Survivable Virtual Network Design and Embedding to Survive a Facility Node Failure

Bingli Guo, Chunming Qiao, *Fellow, IEEE*, Jianping Wang, *Member, IEEE*, Hongfang Yu, Yongxia Zuo, Juhao Li, Zhangyuan Chen, *Member, IEEE*, and Yongqi He

***Abstract*—As virtualization is becoming a promising way to support various emerging application, provisioning survivability to requested virtual networks (VN) in a resource efficient way is important. In this paper, we investigate the survivable VN embedding (SVNE) problem from a new perspective. First, we consider the failure dependent protection (FDP) in which each primary facility node would have a different backup facility node, as opposed to the Failure Independent Protection (FIP) which has been studied before, in order to provide the same degree of protection against a single node failure with less substrate resources. Secondly, we enhance the VN with additional computing and communication resources and design the Enhanced VN (or EVN) before embedding it to the substrate in order to further reduce the amount of substrate resources needed to survive a single facility node failure. The work is the first that combines the FDP with EVN design (FD-EVN) to explore a resource efficient solution to the SVNE problem. After presenting a binary quadratic programming (BQP) formulation of the FD-EVN design problem and a Mixed Integer Linear Programming (MILP) formulation of the EVN embedding (EVNE) problem, we propose two heuristic algorithms for FD-EVN design, as well as an EVNE algorithm that explores primary and backup substrate resources sharing. Simulations are conducted to evaluate the performance of the solutions to the BQP/MILP formulation when possible, and the heuristics. The proposed FD-EVN approach has shown to be resource efficient and in particular, outperform other approaches in terms of request acceptance ratio and embedding cost, although as a tradeoff, requiring more service migration after failures.**

***Index Terms*—Enhanced virtual network design, failure dependent protection, survivable virtual network embedding.**

Manuscript received July 10, 2013; revised September 11, 2013; accepted September 25, 2013. Date of publication November 26, 2013; date of current version December 25, 2013. This work was supported in part by 973 Program of China (No. 2010CB328201 and 2010CB328202), National Natural Science Foundation of China (No. 61205058, 61272462) and Open Fund of State Key Laboratory of Information Photonics and Optical Communications (Beijing University of Posts and Telecommunications), China. Part of this work has been published in OFC/NFOEC 2011.

B. Guo, J. Li, Z. Chen, and Y. He are with the State Key Lab of Advanced Optical Communication Systems and Networks, Peking University, Beijing 100871, China (e-mail: alonggnola@gmail.com; juhao_li@pku.edu.cn; chenzhy@pku.edu.cn; heyongqi@pku.edu.cn).

C. Qiao is with the Department of Computer Science and Engineering, State University of New York, Buffalo, NY 14260 USA (e-mail: qiao@buffalo.edu).

H. Yu is with the Department of Communication and Information Engineering, University of Electronic Science and Technology of China, Chengdu 610051, China (e-mail: yuhf2004@gmail.com).

J. Wang is with the Department of Computer Science, City University of Hong Kong, Hong Kong (e-mail: jianwang@cityu.edu.hk).

Y. Zuo is with the Department of Mechanical Engineering and Science, Jilin University, Changchun 130012, China (e-mail: zuoyx08@mails.jlu.edu.cn).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/JLT.2013.2293193

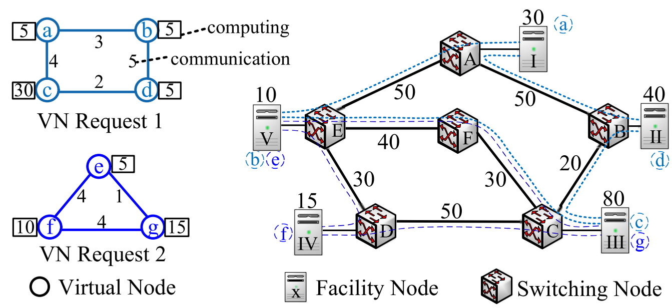


Fig. 1. (a) VN requests; (b) Substrate Network and one of the possible embedding of these VN requests.

I. Introduction

NETWORK virtualization has gained significant attention in recent years as an effective means to share substrate network (SN) infrastructure among several virtual network (VN) service providers so as to improve the utilization of the substrate resources [1], [2]. In the network virtualization environment, provisioning a VN is a promising way to support distributed applications and services which require the coordination of multiple geographically distributed facilities (e.g. storage arrays or computer clusters). With the maturity of the optical network technology and their high-bandwidth and low-latency characteristics, we will be able to deploy a federated computing and networking system (FCNS) [3]–[6], which interconnects a large number of facility nodes with optical network managed or controlled integratedly, to enable such large scale distributed applications.

In the multi-tenant network virtualization environment, one challenging problem raised is how to efficiently embed VN requests with various constraints. This is of utmost importance for increasing the utilization of SN resources and infrastructure providers' revenue. Fig. 1 illustrates a case where two VN requests requested by two applications (or services) are mapped to one SN. Each VN request consists of a collection of virtual (task) nodes connected together by a set of virtual links to form a virtual topology; it also has certain resource requirement, such as computing resources for the nodes and bandwidth resources for the links. Such VN request could be represented by a task graph as shown in Fig. 1(a). To establish the VN, one needs to embed task nodes onto facility nodes, and virtual edges onto one or more substrate paths [2], as well as allocating sufficient substrate resources which should not violate the computing/bandwidth capacity limits of facility node/substrate link. One example is shown in Fig. 1(b). This approach is modeled as the VN embedding (VNE) problem which attracts a broad interest recently [6]–[13]. VNE is extremely important in order to maximize the number of coexisting VNs and increase the utilization of substrate infrastructures. Most of the proposed approaches decompose the problem into two phases, the node embedding phase and the link embedding phase, to reduce the overall complexity of this problem. Some heuristic approaches are employed for node embedding [7], while link embedding is performed using the shortest path or multi-commodity flow (MCF) algorithms [8]. Recent researches tend to solve these two problems either jointly [9], [10] or through coordinating the two phases [11]. Besides the static VNE formulation, adaptive VN provisioning has also been studied to cope with evolving demands (i.e., dynamic changes in service demands) and network anomalies or faults [13].

Meanwhile, some other works focus on carrying out VN mapping in a distributed manner to solve the scalability and resilience problem suffered in centralized manner [12], where a central entity should maintain up to date information about the SN to make the appropriate VN mapping decisions [7]–[11].

However, with network virtualization gaining momentum, the survivability challenges in VNE should also be well investigated. In a large networked computing system, hardware and software failures of facility nodes and communication resources (e.g., links and switching nodes) are norm instead of exception, such as power outages caused by virus attack, disk failures, misconfiguration or fiber cut [14]–[18]. Such failures will force the virtual node (links) assigned to it to be migrated/re-embeded to another facility node at a geographically different location (link-disjoint substrate path). This means that, in order to survive from the disruptions due to such failures, one must reserve redundant facility nodes and bandwidth on fiber links such that after any failure, there are adequate remaining computing and networking resources to migrate/remap the VN request. Accordingly, the problem of minimizing the resources, including computing (CP.) and communication (CM.) resources, reserved for VN request to tolerate substrate failures, (hereafter called Survivable VNE problem, SVNE) is both critical and challenging. Actually, SVNE problem is quite different with the protection approach in IP over WDM network by investigating facility node failure caused virtual node failure problem and employing node migration induced VN remapping as an unique recovery strategy.

To survive a substrate link failure, pre-computed alternative paths in VN are used in general and the bandwidths are allocated before or after a failure. For instance, the reactive detour solution is employed after a substrate link failure in [15], while authors in [16], [17], [19] propose a proactive backup approach (considering the backup resources sharing) to avoid service disruption in reactive restoration approach and improve the substrate resource utilization.

In terms of SVNE capable of recovering/re-embedding the task nodes after a facility node failure, there are two basic approaches: Failure Dependent Protection (FDP [5], [20] and Failure Independent Protection (FIP) [21]–[24], and the differences between them are as follows. In FIP, a host (facility) node is assigned and dedicated to backup all working host (facility) nodes. That is, no matter which working host node fails, the affected task node will be migrated to the only one backup host node. On the other hand, with FDP, each working host node can have a different backup host node under different failure scenarios. In fact, after a failure, even an unaffected task node may be migrated from a working host node to its corresponding backup host node, as a result of re-embedding the entire task graph. In other words, FDP could provide more flexibility in survivable VN designing by allowing task nodes migrating freely after failure, so FIP could be considered as a special case of FDP and FDP is expected to use fewer resources at the cost of more task nodes migrations after a failure.

In addition, conventional approaches to SVNE deal with the redundancy when embedding task nodes/links to the SN [5], [22] (e.g., one maps an N-node VN to at least $N+1$ facility nodes in order to tolerate a single facility node failure.) An alternative is to deal with redundancy even before the embedding process. That is, we could enhance the VN first by including more task nodes and links, and then embed the Enhanced VN (EVN) to the substrate in a one-to-one fashion. Such an alternative has recently been studied in [22]–[24], which are FIP approaches.

In this paper, we propose an SVNE approach that, for the first time, combines FDP with EVN design. We first design an EVN with consideration of FDP and then mapped it on a SN. A unique feature/requirement of our FD-EVN is that each virtual node (link) in FD-EVN may need to emulate multiple virtual nodes (links) in the original VN, since which virtual node (link) for it to emulate depends on the failure occurred. Accordingly, the amount of computing (communication) resources associated with the nodes (links) in the FD-EVN needs to be properly dimensioned (increased from that the original amount to provide sufficient redundancy). How to design and dimension a resources efficient FD-EVN is thus an open and important problem, whose solution will lead to an efficient solution to the SVNE problem.

Motivated by the discussion above, we will study the SVNE problem in this paper with a single facility node failure based on a two-step approach involving: (a) designing an EVN with redundant resource requirement and (b) embedding the EVN on the SN with the resource constraints specified by EVN. The reminder of the paper is organized as follows. The binary quadratic programming (BQP) [25] formulation of EVN design and mixed integer linear programming (MILP) formulation of EVN embedding (EVNE) problem are presented in Section II and III respectively. Heuristic algorithms for EVN design and embedding are proposed in Section IV and their performance is evaluated in Section V prior to the conclusions in Section VI.

II. EVN Design Formulation

In this section, we define the EVN design problem as follows: for a given VN request with N task nodes, enhance the VN with one additional node and a set of appropriate links to connect these $N+1$ nodes, and reserve sufficient computing and communication resources in these nodes and links to guarantee the restorability of VN request after a facility node failure.

As shown in Fig. 2, for a given 5 nodes VN requiring 16 and 24 units of computing and communication resource respectively

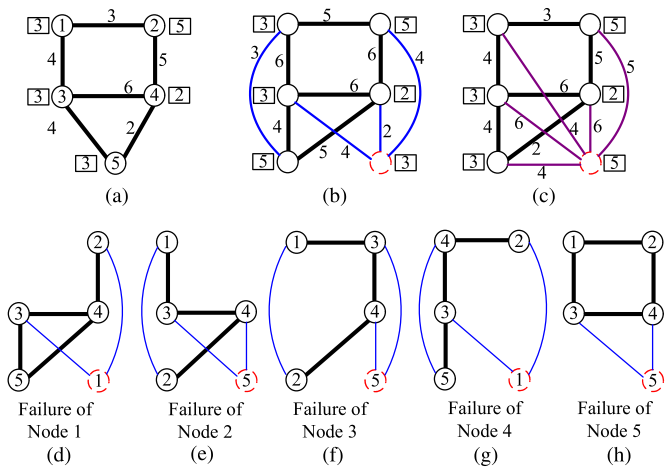


Fig. 2. (a) VN; (b) FD-EVN design; (c) FI-EVN design; (d)-(h) Migration strategy for FD-EVN under different node failure.

[see Fig. 2(a)], a 6-nodes FD-EVN (one redundant node with dashed line is added) is designed in Fig. 2(b) with 5 units of redundant computing resources and 21 units of redundant communication resource to guarantee the task graph could be remapped/restored after any single node failure, as enumerated in Fig. 2(d)–(h). Compared with FD-EVN, EVN with FIP (FI-EVN) shown in Fig. 2(c) needs 5 additional units of computing and 25 additional units of bandwidth resources, which is more than what is needed with FD-EVN. This is because with FI-EVN, any failed node has to be replaced by the redundant/added node, which is not as flexible as in FD-EVN.

Note that even though FD-EVN saves the virtual resources when compared to FI-EVN, one also needs to ascertain that such savings also apply to the substrate resources after the EVN is mapped. In the following, we first formulate the EVN design problem as a BQP in this section, and then the EVNE problem as a MILP in Section III.

A. VN Request Model

A task graph for a VN request is an undirected attributed graph, denoted as $G^V(V^V, E^V)$, where V^V corresponds to a set of task nodes and E^V denotes a set of bidirectional edges among the VN nodes. Each task node specifies the needed computing resource c_s^V, and each edge specifies the requested amount of communication (bandwidth) resource b_{st}^V.

B. Edit Grid

To simplify the description of EVN design for a given N nodes VN, we introduce an attributed $N{+}1$ nodes completely connected graph, referred to as *Edit Grid* [26]. For EVN designing, we embed the VN in the Edit Grid first, and then enhance it by reserving one additional node and several necessary connections, as well as dimensioning CP.&CM. requirements associated with each node or link to guarantee that there are always sufficient resources and connections to reassign the task graph after removing any single node and its associated resources in the Edit Grid.

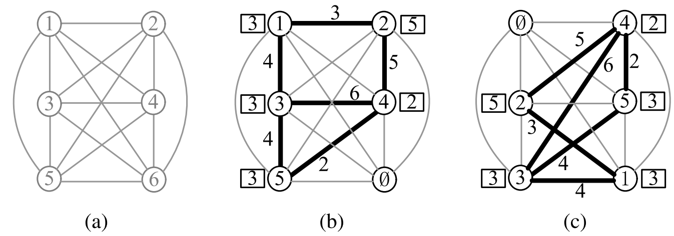


Fig. 3. (a) 6 nodes Edit Grid; (b) An Embedding of Task Graph on Edit Grid (η_0); (d) Isomorphism Graph of Task Graph (η_1).

TABLE I
STATE VECTOR OF EDIT GRID

		η_0	η_1			η_0	η_1
Vertex label	l_1	v_1^V	Ø	Edge attribute	b_{12}	3	0
	l_2	v_2^V	v_4^V		b_{13}	4	0
	l_3	v_3^V	v_2^V		b_{14}	0	0
	l_4	v_4^V	v_5^V		b_{15}	0	0
	l_5	v_5^V	v_3^V		b_{16}	0	0
	l_6	Ø	v_1^V		b_{23}	0	5
Vertex attribute	c_1	3	0		b_{24}	5	2
	c_2	5	2		b_{25}	0	6
	c_3	3	5		b_{26}	0	0
	c_4	2	3		b_{34}	6	0
	c_5	3	3		b_{35}	4	0
	c_6	0	3		b_{36}	0	3
					b_{45}	2	4
					b_{46}	0	0
					b_{56}	0	4

Specifically, Edit Grid is defined as an attributed complete graph G^Ω (V^Ω, E^Ω). For simplicity, let $\eta{\in}L^{N+1} \times C^{N+1} \times B^{N(N+1)/2}$ denote the state vector [26] of the Edit Grid, where L refers to the embedding/assigning relationship between task nodes and the nodes in the Edit Grid, C and B refer to the resource requirements of nodes and links, respectively. And, different state vectors of edit grid indicate different attributed graphs. For example, an embedding of the given task graph in Fig. 2(a) on the Edit Grid [see Fig. 3(a)] is shown in Fig. 3(b), as well as the resource requirements of each node and link. The state vector of the Edit Grid in Fig. 3(b) is presented as η_0 in Table I. It is worth noting that vertices (edges) in G^Ω may have null labels (or 0 resources requirements), which indicates that the element is not part of the embedded graph. Such an element is referred to as ‘dummy’ and allows for *insertion* edits (which involves this node in the embedded graph) by swapping a null label for a non-null label.

At the same time, for the task graph reassigning approach in Edit Grid, we also introduce the *permutation matrices*, which is an $(N{+}1){\times}(N{+}1)$ orthogonal matrix having $PP^T = P^TP = I$ (where I is the identity matrices), to indicate the corresponding state vector transformation of Edit Grid. Assume at this point that the initial state of the Edit Grid η_0 contains the given task graph in its standard placement. And we have another state η_1 such that it describes task graph situated on the Edit Grid in a different way as shown in Fig. 3(c), as well as its state vector shown in Table I. Based on the knowledge in graph theory, the two graphs corresponding to these different states of Edit Grid are isomorphic [14] which suggests that there is a bijection between these two attributed graphs. So, we could employ permutation matrices to implement task graph reassignment in Edit Grid. For example, with the following Edit Grid state η_0:

$$l^{\eta_0} = (1 \quad 2 \quad 3 \quad 4 \quad 5 \quad 0),$$

$$c^{\eta_0} = (3 \quad 5 \quad 3 \quad 2 \quad 3 \quad 0),$$

$$b^{\eta_0} = \begin{pmatrix} 0 & 3 & 4 & 0 & 0 & 0 \\ 3 & 0 & 0 & 5 & 0 & 0 \\ 4 & 0 & 0 & 6 & 4 & 0 \\ 0 & 5 & 6 & 0 & 2 & 0 \\ 0 & 0 & 4 & 2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

where the entry $l_x^{\eta_0} = y$ implies that task node v_y^V is assigned on v_x^Ω of Edit Grid. Then, we could get the state vector of Edit Grid η_1 through

$$l^{\eta_1} = l^{\eta_0} P,\ b^{\eta_1} = (P)^T b^{\eta_0} P,\ c^{\eta_1} = c^{\eta_0} P$$

with $P = \begin{pmatrix} 0\,0\,0\,0\,0\,1 \\ 0\,0\,1\,0\,0\,0 \\ 0\,0\,0\,0\,1\,0 \\ 0\,1\,0\,0\,0\,0 \\ 0\,0\,0\,1\,0\,0 \\ 1\,0\,0\,0\,0\,0 \end{pmatrix}$.

C. EVN Design Formulation

With the discussion above, for a given N-nodes VN, EVN is designed within an Edit Grid with $N+1$ nodes through properly selecting the necessary links between these nodes and dimensioning the resources requirements associated with these nodes and links. Our design objective is to minimize the total amount of such resources while still guaranteeing that if a node fails (that is, the node and its adjacent links are removed in the Edit Grid), we can still assign each node/link in VN to a node/link in the EVN, that has sufficient CP.&CM. resources respectively.

Furthermore, there are two different cases. If re-embedding or migrating the unaffected task node is allowed for failure recovering, it is known as the FD-EVN design problem. Otherwise, after each failure, if the failed node is restored in the only one backup node without migrating other unaffected node, it is referred to as the FI-EVN design problem. Generally speaking, designing FD-EVN is a combinatorial optimization problem and needs to be investigated in depth, while FI-EVN exists exclusively and could be figured out easily (meaning not very clear).

To begin the EVN designing, the task graph is embedded on the same amount of lower-numbered nodes of Edit Grid (i.e., $v_1^\Omega, \ldots, v_N^\Omega$, referred to as primary nodes), as well as the links. This is denoted as η_0. Given the symmetry of the Edit Grid, how the VN is initially assigned would not have impact on the results. Then, we formulate the FD-EVN design problem as a BQP problem. As well, the FI-EVN is also formulated to calculate its resources requirements more expediently.

1) FD-EVN Design: FD-EVN design with the purpose of minimizing the total spare resources needed in the Edit Grid is challenging since there are many different approaches to reassign the task graph after each node failure. However, based on the discussion above, permutation matrix essentially serves to reassign the graph embedded on the Edit Grid gracefully and alter the state of it accordingly. So, FD-EVN design could be formulated as a BQP with permutation matrix, which is presented as following.

Objective Function:

$$\min \left[\sum_{e \in E^\Omega} B_e^\Omega + \alpha \sum_{n \in V^\Omega} C_n^\Omega \right]. \tag{1}$$

Remarks: The objective of FD-EVN design is to minimize the total consumption of CP.&CM. in Edit Grid. α is the weight coefficient which represents the relative importance of bandwidth and computation resources.

Permutation Constraints:

$$\begin{aligned} l^{\eta_k} &= l^{\eta_0} P^k \\ b^{\eta_k} &= \left(P^k\right)^T b^{\eta_0} P^k \\ c^{\eta_k} &= c^{\eta_0} P^k \end{aligned} \tag{2}$$

where P is an $(N+1)\times(N+1)$ orthogonal matrix and $k = (1, \ldots, N)$.

Remarks: Constraint (2) defines the state vector transformation of Edit Grid in term of permutation matrices P. More specifically, it could be helpful to reveal the resource requirement for links and nodes after permutation

$$p_{mn}^k = 1 \text{ for all } m = N+1,\ n = k. \tag{3}$$

Remarks: Constraint (3) ensures that all the single primary node failure scenarios in the Edit Grid are considered. More exactly, P^k with $p_{(N+1)k}^k = 1$ implies that, after this permutation, the task graph is reassigned within Edit Grid, excluding v_k^Ω and its associated links.

Resources Constraints:

$$C_n^\Omega = \max_k c_n^{\eta_k},\ B_{mn}^\Omega = \max_k b_{mn}^{\eta_k},\ \forall n,\ m \in V^\Omega. \tag{4}$$

Remarks: Constraint (4) reveals that the total resource allocated for each virtual link and node must be sufficient to handle the resource requirements in different failure scenarios.

In conclusion, the formulation indicates that this problem is a BQP problem and the solution of this problem is a series of permutation matrix $\{P^k, k = 1, \ldots, N\}$ which determine the migration/reassigning approach after each node failure. Thus, its computation complexity is $(N!)^N$, where $N!$ is the computation complexity for one node failure.

2) FI-EVN Design: FI-EVN design could be worked out easily since the migration strategy for each node failure is specific. After embedding the task graph on Edit Grid, an additional backup node should be included and connected with all primary task nodes, as shown in Fig. 2(c). As well, the total CM.&CP. requirements associated with each node and link are defined as

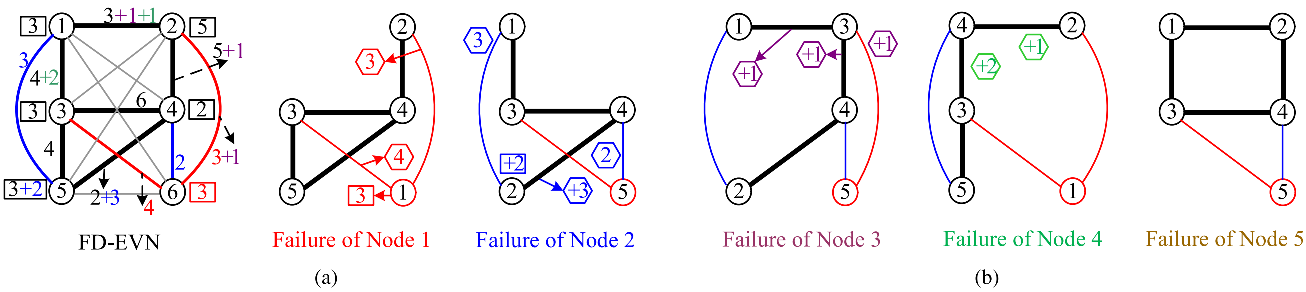


Fig. 4. (a) Edit Grid and the designed FD-EVN; (b) An incremental approach for CP.&CM. resources dimensioning ∗the number in hexagon (quadrangle) represents additional bandwidth (computing) resources needed in virtual links (nodes).

follows:

$$C_n^{\Omega} = c_n^{\eta_0}, \text{ for } n \in [1, \dots N];\ C_{N+1}^{\Omega} = \max_{I \le n \le N} c_n^{\eta_0} \quad (5)$$

$$B_{mn}^{\Omega} = b_{mn}^{\eta_0}, \text{ for } m,\ n \in [1, \dots N];\ B_{m(N+1)}^{\Omega} = \max_{1 \le n \le N} b_{mn}^{\eta_0}. \quad (6)$$

Remarks: (5) defines the resource requirements for the primarily nodes and backup node in Edit Grid respectively. Especially, the maximum CP. resources required by those task nodes should be allocated to the backup node since it would emulate each of them after they fails severally. In the same way, the bandwidth requirement of each redundant link (i.e., the link connecting backup node with one of the primary nodes) is defined in (6).

III. EVNE Formulation

After designing an EVN, in this section, we also formulate its embedding problem as an MILP fitting for both FD-EVN and FI-EVN embedding, since the difference between them in embedding approach could be reconciled by a general resources sharing constraint. More details would be elaborated in the following part.

A. Terminology

We model the SN as an undirected attributed simple graph $G^S\ (V^S, E^S)$, where V^S and E^S is the set of substrate nodes and substrate links respectively. Furthermore, a dynamic embedding approach is considered in this work. So, each substrate node (link) is associated with the residual computing (communication) resource capacity. Also, we define some variables to simplify the formulation

f_{ij}*:* the exact amount of bandwidth will be reserved in substrate link *ij*;

σ_{ij}^{mn}*:* binary variable to indicate the embedding relationship between e_{mn}^{Ω} and e_{ij}^{S}, i.e., $\sigma_{ij}^{mn} = 1$ if e_{mn}^{Ω} is mapped on e_{ij}^{S}; $=$ 0 otherwise.

B. Formulation

Since the VNE problem subject to certain constraints (i.e., each virtual node has location requirement) could be formulated as a mixed integer commodity flow problem [7], the EVNE could be formulated in the same way. So, EVNE formulation is also a node and link coordinated embedding and complies with the conventional residual substrate resource capacity constraints and flow related constraints defined in [7], as well as its objective function (i.e., minimizing the cost of consumed substrate resources). Besides, with respect to the node embedding, we have the assumption that all the virtual nodes in EVN should be mapped on physically isolated substrate nodes.

However, for virtual links embedding, since not all the virtual links or not all their bandwidth would be employed simultaneously under single node failure, some virtual links could share substrate resources if they are embedded on the same substrate link, which would reduce the total substrate bandwidth needed. For example, the FD-EVN for the task graph in Fig. 2(a) is designed within its Edit Grid as shown in Fig. 4(a), both B_{13}^{Ω} and B_{34}^{Ω} require 6 units bandwidth. However, only 10 in total, not 12, need to be allocated in their common substrate links which is the result of fact that: (a) these two virtual links would be employed simultaneously only after the failure of substrate nodes which are mapped on by v_2^{Ω} and v_5^{Ω}, and their total bandwidth needed is 10 (4 for e_{13}^{Ω} +6 for e_{34}^{Ω}); (b) under other substrate nodes failure, only one of these two virtual links exists in the reassigned task graph and needs at most 6 units. What's more, it is worth noting that such link sharing constraints is determined by the re-embedding/migration strategy after node failure (corresponding to the permutation matrix in EVN design approach). So, with the given permutation matrix, the communication bandwidth ***resource sharing constraint*** is formulated as follows:

$$\sum_{e_{mn}^{\Omega}} b_{mn}^{\eta_k} \sigma_{ij}^{mn} \le f_{ij} \le RB_{ij}^{S},\ \forall ij \in E^S,\ \forall k \in [1, N] \quad (7)$$

where RB_{ij}^{S} is the residual bandwidth capacity in e_{ij}^{S}.

Remarks: It guarantees that there are sufficient bandwidth in substrate links to support the requirements of every virtual link in different node failure scenarios.

Thus, rather than solving the link embedding problem based on the multi-commodity flow LP, we need consider the resource sharing issues, which is computational intractable. As a result, heuristic algorithms with acceptable computation complexity are proposed in Section IV.

IV. Heurstics Algorithms

In this section, we propose two heuristic algorithms for FD-EVN design, as well as an EVNE algorithm with resources sharing consideration fitting for both FD-EVN and FI-EVN.

The general idea of the heuristic for FD-EVN design is to consider the failure of primary nodes in Edit Grid sequentially, and in each step, compute the minimum additional resources needed to reassign the task graph based on an incremental approach (i.e., recovering from the current node failure should take not only the survived primary nodes/links resources into consideration, but also the survived redundant resources reserved for previous node failure). After examining all the node failures, EVN is constructed within the Edit Grid with the added redundant resources in each step.

However, as a matter of fact, computing the minimum additional resources needed to convert one attributed graph to another (hereafter called graph alignment problem, *GAP*) is an NP-complete problem which could be reduced from the *Graph Edit Distance* problem [26]. Therefore, we relax some constraints and propose two heuristic algorithms. In detail, we first decompose a graph to a multiset of star structures which retains certain structural information of the original graph. Then, the graph alignment cost could be approximated by the matching cost between two graphs based on their star representations. This approach is elaborated as follows.

A. FD-EVN Design Algorithm

1) Graph Decomposition: Star Structure: A star structure s is an attributed, single-level, rooted tree which can be represented by a 4-tuple $s = (r, B^*, C^*, L^*)$, where r is the root node, B^* is the bandwidth of each link associated with the root node, C^* and L^* are the computation resource requirement and labels of nodes included in this star structure. In this case, the node label is the task node mapped on it in the primary embedding. Edges exist between the root node and its adjacent nodes, and no edge exists among its adjacent nodes.

More exactly, for node v_n in an attributed graph $G\,(V, E, L)$, we can generate a star structure s_n corresponding to v_n in the following way: $s_n = (v_n, B^{*n}, C^{*n}, L^{*n})$ where $B^{*n} = \{B^{*n}_{mn}|$ for all $e_{mn} \in E\}$, $C^{*n} = \{C^{*n}_m$ | for all $e_{mn} \in E\}$ and $L^{*n} = \{l(v_m)|$ for all nodes adjacent with $n\}$. Accordingly, we can derive N star structures for a graph with N nodes. In this way, a graph can be transformed to a multiset of star structures.

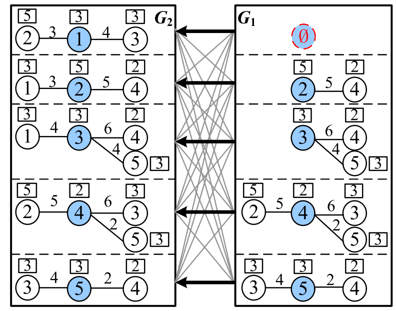
Due to the particularity of star structure, the alignment cost between two star structures can be computed easily as below. For two star structures s_x and s_y, the alignment cost of s_x to s_y is

$$\begin{aligned}\lambda(s_x, s_y) = &\sum_{l(v_m)\in L^{*x}\cap L^{*y}} \left[\left|B^{*y}_{my} - B^{*x}_{mx}\right|_0 + \alpha\left|C^{*y}_m - C^{*x}_m\right|_0\right] \\ &+ \sum_{l(v_m)\in L^{*y}-L^{*x}} \left[B^{*y}_{my} + \alpha C^{*y}_m\right] + \alpha\left|C_y - C_x\right|_0 \\ &+ \delta(v_x, v_y) \qquad (8)\end{aligned}$$

where δ and $|x|_0$ is defined as follows:

$$\delta(v_x, v_y) = \begin{cases} C_{\text{migra.}} & l(v_x) \neq l(v_y) \\ 0 & \text{otherwise} \end{cases}; \quad |x|_0 = \begin{cases} x & \text{if } x \geq 0 \\ 0 & \text{if } x < 0. \end{cases}$$

$\delta(v_x, v_y) \neq 0$ indicates that the label of root node v_x should be reallocated as that for v_y (in this case, this implies that migration would happen in the alignment approach) and $C_{\text{migra.}}$ is the migration cost.



	s_1	s_2	s_3	s_4	s_5
s_0	$\boxed{19}$	19	26	27	15
s_2	16	$\boxed{6}$	16	25	8
s_3	16	9	$\boxed{7}$	20	8
s_4	2	17	19	$\boxed{0}$	5
s_5	9	12	19	18	$\boxed{0}$

Fig. 5. The star representation of residual task graph after failure of node 1 (G_1) and the task graph (G_2); as well the alignment cost matrices based on the star structure alignment cost. Migration cost is set as 1 in this example.

In this way, both the task graph and the residual graph after specific virtual node failure could be represented as a multiset of star structures. With such a graph decomposition, an alignment matrix of these star structures could be constructed with the alignment cost definition above. Therefore, the proposed GAP could be transformed to a bipartite graph matching problem which will be investigated in the following part.

2) Bipartite Graph Matching: Based on the discussion above, the computation of the graph alignment cost is equivalent to solving the vertices assignment problem, which is one of the fundamental combinational optimization problems concerned with finding the minimum weight matching in a weighted bipartite graph. In our case, these two sets of vertices are the two multisets of star structures S_1 and S_2, and the weight of the edge connecting stars in S_1 and S_2 is the alignment cost between the corresponding two stars. Then, Hungarian algorithm [27] could be applied to solve the bipartite graph matching problem in $O[|V|^3]$ time.

To elaborate this approach, Fig. 5 shows how the 5-node VN in Fig. 2 be decomposed into a multiset of five star structures (shown as G_2), and each containing a primary task node (also called root node which is highlighted), its adjacent links and its neighboring nodes. To dimension the redundant resources needed after N_1 failure, the latest (partially enhanced) version of the FD-EVN in the Edit Grid (which at first is the same as the VN), with v_1^Ω and its adjacent links removed, can also be decomposed (shown as G_1). For each star-structure s_x (where $1 \leq x \leq 5$) in G_1, we calculate the cost with the proposed alignment cost definition. In detail, about the alignment cost calculation, we could put it in a different way: it is the cost (in terms of additional CP.&CM. resources and root node relabeling cost) of enhancing s_x (where $1 \leq x \leq 5$) in G_1 with additional nodes and links such that the latter contains the s_y in G_2 as a subgraph. Note that the constraints are: 1) the primary task node r_x of s_x in G_1 has to be assigned to the primary Edit Grid node of s_y in G_2, (after which the latter is relabeled to y), and 2) no other Edit Grid nodes can be relabeled, although new Edit Grid nodes can be created along with new Edit Grid links with additional amount of CP.&CM. resources. This cost is shown as the (x, y)-th entry in an $N \times N$ alignment cost matrix in Fig. 5. In this way, GAP could be figured out through efficiently solving this bipartite graph matching issues, e.g., the optimal alignment/matching are marked in Fig. 5.

At last, based on the achieved graph alignment solution, a partial enhancement to the task graph in Edit Grid could be dimensioned able to handling the failure of N_1, as shown in Fig. 4(b). However, intuitively, the alignment for computing the matching between $S(G_1)$ and $S(G_2)$ is trying to approximate the optimal alignment between the nodes in G_1 and G_2. Note that since the matching in the bipartite graph takes only one node and its neighbors into consideration, there are in fact fewer constraints on the output when determining the optimal matching in the bipartite graphs compared to determining the optimal alignment between two graphs. Thus, the alignment cost that convert G_1 into G_2 based on the bipartite graph matching will either be the same or higher than the optimal alignment cost between the two graphs.

Finally, we repeat the procedure of graph decomposition and the bipartite graph matching for each following node failure to construct the final FD-EVN. As shown in Fig. 4(b), partial enhancement of the EVN for each node failure is presented. It is worth noting that, every step of partial enhancement is based on the latest version of Edit Grid including the additional CP.&CM. resources needed for the pervious node failure, which also implies the sequence of node failure to be considered would have impact on the amount of additional CP.&CM. resources needed. Thus, we develop two heuristic algorithms with different strategy to determine the sequence of node failure to be considered.

More specifically, the rationale behind **FD-EVN:Radom** is to consider the node failure in a random sequence. Moreover, the **FD-EVN:MinRscFirst** is developed based on the observation that, comparing with the node associated with more resources, it is generally true that there are fewer optimal options (i.e., there could be multiple optimal solution for the bipartite graph matching problem) to reassign the task graph from failure of node associated with fewer resources. So, taking the node with fewer resources considered first, we would have a great opportunity to share the redundant resources in a maximum way. The procedures of proposed algorithms are described as follows, where G_σ^Ω indicates the final state of Edit Grid after removing the node v_m^Ω and its associated connections/resources from partial enhancement EVN $G^\Omega(\eta')$, and R_{ADD} represents the partial enhancement to the VN (including newly added links/nodes and incremental resources for existing links/nodes).

B. EVN Embedding

After EVN designing, we need embed it on the SN in a resources efficient way with the constraints defined in the formulation of EVN embedding. For simplicity, nodes and links are mapped separately in this work. First, for the node embedding, we employ an existing heuristics [11] and each virtual node is mapped to a single distinct substrate node with maximum available resources by a one by one embedding.

Regarding to the link embedding, it is derived from the MCF solution and the resource sharing constraint inherited from EVN design approach. The rationale behind the link embedding

---

***Algorithm* FD-EVN: Random (FD-EVN:Rad)**

Input: $G^V(V^V, E^V, L^V)$, α, G^Ω, embed G^V on $G^\Omega \rightarrow G^\Omega(\eta_0)$;
Output: EVN in the Edit Grid and P^k.

1) **Initialization**: $G^\Omega \rightarrow G_*^V$, $\emptyset \rightarrow R_{\mathrm{ADD}}$;

2) **Iteration**:
  set $G^\Omega(\eta_0) \rightarrow G^\Omega(\eta')$;
  for each randomly chose $v_m \in V^V$, **do**
    set $G^\Omega(\eta') - v_m \rightarrow G_\sigma^\Omega$, transform G_σ^Ω and G^V to $S(G_\sigma^\Omega)$ and $S(G^V)$;
    calculate the $S(G_\sigma^\Omega) \rightarrow S(G^V)$ alignment cost matrices;
    use Hungarian to find the node matching/alignment ***M*** with Min[$S(G_\sigma^\Omega) \rightarrow S(G^V)$];
    $G^\Omega(\eta_M') - G_\sigma^\Omega \rightarrow R_{ADD}^k$; // $G^\Omega(\eta_M')$ *is the embedding of task graph on Edit Grid with node assignment M.*
    $R_{ADD}^K \cup G^\Omega(\eta_0) \rightarrow G^\Omega(\eta')$;
    $V^V - \{v_m\} \rightarrow V^V$;
    $P \rightarrow P^k$ (k=m); //*according to the optimal matching M.*

3) output the state vector of G^Ω; P^k ($k \in [1, N]$).
  //*the current state of Edit Grid is EVN embedding*

***Algorithm* FD-EVN: MinRscFirst (FD-EVN:MinFir)**

With the same *Input*, *Output*, **Initialization** as Algorithm 1,

2) **Iteration**:

$$v_m^V \uparrow \left| c_m^V + \sum_{mn \in E^V} b_{mn} \right.$$ // *sorting the nodes in ascending order in terms of the amount of resources associated with each node*;

  do as in **Random** according to this order.

---

algorithm is to separate the bandwidth resource in each EVN link according to the migration strategy after each node failure and reveal the confliction or sharing relationship between virtual resources. Then, implement the embedding approach to share the substrate resources in a maximum way.

In detail, first, the primary bandwidth affiliated with VN request is mapped after the node embedding with MCF solution. Then, the redundant bandwidth in EVN is mapped gradually according to R_{ADD}^k and their potential bandwidth sharing sets. More specifically, the bandwidth request in R_{ADD}^k could share the substrate resources already reserved but not included in b^{η_k}, which is the resources would be used in failure of node v_k. The proposed EVNE algorithm is elaborated as follows.

C. Computation complexity

1) Complexity of FD-EVN Design: In FD-EVN design, graph transformation (including generating the alignment cost matrices), node matching and additional resources calculation are implemented for each node. While the running time of node matching using Hungarian algorithm and additional resources calculation are $O[|V^V|^3]$ and $O[|E^V|]$ respectively, the time

***Algorithm* EVN Embedding: (EVNE)**

Input: EVN, G^S, P^k ($k \in [1, N]$);

Output: σ_{ij}^{mn}, μ_{ij}^{mn}, f_{ij};

1) **Initialization:** embedding the nodes with existing heuristics and solve the splittable MCF problem to embed the bandwidth in G^V on G^S; sort $R_{ADD}^k(mn)$ in ascending order; $G^V \rightarrow G_\sigma^V$; $\mu_{ij}^{mn} = f_{ij}^{mn}$.

2) **Iteration**:

for each $R_{ADD}^k(mn)$, **do**

update the sharing set $G_\sigma^V - b^{\eta_k}$;

compute a minimal-cost path P based on the following edge cost function, w_{ij} is the weight of substrate link ij:

$\left|R_{ADD}^K(mn) - f_{ij}^{m'n'}\right|_0 w_{ij}$, for all $m'n'$ in sharing set;

$G_\sigma^V \cup R_{ADD}^k(mn) \rightarrow G_\sigma^V$;

$f_{ij}^{mn} + f_{ij} \rightarrow f_{ij}$, σ_{ij}^{mn}=1, if $ij \in P$ and $f_{ij} \leq B_{ij}^S$;

$\mu_{ij}^{mn} + R_{ADD}^k(mn) \rightarrow \mu_{ij}^{mn}$.

3) if success, VN request accept; otherwise, reject.

complexity of FD-EVN design is $O[|V^V|(|V^V| + |V^V|^3 + |E^V|)] \approx O[|V^V|^4]$.

2) Complexity of EVNE: In EVNE, node embedding, edge embedding (by solving splittable MCF problem) and weighted path computation are implemented for each EVN link. With the knowledge that, splittable MCF could be solved with running time $O[|E^S| \log |E^S|]$ [28] and the time complexity of weighted path computation is $O[|V^V|^3]$, the time complexity of EVNE is $O[|E^V|(|E^S| \log |E^S| + |V^V|^3)]$.

V. Performance Evaluation

In the proposed two step approach for the SVNE problem, we employ a proactive FDP based EVN design with sufficient redundancy before embedding process in order to conquer the limitation of existing work, which consider the redundancy within the embedding process and FIP based migration approach, in resource efficiency. Accordingly, the performance of optimal solution of EVN design and embedding formula, and heuristic algorithms are presented in this section in terms of average acceptance ratio, embedding cost, as well as the migration frequency after facility node failure.

A. Simulation Settings

For a VN request, the number of VN nodes is randomly determined by a uniform distribution between 2 and 10, and each pair of virtual nodes is randomly connected with probability 0.5. The computing requirements on VN nodes follow a uniform distribution from 1 to 5, as well as the bandwidth on VN links from 10 to 20. The arrivals of VN requests are modeled by a Poisson process (with mean of 10 requests per 100 time units). The duration of the requests follows an exponential distribution with 1000 time units on average. We assume the relative cost of computing and bandwidth is 3 [1], [5], which means $\alpha = 3$. The SN topologies used are randomly generated with 100 (10 for Table II) nodes using the GT-ITM tool [29]. The computing (bandwidth) resources of the substrate nodes (links) are real numbers uniformly distributed between 50 and 100 (200 and 300).

To model the facility node failure scenario, we randomly choose one substrate facility node to fail in every 2000 time units.

TABLE II
Comparison of Resource Needed for Different EVN Design Approach

N_{VN} (CP.+CM.)→		2 (7+16)		3 (10+43)		4 (14+62)	
EVN Design	EVNE	CP.†+CM†	E_C	CP.†+CM.†	E_C	CP.†+CM.†	E_C
FI-EVN	MILP	11+48	95	14+84	152	18+118	294
FD-EVN*	MILP	11+48	90	14+75	36	19+101	241
FI-EVN	EVNE	11+48	95	14+84	155	18+118	303
FD-EVN:Rad	EVNE	11+48	91	14+81	142	20+106	269
FD-EVN:MinFir	EVNE	11+48	90	14+76	39	19+103	259

*means the solution of BQP formula is used; N_{VN} is the number of node in VN; CP.† (CM.†) is that of in EVN; E_C is the total embedding cost.

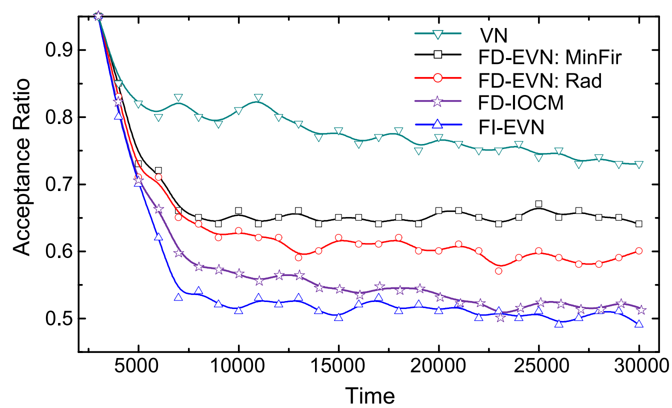


Fig. 6. VN request acceptance ratio over time.

B. Acceptance Ratio

In this section, the service acceptance ratio for VN with the proposed survivable approach is examined, as well as a typical one step approach proposed in [5], e.g. FD-IOCM. Also, the acceptance ratio of VN without survivability requirement is presented as a baseline to gauge the impact of additional amount of resources consumed for survivability on the service provisioning capability of SN. We employ the proposed EVNE as the embedding algorithm for all the EVN designing approach.

In Fig. 6, acceptance ratio drops quickly before 5000 time units because there are sufficient substrate resources for the arrived VN requests, and when the amount of running VN requests in system are stable (after 7500 time units), acceptance ratio would keep consistent. Fig. 6 also depicts that FD-EVN:MinFir and FD-EVN:Rad lead to higher acceptance ratio than FI-EVN algorithm through efficient resources sharing. In particular, comparing with FI-EVN, FD-EVN:MinFir approach

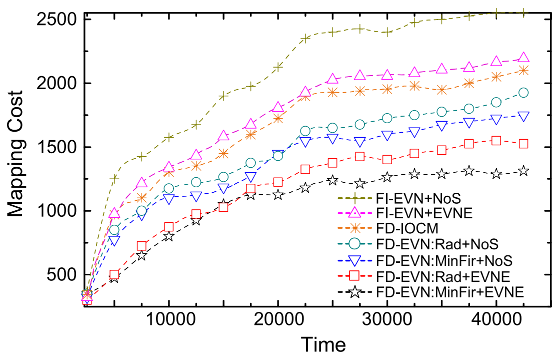


Fig. 7. EVNE cost over time.

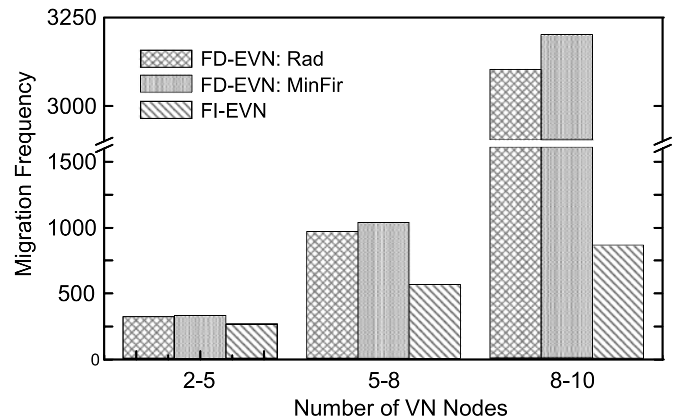


Fig. 8. Node Migration Frequency after Facility Node Failure.

achieves almost 15% improvement of acceptance ratio in the long run. Intuitively, it is the result of fact that, for FI-EVN embedding, the backup node is associated with a lot of resources since it has to emulate every primary node after it fails, and so, embedding of backup node is vulnerable. However, the acceptance ratio suffers at least 8% losses caused by redundant resources consumption for survivability purpose.

Particularly, regarding to the performance of FD-IOCM, first, the proposed two-step FDP approach could achieve a higher acceptance ratio than that of FD-IOCM. Second, although FD-IOCM outperforms FI-EVN during the simulation, they approach each other as time goes on in the simulation.

C. Embedding Cost

In this work, the embedding cost is calculated as the cost of the substrate resources (i.e. cost of CP. on all facility nodes and CM. on all fiber links) consumed to satisfy the EVN resource requirements. And, in this simulation, we assume there are sufficient resources in all the substrate components.

In order to evaluate the efficiency of the proposed heuristics, we also investigate the performance of optimal FD-EVN design and EVNE through solving their formulation with LINGO 8.0, in a 10 nodes SN with small size VN requests. Also, for FI-EVN embedding, both the heuristics and solution of MILP formula are discussed.

In Table II, the total virtual resources in different EVN design approach and their corresponding embedding cost in SN are presented. It indicates that, FD-EVN and FI-EVN consume almost the same redundant computing resources, but FD-EVN could save almost 10% virtual bandwidth resources, which results in a significant reduction of total embedding cost (as shown in Table II E_C).

As well, the same result could be concluded from Fig. 7, while FD-IOCM consumes less substrate resources than FI-EVN based approach but more than FD-EVN based approach. We further compare the embedding cost of different EVN design algorithms with (denoted as +EVNE) or without (denoted as +NoS) consideration of the substrate resources sharing in EVNE approach. Generally, with regular VN arriving and leaving (e.g., the network after 20000 time units), embedding cost of FD-EVN with EVNE has more than 25% saving over FI-EVN with EVNE. In addition, the EVNE could achieve at most 30% embedding cost reduction through employing the substrate bandwidth resources sharing.

D. Migration Frequency

As mentioned above, FD-EVN algorithms achieve higher acceptance ratio and lower embedding cost at the cost of more node migration after facility node failure, which would cause service interruption and should be examined carefully, especially for the application with SLA constraints. We run our simulation in 50000 time unites, which corresponds to about 5000 requests on average in each instance of simulation. The migration frequency after random facility node failure is presented in Fig. 8 in terms of the number of VN nodes.

More specifically, the node migration frequency after failure in FI-EVN is the same as the number of affected VN requests since, in each affected VN request, only the failed facility node would be migrated to its backup host node. Accordingly, Fig. 8 shows that the average migration frequency after a facility node failure in an 8–10 nodes VN request is 3 or 4; and for a 5–8 nodes VN request, it is almost 2. Also, FD-EVN:Random and FD-EVN:MinFir are subject to the same level of migration frequency.

VI. Conclusion

Providing VN survivability in the presence of facility node failures is critical for emerging applications using paradigms such as FCNS and Cloud Computing. To provision survivable VN which can recover from any single facility node failure, in this work, we have taken a holistic approach to provide reliable VN services by first enhancing a service request's resilience with built-in redundancy in a proactive way, and then embedding it. Particularly, we have proposed and formulated a two-step failure dependent VN protection (FD) approach, including designing an EVN and EVN embedding. Our novel failure-dependent VN enhancement has shown to be resource efficient and in particular, outperforms other failure independent two-step approach and previous one-step approach (in which redundant substrate resources are allocated after VNE), in terms of acceptance ratio and embedding cost. However, as a tradeoff, all the performance improvement is achieved at the cost of requiring more service nodes to migrate after failures. Besides, how to make the VN service to survive random node and link failure, as well as providing elastic survivability for VNs with different survivability requirements, would be considered for future work.

References

[1] D. Andersen *et al.*, "Above the clouds: A berkeley view of cloud computing," Univ. Berkeley, Berkley, CA, USA, Tech. Rep., UCB/EECS-2009–28, 2009.

[2] M. Yu, Y. Yi, J. Rexford, and M. Chiang, "Rethinking virtual network embedding: Substrate support for path splitting and migration," in *Proc. ACM SIGCOMM Comput. Commun. Rev.*, 2008, vol. 38, no. 2, pp. 17–29.

[3] J. Zhang, C. Chen, Y. Xiang, W. Zhou, and A. V. Vasilakos, "An effective network traffic classification method with unknown flow detection," *IEEE Trans. Netw. Service Manage.*, vol. 10, no. 2, pp. 133–147, 2013.

[4] C. Kachris and I. Tomkos, "A Survey on optical interconnects for data centers," *IEEE Commun. Surveys Tutorials*, vol. 14, no. 4, pp. 1021–1036, Fourth Quarter 2012.

[5] H. Yu, C. Qiao *et al.*, "Survivable virtual infrastructure mapping in a federated computing and networking system under single regional failures," in *Proc. IEEE GLOBECOM*, 2010, pp. 1–6.

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

[7] Y. Zhu and M. H. Ammar, "Algorithms for assigning substrate network resources to virtual network components," in *Proc. IEEEINFOCOM*, 2006, pp. 1–12.

[8] W. Szeto, Y. Iraqi, and R. Boutaba, "A Multi-Commodity flow based approach to virtual network resource allocation," in *Proc. IEEE GLOBECOM*, 2003, vol. 6, pp. 3004–3008.

[9] I. Fajjari, N. Aitsaadi, G. Pujolle, and H. Zimmermann, "VNE-AC: Virtual network embedding algorithm based on ant colony metaheuristic," in *Proc. IEEE Int. Conf. Commun.*, 2011, pp. 1–6.

[10] J. Lischka and K. Holger, "A virtual network mapping algorithm based on subgraph isomorphism detection," in *Proc. ACM SIGCOMM Workshop Virtualized Infrastructure Syst. Archit.*, 2009, pp. 81–88.

[11] M. Chowdhury, M. R. Rahman, and R. Boutaba, "ViNEYard: virtual network embedding algorithms with coordinated node and link mapping," *IEEE /ACM Trans. Netw.*, vol. 20, no. 1, pp. 206–219, Feb. 2012.

[12] I. Houidi, W. Louati, and D. Zeghlache, "A distributed virtual network mapping algorithm," in *Proc. IEEE Int. Conf. Commun.*, 2008, pp. 5634–5640.

[13] I. Houidi, W. Louati, D. Zeghlache, P. Papadimitriou, and L. Mathy, "Adaptive Virtual Network Provisioning," in *Proc. ACM SIGCOMM Workshop Virtualized Infrastructure Syst. Archit.*, 2010, pp. 41–48.

[14] J. Xu, J. Tang, K. Kwiat, W. Zhang, and G. Xue, "Survivable virtual infrastructure mapping in virtualized data centers," in *Proc. IEEE 5th Int. Conf. Cloud Comput.*, 2012, pp. 196–203.

[15] M. Rahman, I. Aib, and R. Boutaba, "Survivable virtual network embedding," in *Proc. 9th IFIP tc 6th Int. Conf. Network.*, 2010, pp. 40–52.

[16] M. Rahman and R. Boutaba, "SVNE: survivable virtual network embedding algorithms for network virtualization," *IEEE Trans. Netw. Service Manag.*, vol. 10, no. 2, pp. 105–118, 2013.

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

[18] Y. Chen, J. Li, T. Wo, C. Hu, and W. Liu, "Resilient virtual network service provision in network virtualization environments," in *Proc. 16th Int. Conf. Parallel Distrib. Syst.*, 2010, pp. 51–58.

[19] H. Yu, C. Qiao *et al.*, "Migration based protection for virtual infrastructure survivability for link failure," in *Proc. Opt. Fibre Commun. Conf. Expo. /Nat. Fiber Opt. Eng. Conf.*, 2011, pp. 1–3.

[20] C. Qiao, B. Guo, S. Huang, and W. Gu, "A novel two-step approach to surviving facility node failure," in *Proc. Opt. Fibre Commun. Conf. Expo. /Nat. Fiber Opt. Eng. Conf.*, 2011, OWAA3, pp. 1–3.

[21] X. Yu, C. Qiao, T. Wang, and X. Liu, "Survivable logical topology design for distributed computing in WDM networks," in *Proc. Opt.Fibre Commun. Conf. Expo. /Nat. Fiber Opt. Eng. Conf.*, 2009, OMO3, pp. 1–3.

[22] H. Yu, C. Qiao *et al.*, "Cost efficient design of survivable virtual infrastructure to recover from facility node failure," in *Proc. Int. Conf. Commun.*, 2011, pp. 1–6.

[23] X. Liu, C. Qiao, and T. Wang, "Robust application specific and agile private (asap) networks withstanding multi-layer failures," in *Proc. Opt. Fibre Commun. Conf. Expo. /Nat. Fiber Opti. Eng. Conf. /NFOEC*, 2009, OWY1, pp. 1–3.

[24] W.-L. Yeow *et al.*, "Designing and embedding reliable virtual infrastructures," in *Proc. Second ACM SIGCOMM Workshop Virtualized Infrastructure Syst. Archit.*, 2010, pp. 33–40.

[25] M. Juenger *et al.*, *50 Years of Integer Programming 1958–2008: From the Early Years to the State-of-the-Art*. New York, NY, USA: Springer, 2010.

[26] D. Justice and A. Hero, "A binary linear programming formulation of the graph edit distance," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 8, pp. 1200–1214, Aug. 2006.

[27] H. W. Kuhn, "The Hungarian method for the assignment problem," *Naval Res. Logistics*, vol. 2, pp. 83–97, 1955.

[28] G. Karakostas, "Faster approximation schemes for fractional multicommodity flow problems," in *Proc. 13th Annu. ACM-SAIM Symp. Discrete Algorithms*, 2002, pp. 166–173.

[29] E. W. Zegura, K. L Calvert, S. Bhattacharjee *et al.*, "How to model an internet work?," in *Proc. IEEE INFOCOM*, 1996, pp. 594–602.

Bingli Guo received the B.S. degree in telecommunication engineering from Jilin University, Jilin, China, in 2005, and the Ph.D. degree in electromagnetic field and microwave technology from the Beijing University of Post and Telecommunication, Beijing, China, in 2011. He is currently a Research Associate at the High Performance Network group, University of Bristol, U.K. From 2011 to October 2013, he was in State Key Lab of Advanced Communication Systems & Networks in Peking University. His research interests include integrated routing algorithm in converged IP and optical network, survivability strategies in integrated network and cloud computing system, and control plane in software defined optical network.

Chunming Qiao received the B.S. degree in computer science and engineering from the University of Science and Technology, Anhui, China, in 1985, and the Ph.D. degree in computer science from the University of Pittsburgh, Pennsylvania, USA, in 1993. Now, he is the Director of the Lab for Advanced Network Design, Analysis, and Researc with current foci on survivability/availability issues in cloud computing, cyber transportation systems, and smartphone systems. He has published extensively with an h-index of about 60 (according to Google Scholar). He pioneered research on optical burst switching in 1997, and one of his paper has received more than 2,000 citations. In addition, his work on integrated cellular and ad hoc relaying systems in 1999 is also recognized as the harbinger for today's push towards the convergence between heterogeneous wireless technologies, and has been featured in Business Week and Wireless Europe, as well as at the websites of New Scientists and CBC. His research has been funded by several major IT and telecommunications companies including Alcatel Research, Fujitsu Labs, Cisco, Google, NEC labs, Nokia, Nortel Networks, Sprint Advanced Technology Lab, and Telcordia.

Dr. Qiao has given a dozen of keynotes, and numerous invited talks on the above research topics. He has chaired and cochaired a dozen of international conferences and workshops. He was an editor of IEEE TRANSACTIONS ON NETWORKING AND TRANS ON PARALLEL AND DISTRIBUTED SYSTEMS, and a guest editor for several IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS issues. He was the Chair of the IEEE Technical Committee on High Speed Networks and the IEEE Subcommittee on Integrated Fiber and Wireless Technologies which he founded. He was elected to IEEE fellow for his contributions to optical and wireless network architectures and protocols.

Jianping Wang (M'03) received the B.S. and the M.S. degrees in computer science from Nankai University, Tianjin, China, in 1996 and 1999, respectively, and the Ph.D. degree in computer science from the University of Texas at Dallas, Richardson, TX, USA, in 2003. She is currently an Associate Professor at the Department of Computer Science, City University of Hong Kong, Hongkong. Prior to joining City University of Hong Kong, she was at Georgia Southern University and the University of Mississippi as an Assistant Professor. Her research interests include dependable networking, wireless networking, optical networking, cloud computing, service-oriented networking, data center networks, integration of Fiber-Wireless networks, and network coding. She currently serves as an Associate Editor of the IEEE TRANSACTIONS ON MOBILE COMPUTING, IEEE COMMUNICATIONS LETTERS, and *Wiley's Security and Communication Networks Journal*.

Hongfang Yu received the B.S. degree in electrical engineering from Xidian University, Xi'an, China, in 1996. She received the M.S. and Ph.D. degrees in communication and information engineering from the University of Electrical Science and Technology of China (UESTC), Chengdu, China, in 1999 and 2006, respectively.

She is currently with the School of Communication and Information Engineering, UESTC, where she is a Professor.

In recent years, she has authored/coauthored more than 40 research papers on international journals and conferences. Her research interests include network virtualization, software defined networks, and next generation Internet.

Yongxia Zuo received the B.S. degree in computer science and technology from the Changchun University of Technology, Changchun, China, in 2005. She received the M.S. degree in mechanical design and theory from Jilin University, Changchun, China, in 2008. She is currently working toward the Ph.D. degree in mechanical design and theory from Jilin University, Changchun, China. Her research interests include detection and diagnosis of pavement crack, and image processing.

Juhao Li received the B.S. and Ph.D. degrees from the Department of Electronics Engineering of Peking University, Beijing, China in 1999 and 2009, respectively. During 1999–2000, he had been engaged in ZTE Corporation. In 2009, he was in the State Key Laboratory of Advanced Optical Communication Systems & Networks at Peking University, first as a Postdoctoral Research Scientist, as a Lecturer from 2011, as an Associate Professor from 2012. His current research interests include optical switching network, passive optical network, and high-speed optical fiber transmission system.

Zhangyuan Chen received the B.S., M.S., and Ph.D. degrees from the Peking University of China, Beijing, China, in 1991, 1994 and 1997, respectively. Since 1999, he has been with the State Key Laboratory of Advanced Optical Communication Systems and Networks in Peking University as an Associate Professor (1999–2004), and Professor (2004–present) of Electronics Engineering. He w as a visiting scholar in the EECS Department of UC Berkeley (2005–2006). His main research interest is in the area of optical communication network, transmission system, and photonic devices. Dr. Chen is the Council Member of China Institute of Communications, and a member of the IEEE, Optical Society of America, and the Chinese Optics Society.

Yongqi He He received the B.S. degree in physics from Beijing Normal University, Beijing, China, in 1983, the M.S. degree in optical communications from the Beijing University of Posts and Telecommunications, Beijing, China, in 1987, and the Ph.D. degree in optical communications from the Technical University of Denmark, Lyngby, Denmark, in 1996. He is a Professor at the State Key Laboratory of Advanced Optical Communications and Networks, School of Electronic Engineering and Computer Science, Peking University, Beijing, China. His research interests include optical transport networks, high-speed DWDM systems, and future flexible optical network control issues.