Virtual Network Survivability Through Joint Spare Capacity Allocation and Embedding

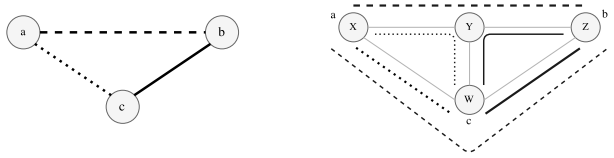
Nashid Shahriar, *Student Member, IEEE*, Shihabur Rahman Chowdhury, *Student Member, IEEE*, Reaz Ahmed, Aimal Khan, Siavash Fathi, Raouf Boutaba, *Fellow, IEEE*, Jeebak Mitra, and Liu Liu

***Abstract*— A key challenge in network virtualization is to efficiently map a virtual network (VN) on a substrate network (SN), while accounting for possible substrate failures. This is known as the survivable VN embedding (SVNE) problem. The state-of-the-art literature has studied the SVNE problem from infrastructure providers' (InPs') perspective, i.e., provisioning backup resources in the SN. A rather unexplored solution spectrum is to augment the VN with sufficient spare backup capacity to survive substrate failures and embed the resulting VN accordingly. Such augmentation enables InPs to offload failure recovery decisions to the VN operator, thus providing more flexible VN management. In this paper, we study the problem of jointly optimizing spare capacity allocation in a VN and embedding the VN to guarantee full bandwidth in the presence of multiple substrate link failures. We formulate the optimal solution to this problem as a quadratic integer program that we transform into an integer linear program. We also propose a heuristic algorithm to solve larger instances of the problem. Based on analytical study and simulation, our key findings are: 1) provisioning shared backup resources in the VN can yield ∼33% more resource efficient embedding compared to doing the same at the SN level and 2) our heuristic allocates ∼21% extra resources compared to the optimal, while executing several orders of magnitude faster.**

***Index Terms*— Network virtualization, survivability, virtual network embedding, spare capacity, joint optimization.**

I. Introduction

INFRASTRUCTURE providers (InPs), such as data center network operators, Internet service providers, and transport network operators are leveraging network virtualization (NV) to offer slices of their networks to service providers (SPs) [1], [2]. It enables InPs to better utilize their substrate network (SN) and to open new revenue streams. More recently, NV has been gaining further traction as one of the cornerstones of 5G mobile networks [3], [4]. However, the benefits of NV come with additional resource management challenges such as efficiently mapping the virtual nodes and links of a virtual network (VN) request onto substrate nodes and paths, respectively. This is known as the VN embedding (VNE) problem [5]. If a VNE solution does not take possible substrate failures into account, then such failures can result in degraded quality of service (QoS) for VNs, leading to service level agreement (SLA) violations. A VN embedding that can survive substrate failures is known as the survivable VNE (SVNE) [6], and has received significant attention from the research community [7]–[15].



(a) VN topology (b) VNE with protection in SN

Fig. 1. Survivability at SN ($WXYZ$) layer. Primary (or backup) embedding of a virtual link is shown with thick (thin) lines. For example, virtual link (a, b) has a primary and backup embedding of $\{(X, Y), (Y, Z)\}$ and $\{(X, W), (W, Z)\}$, respectively.

The SVNE research literature focuses primarily on protection (*i.e.*, pro-actively provisioning backup during embedding) and restoration (*i.e.*, reactively take action after failure) methods [16]. In this paper, we focus on the former, *i.e.*, protection mechanism for SVNE, which is usually faster than restoration approaches [17]. When a VN is embedded with SN layer protection, it is the InP's responsibility to handle substrate resource failures. As an illustrative example, consider the embedding of VN abc on SN $WXYZ$ with SN layer survivability in Fig. 1, where a, b, and c are virtual nodes and W, X, Y, and Z are substrate nodes. In Fig. 1, if the substrate link (X, Y) fails, the InP needs to reroute the affected traffic on the virtual link (a, b) to its backup embedding in the SN, i.e., $\{(X, W), (W, Z)\}$. A key challenge in providing protection is to efficiently utilize resources, since backup resources remain idle until a failure has occurred. Protection based SVNE approaches have adopted different techniques to increase resource efficiency including dedicated backup capacity allocation [14], [15], backup resource sharing [12], [13], [18], multi-path embedding [10], [11], providing weaker forms of survivability [9], and so on.

A rather unexplored spectrum in SVNE is to provide protection at the VN layer, much like providing survivability

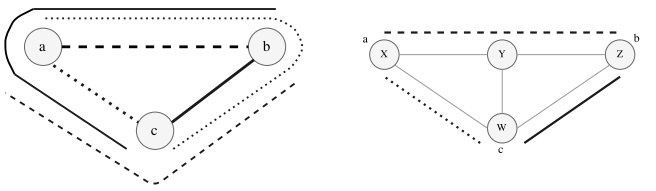
Manuscript received October 4, 2017; revised February 5, 2018; accepted February 27, 2018. Date of publication March 12, 2018; date of current version May 21, 2018. This work was supported in part by Huawei Technologies and in part by an NSERC Collaborative Research and Development Grant. This work benefited from the use of the CrySP RIPPLE Facility at the University of Waterloo. *(Corresponding author: Nashid Shahriar.)*

N. Shahriar, S. R. Chowdhury, R. Ahmed, A. Khan, S. Fathi, and R. Boutaba are with the David R. Cheriton School of Computer Science, University of Waterloo, Waterloo, ON N2L 3G1, Canada (e-mail: nshahria@uwaterloo.ca; sr2chowdhury@uwaterloo.ca; reaz.ahmed93@gmail.com; a273khan@uwaterloo.ca; s5fathi@uwaterloo.ca; rboutaba@uwaterloo.ca).

J. Mitra is with the Huawei Technologies Canada Research Center, Ottawa, ON K2K 3J1, Canada (e-mail: jeebak.mitra@huawei.com).

L. Liu is with Huawei Technologies Co., Ltd., Chengdu, 611731, China (e-mail: liuliu1@huawei.com).

Digital Object Identifier 10.1109/JSAC.2018.2815430



(a) VN with protection (b) VN Embedding on SN $(WXYZ)$

Fig. 2. Survivability at VN (abc) layer. Backup path of a virtual link in a VN is shown with a thin line having same dash pattern. For example, virtual link (a, b) has a backup virtual path $\{(a, c), (c, b)\}$ and an embedding $\{(X, Y), (Y, Z)\}$.

at the upper layer (*e.g.*, Internet Protocol (IP)) of a multi-layer network (*e.g.*, IP over optical such as IP-over-Wavelength Division Multiplexing (WDM) network) [19]–[21]. Fig. 2 shows a VN embedding with protection at the VN layer. With this type of protection, InPs can offload some of the failure management tasks to SPs by augmenting VNs with sufficient spare capacity for backup and embedding the VNs in a way that primary and backup VN resources are not affected by the same substrate resource failure. When a substrate resource fails, it is the SP's responsibility to reroute the affected traffic to the pre-allocated backup resources within the VN. For example, in Fig. 2, when virtual link (a, b) is affected by the failure of substrate link (X, Y), the SP reroutes traffic between a and b along the backup virtual path $\{(a, c), (b, c)\}$.

The motivation for providing survivability at the VN level is to shift control to SPs, as anticipated in future Transport Software Defined Networks (T-SDNs) [22]. T-SDNs are the next generation of transport networks that leverage Software Defined Networking (SDN) technologies and provide full fledged Virtual Transport Networks (VTNs) instead of traditional end-to-end connectivity to SPs [2], [23]. In essence, an SP can manage its VTN in the same way as managing its own transport network, and can deploy its own routing, traffic engineering, and failure management solutions [22]. Consequently, with VN layer survivability, the SP can offer different classes of service with differing survivability guarantees to its customers [24]. For instance, the spare backup capacity allocated to the VN of Fig. 2(a) can also be used for carrying traffic of a service from best-effort protection class during normal operation, while the same capacity can be used to guarantee protection for a high priority service when substrate resources fail. Such different classes of service and utilization of resources could not be easily achieved in a VN with SN level survivability, since the spare capacity at the SN layer is usually transparent to the SP [25].

Another benefit of providing survivability at the VN layer is that it yields more resource efficient embedding compared to providing the same level of survivability at the SN layer [21]. The intuition behind such claim is that the former can offer more opportunities for spare capacity sharing than the latter. In the case of SN layer protection in Fig. 1(b), only the backup capacity on the substrate link (Y, W) can be shared among the backup paths $\{(X, Y), (Y, W)\}$ and $\{(Z, Y), (Y, W)\}$ due to higher path diversity. However, VN layer protection in Fig. 2(a) can exploit lower path diversity in a VN to share spare capacities allocated on all the virtual links with one another. Although backup capacity sharing across different VNs could increase the amount of sharing, such sharing is restricted by the fact that virtual links from different VNs may be embedded on different substrate paths to satisfy the location constraints of virtual nodes. These substrate paths may have small number of substrate links in common, resulting in less opportunities of spare capacity sharing than the VN level sharing. We also validate this claim later in our evaluation (§ VI). Despite the backup sharing advantages, one can argue that the increased number of signaling and rerouting operations required for VN level survivability may lead to slower restoration compared to ensuring survivability at the SN layer. However, A recent study empirically evaluated the impact of providing protection at the SN level, compared to that at the VN level, on a real testbed [22]. The study shows that: (i) both approaches have similar protection switching time during a failure, and (ii) the latter can accommodate more VNs than the former, thanks to its resource efficiency. InPs can thus increase revenue by adopting VN level survivability without severely impacting failure response times.

A major challenge in SVNE with VN level protection is to jointly optimize spare backup capacity allocation in the VN and survivable VN embedding on the SN. Spare capacity allocation and survivable VN embedding have been studied extensively as independent problems [16], [19], [26] and have been proven to be $\mathcal{NP}$-complete and $\mathcal{NP}$-hard, respectively. SVNE with VN level protection stresses the need to solve these two problems simultaneously, since they can affect each other. For example, an optimal spare capacity allocation without considering VN embedding can be suboptimal, or may even render the embedding infeasible. Similarly, an optimal VN embedding without consideration for spare capacity allocation may lead to suboptimal or even infeasible spare capacity allocation later. The intricacy of the problem is exacerbated by spare capacity sharing, which is an efficient technique for minimizing aggregate spare capacity [12], [13], [18]. Spare capacity sharing is possible as long as the virtual links, that use the spare capacity as backup in the event of a substrate link failure, are not impacted by the same substrate failure. This requirement can impose constraints such as having a certain number of disjoint paths for backup provisioning that, in turn, can lead to increased embedding cost. Such path disjointedness requirements can be alleviated by allocating dedicated spare capacities instead of sharing the spare capacity. This can also increase embedding cost. Therefore, there is a trade-off between backup path selection and spare backup capacity sharing in a VN and embedding of the VN with disjoint path constraints. Striking a good balance between these choices is challenging, and mandates a thorough investigation. Existing schemes either do not consider all the subproblems of this joint optimization, or address them in separate independent steps, leading to suboptimal solutions [9], [19], [20], [22], [27].

In this paper, we study the joint optimization problem discussed above with the objective of guaranteeing VN survivability under multiple substrate link failures and minimizing resource usage in the SN. We start with single link failure

scenario and then extend the solution to survive multiple link failures. Specifically, we make the following contributions:

- We formulate a joint optimization model using a Quadratic Integer Program (QIP) to optimally solve spare capacity allocation and survivable VN embedding simultaneously. We transform the QIP into an Integer Linear Program (ILP) without sacrificing its optimality. We provide ILP formulations for solving two extreme cases of spare capacity sharing, one of which defines the upper bound of the cost function. We present a mathematical analysis that dictates how the topological properties of the SN affect the level of spare capacity sharing.
- The ILP formulations for the joint optimization problem are not scalable to large problem instances. Hence, we devise an efficient heuristic algorithm to tackle the computational complexity of the ILP-based solutions.
- We perform simulations to evaluate our solutions for single and double link failures. We restrict our evaluation to one and two link failures since the probability of more than two simultaneous link failures is extremely low [28], [29]. We also compare shared backup protection at the VN level with the same at the SN layer proposed in [13].
- We discuss the signaling mechanism and the enabling technology to realize the protection at the VN layer. We also discuss the flexibilities an SP can have by leveraging the spare capacity on the virtual links of a VN.

This work extends our initial paper presented in [30] on several aspects. First, we extend our previous ILP formulation and heuristic (Alg. 1-3) to handle multiple substrate link failures in § IV-D and § V, respectively. We present ILP formulations for two special cases of the joint optimization problem for single failure in § IV-C. We also provide a mathematical analysis to show the impact of SN topology on the level of spare capacity sharing in § IV-C.3. We extend § VI to include evaluation results of our solutions against double link failures, results of the two special cases of joint optimization presented in § IV, and a quantitative comparison between addressing SVNE at the InP level and that at the SP level. Finally, we provide more in-depth discussion of related work and compare our proposal with the sate-of-the-art.

The rest of this paper is organized as follows. We present the related literature in § II and contrast our work with the state-of-the-art. In § III, we present the system model and problem statement followed by a discussion on how spare capacity can be allocated along a virtual link. Then, we present our QIP formulation for the joint optimization problem and the ILP formulations for the optimal and special cases of the problem along with a mathematical analysis in § IV. We present the design of our heuristic in § V. The evaluation of our solutions are presented in § VI. Finally, we conclude with some future research directions in § VII.

II. Related Work

We discuss the state-of-the-art in SVNE with Protection at SN level and VN level in § II-A and § II-B, respectively. We then contrast our approach with similar works from multi-layer (*e.g.*, IP-over-WDM) network survivability literature in § II-C.

A. SVNE With Protection at SN

Rahman *et al.* [6] were the first to address the SVNE problem using a mixed ILP formulation. Since then a number of subsequent research works have addressed different aspects of SVNE such as substrate node failure [7], [8], [31], leveraging multi-path embedding [10], [11], shared backup protection [12], [13], [18], reactive recovery [32]–[35], and dedicated VN topology protection [14], [15] among others. Reactive recovery approaches are fundamentally different from our approach. They do not preallocate backup resources and take action after a failure. Therefore, we exclude the reactive approaches from our discussion here.

The SVNE literature exhibits a wide spectrum of protection approaches to improve resource utilization. For instance, shared backup approaches proposed in [12], [13], and [18] promote sharing of backup substrate resources for different virtual entities. In contrast, the proposals in [14] and [15] go to another extreme and propose to provide dedicated protection for the whole VN topology, *i.e.*, provision a full copy of the VN as a backup. Their motivation is to strictly satisfy failure recovery SLAs in transport networks carrying high volume of traffic. The approaches described in [10] and [11] try to optimize backup resource allocation by assuming in-network multi-path routing support. They do not share backup among the virtual entities, nor do they provide dedicated protection to the VN topology. These proposals assume a virtual link demand can be realized by multiple paths in the SN. Given that the probability of all the embedding paths failing simultaneously is very low, only a fraction of the virtual link demand needs to be provisioned disjointedly as a backup. All the discussed approaches address the SVNE problem from an InP's perspective, *i.e.*, the InP provisions backup resources for a VN in the SN and manages failure recovery tasks. However, they do not explore the solution space where a VN is embedded in a way that an SP can perform failure handling as we study in this paper.

B. SVNE With Protection at VN

Compared to the approaches that provide protection at the SN level, only a few SVNE approaches focus on providing protection at the VN level. Among them, an empirical study by Wang *et al.* [22] compared different protection schemes for VNs in T-SDNs. The results in [22] show that providing protection at the VN level can increase VN acceptance ratio. However, Wang *et al.* [22] do not provide any mechanism to jointly optimize spare capacity allocation in a VN and embed the VN accordingly. We previously studied a weaker version of the SVNE problem with VN level protection in [9]. This work proposed to embed a VN on an SN in such way that VN connectivity is ensured against multiple substrate link failures. When failure occurs, the SP has to compute alternate paths in the VN to restore the affected traffic which may incur delay. In addition, this work does not guarantee any bandwidth

in case of a failure and only allows the VN to operate in a best effort manner. Barla *et al.* [36] proposed design models for cloud services that provide resiliency either at the VN or at the SN layer. Their resiliency model employs dedicated backup consisting of additional virtual links to reach the recovery data center. In contrast, our approach alleviates the need for changing the VN topology and uses shared spare capacity allocated to existing virtual links to survive link failures.

C. Network Survivability in Multi-Layer Networks

A similar problem to our joint optimization has been studied in the IP-over-WDM literature, namely, *Strongly Survivable Routing (SSR)* [27]. SSR performs mapping and capacity assignment of IP links over lighpaths in a WDM network in way that guarantees connectivity and spare capacity at the IP layer during a failure in either IP or WDM layer. A simplified version of SSR, namely, *Weakly Survivable Routing (WSR)* determines the mapping of an IP network that remains connected upon a WDM link failure. Although SSR and WSR delegate survivability to the IP layer of a multi-layer network, they do not pre-compute backup paths as we propose in this paper. After a link fails, they require a time-consuming step for finding alternate paths using the spare capacity in the IP layer. In our approach for VN layer survivability, we obviate such restoration step by computing and storing the backup paths during VN embedding. Despite the difference, we now discuss some prominent works that address SSR as well as spare capacity allocation problem in multi-layer networks.

Kan *et al.* [19] and Lin *et al.* [27] addressed the SSR problem in two separate steps. In the first step, they solve the WSR problem with distinct objectives. Specifically, Lin *et al.* [27] map IP links on WDM lightpaths such that the smallest capacity of the WDM link in the lightpaths is maximized, whereas Kan *et al.* [19] minimize the maximum IP bandwidth lost due to a WDM link failure. In the second step, they assign spare capacity to the resultant mapping to ensure that bandwidth demands of the IP links can be rerouted with full capacity after a WDM link fails. However, their two stage approach to the problem may not lead to the optimal solution. More importantly, the first stage of their approach can generate a mapping that may become infeasible in the second stage if the required spare capacity cannot be assigned due to some resource constraints. In addition, they do not consider spare capacity sharing to minimize resource usage.

To improve resource efficiency, Liu *et al.* [20] leveraged backup capacity sharing by allocating spare capacity in the top layer of a two-layer network. However, this work assumes that the mapping between top and bottom layers is pre-computed and given as input to the spare capacity allocation problem. Hence, this approach suffers from its inability to achieve optimality similar to [19] and [27]. Kubilinskas and Pioro [21] studied the survivability problem at the IP layer of an IP-over-WDM network using hot standby path protection. Their formulation has two shortcomings. First, it requires a set of pre-computed candidate paths for each IP layer demand. Second, although the formulation supports capacity sharing between a primary path and its standby protection path, it does not support capacity sharing among the protection paths, thus resulting in poor resource utilization. A common feature of the solutions of IP-over-WDM literature is that they assume a fixed placement of IP routers in the network, whereas an SVNE algorithm needs to determine both the mappings of virtual nodes and links. Hence, solutions from IP-over-WDM networks cannot be directly applied to our problem.

III. System Model and Background

We first present basic notations in § III-A and a formal statement of the problem in § III-B. We explain the concept of shared risk groups in § III-C. We then discuss how embedding affects spare capacity allocation on virtual links in § III-D.

A. Basic Notations

1) Substrate Network: We represent the substrate network (SN) as an undirected graph, $G = (V, E)$, where V and E denote the set of substrate nodes (SNodes) and links (SLinks), respectively. The set of neighbors of an SNode $u \in V$ is denoted by $\mathcal{N}(u)$. We associate the following attributes with each SLink $(u, v) \in E$: (i) b_{uv} : bandwidth capacity of the SLink (u, v), (ii) C_{uv} : cost of allocating unit bandwidth on (u, v) for a VLink. We assume that the SNodes are network nodes with sufficient capacity to switch traffic at peak rate between any pair of ports. Therefore, we do not consider any node mapping cost or node capacity constraint.

2) Virtual Network: We represent the virtual network (VN) as an undirected graph $\hat{G} = (\hat{V}, \hat{E})$, where $\hat{V}$ and $\hat{E}$ represent the set of virtual nodes (VNodes) and virtual links (VLinks), respectively. The set of neighbors of a VNode $\hat{v} \in \hat{V}$ is denoted by $\mathcal{N}(\hat{v})$. Each VLink $(\hat{u}, \hat{v}) \in \hat{E}$ has a bandwidth demand $b_{\hat{u}\hat{v}}$. We also have a set of location constraints (LC), $L = \{L(\hat{u}) | L(\hat{u}) \subseteq V,\ \forall \hat{u} \in \hat{V}\}$, such that a VNode $\hat{u} \in \hat{V}$ can only be provisioned on an SNode $u \in L(\hat{u})$. We use a binary variable $\ell_{\hat{u}u}$ (1 if $\hat{u} \in \hat{V}$ can be provisioned on $u \in V$, 0 otherwise), to represent this location constraint. We denote the spare backup bandwidth allocated to a VLink $(\hat{u}, \hat{v}) \in \hat{E}$ that serves as a backup for other VLinks by $S_{\hat{u}\hat{v}}$. We assume VNs are already K-edge connected to survive K SLink failures. K-edge connectivity is a necessary condition to ensure that at least K edge disjoint backup virtual paths always exist for each VLink $(\hat{u}, \hat{v}) \in \hat{E}$ [27]. However, if a VN has less than K-edge connectivity, any VN augmentation strategy such as [9], [37], and [38] can make the VN K-edge connected.

B. Problem Statement

Given an SN $G = (V, E)$, a VN $\hat{G} = (\hat{V}, \hat{E})$, and LC L:

- For each VLink $(\hat{u}, \hat{v}) \in \hat{E}$, allocate spare capacity along a set of K backup virtual paths (VPaths) $\hat{\mathscr{P}}^K_{\hat{u}\hat{v}} = \{\hat{P}^k_{\hat{u}\hat{v}} | 1 \leq k \leq K\}$ in the VN, where $\hat{P}^k_{\hat{u}\hat{v}}$ is the k_{th} VPath between $\hat{u}$ and $\hat{v}$ such that $\hat{P}^k_{\hat{u}\hat{v}}$ is edge disjoint from $(\hat{u}, \hat{v})$ and from each $\hat{P}^j_{\hat{u}\hat{v}} \in \hat{\mathscr{P}}^K_{\hat{u}\hat{v}}$ with $j \neq k$, and $b_{\hat{u}\hat{v}}$ spare bandwidth is available on the VLinks in $\hat{P}^k_{\hat{u}\hat{v}}$ after $(\hat{u}, \hat{v})$ is affected by an SLink failure.

- Map each VNode $\hat{v} \in \hat{V}$ to exactly one SNode, $u \in V$. Multiple VNodes from the same VN request should not be mapped to the same SNode. However, multiple VNodes from different VNs can share an SNode.
- Map each VLink $(\hat{u}, \hat{v})$ to a non-empty substrate path (SPath) $P_{\hat{u}\hat{v}}$ having sufficient bandwidth to accommodate the primary demand of $(\hat{u}, \hat{v})$ and the spare backup bandwidth allocated on $(\hat{u}, \hat{v})$. A VLink $(\hat{u}, \hat{v})$ and the VLinks on its VPath $\hat{P}^k_{\hat{u}\hat{v}}$ are edge disjointedly mapped on the SN to ensure that SLink failures do not affect them at the same time. Similarly, two VLinks present in the two VPaths, such as $\hat{P}^k_{\hat{u}\hat{v}}$ and $\hat{P}^j_{\hat{u}\hat{v}}$ where $j \neq k$, of the same VLink $(\hat{u}, \hat{v})$ are mapped on edge disjoint SPaths to eliminate the risk of both the VPaths failing together.
- Minimize the total cost of allocating bandwidth on the SN to embed the VN equipped with spare bandwidth.

C. Shared Risk Group

VLinks that share at least one SLink on their mapped SPaths share the same risk since all of them can be impacted if the shared SLink fails. In a context where only single SLink failure is considered, a set of VLinks belong to the same shared risk group (SRG) if and only if they share at least one SLink on their mapped SPaths. In contrast, VLinks that do not share any SLink on their mapped SPaths belong to different SRGs. To represent the SRGs, we partition the VLinks into a number of SRGs represented by the set $D = \{d_1, d_2, d_3, \ldots d_{|D|}\}$, where $|D| \leq |\hat{E}|$. A VLink belongs to exactly one SRG $d_i \in D$ and shares at least one SLink on its mapped SPath with other VLinks in d_i. We use the following variable to decide a VLink's membership to an SRG:

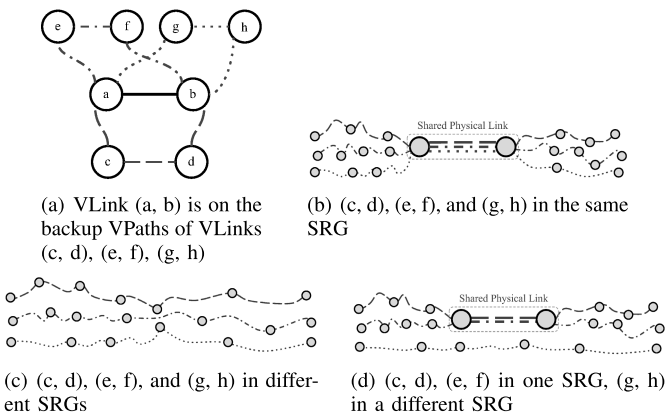
$$d_i^{\hat{u}\hat{v}} = \begin{cases} 1 & \textit{iff } (\hat{u}, \hat{v}) \in \hat{E} \text{ belongs to SRG } d_i \in D, \\ 0 & \text{otherwise.} \end{cases}$$

D. Spare Capacity Assignment Model

Based on how the VLinks form different SRGs during VN embedding, the requirement for spare backup capacity on the VLinks can be different. We explain this fact with a simple example illustrated in Fig. 3. In this example, VLink (a, b) is on the backup VPaths of three other VLinks: (c, d), (e, f), and (g, h) as shown in Fig. 3(a). We can assign different spare capacity on (a, b) to protect (c, d), (e, f), and (g, h), based on how these three VLinks are mapped. Consider the following scenarios regarding their mappings:

All three belong to the same SRG. If all three VLinks are in the same SRG, then they share at least one SLink on their mapped SPaths (Fig. 3(b)). A single SLink failure can affect all three VLinks. Therefore, spare backup capacity allocated on (a, b) should be sufficient to support the bandwidth requirement of all three VLinks, *i.e.*, $S_{ab} = b_{cd} + b_{ef} + b_{gh}$.

All three belong to different SRGs. If all three VLinks belong to different SRGs, then they do not share any SLink on their mapped SPaths (Fig. 3(c)). At most one of the VLinks will be affected by a single SLink failure. Therefore, S_{ab} should be sufficient to support the maximum bandwidth requirement of these three VLinks, *i.e.*, $S_{ab} = \max(b_{cd}, b_{ef}, b_{gh})$.



(a) VLink (a, b) is on the backup VPaths of VLinks (c, d), (e, f), (g, h)

(b) (c, d), (e, f), and (g, h) in the same SRG

(c) (c, d), (e, f), and (g, h) in different SRGs

(d) (c, d), (e, f) in one SRG, (g, h) in a different SRG

Fig. 3. Illustration of different SRGs based on embedding.

Two belong to the same SRG, the third in a different SRG. The mapped SPaths can create multiple SRGs out of these three VLinks. For example, in Fig. 3(d), VLinks (c, d) and (e, f) belong to the same SRG, whereas VLink (g, h) belongs to a different SRG. A single SLink failure will then affect only one group. Therefore, S_{ab} should be sufficient to support the group with the maximum requirement. For the group with (c, d) and (e, f), the bandwidth requirement is $b_{cd} + b_{ef}$. For the other group, the requirement is b_{gh}. Therefore, spare backup bandwidth on (a, b) should be $\max(b_{cd} + b_{ef}, b_{gh})$.

More formally, if a VLink $(\hat{u}, \hat{v}) \in \hat{E}$ is present on the backup VPaths of a set of VLinks $\hat{\mathcal{H}}_{\hat{u}\hat{v}} \subseteq \hat{E}$, and VLinks in $\hat{E}$ form a set of $D = \{d_1, d_2, d_3, \ldots d_{|D|}\}$ SRGs, we can generalize the spare backup bandwidth allocated to $(\hat{u}, \hat{v})$ as:

$$S_{\hat{u}\hat{v}} = \max_{\forall d_i \in D} \left(\sum_{\forall (\hat{x}, \hat{y}) \in \hat{\mathcal{H}}_{\hat{u}\hat{v}}} d_i^{\hat{x}\hat{y}} b_{\hat{x}\hat{y}} \right) \tag{1}$$

IV. Problem Formulation

We first provide a QIP formulation to optimally solve the joint spare capacity allocation and survivable embedding problem for the case of single substrate link failure in § IV-A. We then describe the transformation of the QIP to an ILP, namely *Opt-ILP*, in § IV-B. However, due to an overwhelming number of decision variables and constraints, *Opt-ILP* is only scalable to very small problem instances. Therefore, in § IV-C, we present simplified ILP formulations for two special cases of the joint optimization problem, along with a mathematical analysis that dictates how to select one of the ILP formulations based on the topological properties of an SN. Finally, § IV-D discusses how we can extend our solution to handle multiple independent SLink failures.

A. Quadratic Integer Program Formulation

We first present our decision variables (§ IV-A.1). Then we introduce the constraints (§ IV-A.2) followed by the objective function of our formulation (§ IV-A.3).

1) Decision Variables: For each VLink $(\hat{u}, \hat{v}) \in \hat{E}$, there is a backup VPath $\hat{P}_{\hat{u}\hat{v}}$ that provides protection to $(\hat{u}, \hat{v})$ from a single SLink failure. When any SLink on the VLink's mapped SPath fails, $\hat{P}_{\hat{u}\hat{v}}$ provides the full bandwidth $b_{\hat{u}\hat{v}}$ between the VNodes $\hat{u}$ and $\hat{v}$. The following decision variable defines whether a VLink $(\hat{u}, \hat{v}) \in \hat{E}$ belongs to the VPath protecting a VLink $(\hat{x}, \hat{y}) \in \hat{E}$:

$$z_{\hat{x}\hat{y}}^{\hat{u}\hat{v}} = \begin{cases} 1 & \text{if } (\hat{u}, \hat{v}) \in \hat{E} \text{ is on the backup VPath of } (\hat{x}, \hat{y}) \in \hat{E}, \\ 0 & \text{otherwise.} \end{cases}$$

Note that, $z_{\hat{u}\hat{v}}^{\hat{u}\hat{v}} = 0$, since a VLink's backup VPath has to be edge disjoint from itself.

The following decision variable indicates the mapping between a VLink $(\hat{u}, \hat{v}) \in \hat{E}$ and an SLink $(u, v) \in E$:

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

The VNode to SNode mapping is denoted using the following decision variable:

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

As discussed in § III-C, VLinks that share at least one SLink on their mapped SPaths belong to the same SRG. SRG membership is defined using the decision variable $d_i^{\hat{u}\hat{v}}$, presented in § III-C.

2) Constraints:

a) VNode Mapping Constraints: (2) and (3) ensure that each VNode of a VN is provisioned on an SNode satisfying the provided location constraints. Moreover, (4) constraints an SNode to host at most one VNode from the same VN. Note that VNode mapping follows from the VLink mapping, since there is no cost associated with the VNode mapping.

$$\forall \hat{u} \in \hat{V}, \quad \forall u \in V : y_{\hat{u}u} \leq \ell_{\hat{u}u} \tag{2}$$

$$\forall \hat{u} \in \hat{V} : \sum_{u \in V} y_{\hat{u}u} = 1 \tag{3}$$

$$\forall u \in V : \sum_{\hat{u} \in \hat{V}} y_{\hat{u}u} \leq 1 \tag{4}$$

b) Backup VPath Continuity Constraints: A VLink in a VN is protected by a VPath in the VN to survive a single SLink failure. (5) ensures continuity of a backup VPath protecting a VLink $(\hat{x}, \hat{y}) \in \hat{E}$:

$$\forall (\hat{x}, \hat{y}) \in \hat{E} : \sum_{\forall \hat{v} \in \mathcal{N}(\hat{u}) \setminus \{\hat{y}\}} (z_{\hat{x}\hat{y}}^{\hat{u}\hat{v}} - z_{\hat{x}\hat{y}}^{\hat{v}\hat{u}}) = \begin{cases} 1 & \text{if } \hat{u} = \hat{x} \\ -1 & \text{if } \hat{u} = \hat{y} \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

c) VLink Mapping Constraints: We ensure that every VLink is mapped to a non-empty set of SLinks using (6). Then, (7) makes sure that the in-flow and out-flow of each SNode is equal, except for the SNodes where the endpoints of a VLink are mapped. This ensures that the non-empty set of SLinks corresponding to a VLink's mapping form a single continuous SPath.

$$\forall (\hat{u}, \hat{v}) \in \hat{E} : \sum_{\forall (u,v) \in E} x_{uv}^{\hat{u}\hat{v}} \geq 1 \tag{6}$$

$$\forall \hat{u}, \hat{v} \in \hat{V}, \ \forall u \in V : \sum_{\forall v \in \mathcal{N}(u)} (x_{uv}^{\hat{u}\hat{v}} - x_{vu}^{\hat{u}\hat{v}}) = y_{\hat{u}u} - y_{\hat{v}u} \tag{7}$$

The binary nature of the VLink mapping decision variable and the flow constraint prevent any VLink from being mapped to more than one SPaths, thus, restricting the VLink mapping to the *Multi-commodity Unsplittable Flow Problem* [39].

d) Capacity Constraints: We also need to ensure that we do not over-commit the bandwidth resources we have on the SLinks. To do so, we first compute the spare backup bandwidth allocated to a VLink $(\hat{u}, \hat{v}) \in \hat{E}$ using (1) as follows:

$$S_{\hat{u}\hat{v}} = \max_{\forall d_i \in D} \sum_{\forall (\hat{x}, \hat{y}) \in \hat{E} \setminus \{(\hat{u}, \hat{v})\}} z_{\hat{x}\hat{y}}^{\hat{u}\hat{v}} \times d_i^{\hat{x}\hat{y}} \times b_{\hat{x}\hat{y}} \tag{8}$$

Then, the following constraints prevent any over-commit of the bandwidth resource on the SLinks:

$$\forall (u, v) \in E : \sum_{\forall (\hat{u}, \hat{v}) \in \hat{E}} x_{uv}^{\hat{u}\hat{v}} \times (b_{\hat{u}\hat{v}} + S_{\hat{u}\hat{v}}) \leq b_{uv} \tag{9}$$

Note that (9) is a cubic constraint, since $S_{\hat{u}\hat{v}}$ is quadratic according to (8). Therefore, we take the following steps to linearize $S_{\hat{u}\hat{v}}$ in order to keep (9) in quadratic order. First, we introduce a new variable $g_{\hat{x}\hat{y}}^{\hat{u}\hat{v}}(i)$, defined as follows:

$$g_{\hat{x}\hat{y}}^{\hat{u}\hat{v}}(i) = \begin{cases} 0 & \text{if } z_{\hat{x}\hat{y}}^{\hat{u}\hat{v}} = 1 \text{ and } d_i^{\hat{x}\hat{y}} = 1, \\ 1 & \text{otherwise.} \end{cases}$$

Essentially, for a given VLink $(\hat{u}, \hat{v}) \in \hat{E}$, the zero values of $g_{\hat{x}\hat{y}}^{\hat{u}\hat{v}}(i)$ induce a set of VLinks that belong to the same SRG and have $(\hat{u}, \hat{v})$ on their backup VPaths. The value of $g_{\hat{x}\hat{y}}^{\hat{u}\hat{v}}(i)$ is set using the following constraint:

$$\forall (\hat{u}, \hat{v}) \in \hat{E}, \forall (\hat{x}, \hat{y}) \in \hat{E}, \forall d_i \in D : z_{\hat{x}\hat{y}}^{\hat{u}\hat{v}} + d_i^{\hat{x}\hat{y}} + g_{\hat{x}\hat{y}}^{\hat{u}\hat{v}}(i) \leq 2 \tag{10}$$

We can use $g_{\hat{x}\hat{y}}^{\hat{u}\hat{v}}(i)$ to rewrite (8) in a linear form as follows:

$$S_{\hat{u}\hat{v}} = \max_{\forall d_i \in D} \sum_{\forall (\hat{x}, \hat{y}) \in \hat{E} \setminus \{(\hat{u}, \hat{v})\}} (1 - g_{\hat{x}\hat{y}}^{\hat{u}\hat{v}}(i)) \times b_{\hat{x}\hat{y}} \tag{11}$$

Since our objective function will be a minimization function, we define $g_{\hat{x}\hat{y}}^{\hat{u}\hat{v}}(i)$ such that setting it to 1 minimizes the value of $S_{\hat{u}\hat{v}}$, unless it is constrained to be 0 according to (10). This constrained case will only occur when both $z_{\hat{x}\hat{y}}^{\hat{u}\hat{v}}$ and $d_i^{\hat{x}\hat{y}}$ are 1, as enforced by (10).

e) SRG Constraints: The mapped SPaths of the VLinks from an SRG d_i, must be edge disjoint from the mapped SPaths of the VLinks from a different SRG d_j $(\forall j \neq i)$. This is accounted for in (12). (13) ensures that two VLinks from the same SRG share at least one SLink on their mapped SPaths. Note that a VLink $(\hat{u}, \hat{v}) \in \hat{E}$ cannot be present in

more than one SRGs, which is satisfied by (14).

$$\forall(u,v)\in E,\quad \forall(\hat{u},\hat{v})\in\hat{E},\ \forall(\hat{x},\hat{y})\in\hat{E},$$
$$\forall d_i\in D,\quad \forall d_j\in D\quad s.t.\ \ (\hat{u},\hat{v})\neq(\hat{x},\hat{y}) \text{ and } i\neq j:$$
$$d_i^{\hat{u}\hat{v}}+d_j^{\hat{x}\hat{y}}+x_{uv}^{\hat{u}\hat{v}}+x_{vu}^{\hat{u}\hat{v}}+x_{uv}^{\hat{x}\hat{y}}+x_{vu}^{\hat{x}\hat{y}}\leq 3 \tag{12}$$
$$\forall d_i\in D,\quad \forall(\hat{u},\hat{v})\in\hat{E},\ \forall(\hat{x},\hat{y})\in\hat{E}\quad s.t.\ \ (\hat{u},\hat{v})\neq(\hat{x},\hat{y}),$$
$$\exists(u,v)\in E: d_i^{\hat{u}\hat{v}}+d_i^{\hat{x}\hat{y}}+x_{uv}^{\hat{u}\hat{v}}+x_{vu}^{\hat{u}\hat{v}}+x_{uv}^{\hat{x}\hat{y}}+x_{vu}^{\hat{x}\hat{y}}=4 \tag{13}$$
$$\forall(\hat{x},\hat{y})\in\hat{E}:\sum_{\forall d_i\in D} d_i^{\hat{x}\hat{y}}=1 \tag{14}$$

f) Survivability Constraints: To ensure survivability of the VN under single SLink failure, the mapped SPath of a VLink cannot share any SLink with the mapped SPaths of the VLinks present on its backup VPath. The following constraints make sure this edge disjointedness requirement:

$$\forall(u,v)\in E,\quad \forall((\hat{u},\hat{v}),(\hat{x},\hat{y}))\in\hat{E}\times\hat{E}\quad s.t.\ \ (\hat{u},\hat{v})\neq(\hat{x},\hat{y}):$$
$$z_{\hat{u}\hat{v}}^{\hat{x}\hat{y}}+x_{uv}^{\hat{u}\hat{v}}+x_{vu}^{\hat{u}\hat{v}}+x_{uv}^{\hat{x}\hat{y}}+x_{vu}^{\hat{x}\hat{y}}\leq 2 \tag{15}$$

3) Objective Function: As per the problem statement presented in § III-B, we do not consider any node mapping cost in our VN embedding. Thus, our cost function minimizes the total cost of provisioning the working and spare backup bandwidth for the VLinks of a VN on the SLinks of an SN. This gives us the following objective function:

$$\text{minimize}\left(\sum_{\forall(\hat{u},\hat{v})\in\hat{E}}\ \sum_{\forall(u,v)\in E} x_{uv}^{\hat{u}\hat{v}}\times C_{uv}\times(b_{\hat{u}\hat{v}}+S_{\hat{u}\hat{v}})\right) \tag{16}$$

B. ILP Transformation, Opt-ILP

Our formulation for the joint optimization problem has a quadratic constraint (9) and a quadratic objective function (16). Therefore, the QIP presented in § IV-A is a Quadratically Constrained Quadratic Program (QCQP) and falls into the general category of the Quadratic Assignment Problem (QAP) [40]. Solving a QAP is computationally expensive and is known to be $\mathcal{NP}$-hard [41]. Sahni and Gonzalez [42] proved that even finding an $\epsilon-$approximate solution of QAP is $\mathcal{NP}$-hard. We now present the steps to linearize the QIP by using a technique similar to the one discussed in [43]. For the purpose of linearization, we first put a bound on the spare backup bandwidth of a VLink $(\hat{u},\hat{v})\in\hat{E}$, *i.e.*, $S_{\hat{u}\hat{v}}$: $0\leq S_{\hat{u}\hat{v}}\leq\lambda$, where λ is a very large value. We also introduce a new integer variable $q_{uv}^{\hat{u}\hat{v}}$, defined in terms of $x_{uv}^{\hat{u}\hat{v}}$ as follows:

$$q_{uv}^{\hat{u}\hat{v}}=\begin{cases}S_{\hat{u}\hat{v}} & \text{if } x_{uv}^{\hat{u}\hat{v}}=1,\\ 0 & \text{if } x_{uv}^{\hat{u}\hat{v}}=0.\end{cases}$$

The following constraints enforce the above definition.

$$\forall(u,v)\in E,\quad \forall(\hat{u},\hat{v})\in\hat{E}: q_{uv}^{\hat{u}\hat{v}}\geq 0 \tag{17}$$
$$\forall(u,v)\in E,\quad \forall(\hat{u},\hat{v})\in\hat{E}: S_{\hat{u}\hat{v}}-\lambda\times(1-x_{uv}^{\hat{u}\hat{v}})\leq q_{uv}^{\hat{u}\hat{v}} \tag{18}$$

To elaborate, when $x_{uv}^{\hat{u}\hat{v}}=0$, constraints (17) and (18) become $q_{uv}^{\hat{u}\hat{v}}\geq 0$ and $S_{\hat{u}\hat{v}}-\lambda\leq q_{uv}^{\hat{u}\hat{v}}$, respectively. Since λ is a very large value by definition, the constraints finally reduce to $q_{uv}^{\hat{u}\hat{v}}\geq 0$. On the other hand, when $x_{uv}^{\hat{u}\hat{v}}=1$, constraint (17) and (18) become $q_{uv}^{\hat{u}\hat{v}}\geq 0$ and $S_{\hat{u}\hat{v}}\leq q_{uv}^{\hat{u}\hat{v}}$, respectively. In this later case, constraint (18), *i.e.*, $S_{\hat{u}\hat{v}}\leq q_{uv}^{\hat{u}\hat{v}}$ dominates. Finally, if we include $q_{uv}^{\hat{u}\hat{v}}$ in the minimization objective function, the smallest possible value of $q_{uv}^{\hat{u}\hat{v}}$ will be used to minimize the value of the objective function, yielding $q_{uv}^{\hat{u}\hat{v}}=S_{\hat{u}\hat{v}}$ (for $x_{uv}^{\hat{u}\hat{v}}=1$) and $q_{uv}^{\hat{u}\hat{v}}=0$ (for $x_{uv}^{\hat{u}\hat{v}}=0$).

We now rewrite the capacity constraint (9) as the following linear constraint using $q_{uv}^{\hat{u}\hat{v}}$.

$$\forall(u,v)\in E:\sum_{\forall(\hat{u},\hat{v})\in\hat{E}}(x_{uv}^{\hat{u}\hat{v}}\times b_{\hat{u}\hat{v}}+q_{uv}^{\hat{u}\hat{v}})\leq b_{uv} \tag{19}$$

Similarly, the quadratic objective function can be written in a linearized form as follows:

$$\text{minimize}\left(\sum_{\forall(\hat{u},\hat{v})\in\hat{E}}\ \sum_{\forall(u,v)\in E} x_{uv}^{\hat{u}\hat{v}}\times C_{uv}\times b_{\hat{u}\hat{v}}+C_{uv}\times q_{uv}^{\hat{u}\hat{v}})\right) \tag{20}$$

C. Problem Variations

Opt-ILP is scalable to very small problem instances due to its large number of constraints and variables. For instance, the number of constraints generated by capacity constraints (10) and SRG constraints (12) and (13) are $\hat{E}^3$, $E\times\hat{E}^4$, and $E\times\hat{E}^3$, respectively. Similarly, the number of variables generated by $d_i^{\hat{x}\hat{y}}$ and $g_{\hat{x}\hat{y}}^{\hat{u}\hat{v}}(i)$ are $\hat{E}^2$ and $\hat{E}^3$. To reduce problem complexity, we formulate two simpler variants of *Opt-ILP*, namely *Max-ILP* and *Min-ILP*, that represent maximum and minimum sharing of spare capacity, respectively.

1) ILP for Maximum Spare Capacity Sharing, Max-ILP: Design of *Max-ILP* is motivated by an observation from our evaluation results that for a VLink $(\hat{u},\hat{v})$, *Opt-ILP* assigns the VLinks in $\hat{\mathcal{H}}_{\hat{u}\hat{v}}$ into separate SRGs whenever VLinks in $\hat{\mathcal{H}}_{\hat{u}\hat{v}}$ can be mapped to disjoint SPaths, thus preferring more sharing of the spare capacity. Therefore, *Max-ILP* enforces maximum sharing of spare backup capacity $S_{\hat{u}\hat{v}}$ of a VLink $(\hat{u},\hat{v})$ among the VLinks in $\hat{\mathcal{H}}_{\hat{u}\hat{v}}$ that have $(\hat{u},\hat{v})$ in their VPaths, similar to the example shown in Fig. 3(c). In order to guarantee full bandwidth between pairs of VNodes during an SLink failure, *Max-ILP* forcefully assigns VLinks in $\hat{\mathcal{H}}_{\hat{u}\hat{v}}$ into separate SRGs instead of deciding the assignment dynamically during embedding. In this way, *Max-ILP* eliminates decision variables $d_i^{\hat{x}\hat{y}}$ and $g_{\hat{x}\hat{y}}^{\hat{u}\hat{v}}(i)$ from the capacity and SRG constraints of *Opt-ILP*. Therefore, in *Max-ILP*, we can rewrite (11) as follows and eliminate the constraints in (10):

$$S_{\hat{u}\hat{v}}=\max_{\forall(\hat{x},\hat{y})\in\hat{E}\setminus\{(\hat{u},\hat{v})\}} z_{\hat{x}\hat{y}}^{\hat{u}\hat{v}}\times b_{\hat{x}\hat{y}} \tag{21}$$

Similarly, we can replace SRG constraints (12), (13), and (14) of *Opt-ILP* by the following disjointedness constraints:

$$\forall(u,v)\in E,\quad \forall(\hat{u},\hat{v})\in\hat{E},\ \forall(\hat{x},\hat{y})\in\hat{E}\setminus\{(\hat{u},\hat{v})\},$$
$$\forall(\hat{a},\hat{b})\in\hat{E}\setminus\{(\hat{u},\hat{v}),(\hat{x},\hat{y})\}:$$
$$z_{\hat{x}\hat{y}}^{\hat{u}\hat{v}}+z_{\hat{a}\hat{b}}^{\hat{u}\hat{v}}+x_{uv}^{\hat{x}\hat{y}}+x_{vu}^{\hat{x}\hat{y}}+x_{uv}^{\hat{a}\hat{b}}+x_{vu}^{\hat{a}\hat{b}}\leq 3 \tag{22}$$

These disjointedness constraints ensure that if a VLink $(\hat{u},\hat{v})\in\hat{E}$ is in the backup VPaths of two other VLinks

such as $(\hat{x}, \hat{y}) \in \hat{E} \setminus (\hat{u}, \hat{v})$ and $(\hat{a}, \hat{b}) \in \hat{E} \setminus (\hat{u}, \hat{v})$, $(\hat{x}, \hat{y})$ and $(\hat{a}, \hat{b})$ cannot share an SLink in their mappings. However, $(\hat{x}, \hat{y})$ and $(\hat{a}, \hat{b})$ can share an SLink if their VPaths do not have any common VLink. Therefore, *Max-ILP* consists of the constraints (21), (22), and all the constraints of *Opt-ILP* except the constraints (10), (11), (12), (13), and (14). The same objective function (20) of *Opt-ILP* prevails in *Max-ILP*. Thus, *Max-ILP* replaces higher order variables and constraints of *Opt-ILP* by a lower order constraint. The disadvantage of *Max-ILP* is that it may require more disjoint SPaths to satisfy constraints of (22) than those required by *Opt-ILP*.

2) ILP for No Spare Capacity Sharing, Min-ILP: Max-ILP requires that all the VLinks in $\hat{\mathcal{H}}_{\hat{u}\hat{v}}$ are mapped to disjoint SPaths. This is only achievable in SNs where adequate disjoint SPaths can be found. However, in sparse SNs, *Max-ILP* may become infeasible due to the unavailability of disjoint SPaths. Such infeasibility can also occur in SNs having *trap topological* structure [44], [45]. We have observed this behavior in our evaluation for embedding VNs in sparser SNs. Despite the infeasibility of *Max-ILP*, *Opt-ILP* is able to find a solution in such SNs by striking a balance between the level of spare capacity sharing and the number of disjoint SPaths. Following this observation, we present an ILP formulation, *Min-ILP*, that does not allow any sharing of spare backup capacity as shown in Fig. 3(b). Consequently, all the VLinks in $\hat{\mathcal{H}}_{\hat{u}\hat{v}}$ belong to only one SRG and the SRG constraints (12), (13), and (14) of *Opt-ILP* can be eliminated from *Min-ILP*. In addition, we can exclude decision variables $d_i^{\hat{x}\hat{y}}$ and $g_{\hat{x}\hat{y}}^{\hat{u}\hat{v}}(i)$ and the constraint (11) from *Min-ILP*. However, the disadvantage of *Min-ILP* is that all the VLinks in $\hat{\mathcal{H}}_{\hat{u}\hat{v}}$ can be affected during an SLink failure due to sharing of the same risk. Hence, the spare backup capacity of a VLink $S_{\hat{u}\hat{v}}$ in *Min-ILP* should be adequate enough to serve the bandwidth of all the VLinks in $\hat{\mathcal{H}}_{\hat{u}\hat{v}}$. This increases the spare backup capacity requirement $S_{\hat{u}\hat{v}}$ of a VLink as shown by the following equation:

$$S_{\hat{u}\hat{v}} = \sum_{\forall(\hat{x},\hat{y})\in\hat{E}\setminus\{(\hat{u},\hat{v})\}} z_{\hat{x}\hat{y}}^{\hat{u}\hat{v}} \times b_{\hat{x}\hat{y}} \tag{23}$$

Similar to *Max-ILP*, *Min-ILP* consists of the constraints (23) and all the constraints of *Opt-ILP* except the constraints (10), (11), (12), (13), and (14). The same objective function (20) of *Opt-ILP* prevails in *Min-ILP* as well. In this way, *Min-ILP* replaces all the higher order decision variables and constraints of *Opt-ILP*. Another potential drawback of *Min-ILP* is that it has to explore a larger search space than *Max-ILP* would require due to *Min-ILP*'s lower number of constraints.

3) Comparative Study Between Max-ILP and Min-ILP: Recall from § III-D that based on how the VLinks are sharing risks in their mappings, the spare capacity allocation can be different on a VLink. One extreme case is when all the VLinks in $\hat{\mathcal{H}}_{\hat{u}\hat{v}}$ are mapped disjointedly (*Max-ILP*). Another extreme case is when there is minimal disjointedness in the mapping, *i.e.*, VLinks are mapped disjointedly only when constrained by the backup VPath's survivability constraints (15) (*Min-ILP*). We now present a mathematical analysis showing how the mapped SPath length affects the preference for disjointedness.

Let us assume for the sake of simplicity that VLinks in $\hat{H}_{\hat{u}\hat{v}}$ are not on the backup VPath of any other VLink. We represent the mean mapped SPath length for *Max-ILP* and *Min-ILP* as $\mathcal{P}$ and $\mathcal{R}$, respectively. Note that $\mathcal{P}$ and $\mathcal{R}$ should be different since *Max-ILP* has to satisfy disjointedness constraints (22), thus resulting in longer SPaths. On the other hand, SPaths in *Min-ILP* have no such disjointedness requirements to adhere to, thereby yielding shorter SPaths. Finally, we assume a unit cost of allocating bandwidth on the SLink, *i.e.*, $\forall(u, v) \in E : C_{uv} = 1$. In *Max-ILP*, each VLink in $\hat{\mathcal{H}}_{\hat{u}\hat{v}}$ belongs to different SRGs. Using (21), spare capacity requirement for *Max-ILP* is:

$$S_{\hat{u}\hat{v}}^{max} = \max_{\forall(\hat{x},\hat{y})\in\hat{\mathcal{H}}_{\hat{u}\hat{v}}} b_{\hat{x}\hat{y}} \tag{24}$$

In *Min-ILP*, all the VLinks in $\hat{\mathcal{H}}_{\hat{u}\hat{v}}$ for a given $(\hat{u}, \hat{v}) \in \hat{E}$ belong to the same SRG. Therefore, we can compute spare capacity requirement for *Min-ILP* using (23) as follows:

$$S_{\hat{u}\hat{v}}^{min} = \sum_{\forall(\hat{x},\hat{y})\in\hat{\mathcal{H}}_{\hat{u}\hat{v}}} b_{\hat{x}\hat{y}} \tag{25}$$

For *Max-ILP*, the cost of mapping the VLinks in $\hat{\mathcal{H}}_{\hat{u}\hat{v}} \cup \{(\hat{u}, \hat{v})\}$ is obtained by combining (16) and (24) as follows:

$$cost^{max} = \mathcal{P} \times \left(b_{\hat{u}\hat{v}} + \max_{\forall(\hat{x},\hat{y})\in\hat{\mathcal{H}}_{\hat{u}\hat{v}}} b_{\hat{x}\hat{y}} + \sum_{\forall(\hat{x},\hat{y})\in\hat{\mathcal{H}}_{\hat{u}\hat{v}}} b_{\hat{x}\hat{y}} \right)$$

Similarly, the cost of mapping the VLinks in $\hat{\mathcal{H}}_{\hat{u}\hat{v}} \cup \{(\hat{u}, \hat{v})\}$ for *Min-ILP* can be obtained by combining (16) and (25) as:

$$cost^{min} = \mathcal{R} \times \left(b_{\hat{u}\hat{v}} + 2 \sum_{\forall(\hat{x},\hat{y})\in\hat{\mathcal{H}}_{\hat{u}\hat{v}}} b_{\hat{x}\hat{y}} \right)$$

Now, $cost^{max} < cost^{min}$ will be true if:

$$\mathcal{P} \times \left(b_{\hat{u}\hat{v}} + \max_{\forall(\hat{x},\hat{y})\in\hat{\mathcal{H}}_{\hat{u}\hat{v}}} b_{\hat{x}\hat{y}} + \sum_{\forall(\hat{x},\hat{y})\in\hat{\mathcal{H}}_{\hat{u}\hat{v}}} b_{\hat{x}\hat{y}} \right) < \mathcal{R} \times \left(b_{\hat{u}\hat{v}} + 2 \sum_{\forall(\hat{x},\hat{y})\in\hat{\mathcal{H}}_{\hat{u}\hat{v}}} b_{\hat{x}\hat{y}} \right)$$

This gives us the following final inequality:

$$\frac{\mathcal{P}}{\mathcal{R}} < \frac{b_{\hat{u}\hat{v}} + 2 \times \sum_{\forall(\hat{x},\hat{y})\in\hat{\mathcal{H}}_{\hat{u}\hat{v}}} b_{\hat{x}\hat{y}}}{b_{\hat{u}\hat{v}} + \max_{\forall(\hat{x},\hat{y})\in\hat{\mathcal{H}}_{\hat{u}\hat{v}}} b_{\hat{x}\hat{y}} + \sum_{\forall(\hat{x},\hat{y})\in\hat{\mathcal{H}}_{\hat{u}\hat{v}}} b_{\hat{x}\hat{y}}}$$

When the set $\hat{\mathcal{H}}_{\hat{u}\hat{v}}$ is sufficiently large, we have $b_{\hat{u}\hat{v}} \leq \max_{\forall(\hat{x},\hat{y})\in\hat{\mathcal{H}}_{\hat{u}\hat{v}}} b_{\hat{x}\hat{y}} \ll \sum_{\forall(\hat{x},\hat{y})\in\hat{\mathcal{H}}_{\hat{u}\hat{v}}} b_{\hat{x}\hat{y}}$, yielding $P <\sim 2R$. This gives us the following insight: *as long as the mean mapped SPath length of the Max-ILP does not become about twice the mean mapped SPath length of the Min-ILP, Max-ILP yields an embedding with a lower cost than Min-ILP*. In other words, cost of *Max-ILP* will only exceed the cost of *Min-ILP* when the increase in mean SPath length due to satisfying disjointedness constraints (22) is almost equal to the mean SPath

length computed without the disjointedness constraints (22). Such situation can occur in sparse SNs that lack path diversity. In contrast, dense SNs have higher path diversity and higher number of SLinks. Consequently, the number of edge-disjoint SPaths is also higher in denser SNs than in sparser SNs [46]. To satisfy disjointedness constraints, denser SNs increase mean SPath length by a smaller factor compared to that in sparser SNs. Therefore, *Min-ILP* should be the preferred choice only in extremely sparse SNs, whereas *Max-ILP* can be used in all other types of SNs. This result motivates us to develop a heuristic algorithm that prefers sharing of spare capacity much like the case of *Max-ILP*.

D. Single to Multiple SLink Failures

We now discuss how our solutions for single SLink failure can be extended to handle multiple SLink failures. Since *Max-ILP* closely approximates *Opt-ILP* (see § VI-C.3) and *Max-ILP* is more resource efficient than *Min-ILP* (see § IV-C.3), we focus on extending *Max-ILP* for multiple SLink failures. As discussed in [47], there are two methods to survive against multiple (say K) link failures in a single layer network, a VN in our case. In the first method, for each VLink $(\hat{u}, \hat{v})$, provision spare capacities across K edge-disjoint backup VPaths so that at least one of them provides the full bandwidth even if $(\hat{u}, \hat{v})$ and $K-1$ of the backup VPaths are affected by K simultaneous VLink failures. During VN embedding, the disjoint path requirement translates to the following: a VLink in the k-th edge-disjoint backup VPath (including $(\hat{u}, \hat{v})$) and another VLink in the j-th ($j \neq k$) edge-disjoint VPath should be embedded on disjoint SPaths. In the second method, for each VLink $(\hat{u}, \hat{v})$, provision only one backup VPath $\hat{P}_{\hat{u}\hat{v}}$ such that the VLinks in $\hat{P}_{\hat{u}\hat{v}}$ have enough spare capacity to carry all the traffic of any K VLinks in the VN. In this case, rerouting of traffic during the failure of $(\hat{u}, \hat{v})$ is the same as previous case. However, during the failures of $(\hat{u}, \hat{v})$ and any VLink $(\hat{x}, \hat{y}) \in \hat{P}_{\hat{u}\hat{v}}$, traffic is rerouted to the VPath consisting of $(\hat{P}_{\hat{u}\hat{v}} - (\hat{x}, \hat{y})) \cup \hat{P}_{\hat{x}\hat{y}}$, where $\hat{P}_{\hat{x}\hat{y}}$ is the backup VPath between $\hat{x}$ and $\hat{y}$. This method requires lesser number of disjoint paths compared to the first method, *i.e.*, only $(\hat{u}, \hat{v})$ and any VLink in $\hat{P}_{\hat{u}\hat{v}}$ need to be embedded on disjoint SPaths. However, spare capacity requirement can be unnecessarily high since a VLink in $\hat{P}_{\hat{x}\hat{y}}$ has to carry the traffic of K other VLinks that may have been impacted by K SLink failures, limiting the applicability of the second method. An additional disadvantage of this method is that rerouted traffic may have to traverse many VLinks chained through backup VPaths when K VLinks fail [47]. Hence, we adopt the first method in extending *Max-ILP* to handle the case of multiple SLink failures as discussed in the following. First, we modify the decision variable defining VPath relationships to take into account multiple VPaths as below:

$$z_{\hat{x}\hat{y}}^{\hat{u}\hat{v}}(k) = \begin{cases} 1 & \text{if } (\hat{u}, \hat{v}) \text{ is on the } k_{th} \text{ backup VPath of } (\hat{x}, \hat{y}), \\ 0 & \text{otherwise.} \end{cases}$$

Similar to (5), we need to have VPath continuity constraints for each of backup VPaths of a VLink. In addition, we need the following constraints to ensure the edge-disjointedness of the K backup VPaths of a VLink.

$$\begin{aligned} &\forall k = 1, 2, \ldots, K, \quad \forall j = 1, 2, \ldots, K \quad s.t. \;\; k \neq j, \\ &\forall ((\hat{u}, \hat{v}), (\hat{x}, \hat{y})) \in \hat{E} \times \hat{E} \quad s.t. \;\; (\hat{u}, \hat{v}) \neq (\hat{x}, \hat{y}) : \\ &z_{\hat{u}\hat{v}}^{\hat{x}\hat{y}}(k) + z_{\hat{u}\hat{v}}^{\hat{x}\hat{y}}(j) \leq 1 \end{aligned} \tag{26}$$

Following (15), we need survivability constraints between the mappings of a VLink and the VLinks in each of its backup VPaths. We also need to ensure that two VLinks present in two backup VPaths of a VLink are embedded on disjoint SPaths:

$$\begin{aligned} &\forall k = 1, 2, \ldots, K, \quad \forall j = 1, 2, \ldots, K \quad s.t. \;\; k \neq j, \\ &\forall (\hat{x}, \hat{y}), \quad \forall ((\hat{u}, \hat{v}), (\hat{a}, \hat{b})) \in \hat{E} \times \hat{E} \quad s.t. \;\; (\hat{u}, \hat{v}) \neq (\hat{a}, \hat{b}) : \\ &z_{\hat{u}\hat{v}}^{\hat{x}\hat{y}}(k) + z_{\hat{a}\hat{b}}^{\hat{x}\hat{y}}(j) + x_{uv}^{\hat{u}\hat{v}} + x_{vu}^{\hat{u}\hat{v}} + x_{uv}^{\hat{a}\hat{b}} + x_{vu}^{\hat{a}\hat{b}} \leq 3 \end{aligned} \tag{27}$$

Finally, the spare capacity constraints in (21) need to be modified to handle K SLink failures. Ideally, the spare capacity $S_{\hat{u}\hat{v}}$ on a VLink $(\hat{u}, \hat{v})$ should be sufficient to carry the traffic of any K VLinks that have $(\hat{u}, \hat{v})$ in any of their backup VPaths. Accordingly, we modify the spare capacity requirement as follows:

$$S_{\hat{u}\hat{v}} = \max_{\forall \xi \subset \hat{E} \; s.t. \; |\xi| = K} \sum_{\forall (\hat{x}, \hat{y}) \in \xi} \sum_{1 \leq k \leq K} z_{\hat{u}\hat{v}}^{\hat{x}\hat{y}}(k) \times b_{\hat{x}\hat{y}} \tag{28}$$

As discussed in [47] and [48], not all backup VPaths are used simultaneously even for $K = 2$ in a single layer VN. In fact, spare capacity along VPaths can be efficiently shared by introducing constraints similar to the ones presented in [48] and by taking into account different embedding options of the VLinks of a VN. Extending these constraints for embedding a VN with $K >= 2$ can result in combinatorial number of constraints, and can be investigated as a future research.

V. Heuristic Algorithm

The coordinated node and link mapping of the aforementioned ILP formulations without the disjointedness constraints is at least as hard as the $\mathcal{NP}$-Hard *Multi-commodity Unsplittable Flow Problem* [39], when the sources and destinations of the flows are unknown. To tackle the computational intractability of the ILP formulations, we develop a heuristic algorithm for the joint spare backup capacity allocation and survivable embedding problem against multiple (e.g., K) SLink failures. Our heuristic algorithm is presented as a pseudocode in Alg. 1. The algorithm solves the joint optimization problem for arbitrary K in two steps: (i) estimate the spare backup bandwidth on the VLinks, determine the disjointedness requirements based on this estimation, and perform a VN Embedding (§ V-A), (ii) re-optimize spare backup bandwidth allocations using different backup multiplexing techniques proposed in [47]–[49] (§ V-B).

A. Joint Spare Bandwidth Allocation and VN Embedding

Alg. 1 starts by initializing the estimated spare backup bandwidth of each VLink, $S_{\hat{u}\hat{v}}^{est}$ to 0 and by placing all the VLinks into a single SRG d_1. In addition, it initializes a Max-Priority-Queue $T_{\hat{u}\hat{v}}$ for each $(\hat{u}, \hat{v})$ to be used for spare capacity

Algorithm 1: Embed VN With Protection

1 **function** *VNEmbedding(G, $\hat{G}$, C, σ, K)*
2 $D \leftarrow \{d_1, d_2, \ldots d_{|\hat{E}|}\}$
3 **foreach** $\hat{u} \in \hat{V}$ **do** $nmap_{\hat{u}} \leftarrow$ NIL
4 **foreach** $(\hat{u}, \hat{v}) \in \hat{E}$ **do**
5 $S^{est}_{\hat{u}\hat{v}} \leftarrow 0, \Lambda_{\hat{u}\hat{v}} \leftarrow \sigma, SRG_{\hat{u}\hat{v}} \leftarrow d_1, emap_{\hat{u}\hat{v}} \leftarrow \phi$
6 $T_{\hat{u}\hat{v}} \leftarrow Max - Priority - Queue()$
7 **foreach** *k=1, 2, ..., K* **do**
8 $backup^{est}_{\hat{u}\hat{v}}(k) \leftarrow \phi$
9 $\hat{\mathcal{V}} \leftarrow$ Sort $\hat{u} \in \hat{V}$ in decreasing order of $|\mathcal{N}(\hat{u})|$
10 **foreach** $\hat{u} \in \hat{\mathcal{V}}$ **do**
11 **foreach** $\hat{v} \in \mathcal{N}(\hat{u})$ **do**
12 **foreach** *k=1, 2, ..., K* **do**
13 $backup^{est}_{\hat{u}\hat{v}}(k) \leftarrow$ GetBackup($\hat{G}, (\hat{u}, \hat{v}), \Lambda, emap, k$)
14 **foreach** $(\hat{x}, \hat{y}) \in backup^{est}_{\hat{u}\hat{v}}(k)$ **do**
15 $T_{\hat{x}\hat{y}}.EnQueue(b_{\hat{u}\hat{v}})$
16 $S^{est}_{\hat{x}\hat{y}} \leftarrow \sum_{0 \le i < \min(K,\, T_{\hat{x}\hat{y}}.Size())} T_{\hat{x}\hat{y}}[i]$
17 **if** $SRG_{\hat{u}\hat{v}} = SRG_{\hat{x}\hat{y}}$ **then**
18 Find $d_j \in D$ *s.t.* $SRG_{\hat{w}\hat{z}} \neq d_j, \forall(\hat{w}, \hat{z}) \in \hat{E}$
19 $SRG_{\hat{u}\hat{v}} \leftarrow d_j$
20 $\hat{\mathcal{H}}_{\hat{x}\hat{y}} \leftarrow \{(\hat{a}, \hat{b}) \in \hat{E} | (\hat{x}, \hat{y}) \in backup^{est}_{\hat{a}\hat{b}}(1)\}$
21 **foreach** $(\hat{a}, \hat{b}) \in \hat{\mathcal{H}}_{\hat{x}\hat{y}}$ **do**
22 **if** $SRG_{\hat{u}\hat{v}} = SRG_{\hat{a}\hat{b}}$ **then**
23 Find $d_j \in D$ *s.t.* $SRG_{\hat{w}\hat{z}} \neq d_j, \forall(\hat{w}, \hat{z}) \in \hat{E}$
24 $SRG_{\hat{u}\hat{v}} \leftarrow d_j$
25 **if** $K > 1$ **then**
26 **foreach** *j=1, 2, ..., k-1* **do**
27 **foreach** $(\hat{a}, \hat{b}) \in backup^{est}_{\hat{u}\hat{v}}(j)$ **do**
28 **if** $SRG_{\hat{x}\hat{y}} = SRG_{\hat{a}\hat{b}}$ **then**
29 Find $d_j \in D$ *s.t.* $SRG_{\hat{w}\hat{z}} \neq d_j, \forall(\hat{w}, \hat{z}) \in \hat{E}$
30 $SRG_{\hat{x}\hat{y}} \leftarrow d_j$
31 $best_{\hat{u}} \leftarrow$ NIL, $Q^{best}_{\hat{u}} \leftarrow \phi, c_{best} \leftarrow \infty$
32 **foreach** $l \in L(\hat{u})$ **do**
33 **foreach** $\hat{v} \in \mathcal{N}(\hat{u})$ **do**
34 $W \leftarrow C$
35 $\zeta \leftarrow \{(u, v) \in E | SRG_{\hat{x}\hat{y}} \neq SRG_{\hat{u}\hat{v}} \wedge (u, v) \in emap_{\hat{x}\hat{y}}, \forall(\hat{x}, \hat{y}) \in \hat{E}\}$
36 **foreach** $(m, n) \in \zeta$ **do** $W_{mn} \leftarrow \infty$
37 **if** $nmap_{\hat{v}} \neq NIL$ **then**
38 $Q_{\hat{u}\hat{v}} \leftarrow$CWSP($G, l, nmap_{\hat{v}}, b_{\hat{u}\hat{v}}, W$)
39 **else** $Q_{\hat{u}\hat{v}} \leftarrow \min_{\forall m \in L(\hat{v})} \{\text{CWSP}(G, l, m, b_{\hat{u}\hat{v}}, W)\}$
40 **if** $\exists \hat{v} \in \mathcal{N}(\hat{u}) : Q_{\hat{u}\hat{v}} = \phi$ **then** $c \leftarrow \infty$
41 **else** $c \leftarrow \sum_{\forall \hat{v} \in \mathcal{N}(\hat{u})} Q_{\hat{u}\hat{v}}$
42 **if** $c < c_{best}$ **then**
43 $best_{\hat{u}} \leftarrow l, Q^{best}_{\hat{u}} \leftarrow Q_{\hat{u}}, c_{best} \leftarrow c$
44 **if** $best_{\hat{u}} = NIL$ **then return** $\{\phi, \phi, \phi, \phi\}$
45 $nmap_{\hat{u}} \leftarrow best_{\hat{u}}$
46 **foreach** $\hat{v} \in \mathcal{N}(\hat{u})$ ***and*** $nmap_{\hat{v}} \neq NIL$ **do**
47 $emap_{\hat{u}\hat{v}} \leftarrow Q^{best}_{\hat{u}\hat{v}}, \Lambda_{\hat{u}\hat{v}} \leftarrow Cost(Q^{best}_{\hat{u}\hat{v}})$
48 $\{backup, S\} \leftarrow$ UpdateBackup($\hat{G}, backup, SRG, K$)
49 **return** $\{nmap, emap, backup, S\}$

Algorithm 2: Compute Backup VPath of a VLink

1 **function** *GetBackup($\hat{G}, (\hat{u}, \hat{v}), \Lambda, emap, k$)*
2 **foreach** $(\hat{x}, \hat{y}) \in \hat{E}$ **do**
3 **if** $(\exists j | j < k$ *and* $(\hat{x}, \hat{y}) \in backup^{est}_{\hat{u}\hat{v}}(j))$ **then** $Weight^{est}_{\hat{x}\hat{y}} \leftarrow \infty$
4 **else if** $T_{\hat{x}\hat{y}}[K-1] \geq b_{\hat{u}\hat{v}}$ **then** $Weight^{est}_{\hat{x}\hat{y}} \leftarrow 1$
5 **else if** $emap_{\hat{x}\hat{y}} = \phi$ ***or*** $\min_{(u,v) \in Q_{\hat{x}\hat{y}}} b^{residual}_{uv} \geq b_{\hat{u}\hat{v}}$ **then**
6 $S^{temp}_{\hat{x}\hat{y}} \leftarrow b_{\hat{u}\hat{v}} + \sum_{0 \le i < \min(K-1,\, T_{\hat{x}\hat{y}}.Size())} T_{\hat{x}\hat{y}}[i]$
7 $Weight^{est}_{\hat{x}\hat{y}} \leftarrow (S^{temp}_{\hat{x}\hat{y}} - S^{est}_{\hat{x}\hat{y}}) \times \Lambda_{\hat{x}\hat{y}}$
8 **else** $Weight^{est}_{\hat{x}\hat{y}} \leftarrow \infty$
9 $Weight^{est}_{\hat{u}\hat{v}} \leftarrow \infty$
10 **return** CWSP($\hat{G}, \hat{u}, \hat{v}, b_{\hat{u}\hat{v}}, Weight^{est}$)

computation. $T_{\hat{u}\hat{v}}$ will contain, in descending order, the bandwidths of all the VLinks in $\hat{\mathcal{H}}_{\hat{u}\hat{v}}$. Alg. 1 then proceeds to map the VNodes from the most constrained to the least constrained ones, *i.e.*, in decreasing order of their degrees. If two VNodes have equal degrees then we arbitrarily select one of them. For a VNode $\hat{u}$, Alg. 1 first finds K estimated backup VPaths for each VLink incident to $\hat{u}$ by iteratively invoking GetBackup procedure (Alg. 2). Alg. 2 invokes Constrained Weighted Shortest Path (CWSP) procedure to compute a VPath with at least $b_{\hat{u}\hat{v}}$ bandwidth between $\hat{u}$ and $\hat{v}$ in the VN $\hat{G}$, according to a weight function, $Weight^{est}$.

Alg. 2 first computes the weight function $Weight^{est}$ for all the VLinks and invokes CWSP procedure to obtain k_{th} backup VPath between $\hat{u}$ and $\hat{v}$. For the VLink $(\hat{u}, \hat{v})$, Alg. 2 assigns infinite weights to all the VLinks which have been used by the other backup VPaths of $(\hat{u}, \hat{v})$ to avoid having them appear again in the current backup VPath (Line 3). It gives lower weights to a VLink $(\hat{x}, \hat{y})$, if at least K VLinks in $\hat{\mathcal{H}}_{\hat{x}\hat{y}}$ have bandwidth larger than $b_{\hat{u}\hat{v}}$. This means that $S^{est}_{\hat{x}\hat{y}}$ is already sufficient to serve the bandwidth of K VLinks impacted by K SLink failures, thus allowing $(\hat{u}, \hat{v})$ to share the assigned spare bandwidth $S^{est}_{\hat{x}\hat{y}}$ without increasing it (Line 4). On the other hand, if $\hat{\mathcal{H}}_{\hat{x}\hat{y}}$ has less than K VLinks (i.e., $|T_{\hat{x}\hat{y}}| < K$) or if $b_{\hat{u}\hat{v}}$ is larger than at least one of the first K largest bandwidths of $\hat{\mathcal{H}}_{\hat{x}\hat{y}}$, $S^{est}_{\hat{x}\hat{y}}$ will increase as a result of using $(\hat{x}, \hat{y})$ in the kth backup VPath between $\hat{u}$ and $\hat{v}$. The increased spare capacity requirement is represented by $S^{temp}_{\hat{x}\hat{y}}$ and the amount of increase is $b_{\hat{u}\hat{v}}$ in the first case or the difference between $b_{\hat{u}\hat{v}}$ and the K_{th} element of $T_{\hat{x}\hat{y}}$ in the second case. The weight function of CWSP takes the possibility of increase in the spare capacity requirement and the mapping cost $\Lambda_{\hat{x}\hat{y}}$ of an already mapped VLink $(\hat{x}, \hat{y})$ into account and assigns $(\hat{x}, \hat{y})$ a weight proportional to both of these. In line 7, a special case occurs when a VLink $(\hat{x}, \hat{y})$ is not yet mapped. In this case, $\Lambda_{\hat{x}\hat{y}}$ is set to use the mean SPath length (σ) as an indicator of future cost (Line 5 of Alg. 1). Finally, an infinite weight is set to the VLinks whose mapped SPaths do not have adequate residual capacity to exclude them from the search space (Line 8 of Alg. 2).

After computing each estimated backup VPath $backup^{est}_{\hat{u}\hat{v}}(k)$, Alg. 1 updates $S^{est}_{\hat{x}\hat{y}}$ for all $(\hat{x}, \hat{y}) \in backup^{est}_{\hat{u}\hat{v}}(k)$. To do so, Alg. 1 first inserts $b_{\hat{u}\hat{v}}$ into the right position of the decreasingly sorted queue $T_{\hat{x}\hat{y}}$.

Following (28), Alg. 1 computes $S^{est}_{\hat{x}\hat{y}}$ as the summation of either the first K largest elements of $T_{\hat{x}\hat{y}}$ or all the elements of $T_{\hat{x}\hat{y}}$ if $|T_{\hat{x}\hat{y}}| < K$ (Line 17). It then places $(\hat{u}, \hat{v})$ and $(\hat{x}, \hat{y})$ into different SRGs (Line 20). It also places $(\hat{u}, \hat{v})$ and all other VLinks that use $(\hat{x}, \hat{y})$ in their backup VPaths into different SRGs (Line 25). Finally, it places $(\hat{x}, \hat{y})$ and any VLink $(\hat{a}, \hat{b})$ present in any of the other $K-1$ backup VPaths of $(\hat{u}, \hat{v})$ into different SRGs. After finding the estimated backup VPaths and SRGs of all the incident VLinks of a VNode $\hat{u}$, Alg. 1 finds the mapping of $\hat{u}$ and VLinks incident to $\hat{u}$. It iterates over all candidate SNodes $l \in L(\hat{u})$ and selects the one that results in the least cost mapping for all the VLinks incident to $\hat{u}$ (Line $33-44$). For a specific $l \in L(\hat{u})$ and $\hat{v} \in \mathcal{N}(\hat{u})$, if $\hat{v}$ is already mapped to $nmap_{\hat{v}}$, Alg. 1 computes CWSP from l to $nmap_{\hat{v}}$ (Line 39), while satisfying capacity constraints and SRG constraints in the SN (using the weights in W). To do so, Alg. 1 identifies the set of SLinks that the mapping of $(\hat{u}, \hat{v})$ should be disjoint from and assigns ∞ as their weights (Line 37). On the other hand, if $\hat{v}$ is not mapped yet, it computes CWSPs from l to the SNodes $m \in L(\hat{v})$ and selects the CWSP with the minimum cost (Line 40). After mapping a VNode $\hat{u}$, Alg. 1 maps the VLinks whose both endpoints have already been mapped and updates Λ of the mapped VLinks (Line 48). Upon mapping all the VLinks of a VN, Alg. 1 returns $nmap$, $emap$, $backup$, and S representing the VNode mapping, VLink mapping, backup VPaths, and spare backup capacities, respectively.

B. Reconfiguring the Allocated Spare Backup Bandwidth

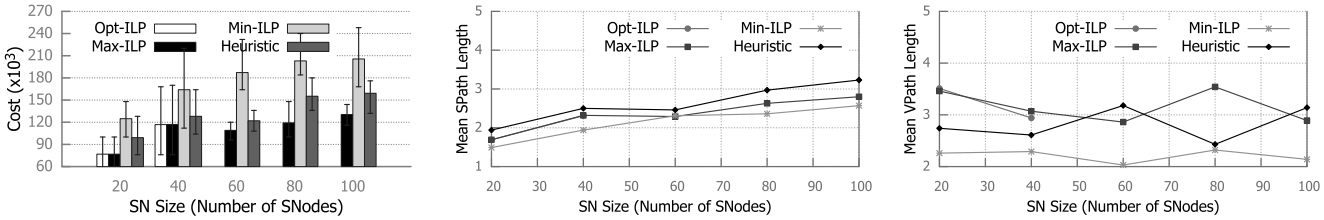
The last phase (Alg. 3) of our heuristic employs different techniques to further optimize the spare capacity allocation. Since Alg. 1 performs spare capacity assignment and embedding of VLinks sequentially based on some estimation, it is possible that spare capacity of initial VLinks may have been allocated using partial backup VPath selection and VLink embedding information. Once complete information is available, sharing of spare capacity can be further enhanced by taking into account SRGs of different failure scenarios and backup VPath multiplexing combinations [47], [48]. Due to space constraints, Alg. 3 illustrates the spare capacity optimization for only single (i.e., $K = 1$) and double (i.e., $K = 2$) link failure scenarios. Techniques similar to [47] and [48] can be adopted to optimize spare capacity for higher (i.e., $K > 2$) failure scenarios albeit the expected huge number of SRG and VPath multiplexing combinations.

For single SLink failure, Alg. 3 leverages p-cycle based protection to optimize the spare backup bandwidth $S_{\hat{u}\hat{v}}$ for each mapped VLink $(\hat{u}, \hat{v}) \in \hat{E}$ [49]. Alg. 3 first finds the longest cycle $\hat{R}$ in $\hat{G}$ such that no pairs of VLinks in $\hat{R}$ share an SLink in their mappings (Line 4). Recall from § III-D that each $(\hat{x}, \hat{y}) \in \hat{R}$ belongs to distinct SRGs for $K = 1$. Therefore, Alg. 3 allocates the maximum of the demands of all the VLinks in $\hat{R}$ to each $S_{\hat{x}\hat{y}} \in \hat{R}$ (Line 6). It then recomputes backup VPath $backup_{\hat{u}\hat{v}}$ for each $(\hat{u}, \hat{v}) \in \hat{E}$ using a process similar to Alg. 2. However, Alg. 3 utilizes VLink embedding information to better compute the backup VPaths. It does so by setting ∞ as the weight of the VLink $(\hat{x}, \hat{y})$ if $(\hat{u}, \hat{v})$ and $(\hat{x}, \hat{y})$ are in the same SRG (Line 9). Alg. 3 also enhances the spare capacity sharing by setting unit weights to the VLinks having already assigned spare capacities (Line 10). Alg. 3 then invokes CWSP procedure with the weight function to compute $backup_{\hat{u}\hat{v}}$ (Line 13). Finally, $S_{\hat{x}\hat{y}}$ for each $(\hat{x}, \hat{y}) \in backup_{\hat{u}\hat{v}}$ is updated accordingly (Line 15).

For two SLink failures, Alg. 3 utilizes backup VPath multiplexing to optimize spare backup bandwidth allocation [47], [48]. Backup multiplexing is based on the observation that not all backup VPaths are used simultaneously for certain failure scenarios, thus allowing to share the spare bandwidth allocated to the common VLinks in those VPaths. For instance, if two VLinks $(\hat{u}, \hat{v}) \in \hat{E}$ and $(\hat{a}, \hat{b}) \in \hat{E}$ do not share any SLink in their SPaths and none of $(\hat{u}, \hat{v})$ and $(\hat{a}, \hat{b})$ is present in the first backup VPaths of the other one, two SLink failures can only impact either both $(\hat{u}, \hat{v})$ and $(\hat{a}, \hat{b})$ or one of $(\hat{u}, \hat{v})$ and $(\hat{a}, \hat{b})$ and one of the VLinks in the first backup VPaths of $(\hat{u}, \hat{v})$ and $(\hat{a}, \hat{b})$. In the

Algorithm 3: Reconfigure Spare Capacities of All VLinks

1 **function** *UpdateBackup*($\hat{G}, backup, SRG, K$)
2 **if** $K = 1$ **then**
3 **foreach** $(\hat{u}, \hat{v}) \in \hat{E}$ **do** $S_{\hat{u}\hat{v}} \leftarrow 0$
4 $\hat{R} \leftarrow$ longest cycle in $\hat{G}$ such that no VLink pair in $\hat{R}$ shares an SLink on their mapped SPaths
5 **foreach** $(\hat{x}, \hat{y}) \in \hat{R}$ **do**
6 $S_{\hat{x}\hat{y}} \leftarrow \max_{\forall(\hat{u},\hat{v})\in\hat{R}}\{b_{\hat{u}\hat{v}}\}$
7 **foreach** $(\hat{u}, \hat{v}) \in \hat{E}$ **do**
8 **foreach** $(\hat{x}, \hat{y}) \in \hat{E} \setminus \{(\hat{u}, \hat{v})\}$ **do**
9 **if** $SRG_{\hat{u}\hat{v}} = SRG_{\hat{x}\hat{y}}$ **then** $Weight_{\hat{x}\hat{y}} \leftarrow \infty$
10 **else if** $S_{\hat{x}\hat{y}} \geq b_{\hat{u}\hat{v}}$ **then** $Weight_{\hat{x}\hat{y}} \leftarrow 1$
11 **else** $Weight_{\hat{x}\hat{y}} \leftarrow (b_{\hat{u}\hat{v}} - S_{\hat{x}\hat{y}})$
12 $Weight_{\hat{u}\hat{v}} \leftarrow \infty$
13 $backup_{\hat{u}\hat{v}} \leftarrow \text{CWSP}(\hat{G}, \hat{u}, \hat{v}, b_{\hat{u}\hat{v}}, Weight)$
14 **foreach** $(\hat{x}, \hat{y}) \in backup_{\hat{u}\hat{v}}$ **do**
15 $S_{\hat{x}\hat{y}} \leftarrow \max(S_{\hat{x}\hat{y}}, b_{\hat{u}\hat{v}})$
16 **else if** $K = 2$ **then**
17 **foreach** $((\hat{u}, \hat{v}), (\hat{a}, \hat{b})) \in \hat{E} \times \hat{E}$ *s.t.* $(\hat{u}, \hat{v}) \neq (\hat{a}, \hat{b})$ **do**
18 **if** $(\{(\hat{u}, \hat{v}) \cap backup^{est}_{\hat{a}\hat{b}}(1)\} = \phi$ *and* $\{(\hat{a}, \hat{b}) \cap backup^{est}_{\hat{u}\hat{v}}(1)\} = \phi$ *and* $\{Q_{\hat{u}\hat{v}} \cap Q_{\hat{a}\hat{b}}\} = \phi)$ **then**
19 **foreach** $(\hat{x}, \hat{y}) \in backup^{est}_{\hat{u}\hat{v}}(1)$ **do**
20 **if** $(\hat{x}, \hat{y}) \in backup^{est}_{\hat{a}\hat{b}}(2)$ **then**
21 $T_{\hat{x}\hat{y}}.DeQueue(\min(b_{\hat{u}\hat{v}}, b_{\hat{a}\hat{b}}))$
22 $S_{\hat{x}\hat{y}} \leftarrow \sum_{0\leq i<\min(K,\, T_{\hat{x}\hat{y}}.Size())} T_{\hat{x}\hat{y}}[i]$
23 **foreach** $(\hat{x}, \hat{y}) \in backup^{est}_{\hat{u}\hat{v}}(2)$ **do**
24 **if** $(\hat{x}, \hat{y}) \in backup^{est}_{\hat{a}\hat{b}}(1)$ **then**
25 $T_{\hat{x}\hat{y}}.DeQueue(\min(b_{\hat{u}\hat{v}}, b_{\hat{a}\hat{b}}))$
26 $S_{\hat{x}\hat{y}} \leftarrow \sum_{0\leq i<\min(K,\, T_{\hat{x}\hat{y}}.Size())} T_{\hat{x}\hat{y}}[i]$
27 **foreach** $(\hat{x}, \hat{y}) \in backup^{est}_{\hat{u}\hat{v}}(2)$ **do**
28 **if** $(\hat{x}, \hat{y}) \in backup^{est}_{\hat{a}\hat{b}}(2)$ **then**
29 $T_{\hat{x}\hat{y}}.DeQueue(\min(b_{\hat{u}\hat{v}}, b_{\hat{a}\hat{b}}))$
30 $S_{\hat{x}\hat{y}} \leftarrow \sum_{0\leq i<\min(K,\, T_{\hat{x}\hat{y}}.Size())} T_{\hat{x}\hat{y}}[i]$
31 **return** $\{backup, S\}$



(a) Cost Analysis (b) SPath Length Analysis (c) VPath Length Analysis

Fig. 4. Impact of SN size on single link failure protection.

first case, both the first VPaths of $(\hat{u}, \hat{v})$ and $(\hat{a}, \hat{b})$ are used together, whereas only the second VPath of either $(\hat{u}, \hat{v})$ or $(\hat{a}, \hat{b})$ is used in the second case. Therefore, the first VPath of $(\hat{u}, \hat{v})$ (or $(\hat{a}, \hat{b})$) and the second VPath of $(\hat{a}, \hat{b})$ (or $(\hat{u}, \hat{v})$) are not used together. The same is true for both the second VPaths of $(\hat{u}, \hat{v})$ and $(\hat{a}, \hat{b})$. Alg. 3 exploits such observations by reducing the spare bandwidths of the common VLinks in those VPaths that will not be used simultaneously for two SLink failure scenarios (Line $17-30$).

C. Running Time Analysis

CWSP procedure is the core of both Alg. 1 and Alg. 3. CWSP procedure is implemented using a modified *Dijkstra's shortest path* algorithm that takes into account the constraints and weights as depicted above. *Dijkstra's* algorithm using a min-priority queue on G runs in $O(|E| + |V| \log |V|)$ time. *CWSP* is invoked $O(|\hat{V}|\mathcal{L}\delta^2)$ times in Alg. 1, where $\mathcal{L}$ and δ are the maximum size of a location constraint set and maximum degree of a VNode, respectively. Therefore, the overall running time of the heuristic is $O(|\hat{V}|\mathcal{L}\delta^2(|E|+|V|\log|V|))$.

VI. Evaluation

We first describe the simulation setup and compared approaches in § VI-A. The performance metrics are defined in § VI-B. Finally, we describe our evaluation results focusing on: (i) evaluation for single link failure scenarios (§ VI-C), (ii) performances of double link failure scenarios (§ VI-E), and (iii) Comparison with Existing SN level survivability (§ VI-D).

A. Simulation Setup

We implement *Opt-ILP*, *Max-ILP*, and *Min-ILP* from § IV using IBM ILOG CPLEX libraries and compare them with a C++ implementation of the heuristic algorithm, referred as *Heuristic* from now on. For each simulation run, we generate an SN and 5 random VNs. For each SN, we take the mean value of each performance metric over the 5 VNs. For single failure scenarios ($K = 1$), SN and VN size is varied between 20 and 100 nodes and 3 and 11 nodes, with increments of 10 and 2, respectively. We change the connectivity of both SNs and VNs by varying the link to node ratio (LNR) from 1.12 to 3.00 and from 1.0 to 2.17, respectively to match with realistic topologies [50], [51]. We make sure VNs are 2- and 3-edge connected to survive single and double link failures, respectively. For double failure cases ($K = 2$), we start with VNs of size 4 since there is no simple graph of size less than 4 that has 3-edge connectivity. For $K = 2$, we vary VN sizes on an SN with size 25 and LNR 1.8. We also compare performances of *Heuristic* for $K = 1$ and $K = 2$ in large scale topologies. In these cases, VN sizes vary between 10 and 100 nodes with 21 and 285 links on SNs with 500 and 1000 nodes with 2016 and 4022 links. Given the number of nodes and links of a problem instance, the source and destination of an SLink or a VLink are decided randomly. Location constraints of VNodes are chosen randomly, and VLink demands are set to 10% of the SLink bandwidths. Simulations are performed on a machine with 2×8-core 2.0 Ghz Intel Xeon E5-2650 processors and 256GB of RAM.

B. Performance Metrics

1) Cost: The cost of embedding a VN computed using (20). We set a unit cost for allocating bandwidth on an SLink, therefore, (20) directly represents resource consumption.

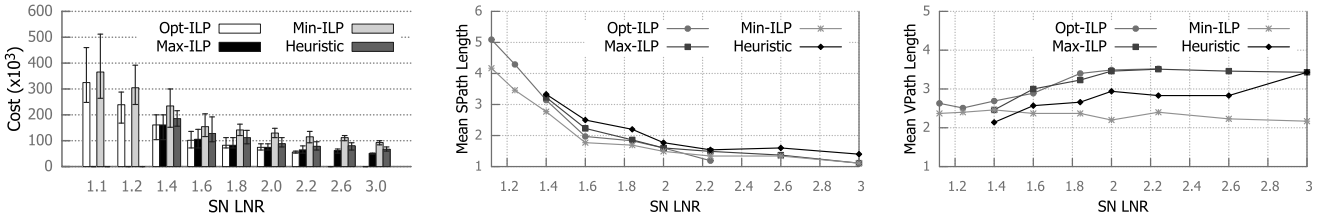
2) Mean SPath Length: The mean length of the SPath used to map a VLink of a VN.

3) Mean VPath Length: The mean length of the VPath used as a backup path for a VLink in a VN.

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

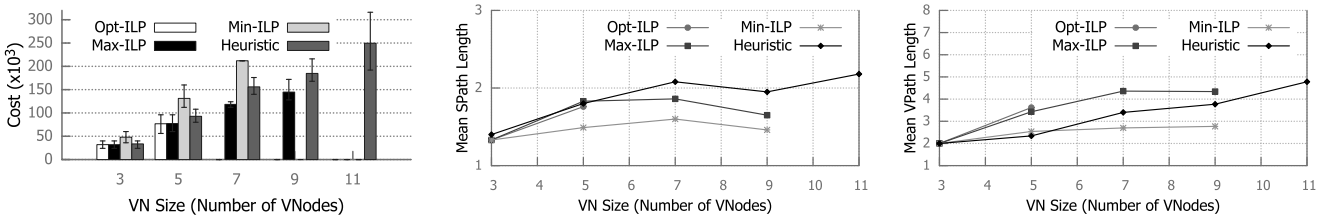
C. Evaluation Results for Single SLink Failure Scenarios

1) Impact of SN Topology: Fig. 4 presents performance metrics for different SN sizes, while keeping the VN size and SN LNR fixed at 5 and 1.8, respectively. Since SNode degrees remain the same, SN diameters increase with increasing SN size. We see that embedding costs in Fig. 4(a) increase for all the compared approaches with increasing SN size. This stems from the fact that the location constraints of the VNodes of a VN are chosen to be far apart from one another in an SN with larger diameter. This behavior is verified in Fig. 4(b) that shows the increase in SPath lengths with the increase in SN sizes. Longer SPaths require higher amount of substrate resources for embedding the VLinks of a VN, hence, the higher cost. Another takeaway from Fig. 4(b) is that mean SPath lengths of *Min-ILP* and *heuristic* are the lowest and highest, respectively. *Min-ILP* selects shorter SPaths for embedding VLinks because of the lower number of disjoint path constraints imposed by the shorter VPaths as shown in Fig. 4(c). *Min-ILP* prefers the shorter VPaths since it has to allocate dedicated spare capacity on all the VLinks in a



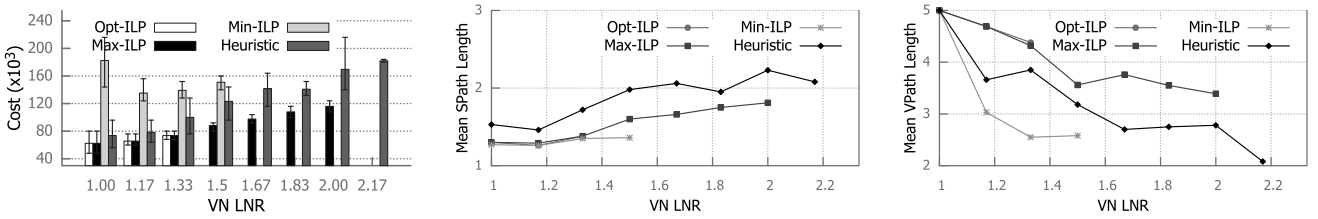
(a) Cost Analysis (b) SPath Length Analysis (c) VPath Length Analysis

Fig. 5. Impact of SN density on single link failure protection.



(a) Cost Analysis (b) SPath Length Analysis (c) VPath Length Analysis

Fig. 6. Impact of VN size on single link failure protection.



(a) Cost Analysis (b) SPath Length Analysis (c) VPath Length Analysis

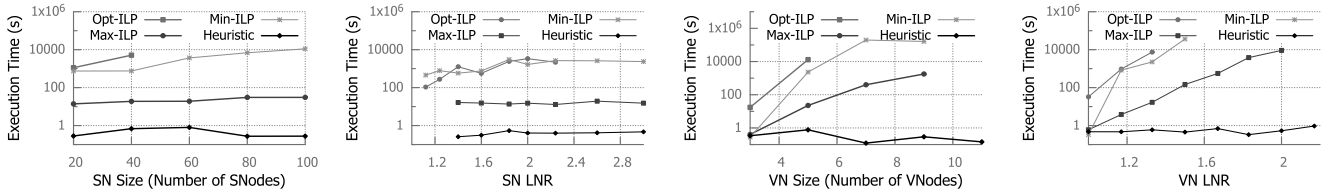
Fig. 7. Impact of VN density on single link failure protection.

VPath. In contrast, all other approaches choose longer VPaths to maximize the spare capacity sharing on VLinks.

Fig. 5 presents the impact of SN density on the performance metrics by varying SN LNRs. For very sparse SNs, *i.e.*, SNs with very low LNRs, *Max-ILP* and *heuristic* fail to find sufficient disjoint SPaths, imposed by the SRG constraints, resulting in infeasible solutions. However, *Opt-ILP* and *Min-ILP* are able to find solutions with very high costs by reducing the number of SRGs (Fig. 5(a)). Lower number of SRGs, in turn, reduces the opportunities of spare capacity sharing on VPaths. Despite reducing SRGs, *Opt-ILP* and *Min-ILP* still have to find disjoint SPaths to satisfy survivability constraints. In a sparse SN, a set of disjoint SPaths between pairs of SNodes becomes significantly longer than the set of non-disjoint SPaths between the same pairs of SNodes. These aforementioned factors contribute to the higher embedding costs for sparse SNs. As SN LNR increases, the number and length of the SPaths (also disjoint SPaths) increase and decrease, respectively, hence the decrease in mean SPath lengths in Fig. 5(b). As a consequence, cost decreases for all the approaches as shown in Fig. 5(a). Fig. 5(c) shows that mean VPath lengths for all the approaches except *Min-ILP* increase initially with increasing SN LNR. When VPath lengths get closer to the VN diameter in these approaches, they remain almost constant with increasing SN LNR. The initial increase in mean VPath length is due to the use of the same VLinks by more VPaths, leading to more spare capacity sharing. However, sparse SNs cannot satisfy the SRG constraints imposed by longer VPaths, therefore, all the approaches except *Min-ILP* select shorter VPaths. Since *Min-ILP* does not allow any spare capacity sharing, there is little impact of SN density on VPath lengths.

2) Impact of VN Topology: Fig. 6 presents performance metrics for different VN sizes, while keeping the SN size, SN LNR, and VN LNR fixed at 50, 1.82, and 1.4, respectively. Fig. 6(a) shows that cost increases as VN size gets larger. This is partially due to the allocation of more SN resources for embedding higher number of VLinks of a larger VN. The other contributor is the longer embedding SPath as shown in Fig. 6(b). SPath length increases with VN size to find the higher number of disjoint SPaths that are required to satisfy the higher number of SRG and survivability constraints induced by longer VPaths shown in Fig. 6(c). In case of *Min-ILP*, VPath lengths increase due to rise in VN diameter resulting from increase in VN size. The other approaches select even longer VPaths to enable more spare capacity sharing on the VLinks of the selected VPaths.

Fig. 7 compares performance metrics by varying the VN LNR, while keeping the SN size, SN LNR, and VN size fixed at 50, 1.82, and 6, respectively. The takeaway from Fig. 7(a), Fig. 7(b), and Fig. 7(b) is that both cost and mean SPath length increase with increasing VN LNR, whereas



(a) Varying SN Size (b) Varying SN Density (c) Varying VN Size (d) Varying VN Density

Fig. 8. Scalability analysis for single link failure protection.

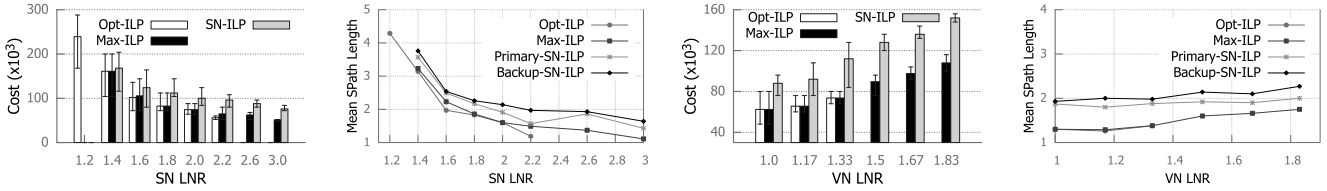
mean VPath length shows an opposite trend. We explain these behaviors as follows. The sparsest connected VN is a ring topology. In such a topology, each VLink $(\hat{u}, \hat{v})$ has exactly one VPath that contains all the other VLinks in the ring except $(\hat{u}, \hat{v})$, and all the VLinks in the ring have to be embedded disjointedly to satisfy the survivability constraints. Hence, all the approaches have the same mean VPath length for the sparsest VN, *i.e.*, VN with the lowest LNR. The results of ring VNs also illustrate that *Min-ILP* allocates ~ 3 times extra resources to guarantee the same level of survivability provided by the other approaches. As VN LNR increases, *Min-ILP* prefers the shortest possible VPaths to minimize the amount of additional resources. Other approaches strike a balance between reducing disjoint path requirements for satisfying survivability constraints and increasing spare capacity sharing as well as disjoint path requirements of SRG constraints induced by shorter and longer VPaths, respectively. Hence, the mean VPath lengths of all the approaches leveraging spare capacity sharing decrease much slower than those of *Min-ILP*, which does not allow any sharing. Despite a decrease in VPath lengths as VN LNR increases, all the approaches except *Min-ILP* have to satisfy higher number of SRG constraints to enable more spare capacity sharing. This contributes to the increasing trend of cost and mean SPath length.

3) Discussion of Results for Max-ILP and Min-ILP: As we see in Fig. 4-8, *Max-ILP* closely approximates *Opt-ILP*, whereas *Min-ILP* provides the upper bound of cost. The proximity between *Max-ILP* and *Opt-ILP* can be explained using the analysis presented in § IV-C.3 and mean SPath length values. Since SPath lengths of *Max-ILP* are always less than twice the SPath lengths of *Min-ILP*, *Max-ILP* yields the lower cost. Hence, for embedding a VLink, *Opt-ILP* prefers spare capacity sharing of *Max-ILP* than dedicated spare capacity allocation of *Min-ILP*. Despite the sequential nature of *Heuristic*, its performance remains closer to those of *Max-ILP* than *Min-ILP*. Specifically, compared to *Max-ILP*, *Heuristic* provisions $\sim$21% additional resources on average.

4) Scalability Analysis: Fig. 8 demonstrates the scalability of the compared approaches by plotting execution times in logarithmic scale. As we can see, *Opt-ILP* only scales to very small networks. The reduction in problem complexities in terms of constraints and variables in *Max-ILP* and *Min-ILP* are reflected in their ability to scale to larger problem instances than *Opt-ILP*. Among them *Min-ILP* has to satisfy the lowest number of disjoint path constraints. It explores a much larger solution space than *Max-ILP*, requiring more time. Following our discussion on problem complexities in § IV-C, VN dimension has a more profound impact on the scalability of the ILP-based approaches than SN dimension. We observe this impact in Fig. 8(c) and Fig. 8(d) that show, with our current hardware, the ILP-based approaches hit ceilings in terms of VN size or VN LNR. Even for the successful cases, the ILP-based approaches require several orders of magnitude more time to solve similar problem instances compared to *Heuristic* that can scale to much larger problem instances.

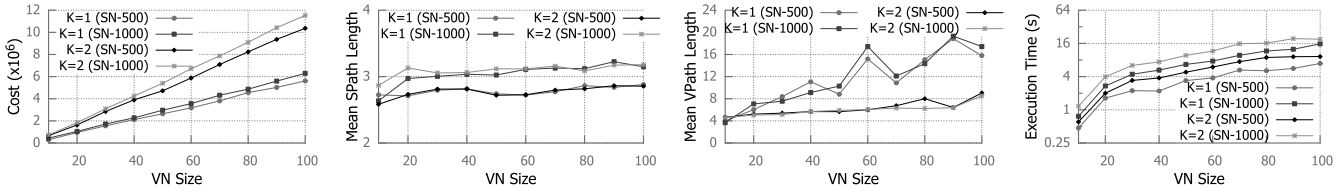
D. Comparison With Existing SN Level Survivability [13]

One of the key motivations for embedding a VN augmented with spare capacity is the hypothesis that more resource efficient embedding is possible with protection at the VN layer compared to doing the same at the SN layer. We provide a quantitative comparison to support this hypothesis. For this scenario, we have explored the SVNE literature to find a representative VN embedding approach that adopts SN level shared spare capacity allocation and found the proposal in [13] as the closest match. However, Chen*et al.* [13] only provide a heuristic algorithm that yields suboptimal solutions. Hence, we have formulated an ILP, namely *SN-ILP*, following the concepts presented in [13] for solving the exact problem of VN embedding with SN layer shared spare capacity allocation and used it as a baseline for comparison. We implement *SN-ILP* using IBM ILOG CPLEX library. For each VLink in a VN, *SN-ILP* allocates SN resources on a primary and a backup SPath disjoint from the primary. However, the spare capacity allocated on the backup SPaths can be shared among the VLinks whose primary SPaths are edge disjoint. For a fair comparison, we juxtapose the performance metrics of *SN-ILP* with those of our ILP formulations that share spare capacity, *i.e.*, *Opt-ILP* and *Max-ILP* in Fig. 9. Fig. 9(a) reveals that, similar to *Max-ILP*, *SN-ILP* yields infeasible solutions for sparse SNs due to lack of path diversity. In all the cases when *SN-ILP* is able to find a solution as shown in Fig. 9(a) and Fig. 9(c), *SN-ILP* incurs higher costs than both *Opt-ILP* and *Max-ILP*. In these cases, *SN-ILP* allocates, on average, $\sim$33% additional resources compared to *Max-ILP*. We explain this behavior with the help of Fig. 9(b) and Fig. 9(d). These figures show that both primary and backup SPath lengths of *SN-ILP* exceed SPath lengths of *Opt-ILP* and *Max-ILP*. This is due to the fact that finding a pair of disjoint SPaths between a pair of SNodes is more difficult than finding an SPath that is disjoint from a set of SPaths between different set of SNodes [52]. Another reason for lower costs of *Opt-ILP*



(a) Cost Analysis (b) SPath Length Analysis (c) Cost Analysis (d) SPath Length Analysis

Fig. 9. Comparison with SN level survivability of [13].



(a) Cost Analysis (b) SPath Length Analysis (c) VPath Length Analysis (d) Execution Time Analysis

Fig. 10. Performance of heuristic with single ($K = 1$) and double ($K = 2$) link failure protections on large scale test cases.

TABLE I
EVALUATION RESULTS FOR DOUBLE ($K = 2$) LINK FAILURES

VNode Count	Cost (ILP)	Cost (Heuristic)	SPath Length (ILP)	SPath Length (Heuristic)	VPath Length (ILP)	VPath Length (Heuristic)
4	112×10^3	127×10^3	1.57	1.77	2.1	2.0
5	158×10^3	184×10^3	1.65	1.93	2.47	2.25
6	-	247×10^3	-	2.0	-	2.4

and *Max-ILP* compared to *SN-ILP* is that *Opt-ILP* and *Max-ILP* employ more spare capacity sharing on the VPaths than spare capacity sharing on the SPaths employed by *SN-ILP*.

E. Evaluation Results for Multiple Link Failure Scenarios

1) Small Scale Performance: Table I compares the performances of ILP formulation presented in § IV-D and *Heuristic* for $K = 2$. As evident from the table, the ILP formulation does not scale beyond 5 node VNs due to the combinatorial number of constraints for $K = 2$. On the other hand, *Heuristic* scales to larger topologies at the expense of incurring higher cost. As seen in the table, the higher cost of *Heuristic* than ILP is attributed to the longer SPaths for embedding VLinks and less sharing of spare capacity along shorter VPaths. Nonetheless, the average cost of *Heuristic* remains within 20% of the ILP.

2) Large Scale Performance: In Fig. 10, we present the performances of the *Heuristic* with $K = 1$ and $K = 2$ for large scale topologies, where multiple failures are more likely to occur [28], [50]. Fig. 10(a) shows that cost increases proportionally with the increase in VN size, following the same trend observed in Fig. 6(a). The $\sim$ 100% increase in embedding cost of $K = 2$ from $K = 1$ is due to the increase in spare backup bandwidth requirements to survive double link failures. The slightly higher embedding cost for an SN with 1000 nodes than for an SN with 500 nodes in both $K = 1$ and $K = 2$ cases is due to the increase in SN diameter similar to the cases in Fig. 4(a). The increase in SN diameter is reflected in Fig. 10(b) that shows SPaths are longer in a 1000 node SN than in the 500 node one. Different behavior is observed in Fig. 10(c) that shows VPath lengths of $K = 1$ are much larger than those of $K = 2$. This is due to adopting different reconfiguration strategies for different failure scenarios by Alg. 3. For $K = 1$, the goal of Alg. 3 is to assign spare bandwidth along the longest p-cycle, thus resulting in longer VPaths. Such strategy can be expensive for $K > 1$, since K disjoint p-cycles are needed to survive K SLink failures. Furthermore, p-cycle based strategy offers less room for spare capacity sharing for $K > 1$, as two VLinks on the p-cycle are on the backup VPaths of each other. Therefore, for $K = 2$, Alg. 3 prefers shorter VPaths as shown in Fig. 10(c) and optimizes spare capacity through backup multiplexing technique described in Alg. 3. Finally, Fig. 10(d) shows that our *Heuristic* is able to find solutions within a reasonable time limit in compliance with the running time analysis presented in § V-C.

VII. CONCLUSION AND FUTURE WORK

In this paper, we have studied the SVNE problem that arises in network virtualization *a.k.a.* network slicing from a new perspective. Instead of managing failure survivability at the SN level, we have proposed to delegate the failure management tasks to the VN level in support of more autonomous VNs. Hence, a new variety of SVNE problem emerges that requires a VN to be augmented with sufficient spare backup capacity and embedded on the SN accordingly to ensure survivability against multiple substrate link failures. For the case of single substrate link failure, we have formulated the optimal solution to this joint optimization problem as a QIP and transformed it into an ILP. We have presented simplified ILP formulations to solve special cases of the problem and presented a mathematical study to analyze the impact of SN topology on the level of spare capacity sharing. We have then extended the most resource efficient ILP formulation for single link failure

to handle the case of multiple substrate link failures. We have also proposed a heuristic algorithm to tackle the computational complexity of the ILP formulations. The heuristic algorithm provides a generic framework to survive multiple link failures and demonstrates spare capacity optimization techniques for single and double link failure scenarios.

We have performed simulations to evaluate our solutions for single and double link failure scenarios. Simulation results show that ILP formulations for the two special cases can provide upper and approximately lower bounds of the solution spectrum. Moreover, the heuristic allocates ∼21% additional resources compared to the approximated lower bound, while executing several orders of magnitude faster. Large scale simulations of the heuristic algorithm shows that protection against double link failures comes at the cost of nearly two times of the resources incurred by protection against single link failures. A quantitative comparison between the SVNE with VN level protection and the traditional SVNE with SN level protection reveals that the former can save ∼33% resources on average, compared to those required by the latter.

In the future, we plan to extend this work to virtual fabrics where the bandwidth requirement is expressed as pairwise bandwidth rather than as per link bandwidth requirement. We intend to study the resource allocation challenges to ensure survivability at the fabric level compared to doing the same at the VN level. We also plan to extend the failure scenario to consider multiple substrate node failures.

REFERENCES

[1] *Amazon Virtual Private Cloud*. Accessed: Oct. 1, 2017. [Online]. Available: http://aws.amazon.com/vpc/

[2] *T-SDN Prototype Demonstration*. Accessed: Oct. 1, 2017. [Online]. Available: https://www.opennetworking.org/images/stories/downloads/sdn-resources/technical-reports/oif-p0105_031_18.pdf

[3] N. Nikaein *et al.*, "Network store: Exploring slicing in future 5G networks," in *Proc. ACM MobiArch*, 2015, pp. 8–13.

[4] P. Rost *et al.*, "Mobile network architecture evolution toward 5G," *IEEE Commun. Mag.*, vol. 54, no. 5, pp. 84–91, May 2016.

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

[6] M. R. Rahman, I. Aib, and R. Boutaba, "Survivable virtual network embedding," in *Proc. IFIP Netw.*, 2010, pp. 40–52.

[7] J. Xu, J. Tang, K. Kwiat, W. Zhang, and G. Xue, "Survivable virtual infrastructure mapping in virtualized data centers," in *Proc. IEEE CLOUD*, Jun. 2012, pp. 196–203.

[8] H. Yu, V. Anand, C. Qiao, and G. Sun, "Cost efficient design of survivable virtual infrastructure to recover from facility node failures," in *Proc. IEEE ICC*, Jun. 2011, pp. 1–6.

[9] N. Shahriar *et al.*, "Connectivity-aware virtual network embedding," in *Proc. IFIP Netw.*, May 2016, pp. 46–54.

[10] M. M. A. Khan, N. Shahriar, R. Ahmed, and R. Boutaba, "Simple: Survivability in multi-path link embedding," in *Proc. IEEE/ACM/IFIP CNSM*, Nov. 2015, pp. 210–218.

[11] R. R. Oliveira *et al.*, "Dos-resilient virtual networks through multipath embedding and opportunistic recovery," in *Proc. ACM SAC*, 2013, pp. 597–602.

[12] T. Guo, N. Wang, K. Moessner, and R. Tafazolli, "Shared backup network provision for virtual network embedding," in *Proc. IEEE ICC*, Jun. 2011, pp. 1–5.

[13] Y. Chen, J. Li, T. Wo, C. Hu, and W. Liu, "Resilient virtual network service provision in network virtualization environments," in *Proc. IEEE ICPADS*, Dec. 2010, pp. 51–58.

[14] Z. Ye, A. N. Patel, P. N. Ji, and C. Qiao, "Survivable virtual infrastructure mapping with dedicated protection in transport software-defined networks [invited]," *J. Opt. Commun. Netw.*, vol. 7, no. 2, pp. A183–A189, 2015.

[15] S. R. Chowdhury *et al.*, "Protecting virtual networks with drone," in *Proc. IEEE/IFIP NOMS*, Apr. 2016, pp. 78–86.

[16] S. Herker, A. Khan, and X. An, "Survey on survivable virtual network embedding problem and solutions," in *Proc. ICNS*, 2013, pp. 1–6.

[17] A. Fischer, J. F. Botero, M. T. Beck, H. de Meer, and X. Hesselbach, "Virtual network embedding: A survey," *IEEE Commun. Surveys Tuts.*, vol. 15, no. 4, pp. 1888–1906, 4th Quart., 2013.

[18] S. Ayoubi, Y. Chen, and C. Assi, "Towards promoting backup-sharing in survivable virtual network design," *IEEE/ACM Trans. Netw.*, vol. 24, no. 5, pp. 3218–3231, Oct. 2016.

[19] D. D.-J. Kan, A. Narula-Tam, and E. Modiano, "Lightpath routing and capacity assignment for survivable ip-over-wdm networks," in *Proc. IEEE DRCN*, Oct. 2009, pp. 37–44.

[20] Y. Liu, D. Tipper, K. Vajanapoom, Y. Liu, D. Tipper, and K. Vajanapoom, "Spare capacity allocation in two-layer networks," *IEEE J. Sel. Areas Commun.*, vol. 25, no. 5, pp. 974–986, Jun. 2007.

[21] E. Kubilinskas and M. Pioro, "Two design problems for the IP/MPLS over WDM networks," in *Proc. DRCN*, 2005, pp. 241–248.

[22] W. Wang *et al.*, "First demonstration of virtual transport network services with multi-layer protection schemes over flexi-grid optical networks," *IEEE Commun. Lett.*, vol. 20, no. 2, pp. 260–263, Feb. 2016.

[23] *OpenFlow-Enabled Transport SDN*. Accessed: Oct. 1, 2017. [Online]. Available: https://www.opennetworking.org/images/stories/downloads/sdn-resources/solution-briefs/sb-of-enabled-transport-sdn.pdf

[24] O. Gerstel and G. H. Sasaki, "Quality of protection (QoP): A quantitative unifying paradigm to protection service grades," *Opt. Netw. Mag.*, vol. 3, no. 3, pp. 40–49, 2002.

[25] P. Demeester *et al.*, "Resilience in multilayer networks," *IEEE Commun. Mag.*, vol. 37, no. 8, pp. 70–76, Aug. 1999.

[26] Y. Liu, D. Tipper, and P. Siripongwutikorn, "Approximating optimal spare capacity allocation by successive survivable routing," in *Proc. IEEE INFOCOM*, Apr. 2001, pp. 699–708.

[27] T. Lin, Z. Zhou, and K. Thulasiraman, "Logical topology survivability in IP-over-WDM networks: Survivable lightpath routing for maximum logical topology capacity and minimum spare capacity requirements," in *Proc. IEEE DRCN*, Oct. 2011, pp. 1–8.

[28] P. Gill, N. Jain, and N. Nagappan, "Understanding network failures in data centers: Measurement, analysis, and implications," in *Proc. ACM SIGCOMM*, 2011, pp. 350–361.

[29] A. Markopoulou, G. Iannaccone, S. Bhattacharyya, C.-N. Chuah, and C. Diot, "Characterization of failures in an IP backbone," in *Proc. IEEE INFOCOM*, Mar. 2004, pp. 2307–2317.

[30] N. Shahriar *et al.*, "Joint backup capacity allocation and embedding for survivable virtual networks," in *Proc. IFIP Netw.*, Jun. 2017, pp. 1–9.

[31] 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. IEEE GLOBECOM*, Dec. 2010, pp. 1–6.

[32] X. Liu, Y. Wang, A. Xiao, X. Qiu, and W. Li, "Disaster-prediction based virtual network mapping against multiple regional failures," in *Proc. IFIP IM*, May 2015, pp. 371–378.

[33] M. Pourvali *et al.*, "Progressive recovery for network virtualization after large-scale disasters," in *Proc. IEEE ICNC*, Feb. 2016, pp. 1–5.

[34] N. Shahriar *et al.*, "Generalized recovery from node failure in virtual network embedding," *IEEE Trans. Netw. Service Manage.*, vol. 14, no. 2, pp. 261–274, Jun. 2017.

[35] S. Ayoubi, C. Assi, Y. Chen, T. Khalifa, and K. B. Shaban, "Restoration methods for cloud multicast virtual networks," *J. Netw. Comput. Appl.*, vol. 78, pp. 180–190, Jan. 2017.

[36] I. B. Barla, D. A. Schupke, M. Hoffmann, and G. Carle, "Optimal design of virtual networks for resilient cloud services," in *Proc. IEEE DRCN*, Mar. 2013, pp. 218–225.

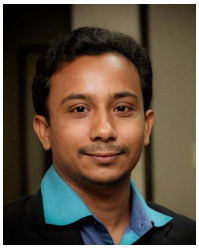
[37] Z. Zhou, T. Lin, K. Thulasiraman, G. Xue, and S. Sahni, "Novel survivable logical topology routing in IP-over-WDM networks by logical protecting spanning tree set," in *Proc. IEEE ICUMT*, Oct. 2012, pp. 650–656.

[38] K. Thulasiraman, T. Lin, M. Javed, and G. Xue, "Logical topology augmentation for guaranteed survivability under multiple failures in IP-over-WDM optical networks," *Opt. Switching Netw.*, vol. 7, no. 4, pp. 206–214, Dec. 2010.

[39] S. Even, A. Itai, and A. Shamir, "On the complexity of time table and multi-commodity flow problems," in *Proc. IEEE FOCS*, Oct. 1975, pp. 184–193.

[40] E. L. Lawler, "The quadratic assignment problem," *Manage. Sci.*, vol. 9, no. 4, pp. 586–599, 1963.

[41] E. Cela, *The Quadratic Assignment Problem: Theory and Algorithms*, vol. 1. New York, NY, USA: Springer, 2013.
[42] S. Sahni and T. Gonzalez, "P-complete approximation problems," *J. ACM*, vol. 23, no. 3, pp. 555–565, 1976.
[43] M. Oral and O. Kettani, "A linearization procedure for quadratic and cubic mixed-integer problems," *Oper. Res.*, vol. 40, no. 1, pp. S109–S116, 1992.
[44] D. A. Dunn, W. D. Grover, and M. H. MacGregor, "Comparison of k-shortest paths and maximum flow routing for network facility restoration," *IEEE J. Sel. Areas Commun.*, vol. 12, no. 1, pp. 88–99, Jan. 1994.
[45] W. Grover, "Distributed restoration of the transport network," *IEE Telecommun. Ser.*, vol. 30, p. 337, 1994.
[46] M.-C. Cai, "The maximal size of graphs with at most k edge-disjoint paths connecting any two adjacent vertices," *Discrete Math.*, vol. 85, no. 1, pp. 43–52, 1990.
[47] W. He, M. Sridharan, and A. K. Somani, "Capacity optimization for surviving double-link failures in mesh-restorable optical networks," *Photon. Netw. Commun.*, vol. 9, no. 1, pp. 99–111, 2005.
[48] W. He and A. K. Somani, "Path-based protection for surviving double-link failures in mesh-restorable optical networks," in *Proc. IEEE GLOBECOM*, vol. 5. Dec. 2003, pp. 2558–2563.
[49] R. Asthana, Y. N. Singh, and W. D. Grover, "P-cycles: An overview," *IEEE Commun. Surveys Tuts.*, vol. 12, no. 1, pp. 97–111, 1st Quart., 2010.
[50] R. Govindan, I. Minei, M. Kallahalla, B. Koley, and A. Vahdat, "Evolve or die: High-availability design principles drawn from googles network infrastructure," in *Proc. ACM SIGCOMM*, 2016, pp. 58–72.
[51] N. Spring, R. Mahajan, and D. Wetherall, "Measuring ISP topologies with Rocketfuel," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 32, no. 4, pp. 133–145, 2002.
[52] F. Iqbal and F. A. Kuipers, "Disjoint paths in networks," in *Wiley Encyclopedia of Electrical and Electronics Engineering*. Hoboken, NJ, USA: Wiley, 2015.



Nashid Shahriar (S'16) received the B.Sc. and M.Sc. degrees in computer science and engineering from the Bangladesh University of Engineering and Technology in 2011 and 2009, respectively. He is currently pursuing the Ph.D. degree with the School of Computer Science, University of Waterloo. His research interests include network virtualization, 5G networks, and network reliability. He received the David R. Cheriton Graduate Scholarship from the University of Waterloo.

Shihabur Rahman Chowdhury (S'13) received the B.Sc. degree in computer science and engineering from BUET in 2009. He is currently pursuing the Ph.D. degree with the School of Computer Science, University of Waterloo. His research interests include virtualization and softwarization of computer networks. He received the Ontario Graduate Scholarship, the Presidents Graduate Scholarship, and the GoBell Scholarship from the University of Waterloo.

Reaz Ahmed received the B.Sc. and M.Sc. degrees in computer science from BUET in 2002 and 2000, respectively, and the Ph.D. degree in computer science from the University of Waterloo in 2007. His research interests include future Internet, information-centric networks, network virtualization, and content sharing peer-to-peer networks with focus on search flexibility, efficiency, and robustness. He was a recipient of the IEEE Fred W. Ellersick Award in 2008.

Aimal Khan received the B.Sc. degree in computer science and engineering from the National University of Sciences and Technology, Pakistan. He is currently pursuing the master's degree in mathematics with the School of Computer Science, University of Waterloo. His research interests include network virtualization.

Siavash Fathi received the B.Sc. degree in computer engineering from the University of Tehran, Iran, in 2016. He is currently pursuing the master's degree in mathematics with the School of Computer Science, University of Waterloo. His research interests include future network architectures, network virtualization, and Internet of drones. He received the International Master's Student Award from the University of Waterloo.

Raouf Boutaba (F'12) received the M.Sc. and Ph.D. degrees in computer science from University Pierre & Marie Curie, Paris, in 1990 and 1994, respectively. He is currently a Professor of computer science with the University of Waterloo. His research interests include resource and service management in networks and distributed systems. He received several best paper awards and recognitions including the Premiers Research Excellence Award, the IEEE ComSoc Hal Sobol Award, the Fred W. Ellersick Award, the Joe LociCero Award, the Dan Stokesbury Award, the Salah Aidarous Award, and the IEEE Canada McNaughton Gold Medal. He was the Founding Editor-in-Chief of the IEEE TRANSACTIONS ON NETWORK AND SERVICE MANAGEMENT from 2007 to 2010 and on the editorial boards of other journals. He is a fellow of the Engineering Institute of Canada and the Canadian Academy of Engineering.

Jeebak Mitra received the M.A.Sc. and Ph.D. degrees in electrical engineering from The University of British Columbia in 2005 and 2010, respectively. From 2010 to 2011, he was a Senior System Engineer with Riot Micro, leading the system level design for a local thermal equilibrium baseband. From 2011 to 2012, he was a Team Leader for physical layer DSP design with BLINQ Networks, Ottawa, focusing on small cell backhaul products. Since 2013, he has been a Senior Staff Engineer with the Huawei Technologies Canada Research Center, Ottawa, in the areas of algorithm design and implementation for coherent high-speed optical transceivers and flexible optical networks. His research interests lie in the area of high-performance communication systems design focusing on optical and wireless networks. He received the Best Student Paper Award at the IEEE Canadian Conference in Electrical and Computer Engineering 2009. He was a co-recipient of the Best Paper Award at CNSM 2017.

Liu Liu received the M.Sc. and Ph.D. degrees in communication and information systems from the University of Electronic Science and Technology of China in 2011 and 2015, respectively. He was a Visiting Scholar of computer science and engineering with the State University of New York at Buffalo from 2012 to 2014. He joined Huawei as a Research Engineer in 2015. His research interests focus on network planning and optimization, uncertainty optimization, approximation algorithms, and cloud computing.