpredictable, a VNT's service can still become unavailable even though it was provisioned with SVNE. Therefore, in practical network operation, each customer usually specifies the required availability in its SLA with the service provider and uses availability as a metric to measure the quality of service that it receives [22,23]. Moreover, customers may have differentiated availability requirements. Therefore, the A-SVNE scheme is very relevant and should be studied carefully. Previously, people have proposed availability-aware provisioning schemes that use different protection schemes to satisfy the availability requirements of lightpaths in WDM networks [24,25]. However, provisioning a VNT is much more complex and definitely needs further studies.

Nevertheless, most of the existing studies on A-SVNE were not for WDM substrate networks and did not consider the substrate resource allocation in discrete wavelength channels or the wavelength continuity constraint for substrate paths. In [16], under the assumption that the SNs are available all the time, Herker *et al.* studied an A-SVNE scheme that used path protection to satisfy the availabilities of VLs. The work in [26] studied an A-SVNE scheme that only considered the explicit availability requirement on VLs, and analyzed the relation between the embedding cost and the availability of substrate network using the A-SVNE algorithm in [16]. An A-SVNE scheme that tried to construct VNTs in an intra-DC network with availability constraints was investigated in [27], where the authors considered the availability requirement of each VNT as a whole and did not address the explicit availability requirement of each virtual component. Note that for the A-SVNE in optical inter-DC networks, a customer may apply differentiated availability requirements not only on VLs but also on VNs, as they may play different roles in their VNT.

In this work, we consider A-SVNE in optical inter-DC networks with more generic availability requirements. Specifically, for each VNT, the customer can apply an



(a) Substrate network (SNT). (b) Virtual network (VNT). (c) A-SVNE Result.

Fig. 1. Example of A-SVNE.

explicit availability requirement on each individual virtual component. We also consider the resource allocation on SLs in discrete wavelength channels and take care of the wavelength continuity constraint in link mapping.

III. Problem Description

A. Network Model

1) Substrate Network Model: We model the SNT, i.e., the optical inter-DC network, as an undirected graph $G^s(V^s, E^s)$, where V^s represents the set of SNs and E^s is for the SL set. Each SN $v^s \in V^s$ has a computing capacity $c^s_{v^s}$ and its availability is $an^s_{v^s}$, and $l^s_{v^s}$ denotes its physical location. We assume that each SL $e^s \in E^s$ can accommodate B^s wavelengths and its availability is $al^s_{e^s}$. Figure 1(a) shows a simple example of SNT. The numbers in braces next to the SNs and SLs represent their residual resources and availabilities. We denote the set of precomputed substrate paths as P^s, and $P^s_{(s,d)}$ is the set of substrate paths from node s to node d.

2) VNT Request: A VNT request is also modeled as an undirected graph $G^r(V^r, E^r)$, where V^r and E^r are the sets of VNs and VLs, respectively. We use the notations $c^r_{v^r}$ and $an^r_{v^r}$ to represent the computing capacity and availability requirements of each VN $v^r \in V^r$, respectively. In addition, each VN v^r also has a preferred location $l^r_{v^r}$, which corresponds to a set of candidate SNs, denoted as $\Phi(v^r)$.[1] Each VL $e^r \in E^r$ has a bandwidth requirement $bw^r_{e^r}$ in number of wavelengths and an availability requirement $al^r_{e^r}$. Figure 1(b) shows a VNT request. The braces near the VNs include the requirements on computing capacity and availability and the candidate SN set, respectively. The bandwidth and availability requirements are labeled on each VL.

B. Availability Analysis for Virtual Components

The availability of a network component is defined as [22]

$$A = \frac{\text{MTBF}}{\text{MTBF} + \text{MTTR}}, \tag{1}$$

where MTBF is the working time between two adjacent failures and MTTR is the duration of the service outage due to network failures. In this work, we assume that the failures on the substrate components (i.e., SNs and SLs) are independent.

1) Virtual Node Availability: As a VN $v^r \in V^r$ is embedded onto a set of SNs m_{v^r} to satisfy its availability requirement, the service on the VN will become unavailable when and only when all its mapped SNs in m_{v^r} are down. Hence, the availability of v^r can be simply obtained as

$$A_{v^r} = 1 - \prod_{v^s \in m_{v^r}} (1 - an^s_{v^s}). \tag{2}$$

2) Virtual Link Availability: First of all, the availability of a substrate path p^s can be calculated as

$$A_{p^s} = \left(\prod_{v^s \in \hat{p}^s} an^s_{v^s}\right) \cdot \left(\prod_{e^s \in p^s} al^s_{e^s}\right), \tag{3}$$

where $v^s \in \hat{p}^s$ represents the SNs on p^s except for its two end nodes and $e^s \in p^s$ is for the SLs on p^s. Note that here, the reason why we do not consider the availabilities of the end nodes is that they are the mapped SNs and their availabilities have already been considered in Eq. (2). Then, the availability of VL e^r can be obtained as

$$A_{e^r} = 1 - \prod_{p^s \in f_{e^r}} (1 - A_{p^s}), \tag{4}$$

where f_{e^r} represents the set of substrate paths that VL e^r is embedded onto, i.e., all of them are node-disjointed.

C. Problem Description of A-SVNE

In A-SVNE, we need to properly allocate resources in the SNT to build a VNT such that the requirements on computing capacity, bandwidth, and availability are all satisfied. Therefore, we may adopt the protection strategy that allocates backup SNs and/or substrate paths to meet the availability requirements. The node mapping and link mapping are as follows.

- *Node Mapping*: Each VN needs to be embedded onto a working SN that can satisfy its computing capacity

[1]Note that in the rest of the paper, we use the preferred SN set instead of the location to represent the location constraint.

requirement. Meanwhile, its availability requirement has to be guaranteed as well. Hence, one or more backup SN(s) may be used if the working SN does not have sufficient availability, i.e., a VN can be mapped onto multiple SNs in the worst case.

- *Link Mapping*: For each VL, we perform availability-aware routing and wavelength assignment (RWA) in the SNT such that the requirements on bandwidth and availability are satisfied. Here, the availability-aware RWA leverages the multipath protection scheme in [28] to guarantee the availability. Note that for a VL, if any of its two end VLs have backup SN(s), we need to set up substrate path(s) between each pair of the mapped SNs, including the working-to-working, working-to-backup, and backup-to-backup connections.

D. Node Mapping

In a VNT, each VN is embedded onto one unique working SN, i.e., any two VNs in the same VNT cannot share the same SN(s). The set of mapped SNs for a VN $v^r \in V^r$ is denoted as m_{v^r}, which includes its working and backup SNs. Specifically, for v_1^r, $v_2^r \in V^r$, $v_1^r \neq v_2^r$,

$$m_{v_1^r} \subseteq \Phi(v_1^r), \qquad m_{v_2^r} \subseteq \Phi(v_2^r), \qquad m_{v_1^r} \cap m_{v_2^r} = \varnothing. \tag{5}$$

All the mapped SNs for the VN must satisfy the computing capacity requirement $c_{v^r}^r$, while they work together to meet the availability requirement $an_{v^r}^r$:

$$c_{v^r}^r \leq c_{v^s}^s, \quad \forall v^s \in m_{v^r}, \tag{6}$$

$$\prod_{v^s \in m_{v^r}} (1 - an_{v^s}^s) \leq 1 - an_{v^r}^r. \tag{7}$$

Here, we assume that the failures on SNs are independent.

E. Link Mapping

With the node mapping determined, each VL is embedded onto the substrate path(s) between the mapped SNs of its two end VNs. We use f_{e^r} to denote the set of substrate paths that VL $e^r = (v_1^r, v_2^r)$ is embedded onto:

$$f_{(v_1^r, v_2^r)} \subseteq \bigcup_{v_1^s \in m_{v_1^r}, v_2^s \in m_{v_2^r}} P_{(v_1^s, v_2^s)}^s.$$

All the substrate paths in f_{e^r} need to satisfy the bandwidth requirement $bw_{e^r}^r$ in number of wavelengths, and they work together to meet the availability requirement $al_{e^r}^r$:

$$\prod_{p^s \in f_{e^r}} (1 - A_{p^s}) \leq 1 - al_{e^r}^r. \tag{8}$$

The substrate paths in f_{e^r} should be node-disjoint except for the end nodes. For each VL, the substrate lightpaths should satisfy the wavelength continuity constraint [29], which means that they are set up all-optically end-to-end [30,31].

Figure 1(c) shows an example on the result of A-SVNE, including both node mapping and link mapping schemes. It can be seen that VN a is only embedded onto a working SN, i.e., SN 1, while VNs b and c are both embedded onto two SNs (working + backup). The working SNs of VNs b and c are SNs 2 and 4, respectively, and their backup SNs are SNs 5 and 6. For VL (a, b), we set up two node-disjoint substrate paths 1-2 and 1-3-5 to connect the working SN of VN a to both the working and backup SNs of VN b. For VL (b, c), the situation is more complicated. Basically, since both VNs have backup SNs, we need to establish four substrate paths for working-to-working, working-to-backup, and backup-to-backup connections. Hence, the working-to-working path is 2-4, the working-to-backup paths are 2-5-6 and 4-5, and the backup-to-backup path is 5-6.

F. Optimization Objective

To improve the acceptance ratio of VNTs in dynamic network operations, the A-SVNE needs to save substrate resources. Therefore, we design the cost of serving a VNT $G^r(V^r, E^r)$ as the total substrate resources that it consumes:

$$\text{cost}(G^v) = \sum_{v^r \in V^r} c_{v^r}^r \cdot |m_{v^r}| + \sum_{e^r \in E^r} \sum_{p^s \in f_{e^r}} bw_{e^r}^r \cdot |p^s|. \tag{9}$$

Then, we need to minimize the cost for each VNT. Figure 1(c) illustrates the A-SVNE result for embedding the VNT in Fig. 1(b) onto the SNT in Fig. 1(a).

IV. ILP Formulation for A-SVNE

In this section, we develop a path-based ILP formulation for the A-SVNE problem. Initially, we use Yen's K-shortest path algorithm to calculate K shortest paths for each node pair in the SNT. The ILP formulation is as follows.

Parameters:

- $G^r(V^r, E^r)$: VNT request.
- $c_{v^r}^r$: Computing resource requirement of VN $v^r \in V^r$.
- $an_{v^r}^r$: Availability requirement of VN $v^r \in V^r$.
- $bw_{e^r}^r$: Wavelength requirement of VL $e^r \in E^r$.
- $al_{e^r}^r$: Availability requirement of VL $e^r \in E^r$.
- $\Phi(v^r)$: Set of the preferred SNs for VN v^r.
- $G^s(V^s, E^s)$: Topology of SNT.
- W: Set of wavelengths on each SL.
- $w_{e^s}^s$: Set of available wavelengths on SL $e^s \in E^s$.
- $al_{e^s}^s$: Availability of SL $e^s \in E^s$.
- $c_{v^s}^s$: Available computing capacity on SN $v^s \in V^s$.
- $an_{v^s}^s$: Availability of SN $v^s \in V^s$.
- P: Set of precalculated substrate paths in SNT.
- A_p: Availability of substrate path p based on Eq. (3).
- s_p, d_p: Source and destination of substrate path p.
- P_{e^s}: Set of substrate paths that use SL e^s ($P_{e^s} \subset P$).

- P_{v^s}: Set of substrate paths that use SN v^s ($P_{v^s} \subset P$).
- $P^s_{v^s}, P^d_{v^s}$: Sets of substrate paths that start from and end at v^s ($P^s_{v^s}, P^d_{v^s} \subset P$), respectively.

Variables:

- π_{v,v^s}: Boolean variable that equals 1 if a VN $v \in V^r$ is mapped onto SN v^s as working, and 0 otherwise.
- π'_{v,v^s}: Boolean variable that equals 1 if a VN $v \in V^r$ is mapped onto SN v^s as backup, and 0 otherwise.
- $\xi_{e,p}$: Boolean variable that equals 1, if a VL $e \in E^r$ is mapped onto substrate path p, and 0 otherwise.
- $x_{p,k}$: Boolean variable that equals 1 if substrate path p uses the kth wavelength, and 0 otherwise.
- $y_{e^s,k}$: Boolean variable that equals 1 if the kth wavelength on SL $e^s \in E^s$ is available, and 0 otherwise.
- $\delta_{v^s,e}$: Boolean variable that equals 1 if VL $e \in E^r$ uses SN v^s as the intermediate node on the mapped paths, and 0 otherwise.

Objective: We extend Eq. (9) to get the total resource consumption of an A-SVNE solution and define the optimization objective as

$$\text{Minimize } \alpha \cdot \sum_{v^s \in V^s} \sum_{v \in V^r} (\pi_{v,v^s} + \pi'_{v,v^s}) \cdot c^r_v + \beta \cdot \sum_{e \in E^r} \sum_{p \in P} \xi_{e,p} \cdot |p| \cdot bw^r_e, \tag{10}$$

where α and β are the positive constants to normalize the consumptions of computing and wavelength resources, and $|p|$ returns the hop count of substrate path p.

Constraints:

1) Node Mapping Constraints:

To ensure that each VN v^r in the VNT is mapped onto one working SN:

$$\sum_{v^s \in \Phi(v^r)} \pi_{v^r,v^s} = 1, \quad \forall\, v^r \in V^r, \tag{11}$$

and to ensure that there may be backup SNs for VN v^r:

$$\sum_{v^s \in \Phi(v^r)} \pi'_{v^r,v^s} \geq 0, \quad \forall\, v^r \in V^r. \tag{12}$$

To ensure that the location constraint of VN v^r is satisfied:

$$\sum_{v^s \notin \Phi(v^r)} (\pi_{v^r,v^s} + \pi'_{v^r,v^s}) = 0, \quad \forall\, v^r \in V^r. \tag{13}$$

To ensure that SN v^s can carry at most one VN in V^r:

$$\sum_{v^r \in V^r} (\pi_{v^r,v^s} + \pi'_{v^r,v^s}) \leq 1, \quad \forall\, v^s \in V^s. \tag{14}$$

2) Node Capacity Constraint:

$$\sum_{v^r \in V^r} (\pi_{v^r,v^s} + \pi'_{v^r,v^s}) \cdot c^r_{v^r} \leq c^s_{v^s}, \quad \forall\, v^s \in V^s. \tag{15}$$

This ensures that if a VN is mapped onto an SN, its computing resource requirement cannot be larger than the available capacity of the SN.

3) Node Availability Constraint:

$$\sum_{v^s \in \Phi(v^r)} (\pi_{v^r,v^s} + \pi'_{v^r,v^s}) \cdot \log(1 - an^s_{v^s}) \leq \log(1 - an^r_{v^r}), \times \forall\, v^r \in V^r. \tag{16}$$

This ensures that the node mapping can satisfy the availability requirement of any $v^r \in V^r$. Note that the availability of a VN is calculated with Eq. (2), which is not linear, and we linearize it by using the logarithmic transformation.

4) Link Mapping Constraints:

To ensure that there will be paths between two mapped SNs for VNs v and u:

$$\sum_{p \in P^s_{v^s_1} \cap P^d_{v^s_2}} \xi_{(v,u),p} \geq \pi_{v,v^s_1} + \pi'_{v,v^s_1} + \pi_{u,v^s_2} + \pi'_{u,v^s_2} - 1, \quad \forall\, v^s_1, v^s_2 \in V^s, \quad \forall\, (v,u) \in E^r. \tag{17}$$

To ensure that if VNs v and u ($v, u \in V^r$) are mapped onto SNs, there should be substrate paths to connect the mapped SNs:

$$\sum_{p \in P^s_{v^s}} \xi_{(v,u),p} \geq \pi_{v_t,v^s} + \pi'_{v_t,v^s}, \quad \forall\, v^s \in V^s, \quad \forall\, v_t \in \{v,u\}, \quad \forall\, (v,u) \in E^r. \tag{18}$$

The following constraints ensure that each substrate path can carry at most one VL, and the end nodes of the path are the embedded SNs of the corresponding VNs on the VL:

$$\sum_{e \in E^r} \xi_{e,p} \leq 1, \quad \forall\, p \in P, \tag{19}$$

$$\xi_{(v,u),p} \leq \pi_{v,s_p} + \pi'_{v,s_p}, \quad \forall\, p \in P, \quad \forall\, (v,u) \in E^r, \tag{20}$$

$$\xi_{(v,u),p} \leq \pi_{u,d_p} + \pi'_{u,d_p}, \quad \forall\, p \in P, \quad \forall\, (v,u) \in E^r. \tag{21}$$

5) Link Availability and Bandwidth Constraints:

These ensure the wavelength continuity and bandwidth constraints on the substrate path chosen for VL $e \in E^r$:

$$\sum_{k \in W} x_{p,k} = bw^r_e \cdot \xi_{e,p}, \quad \forall\, p \in P, \quad \forall\, e \in E^r, \tag{22}$$

$$\sum_{p \in P_{e^s}} x_{p,k} \leq y_{e^s,k}, \quad \forall\, k \in W, \quad \forall\, e^s \in E^s, \tag{23}$$

$$y_{e^s,k} = 0, \quad \forall\, k \in W \backslash w^s_{e^s}, \quad \forall\, e^s \in E^s. \tag{24}$$

The next constraint ensures that if VL $e \in E^r$ is embedded, its availability requirement is satisfied. Here, we also linearize the availability calculation in Eq. (4) with the logarithmic transformation:

$$\sum_{p\in P}\log(1-A_p)\cdot\xi_{e,p}\le\log(1-al^r_e),\quad\forall\, e\in E^r. \tag{25}$$

6) Node-Disjoint Path Protection Constraints:

$$M\cdot\delta_{v^s,e}\ge\sum_{p\in P_{v^s}}\xi_{e,p}-\sum_{p\in P^s_{v^s}\cup P^d_{v^s}}\xi_{e,p},\quad\forall\, e\in E^r,\quad\forall\, v^s\in V^s, \tag{26}$$

$$\sum_{p\in P_{v^s}}\xi_{e,p}\le N\cdot(1-\delta_{v^s,e})+1,\quad\forall\, e\in E^r,\quad\forall\, v^s\in V^s. \tag{27}$$

These constraints ensure that an SN can only be included at most once as an intermediate node in the substrate path(s) of a VL $e=(v,u)\in E^r$, and if the SN is an end node, it can appear more times in the substrate path(s). Hence, the working and backup substrate paths of a VL are node-disjoint. M and N are two positive integers, where M guarantees the binary of $\delta_{v^s,e}$ and N ensures that the end node can have more paths.

The following constraints limit the ranges of the variables:

$$y_{e^s,k}\in\{0,1\},\quad\forall\, e^s\in E^s,\quad\forall\, k\in W, \tag{28}$$

$$\delta_{v^s,e}\in\{0,1\},\quad\forall\, v^s\in V^s,\quad\forall\, e\in E^r. \tag{29}$$

V. A-SVNE Algorithms

Due to its computational complexity, the ILP model is not suitable for large-scale A-SVNE problems. Hence, we propose several time-efficient heuristics in this section.

A. Node Mapping Strategy

We design two node mapping methods based on sequential node selection with AI and the maximum clique search in an AG, respectively.

1) AI-based Sequential Node Mapping: In sequential node mapping, we sort the VNs and SNs based on the AI of the incident VLs and substrate paths. We first define a weight for each VN based on the availability requirements of its incident VLs:

$$w_{v^r}=\prod_{e^r=(v^r,u^r)\in E^r}A_{e^r}. \tag{30}$$

Basically, the greater the weight is, the more availabilities a VN's incident VLs require. Therefore, we map all the VNs in descending order of their weights. We also define a weight for each SN to quantify its embedding potential, by considering the residual computing capacity and the availabilities of the shortest substrate paths to other SNs:

$$w_{v^s}=\frac{c^s_{v^s}}{C_{v^s}}\cdot\prod_{v^s\ne u^s,u^s\in V^s}A_{p^s_{v^s,u^s}}, \tag{31}$$

where C_{v^s} is the initial computing capacity on SN v^s.

The sequential node mapping strategy works as follows. We first sort the VNs and SNs in descending order of the weights in Eqs. (30) and (31), respectively. Then, we sequentially map the VN that has the highest weight to the SN(s) with the highest weight(s), where the VN's requirements on computing capacity, availability, and location are satisfied. Note that if the availability requirement of a VN cannot be satisfied with one SN, we can add one or more backup SNs. Finally, if any VN cannot be mapped successfully, the VNT is blocked. Otherwise, we update the SNT to consider the node mapping. The time complexity of this procedure is $O(|V^r|^2+|V^s|^2+|V^r|\cdot|\Phi(v^r)|)$.

2) MWMC-Based Node Mapping: In AI-based sequential node mapping, we have the dilemma that the mapped SNs for all the VNs cannot be determined simultaneously due to the nonlinearity in availability calculation. Therefore, we transform the node mapping into the MWMC problem [32] by constructing an AG, solving which allows us to map all the VNs at the same time. Specifically, for each VN, we find all the feasible SN sets in which the SNs satisfy the computing capacity requirement while the combined availability of them meets the availability requirement. Then, we insert a node in the AG to represent the feasible SN set. In the AG, two nodes are directly connected with a link if the SN sets that they represent are for two different VNs. Hence, the maximum clique in the AG is a feasible solution to the A-SVNE problem as long as the clique satisfies the constraint in Eq. (5).

Algorithm 1 gives the procedure of the AG construction for MWMC-based node mapping. Line 1 is for the initialization, where E^a and V^a are the link and node sets for the AG. The "for" loop that covers Lines 2–13 obtains the feasible SN sets Γ_{v^r} for each VN v^r if we only consider the constraints on computing capacity and location. Then, the "for" loop covering Lines 14–24 finds the feasible SN sets that satisfy the availability requirement. Note that in Lines 21–23, we apply an upper-limit K_c on the number of SN sets for each VN to control the AG's scale. Finally, Lines 26–30 build the AG.

Figure 2 shows an intuitive example of the AG construction. Here, the SNT is still the one in Fig. 1(a), while the VNT is briefly illustrated in the left subplot of Fig. 2. First, we mark its feasible SN set beside each VN in the VNT. For instance, VN a can be supported with SN 2 or SN 3 individually, or SNs 2 and 3 together. Then, with the SN sets of all the VNs, we build the AG, which includes seven nodes, and the nodes in the same row are for the same VN. The nodes are connected according to the principle we discussed above, i.e., two nodes are directly connected with a link if the SN sets that they represent are for two different VNs. After constructing the AG, we can obtain the maximum clique with $|V^r|=3$ nodes (i.e., the one marked with red links in Fig. 2), which corresponds to a feasible node mapping solution.

Theorem 1: Any maximal clique in the AG should contain $|V^r|$ nodes and is also the maximum clique in the AG.[2]

[2]A clique is a subset of nodes in an undirected graph such that its induced subgraph is complete. A maximal clique is a clique that cannot be extended by including one or more adjacent nodes. A maximum clique is a clique that includes the maximum number of nodes.

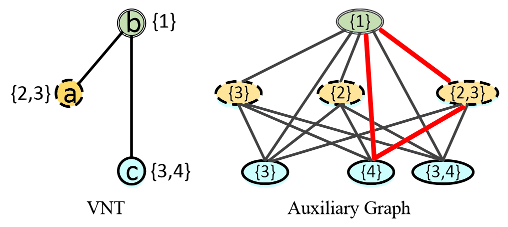


Fig. 2. Example on AG construction for MWMC-based node mapping.

Proof: If the maximum clique contains $|V^r| + 1$ nodes, it must include at least two feasible SN sets for the same VN. This means that the nodes representing two SN sets for the same VN are directly connected in the AG, which, however, violates our principle for constructing the AG. Hence, we can easily prove that the size of any maximal clique in the AG cannot be larger than $|V^r|$. Next, we prove that the size of any maximal clique in the AG cannot be smaller than $|V^r|$ using contradiction. Here, in order to explain clearly, we say a VN is covered if any of its SN sets is included in a maximal clique. Otherwise, we say the VN is left. If the size of a maximal clique is smaller than $|V^r| - 1$, there will be at least one VN left. However, according to our principle for constructing the AG, the nodes for the SN sets of different VNs are directly connected. Therefore, the SN sets of the VN that is left are connected to all the nodes in the maximal clique. This, however, is contradicted by the fact that the clique is maximal. Hence, the maximal clique must cover all the VNs, which in turn proves that in the AG, any maximal clique is also the maximum clique. ■

Note that finding the maximum clique in a general graph is an NP-complete problem, but a maximal clique can be obtained in linear time [32]. Therefore, this good property of AG ensures that our MWMC-based node mapping can be solved in a time-efficient way.

Algorithm 1 AG construction for MWMC-Based Node Mapping

```
1 $E^a = V^a = \varnothing$;
2. foreach $v^r \in V^r$ do
3   $\Phi^t_{v^r} = \Phi_{v^r}$;
4   foreach $v^s \in \Phi_{v^r}$ do
5     if $c^s_{v^s} < c^r_{v^r}$ then
6       $\Phi^t_{v^r} = \Phi^t_{v^r} \setminus \{v^s\}$;
7     end
8   end
9   if $|\Phi^t_{v^r}| = 0$ then
10    return (FAILURE);
11  end
12  store all the feasible SN sets in $\Phi^t_{v^r}$ in $\Gamma_{v^r}$;
13 end
14 foreach $v^r \in V^r$ do
15   foreach each SN set in $\Gamma_{v^r}$ do
16     if the SN set can satisfy the availability requirement
       $an^r_{v^r}$ then
17       insert a node $v^a$ in $V^a$ for the SN set;
18       store the SN set in $\Psi_{v^a}$;
19       $\Omega_{v^r} = \Omega_{v^r} \cup \{v^a\}$;
20     end
21     if $|\Omega_{v^r}| > K_c$ then
22       break;
23     end
24   end
25 end
26 foreach $v^r_1 \in V^r$ do
27   foreach $v^r_2 \in V^r, v^r_1 \neq v^r_2$ do
28     $E^a = \Omega_{v^r_1} \times \Omega_{v^r_2}$;
29   end
30 end
31 return $(G^a)$;
```

When the network resources are abundant, we may obtain multiple maximal cliques. Therefore, we define a weight w_{v^a} for each node v^a in the AG $G^a(V^a, E^a)$ to quantify its embedding potential by considering the availability on the related links. Here, we assume that v^a and u^a are directly connected in the AG, i.e., $(v^a, u^a) \in E^a$. Then, Ψ_{v^a} and Ψ_{u^a} denote the SN sets that v^a and u^a represent, respectively:

$$w_{v^a} = \frac{\min_{p^s \in (\Psi_{v^a} \times \Psi_{u^a})}(A_{p^s})}{|\Psi_{v^a}|}, \tag{32}$$

where $\Psi_{v^a} \times \Psi_{u^a}$ represents the set of substrate paths that connect the SNs in Ψ_{v^a} and Ψ_{u^a}, and A_{p^s} is the availability of substrate path p_s, which can be calculated with Eq. (3). Here, the numerator is the minimum availability on all the shortest substrate paths that end at SNs in Ψ_{v^a}, while the denominator is the number of SNs in Ψ_{v^a}. Hence, a larger weight w_{v^a} means that the resulting node mapping involves a relatively small number of SNs and the availabilities on the feasible substrate paths are high, which is exactly what we want for efficient node mapping.

Combined with VLs, we design a method to solve MWMC-based node mapping quickly. Algorithm 2 shows the details. Basically, we first sort the VLs in descending order of their required availabilities, and then we deal with the VNs on each VL by choosing the node with the maximum weight from the corresponding SN sets in the AG. Note that in order to satisfy the constraint in Eq. (5), Line 6 checks the compatibility among the selected node and those in the clique already before updating the clique, i.e., ensuring that any two VNs in the same VNT cannot share the same SN(s).

In MWMC-based node mapping, the AG construction has the complexity of $O(|V^r| \cdot |\Phi(v^r)| + K_c \cdot |V^r|)$, and the complexity of finding a MWMC is $O(2 \cdot K_c \cdot |E^r|)$.

Algorithm 2 MWMC-Based Node Mapping

```
  input: $G^r$, $G^a$, $\Omega$
  output: Maximum-weight maximum clique $MC$
1   $MC = R_n = \varnothing, R_c = V^a$;
2   foreach $e^r \in E^r$ in non-increasing order of $A_{e^r}$ do
3     foreach $v^r \in e^r$ do
4       if $v^r \bigcap R_n = \varnothing$ then
5       $\widehat{R_c} = \Omega_{v^r} \cap R_c$;
6       remove $v^a$ from $\widehat{R_c}$ that is incompatible with $MC$;
```

```
7          if $\widehat{R_c} = \varnothing$ then
8            return (FAILURE)
9          end
10         select $v^a$ in $\widehat{R_c}$ with maximum weight;
11         $R_n = R_n \cup v^r$;
12         $MC = MC \cup v^a$;
13         $R_c = R_c \cap \{v_1^a : (v^a, v_1^a) \in E^a\}$;
14       end
15     end
16 end
```

B. Link Mapping Strategy

With the node mapping determined, we solve the link mapping in a straightforward way. Specifically, the VLs are sorted in descending order of their availability requirements. For each VL in E^r, we try to find the substrate paths to connect the mapped SNs of its two end-VNs and assign the wavelength resource using first fit [33], to satisfy the bandwidth and availability requirements. Actually, based on the node mapping results, there are three cases for the link mapping that we use complete bipartite graphs to explain, as shown in Fig. 3. Specifically, we may also need to establish substrate path(s) to connect the backup SN(s) to ensure continuous communication when the working SN fails. For the cases in Figs. 3(a) and 3(b), since at most one end VN has backup SNs, we only set up the substrate paths for the working–working and working–backup connections. In Fig. 3(c), for the case that both end VNs have working and backup SNs, three types of connections, i.e., working–working, working–backup, and backup–backup, should be considered, which may cost a lot of redundant resources. Note that here we do not set up dedicated substrate path(s) to directly connect a working SN to its backup SN(s) to save substrate resources. Actually, the link mapping mentioned above does ensure that the communication between them can be realized by going through another working or backup SN with two substrate paths. For instance, in Fig. 3(b), the communication between the working and backup SNs in the left column can be realized by using the working SN in the right column as a relay node and leveraging the two substrate paths in the figure.[3]

With this link mapping scheme, we should ensure that the combined availability of all the substrate paths that are used to support a VL should satisfy the VL's availability requirement, and hence we change Eq. (4) to

$$A_{p^s} = 1 - (1 - A_{e^r})^{\frac{1}{p_n}}, \tag{33}$$

where, p_n is the number of substrate paths used to support VL e^r. Then, the availability on each substrate path should not be less than A_{p^s}. To make the protection valid, the substrate paths should be node-disjoint except for their end SNs, i.e., the source and destination SNs.

[3]It is known that the bandwidth required for periodic status backup between working and backup SNs is much smaller than that for the live communication between two working SNs. Hence, we can assume that the bandwidth requirement of each VL already reserves a small portion for status backup.

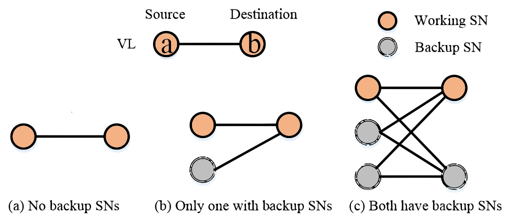


Fig. 3. Three cases for link mapping.

According to the availability requirement of the VL, we establish one or more node-disjoint substrate paths for each edge shown in Fig. 3 to accomplish the link mapping. Basically, we pre-calculate K_p shortest paths and check whether a single path can satisfy the availability requirement. If not, we add node-disjoint paths to improve the availability until the requirement can be satisfied. The time complexity of the link mapping is $O(K_p \cdot |E^r| \cdot (|E^s| + |V^s| \cdot \log(|V^s|))$.

VI. Performance Evaluation

A. Simulation Setup

We evaluate the performance of the proposed algorithms with the six-node topology shown in Fig. 1(a) and the 28-node U.S. backbone topology in [34]. We assume that each substrate component has an availability randomly chosen from [0.995, 0.999, 0.9995, 0.9999]. The initial resources in the SNT are shown in Table I. In the simulations, the failures on substrate components are generated according to their availabilities dynamically, and the MTTR follows the exponential distribution with an average value within [5,10] time units [25]. The VNTs are randomly generated with the GT-ITM tool, and their parameters are also shown in Table I. We consider dynamic network operation and assume that the VNTs come in following the Poisson process with an average rate of λ per time unit, while the holding time follows a negative exponential distribution with an average of $\frac{1}{\mu}$ time units. Hence, the traffic load can be quantified as $\frac{\lambda}{\mu}$ in erlangs. Here, we assume that a time unit is an hour. The availability requirements for VLs and VNs can

TABLE I
Simulation Parameters

	Six-Node	U.S. Backbone
# of SNs	6	28
# of SLs	10	49
SN computing capacity	50 units	100 units
# of wavelengths on each SL	50	100
VN's computing requirement	[1,2] units	[1,3] units
VL's wavelength requirement	[1,2]	[1,3]
# of VNs in each VNT	[2,3]	[2,5]
VN's connectivity rate	0.5	0.5
# of candidate SNs per VN	4	6
Average lifetime	168 time units	168 time units

be chosen from [0.99, 0.995, 0.999, 0.9995, 0.9999], with a distribution of 20:10:5:2:1. The algorithms are implemented in MATLAB, and we solve the ILP with GLPK. In Algorithm 1, K_c is set as 10, while the number of pre-calculated shortest substrate paths in the link mapping, i.e., K_p, equals 10.

B. Performance Metrics

We evaluate the proposed A-SVNE algorithms with three performance metrics as follows:

- *Availability gap*: the difference between the required and provided availabilities for each virtual component.
- *Blocking probability*: the ratio of blocked to total number of VNT requests in each simulation.
- *Penalty*: we adopt the SLA violation penalty in [25] and define the penalty for a VNT in our simulations as

$$P = \sum_i S(a) \cdot \mathrm{Re}\,q(i) \cdot \frac{T_{\text{downtime}}}{T}, \tag{34}$$

where i represents a virtual component in the VNT, $\mathrm{Re}\,q(i)$ is its requirement on computing capacity or bandwidth resources, $S(a)$ is the penalty for violating the availability requirement, which is illustrated in Table II, T_{downtime} is the downtime caused by substrate failures, and T is the holding time of the VNT.

C. Benchmarks and Algorithm Descriptions

We implement the algorithms as benchmarks. In [16], the authors focused on solving the A-SVNE problem with explicit availability requirements on VLs, and we denote their algorithm as R-ILP in our simulations. Basically, R-ILP uses the sequential node selection that sorts the VNs according to their required computing capacities, quantifies an SN's embedding potential with the availabilities on all the incident links, and adopts an ILP to obtain the optimal link mapping results. Here, in order to make the algorithm work, we keep the assumption in [16] that SNs are working all the time. The work in [17] is called SVNE in the simulations, and focuses on protecting the VNTs against single substrate component failures. We name our A-SVNE algorithms according to their node mapping strategies, i.e., AI and MWMC for sequential node mapping based on AI and the MWMC-based approach, respectively. To make a fair comparison with R-ILP, we also modify our algorithms as AI-NN and MWMC-NN to obey the assumption that SNs are working all the time. For the six-node topology, we also simulate our ILP for A-SVNE discussed in Section IV.

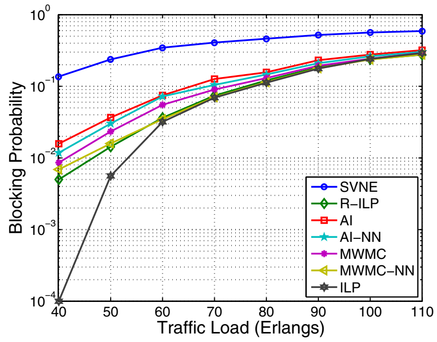
TABLE II
PENALTY FOR DIFFERENT AVAILABILITIES

Availability	0.99	0.995	0.999	0.9995	0.9999
SLA penalty [$S(a)$]	1	2	5	10	20

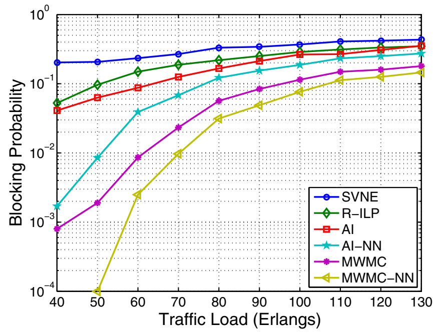
D. Performance Comparisons

1) Blocking Probability: Figure 4 shows the simulation results for blocking probability in the two topologies. We simulate an algorithm for 50,000 time units in each run and average the results from five runs to obtain each data point. Due to its complexity, we only show the results from ILP in the six-node topology. In Fig. 4(a), ILP achieves the best blocking performance among the algorithms, which verifies that it can optimize the resources allocated to each VNT. MWMC-NN performs similarly to R-ILP, because we choose the SN pair for each VL with the highest availability in the node mapping, and make sure that the paths between the SN pair are relatively short. When we need to consider the availabilities of SNs, the blocking probability increases and MWMC and AI achieve worse blocking performance than MWMC-NN and AI-NN, respectively. It is also interesting to notice that among the algorithms that use sequential node mapping, R-ILP outperforms AI-NN because it can optimize the resource allocation in link mapping, while SVNE performs the worst due to consuming too many redundant resources.

We then evaluate the algorithms in the U.S. backbone topology. In Fig. 4(b), MWMC-NN outperforms all the other algorithms and MWMC achieves the second-best blocking performance. This is because the node mapping in MWMC-NN and MWMC takes the availabilities of substrate paths into consideration when building the AG. We still observe that MWMC and AI perform worse than MWMC-NN and AI-NN, respectively. It is interesting to observe that AI-NN achieves better blocking performance than R-ILP this time. This is because as the size of the network increases, the



(a) Six-node topology.



(b) US Backbone.

Fig. 4. Results for request blocking probability.

availabilities of substrate paths decrease on average, and hence we should pay more attention to the availabilities in A-SVNE. As the main differences between AI-NN and R-ILP are the sequence of the node mapping and the weights of SNs, the results confirm the effectiveness of the weights that we design for the VNs and SNs. The algorithms that use sequential node mapping perform worse than the MWMC-based ones because they do not consider the information on VLs in node mapping, which could result in improperly mapped SNs that do not support substrate paths with the required availabilities. SVNE still performs the worst.

Table III shows the results for the distribution of required SNs per VN from MWMC and AI, where we consider different distribution ratios of the availability requirements for virtual components. Table IV shows the three different distributions, and we denote them with the notations l, m, and h, respectively. Here, we sample the results when the traffic load is 50 erlangs, since with this traffic load, the blocking probability from the algorithms is within $[10^{-3}, 10^{-1}]$, which is the reasonable range for practical network operation. The results indicate that most of the VNs are only embedded onto a single SN. This verifies the effectiveness of our proposed A-SVNE algorithms. Specifically, as they can determine the protection schemes for the VNs efficiently based on their actual resource and availability requirements, only a relatively small portion of the VNs need backup SNs. Hence, we can see that the additional complexity due to the one-to-many node mapping is controllable in our A-SVNE.

2) Availability Gap: A high acceptance ratio with guaranteed availability is important for A-SVNE. The availability gap measures the difference between the required and provided availabilities for each virtual component. Specifically, the provided availabilities are obtained by generating dynamic failures on the substrate components according to their availabilities. Table V shows the availability gap for VNs from the algorithms. Here, we collect the data with the simulation scenario that has the traffic load at 100 erlangs in the U.S. backbone topology. The results indicate that since R-ILP, AI-NN, and MWMC-NN do not consider the SNs' availabilities, they cannot guarantee the required availability for the VNs that ask for relatively high availabilities. Therefore, they can provide a negative availability gap, i.e., the provided availability is insufficient when the required availability is higher than 0.995. Meanwhile, due to the fact that SVNE uses dedicated protection for single substrate component failures, it always achieves the highest positive availability gap, which means that it overprovisions too much. Since AI and MWMC consider the availability requirement in the most effective way, they also always provide the positive availability gap and can minimize the gap to avoid overprovisioning too much.

TABLE V
AVAILABILITY GAP PROVIDED FOR VNS ($\times 10^{-3}$)

Required Availability	0.99	0.995	0.999	0.9995	0.9999
SVNE	9.9966	4.9967	0.9966	0.4969	0.0967
R-ILP	7.7536	2.4771	−0.6122	−2.254	−2.5967
AI	8.9533	3.9536	0.5398	0.2135	0.0014
AI-NN	8.4820	3.4769	−0.5451	−0.9166	−1.4416
MWMC	8.8444	3.8340	0.4748	0.1958	0.0115
MWMC-NN	8.8643	3.8366	−0.1204	−0.6195	−1.0035

Table VI shows the percentage of the VLs whose availability requirements are satisfied. We can see that MWMC and AI still always provide the guaranteed availability for VLs, while without considering the availabilities of SNs, the others cannot guarantee the required availabilities for VLs. Due to the fact that multiple substrate component failures can happen simultaneously in the simulations, even SVNE uses dedicated protection, but the scheme still cannot satisfy the availability requirements of 18.72% of VLs when the required availability is 0.9999. MWMC-NN can ensure the availability for 90.37% of VLs, and AI-NN can satisfy a higher percentage of VLs than R-ILP because we consider the availabilities of substrate paths in the weights of SNs.

3) Penalty: We also try to estimate the impacts on the VNTs from the failures of SNs and SLs. Figure 5 shows the results for the SLA violation penalty when the value of MTTR changes in the U.S. backbone topology with the traffic load at 100 erlangs. SVNE achieves the lowest

TABLE III
DISTRIBUTION OF REQUIRED SNS PER VN AT 50 ERLANGS (%)

Required SNs per VN		1	2
h	MWMC	95.56	4.44
	AI	94.78	5.22
m	MWMC	98.50	1.50
	AI	97.88	2.12
l	MWMC	100	0
	AI	99.98	0.02

TABLE IV
AVAILABILITY DISTRIBUTIONS

Notation	Distribution of Availability Requirements (0.99 : 0.995 : 0.999 : 0.9995 : 0.9999)
l	20 : 10 : 5 : 2 : 1
m	5 : 4 : 3 : 2 : 1
h	1 : 1 : 1 : 1 : 1

TABLE VI
PERCENTAGE OF VLS WITH REQUIRED AVAILABILITIES (%)

Required Availability	0.99	0.995	0.999	0.9995	0.9999
SVNE	100	100	100	99.55	81.28
R-ILP	59.62	34.24	82.56	86.16	42.42
AI	100	100	100	100	100
AI-NN	74.18	79.19	92.41	88.94	74.64
MWMC	100	100	100	100	100
MWMC-NN	89.99	91.49	96.77	90.12	90.37

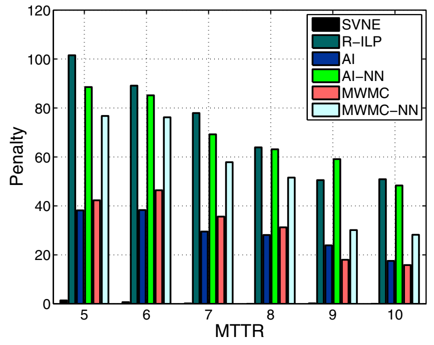


Fig. 5. Penalty for different MTTR values in U.S. backbone at 100 erlangs.

penalty because according to the availabilities of the substrate components, most of the failures are on a single substrate component. Compared with MWMC-NN, MWMC achieves lower penalties, since it considers the availabilities of SNs, while a similar trend applies to AI-NN and AI. The penalty results from AI-NN and R-ILP are comparable. In general, we notice a trade-off between the blocking probability and penalty, which can be controlled with the availability requirements. Specifically, if we increase the availability requirements of virtual components, the blocking probability will increase too, since it would be more difficult to embed a VNT, but at the same time, the penalty will decrease, since the higher availabilities make the VNT more survivable.

4) Availability Value Influence: Finally, we study the influence of availability requirements. Basically, we use the U.S. backbone topology, fix the traffic load at 50 erlangs, and change the distribution ratio of the availability requirements for virtual components as in Table IV. In Fig. 6, we can see that when we increase the availability requirements of the virtual components, the blocking probabilities for all the algorithms also increase. MWMC-NN still always has the best blocking performance. Also, AI-NN still outperforms R-ILP for all the distributions, which once again verifies that the weights that we define for the SNs and VNs are effective. The blocking probabilities for MWMC and AI grow rapidly with the availability requirements, because substrate paths with sufficient availabilities become more and more difficult to find.

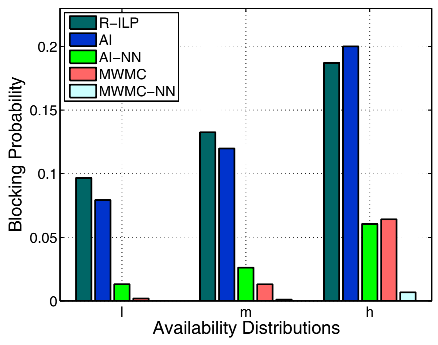


Fig. 6. Blocking probability in U.S. backbone at 50 erlangs.

VII. Conclusion

This paper studied the A-SVNE problem in optical inter-DC networks that use WDM. With A-SVNE, we tried to satisfy the availability requirement of each virtual component in a VNT. We first analyzed the availability of a virtual component based on the availabilities of the substrate link(s) and node(s). Then, we formulated an ILP model to solve the A-SVNE problem by optimizing the node and link mapping jointly. Also, we designed several efficient heuristics that leveraged two node mapping strategies: one was sequential selection with efficient weights for VNs and SNs using the AI, while the other used AGs and transformed the problem into the MWMC problem. Availability on a substrate path was considered in the construction of an AG. An efficient approach combined with VLs was designed to solve the MWMC problem. Finally, we used extensive simulations to compare the proposed A-SVNE algorithms with existing ones in terms of the blocking probability, availability gap, and penalty due to SLA violations, and the results indicated that our algorithms performed better.

Acknowledgment

This work was supported in part by the NCET program under Project NCET-11-0884, the NSFC Project 61371117, the Fundamental Research Funds for the Central Universities (WK2100060010), and the Strategic Priority Research Program of the CAS (XDA06011202). It was also supported by the State Key Laboratory of Advanced Optical Communication Systems and Networks in China.

References

[1] P. Lu, L. Zhang, X. Liu, J. Yao, and Z. Zhu, "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, M. Funabashi, Z. Pan, B. Xiang, L. Paraschis, and S. Yoo, "Jitter and amplitude noise accumulations in cascaded all-optical regenerators," *J. Lightwave Technol.*, vol. 26, pp. 1640–1652, June 2008.

[3] C. Develder, M. De Leenheer, B. Dhoedt, M. Pickavet, D. Colle, F. De Turck, and P. Demeester, "Optical networks for grid and cloud computing applications," *Proc. IEEE*, vol. 100, pp. 1149–1167, May 2012.

[4] L. Zhang and Z. Zhu, "Spectrum-efficient anycast in elastic optical inter-datacenter networks," *Opt. Switch. Netw.*, vol. 14, pp. 250–259, Aug. 2014.

[5] M. Chowdhury and R. Boutaba, "A survey of network virtualization," *Comput. Netw.*, vol. 54, pp. 862–876, Apr. 2010.

[6] Y. Jin, Y. Wen, Q. Chen, and Z. Zhu, "An empirical investigation of the impact of server virtualization on energy efficiency for green data center," *Comput. J.*, vol. 56, pp. 977–990, Aug. 2013.

[7] 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, Feb. 2012.

[8] L. Gong, W. Zhao, Y. Wen, and Z. Zhu, "Dynamic transparent virtual network embedding over elastic optical infrastructures," in *Proc. IEEE Int. Conf. on Communications (ICC)*, June 2013, pp. 3466–3470.

[9] G. Long, Y. Wen, Z. Zhu, and T. Lee, "Revenue-driven virtual network embedding based on global resource information," in *Proc. of GLOBECOM*, Dec. 2013, pp. 2294–2299.

[10] L. Gong, Y. Wen, Z. Zhu, and T. Lee, "Virtual optical network embedding (VONE) over elastic optical networks," *J. Lightwave Technol.*, vol. 32, pp. 450–460, Feb. 2014.

[11] L. Gong, Y. Wen, Z. Zhu, and T. Lee, "Toward profit-seeking virtual network embedding algorithm via global resource capacity," in *Proc. INFOCOM*, 2014.

[12] W. Fang, M. Lu, X. Liu, L. Gong, and Z. Zhu, "Joint defragmentation of optical spectrum and IT resources in elastic optical datacenter interconnections," *J. Opt. Commun. Netw.*, vol. 7, pp. 314–324, Mar. 2015.

[13] J. Yao, P. Lu, L. Gong, and Z. Zhu, "On fast and coordinated data backup in geo-distributed optical inter-datacenter networks," *J. Lightwave Technol.*, vol. 33, pp. 3005–3015, July 2015.

[14] "Calculating the cost of data center outages" [Online]. Available: http://www.emersonnetworkpower.com/fr-EMEA/Latest-Thinking/white-papers/Pages/calculating-the-cost.aspx.

[15] H. Jiang, L. Gong, and Z. Zhu, "Efficient joint approaches for location-constrained survivable virtual network embedding," in *Proc. GLOBECOM*, Dec. 2014, pp. 1810–1815.

[16] S. Herker, X. An, W. Kiess, and A. Kirstadter, "Path protection with explicit availability constraints for virtual network embedding," in *Proc. PIMRC*, 2013, pp. 2978–2983.

[17] Z. Ye, A. Patel, P. Ji, and C. Qiao, "Survivable virtual infrastructure mapping with dedicated protection in transport software-defined networks," *J. Opt. Commun. Netw.*, vol. 7, pp. A183–A189, Feb. 2015.

[18] A. Jarray, Y. Song, and A. Karmouch, "p-Cycle-based node failure protection for survivable virtual network embedding," in *Proc. IFIP*, 2013.

[19] Q. Hu, Y. Wang, and X. Cao, "Survivable network virtualization for single facility node failure: A network flow perspective," *Opt. Switch. Netw.*, vol. 10, pp. 406–415, Nov. 2013.

[20] B. Guo, C. Qiao, J. Wang, H. Yu, Y. Zuo, J. Li, Z. Chen, and Y. He, "Survivable virtual network design and embedding to survive a facility node failure," *J. Lightwave Technol.*, vol. 32, pp. 483–493, Feb. 2014.

[21] H. Yu, C. Qiao, V. Anand, X. Liu, H. Di, and G. Sun, "Survivable virtual infrastructure mapping in a federated computing and networking system under single regional failures," in *Proc. GLOBECOM*, 2010.

[22] X. Chen, F. Ji, and Z. Zhu, "Service availability oriented p-cycle protection design in elastic optical networks," *J. Opt. Commun. Netw.*, vol. 6, pp. 901–910, Oct. 2014.

[23] X. Chen, M. Tornatore, F. Ji, W. Zhou, C. Chen, D. Hu, S. Zhu, L. Jiang, and Z. Zhu, "Flexible availability-aware differentiated protection in software-defined elastic optical networks," *J. Lightwave Technol.*, vol. 33, pp. 3872–3882, Sept. 2015.

[24] J. Zhang, K. Zhu, H. Zang, N. Matloff, and B. Mukherjee, "Availability-aware provisioning strategies for differentiated protection services in wavelength-convertible WDM mesh networks," *IEEE/ACM Trans. Netw.*, vol. 15, pp. 1177–1190, Oct. 2007.

[25] M. Tornatore, F. Dikbiyik, and B. Mukherjee, "(3W-) availability-aware routing in optical WDM networks: When, where and at what time," in *Proc. ICTON*, 2011.

[26] S. Herker, W. Kiess, X. An, and A. Kirstadter, "On the trade-off between cost and availability of virtual networks," in *Proc. IFIP*, 2014.

[27] Q. Zhang, M. Zhani, M. Jabri, and R. Boutaba, "Venice: Reliable virtual data center embedding in clouds," in *Proc. INFOCOM*, 2014, pp. 289–297.

[28] X. Chen, S. Zhu, D. Chen, S. Hu, C. Li, and Z. Zhu, "On efficient protection design for dynamic multipath provisioning in elastic optical networks," *Proc. ONDM*, May 2015, pp. 251–256.

[29] Z. Zhu, X. Chen, F. Ji, L. Zhang, F. Farahmand, and J. Jue, "Energy-efficient translucent optical transport networks with mixed regenerator placement," *J. Lightwave Technol.*, vol. 30, pp. 3147–3156, Oct. 2012.

[30] J. Cao, M. Jeon, Z. Pan, Y. Bansal, Z. Wang, Z. Zhu, V. Hernandez, J. Taylor, V. Akella, and S. Yoo, "Error-free multi-hop cascaded operation of optical label switching routers with all-optical label swapping," in *Optical Fiber Communication Conf.*, Mar. 2003, pp. 1–3.

[31] M. Funabashi, Z. Zhu, Z. Pan, B. Xiang, L. Paraschis, D. Harris, and S. Yoo, "Cascadability of optical 3R regeneration for NRZ format investigated in recirculating loop transmission over field fibers," *IEEE Photon. Technol. Lett.*, vol. 18, pp. 2081–2083, Oct. 2006.

[32] D. West, *Introduction to Graph Theory.* Prentice-Hall, 2001.

[33] W. Lu and Z. Zhu, "Dynamic service provisioning of advance reservation requests in elastic optical networks," *J. Lightwave Technol.*, vol. 31, pp. 1621–1627, May 2013.

[34] Y. Xiong and L. Mason, "Restoration strategies and spare capacity requirements in self-healing ATM networks," *IEEE/ACM Trans. Netw.*, vol. 7, pp. 98–110, Feb. 1999.