IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received September 21, 2018, accepted October 24, 2018, date of publication October 30, 2018, date of current version December 3, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2878797

Survivable Virtual Network Link Protection Method Based on Network Coding and Protection Circuit

YUZE SU, XIANGRU MENG, QIAOYAN KANG, AND XIAOYANG HAN
College of Information and Navigation, Air Force Engineering University, Xi'an 710077, China

Corresponding author: Yuze Su (glgiuip@163.com)

This work was supported in part by the National Natural Science Foundation of China under Grant 61401499 and Grant 61871313.

ABSTRACT Network virtualization is believed to be a promising way for the next generation Internet to overcome network ossification. It allows multiple heterogeneous virtual networks (VNs) to be embedded onto the shared substrate network (SN) to offer more flexibility and better manageability. With the increasing deployments of VNs in a variety of networks, how to protect the VNs against the single substrate link failure has become a key challenge. In this paper, we propose a survivable VN link protection method based on network coding and protection circuit. First, we provide an integer linear programming formulation for the survivable VN link protection to maximize the long-term average revenue to cost ratio. Second, a novel protection circuit technology is introduced to augment the VN at the VN level to reduce the backup resource consumption and provide more flexible VN management. Then, an efficient heuristic virtual network embedding algorithm is developed, which makes full use of the limited resources and transforms the single substrate link protection into the single virtual link protection in multiple protection circuits. Finally, the data units from different links are encoded using network coding, which not only provides 1 + N protection to virtual links but also reduces the recovery delay. The evaluation results show that our method not only has the best acceptance ratio and long-term average revenue to cost ratio, but it also greatly enhances the achievable backup sharing and shortens the average network recovery delay.

INDEX TERMS Survivable virtual networks, link protection, single substrate link failure, augmented VN, network coding, protection circuit.

I. INTRODUCTION

With the rapid development of the Internet, network ossification has become the key factor restricting the innovation and development of Internet architecture [1]. Network virtualization is considered to be a promising solution to the ossification of the current Internet. It can decouple the network infrastructure and network services and allow multiple heterogeneous virtual networks (VNs) to share the substrate network (SN) resources through an abstraction, distribution and isolation mechanism. One of the most challenging problems raised in the network virtualization context is the virtual network embedding (VNE) problem, where the customized VN from the service provider (SP) is mapped to the SN managed by the infrastructure providers (InPs). In general, VNE consists of two major components: the embedding of the virtual node (with the CPU resource requirement) to the substrate node and the embedding of the virtual link (with the bandwidth resource requirement) to the substrate path(s). VNE has been proven to belong to the NP-hard problem [2].

Numerous efforts have been devoted to solve the VNE problem and improve resource utilization [3], [4]. These studies focus on heuristic algorithms for the VNE problem by assuming that the SN remains operational at all times. However, failures in the SN are not uncommon due to a variety of reasons and 70% of network failures are single link failures [5]. In network virtualization, a single substrate link failure affects a large number of VNs and users since multiple virtual links are embedded onto one substrate link and the substrate link also carries huge amounts of traffic. It may significantly degrade the service performance and availability. In many applications, a service outage can incur high penalties in terms of revenue and customer satisfaction. Therefore, some survivable VNE (SVNE) algorithms

are proposed. SVNE refers to finding a VNE while guaranteeing the VN survivability in the event of failures.

VN survivability is the ability of a VN to continuously deliver services in compliance with the given requirements in the presence of failures and other undesired events. VN survivability is crucial for both InPs and service providers (SPs). Hence, a number of mechanisms have been proposed to increase VN survivability against link failures [6]–[8]. They can be classified into two categories: protection [9] and restoration [10]. Protection mechanisms can be classified into dedicated protection and shared protection [11]. In the dedicated case, the backup resources only can be used for the dedicated link. Many dedicated protection techniques have been proposed. E.g., in 1:1 protection, the backup link is only used to transmit data units when the primary link fails. In 1 + 1 protection, data units are simultaneously transmitted on two link-disjoint paths, and the stronger signal is selected to receive the data units. It shortens the recovery delay without retransmitting data units. In addition, it is clear that the backup resource consumption is large in these two dedicated protection mechanisms. To reduce backup resource consumption, shared protection is proposed in which the backup resources can be shared with different links. The 1:N protection is developed from the 1:1 protection, and one backup link is used to protect N primary links with the shared backup resources. M:N protection is similar to 1:N except that M backup links are used to protect N primary links.

In addition, unlike traditional network protection, the VN can be augmented at the VN level with some redundant virtual nodes and links first. Then, the augmented VN is embedded onto the SN to achieve its survivability under VN protection. A recent study evaluates the impact of providing survivability at the SN level compared to that at the VN level. The benefit of providing survivability at the VN layer is that it yields more resource efficient embedding compared to providing the same level of survivability at the SN layer. Despite the backup sharing advantages, one can argue that the increased number of signaling and rerouting operations required for VN level survivability may lead to slower restoration compared to ensuring the survivability at the SN layer. However, a recent study empirically evaluates the impact of providing protection at the SN level compared to that at the VN level using a real testbed [12]. The study shows that providing survivability at the VN level has a similar switching response time during a failure, and VN level survivability can accommodate more VNs compared to doing the same at the SN level. This augmentation enables InPs to offload failure recovery decisions to the VN operator, thus providing more flexible VN management. Hence, providing survivability at the VN level is more profitable for InPs.

In the restoration mechanism, the faulty part in the VN can be re-embedded to avoid the huge backup resource consumption. However, network restoration needs much time, which will cause a long network delay. A widely accepted upper bound on the total recovery time from failures is 50 milliseconds, which restricts the application of the restoration mechanisms [13]. Additionally, its re-embedding success rate is usually lower than the protection mechanism because of the limited resources used for the VN reconfiguration.

As seen above, reducing the backup resource consumption and shortening the recovery delay are the most important factors in survivable VN link protection. To reduce backup resource consumption, a number of protection cycles are proposed to share the backup resources with multiple links, e.g., the p-cycle [14]. However, in some situations, the p-cycle is not feasible, and protection may be not suitable [15]. To shorten the recovery delay, network coding is introduced to transmit combinations of data units from multiple links on a protection link. It does not need failure localization, detection and data retransmission, and the recovery delay is greatly reduced.

In this paper, we propose a survivable VN link protection method based on network coding and protection circuit (SVNLPM-NCPC). At first, the problem of survivable VN link protection is provisioned optimally by the Integer Linear Programming (ILP) formulation. Then, the VN is augmented with protection circuits by a number of links instead of the p-cycles to reduce the resource consumption. We also propose an efficient heuristic VNE to take advantage of the limited resources and reduce the embedding costs of the augmented VN. Finally, network coding is introduced to reduce the recovery delay and avoid data retransmission.

The main contributions of this paper can be summarized as follows.

(i) We propose an efficient approach, which is called protection circuit, to augment the VN at the VN level, which can recover the VN from the single substrate link failure. The basic idea is to connect all end-nodes in primary links and construct protection circuits for each VN. Compared with other schemes, it is more flexible to construct and provides more efficient protection.

(ii) We provide an ILP formulation for survivable VN link protection and propose a heuristic VNE to solve it, which includes a topology aware virtual node embedding based on weighted relative entropy (TAVNE-WRE) and an improved *k*-shortest path virtual link embedding (I-*k*-SPVLE). In TAVNE-WRE, the node topology degree and weighted relative entropy (WRE) method are both used to reduce the resource consumption in virtual node embedding. In I-*k*-SPVLE, the virtual link is embedded onto the substrate link with the shortest hop counts, and the problem of single substrate link protection is transformed into single virtual link protection in multiple protection circuits.

(iii) We introduce the network coding into the survivable VN link protection. With the help of network coding, the survivable VN link protection based on the protection circuit is developed from 1:N protection to 1 + N protection, which not only reduces the recovery delay but also recovers the data units without retransmitting them.

The rest of this paper is organized as follows. In section II, we present related work. In section III, we present the problem statement. The network model and evaluation indicators

are presented in section IV. In section V, we propose the ILP of the survivable VN link protection. The SVNLPM-NCPC is given, and its details are shown in section VI. In section VII, we evaluate the proposed algorithm through extensive simulations and experiments. We conclude this paper in section VIII.

II. RELATED WORK

Some heuristic survivable VNE algorithms under different objectives and constraints have been proposed recently [16], [17]. Recent works about survivable VNE were summarized from two aspects: survivable VNE against link failures and survivable VNE against node failures. In survivable VNE against link failures, link protection and node migration were widely used. Link protection was a proactive technique that reserved backup resources in anticipation of failures so that when a failure occurred, the backup resources were used to recover the link affected by the failure. The objective was to maximize the number of recovered virtual links across all affected VNs while minimizing the total resources that were required for recovery [18].

In addition, to solve the survivable VN link protection problem, a pro-active and hybrid heuristic policy was developed based on a fast re-routing strategy, and a prereserved quota was utilized for backups of each physical link [19]. To cope with the high network resource consumption in dedicated protection, many shared protection methods were proposed. Survivability in multi-path link embedding (SiMPLE) was proposed to provide guaranteed VN survivability against the single link failure while incurring minimal resource redundancy. In the case of multiple arbitrary link failures, SiMPLE provided maximal survivability to the VNs. In addition, a greedy proactive approach was proposed to solve larger instances of the problem in case of single link failures [20]. The problem of survivable VNE in a multi-domain optical network with the objective of minimizing the total network link costs was considered, which guaranteed the connectivity of virtual nodes after any single optical link failure [2]. A hierarchical software defined networking-based control plane was proposed to exchange information between domains, and heuristic approaches for mapping virtual links onto multi-domain optical links were proposed using partition and contraction mechanisms in the virtual topology.

To reduce resource consumption in link protection, the p-cycle was widely used in optical networks and overlay networks [21]. It was typically employed for 1:N protection and if a link failed, the signal could still be received on the other links. Two classes of periodic VN protection against link and node failures were proposed [22]. In the physical layer, a path or segment p-cycle technique and a column generation optimization model were used. In the VN layer, the topology with redundant resources was augmented, and subsequently, a column generation mapping model was applied. In addition, a p-cycle based protection scheme with cycle multiplexing and capacity balance was presented to protect the multicast services under a single link/node failure scenario [23]. The classical Prim algorithm was improved to generate optimized multicast light-trees, and the heuristic p-cycle generation algorithm was designed to connect the destination nodes of the multicast tree to a cycle to protect the entire tree. To solve the problem of jointly optimizing spare backup capacity allocation in a VN and embedding the VN to guarantee full bandwidth in the presence of the single substrate link failure, a quadratic integer program was formulated and transformed into an ILP. In VN layer protection, the p-cycle was used to protect the virtual links [24]. In these previous studies, although the resource consumption was reduced, the recovery delay caused by data rerouting, retransmission and the failure location was still long. Furthermore, the p-cycle was not always feasible, and it also needed much time to recover the faulty network.

To reduce the recovery delay in network protection, network coding was gradually used in some areas, such as overlay networks, the next generation synchronous optical network or the inter-datacenter network [25]–[27]. It offered benefits in terms of energy efficiency, additional security and reduced delay. In network coding, the intermediate nodes not only transmitted data units using network scheduling algorithms but also encoded/decoded them using primitive algebraic operations [28]. Recently, network coding also had been introduced to provide protection against link failures. This was achieved by transmitting combinations of data units from multiple links on a backup link in a manner that enabled each receiver node to recover a copy of the data units that were transmitted on the primary link in case the primary link failed. This could result in recovery from failures without data rerouting or retransmission, hence achieving agile protection. A coding-aware VN mapping framework was proposed that applied network coding to VN protection for the first time and proposed two network coding mechanisms with their corresponding link mapping algorithms [29]. However, it failed to design an efficient link protection scheme to reduce resource consumption.

III. PROBLEM STATEMENT

In this paper, we study the problem of designing an augmented VN and embedding it onto the SN by allocating the limited resources to recover from the single substrate link failure such that the total resource costs can be minimized.

To survive the single substrate link failure, the original VN has to be augmented to form a survivable VN with redundant virtual links. Protection circuit is an efficient approach for protecting working capacities in networks. It is a ring-like pre-configured structure constructed from the spare capacity available in the network. It has a short restoration time and a highly efficient capacity. In predesigned protection, the bandwidth on backup circuits is reserved in advance so that when a failure occurs, the backup paths that are reserved in advance are used to reroute the traffic lost due to the failure.

Additionally, to reduce recovery delay, network coding is introduced to recover the VN without data unit retransmission and failure detection. Network coding refers to performing

linear coding operations on the traffic carried by the network at intermediate network nodes. In this case, a node receives information from all or some of its input links, encodes this information, and sends the information to all or some of its output links.

Although the protection circuit and network coding have been used in some areas, there are some obvious characteristics in a VN such that these technologies cannot be directly introduced into the network virtualization environment without any changes.

(i) In network virtualization, the essence of VNE and the protection mechanism is resource allocation. Although resource allocation is already taken into consideration in the traditional survivable VN design, it is used in either survivable VNE or the protection mechanism's design. Research is limited to either of them alone, but they can be studied together. In this paper, we propose an efficient heuristic VNE and design a novel protection circuit to improve the resource utilization.

(ii) Some virtual links may be embedded onto the same substrate link, or their corresponding substrate links are not link-disjoint paths. If the joint substrate link fails, two or more virtual links in one protection circuit will also fail and the protection method will not work.

(iii) The substrate link is shared with multiple virtual links, and it is easier to construct the protection circuits compared with other traditional networks.

(iv) We select the single substrate link failure as the research object in this paper; the single substrate link failure is quite common in network virtualization. Furthermore, some virtual links are embedded onto the same substrate link, and the single substrate link protection can be transformed into the single virtual link protection in multiple protection circuits, which means that the single virtual link protection is contained in the single substrate link protection.

Therefore, some improvements and changes should be made to the network virtualization. We also make the following operational assumptions.

(i) All links are bidirectional, and the data units are fixed and equal in size.

(ii) When a substrate link fails, the receiving end of this link receives empty data units that contain all zeros.

(iii) The new data unit is transmitted by each end-node both on its primary link and protection circuit in each direction.

IV. NETWORK MODEL AND EVALUATION INDICATORS

A. NETWORK MODEL

1) SUBSTRATE NETWORK

The SN is modeled as a weighted undirected graph $\boldsymbol{G}_S = (N_S, E_S)$ in which the substrate node set and substrate link set are represented by N_S and E_S, respectively. In the substrate nodes, $cpu(n_s)$ is used to denote the available CPU resources of substrate node n_s, and $loc(n_s)$ is used to denote the location attribute of substrate node n_s. Similarly, $bw(e_s)$ is used to denote the available bandwidth resources of substrate link e_s. Each undirected substrate link corresponds to two transmission links that carry data units in two opposite directions.

2) VIRTUAL NETWORK

Similar to the SN, the VN can also be modeled as a weighted undirected graph $\boldsymbol{G}_V = (N_V, E_V)$. N_V represents the virtual node set, and E_V represents the virtual link set. $E_V = E_{Vp} \cup E_{Vb}$, $E_{Vp} \cap E_{Vb} = \varnothing$. E_{Vb} denotes the backup virtual link set, and E_{Vp} denotes the primary virtual link set. In virtual nodes, $cpu(n_v)$ denotes the required CPU resources of virtual node n_v. In virtual links, $bw(e_{vb})$ and $bw(e_{vp})$ are taken to denote the required bandwidth resources of the backup virtual link e_{vb} and the primary virtual link e_{vp}, respectively.

3) VIRTUAL NETWORK EMBEDDING

The VNE can be defined as an embedding function M from $\boldsymbol{G}_V$ to a subset of $\boldsymbol{G}_S$, which can be denoted by M: $\boldsymbol{G}_V \rightarrow (N'_S, E'_S)$, where $N'_S \subseteq N_S$ and $E'_S \subseteq E_S$. The general VNE consists of two stages: node embedding and link embedding.

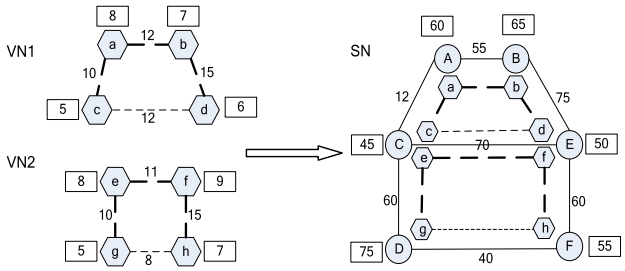


FIGURE 1. An example of VNE.

The example of VNE is shown in Fig. 1. In VN1 and VN2, the number beside each virtual node denotes the value of $cpu(n_v)$ and the number beside each virtual link denotes the value of $bw(e_{vb})$ or $bw(e_{vp})$. In the SN, the number beside each substrate node denotes the value of $cpu(n_s)$ and the number beside each link is the value of $bw(e_s)$. In VN1, virtual link (c,d) is a backup virtual link and others are primary virtual links. In VN2, virtual link (g,h) is a backup virtual link and others are primary virtual links.

In the process of virtual node embedding, the candidate substrate node n_s should have more resources than the resource request of virtual node n_v. Different virtual nodes in one VN cannot be embedded onto the same substrate node. In VN1, the virtual node embedding results are {a→A, b→B, c→C, d→E}. In VN2, the virtual node embedding results are {e→C, f→E, g→D, h→F}.

After virtual node embedding, we embed the virtual link between the embedded virtual nodes. In the process of virtual link embedding, the candidate substrate link should have more resources than the resource request of virtual link. In VN1, the virtual link embedding results are {(a,b)→(A,B), (a,c)→(A,C), (c,d)→(C,E), (b,d)→(B,E)}. In VN2, the virtual link embedding results are {(e,f)→(C,E), (e,g)→(C,D),

(f,h)→(E,F), (g,h)→(D,F)}. Among them, (c,d) and (e,f) are both embedded on the substrate link (C,E).

B. EVALUATION INDICATORS

1) ACCEPTANCE RATIO

The acceptance ratio is denoted by the number of VN requests that are embedded successfully divided by the total number of VN requests. It is as shown in (1).

$$r = \lim_{T\to\infty} \frac{\sum_{t=0}^{T} \left|VN_{\text{map}}(t)\right|}{\sum_{t=0}^{T} |VN(t)| + \delta} \tag{1}$$

Here, δ is infinitely close to 0. $|VN(t)|$ is the number of VN requests at time t, and $\left|VN_{\text{map}}(t)\right|$ is the number of VNs that are embedded successfully at time t.

2) LONG-TERM AVERAGE REVENUE TO COST RATIO

For VN request $\boldsymbol{G}_{\text{V}} = (N_{\text{V}}, E_{\text{V}})$,we denote the revenue $R(G_{\text{V}}, t)$ and cost $C(G_{\text{V}}, t)$ as follows:

$$R(\boldsymbol{G}_{\text{V}}, t) = \alpha \sum_{n_{\text{v}}\in N_{\text{V}}} cpu(n_{\text{v}}) + \sum_{e_{\text{v}}\in E_{\text{V}}} bw(e_{\text{v}}) \tag{2}$$

$$C(\boldsymbol{G}_{\text{V}}, t) = \beta \sum_{n_{\text{v}}\in N_{\text{V}}} cpu(n_{\text{v}}) + \sum_{e_{\text{v}}\in E_{\text{V}}} hops(e_{\text{v}})bw(e_{\text{v}}) \tag{3}$$

where α and β are weighting coefficients to balance the CPU and bandwidth resources, respectively. Without loss of generality, we assume $\alpha = \beta = 1$, which indicates that the importance of the CPU and bandwidth resource is similar [17,30]. Additionally, $hops(e_{\text{v}})$ is the hop count in the substrate link corresponding to the virtual link e_{v}.

The long-term average revenue to cost ratio can be defined as follows:

$$R/C = \lim_{T\to\infty} \frac{\sum_{t=0}^{T} \sum_{\boldsymbol{G}_{\text{V}}\subset VN_{\text{map}}(t)} \boldsymbol{R}\left(\boldsymbol{G}_{\text{V}}, t\right)}{\sum_{t=0}^{T} \sum_{\boldsymbol{G}_{\text{V}}\subset VN_{\text{map}}(t)} \text{C}\left(\boldsymbol{G}_{\text{V}}, t\right)} \tag{4}$$

Here, $VN_{\text{map}}(t)$ is the VNs that are embedded successfully at time t. In survivable VN link protection, if the failed link cannot be protected, the corresponding VN will fail. It also causes the penalty $PCost$, which can be defined as follows:

$$PCost(\boldsymbol{G}_{\text{V}}) = \mu \cdot R(\boldsymbol{G}_{\text{V}}, t) \tag{5}$$

Here, μ is the penalty coefficient. In this paper, $\mu = 3$. Therefore, the long-term average revenue to cost ratio can be redefined as follows:

$$R'/C = \lim_{T\to\infty} \frac{\sum_{t=0}^{T} \sum_{\boldsymbol{G}_{\text{V}}\subset VN_{\text{map}}(t)} (R(\boldsymbol{G}_{\text{V}}, t) - PCost(\boldsymbol{G}_{\text{V}}, t))}{\sum_{t=0}^{T} \sum_{\text{G}_{\text{V}}\subset VN_{\text{map}}(t)} C(\boldsymbol{G}_{\text{V}}, t)} \tag{6}$$

3) BACKUP BANDWIDTH RATIO

The backup bandwidth ratio is denoted as the utilization of the backup bandwidth resource. It is shown in (7).

$$BBR = \lim_{T\to\infty} \frac{\sum_{t=0}^{T} \sum_{\boldsymbol{G}_{\text{V}}\subset VN_{\text{map}}(t)} \sum_{e_{\text{vb}}\in E_{\text{Vb}}} hops(e_{\text{vb}})bw(e_{\text{vb}})}{\sum_{t=0}^{T} \sum_{\boldsymbol{G}_{V}\subset VN_{map}(t)} \sum_{e_{\text{vp}}\in E_{\text{Vp}}} hops(e_{\text{vp}})bw(e_{\text{vp}})} \tag{7}$$

Here, $hops(e_{\text{vb}})$ and $hops(e_{\text{vp}})$ are used to denote the hop counts of the backup virtual link e_{vb} and the primary virtual link e_{vp}, respectively.

4) AVERAGE NETWORK RECOVERY DELAY

The average network recovery delay is used to denote the recovery efficiency of faulty VNs. It is shown in (8).

$$a\_t = \lim_{T\to\infty} \frac{\sum_{t=0}^{T} r\_t(fl(t))}{\sum_{t=0}^{T} num(fl(t))} \tag{8}$$

Here, $r\_t(fl(t))$ is the network recovery delay of failure $fl(t)$. $num(fl(t))$ is the total failure number at time t.

5) AVERAGE FAILURE RECOVERY RATIO

The average failure recovery ratio is denoted by the number of failures that are recovered successfully divided by the total number of failures. It is shown in (9).

$$a\_r = \lim_{T\to\infty} \frac{\sum_{t=0}^{T} lf_{\text{suc}}(t)}{\sum_{t=0}^{T} num(fl(t))} \tag{9}$$

Here, $lf_{\text{suc}}(t)$ is the number of failures that are recovered successfully at time t.

V. ILP OF SURVIVABLE VN LINK PROTECTION

In this section, we formulate the ILP of survivable VN link protection. The objective function and constraints can be expressed as follows.

A. OBJECTIVE FUNCTION

$$\max\{\lim_{T\to\infty} \frac{\sum_{t=0}^{T} \sum_{\boldsymbol{G}_{\text{V}}\subset VN_{\text{map}}(t)} (R\left(\boldsymbol{G}_{\text{V}}, t\right) - PCost\left(\boldsymbol{G}_{\text{V}}\right))}{\sum_{t=0}^{T} \sum_{\boldsymbol{G}_{\text{V}}\subset VN_{\text{map}}(t)} C\left(\boldsymbol{G}_{\text{V}}, t\right)}\} \tag{10}$$

In this paper, our object is to get the maximum long-term average revenue to cost ratio.

B. CONSTRAINTS

$$\forall n_u \in N_V, \quad \forall n_i \in N_S$$
$$x(n_u, n_i) = \begin{cases} 1 & iff\ n_u \text{ is embedded onto } n_i \\ 0 & \text{otherwise} \end{cases} \tag{11}$$
$$\forall e^1_{um} \in E_{Vp}, \quad \forall e^2_{um} \in E_{Vb}, \forall e^3_{ij} \in E_S$$
$$x(e^1_{um}, e^3_{ij}) = \begin{cases} 1 & iff\ e^1_{um} \text{ is embedded onto } e^3_{ij} \\ 0 & \text{otherwise} \end{cases} \tag{12}$$
$$x(e^2_{um}, e^3_{ij}) = \begin{cases} 1 & iff\ e^2_{um} \text{ is embedded onto } e^3_{ij} \\ 0 & \text{otherwise} \end{cases} \tag{13}$$

In constraint (11), if virtual node n_u is embedded onto substrate node n_i, $x(n_u, n_i) = 1$. Otherwise, $x(n_u, n_i) = 0$. In constraint (12), if the primary virtual link e^1_{um} is embedded onto the substrate link e^3_{ij}, $x(e^1_{um}, e^3_{ij}) = 1$. Otherwise, $x(e^1_{um}, e^3_{ij}) = 0$. In constraint (13), if the backup virtual link e^2_{um} is embedded onto the substrate link e^3_{ij}, $x(e^2_{um}, e^3_{ij}) = 1$. Otherwise, $x(e^2_{um}, e^3_{ij}) = 0$.

$$\forall n_u \in N_V, \quad \forall n_i \in N_S$$
$$cpu(n_u) \times x(n_u, n_i) \le cpu(n_i) \tag{14}$$
$$\forall e^1_{um} \in E_{Vp}, \forall e^2_{um} \in E_{Vb}, \forall e^3_{ij} \in E_S$$
$$\sum_{e^1_{um} \in E_{Vp}} bw(e^1_{um}) \times x(e^1_{um}, e^3_{ij}) \le bw(e^3_{ij}) \tag{15}$$
$$\sum_{e^2_{um} \in E_{Vb}} bw(e^2_{um}) \times x(e^2_{um}, e^3_{ij}) \le bw(e^3_{ij}) - \sum_{e^1_{um} \in E_{Vp}} bw(e^1_{um}) \times x(e^1_{um}, e^3_{ij}) \tag{16}$$

Constraint (14) ensures that the candidate substrate node n_i should have more resources than the resource request of virtual node n_u. Constraint (15) ensures that the candidate substrate link e^3_{ij} should have more resources than the resource request of the primary virtual link e^1_{um}. Constraint (16) ensures that the sum of the resource requests of the backup virtual link e^2_{um} and the primary virtual link e^1_{um} should be smaller than the resources of the candidate substrate link e^3_{ij}.

$$x(n_u, n_i) \times dis(loc(n_i), loc(n_u)) \le D(n_i) \tag{17}$$

Constraint (17) is the location constraint and $dis(loc(n_i), loc(n_u))$ denotes the distance between the substrate node n_i and virtual node n_u.

$$\forall n_u \in N_V, \quad \forall n_i \in N_S, \forall e^1_{um} \in E_{Vp}, \forall e^3_{ij} \in E_S$$
$$\sum_{e^3_{ji} \in E_S} x(e^1_{um}, e^3_{ji}) - \sum_{e^3_{ij} \in E_S} x(e^1_{um}, e^3_{ij}) = \begin{cases} 1, & x(n_u, n_j) = 1 \\ -1, & x(n_m, n_j) = 1 \\ 0, & \text{otherwise} \end{cases} \tag{18}$$

Constraint (18) is the connectivity constraint. It refers to the flow conservation constraint for routing one unit of net flow from virtual node u to node m.

$$\forall n_u \in N_V, \sum_{n_i \in N_S} x(n_u, n_i) \le 1 \tag{19}$$
$$\forall e^1_{um} \in E_{Vp}, \forall e^2_{um} \in E_{Vb}, \forall e^3_{ij} \in E_S$$
$$\sum_{e^3_{ij} \in E_S} x(e^1_{um}, e^3_{ij}) \le 1 \tag{20}$$
$$\sum_{e^3_{ij} \in E_S} x(e^2_{um}, e^3_{ij}) \le 1 \tag{21}$$

Constraint (19) ensures that each virtual node is embedded on up to one substrate node. It means that if the virtual node n_u is embedded successfully, it must be embedded on up to one substrate node. Constraint (20) ensures that each primary virtual link is embedded on up to one substrate link. Constraint (21) ensures that each backup virtual link is embedded on up to one substrate link.

$$\forall e^2_{um} \in E_p, E_p \subset E_{Vb}, \quad \forall e^3_{ij} \in E_S$$
$$\sum_{e^2_{um} \in E_P} x(e^2_{um}, e^3_{ij}) \le 1 \tag{22}$$

Constraint (22) ensures that all backup virtual links in one protection circuit are embedded on different substrate links, in which E_p denotes the backup link set in one protection circuit.

$$\delta(e^1_{um}, e^2_{um}) = \begin{cases} 1 & iff\ e^1_{um} \text{ is protected by } e^2_{um} \\ 0 & \text{otherwise} \end{cases} \tag{23}$$
$$if\ \delta(e^1_{um}, e^2_{um}) = 1, \quad x(e^1_{um}, e^3_{ij}) + x(e^2_{um}, e^3_{ij}) \le 1 \tag{24}$$

In constraint (23), if the primary virtual link e^1_{um} is protected by the backup virtual link e^2_{um}, $\delta(e^1_{um}, e^2_{um}) = 1$. Otherwise, $\delta(e^1_{um}, e^2_{um}) = 0$. Constraint (24) ensures that the primary backup link e^1_{um} and its backup virtual link e^2_{um} cannot be embedded on the same substrate link.

$$\forall e^1_{um} \in E_{Vp}, \quad \forall e^1_{pq} \in E_{Vp}, \ \forall e^3_{ij} \in E_S$$
$$\eta(e^1_{um}, e^3_{ij}) = \begin{cases} 1 & iff \text{ the backup virtual link of } e^1_{um} \text{ uses } e^3_{ij} \\ 0 & \text{otherwise} \end{cases} \tag{25}$$
$$\xi(e^1_{um}, e^1_{pq}) = \begin{cases} 1 & iff\ e^1_{um} \text{ and } e^1_{pq} \text{ are not link disjoint paths} \\ 0 & \text{otherwise} \end{cases} \tag{26}$$
$$\eta(e^1_{um}, e^3_{ij}) + \eta(e^1_{pq}, e^3_{ij}) + \xi(e^1_{um}, e^1_{pq}) \le 2 \tag{27}$$

In constraint (25), if the backup virtual link of e^1_{um} uses the substrate link e^3_{ij}, $\eta(e^1_{um}, e^3_{ij}) = 1$. Otherwise, $\eta(e^1_{um}, e^3_{ij}) = 0$. In constraint (26), if the primary virtual links e^1_{um} and e^1_{pq} are not link disjoint paths, $\xi(e^1_{um}, e^1_{pq}) = 1$. Otherwise, $\xi(e^1_{um}, e^1_{pq}) = 0$. Constraint (27) indicates that only link-disjoint primary virtual links can share the same substrate link.

This ILP is known to be the NP-hard problem, and solving it is computationally intractable. Even though optimal

results can be obtained by some exact algorithms or the open source linear programming toolkit GLPK, they may incur exponentially increasing running times. Consequently, they cannot be scaled to address large VN embedding and protection problems. Most researchers solve the ILP problem of survivable VN link protection by proposing a corresponding heuristic algorithm that has a short computational time and gets an approximate optimal solution [17], [19], [20]. Therefore, we propose a novel heuristic algorithm called SVNLPM-NCPC to solve the ILP of the survivable VN link protection formulated in Section V.

VI. SVNLPM-NCPC

A. THE PROCESS OF SVNLPM-NCPC

As seen from Fig. 2, we augment the VN with the protection circuit when a new VN request arrives and embed the augmented VN request using TAVNE-WRE and I-*k*-SPVLE. If either the virtual node embedding or link embedding process fails, the original VN without augmentation is selected to embed. If the augmented VN is embedded successfully, network coding technology is introduced to protect the virtual links and transform the 1:N protection into 1 + N protection.

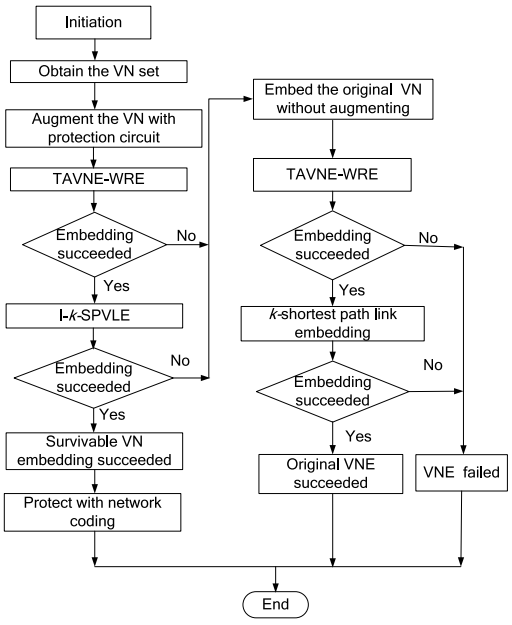


FIGURE 2. The process of SVNLPM-NCPC.

B. AUGMENTING VN WITH PROTECTION CIRCUIT

Augmenting the VN with the protection circuit is different from the p-cycle protection in traditional networks. In traditional networks, the backup links used for constructing the p-cycle must be real links in the networks. However, we can add some new virtual links into the original VN as backup links to construct the protection circuit. Then, we embed the augmented VN onto the SN. If VNE is successful, the added backup virtual links are embedded onto the real substrate links successfully and they can be used to transmit data units in a suitable manner. If VNE fails, we remove the added backup virtual links from the augmented VN and embed the original VN.

The details of augmenting the VN with protection circuits are shown in Algorithm 1.

Algorithm 1 Augmenting VN With Protection Circuits

Input: The protected virtual links
Output: The protection circuit set $\boldsymbol{P}$

1. Construct the set of end-nodes in all protected virtual links and denote it as $\boldsymbol{P}_{\text{node}}$
2. **for** $i = 1$:length($\boldsymbol{P}_{\text{node}}$)
3. **if** $\boldsymbol{P}_{\text{node}}$ is empty
4. Return VN_AUGMENTING_FINISHED
5. **break**
6. **else**
7. Construct a bidirectional protection circuit $\boldsymbol{P}_i$ that goes through the end-nodes in $\boldsymbol{P}_{\text{node}}$
8. Remove the nodes that have been gone through by $\boldsymbol{P}_i$ from $\boldsymbol{P}_{\text{node}}$
9. **end if**
10. **end for**

We set an example to illustrate this point shown in Fig. 3.

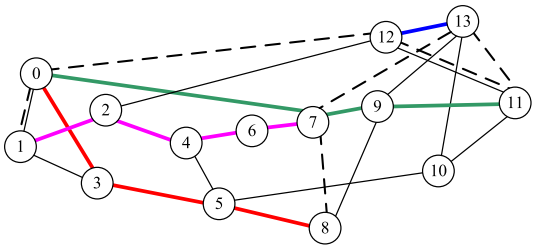


FIGURE 3. Example of augmenting the VN with a protection circuit.

We set the realistic network topology NSFNET as the virtual topology and augment it with a protection circuit, as shown in Fig. 3. There are four protected virtual links: (0,3,5,8), (1,2,4,6,7), (0,7,9,11) and (12,13). They are shown as colored lines. The protection circuit is (1,0,12,11,13,7,8), and it is shown as dashed lines. All end-nodes of these four virtual links are gone through by the protection circuit, and the backup resources are shared by all these four virtual links. Additionally, some virtual links cannot be protected by one protection circuit, and we can construct more protection circuits. However, if two virtual links in a protection circuit fail simultaneously, the protection circuit cannot recover them at the same time. Therefore, some actions should be taken in VNE to ensure that any two virtual links in the same protection circuit cannot be embedded onto the same substrate link.

C. TAVNE-WRE

In the survivable VN, the protection circuit is used to augment the VN and protect it against the single substrate link failure. If more substrate link resources are allocated to VNE, the link

resources used for protection are limited. It is important to reduce the bandwidth resource consumption in VNE.

In VNE algorithms, two-stage VNE is widely used, and the most important factors that affect VNE performance are the ranking indicators and ranking method. At first, the node topology degree is proposed to take the network topology into consideration. Then, a WRE method is introduced to rank the nodes that have multiple coefficients.

1) NODE RANKING INDICATORS

In the node embedding stage, the node ranking indicators are selected to rank the virtual nodes and substrate nodes. The node CPU resources and node adjacent link bandwidth resources are usually used as resource indicators [31]. In this paper, the node topology degree is introduced as the topological indicator into the virtual node and substrate node rankings with different definitions.

The node topology degree reflects the importance of the node from the topological point of view, as shown in (28):

$$CC(n_i) = \frac{1}{\sum\limits_{n_j \in \psi(n_i)} d_{ij}} \tag{28}$$

It represents the reciprocal of the total distance between node n_i and the other nodes in set $\Psi(n_i)$. d_{ij} is calculated as the distance between nodes n_i and n_j. In the SN, the substrate topology has a large number of nodes and links. It is difficult to calculate all distances between any two nodes. Therefore, $\Psi(n_i)$ is a set of substrate nodes corresponding to virtual nodes that are connected to node n_i and have been embedded. In the VN, the number of virtual nodes is less than the number of substrate nodes and $\Psi(n_i)$ is the set of all virtual nodes except node n_i.

2) WRE METHOD

WRE is a novel multiple factor ranking method that improves upon technique for order preference by similarity to an ideal solution. In the WRE method, it is easy to expand or add other node ranking indicators, and the weighting coefficients of ranking indicators can change in different environments. The relative entropy of systems A and B in states $\boldsymbol{A}_i$ and $\boldsymbol{B}_i$ $(i = 1, 2, \ldots, N)$ can be defined as follows:

$$C = \sum_{i=1}^{N} [\boldsymbol{A}_i \lg \frac{\boldsymbol{A}_i}{\boldsymbol{B}_i} + (1 - \boldsymbol{A}_i) \lg \frac{1 - \boldsymbol{A}_i}{1 - \boldsymbol{B}_i}] \tag{29}$$

In (29), $\boldsymbol{C}$ is the relative entropy of systems A and B. Suppose that there are N nodes in the VN, and each node has M evaluation indicators. The jth indicator coefficient of the ith node is denoted as $X_{ij}(i = 1, \ldots, N; j = 1, \ldots, M)$, and all coefficients of the network nodes constitute a decision matrix $\boldsymbol{X}$ which is denoted as follows:

$$\boldsymbol{X} = \begin{pmatrix} x_{11} & x_{12} & \cdots & x_{1M} \\ x_{21} & x_{22} & \cdots & x_{2M} \\ \vdots & \vdots & \ddots & \vdots \\ x_{N1} & x_{N2} & \cdots & x_{NM} \end{pmatrix} \tag{30}$$

To facilitate a fair comparison between different indicators, the coefficient is normalized. The normalized decision matrix is $\boldsymbol{R} = (r_{ij})_{N \times M}$. The weighting coefficient of the jth indicator is expressed as $\omega_j (j = 1, \ldots, M, \sum \omega_j = 1)$, and the weighted normalized decision matrix is denoted as follows:

$$\boldsymbol{Y} = \boldsymbol{X}\boldsymbol{\omega} = \begin{pmatrix} x_{11}\omega_1 & x_{12}\omega_2 & \cdots & x_{1M}\omega_M \\ x_{21}\omega_1 & x_{22}\omega_2 & \cdots & x_{2M}\omega_M \\ \vdots & \vdots & \ddots & \vdots \\ x_{N1}\omega_1 & x_{N2}\omega_2 & \cdots & x_{NM}\omega_M \end{pmatrix} \tag{31}$$

According to matrix $\boldsymbol{Y}$, the positive and the negative ideal solutions A^+ and A^- are determined as follows:

$$\boldsymbol{A}^+ = \{\max(y_{i1}, y_{i2}, \ldots, y_{iM})\} = \{y_1^{\max}, y_2^{\max}, \ldots, y_M^{\max}\} \tag{32}$$

$$\boldsymbol{A}^- = \{\min(y_{i1}, y_{i2}, \ldots, y_{iM})\} = \{y_1^{\min}, y_2^{\min}, \ldots, y_M^{\min}\} \tag{33}$$

The relative entropy of each solution to the positive and negative ideal solutions is calculated as follows:

$$D_i^+ = \sum_{\mathrm{j}=1}^{\mathrm{M}} \left[y_j^{\max} \lg \frac{y_j^{\max}}{y_{ij}} + (1 - y_j^{\max}) \lg \frac{1 - y_j^{\max}}{1 - y_{ij}} \right] \tag{34}$$

$$D_i^- = \sum_{\mathrm{j}=1}^{\mathrm{M}} \left[y_j^{\min} \lg \frac{y_j^{\min}}{y_{ij}} + (1 - y_j^{\min}) \lg \frac{1 - y_j^{\min}}{1 - y_{ij}} \right] \tag{35}$$

The similarity between each solution and the ideal one is calculated based on (36).

$$Z_i = \frac{D_i^-}{D_i^- + D_i^+}, \quad 0 \le Z_i \le 1 \tag{36}$$

In TAVNE-WRE, the node CPU resources, node adjacent link bandwidth resources and node topology degree are selected as the node ranking indicators. The values of these three node ranking indicators are input into WRE to calculate the virtual node importance $Z(n_v)$ and substrate node importance $Z(n_s)$.

3) THE PROCESS OF TAVNE-WRE

The main process of node embedding algorithm called TAVNE-WRE is summarized in Algorithm 2.

D. I-k-SPVLE

In the virtual link embedding stage, the I-k-SPVLE is proposed to embed the virtual links onto the substrate links that have the shortest hop counts and maximum bandwidth resource balance degree. It can also ensure that all virtual links in the same protection circuit will not be embedded onto the same substrate link.

The bandwidth resource balance degree is calculated as shown in (37):

$$BBD(p_k) = \frac{\min\limits_{e_s \in P_k} bw(e_s)}{\max\limits_{e_s \in P_k} bw(e_s)} \tag{37}$$

Algorithm 2 TAVNE-WRE

Input: G_S, G_V
Output: *NodeMappingList*

```
for each virtual node n_v ∈ N_V
    Calculate the virtual node importance Z(n_v)
end for
Rank virtual nodes based on Z(n_v) from large to small
Save virtual node embedding order into VirtualNodeList
for each virtual node in VirtualNodeList
    Select the candidate substrate node set Can(n_vi) and remove the embedded substrate nodes OpSubNode from Can(n_vi) Can(n_vi) = Can(n_vi) -OpSubNode
    if Can(n_vi) is empty
        Return NODE_EMBEDDING_FAILED
    else
        for each candidate node n_s in Can (n_vi)
            Calculate the substrate node importance Z(n_s)
        end for
        Embed n_v onto substrate node n_s with the largest Z(n_s) value and save it into NodeMappingList
        Update the embedded substrate nodes set OpSubNode
    end if
end for
Return NODE_EMBEDDING_SUCCEEDED
```

Algorithm 3 I-k-SPVLE

Input: G_S G_V *NodeMappingList*
Output: *LinkMappingList*

```
for each virtual link e_v ∈ E_V
    Search the k shortest paths between source node n_i and destination node n_j in SN, save them intoPathlist
    if Pathlist is empty
        Return LINK_EMBEDDING_FAILED
    else
        for each p_i ∈ Pathlist
            if the virtual link in the same protection circuit has embedded onto it
                Remove it from Pathlist
            end if
        end for
        Embed e_v to p_k with max BBD(p_k) and save them into LinkMappingList
    end if
end for
Return LINK_EMBEDDING_SUCCEEDED
```

where $\max_{e_s \in P_k} bw(e_s)$ and $\min_{e_s \in P_k} bw(e_s)$ are the maximum and minimum available bandwidth resources of the substrate links in p_k, respectively. p_k is the candidate substrate path in the virtual link embedding, and P_k is the candidate substrate path set. This indicator is used to balance the link resource consumption in the SN, which also reduces the possibility of resource fragmentation.

The details of I-k-SPVLE are given in Algorithm 3.

After embedding the virtual link using I-k-SPVLE, any two virtual links that are protected by the same protection circuit cannot be embedded onto the same substrate link. If a substrate link fails, all virtual links embedded onto it will be affected, and they can be recovered by their own protection circuit. Therefore, the single substrate link protection is transformed into the single virtual link protection in multiple protection circuits.

E. SURVIVABLE VN LINK PROTECTION BASED ON NETWORK CODING

Network coding means that the node receives information from some of its input links, encodes it, and then sends it to some of its output links. This approach can enhance the network capacity and offer protection without detecting the failures. In this paper, data units are transmitted both on the primary link and the protection circuit. The normal data units are transmitted on the primary link, and the coded data units are transmitted on the protection circuit.

The protection circuit is bidirectional, which corresponds to a clockwise half circuit T and a counter-clockwise half circuit S. In the protection circuit, the two ordered sets, $\boldsymbol{S} = (S_1, S_2, \ldots, S_N)$ and $\boldsymbol{T} = (T_1, T_2, \ldots, T_N)$, have equal lengths. If two nodes communicate, they must be in different ordered sets.

Nodes S_i and T_j belong to the two ordered sets $\boldsymbol{S}$ and $\boldsymbol{T}$, respectively. The data unit transmitted from S_i to T_j is denoted as u_i, and the data unit transmitted from T_j to S_i is denoted as d_j. The data unit can be zero in the case of a failure on the primary link between S_i and T_j. The data unit transmitted on the two unidirectional paths S and T is in the rounds that are started by nodes S_1 and T_1, respectively.

To facilitate the description of the network encoding executed by each node in $\boldsymbol{S}$ and $\boldsymbol{T}$, we first define the following symbols in Table 1.

For example, there are four primary virtual links (T_1, S_3), (T_2, S_4), (T_3, S_1) and (T_4, S_2) in Fig. 4. The protection circuit is $(T_2, T_3, T_4, S_4, S_3, S_2, S_1, T_1)$. All these primary virtual links and protection circuit are bidirectional.

In this paper, the network encoding uses **modulo two additions**, i.e., Exclusive-OR (XOR) operations. In Fig. 4, we can see that one protection circuit is used to protect all four virtual links together, and the data units are transmitted on both the primary links and the protection circuit. The encoding operations work as follows, and all data units belong to the same round.

The node has access to the data units generated by it and the data units received on the primary link and the protection circuit. Then, it encodes the data units and sends them on protection circuits T and S.

TABLE 1. Description of symbols.

Symbol	Description
$\gamma(S_i)$	the next node downstream from S_i on S
$\gamma^{-1}(S_i)$	the next node upstream from S_i on S
$\mu(S_i)$	the next node downstream from S_i on T
$\mu^{-1}(S_i)$	the next node upstream from S_i on T
$\gamma(T_j)$	the next node downstream from T_j on S
$\gamma^{-1}(T_j)$	the next node upstream from T_j on S
$\mu(T_j)$	the next node downstream from T_j on T
$\mu^{-1}(T_j)$	the next node upstream from T_j on T
$U_S(S_i)$	the set of upstream nodes of S_i on S
$U_T(T_j)$	the set of upstream nodes of T_j on T
$N(\boldsymbol{S})$	the subset of nodes that have a primary path connection to the nodes in $\boldsymbol{S}$

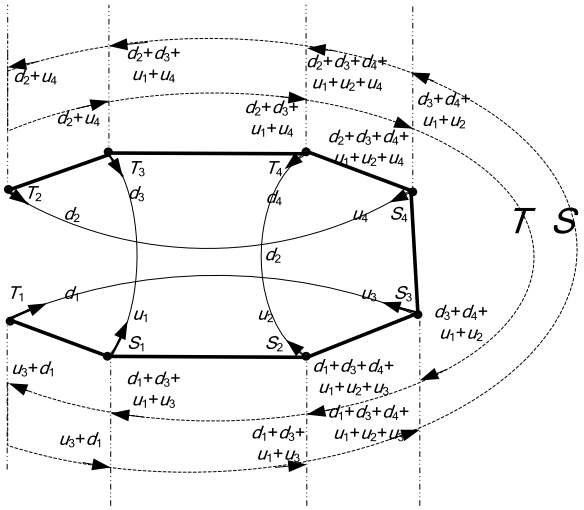


FIGURE 4. An example of survivable VN link protection based on network coding and protection circuit.

In circuit S, node S_i computes $g_{\gamma^{-1}(S_i)\to S_i} + (u_i + d_j)$ and sends the result to $\gamma(S_i)$.

$$g_{S_i\to\gamma(S_i)} = g_{\gamma^{-1}(S_i)\to S_i} + (u_i + d_j) \tag{38}$$

In circuit T, node S_i computes $k_{\mu^{-1}(S_i)\to S_i} + (u_i + d_j)$ and sends the result to $\mu(S_i)$.

$$k_{S_i\to\mu(S_i)} = k_{\mu^{-1}(S_i)\to S_i} + (u_i + d_j) \tag{39}$$

For example, on the protection circuit in Fig. 4, the following occurs.

i) Node T_2 has access to data unit d_2 (that it generated) and data unit u_4, which was received on the primary link from S_4. First it adds d_2 and u_4, and then it transmits $(d_2 + u_4)$ on the circuit T. Node T_3 receives $(d_2 + u_4)$ on circuit T, it also receives u_1 on the primary link from S_1. It adds its own data unit d_3 to the received data units $(d_2 + u_4)$ and u_1. Then, it transmits $(d_2 + d_3 + u_1 + u_4)$ on the circuit T. Node T_4 will repeat the same operation and transmit $(d_2 + d_3 + d_4 + u_1 + u_2 + u_4)$ on the protection circuit T.

ii) Along the same direction of the protection circuit, node S_4 receives $(d_2 + d_3 + d_4 + u_1 + u_2 + u_4)$ and d_2 from circuit T and node T_2, respectively. It adds them to u_4 (that it generated) and transmits $(d_3 + d_4 + u_1 + u_2)$ on circuit T. Node S_3 repeats the same operation and transmits $(d_1 + d_3 + d_4 + u_1 + u_2 + u_3)$ on circuit T. Node S_2 transmits $(d_1 + d_3 + u_1 + u_3)$ on circuit T. Node S_1 transmits $(d_1 + u_3)$ on circuit T. Node T_1 removes $(d_1 + u_3)$ from circuit T.

iii) The same operations can be repeated on the counter-clockwise circuit S, and the corresponding results can be obtained.

Data units not only transmit on the primary link, but also transmit as the second copy on the protection circuit to protect against the single link failure. If the primary link between nodes S_i and T_j fails, the data units transmitted on the primary link are zeros. Therefore, at node S_i, we only compute the data units that were received on the protection circuit and generated by itself. The data units received on the protection circuit include data unit $g_{\gamma^{-1}(S_i)\to S_i}$, which is received from the nodes upstream of S_i on circuit S, and data unit $k_{\mu^{-1}(S_i)\to S_i}$, which is received from the nodes upstream of S_i on circuit T.

$$g_{\gamma^{-1}(S_i)\to S_i} = \underbrace{\Big(\sum_{\{k:S_k\in U_S(S_i)\cap \boldsymbol{S}\}} u_k + \sum_{\{k:T_k\in N(U_S\{S_i\}\cap \boldsymbol{S})\}} d_k\Big)}_{\text{nodes upstream of } S_i \text{ on circuit } S \text{ in set } S} + \underbrace{\Big(\sum_{\{k:T_k\in U_S(S_i)\cap \boldsymbol{T}\}} d_k + \sum_{\{k:S_k\in N(U_S(S_i)\cap \boldsymbol{T})\}} u_k\Big)}_{\text{nodes upstream of } S_i \text{ on circuit S in set T}} \tag{40}$$

$$k_{\mu^{-1}(S_i)\to S_i} = \underbrace{\Big(\sum_{\{k:S_k\in U_T(S_i)\cap \boldsymbol{S}\}} u_k + \sum_{\{k:T_k\in N(U_T(S_i)\cap \boldsymbol{S})\}} d_k\Big)}_{\text{nodes upstream of } S_i \text{ on circuit } T \text{ in set } S} + \underbrace{\Big(\sum_{\{k:T_k\in U_T(S_i)\cap \boldsymbol{T}\}} d_k + \sum_{\{k:S_k\in N(U_T(S_i)\cap \boldsymbol{T})\}} u_k\Big)}_{\text{nodes upstream of } S_i \text{ on circuit T in set T}} \tag{41}$$

$$\begin{aligned} g_{\gamma^{-1}(S_i)\to S_i} + k_{\mu^{-1}(S_i)\to S_i} + u_i &= \underbrace{\Big(\sum_{\{k:S_k\in U_S(S_i)\cap \boldsymbol{S}\}} u_k + \sum_{\{k:T_k\in N(U_S\{S_i\}\cap \boldsymbol{S})\}} d_k\Big)}_{\text{nodes upstream of } S_i \text{ on circuit } S \text{ in set S}} \\ &+ \underbrace{\Big(\sum_{\{k:T_k\in U_S(S_i)\cap \boldsymbol{T}\}} d_k + \sum_{\{k:S_k\in N(U_S(S_i)\cap \boldsymbol{T})\}} u_k\Big)}_{\text{nodes upstream of } S_i \text{ on circuit S in set T}} \\ &+ \underbrace{\Big(\sum_{\{k:S_k\in U_T(S_i)\cap \boldsymbol{S}\}} u_k + \sum_{\{k:T_k\in N(U_T(S_i)\cap \boldsymbol{S})\}} d_k\Big)}_{\text{nodes upstream of } S_i \text{ on circuit T in set S}} \\ &+ \underbrace{\Big(\sum_{\{k:T_k\in U_T(S_i)\cap \boldsymbol{T}\}} d_k + \sum_{\{k:S_k\in N(U_T(S_i)\cap \boldsymbol{T})\}} u_k\Big)}_{\text{nodes upstream of } S_i \text{ on circuit T in set T}} + u_i \end{aligned}$$

$$= (\sum_{\{k:S_k \in T \setminus \{S_i\}\}} u_k + \sum_{\{k:T_k \in T\}} d_k) + (\sum_{\{k:S_k \in T\}} u_k + \sum_{\{k:T_k \in T \setminus \{T_j\}\}} d_k) + u_i$$
$$= u_i + d_j + u_i$$
$$= d_j \quad (42)$$

For example, we assume that the primary link between S_2 and T_4 fails. At node S_2, the node upstream of S_2 on circuit S in set $\boldsymbol{S}$ is $\{S_1\}$. The node upstream of S_2 on circuit S in set $\boldsymbol{T}$ is $\{T_1\}$. The node upstream of S_2 on circuit T in set $\boldsymbol{S}$ is $\{S_3, S_4\}$. The node upstream of S_2 on circuit T in set $\boldsymbol{T}$ is $\{T_2, T_3, T_4\}$.

$$g_{\gamma^{-1}(S_2)\to S_2} = \underbrace{(u_1 + d_3)}_{S_1} + \underbrace{(d_1 + u_3)}_{T_1}$$
$$= (u_1 + u_3) + (d_1 + d_3) \quad (43)$$
$$k_{\mu^{-1}(S_2)\to S_2} = \underbrace{((u_3 + u_4) + (d_1 + d_2))}_{S_3 \text{ and } S_4} + \underbrace{((d_2 + d_3 + d_4) + (u_4 + u_1 + u_2))}_{T_2,\ T_3 \text{ and } T_4}$$
$$= (u_1 + u_2 + u_3) + (d_1 + d_3 + d_4) \quad (44)$$
$$g_{\gamma^{-1}(S_2)\to S_2} + k_{\mu^{-1}(S_2)\to S_2} + u_2 = ((u_1 + u_3) + (d_1 + d_3)) + ((u_1 + u_2 + u_3) + (d_1 + d_3 + d_4)) + u_2$$
$$= d_4 \quad (45)$$

Therefore, the original data unit d_4 transmitted from T_4 to S_2 is recovered. Similarly, node T_4 can recover the original data unit u_2 in the same way.

F. COMPLEXITY ANALYSIS

The SVNLPM-NCPC includes augmenting the VN with protection circuit, TAVNE-WRE, I-k-SPVLE and network coding. In the augmented VN, it does not like the other protection cycles that are constructed in the SN and have to select the shortest link between nodes. Our protection circuit is constructed by adding some new virtual links that go through the end-nodes of the protected virtual links at the VN level without selecting the shortest link between nodes. Its complexity can be neglected. In VNE, the complexity of TAVNE-WRE is $O(|N_V| + |N_V|\,|N_S|^2)$, in which $|N_V|$ and $|N_S|$ are the total numbers of virtual nodes and substrate nodes, respectively. The complexity of I-k-SPVLE is $O(k\,|N_S|\,(|E_S| + |N_S|\lg|N_S|))$, and $|E_S|$ represents the total number of substrate links. In network coding, the complexity of XOR operations can also be neglected. Therefore, the total complexity of the SVNLPM-NCPC algorithm is $O(|N_V| + |N_V|\,|N_S|^2 + k\,|N_S|\,(|E_S| + |N_S|\lg|N_S|))$.

VII. SIMULATION

To validate the performance of the SVNLPM-NCPC proposed in this paper, three comparative experiments are established in this section. The performance of the SVNLPM-NCPC algorithm is compared with four algorithms in the first simulation experiment. Next, we simulate the impact of the backup mechanisms on the SVNLPM-NCPC. Finally, we evaluate the influence of the VNE mechanisms on the SVNLPM-NCPC.

A. SIMULATION ENVIRONMENT

In this paper, the SN topology and VN topology are generated by the improved Salam network topology random generation algorithm [32]. The SN is composed by 100 nodes and 500 links. The positions of the substrate nodes follow a uniform distribution in the scope of 1000×1000 distance units. The initial available CPU resources of the substrate nodes and the bandwidth resources of the substrate links are real numbers following a uniform distribution between 50 and 100.

The arrivals of VN requests follow a Poisson process with an average arrival rate of 5 per 100 time units, and the lifetime of each VN request is modeled by an exponential distribution with an average of 1000 time units. For each VN request, the number of nodes is uniformly distributed between 4 and 8. The link connectivity rate is 0.5. The required CPU and bandwidth resources are real numbers that are uniformly distributed between 0 and 5. In addition, all virtual nodes have a constant position constraint value of $D = 500$. The weighting coefficient of the node CPU resources ω_1 is 0.1, the weighting coefficient of the node adjacent link bandwidth resources ω_2 is 0.2 and the weighting coefficient of the node topology degree ω_3 is 0.7. We randomly select 3 links in each VN as primary links that need to be protected and save them into set I_{VL}. Then, we save the substrate links embedded by the virtual links in I_{VL} into set I_{SL}. We randomly select $\lceil \lambda \cdot length(I_{SL}) \rceil$ substrate links from I_{SL} and save them into the substrate link failures set SL_F, where λ is the failure rate and $\lambda = 0.1$. Additionally, the failed substrate links in SL_F must be operational before failing.

The computer used for the simulation experiments is a Lenovo Tianyi 510Pro with the Windows 10 operating system. The hardware platform is composed of an Intel Core i7-7700 3.6 GHz processor with 8GB of RAM. The analysis software is Matlab R2007a. We run our simulations for 50000 time units so that the performance is in a stable state. In all test cases, the results are averaged over 15 runs, and we show the margin of error with a 95% confidence level [11].

B. COMPARISONS OF DIFFERENT ALGORITHMS

In this paper, our simulations focus on the five algorithms listed in Table 2, and all these algorithms use the same SN and VN requests.

Fig. 5 illustrates the acceptance ratios of the five algorithms in the stable state. We can see that the acceptance ratio of SVNLPM-NCPC is close to 0.69, and it is the highest in all algorithms, which exceeds the 1 + 1 protection by 45.8%. In SVNLPM-NCPC, the protection circuit is introduced to protect the virtual links and reduce resource consumption. Additionally, in the process of VNE, the WRE method and

TABLE 2. Comparison of algorithms.

Algorithm	Description
SVNLPM-NCPC	Survivable VN link protection method based on network coding and protection circuit proposed in this paper. The node embedding algorithm is TAVNE-WRE and the link embedding algorithm is I-*k*-SPVLE.
1+1 protection	1+1 VN link protection algorithm proposed in [19]. The node embedding algorithm is D-ViNE and the link embedding algorithm is the *k*-shortest path algorithm.
SBPP	Shared on-demand VN link protection algorithm proposed in [17]. The node embedding algorithm is the Greedy algorithm and the link embedding algorithm is the *k*-shortest path algorithm.
SiMPLE-PR	Shared backup VN link protection proposed in [20]. The node embedding algorithm is random node mapping and the link embedding algorithm is multi-path link embedding.
G-Coding	Survivable VN embedding algorithm with the general coding mechanism proposed in [29]. The node embedding algorithm is Breadth-First Searching (BFS) and the link embedding algorithm is the path splitting algorithm.

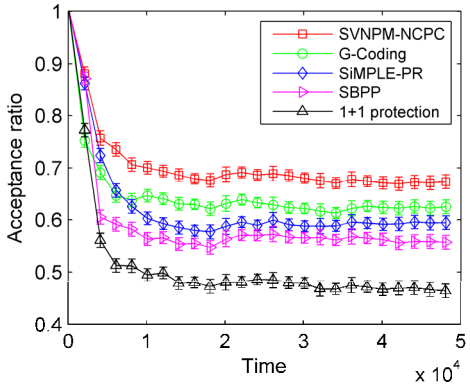


FIGURE 5. Comparison of the acceptance ratios.

topological indicators are both used to shorten the hop counts of the substrate links and reduce the link resource consumption in TAVNE-WRE. In I-*k*-SPVLE, the bandwidth resource balance degree is used to balance the link resource consumption in the SN, which also reduces the possibility of network fragmentation. Therefore, more virtual links can be protected by sparse backup resources. G-coding uses the BFS algorithm to embed the virtual node and the path splitting algorithm to embed the virtual links. Its acceptance ratio exceeds the 1+1 protection by 39.6%. With the help of BFS, virtual nodes and links are embedded at the same stage, which is similar to one-stage VNE and saves link bandwidth resources. In path splitting, one virtual link can be embedded onto several substrate links, which is quite different from the normal virtual link embedding method. It makes full use of the limited substrate link resources, but this technology is difficult to implement in real network operations. SiMPLE-PR is a proactive survivable VN link protection algorithm, and by exploiting path diversity in the physical network, SiMPLE provides guaranteed VN survivability against the single link failure while incurring minimal resource redundancy. It achieves approximately 5.17% and 27.1% higher acceptance ratios than SBPP and 1 + 1 protection, respectively. SBPP is a typical shared protection method, and its acceptance ratio exceeds the 1 + 1 protection by 20.1%.

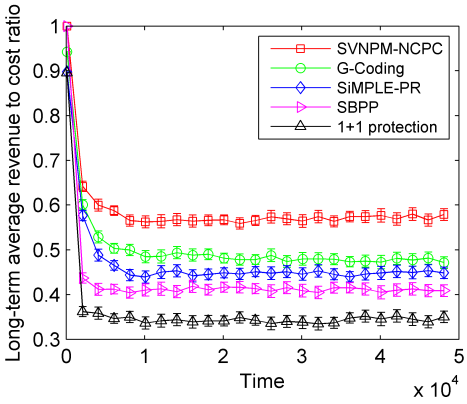


FIGURE 6. Comparison of long-term average revenue to cost ratios.

Fig. 6 illustrates the long-term average revenue to cost ratios of the five algorithms in the stable state. The 1 + 1 protection has the lowest acceptance ratio, and the long-term average revenue to cost ratio is obviously lower than those of the other four algorithms. The backup resources in SBPP are shared with many virtual links, more resources can be used to embed or protect VNs, and its long-term average revenue to cost ratio exceeds the 1 + 1 protection by 14.3%. SiMPLE-PR takes the multi-path link embedding into consideration, which makes full use of the bandwidth resources and reduces resource consumption. It achieves approximately 12.5% higher long-term average revenue to cost ratio than SBPP. G-coding not only uses BFS to shorten the hop counts between the virtual nodes, which reduce the embedding costs, but it also introduces path splitting to improve the acceptance ratio. Its long-term average revenue to cost ratio exceeds SiMPLE-PR by 11.1%. The SVNLPM-NCPC proposed in this paper takes the topological indicators into consideration and uses the WRE method to increase the weighting coefficients of them, which reduces the VNE costs. The I-*k*-SPVLE is used to reduce the possibility of network fragmentation, and it improves the revenues. Additionally, in the link protection process, the protection circuit is more efficient than the traditional shared protection method, which is used in SBPP, SiMPLE-PR and G-coding. Its revenue to cost ratio is the highest and is close to 0.59.

Fig. 7 illustrates the backup bandwidth ratios of five algorithms in the stable state. As seen, the backup bandwidth ratio of 1 + 1 protection is the highest, which means that its bandwidth resource utilization is the lowest. SBPP can share the backup resources with other virtual links, which reduces the backup bandwidth ratio, and it achieves approximately a 40% lower backup bandwidth ratio than 1 + 1 protection. SiMPLE-PR embeds one virtual link into several substrate

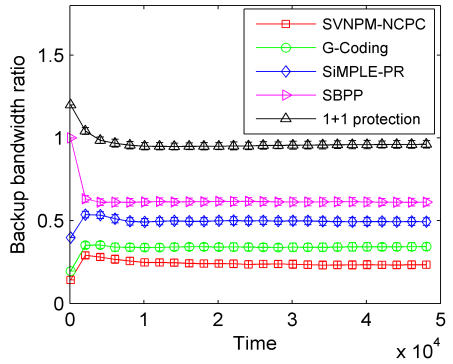


FIGURE 7. Comparison of the backup bandwidth ratios.

links, which makes full use of the bandwidth resources and reduces the backup bandwidth ratio. In the G-coding mechanism, the BFS algorithm and path splitting algorithm are used in VNE to shorten the hop counts between substrate nodes and reduce the bandwidth consumption. It achieves approximately 36.7% and 24% lower backup bandwidth ratios than SBPP and SiMPLE-PR, respectively. Our SVNLPM-NCPC has a similar protection performance to 1 + N protection and makes full use of the limited resources by using TA-VNE and I-*k*-SPVLE. Its backup bandwidth ratio is the lowest.

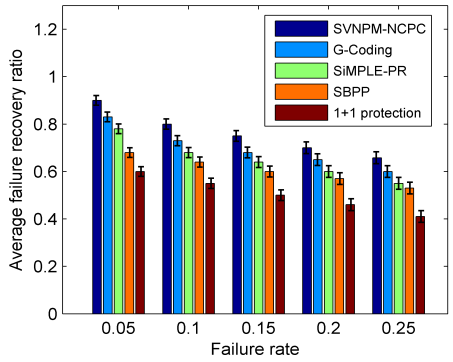


FIGURE 8. Comparison of the average failure recovery ratios.

Fig. 8 illustrates the average failure recovery ratios of the five algorithms in the stable state. All average failure recovery ratios gradually decrease with the increase of the failure rate. Among them, SVNLPM-NCPC achieves approximately 11.1%, 16.9%, 32.4% and 45.2% higher average failure recovery ratios than G-Coding, SiMPLE-PR, SBPP and 1 + 1 protection, respectively. The average failure recovery ratios of G-Coding and SiMPLE-PR are both lower than SVNLPM-NCPC, but they are still higher than SBPP and 1 + 1 protection. The average failure recovery ratio of 1 + 1 protection is the lowest.

Fig. 9 illustrates the average network recovery delays of the five algorithms in the stable state. It is easy to find that the average network recovery delay has nothing to do with the failure rate. The average network recovery delays of SVNLPM-NCPC and G-Coding are nearly equal, which

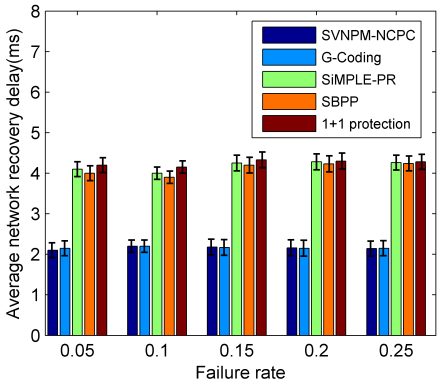


FIGURE 9. Comparison of the average network recovery delays.

are close to 4.1 ms. The average network recovery delays of SiMPLE-PR, SBPP and 1 + 1 protection are close to 2.05 ms. SiMPLE-PR, SBPP and 1+1 protection take approximately 50% more time than SVNLPM-NCPC and G-Coding to recover from failures due to the long switch reconfiguration and traffic rerouting, which are not required in network coding protection. The SVNLPM-NCPC introduces network coding to protect the virtual links against the single substrate link failure, which transforms the 1:N protection into 1 + N protection and shortens the average network recovery delay.

TABLE 3. Average execution time.

Algorithm	Average Execution Time (ms)
SVNLPM-NCPC	70
1+1 protection	1656
SBPP	49
SiMPLE-PR	423
G-Coding	463

Table 3 illustrates the average execution time of the five algorithms in the stable state. On average, to embed each VN request and offer link protection across experiments, SVNLPM-NCPC takes 70 ms. 1+1 protection takes 1656 ms. SBPP, SiMPLE-PR and G-Coding takes 49 ms, 423 ms and 463 ms, respectively. Among them, 1 + 1 protection uses the D-ViNE algorithm to embed the virtual node. It takes more time than other heuristic virtual node embedding algorithms. SiMPLE-PR and G-Coding use multi-path link embedding and the path splitting algorithm to embed the virtual links, respectively. They increase the complexity of the algorithm. SVNLPM-NCPC conducts network coding and node ranking based on WRE. Its average execution time is longer than SBPP but shorter than other algorithms.

C. COMPARISONS OF DIFFERENT BACKUP MECHANISMS

In this section, we compare the performance of different backup mechanisms. To make a fair comparison, two baseline algorithms called SBPP-1 and 1 + 1 protection-1 are proposed. The SBPP-1 is developed from the traditional SBPP,

and we select the shared pre-allocation as the backup mechanism that allocates backup resources for substrate links during the network's pre-configuration phase. The 1 + 1 protection-1 selects 1 + 1 protection as the backup mechanism. Both of them use the same VNE algorithm, which is proposed in SVNLPM-NCPC.

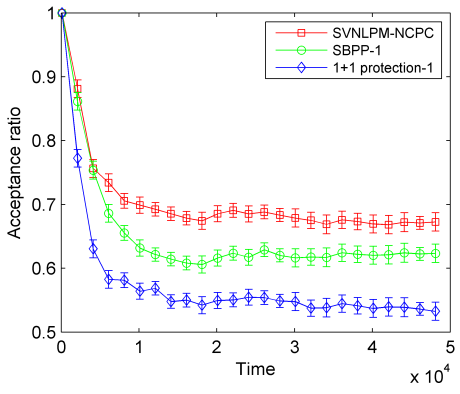


FIGURE 10. Comparison of the acceptance ratios.

As seen from Fig. 10, the SVNLPM-NCPC with the protection circuit at the VN level has the highest acceptance ratio. It reduces the resource consumption in VN protection, and more VNs are embedded successfully. In SBPP-1, the backup resources are allocated to substrate links at the SN level. Although some virtual links can share the backup resources, the backup substrate link resources in the substrate link that are not embedded by the virtual link are wasted. Its acceptance ratio is lower than SVNLPM-NCPC, but it is still 22.9% higher than 1 + 1 protection-1, which allocates backup resources for dedicated virtual links.

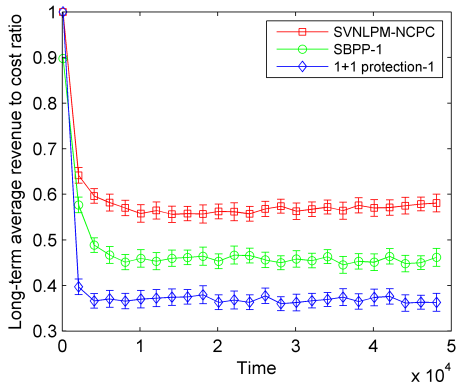


FIGURE 11. Comparison of the long-term average revenue to cost ratios.

Fig. 11 illustrates the long-term average revenue to cost ratios of the three algorithms in the stable state. In 1 + 1 protection-1, the backup resource is used to protect the dedicated virtual link. Its resource utilization is the lowest, and its long-term average revenue to cost ratio is lower than the other two algorithms. In the SBPP-1 algorithm, the backup resources can be shared with some virtual links, and its long-term average revenue to cost ratio is 23.6% higher than 1+1 protection-1. In SVNLPM-NCPC, the protection circuit is developed from the p-cycle, and the backup resources are allocated at the VN level, which makes full use of the limited bandwidth resources. Its long-term average revenue to cost ratio is the highest.

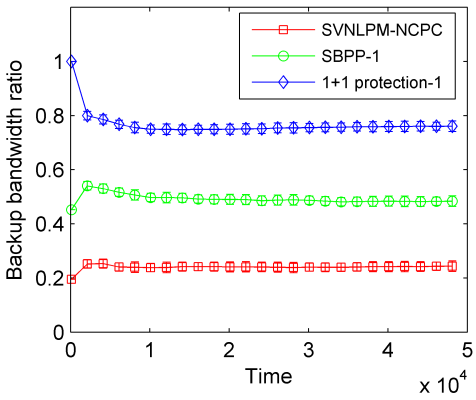


FIGURE 12. Comparison of the backup bandwidth ratios.

Fig. 12 illustrates the backup bandwidth ratios of the three algorithms in the stable state. The SVNLPM-NCPC with the protection circuit at the VN level has the lowest backup bandwidth ratio, which is followed by SBPP-1 and 1 + 1 protection-1. Therefore, the SVNLPM-NCPC has the best backup resource utilization.

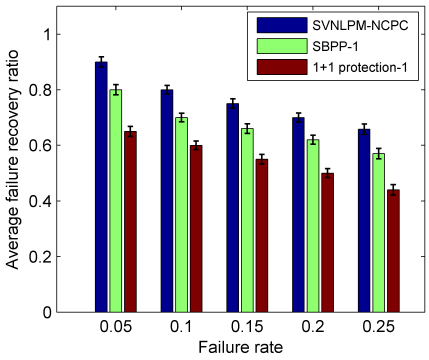


FIGURE 13. Comparison of the average failure recovery ratios.

Fig. 13 illustrates the average failure recovery ratios of the three algorithms in the stable state. SVNLPM-NCPC has the highest average failure recovery ratio because it reduces the resource consumption in virtual link protection with the help of the protection circuit, and more link resources are used to protect the virtual links. SBPP-1 shares backup resources with other virtual links at the SN level, and its average failure recovery ratio is 21.2% higher than that of 1+1 protection-1.

Fig. 14 illustrates the average network recovery delays of the three algorithms in the stable state. The average network recovery delay of SVNLPM-NCPC is shorter than that of SBPP-1 and 1+1 protection-1. In SVNLPM-NCPC, network

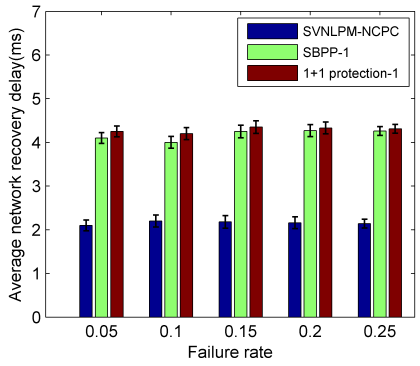


FIGURE 14. Comparison of the average network recovery delays.

coding is introduced into the survivable VN link protection, which avoids the long switch reconfiguration and traffic rerouting. Therefore, the average network recovery delay of SVNLPM-NCPC is the shortest.

D. COMPARISONS OF DIFFERENT VNE MECHANISMS

In this section, we compare the performance of the different VNE mechanisms. For a fair comparison, four baseline algorithms called SVNLPM-NCPC1, SVNLPM-NCPC2, SVNLPM-BFS and SVNLPM-Greedy are proposed. By comparing SVNLPM-NCPC, SVNLPM-NCPC1 and SVNLPM-NCPC2, the effect of the weighting coefficients on the performance of the algorithm is discussed. By comparing SVNLPM-NCPC, SVNLPM-BFS and SVNLPM-Greedy, the effect of the VNE mechanism on the performance of the algorithm is discussed.

In SVNLPM-NCPC1, the TAVNE-WRE is used, in which $\omega_1 = 0.2$, $\omega_2 = 0.15$ and $\omega_3 = 0.65$. In SVNLPM-NCPC2, the TAVNE-WRE is used, in which $\omega_1 = 0.3$, $\omega_2 = 0.1$ and $\omega_3 = 0.6$. The SVNLPM-BFS is developed from SVNLPM-NCPC and the VNE algorithm in G-coding is introduced as the VNE algorithm of SVNLPM-BFS. The SVNLPM-Greedy is developed from SVNLPM-NCPC and the VNE algorithm in SBPP is introduced as the VNE algorithm of SVNLPM-Greedy.

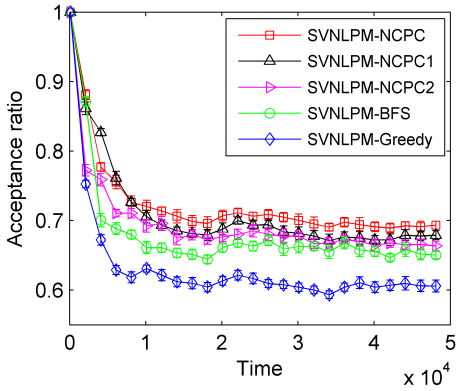


FIGURE 15. Comparison of the acceptance ratios.

Fig. 15 illustrates the acceptance ratios of the five algorithms in the stable state. The SVNLPM-NCPC achieves a 1.7%, 4.1%, 6.1% and 14.4% higher acceptance ratio than SVNLPM-NCPC1, SVNLPM-NCPC2, SVNLPM-BFS and SVNLPM-Greedy, respectively. In SVNLPM-NCPC, the WRE method is introduced to change the weighting coefficients. Compared with SVNLPM-NCPC1 and SVNLPM-NCPC2, the weighting coefficient of the node CPU resources is decreased, and the weighting coefficients of the topological indicators are increased in SVNLPM-NCPC. In this simulation environment, the bandwidth resource request is larger than the node CPU resources, and most failed VNs in embedding are caused by a lack of bandwidth resources. Therefore, increasing the weighting coefficients of the topological indicators can shorten the hop counts of the substrate links and reduce the link resource consumption of VNE. Additionally, the possibility of network fragmentation is reduced with the help of the bandwidth resource balance degree in I-k-SPVLE. SVNLPM-BFS introduces the BFS, which shortens the hop counts of the substrate links. Its performance is better than the SVNLPM-Greedy algorithm.

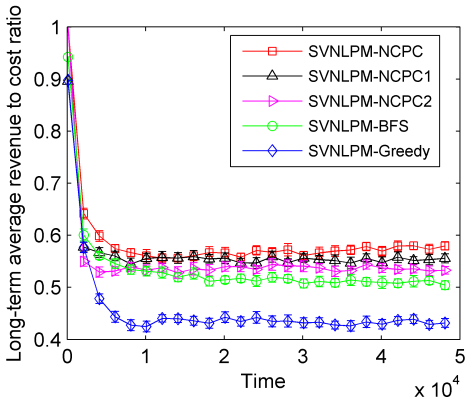


FIGURE 16. Comparison of the long-term average revenue to cost ratios.

Fig. 16 illustrates the long-term average revenue to cost ratios of the five algorithms in the stable state. The SVNLPM-NCPC achieves a 3.9%, 6.2%, 9.33% and 28.1% higher long-term average revenue to cost ratio than SVNLPM-NCPC1, SVNLPM-NCPC2, SVNLPM-BFS and SVNLPM-Greedy, respectively.

In summary, our SVNLPM-NCPC performs significantly better than other survivable VN link protection methods in terms of the acceptance ratio, the long-term average revenue to cost ratio, the backup bandwidth ratio, the average failure recovery ratio and the execution time. It can make full use of the limited resources in both the VN augmenting with the protection circuit and VNE. Additionally, the introduction of network coding shortens the average failure recovery delay. Therefore, SVNLPM-NCPC is suitable in survivable VN link protection against the single substrate link failure.

VIII. CONCLUSION

In this paper, we propose the SVNLPM-NCPC to protect the VN against the single substrate link failure. We provide an ILP formulation and augment the VN with the

protection circuit to reduce the backup resource consumption. Then, we design an efficient heuristic VNE algorithm that includes TAVNE-WRE and I-*k*-SPVLE to solve the ILP. In TAVNE-WRE, the WRE is introduced to increase the weighting coefficients of the topological indicators. In I-*k*-SPVLE, we reduce the embedding costs by introducing the bandwidth resource balance degree and transform the single substrate link protection into the single virtual link protection in multiple protection circuits. Additionally, network coding is introduced to reduce the average network recovery delay. Finally, three experiments are designed in the simulation and evaluation stages to demonstrate the performance of SVNLPM-NCPC. The first experiment verifies that, compared with other typical survivable VN link protection algorithms, the proposed SVNLPM-NCPC algorithm not only has the best acceptance ratio and long-term average revenue to cost ratio but also greatly enhances the achievable backup sharing. It still shortens the recovery time and execution time. The second experiment assesses the effect of the backup mechanism on the performance of SVNLPM-NCPC. The protection circuit constructed at the VN level has better performance than the other two backup mechanisms. The third experiment evaluates the influence of the VNE algorithm on SVNLPM-NCPC. The VNE proposed in this paper, which includes TAVNE-WRE and I-*k*-SPVLE, has excellent performance compared with other VNE algorithms. The next step is to study the survivable VN link protection against multiple substrate link failures in the future.

REFERENCES

[1] J. Sun *et al.*, "A reliability-aware approach for resource efficient virtual network function deployment," *IEEE Access*, vol. 6, pp. 18238–18250, 2018.

[2] S. Hong, J. P. Jue, P. Park, H. Yoon, H. Ryu, and S. Hong, "Survivable virtual topology design in multi-domain optical networks," *IEEE/OSA J. Opt. Commun. Netw.*, vol. 8, no. 6, pp. 408–416, Jun. 2016.

[3] R. Mijumbi, J. Serrat, J.-L. Gorricho, N. Bouten, F. De Turck, and R. Boutaba, "Network function virtualization: State-of-the-art and research challenges," *IEEE Commun. Surveys Tuts.*, vol. 18, no. 1, pp. 236–262, 1st Quart., 2016.

[4] N. Shahriar *et al.*, "Virtual network survivability through joint spare capacity allocation and embedding," *IEEE J. Sel. Area Commun.*, vol. 36, no. 3, pp. 502–518, Mar. 2018.

[5] A. Markopoulou, G. Iannaccone, S. Bhattacharyya, C.-N. Chuah, Y. Ganjali, and C. Diot, "Characterization of failures in an operational IP backbone network," *IEEE ACM Trans. Netw.*, vol. 16, no. 4, pp. 749–762, Aug. 2008.

[6] A. Hmaity, F. Musumeci, and M. Tornatore, "Survivable virtual network mapping to provide content connectivity against double-link failures," in *Proc. DRCN*, Paris, France, 2016, pp. 160–166.

[7] R. Li, Q. Wu, Y. Tan, and J. Zhang, "On the optimal approach of survivable virtual network embedding in virtualized SDN," *IEICE Trans. Inf. Syst.*, vol. E101.D, no. 3, pp. 698–708, Mar. 2018.

[8] J. Sun, Y. Zhang, D. Liao, G. Sun, and V. Chang, "AI-based survivable design for hybrid virtual networks for single regional failures in cloud data centers," *Cluster Comput.*, vol. 4, no. 5, pp. 1–11, May 2018.

[9] A. Khan, X. An, and S. Iwashina, "Virtual network embedding for telco-grade network protection and service availability," *Comput. Commun.*, vol. 84, no. 6, pp. 25–38, Jun. 2016.

[10] A. Aguado *et al.*, "Dynamic virtual network reconfiguration over SDN orchestrated multitechnology optical transport domains," *J. Lightw. Technol.*, vol. 34, no. 8, pp. 1933–1938, Apr. 15, 2016.

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

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

[13] A. E. Kamal and O. Al-Kofahi , "Efficient and agile 1+N protection," *IEEE Trans. Commun.*, vol. 59, no. 1, pp. 169–180, Jan. 2011.

[14] B. Li *et al.*, "P-cycle based protection scheme with cycle multiplexing and capacity balance for multicast service in substation communication network," *Int. J. Elect. Power Energy Syst.*, vol. 102, no. 5, pp. 340–348, Nov. 2018.

[15] A. E. Kamal, A. Ramamoorthy, L. Long, and S. Li, "Overlay protection against link failures using network coding," *IEEE/ACM Trans. Netw.*, vol. 19, no. 4, pp. 1071–1084, Aug. 2011.

[16] Y. Wang, X. Li, B. Guo, T. Gao, W. Li, and S. Huang, "Survivable virtual optical network mapping in elastic optical networks with shared backup path protection," in *Proc. WOCC*, Chengdu, China, May 2016, pp. 1–4.

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

[18] H. Sandra, K. Ashiq, and X. An, "Survey on survivable virtual network embedding problem and solutions," in *Proc. ICNS*, Lisbon, Portugal, 2013, pp. 94–104.

[19] M. R. 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, Jun. 2013.

[20] M. M. A. Khan, N. Shahriar, R. Ahmed, and R. Boutaba, "Multi-path link embedding for survivability in virtual networks," *IEEE Trans. Netw. Service Manag.*, vol. 13, no. 2, pp. 253–266, Jun. 2016.

[21] X. Guo, J. Huang, H. Liu, and Y. Chen, "Efficient P-cycle combination protection strategy based on improved genetic algorithm in elastic optical networks," *IET Optoelectron.*, vol. 12, no. 2, pp. 73–79, Feb. 2018.

[22] A. Jarray and A. Karmouch, "Cost-efficient mapping for fault-tolerant virtual networks," *IEEE Trans. Comput.*, vol. 64, no. 3, pp. 668–681, Mar. 2015.

[23] X. Chen, M. Zhou, S. Zhu, S. Kang, L. Sun, and Z. Zhu, "Optimizing FIPP-*p*-cycle protection design to realize availability-aware elastic optical networks," *IEEE Commun. Lett.*, vol. 22, no. 1, pp. 65–68, Jan. 2018.

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

[25] S. Kafaie, M. H. Ahmed, Y. Chen, and O. A. Dobre, "Performance analysis of network coding with IEEE 802.11 DCF in multi-hop wireless networks," *IEEE Trans. Mobile Comput.*, vol. 17, no. 5, pp. 1148–1161, May 2018.

[26] N. B. El Asghar, I. Jouili, and M. Frikha, "Survivable inter-datacenter network design based on network coding," in *Proc. AICCSA*, Hammamet, Tunisia, 2017, pp. 1192–1197.

[27] T. H. Dao, "On optimal designs of transparent WDM networks with 1+1 protection leveraged by all-optical XOR network coding schemes," *Opt. Fiber Technol.*, vol. 40, no. 1, pp. 93–100, Jan. 2018.

[28] T. H. Dao, "An optimal design framework for 1+1 routing and network coding assignment problem in WDM optical networks," *IEEE Access*, vol. 5, pp. 22291–22298, 2017.

[29] Z. Wang, J. Wu, and D. Cheng, "Coding-aware virtual network mapping for surviving single link failure," in *Proc. IEEE ICC*, Sydney, NSW, Australia, Jun. 2014, pp. 3025–3030.

[30] Q. Hu, Y. Wang, and X. Cao, "Survivable network virtualization for single facility node failure: A network flow perspective," *Opt. Switching Netw.*, vol. 10, no. 4, pp. 406–415, 2013.

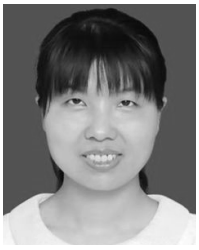
[31] S. Gong, J. Chen, S. Zhao, and Q. Zhu, "Virtual network embedding with multi-attribute node ranking based on TOPSIS," *KSII Trans. Internet Inf.*, vol. 10, no. 2, pp. 522–541, Feb. 2016.

[32] Z. Zhao, X. Meng, Y. Su, and Z. Li, "Virtual network embedding based on node connectivity awareness and path integration evaluation," *KSII Trans. Internet Inf.*, vol. 11, no. 7, pp. 3393–3412, Jul. 2017.

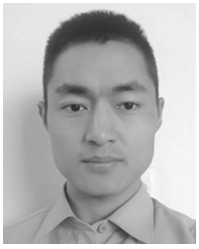
YUZE SU was born in Jinan, China, in 1990. He received the B.S. degree in communication engineering from the Changsha University of Science and Technology, Changsha, in 2013, and the M.S. degree in information and communication engineering from Air Force Engineering University, Xi'an, China, in 2016, where he is currently pursuing the Ph.D. degree in information and communication engineering. His research interests include network virtualization and survivable virtual network.

XIANGRU MENG was born in Lantian, China, in 1963. He received the B.S., M.S., and Ph.D. degrees from Xi'an Jiaotong University, China, in 1985, 1988, and 1994, respectively. He is currently a Professor with Air Force Engineering University, Xi'an, China.

From 1995 to 1997, he was a Visiting Scholar at the University of Electronic Science and Technology, Chengdu, China. His research interests include next generation Internet, network virtualization, and survivable network.



QIAOYAN KANG was born in Yongchun, China, in 1980. She received the B.S., M.S., and Ph.D. degrees from Air Force Engineering University, China, in 2001, 2004, and 2008, respectively. She is currently an Assistant Professor with Air Force Engineering University, Xi'an, China. Her research interests include network management and survivable network.



XIAOYANG HAN was born in Zhumadian, China, in 1986. He received the B.S. and M.S. degrees from Air Force Engineering University, China, in 2009 and 2017, respectively, where he is currently pursuing the Ph.D. degree. His research interests include survivable virtual network and software defined network.