One Node-Fault Tolerant Graph with Node's types for survivable Virtual Network Request with Service Type

Kun Xie[1], *Member, IEEE*,Heng Tao[1]
[1] College of Computer Science and Electronic Engineering,Hunan University, China

***Abstract*—As virtualization is becoming a promising way to support various emerging application, provisioning survivability to requested embeded virtual network(VN) which is embedded in substrate networks (SN) in a resource efficient way is important. In this paper, 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 embeded VN with additional computing and communication resources and design the Enhanced eVN (or EeVN) after embedding it to the substrate in order to further reduce the amount of substrate resources needed to survive a single facility node failure. In a virtualized infrastructure where physical resources are shared, a single physical server failure will terminate several virtual serves and crippling the virtual infrastructures which contained those virtual servers. To guarantee some level of reliability, each virtual infrastructure, at instantiation, should be augmented with backup virtual nodes and links.**

I. Introduction

In network virtualization, the primary entity is the Virtual Network (VN). A VN is a combination of active and passive network elements (network nodes and network links) on top of a Substrate Network (SN). Virtual nodes are interconnected through virtual links, forming a virtual topology. By virtualizing both node and link resources of a SN, multiple virtual network topologies with widely varying characteristics can be created and co-hosted on the same physical hardware.

With network infrastructure rapidly becoming virtualized, shared and dynamically changing, it is essential to have strong reliability support from the physical infrastructure, since a single physical server or link failure affects several shared virtual entities. Providing reliability is often linked with over-provisioning both computational and network capacities, and employing load balancing for additional robustness. However, we do not talk about the capacity requirement and geographical location requirement of virtual node, and the bandwidth capacity requirement of virtual link in this paper.

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 [1], [2], [3], [4], [5]. 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.

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 [6] and Failure Independent Protection (FIP) [7], 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.

To survive a substrate link failure, pre-computed alternative paths inVN 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 [2], while authors in [4] propose a proactive backup approach (considering the backup resources sharing) to avoid service disruption in reactive restoration approach and improve the substrate resource utilization.

Through synchronization [8], [9] and migration techniques [10], [11] on virtual machines and routers, we suppose that every virtual network have specific service functions, in addition that fault tolerant can be introduced at the virtualization layer [7].

As nodes often represent expensive components (servers) and

edges represent less expensive interconnections (links), most attention has been devoted to node faults, i.e., the removal of nodes (and their incident edges), rather than edge faults where only edges are removed.

Our paper focus on the problem of resource deployment for virtual network infrastructure with reliability guarantee. Practical issue such as forwarding graph embedding and chain composition.

The organization os this paper is as follow. In the next section, we briefly describe the background, notations and define reliability in Sec.IV.

II. EeVSN Design Formulation

the software-defined NFV architecture further offers agile traffic steering and joint optimization of network functions and resources 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.

write the bandwith and compputting resource between FI and FD.

A. VN Reqeust Model

A task graph for a VN request is an undirected attributed graph, denoted asGV (V V ,EV), where V V corresponds to a set of task nodes andEV denotes a set of bidirectional edges among the VN nodes. Each task node specifies the needed computing resource cVs , and each edge specifies the requested amount of communication (bandwidth) resource bV st .

B. EeVSN 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).

In conclusion, the formulation indicates that this problem is a BQP problem and the solution of this problem is a series of permutation matrix which determine the migration reassigning approach after each node failure. Thus, its computation complexity is (N)N , whereN is the computation complexity for one node failure.

III. EeVSN 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.

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.

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.

IV. Problem Formulation

We consider a resource deployment problem in a virtualized network infrastructure, such as a data center, where the virtualized resources can be leased with reliability guarantees. Each resource lease request is modeled as undirected graph $G = (N, E, S)$. N is a set of compute nodes, E is a set of edges and S is a partial set of service functions. We call this a virtual network(VN).

A. Reliability,Survivable

Reliability is guaranteed on the set of critical nodes $C \subset N$ of a VN through redundant virtual nodes with backup nodes set $B(V, S)$. A backup (redundant) node b must be able to assume full execution of a failed critical node c. Hence, the backup node must have sufficient resources in terms of computation

B. How many backups?

The number of redundant nodes depend on the physical mapping, and the failure models of both the physical nodes and the virtual infrastructure.

The problem is to allocate least resources for a VN G,including redundancy such that a reliability guarantee of at least r is achieved.

C. Node-Fault Graph

1) NFT

We refer to the concept of k-NFT [12]. Let $G(V, E)$ be a graph with n nodes and q edges. An (n+k)-node graph G^* is k-node fault-tolerant, or k-NFT, with respect to G if every graph $G^* - R$ obtained by removing any nodes set R of $k > 0$ nodes from G^* contains G. Generally speaking, G is subgraph isomorphism of $G^* - R$. We will refer to G^* as a k-NFT supergraphs of G or simply as a k-NFT(G). We also say $G^* \cong$ k-NFT(G), the set of all k-NFT(G) supergraphs of G. The complete graph K_{n+k} of $n + k$ nodes is trivially a k-NFT supergraph of every G that contains up to n nodes. We are concerned mainly with k-NFT graphs that satisfy the following optimality criterion: If G^* has the smallest number $|E(G^*)|$ of edges among all (n + k)-node supergraphs that are k-NFT with respect to G, then G^* is optimally k-NFT with respect to G. The number NFT_{ec}(G,k) =$|E(G^*) - E(G)|$ is called the k-NFT

edge cost of G. The number NFT_{nc}(G,k) =$|V(G^*) - V(G)|$ is called the k-NFT node cost of G, however NFT_{nc}(G,k) = k with respect with k node fault tolerant graph of graph G .

2) SNFT

When every nodes of graph $G(V, E, S)$ have a specific flags set S(corresponding to service functions) and every nodes is marked by only one specific flag, which is denoted by node flag. An (n+k)-node graph G^o is k-specific node fault tolerant, or k-SNFT, with respect to $G(V, E, S)$ if every graph $G^o - R$ obtained by removing any nodes set R of $k > 0$ nodes from G^o contains G, G is subgraph isomorphism of $G^o - R$ and the node flag of any nodes n of G belong specific flags set of node n^o corresponding to node n. We will refer to G^o as a k-SNFT supergraphs of G or simply as a k-SNFT(G). The number $SNFT_{nc}$(G,k) =$|V(G^o) - V(G)|$ is called the k-SNFT node cost of G. The number $SNFT_{ec}$(G,k) =$|E(G^o) - E(G)|$ is called the k-SNFT edge cost of G.

3) SNFT with B

After our simple inference, if we set the $SNFT_{nc}$(G,k) is the main cost than edge cost, the number of $SNFT_{nc}$(G,k) must equal k, moreover the added node could have any specific flags. but the added node should be limited in specific flags set after our analyzing the real-world practical phenomenon.

Suppose added nodes is from backup nodes set $B(V, S)$, where every nodes of backup nodes set B have a specific flags set. The k-SNFT(G, B) is that requesting a k specific node fault tolerant graph of graph G and added nodes belong backup nodes set B.

D. Survivable Virtual Network Request

In any real virtual networks, any nodes of virtual network are deployed with a service function and the physical node could run some various service functions. After virtual network VN is embedded into substrate network, a node of virtual network could fail so that we should obtain survivable VN to be embedded into substrate network so that recovery from the failure even though one node of virtual network failed. The added nodes of survivable virtual network are embedded into substrate network ultimately, the added nodes have specific service functions similarly.

There are different combinations of service function in any nodes of virtual network, and there is only one type of service function running on the corresponding virtual node at that moment.

There exist many backup virtual nodes $B(V, S)$ for guaranteeing maintaining former topological structure of virtual network request $G(V, E, S)$ once one fault node v appeared in virtual network request $G(V, E, S)$. Solving the survivable request of virtual network is of equivalence with asking 1-SNFT(G,B) of graph G and backup nodes set B.

An example of a typical survivable request of virtual network is show in Fig.1.

Generally speaking, survivable request of virtual network is 1-specific node fault-tolerant of virtual network $G(V, E, S)$ with backup nodes set $B(V, S)$. Inserting any edges among nodes $V \cup B$ to construct G^o as shown in Fig.2,3, for which G is a subgraph of $G^o - v(\forall v \in V \cup B^o, B^o \subset B)$.

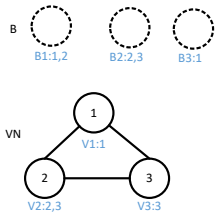

Fig. 1. SVNR

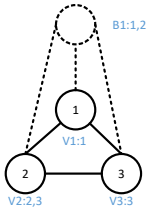

Fig. 2. optimal SVNR

E. reduce immigration when node fail

However, this poses a limitation since the result only guarantees graph isomorphism and not equality. In other words, there may be a need to physically swap remaining VMs while recovering from some failure in order to return to the original infrastructure G. Recovery may then be delayed, or require more resources for such swapping operations.

F. Survivable Virtual Network Request

We would link some edges amongst nodes $V \cup B$ to construct 1-SNFT graph of arbitrary graph G as show Fig.2,3, when only one node of VN failed, the former graph of VN is also subgraph isomorphism of graph of SVN. For example, when node v3 failed the node v2 run service 3 and backup node run service 2 so that the graph of VNR is subgraph isomorphism of SVN-v3 as shown in figure 4,5,6,the red edge and red node is disable.

It is our objection that firstly minimize number of added backup nodes, then minimize number of added edges to construct

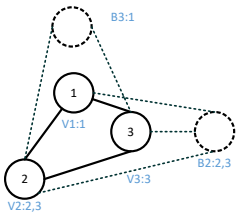

Fig. 3. non-optimal SVNR

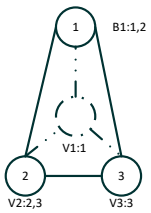

Fig. 4. node v3 failed in optimal SVNR

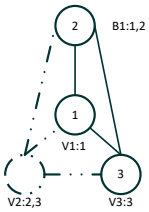

Fig. 5. node v3 failed in optimal SVNR

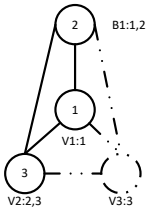

Fig. 6. node v3 failed in optimal SVNR

specific node fault tolerant graph. The graph of SVN, which is 1-NFT with node service function with respect to the graph of VN.

G. Complexity of Problem

this computation complexity of k-SNFT problem k-NFT($G(V,E,S)$) is NP, whose proof is below. when just exist one service function type in SVNR, the problem is degenerated into one-node fault-tolerant graph problem. when SVNR just have one type of service function, we must add only one backup node and add some edges into graph of VN to construct SVN, meantime the simplified problem is of equivalance to problem that one-node fault-tolerant graph problem which is NP.

when inserted node sets is previous given, the problem k-SNFT($G(V,E,S),B(V,S)$) is subproblem of k-SNFT($G(V,E,S)$).

Suppose topological structure of graph G is path P, k-SNFT($P(V,E,S),B(V,S)$) is NP, the proof is below, exist one node n(n$\in V$) and we add backup nodes and edge to protect the node n, the former graph P firstly is changed into $P+n$, nevertheless topological structure of $P+n$ may is not path, or be tree and non-tree graph. then protect other nodes of V.

the computation complexity of k-SNFT($T(V,E,S),B(V,S)$) is similarly available.

edge fault tolerant problem is subgraph of node fault tolerant problem

V. Requesting Survival Virtual Network Using Integer Linear Programming

The one of contributions of this paper is the modelization of specific node fault tolerant problem as an optimization. The general technique used to solve this problem is a combined algorithm consisting of enumeration and Ullmann Algorithm [13], which is proposed as a algorithm of exponential complexity which is brute force method briefly. More precisely, we propose an Integer Linear Program(ILP) [14] model of this problem. In the following, We briefly explain how to solve a problem using ILP.

A. Modeling the SNFT problem

In this section, we formulate the problems discussed in the previous sections with Integer Linear Program(ILP) method. Though we have proved that these problems are NP-complete, as we will demonstrate later, the ILP formulation of the problem provides a very viable tool for solving it, for most real-world virtual network request which typically have less than a few hundred nodes and edges. Throughout this section, we will focus exclusively on the survival virtual network request $[G(V,E,S),B(V,S))]$, we assume that G is simple directed service label graph.

To begin with, we introduce some necessary notation:

MBG: $[mbg_{u,v}]_{(|V|+|B|)\times(|V|+|B|)}$, the adjacency matrix of graph G^o. where

$$mbg_{u,v}=\begin{cases}1\ if\ edge\ e\ originates\ from\ u\ to\ v\\ 0\ otherwise\end{cases}$$

MAG: $[mag_{u,v}]_{|V|\times|V|}$, the adjacency matrix of arbitrary graph G. Where

$$mag_{u,v}=\begin{cases}1\ if\ edge\ e\ originates\ from\ u\ to\ v\\ 0\ otherwise\end{cases}$$

T^l: $[t^l_{u,v}]_{|V|\times(|V|+|B|)}$, The l-th node permutation matrix against l-th node failed. Where

$$t^l_{u,v}=\begin{cases}1\ if\ node\ u\ transformed\ to\ node\ v\\ 0\ otherwise\end{cases}.$$

MBS: $[mbs_{u,v}]_{(|V|+|B|)\times|S|}$, incidence matrix of specific services of the SNFT graph G^o. where

$$mbs_{u,v}=\begin{cases}1\ if\ node\ u\ have\ specific\ flag\ v\\ 0\ otherwise\end{cases}.$$

MAS: $[mas_{u,v}]_{|V|\times|S|}$, incidence matrix of specific services of the arbitrary graph G. where

$$mas_{u,v}=\begin{cases}1\ if\ node\ u\ have\ specific\ flag\ v\\ 0\ otherwise\end{cases}.$$

$AugB$:$[aug_i]_{|B|}$, whether the i-th backup nodes is used or not. where

$$aug_i=\begin{cases}1\ if\ node\ i\ is\ used\\ 0\ otherwise\end{cases}.$$

The problem of requesting 1-specific node fault tolerant graph $[G(V,E,S),B(V,S))]$ can be formulated as the folowing ILP problem:

$$such\ as:\ T^l*(T^l*MBG)^{'}\geq MAG \tag{1}$$

$$\sum_{1\leq v\leq(n+b)}T^l_{iv}=1(1\leq i\leq n) \tag{2}$$

$$\sum_{1\leq u\leq n}T^l_{uj}\leq 1(1\leq j\leq(n+b))$$

$$\sum_{1\leq u\leq n,1\leq v\leq(n+b)}T^l_{uv}=n$$

$$T^k_{uk}=0(1\leq u\leq n) \tag{3}$$

$$MAG_{uv}\leq MBG_{uv}(1\leq u\leq n,1\leq v\leq n) \tag{4}$$

$$MAS\leq T^l*MBS \tag{5}$$

$$\frac{\sum_{1\leq v\leq(n+b)}MBG_{(n+i)v}+n-1}{n}\leq B_i(1\leq i\leq b) \tag{6}$$

$$\tag{7}$$

- Constraint(1) guarantees that when the l-th critical node of graph G failed, we could find subgraph which is subgraph isomorphism of SNFT graph G^o. we represent the mapping relation as permutation matrix in view of Ullman [13]
- Constraint(3) implies that operation of the transform matrix of mapping relation is appropriate and correct format in respect with one node failure.
- Constraint(4) guarantees that SNFT graph G^o must be the augmented graph of previous graph G.
- Constraint(5) guarantees that when the l-th node failed all nodes of the transformed graph still have specific flag, which is specific flag of corresponding node of previous graph G.

B. Objective Function and Approximation

We seek to minimize the amount of resource used for a VInf. The object function of the adapted ILP is then

$$min : \alpha \sum_{1 \le i \le b} B_i + \beta \times (\sum_{1 \le u \le (n+b) 1 \le v \le (n+b)} MBG_{uv} - \sum_{1 \le u \le n, 1 \le v \le n} MAG_{uv}) \tag{8}$$

where α and β are node and link weights, respectively. To achieve minimize the amount of nodes, the weight can be set as $(n^2 + 1)$ and 1, respectively.

The presence of the boolean variables turns the linear program into a NP-Hard problem. An alterative is to relax the boolean variables to real-value variables, obtain an approximate virtual node transformation.

$$dp[i][W_1][W_2]\dots[W_m] = \left\{ \begin{array}{l} dp[i-1][W_1 - w_i][W_2]\dots[W_m] + v_i \\ dp[i-1][W][W_2 - w_i]\dots[W_m] + v_i \\ dp[i-1][W][W_2]\dots[W_m - w_i] + v_i \end{array} \right\} \tag{9}$$

$n * \prod_{i=1}^{m} C_i$
$m * n * \prod_{i=1}^{m} C_i$

VI. Idea

That is, the computation complexity of k-SNFT problem is NP, how do we propose reasonable heuristic methods to solve the problem? Moreover, existed optimal algorithms is all exponential time algorithm.

For example, we suppose a general graph as show in Fig.7, which consist of tree, path and biconnected component, therefore requesting the k-SNFT graph of the original graph is intractable. That is, we firstly utilized biconnected component algorithm to partition the original graph into two styles of graph(tree graph and biconnected component graph. Besides, small granularity's graphs is easily to be dealed and calculated as show in fig.8, however the original problem of k-SNFT is reduced into approximated optimal answer problem. Then we obtain any node fault-tolerant graph with service functions of every graphs of small granularity. At last, we integrate all 1-SNFT graphs of small granularity into the final 1-SNFT graph original graph. The advantages of our heuristic method are minimizing the effort required for reconfiguration and parallelly processing every graphs of small granularity which is easily to be solved.

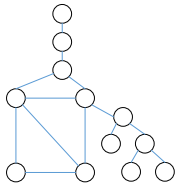

Fig. 7. General Graph

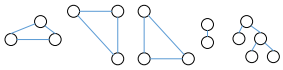

Fig. 8. Elementary Graph

A. Non-optimal to Redundant Links Without swapping

To overcome the aforementioned limitations and time complexityIV-G, we choose decompose original general graph into elementary graphs of small granularity, at the expense of incurring more redundant resources for non-optimal solution but time-efficient. Adding link to elementary graph is much more stable for whole virtual network and reduce swapping/rearrangement problem of whole graph.

VII. 1-SNFT graph construction for arbitrary graph

The proposed algorithm of exploring node fault tolerant for arbitrary graph with specific flags is based on an extension of essential idea of Harary and Hayes' theorem [12]. The heuristic algorithm applies the concept of Divide and Conquer to obtain a 1-SNFT graph with relatively(nonoptimal) less amount of edges. It works on arbitrary graph with specific flags consisting of elementary tree T_e and circles C_e which can be solved by our proposed optimal algorithms easily and concurrently such as Integer Linear Programmer(ILP) in sec.V. The elementary

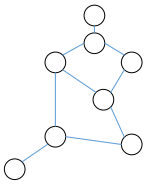

Fig. 9. add node for request minimum cycle of biconnected component graph from Decomposition of Originnal Graph

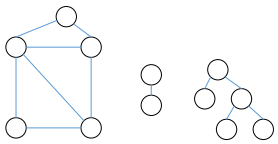

Fig. 10. BCC or non-BCC($subG_i$)

circles are the minimal circles which do not contain further circles inside.

Constructing the 1-SNFT graph for arbitrary flag label graph through proposed divided and conquer approach in Alg.1. After initializing the G^o as emptyset, the graph G is decomposed to individual elementary graphs G_e in Alg.2, whose 1-SNFT supergraphs G_i^o are further constructed. After that the G^o is multiply assigned by graph-theoreticly union between the G^o and current G_i^o iteratively.

The standard graph operations such as union, join, and composition [15] Binary operations create a new label graph from two initial ones $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$, such as: graph union: $G_1 \cup_g G_2 = (V_1 \cup V_2, E_1 \cup E_2)$.

The construct of G_i^o with respect to G_i is based on the approach from Sec.V.

Algorithm 1 Construct 1-SNFT(G,B) for arbitrary graph $G(V, E, S)$ and backup nodes $B(V, E)$

Require: $G(V, E, S)$: arbitrary graph with specific flags; $B(V, S)$ backup nodes with specific flags.
Ensure: 1-SNFT(G,B): specific node fault tolerant of graph G.

$G^o \leftarrow \emptyset$
$G_e \leftarrow$ DecomposeGraph($G(V, E, S)$)8
for all G_i such that $G_i \in G_e$ **do**
  $G_i^o \leftarrow$1-SNFT-ILP(G,G_i,B)
  $G^o \leftarrow (G_i^o \cup_g G^o)$
end for
for all $i, j \in L; u, v \in V_e$ **do**
  $G^o \leftarrow G^o \cup E_{u,v}$
end for
return G^o

BCC-Decomposition: decompose a graph into two kinds of graph, tree and biconnected component graph.

ShortestHopPath: obtain the shortest hop path in Graph $subG_i$ of source node is v and destination node is u. Hence, there are many redundant cycles which is the same topological.

Algorithm 2 Decompose Graph

Require: $G(V, E, S)$: arbitrary graph with specific flags.
Ensure: G_e: elementary graphs of arbitrary graph G.

$G_e \leftarrow \emptyset$
$subG_i \leftarrow$ BCC-Decomposition($G(V, E, S)$)10
if $subG_i$ is biconnected graph **then**
  $C_e \leftarrow \emptyset$
  for all $E_{uv} \in E$ **do**
    $P \leftarrow$ShortestHopPath(E_{uv})
    $C_e \leftarrow C_e \cup (C = P \cup E_{uv})$
  end for
  $C_e \leftarrow$ReduceRedundantCycle(C_e)Alg.3
  $G_e \leftarrow G_e \cup C_e$
else
  $G_e \leftarrow G_e \cup subG_i$
end if
return G_e

ReduceRedundantIsormophismLabelGraph: Some cycles of cycle set C_e is isomorphic in labeled graph G so that we should sieve them for reducing unnecessary computation.

ConstructGraph: Remove the all edges of cycle, then add a new augmented node and link between all nodes of cycle and the augmented node as show in Fig.9.

MinimumSubsetCover: minimizing augmented nodes who link with all nodes of cycles is minimum subset cover problem, which is non-polynomial complete problem, and taking greedy algorithm solving the NPC problem [16].

Algorithm 3 Reduce Redundant Cycle

Require: C_e: cycle set.
Ensure: C_{min}: Irredundant Cycle Set.

$C^b \leftarrow \emptyset$
$C_{min} \leftarrow \emptyset$
$C_e \leftarrow$ReduceRedundantIsormophismLabelGraph(C_e)
for all G_i such that $G_i \in G_e$ **do**
  $G_i^b \leftarrow$ConstructGraph(G_i)
  $G^b \leftarrow (G_i^b \cup_g G^b)$
end for
$I \leftarrow$MinimumSubsetCover(G^b)
for all G_i such that $G_i \in G_e$ **do**
  if i $\in I$ **then**
    $C_{min} \leftarrow (C_{min} \cup G_i)$
  end if
end for

A. Consider different adjacent cycles

MCC: $[mcc_{u,v}]_{(|V|+|B|)\times|V}$, incidence matrix of . where $mcc_{u,v} = \begin{cases} 1\ if \\ 0\ otherwise \end{cases}$.

VIII. Evaluation

In the proposed two step for $EVSNR$, we employ a proactive failure dependent protection 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 failure independent protection 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. when a substrate node fail, which is placed with two virtual node.

A. Simulation Settings

For any VN request, the number of VN nodes is randomly determined by a uniform distribution between 4 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 5 to 10, 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 [6], [17], which

means $\alpha = 3$. The SN topologies used are randomly generated with 40 nodes using the GT-ITM tool [18] The computing (bandwidth) resources of the substrate nodes (links) are real numbers uniformly distributed between 30 and 50 (100 and 150).

B. Acceptance Ratio

In this section, the service acceptance ratio for VN with the proposed survivable approach is examined, as well as a typical integer liner program method. 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 $EVSNR$ 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 FDEVN: 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 achieves almost 15long 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 8resources consumption for survivability purpose.

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.

We further compare the embedding cost of different EVN design algorithms with (denoted as +Shared) or without (denoted as +NoShared) consideration of the substrate resources sharing in EVNE approach. Generally, with regular VN arriving and leaving, embedding cost.

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 30000 time unites, which corresponds to about 2000 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.

The physical infrastructure consists of 40 compute nodes with capacity uniformly distributed between 50 and 100 units. These nodes are randomly connected with a probability of 0.4 occurring between any two nodes, and the bandwidth on each physical link is uniformly distributed between 50 and 100 units.

Accepted VNR Ratio	Ratio of virtual networks that were successfully e
Stress	Average number of virtual links/nodes that have b
Utilization	Bandwidth/CPU utilization of substrate links/node
Path Length	Length of communication paths assigned to virtua
Cost, Revenue, and Cost/Revenue	Cost: Sum of CPU and bandwidth resources being
Active Nodes	Number of nodes that need to be active in order t
Power Consumption	Power consumption of all Active nodes. Several p
Runtime	Average runtime of the algorithm.
Initialization Overhead	Some algorithms come with initialization cost (in
Hidden Hops Ratio	Ratio of hidden hops, e.g., the number of nodes o
Communication Overhead	Communication overhead of distributed algorithms

VInf requests arrive randomly over a timespan of 800 time slots and the inter-arrival time is assumed to follow the Geometric distribution at a rate of 0.75 per time slot. The resource lease times of each VInf follows the Geometric distribution as well at a rate of 0.01 per time slot. A high request rate and long lease times ensures that the physical infrastructure is operating at high utilization. Each VInf consists of nodes between 2 to 10, with a compute capacity demand of 5 to 20 per node. Up to 90% of these nodes are critical and all failures are independent with probability 0.01. Connectivity between any two nodes in the VInf is random with probability 0.4, and the minimum bandwidth on any virtual link is 10 units. There are two main sets of results: (i) scaling the maximum bandwidth of a virtual link from 20 to 40 units while reliability guarantee of every VInf is 99.99%, and (ii) scaling the reliability guarantee of each VInf from 99.5% to 99.995% while the maximum bandwidth of a virtual link is 30 units.

Accept ratio mapping cost migration cost cpu time per Vinf

REFERENCES

[1] J. Xu, J. Tang, K. Kwiat, W. Zhang, and G. Xue, "Survivable virtual infrastructure mapping in virtualized data centers," in *Cloud Computing (CLOUD), 2012 IEEE 5th International Conference on*, pp. 196–203, IEEE, 2012.

[2] M. R. Rahman, I. Aib, and R. Boutaba, "Survivable virtual network embedding.," in *Networking*, vol. 6091, pp. 40–52, Springer, 2010.

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

[4] T. Guo, N. Wang, K. Moessner, and R. Tafazolli, "Shared backup network provision for virtual network embedding," in *Communications (ICC), 2011 IEEE International Conference on*, pp. 1–5, IEEE, 2011.

[5] Y. Chen, J. Li, T. Wo, C. Hu, and W. Liu, "Resilient virtual network service provision in network virtualization environments," in *Parallel and Distributed Systems (ICPADS), 2010 IEEE 16th International Conference on*, pp. 51–58, IEEE, 2010.

[6] 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 *Global Telecommunications Conference (GLOBECOM 2010), 2010 IEEE*, pp. 1–6, IEEE, 2010.

[7] W.-L. Yeow, C. Westphal, and U. C. Kozat, "Designing and embedding reliable virtual infrastructures," *ACM SIGCOMM Computer Communication Review*, vol. 41, no. 2, pp. 57–64, 2011.

[8] T. C. Bressoud and F. B. Schneider, "Hypervisor-based fault tolerance," *ACM Transactions on Computer Systems (TOCS)*, vol. 14, no. 1, pp. 80–107, 1996.

[9] B. Cully, G. Lefebvre, D. Meyer, M. Feeley, N. Hutchinson, and A. Warfield, "Remus: High availability via asynchronous virtual machine replication," in *Proceedings of the 5th USENIX Symposium on Networked Systems Design and Implementation*, pp. 161–174, San Francisco, 2008.

[10] C. Clark, K. Fraser, S. Hand, J. G. Hansen, E. Jul, C. Limpach, I. Pratt, and A. Warfield, "Live migration of virtual machines," in *Proceedings of the 2nd Conference on Symposium on Networked Systems Design & Implementation-Volume 2*, pp. 273–286, USENIX Association, 2005.

	$C_{P_{2.2}}$	$C_{P_{2.3}}$	C_{P_3}	C_{P_4}	$C_{B_{1.1}}$	$C_{B_{1.2}}$	$C_{B_{2.1}}$	$C_{B_{2.4}}$	$C_{B_{3.2}}$	$C_{B_{3.2}}$
R_{V_1}	∞	∞	∞	∞	N+(2)+4+5+3	∞	N+(2)+4+5+3	∞	∞	∞
R_{V_2}	4	∞	∞	∞	∞	N+(3)+4+6	∞	∞	N+(3)+4+6	∞
R_{V_3}	∞	M+(2)+5	5	∞	∞	∞	∞	∞	∞	N+(5)+5+6
R_{V_4}	∞	∞	∞	3	∞	∞	∞	N+(6)+3	∞	∞

	C_{P_1}	C_{P_3}	C_{P_4}	$C_{B_{1.1}}$	$C_{B_{1.2}}$	$C_{B_{2.1}}$	$C_{B_{2.4}}$	$C_{B_{3.2}}$	$C_{B_{3.2}}$
R_{V_1}	4	∞	∞	M+4	∞	N+(2)+4+5+3	∞	∞	∞
R_{V_2}	∞	∞	∞	∞	M+(1)+4+1	∞	∞	N+(3)+4+6	∞
R_{V_3}	∞	6	∞	∞	∞	∞	∞	∞	N+(5)+5+6
R_{V_4}	∞	∞	0	∞	∞	∞	N+(6)+3	∞	∞

	C_{P_1}	$C_{P_{2.2}}$	$C_{P_{2.3}}$	C_{P_4}	$C_{B_{1.1}}$	$C_{B_{1.2}}$	$C_{B_{2.1}}$	$C_{B_{2.4}}$	$C_{B_{3.2}}$	$C_{B_{3.3}}$
R_{V_1}	5	∞	∞	∞	$M+5$	∞	N+(2)+4+5+3	∞	∞	∞
R_{V_2}	∞	6	∞	∞	∞	M+6	∞	∞	N+(3)+4+6	∞
R_{V_3}	∞	∞	M+(2)+1	∞	∞	∞	∞	∞	∞	N+(5)+5+6
R_{V_4}	∞	∞	∞	0	∞	∞	∞	N+(6)+3	∞	∞

[11] Y. Wang, E. Keller, B. Biskeborn, J. van der Merwe, and J. Rexford, "Virtual routers on the move: live router migration as a network-management primitive," in *ACM SIGCOMM Computer Communication Review*, vol. 38, pp. 231–242, ACM, 2008.

[12] F. Harary and J. P. Hayes, "Node fault tolerance in graphs," *Networks*, vol. 27, no. 1, pp. 19–23, 1996.

[13] J. R. Ullmann, "An algorithm for subgraph isomorphism," *Journal of the ACM (JACM)*, vol. 23, no. 1, pp. 31–42, 1976.

[14] A. Schrijver, *Theory of linear and integer programming*. John Wiley & Sons, 1998.

[15] D. B. West *et al.*, *Introduction to graph theory*, vol. 2. Prentice hall Upper Saddle River, 2001.

[16] T. H. Cormen, *Introduction to algorithms*. MIT press, 2009.

[17] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. H. Katz, A. Konwinski, G. Lee, D. A. Patterson, A. Rabkin, I. Stoica, *et al.*, "Above the clouds: A berkeley view of cloud computing," tech. rep., Technical Report UCB/EECS-2009-28, EECS Department, University of California, Berkeley, 2009.

[18] E. W. Zegura, K. L. Calvert, and S. Bhattacharjee, "How to model an internetwork," in *INFOCOM'96. Fifteenth Annual Joint Conference of the IEEE Computer Societies. Networking the Next Generation. Proceedings IEEE*, vol. 2, pp. 594–602, IEEE, 1996.